Revue Électronique Francophone d'Informatique Graphique, Volume 1, Numéro 1, pp. 1–8, 2007

ToD & DyP, Navigation en temps réel en environnement dynamique.

Thomas LOPEZ [1] et Fabrice LAMARCHE[1]

[1]Équipe BUNRAKU,
IRISA, Campus de Beaulieu, F-35042 Rennes, FRANCE
{thomas.lopezlfabrice.lamarche}@irisa.fr

Résumé

Lorsque l'on traite du peuplement automatisé de bases géométriques 3D par des humanoïdes de synthèse, la modélisation de la navigation est primordiale puisque cette navigation intervient dans la majeure partie des comportements exhibés. Dans de nombreux domaines applicatifs, la nécessité de gérer la navigation dans des environnements dynamiques se fait sentir (mondes virtuels régis par les lois de la physique ou usines numériques dans lesquelles les échafaudages peuvent changer de place par exemple). Il s'agit là de la problématique de ce travail : comment gérer la navigation des humanoïdes virtuels au sein d'environnements dont la topologie peut changer au cours du temps i.e. dont les relations d'accessibilités entre zones peuvent évoluer au fil de la simulation de manière non-prédictible. À la différence des algorithmes actuels, les objets déplacés dans l'environnement peuvent être considérés comme des obstacles mais aussi comme des objets pouvant permettre d'accéder à de nouvelles zones de navigation. La solution proposée découpe ce problème en deux sous problèmes : la détection et le suivi de l'évolution des relations topologiques via l'algorithme ToD (Topology Detection) et le raffinement de cette information pour la planification de chemin via l'algorithme DyP (Dynamic Planner).

When automatically populating 3D geometric databases with virtual humanoids, modelisation of the navigation behavior is essential since navigation is used in most exhibited behaviors. In many application fields, the need to manage navigation in dynamic environments arises (virtual worlds taking physics laws into account, numeric plants in which step stools can be moved,...). This study focuses on the following issues : how to manage the navigation of virtual entities in such dynamic environments where topology may change at any time i.e. where accessibility between areas may change during runtime with no predictability. Contrary to current algorithms, movable items are not only considered as obstacles in the environment but can also help virtual entities in their navigation. The algorithm we propose splits that problem into two parts : detecting and updating topology relations in real time using the ToD algorithm (Topology Detection) and then refining those relations for path planning in the dynamic environments thanks to the DyP algorithm (Dynamic Planner).

Mots clé : Planification de chemin, environnements dynamiques, détection de collisions

1. Introduction

Des applications de réalité virtuelle en passant par les jeux vidéos ou les jeux sérieux (serious games), la modélisation et la simulation d'environnements virtuels peuplés sont au cœur de nombreuses problématiques. Dans divers cadres applicatifs, la structure des environnements 3D dans lesquels s'effectuent les simulations tend à se modifier au cours du temps, que ce soit par l'action de l'utilisateur ou d'un scénario (par exemple, la modification du positionnement d'échafaudages en cours de construction d'un avion dans le cadre d'usines numériques) ou encore en conséquence d'une simulation physique (par exemple, la modélisation d'environnements destructibles dans le domaine des jeux vidéos / jeux sérieux). Ces changements de configuration ont un impact sur la topologie. Les différents objets en mouvements peuvent en effet être considérés soit comme des obstacles ou au contraire comme une aide à la navigation. De ce fait, l'accessibilité ou l'inaccessibilité des zones peuvent donc

être modifiées au cours de la simulation. Ces modifications de la topologie ont un impact sur la navigation des humanoïdes virtuels puisque ces derniers doivent alors considérer les nouvelles opportunités offertes ou encore invalider certains déplacements.

Dans cet article, nous présentons l'algorithme **ToD & DyP** (To*pology* D*etection* and Dy*namic* P*lanner*) dédié à la planification de chemin temps réel dans des environnements dynamiques i.e. dans lesquels la topologie peut varier de manière non prédictible. Cet algorithme se décompose en deux parties. Le rôle de ToD est d'identifier au fil du temps la topologie, c'est à dire les relations d'accessibilité entre les objets composant une scène. DyP exploite alors ces informations pour construire et maintenir à jour une carte de cheminement utilisée pour la planification de chemin. Contrairement à la majorité des algorithmes actuels, où les seuls éléments dynamiques considérés sont des obstacles, ToD & DyP considère que les objets dynamiques peuvent également être utilisés pour aider à la navigation afin d'accéder à de nouvelles zones par exemple.

La suite de cet article est organisée comme suit. Dans un premier temps, nous effectuerons un rapide tour d'horizon des techniques employées pour la représentation des environnements et la planification de chemin. Puis, après une présentation générale de l'algorithme ToD & DyP, nous détaillerons les processus d'identification de la topologie (ToD) et de création / mise à jour des cartes de cheminement (DyP). Enfin nous présenterons quelques résultats et discuterons du modèle et de ses performances.

2. Travaux antérieurs

La planification de chemin et la représentation des environnements statiques ont été très largement étudiées, notamment dans le domaine de la robotique où la navigation est une question essentielle [Lat91] [LaV06] et également dans le domaine de l'animation comportementale. Deux types d'approches sont généralement distinguées pour la planification de chemin : les approches à requêtes multiples, où une unique représentation de l'espace navigable est construite et sera utilisée pour l'ensemble des requêtes effectuées, et les approches à requêtes simples, où chaque requête donne lieu à la construction d'une représentation partielle de l'espace navigable afin de résoudre la requête demandée.

Dans le cas des approches à requêtes multiples, on représente l'environnement dans sa totalité pour ensuite utiliser cette représentation tout au long de la simulation afin de résoudre les différentes requêtes de planification. Les cartes de cheminement ou encore les décompositions en cellules sont parmi les méthodes les plus utilisées. La **décomposition en cellules** consiste à découper l'espace libre en cellules qui sont ensuite utilisées afin de construire un graphe de connectivité de l'environnement. Dans ce graphe, les nœuds représentent les cellules et les arcs traduisent la connectivité entre les cellules. Deux familles de méthodes peuvent être distinguées : la décomposition en cellules approchée et la décomposition en cellules exacte. La première consiste à utiliser des formes prédéfinies (grilles uniformes [Kuf04] [LH04] [SYN01], quadtrees [ST05], cylindres [PLT05]) afin de subdiviser l'environnement et l'union de ces cellules est strictement incluse dans l'espace libre. La décomposition en cellules exacte consiste à calculer un ensemble de cellules dont l'union est strictement égale à l'espace libre (triangulation de Delaunay contrainte [Lam09] [KBT03], polygones convexes). Les **cartes de cheminement** consistent elles à construire un réseau de chemins standardisés (lignes, courbes) à travers l'espace libre de l'environnement. On peut notamment citer les méthodes probabilistes (PRM) qui échantillonnent de manière aléatoire l'espace libre et connectent ensuite ces positions ainsi échantillonnées [BBLA02] [KSLO96] [CLS03]. D'autres algorithmes se basent sur les arêtes du diagramme de Voronoï généralisé afin de créer une carte de cheminement maximisant la distance aux obstacles [SGA*07] [HIKL*99].

Dans le cadre des approches à requête simple, un chemin est trouvé dans l'espace libre à l'instant où la requête est formulée. Aucune connaissance de l'espace libre dans son ensemble n'existe et celui-ci n'est exploré qu'au moment de la requête afin de trouver un chemin entre la position courante et le but fixé. L'approche des **RRT** (*Rapidly-exploring Random Trees*) [LK01] permet de gérer des requêtes simples en connectant la racine de l'arbre d'exploration à la position courante du demandeur et en étendant ensuite l'arbre de recherche jusqu'à atteindre le but demandé. De nombreuses extensions de cet algorithme permettent d'optimiser l'exploration de l'espace libre et de diminuer les temps de calcul. On retrouve notamment les *RRT-Connect* [KL00] qui utilisent deux arbres d'exploration que l'on cherche à connecter ensemble, l'un partant de la position courante et le second du but fixé. L'algorithme *RRF* (*Reconfigurable Random Forest*) [LS02] propose une structure de stockage des arbres précédemment calculés, grâce à un algorithme de type RRT, pour une réutilisation ultérieure.

La plupart des travaux traitant de la planification de chemin sont centrés sur des environnements statiques, c'est à dire dont la topologie ne va pas varier au fil de la simulation. Les environnements dynamiques n'ont pour le moment pas été très étudiés. La notion d'environnement dynamique peut être considérée sous plusieurs aspects. Il peut tout d'abord s'agir d'un environnement contenant des objets ou entités mobiles et jouant le rôle d'obstacle pour les autres entités navigantes. On peut également envisager d'avoir des zones navigables dynamiques dans l'environnement considéré. Les algorithmes actuels permettant la gestion d'environnements dynamiques considèrent uniquement les objets dynamiques comme des obstacles. Ainsi, l'utilisation de cartes de cheminement probabilistes, grâce à des remises à jour locales, permet de naviguer dans un environnement comprenant des obstacles mobiles [vdBFK06]. Certains algorithmes contournent le problème de la dynamique de l'en-

vironnement en divisant la planification globale en une succession de planifications locales. Ainsi, lors de chacune des planifications, l'environnement local est considéré comme statique [LK05] [LK06]. Enfin, Sud et al. proposent deux algorithmes pour la planification de chemin en environnements dynamiques. Le premier, AERO (*Adaptative Elastic ROadmaps*), utilise des forces d'attraction/répulsion afin de déformer le graphe de navigation [SGA*07]. Ces forces sont générées par les obstacles et les agents se déplaçant dans l'espace. Le second algorithme, MaNG (*Multi-agent Navigation Graph*), propose une subdivision de l'environnement à l'aide de diagrammes de Voronoï d'ordres 1 et 2 [SAC*08].

Bien que les environnements dynamiques soient de plus en plus présents dans les mondes virtuels, peu de solutions ont pour le moment été proposées pour faire de la planification de chemin. De nombreux algorithmes existent pour la planification de chemin temps réel en environnements statiques, de nombreux pré-calculs pouvant en général être effectués afin d'alléger le coût calculatoire au cours de la simulation. Dans le cadre d'applications dynamiques interactives, les temps de réponses entre les changements intervenant dans l'environnement et la mise à jour de ce dernier doivent être très courts. Il est donc nécessaire d'utiliser d'autres approches plus efficaces lorsque l'on commence à considérer ce type d'environnement. D'autre part les environnements dynamiques actuellement considérés restent simples en ce concentrant essentiellement sur l'utilisation d'obstacles mobiles. Afin d'avoir des applications interactives permettant de gérer des environnements virtuels dynamiques, au sens de leur changement de topologie, il apparaît donc le besoin de pouvoir représenter les notions d'obstacles mais également d'accessibilités entre les objets dynamiques de notre environnement et enfin de gérer des changements non-prédictibles de la structure de l'environnement en temps réel. **ToD & DyP** propose de nouvelles solutions pour la gestion de tels environnements en temps réel.

3. ToD & DyP : Présentation

ToD & DyP (To*pology* D*etection* and Dy*namic* P*lanner*) est un algorithme pour la planification de chemin en temps réel dans des environnements dynamiques i.e. des environnements dans lesquels la topologie peut varier de manière non prédictible. Le rôle de ToD est d'identifier la topologie, c'est à dire les relations d'accessibilité entre les objets composant une scène. DyP exploite ensuite ces informations pour construire et maintenir à jour une carte de cheminement utilisée pour la planification de chemin. La **Figure 1** présente le schéma de fonctionnement de notre algorithme dont nous allons maintenant voir les principes.

ToD prend en données d'entrée les différents objets composant notre scène. Ces objets géométriques sont considérés dynamiques dans l'algorithme. Afin de détecter la topologie de cette scène, nous augmentons la représentation de ces objets à l'aide de **Volumes d'Interaction**. Ces derniers caractérisent les différentes zones d'influence d'un objet sur les éléments qui l'entoure. Ils permettent ainsi de repérer des relations d'accessibilité et d'obstacle lors de l'analyse des interactions entre nos objets. Le résultat de cette analyse est un **Graphe d'Interaction**, ce graphe représente la topologie globale de l'environnement. ToD va ensuite être en charge du suivi et du maintient de la cohérence de ce graphe au fil de la simulation.

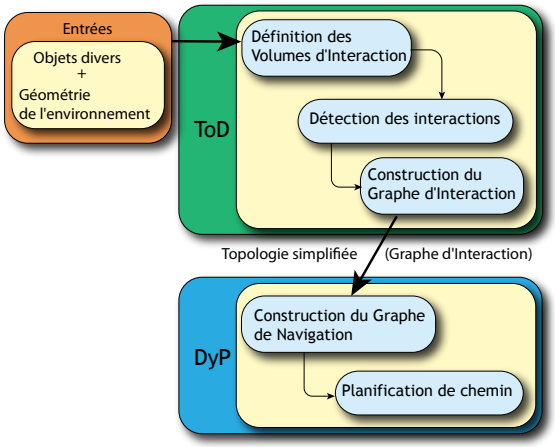

Figure 1: *Schéma de fonctionnement et d'ordonnancement de l'algorithme ToD & DyP*

DyP est ensuite en charge de la construction d'une carte de cheminement et de la planification de chemin. Un Graphe d'Interaction issu de ToD est fourni en entrée. DyP va associer des cartes de cheminement à chaque espace navigable de notre environnement. Il est également chargé de trouver les interconnexions réelles entre les différents objets de notre scène à partir du Graphe d'Interaction. Un **Graphe de Navigation** stocke ensuite ces données et les maintient valides tout au long de la simulation. Ce Graphe de Navigation peut également être perçu comme la carte de cheminement globale de l'environnement. La planification de chemin est ensuite exécutée à l'intérieur de ce Graphe.

ToD & DyP permet donc une représentation de la topologie de notre environnement en temps réel grâce à un découpage de l'algorithme en deux parties distinctes. ToD permet la détection de la topologie et le maintient de la représentation associée et DyP permet de son côté la planification de chemin dans l'environnement ainsi représenté.

4. ToD : Détection de la topologie

ToD (To*pology* D*etection*) est un algorithme de détection de topologie en environnement dynamique. Il caractérise les relations d'accessibilité entre les objets géométriques de la scène et maintient ensuite le Graphe d'Interaction représentant cette topologie. Comme vu précédemment, les ob-

jets que nous considérons peuvent avoir des rôles d'obstacle mais également aider à la navigation des humanoïdes. La topologie de notre environnement va donc dépendre des accessibilités induites par ces objets. Nous allons maintenant décrire les trois parties de ToD : Définition des Volumes d'Interaction, Détection des relations existantes et Construction du Graphe d'Interaction.

4.1. Définition des Volumes d'Interaction

Nous représentons deux types de relation à l'intérieur de notre environnement. Tout d'abord des relations d'obstacle (*"Cette caisse posée au sol au milieu du passage empêche-t-elle mon déplacement ?"*), puis les relations d'accessibilité entre les différents espaces navigables de l'environnement (*"Est-il possible de monter sur cette caisse posée au sol pour accéder à un autre endroit ?"*). Chaque objet peut présenter des zones sur lesquelles il est possible de naviguer (le dessus d'une caisse en bois pour des humanoïdes par exemple), nous appelons ces zones particulières de l'environnement des **Zones Navigables**. La Zone Navigable associée à un objet O sera notée par la suite $Zn(O)$.

La détection des relations (accessibilité ou obstacle) entre les objets de notre environnement repose sur le concept de **Volumes d'Interaction**, noté $V(O)$. Ces Volumes d'Interaction peuvent être de deux types : **Volumes d'Accessibilité** et **Volumes d'Interdiction**. Un Volume d'Accessibilité caractérise l'ensemble des points de l'environnement qui sont accessibles depuis une Zone Navigable. Un Volume d'Intersection quant à lui représente l'ensemble des points de l'environnement interdits à la navigation à cause de la gêne occasionnée par la présence d'un objet. Ces volumes associés à un objet O seront respectivement notés $V_A(O)$ et $V_I(O)$. Un objet O est donc caractérisé par une Zone Navigable, $Zn(O)$, un Volume d'Accessibilité associé à cette Zone Navigable, $V_A(O)$, et un Volume d'Interdiction associé à l'objet lui-même, $V_I(O)$.

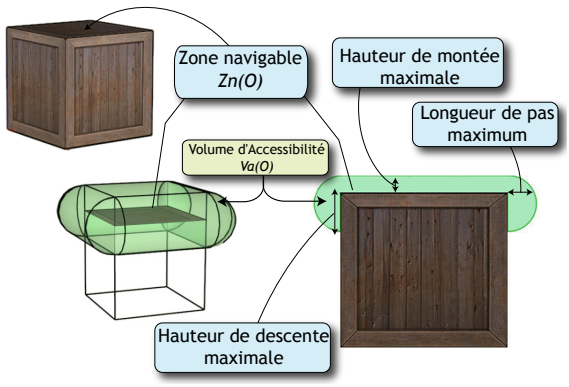

Figure 2: *Définition du Volume d'Accessibilité*

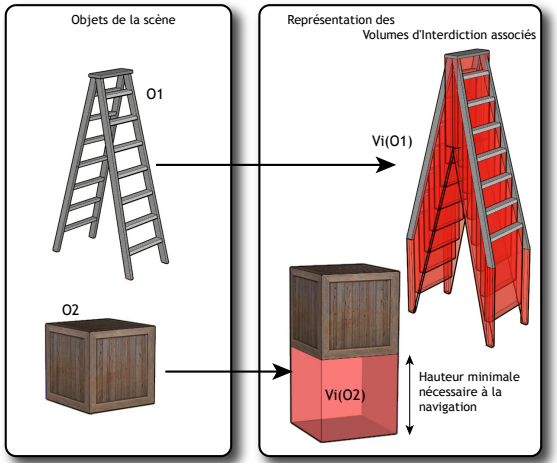

Figure 3: *Définition du Volume d'Interdiction*

La définition de la forme des Volumes d'Interaction repose sur les capacités des entités navigantes peuplant notre environnement. Dans notre cas, nous considérons des humanoïdes. Les notions d'accessibilité et la notion d'obstacle sont étroitement liées aux caractéristiques de ces humanoïdes. Ainsi, nous définissons la forme des Volumes d'Interaction en se basant sur ces capacités. Trois critères nous ont parus essentiels pour définir l'accessibilité entre deux zones distinctes (cf. **Figure 2**) : la longueur des enjambées de l'humanoïde, la hauteur maximale de ses pas lorsqu'il monte une marche et lorsqu'il en descend une. La hauteur de pas permet de définir un volume au dessus de la Zone Navigable, représentant ainsi les points accessibles par l'humanoïde quand il monte par exemple une marche. La combinaison des trois paramètres permet de définir une sorte de cylindre déformé autour des bordures de la Zone Navigable. L'utilisation de deux hauteurs distinctes pour la montée et la descente est volontaire car il est en effet plus aisé de descendre une marche haute que de la monter. Cela a pour conséquence de créer des relations d'accessibilité qui ne seront pas bijectives entre les Zones Navigables. Afin de caractériser la notion d'obstacle, nous n'utilisons qu'un seul paramètre correspondant à la hauteur minimale nécessaire pour la navigation de l'humanoïde. Un volume est donc défini à la verticale de l'objet (par translation de l'objet selon l'axe vertical, cf. **Figure 3**) et correspond à l'ensemble des points où il est impossible pour l'humanoïde de se tenir.

4.2. Détection des interactions

La création de Zones Navigables et de Volumes d'Interaction permet la définition de différentes relations. Si une Zone Navigable $Zn(O1)$ entre en intersection avec un Volume d'Interdiction $V_I(O2)$, il en découle que $Zn(O1)$ va être en partie obstruée par l'objet et donc toute navigation sera impossible à cet endroit. Pour deux objets $O1$ et

$O2$, la relation d'obstruction sera donc caractérisée par : $Zn(O1) \cap V_I(O2) \neq \emptyset$. De manière comparable, si une intersection est détectée entre une Zone Navigable $Zn(O2)$ et un Volume d'Accessibilité $V_A(O1)$ cela signifie une accessibilité de $O1$ vers $O2$. Cette accessibilité est donc définie par : $V_A(O1) \cap Zn(O2) \neq \emptyset$. La représentation de notre topologie va donc dépendre de la détection de ces intersections.

La gestion d'interpénétration entre des volumes fait partie du domaine bien connu qu'est la détection de collisions. Les moteurs de détection de collisions ont pour rôle de repérer des collisions ou interpénétrations entre les objets dans tous types de monde virtuel. Ces algorithmes de détection de collisions sont optimisés et ont fait l'objet de nombreuses contributions [LG98] [KHI*07], notamment dans le domaine de la réalité virtuelle.

Nous proposons donc l'utilisation de tels moteur de détection de collisions afin d'évaluer notre topologie. La détection des relations existant dans notre environnement se ramène donc à une simple détection de collisions entre les Volumes d'Interaction définis dans notre scène. Il s'agit ensuite de filtrer les collisions détectées et de conserver celles qui sont pertinentes quant à la topologie :

- Accessibilité de $O1$ vers $O2$: Collision entre un Volume d'Accessibilité $V_A(O1)$ et une Zone Navigable $Zn(O2)$;
- Obstruction de $O1$ sur $O2$: Collision entre un Volume d'Interdiction $V_I(O1)$ et une Zone Navigable $Zn(O2)$;
- Potentielle obstruction de $O1$ sur $O2$ lors d'une changement de Zone Navigable : Collision entre un Volume d'Interdiction $V_I(O1)$ et un Volume d'Accessibilité $V_A(O2)$.

La ré-utilisation des algorithmes de détection de collisions possède plusieurs bonnes propriétés. Tout d'abord, les performances de ces moteurs permettent à ToD une détection des relations topologiques en temps réel. Cette gestion temps réel permet ainsi une simulation interactive où l'utilisateur va pouvoir modifier la structure de l'environnement à son gré. D'autre part, ces algorithmes sont utilisés par les moteurs physiques qui sont dorénavant intégrés dans de nombreux environnements virtuels. Il sera donc aisément possible d'étendre l'application de notre algorithme à des environnements utilisant ces moteurs physiques.

4.3. Graphe d'Interaction

Afin de stocker les informations fournies par la détection de collision entre les Volumes d'Interaction, nous utilisons un **Graphe d'Interaction**. Ce graphe est une structure représentant les relations existant entre les différents objets de l'environnement.

Un objet O de notre environnement est représenté dans le Graphe d'Interaction par un nœud (cf. **Figure 4**). Une relation entre deux objets $O1$ et $O2$ est représentée sous la forme d'un arc. Les arcs ainsi créés sont étiquetés en fonction de la nature de cette relation. Ainsi, la détection d'une relation d'accessibilité de $O1$ vers $O2$ ($V_A(O1) \cap Zn(O2) \neq \emptyset$) entraîne la création d'un arc de type *accessible* entre les nœuds associés. Puisque nos relations entre les zones d'accessibilités ne sont pas bijectives, les arcs de notre graphe sont donc orientés afin de représenter ces connexions topologiques. La détection d'une interdiction de $O2$ sur $O1$ ($Zn(O1) \cap V_I(O2) \neq \emptyset$) entraîne un arc de type *interdiction* entre $O1$ et $O2$ et d'un arc de type *obstacle* entre $O2$ et $O1$.

La représentation de la topologie par ce graphe est une représentation "optimiste" i.e. un lien d'accessibilité entre deux nœuds ne représente pas une accessibilité *certaine* entre les deux Zones Navigables de ces objets mais une accessibilité *potentielle*. En effet, cette accessibilité pourra être remise en question à partir du moment où l'on considère également les relations d'obstacle présentes dans l'environnement.

Le Graphe d'Interaction ainsi défini nous permet également d'avoir une localité des informations représentées. En effet, les liens représentent des relations entre deux objets $O1$ et $O2$, ces relations étant de plus décomposées en relation d'accessibilité et d'interdiction. Ainsi, la modification de l'état d'un objet dans notre environnement (position, suppression,...) aura une influence très localisée au niveau du graphe et seuls les liens en relation avec cet objet seront à remettre en question. Ce Graphe d'Interaction nous permet donc une identification et une localisation des changements apparus dans l'environnement.

4.4. ToD : Conclusion

Nous venons donc de voir les principes de l'algorithme ToD. Partant d'un environnement dynamique, nous caractérisons donc des Volumes d'Interaction qui, à l'aide d'un moteur de détection de collisions, détectent les relations d'accessibilité ou d'obstacle entre les objets dynamiques de notre scène. Dans un contexte d'environnement utilisant un moteur physique, l'utilisation des Volumes d'Interaction associés aux objets permet de ré-exploiter une partie des calculs déjà effectués par les moteurs de détection de collisions. Bien que surchargeant les calculs initiaux, cela permet un maintient temps réel du Graphe d'Interaction.

Toutefois, la topologie ainsi représentée n'est qu'une estimation de la topologie réelle. Par exemple, il n'est pas possible de savoir directement par la lecture du Graphe d'Interaction si une Zone Navigable $Zn(O1)$ partiellement interdite par les Volumes d'Interdiction de deux objets, $V_I(O2)$ et $V_I(O3)$, est entièrement interdite ou si l'on peut tout de même y naviguer, comme le montre la **Figure 5**. Nous basant sur le Graphe d'Interaction, nous allons raffiner les informations d'accessibilité contenues afin de construire une carte de cheminement représentant la topologie réelle de notre environnement. Ce raffinement, ou filtrage, est effectué à l'aide de l'algorithme DyP.

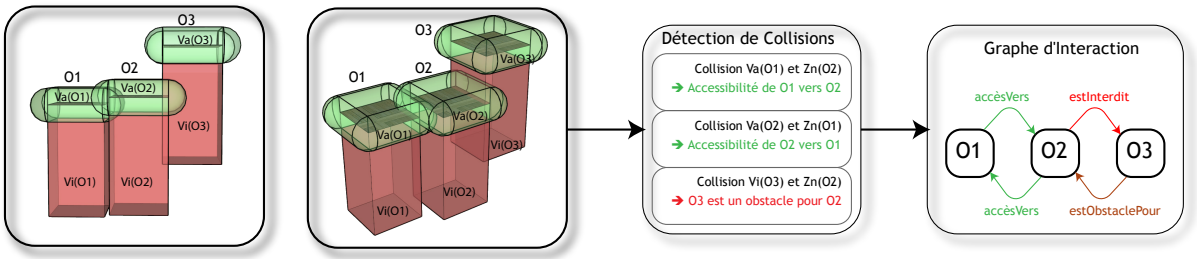

Figure 4: *Construction du Graphe d'Interaction à partir de la détection de collisions entre les Volumes*

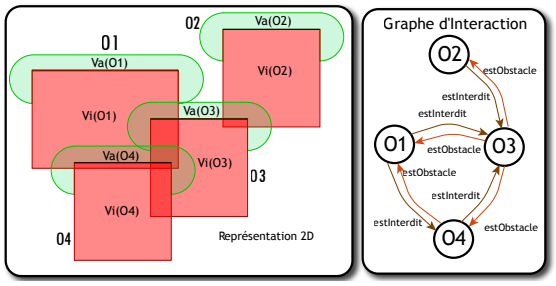

Figure 5: *Certaines représentation dans le Graphe d'Interaction sont trop ambigües pour déterminer si il est possible de naviguer. Dans l'exemple ci-dessus, la $Zn(O3)$ est partiellement interdite par deux Volumes d'Interdiction, $V_I(O1)$ et $V_I(O2)$, mais reste navigable. $Zn(O4)$ est également interdite par deux Volumes d'Interdiction, $V_I(O1)$ et $V_I(O3)$, mais n'est pas navigable. Au niveau du Graphe d'Interaction, ces deux configurations sont représentées de la même manière, il va donc falloir résoudre ces deux cas.*

5. DyP : Navigation dans l'environnement virtuel

ToD produit un Graphe d'Interaction dans lequel sont regroupés des informations relatives à la topologie de notre environnement. Ce sur-ensemble de la topologie rend compte des relations d'accessibilité entre les différents objets de notre scène mais également des obstructions présentes. Ce graphe permet également d'identifier des structures topologiques ambigües mais ne permet cependant pas de lever ces ambiguïtés. DyP, par l'intermédiaire des résultats fournis par ToD, introduit une structure appelée **Graphe de Navigation** qui caractérise la topologie réelle de notre environnement en se basant sur la construction de cartes de cheminement. Pour ce faire, nous associons des cartes de cheminement aux objets puis nous créons une carte de cheminement globale au niveau de l'environnement et la stockons dans le Graphe de Navigation.

5.1. Cartes de cheminement associées aux objets

Les objets utilisés dans les environnements dynamiques vont généralement posséder des Zones Navigables auxquelles sont associées des cartes de cheminement. Ces cartes de cheminement, grâce aux informations apportées par le Graphe d'Interaction, sont filtrées afin de considérer les zones contenant des obstacles. Ainsi, les notions d'inaccessibilités dues à la présence d'éléments obstruant le passage sont directement prises en compte au niveau de ces cartes de cheminement.

Afin de stocker les cartes de cheminement associées aux objets de notre scène, nous introduisons une nouvelle structure appelée Graphe de Navigation. Les nœuds de ce graphe représentent, comme dans le Graphe d'Interaction, les différents objets composant notre scène auxquels sont associés les cartes de cheminement.

5.2. Carte de cheminement associée à l'environnement

Afin de planifier un chemin dans l'environnement, il faut ensuite interconnecter les différentes Zones Navigables les unes aux autres. Connaissant les cartes de cheminement locales aux objets, il s'agit alors de les lier entre elles dans le Graphe de Navigation. Ces connexions sont réalisées à l'aide du Graphe d'Interaction qui permet de connaître les relations d'accessibilités existant dans l'environnement :

- Si une relation d'accessibilité existe au niveau du Graphe d'Interaction, nous tentons de connecter les 2 cartes de cheminements dans le Graphe de Navigation.
- Si un Volume d'Interdiction est en interaction avec un Volume d'Accessibilité, nous effectuons un filtrage afin de vérifier la validité du lien d'accessibilité.

5.3. DyP : Conclusion

La construction du Graphe de Navigation est donc basée sur la structure du Graphe d'Interaction fourni par ToD. Ce Graphe de Navigation est en fait une carte de cheminement à deux niveaux de détail. Le premier niveau correspond à une carte de cheminement entre les Zones Navigables des différents objets, cette carte de cheminement est représentée par les arcs du Graphe de Navigation. Le second niveau

correspond à des cartes de cheminement locales associées directement aux objets de la simulation. Ainis, dans un premier temps, la planification de chemin permet de trouver quelles Zones Navigables doivent être traversées afin d'atteindre l'objectif fixé et dans un second temps une planification locale est effectuée à l'intérieur de chaque Zone Navigable afin de trouver le chemin exact à emprunter.

Baser notre Graphe de Navigation sur le Graphe d'Interaction présente plusieurs avantages. Tout d'abord, la propriété de localité du Graphe d'Interaction est également retrouvée dans le Graphe de Navigation : un changement local de la topologie de l'environnement entraîne seulement une remise à jour locale des cartes de cheminement. Toutefois, la séparation de ces deux graphes est nécessaire puisque notre topologie doit être remise à jour constamment tandis qu'au niveau de notre Graphe de Navigation nous pouvons la remettre à jour en fonction du contexte et des requêtes formulées par les humanoïdes. La construction de ce Graphe de Navigation et des cartes de cheminement associées n'a pour le moment pas été implémentée. Des choix de représentation des cartes de cheminement doit être encore effectués afin de réaliser cette mise en œuvre.

6. Résultats

Un prototype de ToD a été implémenté en utilisant la détection de collision fournie par la librairie Bullet Physics[†]. La scène de test est composée d'un environnement statique dans lequel divers objets peuvent être ajoutés, supprimés ou déplacés par l'utilisateur. Les objets utilisés dans la simulation sont des caisses de tailles variables ainsi que des escabeaux (cf. Figure 6).

L'algorithme ToD a été testé sur un Intel(R) Core(TM)2 Extreme, CPU X7900 à 2,80GHz, avec une carte graphique NVidia Quadro FX 3600M. Les hauteurs de pas à la montée et à la descente ont été fixées aux mêmes valeurs (30cm) ainsi que la longueur de pas maximale. La hauteur navigable minimum a été paramétrée à 1m. La représentation des Volumes d'Interactivité pour les caisses est assez simple : un Volume d'Accessibilité, un Volume d'Interdiction et une Zone Navigable. La représentation des escabeaux est plus complexe, un Volume d'Accessibilité plus un Volume d'Interdiction étant associés à chaque marche. Notre escabeau comportant un total de 15 marches, il y a donc environ 30 Volumes d'Interaction qui y sont associés auxquels il faut ajouter les 15 Zones Navigables correspondantes. Les performances de ToD lors des différents tests sont décrites dans la Table 1.

Au cours de l'exécution de ToD, 85% du temps de calcul est dédié à la détection des collisions. La mise à jour du Graphe d'Interaction prend elle les 15% restant. Il faut toutefois noter que Bullet n'est pas la librairie offrant le moteur de détection de collision le plus performant et qu'il faudra comparer les performances avec d'autres moteurs. D'autre part, les continuités spatiales et temporelles de la simulation ne sont pas prises en compte actuellement et les objets statiques sont donc pour le moment considérés comme des objets dynamiques. De ce côté, les performances pourraient également être améliorées. Enfin, la visualisation du Graphe d'Interaction créé est également possible, les liens d'accessibilité et/ou d'obstacle entre les objets sont en effet visualisable et permettent d'avoir une bonne vision du principe de localité qui est apporté par le Graphe d'Interaction (cf. Figure 7).

Objets dans la scène		Bullet	ToD : Exécution	
Caisses	Escabeaux	Objets	Durée	FPS
50	0	150	7,9 ms	60
50	10	600	22 ms	20
0	20	900	36 ms	15

Table 1: *Performances de ToD lors de différents tests.*

7. Conclusion

Dans cet article, nous avons présenté l'algorithme ToD & DyP dédié à la planification de chemin en temps réel dans des environnements dynamiques. Cet algorithme repose sur la détermination et le suivi de l'évolution des relations topologiques (ToD) afin de construire / mettre à jour une carte de cheminement (DyP) permettant la planification de chemin en environnement dynamique. ToD & DyP a été conçu pour le temps réel pour pouvoir être utilisé dans des applications interactives. A contrario des algorithmes usuels, ToD & DyP ne considère pas les objets en mouvement uniquement comme des obstacles mais cherche également à caractériser les nouvelles possibilités de déplacement offertes par ces derniers. Les Volumes d'Interaction et plus particulièrement les Volumes d'Accessibilité associés aux objets permettent de décrire différentes possibilités de déplacement pour des humanoïdes comme la marche mais aussi les sauts. La définition de ces volumes peut être liée aux capacités de déplacement de chaque humanoïde et permet donc de prendre en compte différentes morphologies et aptitudes de déplacement. Pour sa part, le Graphe d'Interaction permet de localiser les modifications effectuées dans l'environnement et identifie de manière pertinente les lieux où les cartes de cheminement doivent être remises à jour par DyP.

À court terme, nos travaux vont se focaliser sur DyP afin de générer / mettre à jour rapidement les cartes de cheminement en exploitant au mieux l'évolution et / ou les changements de topologie identifiés par ToD. Dans un second temps, les travaux se concentreront sur l'animation d'humanoïdes de synthèse et l'adaptation des postures en cours de navigation. Ces travaux seront couplés à une analyse du comportement humain afin de (1) déterminer au mieux la

[†] http ://www.bulletphysics.com/

forme des Volumes d'Accessibilité à partir des capacités de déplacement mesurées et (2) obtenir des animations réalistes lors des déplacements.

Références

[BBLA02] BURCHAN BAYAZIT O., LIEN J.-M., AMATO N. M. : Roadmap-based flocking for complex environments. In *Proc. 10th Pacific Conference on Computer Graphics and Applications* (9–11 Oct. 2002), pp. 104–113.

[CLS03] CHOI M. G., LEE J., SHIN S. Y. : Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Trans. Graph.. Vol. 22*, Num. 2 (2003), 182–203.

[HIKL*99] HOFF III K., KEYSER J., LIN M., MANOCHA D., CULVER T. : Fast computation of generalized voronoi diagrams using graphics hardware. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), ACM Press/Addison-Wesley Publishing Co. New York, NY, USA, pp. 277–286.

[KBT03] KALLMANN M., BIERI H., THALMANN D. : Fully dynamic constrained delaunay triangulations. *Geometric Modelling for Scientific Visualization. Vol. 3* (2003).

[KHI*07] KOCKARA S., HALIC T., IQBAL K., BAYRAK C., ROWE R. : Collision detection : A survey. *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on* (Oct. 2007), 4046–4051.

[KL00] KUFFNER J. J. J., LAVALLE S. M. : Rrt-connect : An efficient approach to single-query path planning. In *Proc. IEEE International Conference on Robotics and Automation ICRA '00* (24–28 April 2000), vol. 2, pp. 995–1001.

[KSLO96] KAVRAKI L. E., SVESTKA P., LATOMBE J. C., OVERMARS M. H. : Probabilistic roadmaps for path planning in high-dimensional configuration spaces. 566–580.

[Kuf04] KUFFNER J. J. : Efficient optimal search of euclidean-cost grids and lattices.

[Lam09] LAMARCHE F. : Topoplan : a topological path planner for real time human navigation under floor and ceiling constraints. In *Eurographics* (2009).

[Lat91] LATOMBE J.-C. : *Robot motion planning*. Boston : Kluwer Academic Publishers, Boston, 1991.

[LaV06] LAVALLE S. : *Planning Algorithms*. Cambridge University Press, 2006.

[LG98] LIN M. C., GOTTSCHALK S. : Collision detection between geometric models : a survey. In *Proceedings of the 8th IMA Conference on the Mathematics of Surfaces (IMA-98)* (Winchester, UK, septembre 1998), Cripps R., (Ed.), vol. VIII de *Mathematics of Surfaces*, Information Geometers, pp. 37–56.

[LH04] LI T.-Y., HUANG P.-Z. : Planning humanoid motions with striding ability in a virtual environment. In *Proc. IEEE International Conference on Robotics and Automation ICRA '04* (Apr 26–May 1, 2004), vol. 4, pp. 3195–3200.

[LK01] LAVALLE S., KUFFNER J. : Rapidly-exploring random trees : Progress and prospects. 293.

[LK05] LAU M., KUFFNER J. J. : Behavior planning for character animation. In *SCA '05 : Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM, pp. 271–280.

[LK06] LAU M., KUFFNER J. J. : Precomputed search trees : planning for interactive goal-driven animation. In *SCA '06 : Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2006), Eurographics Association, pp. 299–308.

[LS02] LI T., SHIE Y. : An incremental learning approach to motion planning with roadmap management. In *IEEE International Conference on Robotics and Automation, 2002. Proceedings. ICRA'02* (2002), vol. 4.

[PLT05] PETTRE J., LAUMOND J., THALMANN D. : A navigation graph for real-time crowd animation on multilayered and uneven terrain. In *First International Workshop on Crowd Simulation* (2005).

[SAC*08] SUD A., ANDERSEN E., CURTIS S., LIN M. C., MANOCHA D. : Real-time path planning in dynamic virtual environments using multiagent navigation graphs. 526–538.

[SGA*07] SUD A., GAYLE R., ANDERSEN E., GUY S., LIN M., MANOCHA D. : Real-time navigation of independent agents using adaptive roadmaps. In *VRST '07 : Proceedings of the 2007 ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2007), ACM, pp. 99–106.

[ST05] SHAO W., TERZOPOULOS D. : Environmental modeling for autonomous virtual pedestrians. In *SAE Symposium on Digital Human Modeling for Design and Engineering* (2005), pp. 1–8.

[SYN01] SHILLER Z., YAMANE K., NAKAMURA Y. : Planning motion patterns of human figures using a multilayered grid and the dynamics filter. In *Proc. ICRA Robotics and Automation IEEE International Conference on* (2001), vol. 1, pp. 1–8.

[vdBFK06] VAN DEN BERG J., FERGUSON D., KUFFNER J. : Anytime path planning and replanning in dynamic environments. In *Proc. IEEE International Conference on Robotics and Automation ICRA 2006* (May 15–19, 2006), pp. 2366–2371.

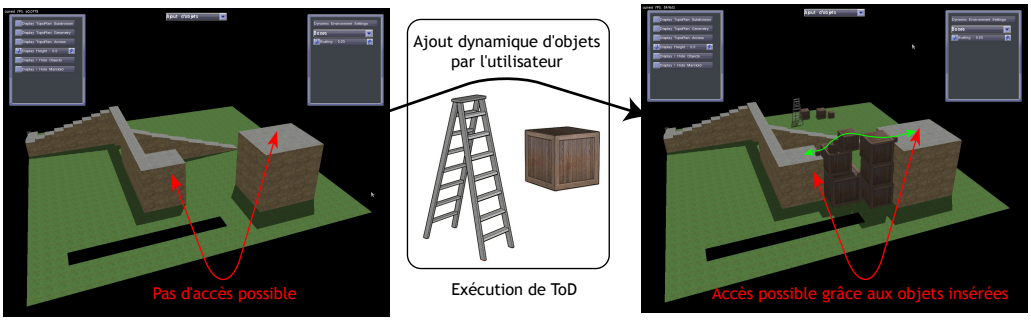

Figure 6: *Environnement de test : insertion d'objets divers afin de modifier dynamiquement l'environnement. Les objets insérés sont des caisses et des escabeaux.*

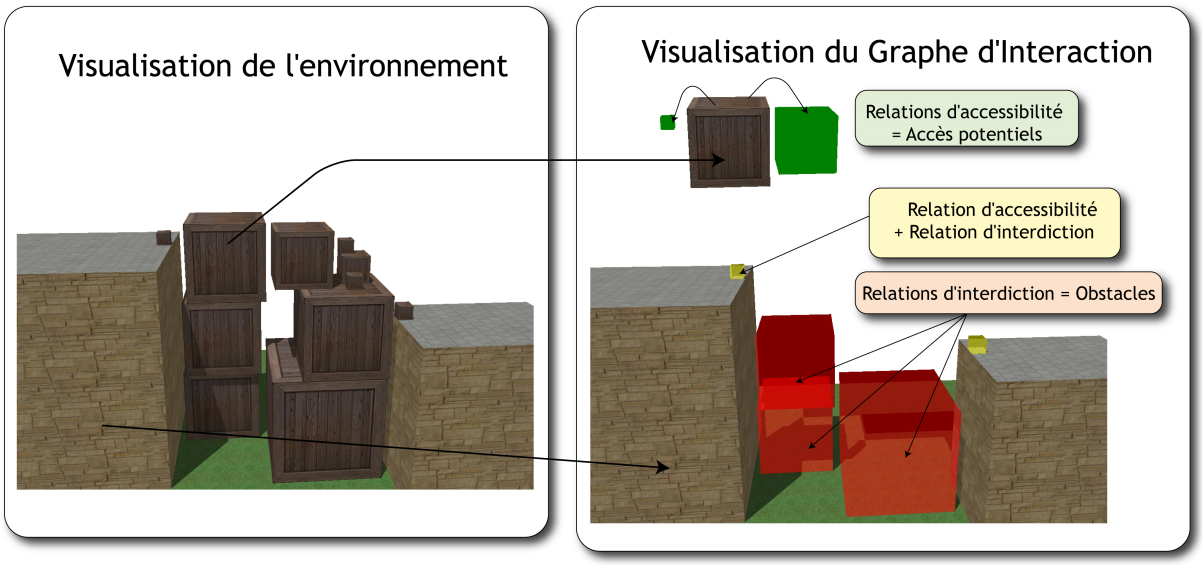

Figure 7: *Environnement de test : Visualisation de l'environnement de test et d'une partie Graphe d'Interaction correspondant.*