Une architecture pour le Retour d'Efforts.

T.Meyer, G.Andrade-Barroso & B.Arnaldi.

IRISA 35042 Rennes Cedex

[Tangi.Meyer|Guillermo.Andrade-Barroso|Bruno.Arnaldi]@irisa.fr

Résumé : *Ce document présente l'architecture pour la simulation avec retour d'efforts que nous avons développée. Notre objectif est de proposer des outils facilitant la mise en oeuvre d'applications métier. Nous avons implémenté des interfaces qui permettent une intégration générique des éléments logiciels nécessaires au retour d'efforts. Nous décrivons également les évolutions en cours et les perspectives qu'offre notre architecture.*

Mots-clés : retour d'efforts, architecture logicielle, réalité virtuelle.

1 Introduction

De nombreuses applications de réalité virtuelle exploitent le retour d'efforts depuis que certains périphériques comme le PHANToM (c)[29] ou le Virtuose (c) de Haption[1] sont "sortis des laboratoires" pour être commercialisés. Des domaines comme la CAO (assemblage virtuel, ergonomie, design...), la médecine (chirurgie, rééducation...) ou la formation, bénéficient grandement du développement des techniques de réalité virtuelle et plus particulièrement du retour d'efforts.

OpenMASK(c)[2] est une plate-forme Open Source de développement d'applications pour la réalité virtuelle [10], issue des travaux du projet SIAMES de l'IRISA. De nombreux travaux exploitant cet outil ont été réalisés : simulation d'environnements urbains [14], animation d'humanoïdes [22], manipulation coopérative [28] et distante de données industrielles...

Dans le cadre de l'extension de cette plate-forme, le développement d'applications de réalité virtuelle exploitant le retour d'efforts est un objectif naturel. Dans la logique des travaux de l'équipe SIAMES, nous cherchons à proposer des fonctionnalités et non un outil spécialisé. Celles-ci ont pour objectif de faciliter la mise en oeuvre des outils nécessaires à la simulation avec retour d'efforts. En effet, bien que le retour d'efforts soit un dénominateur commun aux applications sus-citées, il semble irréaliste de vouloir développer un outil universel pour sa simulation tant les besoins en terme de modèles, de formats, sont différents. A partir de cette constatation et dans un soucis d'ouverture d'OpenMASK vers le plus grand nombre d'applications de réalité virtuelle, nous avons cherché à identifier les principaux besoins en terme d'outils logiciels pour la simulation avec retour d'efforts. Nous avons alors défini une architecture logicielle modulaire et ouverte afin que les utilisateurs d'OpenMASK puissent facilement intégrer leur travaux et développer des applications plus spécialisées. Nous présentons ici, les travaux réalisés autour d'OpenMASK dans ce but.

La suite du document s'organise de cette façon : à partir d'un état de l'art, nous identifions les principaux éléments logiciels nécessaires à la simulation avec retour d'efforts. Nous présentons ensuite l'architecture développée et les premiers résultats obtenus. Les évolutions envisagées sont exposées dans la quatrième partie du document. Enfin nous présentons les perspectives qu'offre notre architecture.

2 État de l'art

2.1 Exemples d'outils pour la simulation avec retour d'efforts

De nombreux développements ont déjà été réalisés pour des applications spécifiques. Voxmap Point Shell [31] de Boeing (c) est un logiciel destiné à la navigation interactive dans des environnements de CAO avec un périphérique à retour d'efforts. La scène simulée est représentée sous forme de voxels et l'objet que l'utilisateur manipule sous la forme d'un nuage de points. L'interaction entre la pièce mobile et le reste de l'environnement est réalisé au

[1] http ://www.haption.com

[2] http ://www.openmask.org

travers d'une méthode dite de pénalité appliquée aux points du nuage qui intersectent avec les voxels de plus petite dimension. La configuration à chaque pas de temps est calculée en intégrant les équations de Newton-Euler. Le périphérique haptique est lié à la simulation par un couplage virtuel [20] qui permet de renvoyer un effort vers le manipulateur. Ce principe est également utilisé par D.Baraff [12] pour associer un autre type de périphérique haptique à une bibliothèque logicielle de simulation dynamique pour les corps rigides. Des exercices d'apprentissage simples peuvent être construits à partir de ce système. S.Cotin [25] présente une autre application de la simulation avec retour d'efforts : la chirurgie. Un modèle déformable de foie est simulé et lié à un périphérique haptique spécialisé pour la simulation de chirurgie laparoscopique. L'auteur utilise ici les éléments finis pour calculer une déformation due à l'interaction de l'utilisateur.

Ces exemples mettent en évidence l'étendue des différents domaines et outils logiciels qui sont utilisés pour la simulation avec retour d'efforts.

2.2 Les éléments nécessaires à la simulation avec retour d'efforts

2.2.1 Généralités

Dans la plupart des approches, la scène virtuelle associée à une simulation avec retour d'efforts est un monde contenant des objets dont le comportement est dicté par des lois dans la plupart des cas physiques. Il est possible d'établir des domaines du monde à l'intérieur desquels il règne des lois de comportements bien définis et dont la nature ne change pas au cours du temps. Ces domaines sont par exemple les volumes des objets rigides ou déformables et les avatars des utilisateurs. Ce sont les interactions entre les différentes frontières des domaines qui vont déterminer les conditions aux limites ou de passage d'un domaine à l'autre. Ces conditions aux limites vont modifier l'évolution interne des domaines. Ainsi deux objets rigides se touchant en un point (contact entre deux domaines volumiques) génèrent du frottement entre leurs surfaces (frontières). Ce frottement va définir des forces (conditions aux limites) qui vont modifier les déplacements des objets (comportements).

De ces définitions découlent les éléments nécessaires à la simulation :

– des outils capables de déterminer le lieu des interactions entre les frontières,
– des outils permettant de produire les conditions aux limites associées à ces interactions,
– des outils définissant le comportement d'un domaine à partir de son état actuel et des conditions aux limites associées à ses frontières.

Dans la plupart des travaux sur la réalité virtuelle avec retour d'efforts, ces outils prennent les dénominations respectivement de ***détecteur de collision***, de ***traitement du contact*** et de ***solveur physique*** ou de ***solveur mécanique***.

2.2.2 Détection de collisions.

La détection de collisions a été étudiée dans de nombreux domaines comme la robotique et l'informatique graphique. Les besoins ou objectifs peuvent être très différents (application temps-réels, résultat exact, adaptation des données CAO, déformations, etc.). De ce fait, de nombreux algorithmes ont été développés [19] [30][1][6][26].

Toutefois, les étapes essentielles de leur fonctionnement peuvent être décrites simplement par la figure 1. Après une mise à jour des positions et des vitesses des objets concernés, on réalise le test de détection de collisions. Les données géométriques obtenues en résultat sont traitées afin d'être envoyées au calculateur de réponses au contact.

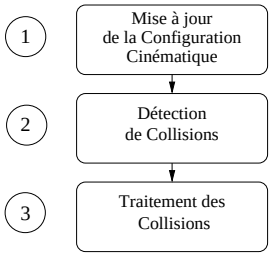

FIG. 1 – Fonctionnement simplifié de la détection de collisions.

2.2.3 Traitement du Contact

Il existe plusieurs traitements possibles de la collision : par une méthode de type pénalités [32], par une méthode impulsionnelle [7] ou par une méthode dite LCP [11]. Chacune produit des informations à traiter par le solveur mécanique. Le résultat de ce traitement est l'apparition de forces de frottement et de non pénétration ou directement de changements dans les champs de vitesses des objets ou dans celui des accélérations. Ces effets seront traités à différents niveaux dans le solveur mécanique.

2.2.4 Mécanique

Il existe de nombreux types de simulateurs physiques. Il sont parfois spécialisés. Ils peuvent être basés sur des algorithmes et des méthodes de résolution très différentes. La robotique est à l'origine d'outils spécialisés dans le traitement des chaînes poly-articulées. SMR [3] par exemple dispose d'un traitement optimisé des chaînes poly-articulées fermées. D'autres logiciels permettent la simulation physique d'ensembles de corps rigides : Dynamo [4], ODE[3], ou encore CONTACT [27]. Vortex de CMLabs[4] a été développé pour les jeux vidéo. Pour animer les corps déformables, Les méthodes retenues sont souvent des modèles mécaniques simplifiés afin de satisfaire les contraintes de l'animation interactive [18].

Bien que d'un point de vue général, ces logiciels permettent de traiter l'animation mécanique, leur variété met surtout en évidence la diversité des problèmes à traiter. Toutefois, comme dans le cas de la détection de collisions, il est possible de décrire les grandes étapes de la résolution du système mécanique (Cf. figure 2). Le système mécanique est construit à partir de l'ensemble des informations dynamiques de la scène. Le système d'équations obtenu lie les forces et contraintes aux accélérations des différents degrés de liberté du système. Une intégration numérique produit alors un nouvelle configuration cinématique.

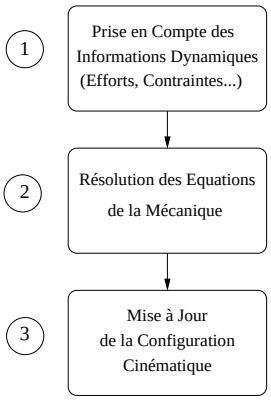

FIG. 2 – Résolution Mécanique simplifiée.

2.2.5 Manipulation interactive haptique

L'immersion de l'utilisateur passe par une manipulation interactive d'objets virtuels. Dans le cas d'une souris, cela consiste à commander directement en position l'objet que l'on manipule. L'introduction du retour d'efforts nécessite une approche plus évoluée : il ne s'agit plus de se contenter de visualiser des déplacements, il faut également renvoyer des informations haptiques à l'utilisateur.

Le principe directeur peut être décrit de la façon suivante : nous disposons de deux représentations d'un même objet, la représentation "réelle" qui est asservie en position et vitesse au périphérique haptique, ainsi qu'une représentation virtuelle dont les position et vitesse sont calculées par la simulation. L'objectif est de faire coïncider ces deux représentations ou tout du moins à les faire tendre l'une vers l'autre. Pour cela, il faut définir un outil bilatéral qui permettra à l'utilisateur de ressentir les efforts nécessaires à cette correspondance (et qui permettra une prise en compte de l'influence de l'utilisateur dans la simulation).

[3]http ://www.q12.org/ode/
[4]http ://www.cm-labs.com/

Plusieurs méthodes ont été proposées : nous avons évoqué le couplage virtuel [20] dans le paragraphe 2.1. Cette technique exploitée par [5], [31] et [12] entre autres, s'accorde très bien avec le principe du rendu d'impédance [9] (mesure de déplacement et renvoi de force, figure **??** (a)). Un lien composé d'un ressort et d'un amortisseur virtuel entre les positions simulée et issue du périphérique permet d'envoyer des informations d'efforts au périphérique et au module de résolution mécanique à chaque pas de temps.

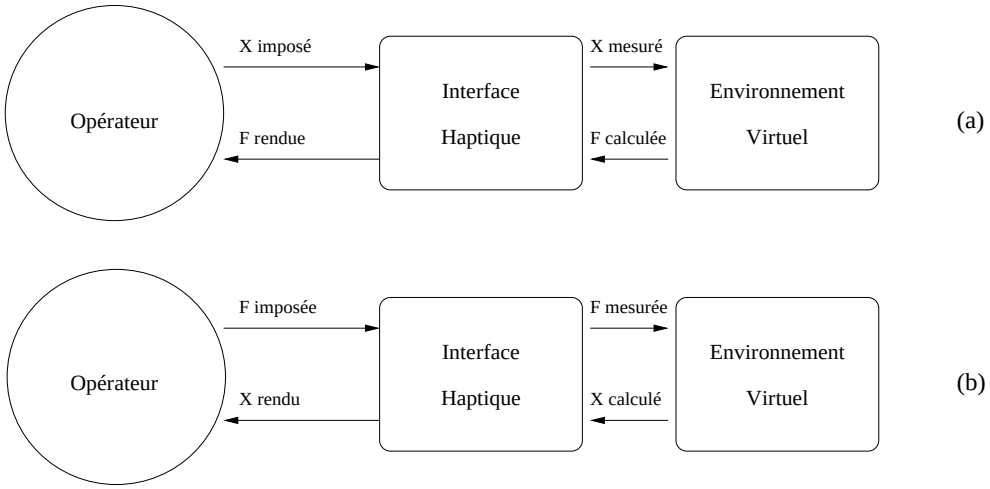

FIG. 3 – Représentation simplifiée du rendu d'impédance (a) et du rendu d'admittance (b).

Une autre approche "par contraintes" a été proposée [8]. Il s'agit d'imposer des contraintes dynamiques entre la position réelle de l'objet manipulé et sa position virtuelle ("God-object"). Le "Proxy" de [13] est associé à une méthode de rendu haptique de type "shading". Ces méthodes permettent d'éviter des incohérences en imposant un suivi plus strict des surfaces. Par exemple, l'utilisateur ne traversera pas un objet lors d'un suivi d'une surface de faible épaisseure.

Comme pour le problème de la détection de collisions et le traitement de la mécanique, la manipulation interactive pour le retour d'efforts peut être résolue de plusieurs façon. La technique la plus adaptée dépend souvent de l'application étudiée.

2.2.6 Identification du besoin

La variété des applications et des techniques sus-citées rend très complexe le développement d'une solution universelle. Cependant, les blocs logiciels représentés par les figures 1, 2 et **??** permettent de faire abstraction de la méthode employée tout en définissant des interfaces communes.

3 Architecture

Notre travail consiste en une formalisation des échanges d'informations pour définir une architecture modulaire ouverte pour la simulation avec retour d'efforts.

3.1 Principe

Le principe de notre architecture est d'offrir des outils génériques pour l'intégration. Nous avons développé plusieurs interfaces abstraites pour les modules logiciels indispensables que nous avons identifiés dans la section précédente.

La figure 5 présente la première étape de nos travaux. Nous l'avons basée sur le fait que les modules mécaniques étaient plus aisément commandables en forces. Ainsi nous considérons que le traitement de la collision est réalisé au travers d'un modèle de pénalité. De la même façon, nous avons retenu le couplage virtuel comme liaison périphérique/simulation. La figure 4 présente le modèle général de nos modules.

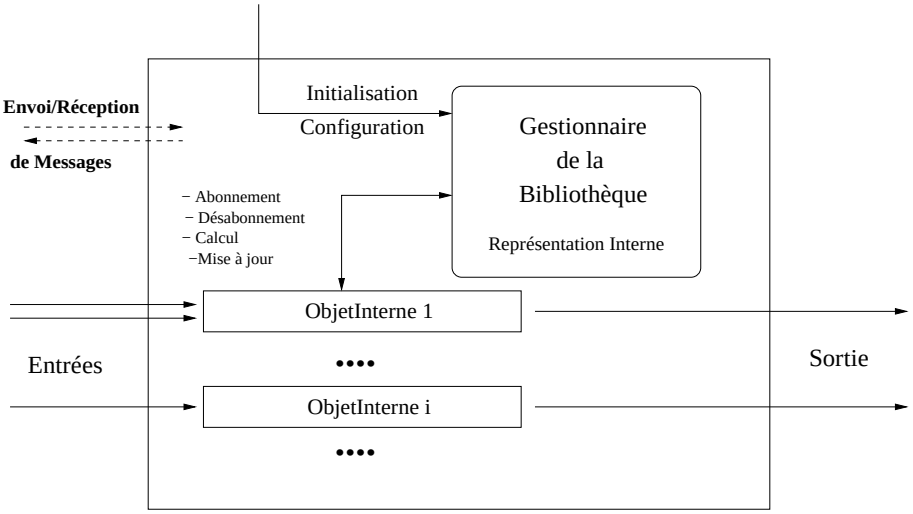

FIG. 4 – Représentation d'un module.

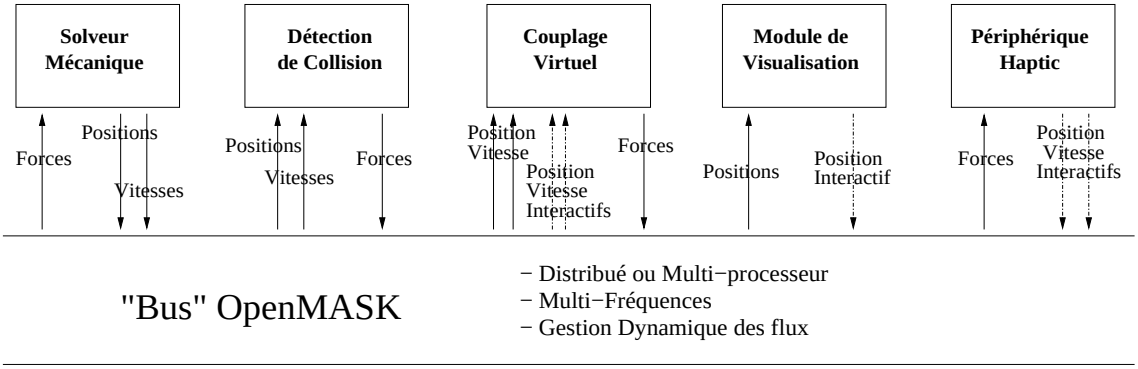

FIG. 5 – Représentation simplifiée de l'architecture.

Notre architecture exploite plusieurs propriétés d'OpenMASK :

– La notion de bus logique et l'architecture modulaire.
– le module de visualisation avancée qui nous évite des travaux complexes concernant le rendu.
– les fonctionnalités distribuées et multi-processeurs.

Mais avant tout, ce sont les outils de communication entre les modules qui nous intéressent. Plus particulièrement, nous exploitons les notions de flux d'entrées et de sorties pour les signaux continus et les messages pour les signaux discrets.

Dans le premier cas, il s'agit pour un module de s'abonner à des informations provenant d'un autre module ou encore de produire des sorties qui seront connectées aux entrées d'un module différent. Nous avons développé des interfaces basées sur les principes présentés par les figures 2 et 1, qui généralisent le type des entrées et des sorties des modules afin de rendre ceux-ci compatibles. Comme le montre la figure 5, un module "solveur mécanique" dispose d'entrées de type forces. Celles-ci permettent la prise en compte d'éventuels contacts[5] ou d'interactions de type couplage virtuel. Après intégration des équations de la mécanique, le solveur propose une nouvelle configuration cinématique par l'intermédiaire de ses sorties de types position et vitesse. Un module de détection de collisions peut se brancher sur ces sorties afin de mettre à jour sa représentation interne du monde. C'est également le cas pour un module de type couplage virtuel ou pour le module de visualisation.

OpenMASK nous permet une gestion dynamique des flux. Pour cela nous exploitons les messages : ils facilitent l'échange d'informations typées entre deux modules. En particulier, nous provoquons l'abonnement ou le désabonnement des entrées à un flux par leur intermédiaire. Dans le cas de l'interaction avec une souris par

[5]Notre premi`ere architecture est bas´ee sur un mod`ele de type p´enalit´e en ce qui concerne le traitement de la collision.

exemple, un couplage virtuel est créé entre la souris et l'objet virtuel dont on prend le contrôle. Celui-ci génère des efforts qui s'exercent sur l'objet manipulé. Le solveur mécanique chargé de l'animation de cet objet doit donc s'abonner à la sortie *force* du couplage virtuel. Dans ce but, un message de type `AddForceStream` est envoyé par la souris au système mécanique qui en réponse crée une entrée de type *force* qui se connecte à la sortie spécifiée. Le désabonnement est réalisé de la même façon grâce à un message `DeleteForceStream`.

3.2 Premiers résultats

Les objectifs de cette plate-forme sont avant tout de faciliter le développement d'applications métier. Pour cela l'interchangeabilité des modules est un élément essentiel. Afin de valider cet objectif, nous avons procéder au développement de simulation basées sur différentes bibliothèques logicielles. En ce qui concerne les solveurs mécanique, Dynamo, SMR et CONTACT ont été portés avec succès. Notre interface s'est avérée fonctionnelle et a permis des développements rapides.

Dans le même but, des travaux sur diverses librairie de détection de collisions ont été réalisés : VCollide, SWIFT++ [24], les résultats étant traités par un modèle de pénalité.

L'architecture nous permet également de mixer dans une même scène la simulation de plusieurs systèmes basés sur diverses librairies.

Nous avons également validé la manipulation interactive de type "couplage virtuel" unilatéral (commande en force de la simulation par la souris). L'interface générique permet une manipulation interactive transparente des objets quelque soit le solveur mécanique qui l'anime.

4 Évolution de l'architecture

4.1 Généralisation du couplage virtuel

Notre architecture permet entre autre de faire coexister plusieurs systèmes de type simulateur dynamique de corps rigides. Commander en position ce type d'outil peut s'avérer délicat en particulier si le système mécanique est constitué d'une chaîne comprenant des contraintes bilatérales telles que des rotules, des pivots... En effet, les manipulations directes risquent d'engendrer des violations de contraintes qui peuvent faire diverger le système lors de l'intégration numérique.

Afin d'éviter ces écueils, il est souhaitable de recourir à un mode de commande plus proche de la dynamique (i.e. du second ordre). Le couplage virtuel présente des caractéristiques intéressantes pour transformer les sorties de type position d'un système en entrées de type force pour un autre solveur mécanique. Par exemple, dans le cas d'une simulation de téléopération (Cf. figure 6), il peut être intéressant de travailler avec deux logiciels différents : le premier issu de travaux en robotique permettra une manipulation du robot virtuel, par contre il sera sans doute moins pertinent pour des travaux de suivi de surface, d'où l'exploitation éventuelle d'un second logiciel. Le couplage virtuel permet un lien transparent entre ces outils : chaque logiciel anime une représentation d'un objet (ici l'extrémité du robot) et elles sont ainsi associées par une liaison bilatérale.

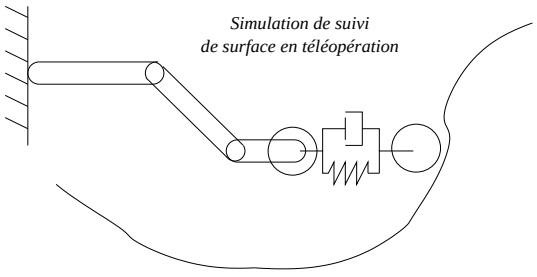

FIG. 6 – Exemple de généralisation du couplage virtuel.

Nous avons donc étendu l'utilisation de la notion de couplage virtuel : il nous permet de faire communiquer des systèmes mécaniques basés sur des logiciels différents.

4.2 Types des Entrées/Sorties

La généralisation du couplage virtuel conduit à rencontrer des systèmes équivalents à la figure 7 où l'objet A et son image sont animés par des simulateurs physiques différents tout en étant liés par un couplage virtuel.

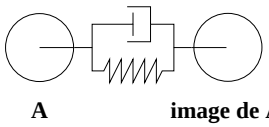

FIG. 7 – Couplage Virtuel généralisé.

La représentation de ce système pour notre architecture est celle de la figure 8. Où le *solveur mécanique* n °1 anime l'objet A et le *solveur mécanique* n °2 son image (X représente ici un vecteur position+vitesse).

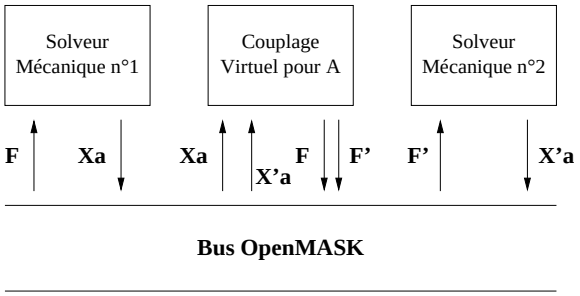

FIG. 8 – Architecture et couplage virtuel généralisé.

Cette configuration peut poser quelques problèmes dans le cas d'une utilisation distribuée d'OpenMASK. Plus particulièrement, les entrées de type forces des solveurs mécaniques risquent de conduire à des instabilités dues à des erreurs de fonctionnement du réseau (latence, perte de données...). Nous avons donc décidé d'instaurer une communication de flux position+vitesse entre les solveurs mécaniques en intégrant le couplage virtuel sous la forme d'un pré-traitement des entrées des solveurs afin d'obtenir une architecture proche de la figure 9.

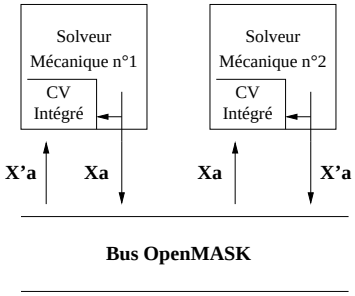

FIG. 9 – Couplage Virtuel intégré.

Nous avons réalisé, avec Simulink (c) [17], une étude comparative de stabilité entre ces deux modes de traitement du couplage virtuel. L'objectif est d'étudier la convergence entre Xa et $X'a$ lorsque le système mécanique est soumis à des perturbations. Les données obtenues[6] mettent en évidence une stabilité accrue et une convergence améliorée lorsqu'on intègre le couplage virtuel au solveur mécanique (modèle de la figure 9).

Nous cherchons à généraliser la communication de type position. En effet, outre les avantages en terme de stabilité, ce modèle permet d'ouvrir le panel des traitements applicables à la collision par exemple. Il devient possible d'appliquer des modèles plus évolués comme des contraintes de type non-pénétration ou contact point/plan. Un

[6] Les r´esultats de cette ´etude feront l'objet d'une publication ult´erieure.

message permettra de spécifier, au moment de la création de l'entrée de type position généralisée, le traitement à appliquer (Cf. figure 10).

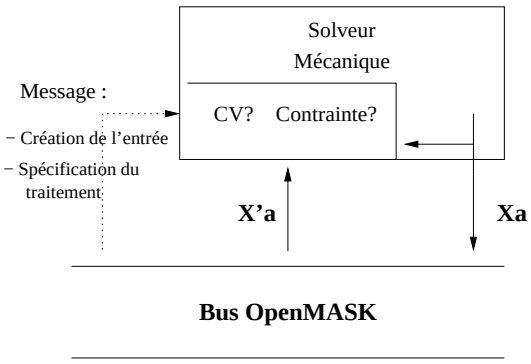

FIG. 10 – Principe d'une entrée généralisée.

5 Perspectives

5.1 Objets déformables

Plusieurs applications du retour d'efforts concernent les objets déformables : la chirurgie [25], la sculpture virtuelle [15] ... Valider notre architecture pour les modèles de corps déformables sera un de nos objectifs futurs. Les particularités d'un solveur mécanique pour ce type de modèle ne paraissent pas incompatibles avec les développements réalisés.

Les intérêts de tels développement sont multiples : il sera possible de coupler la simulation de chirurgie à un simulateur de robotique afin de réaliser une application de téléopération chirurgicale. L'aspect distribué d'OpenMASK nous permettra en outre de répartir les calculs aisément pour satisfaire les besoins en performance.

5.2 Localisation des contraintes logicielles du retour d'efforts

La performance constitue une contrainte forte pour une simulation avec retours d'efforts. En effet, la perception haptique impose des fréquences de fonctionnement élevées pour la simulation : la bande passante de la perception des efforts pour un manipulateur se situe à quelques dizaines de hertz mais celle de la perception tactile est de l'ordre de 300Hz [16] [2].

Les nouveaux types d'environnements de visualisation comme les salles immersives ou les CAVE (c) permettent de travailler sur des modèles de grandes dimensions comme des véhicules à l'échelle 1. Dans la plupart des cas, à un instant donné, le retour d'efforts ne concerne qu'une partie réduite de la scène. Différents travaux sur les modélisations pour la chirurgie à retour d'efforts se rapprochent de la notion de niveaux de détails haptiques [21]. Hayward [23] propose deux niveaux de maillages éléments finis fonctionnant à des fréquences différentes. Debunne [18] fait cohabiter plusieurs maillages de résolutions différentes.

Notre architecture nous permet d'envisager une localisation géométrique des contraintes du retour d'efforts qui se rapprocherait de cette idée de niveau de détails haptiques. En déterminant une zone d'intérêt autour de l'objet que l'utilisateur manipule, il devient possible de séparer les objets de la scène en deux groupes : une partie d'entre eux sera directement concernés par l'interaction haptique (fréquences élevées de simulation), le reste de la scène aura un comportement découplé de l'interaction (contraintes logicielles relachées : fréquences faibles).

5.3 Travail coopératif

Un de nos objectifs est également de développer des applications de travail coopératif. Notre architecture permet la mise en oeuvre de différents périphériques tout en conservant les mêmes interfaces de communication avec la

simulation.

Nous envisageons également d'exploiter les travaux sur l'interaction à distance réalisés sur la plate-forme GASP à l'origine d'OpenMASK. Le réseau VTHD a en effet permis des manipulations interactives entre le projet I3D de l'INRIA Rocquencourt et le projet SIAMES de Rennes. Des démonstrations de manipulations coopératives à distance intégrant le retour d'efforts sont actuellement à l'étude.

6 Conclusions

Nous avons présenté dans ce document les travaux que nous avons réaliser dans le but d'intégrer des applications de réalité virtuelle avec retour d'efforts sur la plate-forme OpenMASK. Nous proposons une architecture modulaire destinée à faciliter l'insertion de bibliothèques logicielles spécialisées. Divers développements nous ont permis de valider la première version de cette architecture.

Actuellement, nous développons des évolutions de nos interfaces afin de rendre plus générique et plus robuste cette architecture. Les perspectives sont relativement nombreuses. L'extention aux simulations avec retour d'efforts pour les corps déformables en est une, tout comme les applications coopératives et distantes.

Références

[1] A.Gregory, M.Lin, S.Gottschalk, and R.Taylor. A framework for fast and accurate collision detection for haptic interaction. In *IEEE Virtual Reality 1999 Conference*, volume 1, pages 38–45, Houston, Texas, USA, March 1999.

[2] A.Lecuyer. *Contribution à l'étude des retours haptique et pseudo-haptique et de leur impact sur les simulations d'opérations de montage/démontage en aéronautique*. PhD thesis, University Paris XI, November 2001.

[3] G. Andrade. *Modélisation et adaptation du mouvement de robots tout-terrain*. PhD thesis, University Paris 6, Paris, France, September 2000.

[4] B. Barenbrug. *Designing a Class Library for Interactive Simulation of Rigid Body Dynamics*. PhD thesis, Technische Universiteit Eindhoven, april 2000.

[5] B.Chang and J.E.Colgate. Real-time impulse-based simulation of rigid body systems for haptic display. In *ASME International Mechanical Engineering Congress and Exhibition*, pages 145–152, Dallas, Texas, USA, 1997.

[6] B.Mirtich. V-clip : Fast and robust polyhedral collision detection. *ACM Transactions on Graphics*, 17(3) :177–208, July 1998.

[7] B.V.Mirtich. *Impulse-based Dynamic Simulation of Rigid Body Systems*. PhD thesis, University of California at Berkeley, Fall 1996.

[8] C.B.Zilles and J.K.Salisbury. A constraint-based god-object method for haptic display. In *IEEE Conference on Conference on Intelligent Robots & Systems*, volume 3, August 1995.

[9] C.R.Carignan and K.R.Cleary. Closed-loop force control for haptic simulation of virtual environnements. *Haptic-e : http ://www.haptic-e.org*, 1(2), February 2000.

[10] David Margery, Bruno Arnaldi, Alain Chauffaut, Stéphane Donikian, and Thierry Duval. Openmask : {Multi-Threaded — Modular} animation and simulation {Kernel — Kit } : a general introduction. In Simon Richir, Paul Richard, and Bernard Taravel, editors, *VRIC 2002 Proceedings*, pages 101–110. ISTIA Innovation, June 2002.

[11] D.Baraff. Fast contact force computation for non-penetrating rigid bodies. In *SIGGRAPH 94 Proceedings*, 1994.

[12] D.Baraff, P.J.Berkelman, and R.L.Hollis. Interaction with a realtime dynamic environment simulation using a magnetic levitation haptic interface device. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 3261–3266, Detroit, Michigan, USA, May 1999.

[13] D.C.Ruspini, K.Kolarov, and O.Khatib. Haptic interaction in virtual environments. In *IEEE International Conference on Intelligent Robots & Systems IROS'97*, Grenoble, France, September 1997.

[14] S. Donikian and G. Thomas. Modeling virtual urban environments for multi-modal driving simulation. In *UM3'99, Int. Workshop on Urban 3D/Multi-Media mapping*, pages 103–110, Institute of Industrial Science (IIS), The University of Tokyo, Japan, 1999.

[15] E.Ferley. *Sculpture Virtuelle*. PhD thesis, Institut National Polytechnique de Grenoble, 2002.

[16] G.C.Burdea and P.Coiffet. *La Réalité Virtuelle.* Hermes, 1993.

[17] G.D.Buckner. Simulink : a graphical tool for dynamic system simulation. In *NCSU ASME Technical Sessions*, October 11 2001.

[18] G.Debunne. *Animation multirésolution d'objets en temps-réel*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, France, December 2000.

[19] J.D.Cohen, M.Lin, D.Manocha, and M.K.Ponamgi. I-collide : An interactive and exact collision detection system for large scale environments. In *ACM Internationnal 3D Graphics Conference*, pages 189–196, 1995.

[20] J.E.Colgate, M.C.Stanley, and J.M.Brown. Issues in the haptic display of tool use. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 140–145, Pittsburgh, PA, USA, 1995.

[21] J.Zhang, S.Payandeh, and J.Dill. Haptic subdivision : an approach to defining level-of-detail in haptic rendering. In *IEEE 10th Symposium On Haptic Interfaces for Virtual Environments & Teleoperator Systems*, Orlando, Florida, USA, March 2002.

[22] N.Courty, S.Menardais, D.Margery, F. Lamarche, S.Donikian, and F.Devillers. Towards believable autonomous actors in real-time applications. In *IMAGINA'02*, Monaco, february 2002.

[23] O.R.Astley and V.Hayward. Multirate haptic simulation achieved by coupling finite element meshes through norton equivalents. In *IEEE International Conference on Robotics & Automation*, volume 1, pages 989–994, Leuven, Belgium, May 1998.

[24] S.A.Ehmann and M.C.Lin. Swift : Accelerated proximity queries between convex polyhedra by multi-level voronoi marching. Technical report, Computer Science Department, University of North Carolina at Chapel Hill, 2000.

[25] S.Cotin and H.Delingette. Real-time surgery simulation with haptic feedback using finite element. In *IEEE International Conference on Robotics & Automation*, volume 3, pages 3739–3744, Leuven, Belgium, May 1998.

[26] S.Redon, A.Kheddar, and S.Coquillart. An algebraic solution to the problem of collision detection for rigid polyhedral objects. In *IEEE International Conference on Robotics & Automation*, San Fransisco, CA, USA, April 2000.

[27] S.Redon, A.Kheddar, and S.Coquillart. Gauss' least constraints principle and rigid body simulations. In *IEEE International Conference on Robotics & Automation*, Washington, DC, USA, May 2002.

[28] T.Duval and D.Margery. Building objects and interactors for collaborative interactions with gasp. In *Proceedings of the Third International Conference on Collaborative Virtual Environments (CVE'2000)*, pages 129–138. ACM, September 2000.

[29] T.H.Massie and J.K.salisbury. The phantom haptic interface : A device for probing virtual object. In *ASME International Mechanical Engineering Congress and Exhibition*, pages 295–300, Chicago, Illinois, USA, 1994.

[30] T.Hudson, M.Lin, J.Cohen, S.Gottschalk, and D.Manocha. V-collide : Accelerated collision detection for vrml. In *VRML'97*, Monterey, CA, USA, 1997.

[31] W.A.McNeely, K.D.Puterbaugh, and J.J.Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *SIGGRAPH*, volume 1, pages 401–408, Los Angeles, California, USA, August 1999.

[32] Y.T. Wang, V. Kumar, and J. Abel. Dynamics of rigid bodies undergoing multiple friction contacts. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2764–2769, Nice, France, May 1992.