Atelier de Sculpture Virtuelle Multi-utilisateurs

Patrice Torguet, Frédéric Rubio, René Caubet
Institut de Recherche en Informatique de Toulouse
118 Route de Narbonne
31062 Toulouse Cedex, France
{torguet | rubio}@irit.fr

RÉSUMÉ :
La réalité virtuelle distribuée, grâce aux nouveaux paradigmes d'interaction qu'elle propose, offre d'intéressantes perspectives pour la modélisation interactive d'objets et de scènes.
Nous envisageons de créer un atelier de sculpture virtuelle où les artistes pourront modeler des objets à partir d'une forme pré-existante ou d'un quelconque bloc de matière.
Cet article présente une première implémentation de cette application s'appuyant sur une plate-forme générique de développement d'environnements virtuels multi-utilisateurs : VIPER.

MOTS CLÉS : Réalité Virtuelle, Modélisation, Collecticiel, Interactivité, Modélisation Déclarative.

INTRODUCTION

Les systèmes de réalité virtuelle se limitaient à leur début à des visites de lieux imaginaires ou reconstitués.

Les objets constituants ces environnements n'étaient qu'une représentation symbolique d'objets plus complexes (une maison était représentée par un cube, …).

Lorsque les concepts de base de la réalité virtuelle ont été bien établis, les concepteurs ont pu se consacrer davantage aux problèmes d'interactivité de leur système (aspects jusqu'alors moins développés).

L'innovation créée par ces nouveaux systèmes ne compensait plus le manque de convivialité et d'ergonomie des premières réalisations. Aussi, on a pu constater une nette amélioration des interfaces proposées grâce aux progrès de la technologie (puissances des machines, nouveaux périphériques, …) et grâce à des études approfondies sur l'interactivité des systèmes de réalité virtuelle. Ce sont ces aspects qui nous intéressent plus particulièrement dans le cadre de cet article.

La réalité virtuelle distribuée vient ajouter à ces avantages l'aspect collecticiel (entre autres choses) qui est très intéressant pour certains types d'applications.

Les premières réalisations dans ce domaine furent militaires (se reporter à [6]).

D'autres types d'applications ont donné lieu à des réalisations : visite de lieux reconstitués [8], définition de plateformes pour des applications collecticielles [2].

Nous pensons qu'il serait intéressant d'étudier les apports en terme d'ergonomie et de convivialité de la réalité virtuelle distribuée à la modélisation interactive d'objets.

Nous envisageons donc de créer un atelier de sculpture où les personnes évoluant dans ce monde virtuel pourront collaborer à la création de formes.

PRÉSENTATION DU PROJET

La qualité des interfaces des systèmes de réalité virtuelle a rapidement progressé.

Des efforts importants ont été accomplis dans le domaine de la visualisation. Les images, sur l'écran ou dans le casque de visualisation, sont désormais plus réalistes (textures et matériaux, complexité géométrique des objets) [1].

L'interactivité s'est considérablement améliorée : la sensation d'immersion est complète ; la manipulation des objets est aussi plus réaliste (préhension d'objets, détection des collisions entre usagers et constituants de la scène) [1].

On peut enfin signaler que divers domaines d'application ont donné lieu à des études spécifiques (télé-opération [5], télé-virtualité [8], simulations militaires [6], simulations chirurgicales [3], …).

Néanmoins, certains axes de recherche ont été beaucoup moins explorés. C'est notamment le cas de la modélisation interactive d'objets.

Nous pouvons distinguer deux axes principaux dans l'évolution des modeleurs graphiques : les fonctionnalités offertes aux utilisateurs et les techniques d'interaction.

Ces études ont permis l'émergence de nouvelles techniques de modélisation.

Celles-ci, peu précises d'un point de vue "numérique", sont plus intuitives et certainement plus ergonomiques. Elles peuvent se regrouper en deux catégories :

• la déformation par manipulation directe.

L'opérateur transforme l'objet en modifiant directement sa structure géométrique par l'intermédiaire de fonctions mathématiques ou de forces appliquées sur l'objet.

• la déformation par manipulation indirecte.

Cette catégorie d'opérateurs est totalement indépendante des modèles géométriques. L'utilisateur n'agit plus directement sur la forme de l'objet. Il passe par le biais d'une entité de déformation qui l'englobe. Il s'agit pour l'utilisateur de déplacer les points de contrôle définissant le volume englobant (treillis).

C'est notamment dans cette catégorie que l'on retrouve les FFD [11] qui seront présentés plus en détail par la suite.

L'apport de ces nouvelles techniques de modélisation en matière de facilité d'utilisation et d'interactivité est indéniable. Néanmoins elles présentent certaines lacunes.

Les techniques d'interaction à base de manipulation directe sont trop dépendantes du modèle géométrique sous-jacent.

Le problème lié aux techniques de manipulation indirecte est le décalage évident entre les transformations réalisées par l'utilisateur sur l'entité de déformation et le résultat observé sur l'objet (différence prévisible car le treillis n'est qu'une approximation de l'objet à déformer).

Néanmoins ces nouvelles méthodes ont nettement amélioré la qualité et l'ergonomie des modeleurs.

Tous ces travaux traduisent un besoin d'interfaces nouvelles pour les modeleurs graphiques.

La modélisation déclarative [4] s'inscrit parfaitement dans cette démarche. L'idée générale est qu'il n'est plus nécessaire de communiquer d'informations numériques à la machine [10].

Dans le cas de la modélisation d'objets, l'utilisateur n'a plus à se préoccuper du type de modélisation pas plus que des moyens mis en œuvre pour exécuter ses commandes.

La réalité virtuelle, grâce aux nouveaux dispositifs d'entrée qu'elle propose, offre de nouvelles perspectives pour l'ergonomie de la modélisation, en introduisant de nouveaux concepts d'interaction.

Notre but est d'intégrer la modélisation déclarative dans un environnement de réalité virtuelle. Nous offrons à l'utilisateur la manière la plus naturelle qui soit pour exprimer sa commande : il la réalise lui-même en reproduisant des gestes qu'il effectuerait naturellement dans une situation réelle (i.e. marteler un objet, le forer, …).

Dans cette optique, nous envisageons de créer un atelier de sculpture virtuelle où l'artiste pourra modeler des objets à partir d'une forme pré-existante ou d'un quelconque bloc de matière.

Nous avons défini deux types d'usagers qui peuvent évoluer dans cet atelier virtuel. Chacun est doté d'une forme géométrique (main) et d'un point de vue. De plus on lui assigne un rôle spécifique [9].

La première catégorie d'utilisateur regroupe les "visiteurs". Un visiteur est un participant passif de l'atelier virtuel. Il peut se déplacer, visiter les différentes pièces, mais il ne peut pas intervenir dans une tâche de modélisation. Toute personne qui rejoint la session de simulation reçoit les droits d'un visiteur.

Le second type d'usager définit les "sculpteurs". Un sculpteur participe activement à la modélisation. Il déforme les objets par l'intermédiaire de l'outil qu'il détient (outils qui seront décrits par la suite). Les objets constituants la scène sont soit créés lors de la construction de l'environnement virtuel, soit créés par un utilisateur.

A tout instant, un visiteur peut devenir un sculpteur. Il lui suffit de "saisir" un outil.

Ces outils sont présentés sur des tables que l'on trouve dans chaque pièce de l'atelier.

Notre objectif est de rendre la tâche de modélisation plus intuitive. Ainsi nos outils ont un rôle double : leur forme illustre non seulement leur fonction (marteau, pince, …) mais aussi la façon de les utiliser (taper avec le marteau, …). La métaphore du sculpteur est pleinement réalisée : il y a correspondance directe entre le geste effectué par l'utilisateur dans le monde réel et l'action réalisée par son clone dans le monde virtuel.

L'aspect collecticiel de cette application induit des risques de conflit quand deux sculpteurs désirent travailler sur le même objet. Aussi, nous avons mis en place trois systèmes de verrous sur les objets.

Le premier niveau est le moins contraignant : la seule restriction est que plusieurs utilisateurs n'ont pas le droit de déformer simultanément la même partie d'un objet.

Si un objet est défini par une hiérarchie de formes géométriques, le second niveau permet à des utilisateurs de travailler sur des éléments distincts (pied et plateau d'une table).

Enfin le dernier niveau est le plus strict : un sculpteur peut se réserver l'exclusivité de l'objet.

Pour verrouiller un objet, l'utilisateur doit se munir d'un verrou qu'il trouve sur la table des outils et qu'il place sur l'objet concerné. La pose d'un verrou est signifiée aux autres utilisateurs par l'apparition d'un cube transparent qui englobe l'objet.

Pour mettre en oeuvre cet atelier de sculpture virtuelle, nous utilisons une plate-forme de développement d'applications de réalité virtuelle : VIPER.

L'ARCHITECTURE LOGICIELLE DE VIPER

VIPER (VIrtuality Programming EnviRonment) est une plate-forme générique permettant la gestion d'environnements virtuels et, plus généralement, le développement d'applications de Réalité Virtuelle. Ce système gère en temps réel plusieurs utilisateurs munis de périphériques spécialisés grâce à une architecture distribuée. Nous allons présenter ici la structure générale d'un environnement virtuel telle qu'elle est définie dans VIPER (pour plus de détail sur la plate-forme se référer à [13]).

Un environnement virtuel est composé d'entités, de stimuli et d'un univers virtuel (Fig. 1).

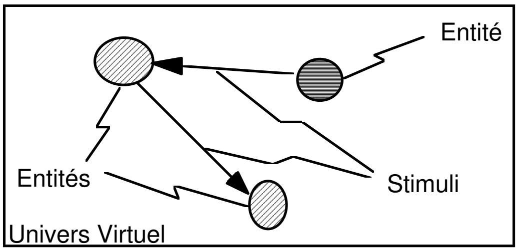

Figure 1 : Structure d'un environnement virtuel.

• La notion d'*entité* permet de gérer de façon uniforme les décors des mondes virtuels, les objets virtuels mais aussi des "clones" (objet associé à un utilisateur [7], à une application [14] [2] ou à un robot, qui permet à celui-ci de participer à la simulation comme si il était réellement présent dans l'univers virtuel). Les entités sont autonomes et possèdent un ensemble de propriétés et de comportements. Elles sont conceptuellement regroupées en familles (population des entités ayants les mêmes propriétés et les mêmes comportements). Globalement les systèmes simulés sont homogènes (beaucoup d'objets regroupés dans peu de familles).

• Les *stimuli* sont les véhicules des interactions entre les entités.

• L'*univers virtuel* représente le milieu tridimensionnel dans lequel les entités coexistent et interagissent.

Dans cette partie nous allons décrire la structure des entités présentes dans l'environnement virtuel en exposant le mécanisme de communication inter-entité et les mécanismes associés à la définition et à la prise en compte du comportement de ces entités (Fig. 2).

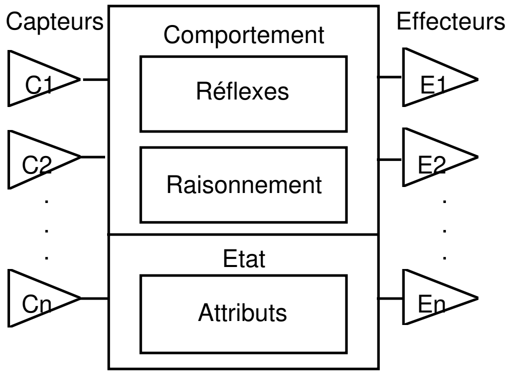

Figure 2 : Schéma structurel d'une entité.

Les entités et le milieu, dans lequel elles évoluent, interagissent de deux façons :

- depuis le milieu vers les entités : un changement du milieu constitue un stimulus, tel un signal visuel ou sonore, perçu par les entités. Un capteur spécifique reçoit ce stimulus, l'interprète puis transmet une information à la partie comportementale de l'entité.
- depuis chaque entité vers le milieu : après analyse de l'information, la partie comportementale modifie l'état interne de l'entité et, si nécessaire, commande une action aux effecteurs. Ceux-ci agissent sur le milieu et en le modifiant créent de nouveaux stimuli.

Nous proposons de modéliser chaque type d'interaction avec quatre classes d'éléments : des capteurs, des effecteurs, des images et des espaces d'images (Fig. 3).

Une image est l'aspect perceptible d'une entité dans son milieu (forme graphique, icône sonore...).
Un espace d'images est l'ensemble des images des entités liées à un certain type de stimuli. On peut le définir comme une projection de l'univers virtuel selon un axe propre à un stimuli (espaces des formes graphiques, espaces des sons...).
Un capteur saisit une vue instantanée de l'espace d'image qu'il filtre puis interprète.
Un effecteur produit, par l'exécution d'une action, une image qu'il place dans l'espace d'images.

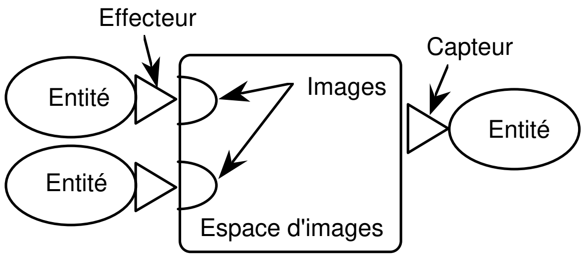

Figure 3 : Mécanisme de communication inter-entités.

Pour la définition de comportements, nous proposons la partition suivante :

- une composante "réflexe" où sont définis les réflexes de l'entité aux stimuli provenant de son environnement (via les capteurs). On peut citer en exemple de réflexe, la réaction d'une entité à une collision avec une autre entité.
- une composante "raisonnement" où est définie la partie nécessitant une analyse plus complexe et un caractère plus volontaire. La poursuite d'un but à atteindre par une entité illustre bien la notion de raisonnement.

L'introduction de cette partition permet de dissocier la partie active du comportement, encapsulée dans la composante "raisonnement", de la partie réactive, définie au niveau de la composante "réflexe".

VIPER est écrit en langage C++ et utilise les concepts de la programmation par objets de façon à supporter des applications spécifiques dans divers domaines. Ceci est fait grâce aux mécanismes d'héritage et d'instanciation générique qui permettent de spécialiser des classes d'entités, de capteurs, d'effecteurs, d'images ou d'espaces d'images pour une utilisation plus spécifique.

Nous allons maintenant décrire l'outil mathématique réalisant les déformations d'objets.

LES DEFORMATIONS DE FORME LIBRE VIRTUELLES

C'est à Sederberg [11] que nous devons la définition des Free-Form Deformation (FFD). Déformer par FFD consiste à enfermer l'objet (ou une partie de celui-ci) dans un bloc de plastique souple. Dès lors, les déformations ne sont plus appliquées directement à l'objet mais à l'espace qui l'englobe. Les transformations subies par l'objet ne sont qu'une interpolation de la déformation du volume englobant.

Les FFD ont fait l'objet de nombreux articles. Aussi, nous ne rappellerons que brièvement leurs fondements mathématiques.

Le volume englobant un objet est un parallélépipède qui définit une partition de l'espace en deux sous-ensembles : les points intérieurs et les points extérieurs. Ces derniers ne seront pas affectés par le processus de déformation qui se décompose en plusieurs étapes.

Il s'agit tout d'abord de définir le volume englobant en indiquant son origine, ses axes directeurs et le nombre de points de contrôle selon chacun de ces axes. On crée ainsi un treillis de points dont les coordonnées sont exprimées dans le repère propre au treillis (Fig. 4).

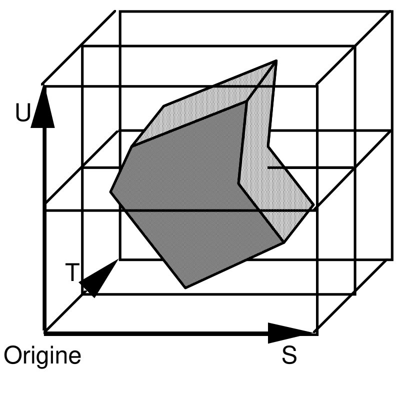

Figure 4 : Un exemple de treillis.

Dans une seconde phase, il faut calculer les nouvelles coordonnées des points de l'objet dans le repère local.

A partir de ce moment, toute déformation du treillis (déplacement d'un ou de plusieurs points de contrôle) peut être répercutée à l'objet par une interpolation de type Bézier ou B-Spline. L'exemple suivant illustre une interpolation à base de polynômes de Bernstein :

$$P_{ffd} = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \left[\sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \left[\sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k P'_{ijk} \right] \right]$$

où (s, t, u) sont les coordonnées locales du point traité,

et P'_{ijk} sont les points de contrôle après déplacement.

Cet outil illustre parfaitement la métaphore du sculpteur : du point de vue de l'utilisateur, il suffit de modeler le bloc de plastique pour déformer l'objet, ce qui est très intuitif. Dans la réalité, on s'aperçoit qu'il n'est pas toujours facile d'obtenir certains effets (comme aplatir une bosse).

Nous avons donc été amenés à concevoir un système qui masquerait totalement la notion de fonctions mathématiques pour la modélisation d'objets. Nous proposons à l'utilisateur un ensemble d'outils ne nécessitant que des paramètres ayant une signification réelle (partie de l'objet qui sera modifiée, taille de la tête d'un marteau, …) et communiqués de la manière la plus naturelle (désignation graphique, choix du marteau, …).

Dans cette optique, la réalité virtuelle nous fournit la possibilité de mettre en œuvre ces concepts, notamment avec les outils auxquels sont associés un geste qu'il est facile d'analyser : une main qui tient un marteau et qui "tape" avec force une surface est un événement facilement interprétable (du moins dans notre application).

A cet effet, nous étudions un ensemble d'outils basés sur les FFD tels que poinçon, pince, …

Nous allons maintenant présenter un des outils qui ont été réalisé : le marteau.

Un "coup de marteau" est traité de la façon suivante : dès que la tête du marteau (en fait un point de la tête) pénètre dans un objet, le treillis est orienté selon la trajectoire de l'outil (deux dernières positions) et translaté au point d'impact.

Les coordonnées des points de l'objet sont calculées dans le repère associé au treillis.

Ce dernier est modifié (avec prise en compte de la rigidité) puis l'objet est déformé. La profondeur est calculée en fonction de la rigidité de l'objet. Précisons ici que cette rigidité n'a rien de physique mais elle permet de varier les effets. Ainsi pour un objet donné et un type de marteau, nous créons un treillis (Fig. 5).

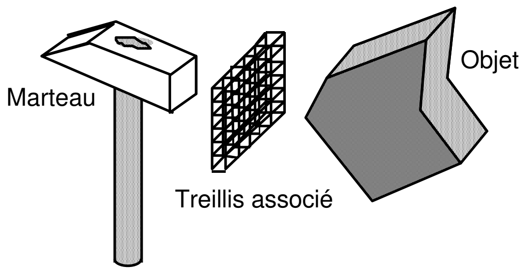

Figure 5 : Un type d'outils

LA NOTION D'OUTIL DANS VIPER

Un outil dans le modèle VIPER est un type particulier d'entité qui est manipulé par une ou plusieurs entités et qui, à son tour, manipule une ou plusieurs entités (Fig. 6).

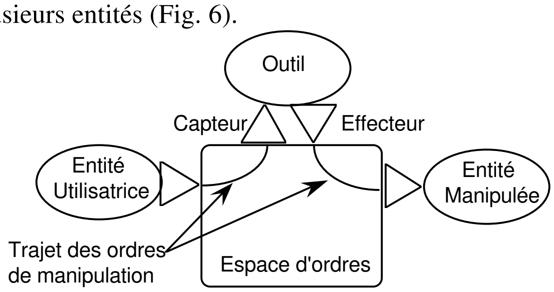

Figure 6 : Illustration de la manipulation via un outil.

Un outil possède au moins un capteur, appelé capteur d'ordre, qui récupère les ordres qui lui sont transmis ; ainsi qu'un effecteur d'ordre qui envoie une commande de manipulation aux entités contrôlées par l'outil.

Le comportement d'une telle entité est typiquement un comportement réflexe déclenché par la récupération d'un ordre donné à l'outil et qui, en fonction du résultat de l'analyse, va produire, si nécessaire, un ordre de manipulation.

Chaque entité utilisatrice d'un outil doit posséder un effecteur d'ordre. Symétriquement, chaque entité manipulée dispose d'un capteur d'ordre et d'un réflexe associé.

Un ordre est un type d'image spécifique qui est composé de :
- du nombre d'entités manipulées,
- des identificateurs de ces entités,
- du type de manipulation,
- des paramètres de cette manipulation.

Nous allons illustrer ces principes par la mise en œuvre d'un "coup de marteau" (Fig. 7).

Dans cet exemple, l'entité "clone" de l'utilisateur contrôle l'entité "marteau" en lui envoyant des ordres de manipulation. Un tel ordre est composé de l'identificateur de l'outil, du label "coup de marteau" et comme paramètres les deux dernières positions de la main.

L'entité marteau analyse ces données pour déterminer si la tête du marteau heurte effectivement un objet. Si une collision est détectée, l'entité marteau envoie un ordre de manipulation à l'objet touché. Cet ordre comprend l'identificateur de l'objet, le type de l'interaction ("martèlement"), les deux dernières positions de la tête du marteau ainsi que son diamètre.

Dès la réception de cet ordre, l'entité objet crée un treillis, le modifie conformément aux paramètres (type de déformation, positions, diamètre) et enfin elle calcule sa nouvelle forme.

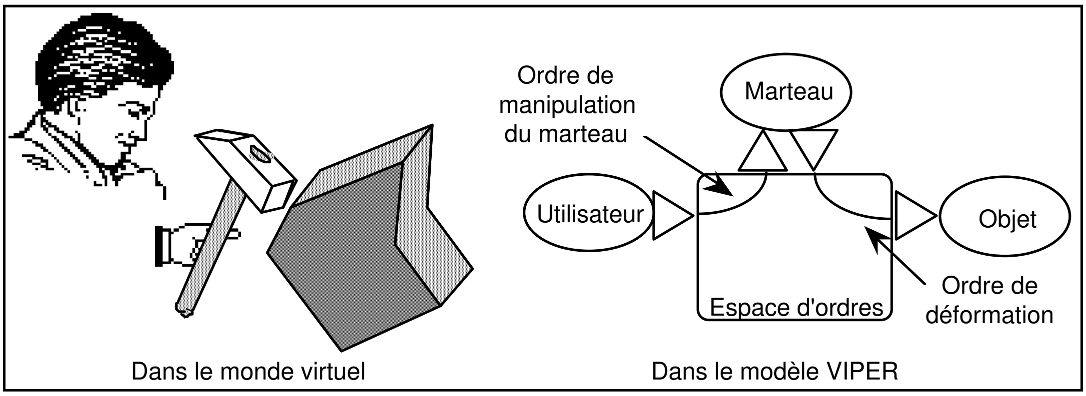

Figure 7 : Le "coup de marteau".

ASPECTS MULTI-UTILISATEURS

L'introduction d'un espace d'images, qui joue un rôle de médiateur dans la communication, permet de simplifier la gestion des interactions dans un cadre distribué (collecticiel). L'espace d'images devient alors un agrégat distribué d'images qui gère les interactions de façon autonome. De plus, il traite uniformément les interactions locales à un site et les interactions impliquants des entités qui sont gérées par des sites distants.

Nous allons montrer, maintenant, comment certains types d'espaces d'images distribués peuvent résoudre les problèmes soulevés par l'aspect collecticiel de notre application.

Le premier problème apparaît dès que plusieurs visiteurs évoluent ensemble dans l'atelier. Un visiteur doit avoir conscience de la présence des autres utilisateurs. Pour résoudre ce problème on définit un type particulier d'espace d'image distribué, appelé espace d'image dupliqué. C'est un agrégat qui lors de la modification d'une de ses images maintient la cohérence entre l'image originale et des copies de celle-ci qui figurent sur chaque site (Fig. 8).

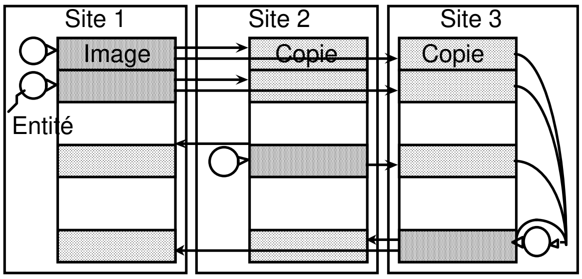

Figure 8 : Un espace d'images dupliqué.

La perception visuelle illustre bien ce principe. Chaque objet et utilisateur est pourvu d'une "image-forme" contenant : sa position, son orientation et une référence à sa forme géométrique (ensemble de facettes triangulaires représentant l'enveloppe de l'objet).

Un espace d'images dupliqué contient toutes les image-formes, ainsi lors de la modification de l'une d'elles (déplacement, rotation ou déformation de l'objet), l'espace d'images maintient la cohérence entre les différentes copies. Par la suite, chaque entité disposant de capteurs en relation avec l'espace d'image a accès à la nouvelle version de l'image.

L'existence d'utilisateurs de type sculpteur crée un nouveau problème : l'entité effectrice agissant sur une entité peut être située sur un autre site.

Comme nous l'avons vu dans la section précédente, l'entité manipulatrice envoie un ordre à l'entité manipulée.

Dans un contexte distribué, un espace d'images dupliqué pourrait résoudre le problème de sites différents. On peut néanmoins remarquer qu'un ordre donné ne concerne que l'entité manipulatrice et les entités manipulées. Il n'est donc pas nécessaire de le communiquer à tous les sites prenant part à la simulation.

L'adjonction du numéro de site de l'entité dans son identificateur permet de limiter la transmission d'un ordre aux sites intéressés (Fig. 9).

Les aspects multi-utilisateurs entraînent d'autres problèmes liés aux conflits risquant de survenir lorsque deux utilisateurs tentent de réaliser des manipulations antagonistes.

Ces antagonismes peuvent intervenir à deux niveaux correspondant à la granularité de l'information manipulée.

Si l'unité de manipulation est l'objet, deux sculpteurs devront obligatoirement travailler sur des objets différents. Dans ce cas, nous proposons la pose explicite d'un verrou graphique sur l'objet. Ce verrou devient alors un outil qui n'est manipulé que par un utilisateur à la fois. L'utilisateur va le saisir, l'amener sur un objet et le lâcher (*drag and drop* des interfaces 2D). L'objet sera désormais verrouillé (Fig. 10) jusqu'à ce que ce même utilisateur retire le verrou.

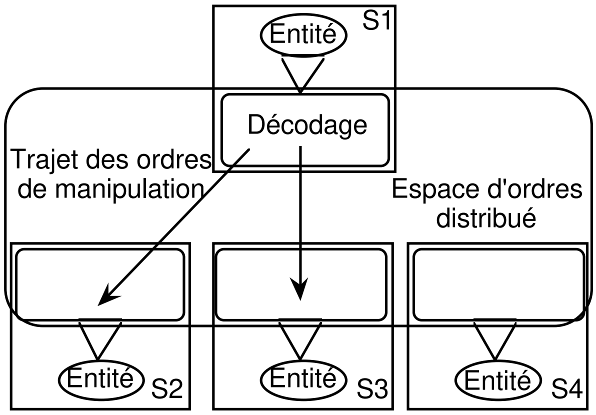

Figure 9 : L'espace d'ordres distribué.

Si l'unité de manipulation est une sous partie de l'objet (objet défini par une hiérarchie de formes géométriques) nous proposons la pose d'un verrou sur cette sous-partie.

Si aucun verrou n'est placé, on autorise deux manipulations sur un même objet en vérifiant qu'il n'y ait pas de recouvrement spatio-temporel des différents treillis mis en jeu.

Ce concept est intéressant car il nous permettra à terme de faire évoluer notre application vers la collaboration de plusieurs sculpteurs à une même déformation (torsion de barres, aplatissement, étirements...).

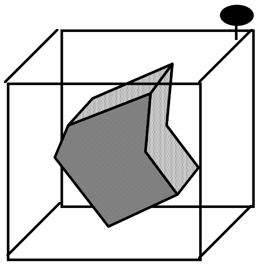

Figure 10 : Un verrou posé sur un objet

CONCLUSION ET PERSPECTIVES

Nous allons désormais nous consacrer à améliorer la robustesse de notre application. Il subsiste encore quelques problèmes à deux niveaux : limite intrinsèque des outils virtuels et gestion du travail coopératif.

La plupart des outils de modélisation sont opérationnels mais nous n'avons pas encore obtenu la précision souhaitée (Marteler exactement l'endroit désiré n'est pas chose facile). Nous étudions à cet effet des outils complémentaires de visualisation (tel qu'une loupe) et de désignation pour lesquels de nouvelles métaphores devront être définies.

La notion de rôle telle que nous l'avons définie n'est peut être pas assez contraignante (un visiteur peut devenir à volonté un sculpteur). Notre souci majeur, lorsque nous avons établi les bases de cet atelier de sculpture, était la liberté d'action des participants. C'est pourquoi il n'existe pas de superviseur dans cet environnement virtuel.

Nous envisageons donc de créer un système de "badge" ou de "casquette". Ils seront utilisés pour définir les rôles. L'atelier virtuel sera divisé en zones d'influence (atelier de travail, pièce, …). Chacune possédera un nombre précis de sculpteurs (aucun pour les galeries d'exposition, un, deux, …), et éventuellement d'un certain nombre de superviseurs (arbitre chargé de régler des conflits). Les badges seront disponibles sur des présentoirs répartis dans l'espace d'influence. Après utilisation, ils seront reposés sur le présentoir le plus proche. Le rôle d'un participant pourrait être symbolisé par un code couleur.

Il peut néanmoins survenir des situations conflictuelles. Dans ce cas, et en l'absence d'arbitre, comment communiquer à un sculpteur concurrent ses propres intentions ?

Bien que la spécificité graphique tridimensionnelle de notre application ne facilite pas cette tâche, un début de solution est certainement la mise à disponibilité d'une communication orale entre sculpteurs basée sur la notion d'aura [12]. Les utilisateurs seront alors munis d'une aura représentée par une sphère de communication. Ils pourront ainsi dialoguer si leurs sphères entrent en contact.

Enfin, la représentation géométrique actuelle des usagers (main seule) étant trop succincte pour permettre un suivi détaillé de l'évolution du travail de chacun, il faudra l'améliorer.

Ces travaux ont essentiellement permis une première validation de la plate-forme de conception d'applications de réalité virtuelle distribuées et de mettre en oeuvre de nouveaux outils de modélisation interactive.

De plus les principaux concepts de base peuvent être repris pour d'autres types d'applications qui nécessitent des manipulations collectives d'objets virtuels : sculpture de fresques de grande dimension (en utilisant au mieux les spécialités des différents sculpteurs : visages humains, animaux...), applications militaires (déformation de terrain dues à des impacts d'obus ou à des explosions), décoration d'intérieurs, simulations chirurgicale ...

BIBLIOGRAPHIE

1 Astheimer, P., Felger, W., Müller, S., Virtual Design: a Generic VR System for Industrial Applications. *Computer & Graphics, Vol. 17, N° 6,* November/December 1993.

2 Carlsson, C. and Hagsand, O. DIVE - a Platform for Multi-User Virtual Environments. *Computer & Graphics, Vol. 17, N° 6,* November/December 1993.

3 Delingette H., Subsol, G., Cotin, S., Pignon J. A Craniofacial Surgery Simulation Testbed. *Visualization for Biomedical Computing (VBC'94)* Rochester, USA October 1994.

4 Gaildrat, V., Caubet, R., Rubio, F., Declarative Scene Modelling with Dynamic Links and Decision Rules Distributed Among the Objects. *IFIP International Conference on Computer Graphics ICCG93* Bombay, Février 1993.

5 Hine, B. et al., The Application of Telepresence and Virtual Reality to subsea explorations. *The 2nd WorkShop on Mobile Robots for Subsea Environments*, Proceedings ROV'94, May 1994, Monterey CA.

6 Macedonia, M.R., Zyda, M.J., Pratt, D.R., Barham, P.T., Zeswitz, S. NPSNET: A Network Software Architecture For Large Scale Virtual Environments. *Presence Vol. 3, N° 4*, Fall 1994.

7 Mouli, R., Duthen, Y., Caubet, R., In VitrAm (In Vitro Animats, a behavioural simulation model). *2nd IEEE International Workshop RO-MAN'93*, Tokyo, 3-5 Novembre 1993.

8 Quéau, P., Televirtuality: the merging of telecommunications and virtual reality. *Computer & Graphics, Vol. 17, N° 6,* November/December 1993.

9 Rapport de l'atelier "collecticiel". *Comptes rendus des ateliers d'IHM'93*, Lyon, Octobre 1993.

10 Rubio, F., Gaildrat, V., Vigouroux, N., Caubet, R., Les commandes progressives dans une interface multimodale. *Actes d'IHM'94*, Lille, Décembre 1994.

11 Sederberg, T. W., Parry, S.R., Free-Form Deformation of Solid Geometric Models. *Actes de Siggraph 1986*.

12 Ståhl, O., Implementation issues of aura based tools. *Proceedings of the 5th MultiG WorkShop*, Stockholm, December 1992.

13 Torguet, P., Caubet, R., VIPER (VIrtuality Programming EnviRonment): A virtual reality applications design platform. *Actes du 2nd Eurographics Workshop on Virtual Environments*, Janvier 1995.

14 West, A. J., Howard, T. L. J., Hubbold, R. J., Murta, A. D., Snowdon, D. N., Butler, D. A., AVIARY - A Generic Virtual Reality Interface for Real Applications. *Virtual Reality Systems, sponsored by the British Computer Society*, Mai 1992.