Master research Internship

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Bibliographic report

Sailor vs Poseidon : Enhancing Training Simulation in 3D Collaborative Virtual Environments

---

Domain : Technology for Human Learning, Human-Computer Interaction

Author :
Gwendal Le Moulec

Supervisors :
Valérie Gouranton
Thi Thuong Huyen Nguyen
Fernando Argelaguet
Anatole Lécuyer
Hybrid

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Résumé

Here should appear your abstract - Should be 15 lines long

Table des matières

1 Introduction

L'apprentissage humain est un des principaux champs d'application de la Réalité Virtuelle (RV). L'utilisation d'outils virtuels permet en effet de s'entraîner dans des conditions favorables et paramétrables avant de se confronter à des situations réelles, qui peuvent être trop risquées ou trop complexes pour un apprenti. Les opérations de maintenance de certaines pièces mécaniques peuvent par exemple s'avérer dangereuses pour une personne inexpérimentée, qui risque alors de se blesser gravement. Dans le cas d'opérations chirurgicales, les erreurs sont irréversibles, sauf sur des corps virtuels. L'apprentissage virtuel constitue donc une coopération entre un apprenti et un entraîneur réel ou virtuel. Le but de l'apprenti est de réaliser un certain nombre de tâches définies par l'entraîneur. Ce dernier évalue les performances de l'apprenti en tenant en général compte du temps de réalisation. Il peut paramétrer l'apprentissage de façon à rendre la réalisation plus ou moins difficile, selon les objectifs de l'apprentissage et le niveau de l'apprenti.

Plusieurs apprentis peuvent coopérer dans le même environnement virtuel. De tels environnements sont appelés Environnements Virtuels Collaboratifs (EVC). Qu'ils soient destinés à l'apprentissage ou non, les EVC doivent permettre la collaboration d'utilisateurs physiquement présents au même endroit (e.g grâce à un dispositif immersif de type CAVE [1]), ou bien répartis sur des sites distants. Certaines applications d'apprentissage impliquent inévitablement un travail en équipe. Les tâches de co-manipulation d'objets sont les plus classiques, telles que le déplacement d'objets lourds, ou les tâches de maintenance complexes.

2 Prise de conscience

En EV collaboratif ou non, il est essentiel de prendre conscience des événements extérieurs, qu'ils soient directement liés aux tâches de l'utilisateur ou pas. La prise de conscience se fait principalement à l'aide de métaphores. On distingue notamment deux types de prises de conscience : la prise de conscience d'événement ou de faits généraux qui ne sont pas directement liés à une tâche à réaliser en particulier et la prise de conscience destinée à faciliter un tâche donnée.

2.1 Prise de conscience d'événements

La prise de conscience des événements se déroulant dans un EV à pour objectif de ne pas perdre les utilisateurs, qui doivent comprendre ce qu'il se passe autour d'eux. Quelques fois, cela peut même avoir une incidence sur les tâches à réaliser. Par exemple, les échanges d'avatars de l'article [1] sont importants à prendre en compte pour l'utilisateur car il faut successivement demander de l'aide à des experts précis qui peuvent changer d'avatars plusieurs fois au cours de la simulation. Cependant, ce genre d'événements pouvant se produire n'importe quand et sans raison particulière, il ne peut pas être classé dans les facilitations de tâches (voir section 2.2) car il n'est pas destiné à guider l'utilisateur.

Les faits et événements peuvent être regroupés en différentes catégories (voir [2]) :

1. Dispositif sous forme de salle à 3, 4, 5 ou 6 murs donnant l'impression d'être plongé dans l'environnement virtuel 3D.

FIGURE 1 – Barrière de dissuasion pour éviter une collision avec le mur réel [2].

— **les utilisateurs :** il faut pouvoir retranscrire leurs activités. Pour cela, les supports des métaphores sont leurs avatars. Un cas d'usage serait le fait de mettre en évidence un avatar qui a un rôle particulier dans l'environnement, par le moyen d'une icône placée au-dessus de lui par exemple.
— **l'environnement virtuel :** des objets ou des zones de la scène peuvent se caractériser par leur état. Dans certaines applications, certains objets sont manipulables et d'autres non. Il peut alors être intéressant de coder cette information par un code couleur.
— **les interactions :** lorsqu'un utilisateur interagit avec un objet, il peut être utile de représenter cette information par un lien entre l'utilisateur et l'objet.
— **l'environnement physique :** il ne faut pas oublier que les utilisateurs évoluent avant tout dans un environnement réel constitué d'obstacles, par exemple les murs dans un système CAVE ou les autres utilisateurs. A défaut de pouvoir éviter les collisions, Il faut au moins diminuer les risques. Une possibilité est d'afficher un message de prévention lorsqu'un utilisateur est trop prêt d'un mur, ou une barrière virtuelle destinée à dissuader l'utilisateur, comme présenté sur la figure 1.
— **Les erreurs internes :** certaines erreurs du système peuvent introduire des incohérences dans l'EV. C'est notamment le cas lors de problèmes de réseau pour un EVC : si le système chargé de gérer un objet ou un avatar particulier ne communique plus de données, les interactions qu'auraient alors les utilisateurs avec cet avatar ou cet objet seraient incohérentes. Il peut y avoir aussi un déphasage entre ce que voient deux utilisateurs différents. Dans ces deux cas, il est important de faire prendre conscience de l'anomalie, en gelant l'objet par exemple.

Quelque soit le type de métaphore à mettre en œuvre, il faut toujours faire un compromis entre la qualité de la métaphore et son coût. La qualité d'une métaphore se mesure par sa capacité à transmettre une information correcte, compréhensible et complète, ainsi que par son influence sur le sentiment d'immersion éprouvé par les utilisateurs. Le coût inclut bien sûr la consommation de ressources, mais se caractérise surtout par la distraction inutile que peut introduire une métaphore, ou l'encombrement de la scène. Par exemple, dans le cas d'un objet vu sous différents états selon les utilisateurs à cause de problèmes liés au réseau, une solution consiste à faire apparaître à côté de l'objet un fantôme par état différent [2]. Cependant, faire apparaître trop d'états peut encombrer l'espace, ce qui n'est pas souhaitable.

Un exemple qui montre bien les différents niveaux de métaphores privilégiant soit certains

critères de qualité, soit un faible coût est celui de l'échange d'avatars de [1]. Une des expériences présentées consiste à tester différentes métaphores permettant de prendre conscience d'un échange d'avatars en tant que témoin extérieur à l'échange. Le témoin doit réaliser une séquence d'opérations de maintenance sur une voiture. Certaines opérations requièrent l'expérience d'un expert incarnant un avatar. Il y a plusieurs experts à appeler au cours de la simulation et ces derniers échangent régulièrement leurs avatars. Pour faciliter la compréhension, chaque expert est associé à une couleur. Trois métaphores différentes sont testées :

1. *Flickering* : chaque avatar impliqué dans l'échange clignote en prenant la couleur de l'expert qui le contrôle, puis en prenant la couleur du nouvel expert.
2. *Ghosts* : un fantôme de la couleur de l'expert apparaît à côté de l'avatar d'origine et se déplace jusqu'à l'avatar de destination.
3. *Popup* : une simple popup apparaît et indique quel échange a lieu.

La vidéo `http://youtu.be/Yu4YpGzKXes` montre un exemple d'échange pour chaque métaphore. Les critères de qualité et de coût ont étés évalués par le moyen d'un questionnaire, sur une population de test composée de 54 participants divisé en trois groupes (un pour chaque métaphore). Chaque critère est évalué par une ou plusieurs affirmations (e.g "La métaphore est facile à comprendre" ou "La métaphore distrait de la tâche") notées par un degré de satisfaction allant de 1 à 5. les résultats révèlent entre autres que même si *Ghosts* est jugée de bien meilleure qualité que *Popup* sur tous les critères, *Popup* est la métaphore qui donne le meilleur temps d'exécution des tâches. *Ghosts* est aussi utilisée dans une autre expérience de l'article, qui compare des métaphores représentant un échange d'avatars dans lequel l'utilisateur est impliqué. Elle est alors considérée par une partie des utilisateurs comme étant trop distrayante, ce qui rentre dans les critères de coût.

Les compromis à faire dépendent beaucoup du type d'événement ou de fait à représenter. Pour apporter des informations sur un objet de l'EV, une bonne solution consiste à jouer sur sa couleur ou sa transparence, car cela n'encombre pas l'EV et permet de bien identifier quel objet est concerné. La difficulté et alors de choisir une couleur ou un degré de transparence qui n'altère pas la perception de l'environnement. Par exemple, il n'est pas souhaitable de faire disparaître complètement un mur occultant une zone d'intérêt, car cela risque de faire oublier la présence du mur. Il vaut mieux en rendre un pan moyennement transparent. Dans le cas de la remontée d'erreurs internes ou d'événements de faible importance, il semble à priori plus souhaitable d'avoir recours à des popups discrètes plutôt que d'encombrer l'EV qui sert de support direct aux informations plus importantes.

2.2 Facilitation de tâches

Lorsqu'un utilisateur doit effectuer une tâche précise, par exemple atteindre une zone cible, il peut être guidé au moyen de métaphores. Ces indications ont pour but d'améliorer la performance d'un utilisateur ou de le diriger plus rapidement vers les points d'intérêt plutôt que de le laisser perdre du temps en cherchant par lui-même.

Certains guidages ont un but purement didactique, c'est à dire qu'ils s'utilisent durant une phase d'entraînement pour apprendre à l'utilisateur comment effectuer une tâche. Un bon exemple est celui qui est cité dans l'article [3], où un entraîneur montre à un apprenti des pièces de moteur

de voiture en les mettant en évidence [2]. L'apprenti subit ensuite une phase d'évaluation au cours de laquelle il tente de retrouver les pièces successivement montrées sans avoir recours à la mise en évidence. D'autres guidages sont inhérents à la tâche à effectuer et le fait de les enlever perdrait l'utilisateur, même très expérimenté. Un exemple est celui qui est reporté dans l'article [4], qui montre une technique de manipulation d'objet avec 6 Degrés De Liberté (DDL) à base de 3 points de contact. Ces points sont représentés par des boules rouges. Leur absence empêcherait d'avoir un repère spatial pour maîtriser correctement la manipulation.

Les métaphores de guidage indiquent en général soit directement un objet ou une zone cible (par mise en évidence), soit la direction vers la cible. Dans tous les cas, Le critère recherché permettant d'évaluer la métaphore est son efficacité, c'est à dire sa capacité à rendre la tâche plus facile. Les paramètres que l'on recherche à minimiser sont le temps d'exécution de la tâche et le niveau de stress ou de fatigue que la métaphore induit. Ces critères sont différents de ceux qui permettent d'évaluer une métaphore représentant un événement ou un fait non directement corrélé à la tâche : ce que l'on cherchait à minimiser, c'était la distraction ou l'encombrement qu'une métaphore induisait, ce qui pouvait effectivement gêner l'utilisateur dans le cadre de l'exécution d'une tâche "plus importante" que ce qui se passe autour. Dans le cas d'une métaphore de guidage, la distraction est a priori petite par rapport à la concentration induite. Il est plus important de vérifier que cette concentration n'est pas accompagnée d'effets de bord. Par exemple, [3] teste l'influence de l'absence de métaphore pour mettre en évidence les objets, ce qui oblige l'apprenti à se déplacer jusqu'à l'entraîneur pour regarder par-dessus son épaule. Ceci introduit une gêne due à la proximité (voir la section 4), ce qui n'est pas souhaitable.

Dans le cas de métaphores de guidage introduites par un entraîneur, la perception que celui-ci a de son apprenti et du contexte dans lequel celui-ci se trouve est également à prendre en compte. En effet, l'entraîneur peut vouloir adapter dynamiquement la métaphore pour rendre la tâche plus aisée ou au contraire plus dure, selon les performances de l'apprenti. Il est également possible d'envoyer des métaphores visuelles permettant de donner à l'apprenti des indications sur ses performances. Il faut alors représenter les informations dont à besoin l'entraîneur. Un cas d'usage est donné dans [5] (section 2) : un apprenti est immergé dans un EVC où il doit trouver des cibles. L'entraîneur est quant à lui en face d'une carte interactive de l'EVC sur laquelle il peut observer la cible, l'utilisateur, son champ de vision et les métaphores de guidage testées unes à unes : un ensemble de flèches traçant le chemin, une boussole accompagnant l'apprenti et une source de lumière provenant de la cible. L'entraîneur peut zoomer sur la carte et même emprunter le point de vue de l'apprenti. Il peut lui envoyer des signaux de couleur pour lui indiquer s'il se dirige dans la bonne ou la mauvaise direction. Il peut aussi adapter les paramètres de la métaphore de guidage, par exemple en ajoutant des flèches pour la première métaphore testée. La figure 2 montre un exemple de vue de l'apprenti (à gauche) et de l'entraîneur (à droite). Il n'y a pas eu de comparaison de différentes métaphores dans ce cadre.

2. La technique est alors un peu particulière car elle consiste simplement à rendre invisibles ou transparents les obstacles qui occultent la pièce du champ de vision de l'apprenti.

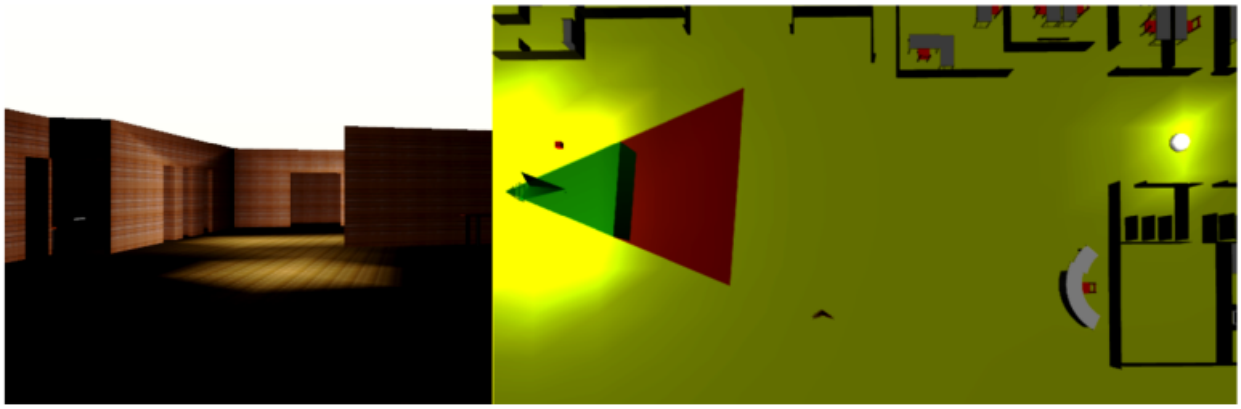

FIGURE 2 – Éclairage d'un chemin et assombrissement des alentours [5]. Vue de l'apprenti (à gauche) et de l'entraîneur (à droite).

3 Interagir avec l'environnement

La majorité des applications de RV introduisent la possibilité d'interagir avec les objets de la scène, ce qui sert à l'accomplissement de tâches. L'interaction la plus classique consiste à manipuler des objets, par exemple pour les transporter ou changer leur position.

3.1 Manipulation d'objets

La manipulation d'objets en coopération est une opération qui met en jeu des interactions complexes entre les utilisateurs et l'objet manipulé. D'une part, il faut que tous les utilisateurs aient une bonne perception de l'objet et des points de contacts impliqués pour leur interaction. D'autre part, chaque utilisateur doit avoir une bonne perception de l'influence des autres utilisateurs sur l'objet manipulé. Dans la vraie vie, ces informations sont transmises par le retour des forces, par exemple dans le cas du déplacement d'un meuble lourd, les utilisateurs ressentent les forces exercées par chaque acteur. En EVC, le retour de force n'est possible qu'avec certains appareils. Lorsque cela est nécessaire, ces informations sont retranscrites sous forme de métaphores visuelles particulières représentant les interactions et leurs paramètres (e.g intensité et direction de la force). Comme la manipulation en coopération est un problème d'EVC, il faut gérer la communication des informations entre chaque utilisateur, ainsi que la synchronisation et la cohérence vis-à-vis de l'état de l'environnement vu par chaque utilisateur, qui opèrent potentiellement sur des terminaux différents.

3.1.1 Exploitation des 6DDL

L'un des problèmes les plus étudiés en manipulation d'objet est l'exploitation des 6DDL (transitions et rotations selon les trois axes du repère 3D de l'objet). En effet, la saisie d'objets se fait la plupart du temps avec des appareils à un seul point de contact, ce qui est différent de la main dans le monde réel, car la main permet de définir une multitude de points et de surfaces de contacts. Ainsi, un unique utilisateur manipulant un objet ne peut établir que deux points de contact en même temps (un par main). Ces points définissent alors une droite autour de laquelle il est impossible

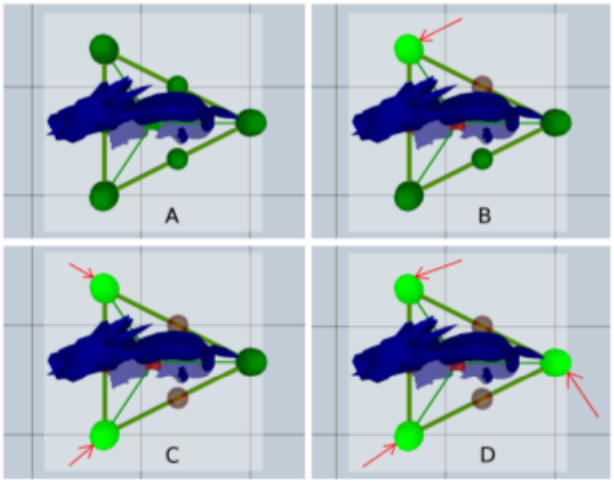

FIGURE 3 – *7-Handle* [5]. (A) Tous les points de contact sont verrouillés (vert foncé), l'objet ne peut pas bouger. (B) Un point de contact est déverrouillé (vert clair), la rotation autour de l'axe opposé est possible. 2 points secondaires et le point tertiaire sont désactivés (rouge). (C) les rotations autour du point verrouillé sont possibles. (D) Tous les mouvements sont possibles (équivalent d'un contact à trois points).

de tourner (l'image d'une droite par une rotation autour d'elle même est elle même). Seuls 5 DDL sont alors possibles. Ceci ne permet pas d'avoir d'interactions naturelles avec les objets manipulés. Une solution à ce problème est proposée par [4]. Elle consiste à définir trois points de contacts non alignés, rendant ainsi possible l'exploitation des 6DDL. Cependant, cette manipulation requiert la coopération de deux utilisateurs au moins. Une alternative est proposée par [5] (section 3). Celle-ci consiste à définir sept points de contact. Ces points sont en réalité décorrélés de l'objet et placé sur un outil, le *7-Handle*, constitué d'un triangle. Les sept points de contact sont les sommets du triangle (points principaux), les milieux des segments (points secondaires) et son centre de gravité (point tertiaire). L'outil est associé avec l'objet à manipuler en mettant en correspondance leur centre de gravité. La manipulation se fait alors en deux phases :

1. **Configuration :** il est possible d'adapter la taille du *7-Handle* et la position des points en fonction de l'objet à manipuler.
2. **Manipulation :** l'utilisateur peut manipuler les points du *7-Handle*, chacun d'entre eux asservissant un point correspondant de l'objet. Il est possible de verrouiller un ou plusieurs points principaux, qui cessent alors de suivre le mouvement en adoptant une position fixe. Cela permet de compenser le manque d'utilisateurs et donc d'exploiter les 6DDL. Les points secondaires et le point tertiaire peuvent être utilisés à condition qu'aucun point adjacent ne soit déverrouillé. Dans le cas contraire, ces points sont désactivés (on ne peut plus les saisir mais ils ne sont pas bloqués). la figure 3 permet de mieux comprendre le fonctionnement du *7-Handle*.

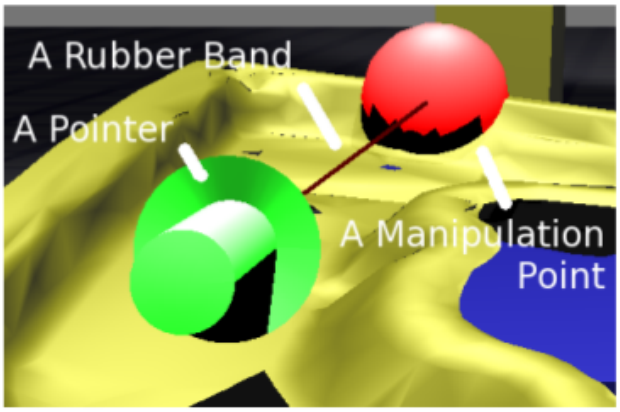

FIGURE 4 – Points de contact et pointeurs d'utilisateur [4].

3.1.2 Prendre conscience de l'influence d'un coopérateur

Il est important de prendre conscience de l'action d'un coopérateur lors d'une manipulation, car cela permet d'adapter sa propre interaction en conséquence. Un moyen simple de représenter la force exercée par un utilisateur sur un objet est de représenter le vecteur correspondant, avec pour origine le point de contact concerné. Dans certains cas, ce point de contact n'est pas sur l'objet (e.g cas du *7-Handle* [5]), ou l'utilisateur contrôle ce point de contact au moyen d'un pointeur, la main réelle n'est alors pas en contact direct avec l'objet 3D. Il est donc nécessaire de montrer quel utilisateur contrôle quel point de contact. Cela peut se faire au moyen d'un pointeur virtuel relié au point de contact et indiquant la direction de l'opérateur ainsi que son identité (par un code couleur), comme c'est le cas dans [4] (voir figure 4). La proximité entre l'appareil et le point de contact est ici également représentée par un code couleur (vert si l'appareil est éloignée, rouge s'il est proche). En effet, une main éloignée du point de contact créé beaucoup d'instabilités.

3.2 Interactions coopératives : un problème de synchronisation

La manipulation d'objets n'est pas le seul moyen d'interagir avec l'environnement. Il est en effet possible d'écrire sur certains objets pour utiliser une communication écrite, utile si tous les utilisateurs ne disposent pas d'appareils de restitution sonore [2]. Il est aussi possible d'interagir de manière plus surréaliste, en changeant la couleur des objets, en en créant et en en détruisant, ou encore en déplaçant des murs.

Quelque soit le type d'interaction, la gestion de la coopération ne peut pas se passer de mécanismes de synchronisation. En effet, la transmission des événements sur un réseau peut conduire à des ordres différents de réception et donc à des incohérences entre les états de chaque processus. Dans le cas des EVC, il existe deux possibilités afin de transmettre le changement d'état d'un objet sur le réseau :

1. le processus responsable du changement d'état diffuse des copies de l'objet modifié ;
2. le processus responsable du changement d'état diffuse les opérations qui ont conduit au changement d'état, afin qu'elles soient à leur tour appliquées sur chaque copie locale de l'objet.

En matière de synchronisation des collaborations en EVC, on distingue trois niveaux [6] :

1. Il n'y a pas d'interactions possibles entre les utilisateurs et l'environnement.
2. Seules des interactions individuelles sont possibles.
3. Des interactions collaboratives sont possibles. On distingue alors deux sous cas : (3.1) seules des interventions indépendantes sont permises en simultané (e.g déplacement et changement de la couleur) ou (3.2) le système gère les interventions interdépendantes (e.g en calculant la couleur moyenne lors d'un changement de couleur).

Dans le premier cas, il n'y a évidement pas besoin de mécanismes de synchronisation. Dans le deuxième, il faut pourvoir les objets de verrous qui empêchent l'intervention de plus d'une personne sur un objet. C'est une technique défensive, qui permet d'éviter des interactions incohérentes sur un même objet, par exemple deux changements de couleurs divergents simultanés. Le troisième cas est traité dans l'article [6]. Il faut garantir deux propriétés :

1. Tous les processus doivent s'accorder sur l'ordre d'exécution des opérations sur un même objet.
2. Tous les processus doivent s'accorder sur la simultanéité des opérations sur un même objet, c'est à dire que si un processus considère que deux opérations ont eu lieu simultanément, alors tous les processus doivent considérer la même chose. Cette propriété est utile pour le cas 3.2.

Avant tout, il faut définir ce qu'est une opération sur un objet. L'article [6] la définit comme étant le changement de valeur d'un gestionnaire d'interaction (GI). Un GI est matérialisé dans l'EV par une poignée ou une boule par exemple. En particulier, les points de contact sont des GI permettant de changer les coordonnées d'un point. D'autres GI peuvent par exemple permettre de changer la valeur de la couleur ou de la transparence d'un objet. Ainsi les opérations peuvent être associées à des événements ordonnés dans le temps.

La première propriété est garantie à l'aide d'une contrainte qui doit être imposée sur les messages échangés sur le réseau et dont se charge l'objet concerné : il faut que les messages soient diffusés en ordre complet, c'est à dire que tous les processus les reçoivent dans le même ordre. Si les messages sont les opérations effectuées, alors elles sont prises en compte dans le même ordre par toutes les fonctions de transformation qui sont les mêmes pour tous les processus. Sauf cas (très) exceptionnel, ces fonctions sont déterministes et donnent donc le même résultat pour la même séquence d'opération. Si les messages sont des copies de l'objet, alors la cohérence entre les processus est obtenue de manière évidente.

La deuxième propriété s'obtient grâce à un mécanisme d'inscription / désinscription au moment où un utilisateur commence à interagir avec un objet (resp. termine son interaction). A chaque exécution d'opération, l'objet considère que tous les utilisateurs inscrits sont en activité simultanée et opère d'abord la diffusion d'un message contenant le nombre d'inscrits. Ensuite, à chaque opération effectuée, la valeur du GI correspondant est à son tour diffusée. Chaque processus attend donc le nombre d'opérations correspondant au nombre reçu au préalable, avant de calculer la transformation résultante. Si un utilisateur reste trop longtemps "détenteur" d'un GI sans pour autant réaliser d'interaction, la valeur du GI est considérée constante et un signal spécial est diffusé afin de ne pas faire attendre inutilement les processus.

4 Aspects sociaux

Comme vu en section 2.2, la gêne induite par une métaphore est à minimiser. En EVC immersif, il existe une gêne particulière liée à la position des acteurs les par rapport aux autres : chaque individu définit une zone "intime" autour de lui, dont le rayon varie entre 15cm et 40cm selon les personnes. En règle générale, si une personne pénètre cette zone, cela provoque une réaction de l'individu, qui va du simple recul à un énervement. Cette notion a été introduite par Edward T. Hall dans l'ouvrage [9]. Cette conception de l'espace s'appelle la proxémique. Elle est mise en jeu si deux acteurs physiques se retrouvent dans le même espace immersif.

En EVC, et particulièrement dans les applications d'apprentissage, il existe plusieurs cas qui impliquent une forte proximité physique entre deux acteurs : un entraîneur peut avoir besoin de prendre le point de vue se son apprenti si celui-ci demande de l'aide, un apprenti lui-même peut avoir besoin d'observer son entraîneur lors d'une démonstration, deux coopérateurs peuvent avoir besoin de manipuler un même petit objet en même temps... en guise d'exemple, l'article [3] présente un cas d'usage mettant en jeu d'une part un apprenti devant apprendre les positions de pièces dans un moteur de voiture, et d'autre part son entraîneur qui lui montre lesdites pièces. Dans certains cas, l'apprenti doit observer par dessus l'épaule de l'entraîneur, ce qui provoque effectivement une gêne pour ce dernier.

Pour le problème de la manipulation, la solution est d'utiliser des points de contact distants de l'objet (voir la section 3.1). Pour le problème de l'observation, la solution au problème consiste à utiliser un pointeur projetant un rayon de lumière virtuel pour mettre en évidence l'objet. Comme la nécessité de regarder par dessus l'épaule est en général due à un obstacle qui bloque le champ de vision, le rayon doit effacer partiellement l'obstacle, comme cela est présenté dans [3]. Cette technique n'améliore pas l'efficacité de l'apprentissage, mais limite au moins la gêne.

5 Scénario

En EVC, les utilisateurs ont en général un but à atteindre qui nécessite l'exécution de plusieurs tâches successives. Cependant, tout ne doit pas nécessairement s'opérer de manière linéaire. L'ensemble des utilisateurs peut se scinder en plusieurs groupes qui réaliseront des tâches en parallèle et des choix peuvent être offerts entre plusieurs tâches possibles... ce qui génère des embranchements complexes. On appelle cela un scénario, qu'il faut formaliser afin de gérer efficacement l'enchaînement des tâches. Ainsi, le cycle de vie d'une tâche est constitué des étapes successives "définition, exécution, conséquence". Pour illustrer, considérons un scénario simple constitué de deux tâches : un utilisateur doit placer une pièce sur un support, puis lui appliquer des transformations. Le moteur de scénario prépare d'abord les commandes pour la première tâche (définition) : des GI de manipulation (haut, bas, gauche, droite, en avant, en arrière) sont mis en place pour déplacer la pièce à l'aide d'une machine par exemple. Puis l'opérateur exécute les commandes unes à unes afin de placer la pièce sur le support (exécution). Une fois la fin le but atteint, le système désactive les GI (conséquence) et prépare les commandes pour la tâche suivante (définition) et ainsi de suite.

Les scénarios sont souvent modélisés sous forme de machine à états [7]. Les états représentent

chacun une tâche et les transitions sont gardées par les conditions de fin de tâche. Cependant, cette représentation a l'inconvénient de ne pas définir directement la notion d'états courants [3]. Pour cela il faut avoir recours à des visiteurs du graphe. D'autres modèles se basent sur les réseaux de Petri, comme #SEVEN [8], car ils ont l'avantage de fournir la notion d'états courants grâce aux tokens. Un exemple simple permettant de comprendre la signification exacte des tokens consiste à considérer que chaque token représente l'un des terminaux physiques impliqués dans l'application collaborative. Cela permet de paralléliser l'exécution du scénario en fournissant à chaque terminal des actions différentes. Par exemple, lors d'une opération d'assemblage de deux pièces, l'un des terminaux permettra de déplacer une seule des deux pièces et réciproquement. Quant aux places du réseau de Petri, elles représentent les tâches du scénario et les transitions représentent les événements permettant le passage à une tâche suivante (complétion de la tâche précédente). #SEVEN introduit en plus la notion de capteurs et d'effecteurs, qui sont attachés aux transitions : le capteur d'une transition écoute l'environnement en attendant la validité de la condition de fin de tâche (e.g la pièce a atteint son support). Une fois celle ci validée, il active la transition qui fait passer les tokens aux états suivants, puis l'effecteur s'active en produisant les conséquences de l'action sur l'environnement (e.g désactiver les commandes devenues inutiles et activer celles qui correspondent aux états suivants).

#SEVEN utilise en réalité une hiérarchie de réseaux de Petri : les places du réseau principal sont en fait des réseaux de Petri fils. En effet, une tâche peut elle même être décomposée en sous-tâches. Par exemple, le fait de déplacer une pièce au moyen des commandes *haut, bas, gauche, droite, en avant, en arrière* peut être vu comme une succession en boucle de tâches correspondant chacune à l'une de ces commandes. Chaque tâche principale est ainsi décomposée en un réseau de Petri constitué de places initiales et de places finales. Lorsqu'un token atteint une place principale, celle-ci n'est pas immédiatement considérée atteinte. D'abord, le token est dupliqué sur les places initiales du sous-réseau. Lorsque les places finales sont toutes atteintes par un token, la place principale reçoit effectivement le token original.

Ce modèle propose également une formalisation de la répartition des tâches entre les acteurs. Chacun d'entre eux est associé à un token. Chaque token est assigné à un ensemble d'actions possibles (les commandes activées), qui définissent ce que l'on appelle un rôle. Ainsi, chaque acteur hérite du rôle du token auquel il appartient. Les effecteurs peuvent modifier l'ensemble des actions qui caractérisent un token afin de faire évoluer les rôles.

La vidéo `https://vimeo.com/104937822` illustre parfaitement #SEVEN en action et permet de bien comprendre toutes ces notions.

6 Conclusion

Les métaphores permettant de guider l'entraîneur ne sont pas évaluées. Détecter le début et la fin d'une activité avec un objet (Margery) ? Comment ? Le problème de proximité physique ne se pose pas si les acteurs ne sont pas physiquement situés dans le même espace immersif. Dans ce cas, il existe la possibilité de changer de point de vue virtuellement comme cela est présenté

3. pluriel : dans un scénario il peut y avoir plusieurs états courants car des tâches peuvent être effectuées en parallèle.

dans l'expérience de guidage de [5] (section 2), ou même d'échanger les avatars. Cependant, il faut faire attention aux métaphores dites intrusives : la fait de rapprocher deux avatars peut produire le même effet qu'en cas de collaboration physique... (//à compléter : main intrusive, prise de contrôle etc)

Références

[1] LOPEZ, Thomas, BOUVILLE, Rozenn, LOUP-ESCANDE, Emilie, et al. *Exchange of avatars : Toward a better perception and understanding. Visualization and Computer Graphics, IEEE Transactions on*, 2014, vol. 20, no 4, p. 644-653.

[2] NGUYEN, Thi Thuong Huyen, DUVAL, Thierry, et al. A Survey of Communication and Awareness in Collaborative Virtual Environments. In : *2014 International Workshop on Collaborative Virtual Environments (3DCVE)*. 2014.

[3] ARGELAGUET SANZ, Fernando, KUNERT, André, KULIK, Alexander, et al. Improving co-located collaboration with show-through techniques. 2010.

[4] AGUERRECHE, Laurent, DUVAL, Thierry, LÉCUYER, Anatole, et al. Short paper : 3-hand manipulation of virtual objects. JVRC 2009, 2009.

[5] NGUYEN, Thi Thuong Huyen. *Proposition of new metaphors and techniques for 3D interaction and navigation preserving immersion and facilitating collaboration between distant users*, PhD thesis, 2014.

[6] MARGERY, David, ARNALDI, Bruno, et PLOUZEAU, Noel. *A general framework for cooperative manipulation in virtual environments*. Springer Vienna, 1999.

[7] CREMER, James, KEARNEY, Joseph, et PAPELIS, Yiannis. HCSM : a framework for behavior and scenario control in virtual environments. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 1995, vol. 5, no 3, p. 242-267.

[8] CLAUDE, Guillaume, GOURANTON, Valérie, BERTHELOT, Rozenn Bouville, et al. Short Paper :# SEVEN, a Sensor Effector Based Scenarios Model for Driving Collaborative Virtual Environment. In : *ICAT-EGVE, International Conference on Artificial Reality and Telexistence, Eurographics Symposium on Virtual Environments*. 2014. p. 1-4.

[9] E. T. Hall. *The Hidden Dimension*. Doubleday, 1966.