DEA Algorithmique

Mémoire de stage

Florent Duguet

Squelette de Visibilité Robuste

Directeur de stage : George DRETTAKIS

Filière : Géométrie Algorithmique

Directeur de la filière : M. Jean-Daniel BOISSONNAT

Directeur du DEA Algorithmique : M. Jean-Marc STEYAERT
Correspondant à l'ENST Paris : M. Francis SCHMITT

Organisme d'accueil :

Action REVES
INRIA Sophia-Antipolis
2004 route des lucioles
BP 93
FR-06902 Sophia Antipolis

Dates du stage : du 1 avril au 19 septembre 2001

17 septembre 2001

Table des matières

1 Introduction images de synthèse

Les images de synthèse occupent une place de plus en plus grande dans notre environnnement. Elles sont partout, jeux vidéo, cinéma, publicité, conception virtuelle (design/CAO), même là où on les attend le moins : musées avec visites virtuelles interactives.

Cette fuite en avant pour la production et l'utilisation des images de synthèse a plusieurs motivations. La première qui nous vient à l'esprit est la création d'effets spéciaux, car elle touche directement le grand public. Ces effets spéciaux sont partie intégrante de spots publicitaires, ou de films. De plus en plus des films entiers sont réalisés en images de synthèse. Ils sont la vitrine du domaine de l'infographie et de l'animation, et le principal moteur de son développement. L'utilisation des images de synthèse permet toutes les fantaisies aux réalisateurs et donne aux spectateurs une forte impression de réalité à ces images ; cette impression de réalité est désormais appelée réalité virtuelle.

L'architecture et l'urbanisme sont aussi de grands consommateurs d'images de synthèse ; il va sans dire que la possibilité d'observer un bâtiment dans son environnement avant même d'avoir commencé l'ouvrage est un luxe très appréciable.

Le milieu médical utilise de plus en plus les images de synthèse pour analyser des données reccueillies sur des patients à l'aide de scanners, ou autre appareils non intrusifs. De plus, en collaboration avec la robotique, la réalité virtuelle permet de proposer des simulations d'opérations et faire ainsi progresser la pratique de la médecine.

L'industrie des jeux vidéo est un tremplin pour les images de synthèse ; toujours plus réalistes, ils sont par essence consommateurs de réalité virtuelle. L'imagination des joueurs peut évoluer dans des mondes purement virtuels avec un réalisme à la carte.

Finalement, le design et la CAO sont de grands consommateurs d'images de sythèse. Ils occupent une place de plus en plus importante dans la phase de conception des produits de consommation ; mais aussi pour le commerce électronique en plein essort.

Images de synthèse

La génération d'images de synthèse se compose de plusieurs étapes relativement indépendantes. En voici les grandes lignes :

- Modélisation / Animation
- Simulation / Eclairage
- Rendu

La Modélisation

La première phase consiste à exprimer à la machine le monde virtuel que l'on souhaite construire. Cette phase est relativement indépendante des autres quoique dépendante du résultat escompté. Selon le résultat que l'on souhaite obtenir, cette phase de *modélisation* sera cruciale, ou moins importante.

Beaucoup de travaux en géométrie sont consacrés à la modélisation. La modélisation de mondes virtuels, consistant en la description de scènes pour la machine, est une partie

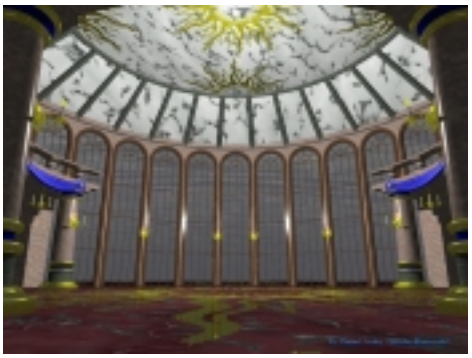


image obtenue par lancé de rayons - site www.povray.org/ - auteur : Daniel Corley <dcorley@umr.edu>

image obtenue par radiosité - Projet ARCADE - Aircraft model data - courtesy of Lightwork Design, UK.

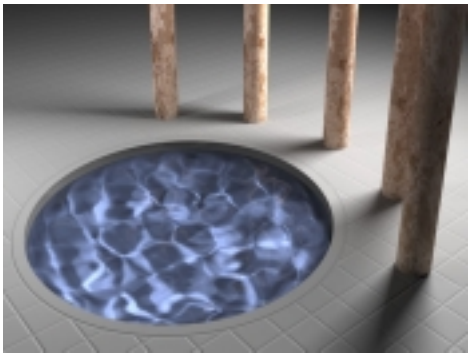


image obtenue par Métropolis - tirée de la thèse de Eric Veach - "Robust Monte Carlo Methods for Light Transport Simulation", Ph.D. dissertation, Stanford University, December 1997

image obtenue par lancé de particules - tirée de l'article de Henrik Wann Jensen and Per H. Christensen : "Efficient Simulation of Light Transport in Scenes with Participating Media using Photon Maps". SIGGRAPH'98

FIG. 1 – Quelques images obtenues à l'aide des différentes techniques d'éclairage

à forte connotation géométrique. Dans cette phase, on exprime l'environnement que l'on souhaite étudier à l'aide de modèles géométriques.

Les techniques les plus utilisées de nos jours, bien qu'en évolution actuellement, sont les modélisations à base de polygones. Les polygones reignent en maîtres dans le monde du graphique et la plupart des modèles utilisés sont faits à l'aide de polygones. D'autres modélisations existent, mais leur manipulation est plus difficile dans certains cas si bien que leur utilisation est restreinte à des situations dédiées. Citons les modélisations à base des primitives graphiques évoluées comme les splines et surfaces implicites, sans oublier bien sûr l'engouement récent pour les surfaces de subdivision.

Bien que passionnant, nous ne rentrerons pas dans le domaine de la modélisation ici, et nous nous satisferons de modèles de scènes polygonaux, très répendus, et souvent dénominateur commun de toute modélisation à une approximation près.

Simulation - Eclairage

La seconde étape relève plutôt du monde de la physique. La simulation de l'éclairage n'est autre qu'une simulation physique (optique plus exactement) des échanges lumineux, c'est à dire, l'étude des ondes électro-magnétiques dans la scène. Au fil des années et des progrès technologiques des ordinateurs, les techniques de simulation ont sans cesse évolué.

Partant de méthodes approximatives et empiriques, les techniques de simulation de l'éclairage se sont de plus en plus approchées d'une modélisation physique des phénomènes électro-magnétiques. Les grandes étapes de cette évolution sont les suivantes :

- Modèle d'éclairage local (Lambertien, Gouraud, Phong)
- Lancé de rayons
- Radiosité
- Tracé de chemins
- Tracé de particules

Chacune de ces modélisations a bien sûr son lot de points forts et de points faibles. Sans entrer dans les détails, essayons de résumer : les modèles d'éclairage locaux sont extrêmement rapides puisque la quantité d'information traitée simultanément est très faible (modèle local), mais ce au détriment du réalisme. Les méthodes de simulation globales de l'éclairage sont basées sur l'équation du rendu (*rendering equation*). Elles correspondent à une intégration de cette équation en effectuant différentes approximations.

Le lancé de rayons, tel qu'exposé initialement est un modèle partiellement global, plus réaliste ; il combine à la fois simulation et rendu, ce qui rend son utilisation moins flexible : la simulation dépend du point de vue. La radiosité modélise très bien les objets mats (diffus), elle est basée sur un modèle physique ; de plus le résultat de la simulation est indépendant du point de vue, ce qui le rend plus intéressant car une même simulation peut être utilisée pour des balades virtuelles. Hormis les problèmes numériques, elle souffre de ne pouvoir représenter tout une catégorie d'objets : les objets non lambertiens, comme les objets réfléchissant partiellement ou entièrement. Le tracé de chemin est une extension du lancé de rayons consistant à calculer des chemins entre un point et les sources lumineuses ; cette technique a évolué pour donner naissance à la technique *Metropolis*, méthode probabiliste basée sur la mutation des chemins vers un état stationnaire. Enfin le tracé de particules semble être une méthode très performante, elle permet de modéliser tout type d'échange lumineux, et tout type d'objet ; très à la mode, elle est de plus en plus utilisée, en particulier sous la forme de *photon map*.

Le dénominateur commun de toutes ces méthodes globales reste l'équation du rendu dont un des éléments est la fonction de visibilité. La fonction de visibilité de l'équation du rendu est booléenne et répond à la question x *voit* y *?* Selon l'approximation utilisée pour la simulation de l'éclairage, cette requête est plus ou moins difficile à reformuler ; en effet pour les méthodes basées sur le lancé de rayons, la requête reste booléenne : x *voit* y *?* en revanche pour la radiosité la requête devient *le polygone* P *voit-il le polygone* Q *?* et la réponse n'est plus booléenne, elle est intégrée dans les calculs d'éclairage dans un terme géométrique, le facteur de formes.

Dans chacune de ces méthodes, la visibilité est un élément primordial et il s'avère

dans la pratique que près de 90% du temps de calcul est passé à calculer cette fameuse fonction de visibilité : le lancé de rayons est par essence un calcul de visibilité !

Rendu

La dernière étape consiste à transformer le résultat de la simulation en une image ou en une séquence d'images, c'est le rendu. Cette phase est déjà faite dans le cas du lancé de rayons, mais reste à faire pour les autres. Les techniques de rendu sont variables, allant de l'affichage de polygones par projection au lancé de rayons (sur la scène déjà éclairée).

Les principales difficultés du rendu sont liées à la précision (ou imprécision) des ordinateurs, ainsi qu'aux problèmes d'échantillonnage. Le monde informatique étant un monde discret (au sens mathématique bien sûr), calculer une image nécessite méthodes et astuces.

Visibilité

Dans chacune des étapes de cette *chaîne de production* des images de synthèse, nous avons vu la nécessité de développer de nouvelles techniques, et autres astuces afin d'obtenir un résultat satisfaisant. Pour la simulation de l'éclairage global, ainsi que pour le rendu, nous effectuons des calculs de visibilité. Par exemple, un modèle local d'éclairage ne prend pas en compte les ombres, leur calcul nécessite une extension à la méthode ; toute l'information de la scène est nécessaire pour ce calcul. Pour le rendu, l'occultation d'objets (objet en cachant un autre totalement ou partiellement) est aussi une étape nécessitant des calculs de visibilité. Nous allons voir que la visibilité occupe une place prépondérante dans la simulation de l'éclairage ainsi que dans le rendu, mais aussi dans d'autres domaines, comme la robotique.

Nous allons étudier une technique de calcul de visibilité pour une scène modélisée à l'aide de polygones. Ce choix est certes restrictif mais comme nous l'avons vu, il reste le dénominateur commun à beaucoup de modélisations : on peut mailler (transformer en polygones) la plupart des modèles.

La visibilité est un problème difficile. On veut extraire l'information de visibilité d'une scène afin de pouvoir y effectuer des requêtes. Cette information de visibilité se doit d'être globale : un objet proche peut être caché, alors que l'on peut voir des objets très éloignés, il faut donc prendre en compte toute la scène pour chaque calcul. De plus, cette information de visibilité doit être extrêmement riche, en effet, il incombe à la visibilité de répondre à des requêtes comme *voit-on cet objet depuis ce point de vue ?* et ce pour n'importe quel objet et n'importe quel point de vue.

Cependant l'information de visibilité dans son intégralité ne nous intéresse pas forcément, ce qui nous intéresse, c'est précisément les réponses aux requêtes que l'on peut formuler. Ces requêtes seront dans le cas de la simulation de l'éclairage liées à l'équation du rendu et à sa fonction de visibilité, alors que dans le cas du rendu, elles concerneront le point de vue de la caméra. La visibilité d'une scène va nous intéresser

d’un point de vue utilisateur, c’est à dire comme un outil, nous allons donc calculer la partie d’information dont on aura besoin.

Par exemple, dans un modèle d’éclairage global, on souhaitera par exemple savoir si un point de la scène *voit* une source de lumière, c’est à dire s’il existe un chemin direct pour la lumière entre la source et le point. Cette information nous permettra en effet de calculer les ombres dans la scène. Ou encore, pour le rendu, savoir si un objet est caché par un autre afin de ne pas le dessiner (ou partiellement), ce qui nous permettra de ne rendre que l’information pertinente de la scène ; nous verrons plus loin le lien entre ces deux exemples.

Nous allons exposer quelques travaux antérieurs sur la visibilité en général pour motiver notre travail, ainsi que pour présenter la visibilité analytique.

2 Etat de l'art

2.1 Préliminaires

La visibilité est un problème central en synthèse d'images. En effet, pour un observateur, l'occultation ou non d'un objet ne pose pas de problème, la réponse est intuitive, et tombe sous le sens. En revanche, ce n'est pas le cas pour un ordinateur. Les calculs de visibilité se retrouvent dans bien des domaines de l'informatique : les images de synthèse, la vision, la robotique.

Les problèmes de visibilité ont été abordés de différentes manières dans la littérature ; nous allons énumérer les plus connues afin de donner un aperçu de l'état de l'art en matière de visibilité. Certaines méthodes donnent une information exacte, d'autres sont des méthodes approchées.

Nous allons rappeler quelques travaux antérieurs sur les problèmes de visibilité ; cette liste n'est pas exhaustive mais donne un bon aperçu de l'état de l'art en matière de visibilité. Puis nous motiverons notre travail en exprimant les problèmes majeurs des méthodes précédentes et dans quelle mesure nous tenterons d'y répondre.

Une étude approfondie sur la visibilité en général a été faite par Frédo Durand [Dur99] dans sa thèse de doctorat.

2.2 Visibilité analytique

Graphe d'aspect - *Aspect Graph* La reconnaissance de formes basée sur modèles nécessite une représentation des objets qui soit orientée *point de vue*, c'est à dire une structure qui permet de coder toutes les vues possibles d'un objet. Kœnderink et Van Doorn [KvD79] ont développé le potentiel visuel (*visual potential*) d'un objet, plus connu sous le nom de graphe d'aspect (*aspect graph*). Cette technique consiste à partitionner l'ensemble des points de vue en cellules telles que dans chaque cellule, la vue d'un objet soit qualitativement invariante (voir figure 2 pour illustration). L'ensemble des vues d'un objet est ensuite structurée en graphe de la façon suivante : à chaque nœud correspond une vue générale (ou aspect) de l'objet, le nœud est donc associé à une cellule de l'espace des points de vue ; et à chaque arc correspond un événement de visibilité, c'est à dire une transition entre deux aspects de l'objet. Le graphe d'aspect est le graphe dual de la partition de l'espace des points de vue en cellules.

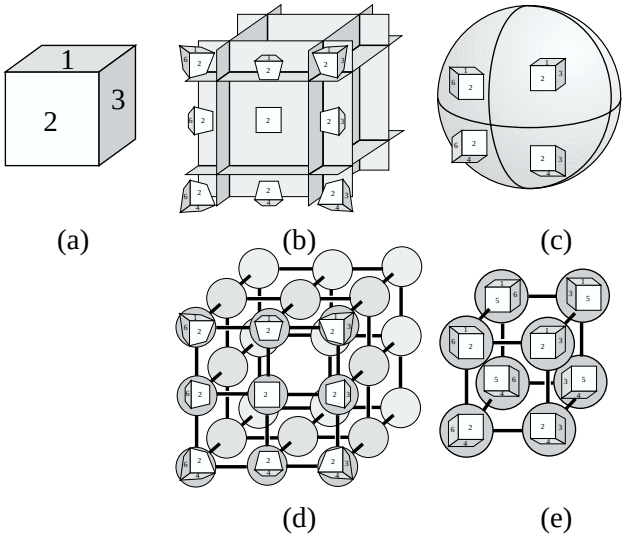
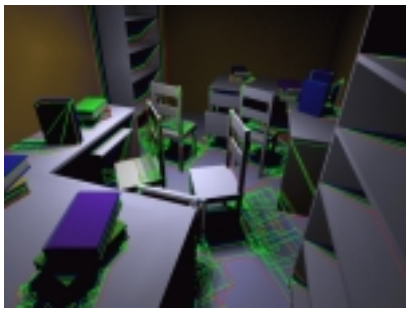


FIG. 2 – Illustration du graphe d'aspect : (a) objet 3D, (b) découpage en cellules, (c) découpage en projection orthographique, (d) graphe d'aspect, (e) graphe en projection orthographique image tirée de [Dur99]

Maillage de discontinuité - *Discontinuity Meshing* Le calcul de la limite entre ombre et pénombre, et entre pénombre et éclairement total peut être fait analytiquement pour des scènes polygonales. Ce calcul est un calcul de visibilité : il consiste à déterminer les limites de visibilité entre les objets et les sources. Si l'on intègre ces limites au maillage initial de la scène, on obtient ce que l'on appelle le maillage de discontinuité. Dans ce maillage, chaque polygone est soit completement à l'ombre soit complètement visible, soit dans la pénombre. Paul Heckbert [Hec92] a proposé un algorithme de construction d'un tel maillage pour le calcul du facteur de formes utilisé en radiosité. Le maillage de discontinuité est un découpage de la scène en cellules dans lesquelles la forme de la source est invariante.



Un exemple de maillage de discontinuité

obtenu à l'aide de l'algorithme de Drettakis et Fiume [DF94]

Projection inverse - *Backprojection* Drettakis et Fiume [DF94] (et Stewart et Ghali [SG94]) ont présenté une technique de calcul exact de l'éclairage par des sources étendues. Cet algorithme est basé sur deux concepts : le maillage de discontinuité complet, et la projection inverse. La projection inverse est utilisée pour calculer la partie visible d'une source étendue (polygonale) depuis un point quelconque d'une surface de la scène. Cette technique consiste à recenser les éléments caractéristiques de l'échange entre la source et un récepteur. Ces éléments appelés *bp-elements* vont être associés au point dont on souhaite calculer la projection inverse pour déterminer la partie visibile de la source. Ils participent à la définition des limites de visibilité. Cette technique permet de calculer des images de grande qualité, et ce très efficacement.

Portails, antipénombre - *Portals* Teller et Hanrahan [TH94] ont proposé une technique de calcul de visibilité pour les scènes à géométrie de type architecturale, c'est à dire des modèles de bâtiments (intérieurs). Le principe de l'algorithme est de découper la scène en boîtes convexes appellées cellules, dans lesquelles le calcul de visibilité est trivial, et de recoller ces cellules par des portails. Pour un bâtiment à pièces rectangulaires, les cellules sont les pièces, et les portails sont les portes. Ensuite, pour calculer l'information de visibilité, on calcule entre deux cellules l'ensemble des portails qui interviennent et on coupe l'ensemble des droites entre les deux cellules par les portails. On obtient ainsi un *shaft* (faisceaux) de visibilité dans lequel toute droite est libre. On peut donc avec cette structure calculer rapidement si deux polygones, de n'importe quelle cellule, sont visibiles mutuellement, partiellement ou pas du tout.

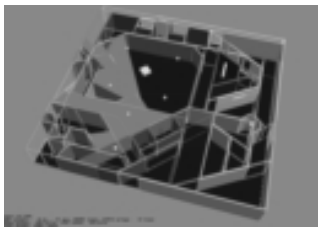


Illustration des portails
(image tirée de l'article [TH94])

Afin de calculer l'ensemble des droites qui traversent avec succés une séquence de portails, il est nécessaire de calculer l'anti-ombre (*antiumbra*), et l'anti-pénombre (*antipenumbra*). Teller a proposé un algorithme pour effectuer ce calcul : [Tel92a] ; voir l'illustration. Pour faire ces calculs, Teller aborde la notion de droite poignardante extrême (*extremal stabbing line*), et d'ensemble critique de droites de dimension un (*critical line set - extremal swath*). Il pose alors une pierre fondatrice de la visibilité analytique.

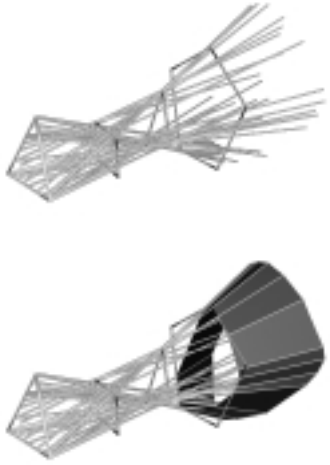


Illustration du calcul de l'anti-ombre et anti-penombre
(image tirée de l'article [Tel92a])

Le Complexe de Visibilité 2D - *Visibility Complex* Pocchiola et Vegter [PV96] ont développé le complexe de visibilité dans le cas de scènes 2D. Le complexe étudie les segments libres maximaux, c'est à dire les segments du plan (dans l'espace objet) de longueur maximale n'entrant pas en intersection avec l'intérieur des objets de la scène. Il en découle que les extrémités de tels segments sont sur les bords des objets.

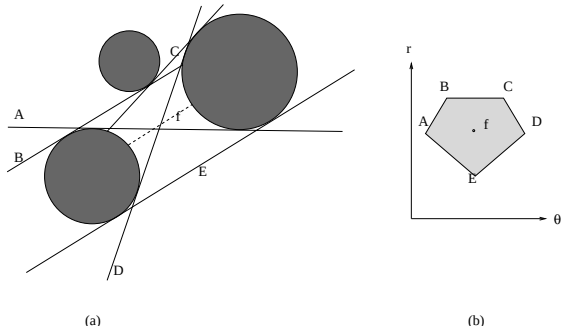


FIG. 3 – Illustration du complexe de visibilité 2D : face du complexe, (a) dans la scène ; (b) dans l'espace des droites.

Pocchiola et Vegter ont proposé un algorithme de construction du complexe optimal en sortie, pour des scènes d'objets lisses.

Le Complexe de Visibilité 3D - *3D Visibility Complex* Durand et al. [DDP96] ont proposé une généralisation du complexe de visibilité au cas de scènes 3D composées d'objets lisses et polygonaux. L'ensemble des segments maximaux est un sur-ensemble d'une variété de dimension 4 (portée à 5 à cause des branchements). Les faces du complexe sont délimitées par des segments tangents (dimension 3), des segments bitangents (dimension 2), tritangents (dimension 1), et finalement les sommets sont des segments quadritangents.

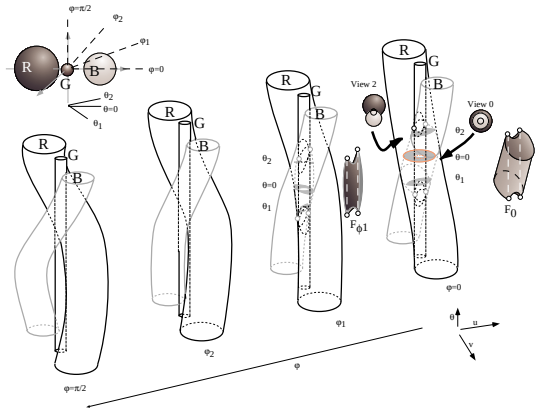


FIG. 4 – Illustration du complexe de visibilité. (image tirée de l'article [DDP96])

Le Squelette de Visibilité - *Visibility Skeleton* Frédo Durand et al. [DDP97] ont présenté une structure de données contenant l'information de visibilité d'une scène polygonale pertinente pour les calculs d'éclairage : le squelette de visibilité. Le squelette de visibilité peut être vu comme un graphe dans l'espace des droites dont les nœuds et les arcs sont les événements de visibilité de dimension zéro et un respectivement. Les nœuds sont des droites poignardantes extrêmes, et les arcs sont des ensembles de droites critiques de dimension un. Le squelette de visibilité a été implanté et utilisé pour une simulation d'illumination globale dans laquelle le facteur de formes Point-Source étendue a pu être calculé analytiquement. Les limites d'ombre et de pénombre peuvent être rapidement calculées considérant n'importe quel polygone comme source (illumination globale). L'information de visibilité ainsi caluclée est exacte.

Le squelette de visibilité est une structure de complexité spatiale plus que quadratique, ce qui rend les besoins en mémoire très grands dans le cas des scènes graphiques d'aujourd'hui (plusieurs centaines de milliers de polygones).

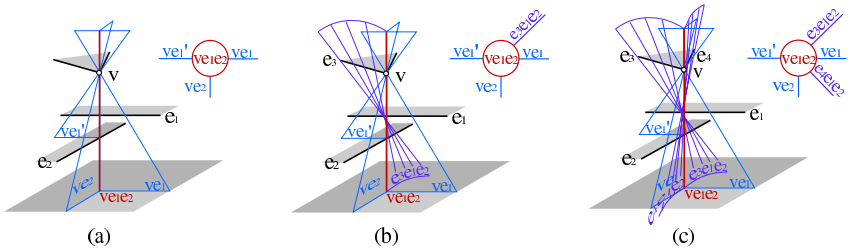


FIG. 5 – Illustration du squelette de visibilité. (image tirée de l'article [DDP97])

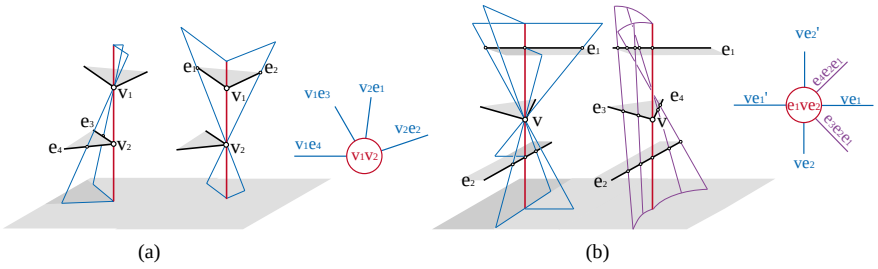
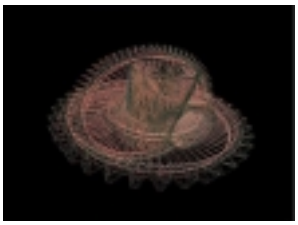


FIG. 6 – Illustration du squelette de visibilité. (image tirée de l'article [DDP97])

2.3 Visibilité approchée

Le tampon de profondeur - *Depth-Buffer*
Considérons une scène polygonale dans un espace à trois dimensions. On souhaite visualiser cette scène sur un écran, tout du moins une projection de cette scène. On choisit donc un point de vue (caméra de la scène), et une méthode de projection (en perspective le plus souvent). Un dessinateur pourrait intuitivement éliminer les objets et parties d'objets invisibles de la scène, cependant c'est un réel problème de visibilité. Quels objets doit-on dessiner ? Sont-ils tous visibles depuis le point de vue choisi ? Nous avons besoin de déterminer les parties visibles des objets de la scène afin d'en dessiner la projection de façon cohérente sur le plan image. Une première méthode consiste à calculer la distance entre chaque objet et le point de vue et de dessiner leurs projections respectives par ordre décroissant de profondeur ; cette méthode s'appelle l'algorithme du peintre. Le principal problème de cette méthode peut être illustrée par un jeu de mikado. En effet il se peut que trois objets ne puissent être ordonnés de manière à ce que chacun se trouve derrière l'autre, on peut rencontrer un cycle. Des astuces existent pour *s'en sortir*, mais elles rendent l'algorithme plus complexe, surtout dans le cas d'objets courbes. Catmull [Cat74] a proposé une technique consistant à conserver pour chaque pixel de l'image la propondeur du point ayant été dessiné sur le pixel. Pour chaque nouveau point que l'on souhaite dessiner, il suffit de comparer sa profondeur à celle déjà enregistrée dans le tampon et le cas échéant dessiner et modifier le tampon ou ne rien faire. Grâce à son extrême simplicité, cette technique a été intégrée dans les cartes graphiques. Les utilisations de cette technique ne se comptent plus.

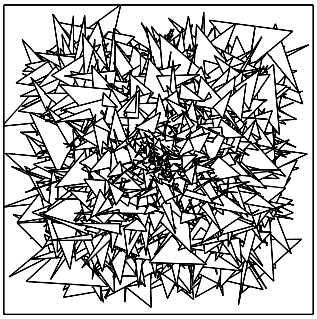


Objets en fil de fer

Tampon de profondeur

Résultat

Carte de visibilité approchée - *Approximate Visibility Map* Stewart et Karkanis [SK98] ont proposé une méthode permettant de calculer approximativement la visibilité depuis un point, c'est à dire la partie de la scène visible depuis un point de vue. Cette méthode consiste à effectuer un rendu de la scène dans un tampon à l'aide de la technique du tampon de profondeur, en donnant à chaque pixel un code couleur correspondant au polygone qu'il représente. Une fois ce tampon calculé, un algorithme permet d'en extraire un graphe dont les sommets sont soit les sommets de la scène visibles depuis le point de vue considéré, soit des intersections entre projections d'arêtes ; les arcs sont les portions d'arêtes visibiles de la scène, et les faces sont les portions visibles de la scène. Cette technique permet d'utiliser l'accélération matérielle du tampon de profondeur et donne une approximation de la partie visibile de la scène depuis un certain point de vue. Elle permet en particulier de calculer une approximation du facteur de formes, ou encore du maillage de discontinuité.



Carte de visibilité approchée.

(image tirée de l'article [SK98])

Carte d'ombres - *Shadow Map* William [Wil78] a proposé un algorithme permettant de calculer les ombres projettés par une source directionnelle. Cette technique est basée sur l'utilisation du tampon de profondeur. La technique s'applique en deux passes : une première passe consiste à dessiner la scène depuis le point de vue de la source selon un mode de projection correspondant au type de source. Une fois ce premier rendu effectué, la carte de profondeur pour la source est stockée. Ensuite, lors du rendu de l'image (point de vue le la caméra), chaque pixel est reprojetté dans la vue de la source, et selon la profondeur obtenue, comparée à celle stockée dans la carte de profondeur, le pixel est marqué comme éclairé ou à l'ombre.
Le principal avantage du *shadow map* est qu'il peut traiter toute géométrie traitée par le tampon de profondeur.

2.4 Suppression d'objets cachés - *Occlusion culling*

Scènes à géométrie architecturale Teller et al. [TS91] ont proposé une technique de suppression d'objets cachés pour des scènes à géométrie architecturale. La méthode tire parti des caractéristiques des modèles afin de ramener le problème à un problème 2D. Les scènes en entrée doivent représenter des bâtiments dont les murs sont alignés avec les axes. L'idée est de subdiviser la scène en cellules, et placer entre les cellules pouvant se *voir* des portails. Ensuite, on cherche s'il existe des lignes de mire (*sight lines*), au travers d'une séquence de portails. On propage alors la visibilité de cellule en cellule en construisant un arbre appelé arbre poignardant (*stab tree*). Le parcours dans cet arbre donne l'ensemble des cellules visibles depuis une cellule fixée. Ensuite, pour la visibilité depuis un point de vue, on parcours l'arbre en prenant en compte le cone de visibilité depuis le point de vue. Cette approche a été améliorée par la suite pour prendre en compte des modèles de scènes moins restrictifs.

Z-Buffer Hiérarchique - *Hierarchical Z-Buffer* Green et al. [GKM93] ont proposé une technique de suppression d'objets cachés basée sur le concept du tampon de profondeur. Cette technique consiste à avoir une approche hiérarchique du problème. L'approche est la suivante : au lieu d'utiliser le tampon de profondeur à une résolution donnée sur tous les objets de la scène, l'idée est d'utiliser une hiérarchie de tampons de profondeur à des résolutions différentes et de tester si des objets sont cachés en utilisant leur boîte englobante. Cette technique permet de prendre en compte plusieurs formes de cohérence : la cohérence spatiale, la cohérence dans l'espace image, et la cohérence temporelle. Pour la cohérence spatiale, les objets proches sont regroupés dans une hiérarchie de boîtes englobantes ; la cohérence dans l'espace image et la cohérence temporelle sont utilisées pour déterminer et mettre à jour progressivement une liste d'objets visibles à dessiner (les plus proches).

Hiérarchie de cartes d'occultation - *Hierarchical Occlusion Maps* Zhang et al. [ZMHI97] ont proposé une technique de suppression d'objets occultés conservative. Cette technique consiste à étudier la scène hiérarchiquement et supprimer des objets éventuellement occultés pour un point de vue donné à plusieurs niveaux de la hiérarchie. Les cartes d'occltation sont construites à partir d'objets occultants préalablement sélectionnés selon des critères stricts. Une fois les cartes établies, elles sont appliquées à la scène hiérarchique pour le rendu. Les occultants sont mis à jour lors de balades.

2.5 Structures d'accélérations de tracé de rayons

Parcours dans une grille Woo et Amanitides [AW87] ont proposé une technique rapide de parcours de grille régulière. Le concept de la grille consiste à subdiviser l'espace en cellules et à référencer dans chaque cellule les objets de la scène qui occupent un espace commun avec la cellule. Une fois ces cellules construites, les tests d'intersection entre un rayon et la scène sont accélérés, en effet, on ne parcours que les objets des

cellules de la grille en intersection avec le rayon. Les objets éloignés du rayon ne sont donc jamais testés ; voir figure 7 pour illustration.

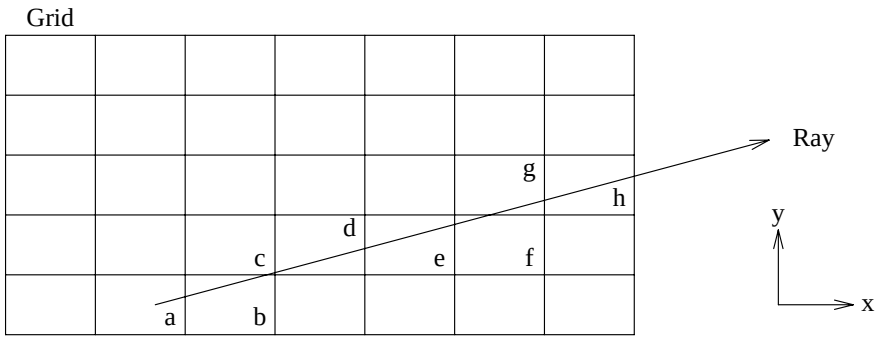


FIG. 7 – Parcours dans une grille - image tirée de l'article [AW87]. Les cellules parcourues par le rayon sont a, b, c, d, e, f, g, h dans cet ordre.

Grille récursive Jevans et Wyvill [JW89] ont proposé une structure d'accélération à base de grille dont chaque cellule est elle-même une grille. Cette technique permet d'avoir une structure dont la finesse / résolution est adaptée à la complexité locale de la scène. Ainsi pour les cellules vides (ou presque), elles sont conservées telles-que, mais les cellules plus remplies sont à leur tour subdivisées. Cette technique permet une plus grande flexibilté dans le choix de la résolution de la grille. De part le caractère récursif de cette technique, la structure est communément appelée grille récursive.

Hiérarchie de grilles uniformes - *Hierarchy of Uniform Grids* Cazals et al. [CDP95] ont proposé une technique à plusieurs niveaux. Les éléments de la scène sont regrouppés en classes de taille. En effet pour une grille très subdivisée, un même objet sera référencé par un très grand nombre de cellules, il sera donc dans certains cas testé plusieurs fois. L'idée est de construire une grille dans laquelle on référence non pas des objets mais des groupes d'objets. Ces groupes d'objets sont eux-mêmes structurés dans des grilles éventuellement récursives. Le parcours de cette structure est analogue à celui d'une grille récursive : on utilise la première grille pour tester avec quel groupe d'objet le rayon peut entrer en intersection, puis pour chaque groupe rencontré, on parcours la sous-grille concernée. Cette technique est faite pour rester performante pour des scènes comportant à la fois de grands objets et de petits objets.

2.6 Bases mathématiques - Outils utilisés

Nous allons aborder les problèmes de visibilité sous un angle plus mathématique et plus théorique que nos prédescesseurs. Ceci n'est pas une recherche mathématique, mais nous allons utiliser des outils relativement évolués pour effectuer nos calculs ; nous les introduirons en détail progressivement tout au long du discours, lorsqu'ils seront nécessaires. Cependant, nous allons énumérer les plus importants et ceux qui ont fait que cette nouvelle approche a pu être envisagée.

Le squelette de visibilité est une approche *line-space* de la visibilité, c'est à dire que l'on étudie les droites de l'espace monde. Pour manipuler ces droites, nous allons utiliser une paramétrisation de Plücker ([]). Nous détaillons cette paramétrisation en annexe pour les généralités ; quelques points clés cependant sont donnés dans le corps du document.

Cette nouvelle approche plus générale va nous amener à manipuler ces droites, c'est à dire des éléments d'un espace vectoriel de dimension 6. Cette paramétrisation, et les calculs dans cet espace donnent malheureusement naissance à des relations non linéaires (quadratiques pour être exact), ce qui fait sortir notre raisonnement des limites de l'algèbre linéaire. Afin de développer un raisonnement théorique fondé, nous allons être amenés à manipuler des objets algébriques comme les Idéaux de polynômes, et les Variétés algébriques. Nous n'utiliserons que quelques notions liées à ces objets, mais leur utilisation nous est nécessaire.

Nous avons choisi d'effectuer la plupart des calculs de visibilité (lancé de rayons), *à un epsilon près*, c'est à dire à dire par exemple si une droite passe *presque* par un point. En effet ces considérations d'approximations sont nécessaires car les données que l'on manipule, ainsi que les nombres avec lesquels nous faisons les calculs sont à précision finie (fixée pour plus de performances). Nous allons donc effectuer nos calculs dans une arithmétique d'intervalles.

3 Le Squelette de visibilité

Cette partie est l'exposé réduit de l'article de Durand, Drettakis et Puech[DDP97], ainsi que du chapitre de la thèse de Durand [Dur99] qui en traite. Les figures en sont tirées.

Nous avons vu précédemment plusieurs techniques de calcul de la visibilité globale d'une scène : le graphe d'aspect (e.g. [KvD79]), l'antipenombre et les portails ([TH94],[Tel92a]), les maillages de discontinuité et projection inverse ([DF94], [SG94]). Dans chacune de ces méthodes, les changements de visibilité sont caractérisés par les ensembles de droites critiques (*critical line swath*) et les droites poignardantes extrêmes (*extremal stabbing lines*), qui sont le lieu des événements de visibilité.

3.1 Présentation

Le squelette de visibilité est une structure de données ayant la forme d'un graphe dont les nœuds sont les droites poignardantes extrêmes et les arcs sont les ensembles de droites critiques.

Ces événements de visibilité peuvent être identifiés et classés dans un catalogue pour le cas des scènes polyonales. Les droites poignardantes extrêmes et les ensembles de droites critiques vont donc être affectés d'un type défini ci-après. Le type est conditionné par l'ensemble de générateurs qui définissent l'événement de visibilité, un générateur étant soit une arête, soit un sommet (éventuellement une face dans certains cas). La scène polygonale est donc considérée comme un ensemble de générateurs.

Les droites poignardantes extrêmes :

- VV : droite définie par deux sommets du maillage
- VEE : droite définie par un sommet et deux arêtes
- $E4$: droite définie par quatre arêtes
- F_vV : droite définie par une face et deux sommets dont un appartient à la face.
- F_vE : droite définie par une face, un sommet de la face et une arête
- FF : droite définie par deux faces
- FEE : droite définie par une face et deux arêtes

Par la suite, nous parlerons indifféremment de nœud du squelette ou de droite poignardante extrême, en utilisant éventuellement leur formulation *codée* (ci-dessus).

Les ensembles de droites critiques sont des arcs reliant les nœuds du squelette, il n'est donc pas nécessaire de les identifier par leurs générateurs, la structure de graphe donne implicitement leur type.

Dans la figure 8, nous présentons un exemple de droite poignardante extrême ainsi que les arcs qui y sont reliés.

catalogue d'événements visuels et de leurs adjacences Pour construire ce graphe, l'algorithme utilise un catalogue de nœuds et d'adjacences , qui permettent de définir

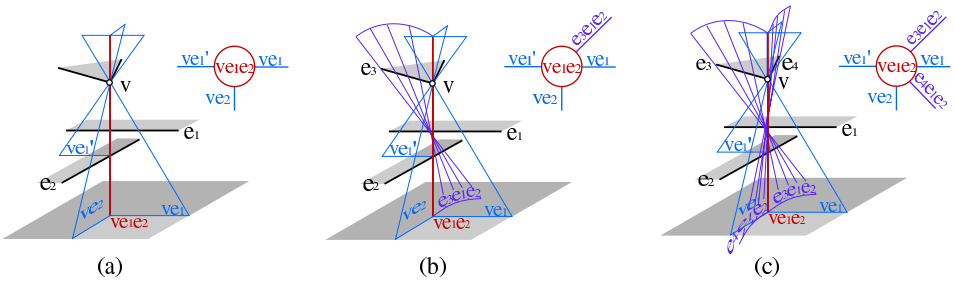
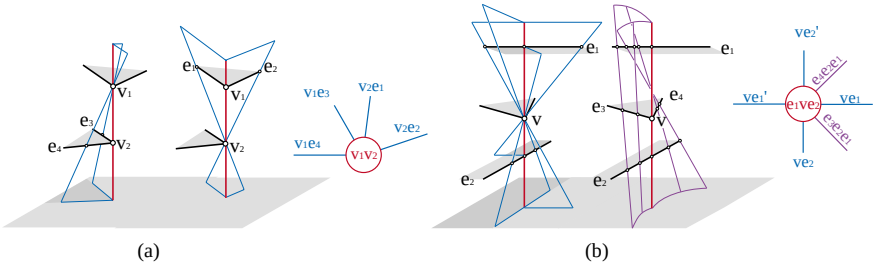
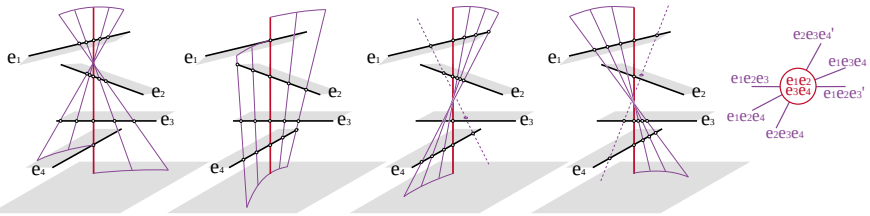


FIG. 8 – Une droite poignardante extrême de type VEE, (a) : les deux arcs s'appuyant sur V, (b) et (c) les triples arêtes qui sont des arcs, portion de quadrique. (image tirée de l'article [DDP97])

implicitement les arcs. Ce catalogue donne de façon exhaustive [1] l'ensemble des types de droites poignardantes extrêmes et de leurs adjacences. Nous avons vu figure 8 le nœud VEE. Ci-contre, le nœud VV, et le nœud $E4$.



Adjacences pour le nœud VV



Adjacences pour le nœud $E4$

Données en entrée Le modèle de scène fournit les adjacences entre sommets, arêtes et faces des polyèdres. Chaque élément se voit affecter un numéro d'identification unique. Toutes les arêtes sont supposées orientées (de façon arbitraire).

[1] Note : le catalogue n'est exhaustif que dans le cas de scènes génériques, c'est à dire sans dégénérescences.

Structure de données La structure de données est la suivante : des nœuds, des arcs, et la structure globale du squelette.

Chaque nœud (*Node*) contient la liste des arcs adjacents, les faces F_{up} et F_{down} qui donnent les limites spaciales de la droite poignardante extrême (pouvant être un segment, une demi-droite ou une droite), et les deux points d'intersection de la droite avec ces faces le cas échéant.

Chaque arc contient les deux nœuds qu'il relie, deux paramètres correspondant aux deux positions respectives des deux nœuds sur une arête de l'arc, ainsi que les deux faces de début et de fin.

Le squelette contient la liste des nœuds et la liste des arcs. Dans la première formulation ([DDP97]), les arcs étaient stockés dans un tableau à deux entrées : les deux polygones sur lesquels il s'appuyait. Ce tableau étant presque partout vide, la structure de stockage a été un arbre de recherche.

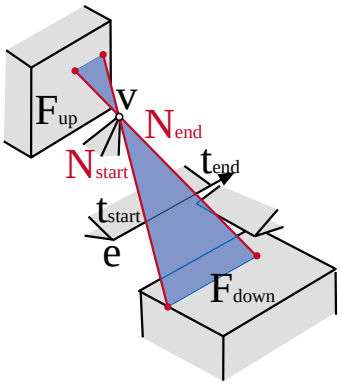
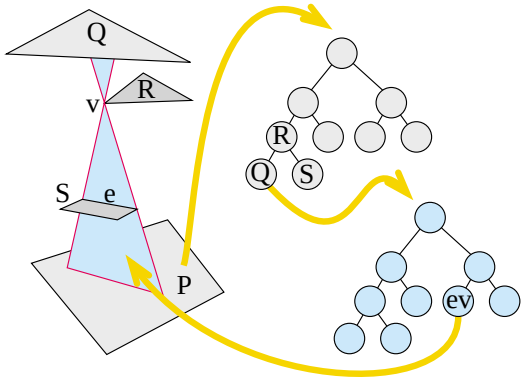


illustration de la sctructure de données



small structure d'accès à l'information de visibilité entre deux faces

Utilisation du squelette Afin d'accéder à l'information du squelette de visibilité, nous devons pouvoir retrouver les nœuds et les arcs qui se trouvent entre deux faces. Pour celà, à chaque face est associée un arbre de recherche sur tous les polygones pouvant être *liés* par un arc, c'est à dire pour lesquels un arc peut avoir l'un pour F_{up} et l'autre pour F_{down}. Et pour chaque feuille de l'arbre, un arbre sur les arcs liant les deux polygones.

3.2 Algorithme de construction

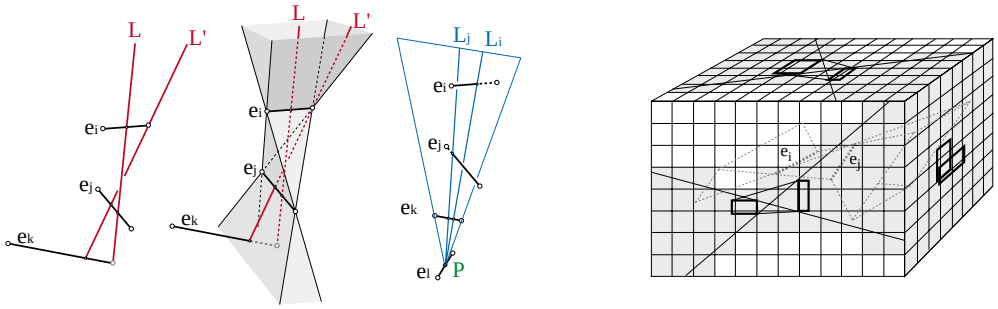
Détection des nœuds

Le principe général de la détection des nœuds consiste à proposer une droite poignardante extrême engendrée par un ensemble de générateurs (c'est à dire une droite qui passe par tous les générateurs). On teste ensuite si un objet se trouve entre les générateurs, auquel cas la droite est annulée et le nœud n'est pas crée (test d'occultation).

Nœuds triviaux Les nœuds les plus simples à traiter sont les VV, les F_{vv} et les FE. On effectue une boucle sur les générateurs potentiels (sommets, arêtes et faces), puis on utilise un lancer de rayon afin de valider la droite (test d'occultation). Le lancé de rayons donne de plus les extrémité du nœud. La complexité semble être $O(n^2 r(n))$ avec n la taille de la scène (nombre de générateurs).

Nœuds VEE et $E4$ Pour calculer ces nœuds, on effectue une boucle sur les paires d'arêtes. Pour chaque paire d'arêtes, l'ensemble des droites passant par les deux arêtes forme une sorte de sablier, ou double coin (voir figure 9). Tout sommet se situant dans ce sablier peut générer un nœud de type VEE. Afin d'optimiser la recherche de tels éléments, on utilise une grille (voir figure 9) et on ne recherche les sommets que dans les cellules de la grille en intersection avec le sablier.

Pour le calcul des $E4$, on boucle sur les arêtes en intersection avec le sablier, on a donc trois arêtes, que l'on note e_i, e_j, e_k. Ensuite, on boucle sur les arêtes en intersection avec le sablier pour déterminer une quatrième arête e_l. On obtient deux ensembles critiques : $e_i e_j e_k$ et $e_i e_j e_l$. Une recherche dichotomique est ensuite utilisée pour trouver (s'il existe) le point sur e_l par lequel passe notre droite $E4$. Cette dichotomie est faite sur un point P de e_l, en cherchant le zéro de l'angle formé par les deux droites L_i et L_j. Les droites L_i et L_j étant les droites définies par l'intersection du plan (P, e_i) et avec e_j et e_k respectivement. La complexité du pire cas est $O(kn^4 r(n))$, avec k celle de la dichotomie, mais l'utilisation du sablier et de la grille donne des résultats satisfaisants sur les scènes *raisonnables* utilisées dans les tests.



sablier et algorithme de calcul des $E4$

grille

FIG. 9 – Nœuds VEE et $E4$

Nœuds engendrés par des faces Pour calculer les nœuds engendrés par des faces, on calcule l'intersection des arêtes de la scène avec le plan support de la face (voir figure 10). Pour toute arête qui coupe ce plan, on essaie de créer les nœuds F_vE avec les sommets de la face.

Pour chaque paire d'arêtes, en intersection avec le plan de la face, on teste si on peut générer un FEE, pour celà, on teste si la droite passant par les deux points d'intersection du plan et des arêtes passe par la face ou non (ce test se fait dans le plan).

Finalement, nous testons les couples de faces, la droite proposée est l'intersection des deux plans support des faces.

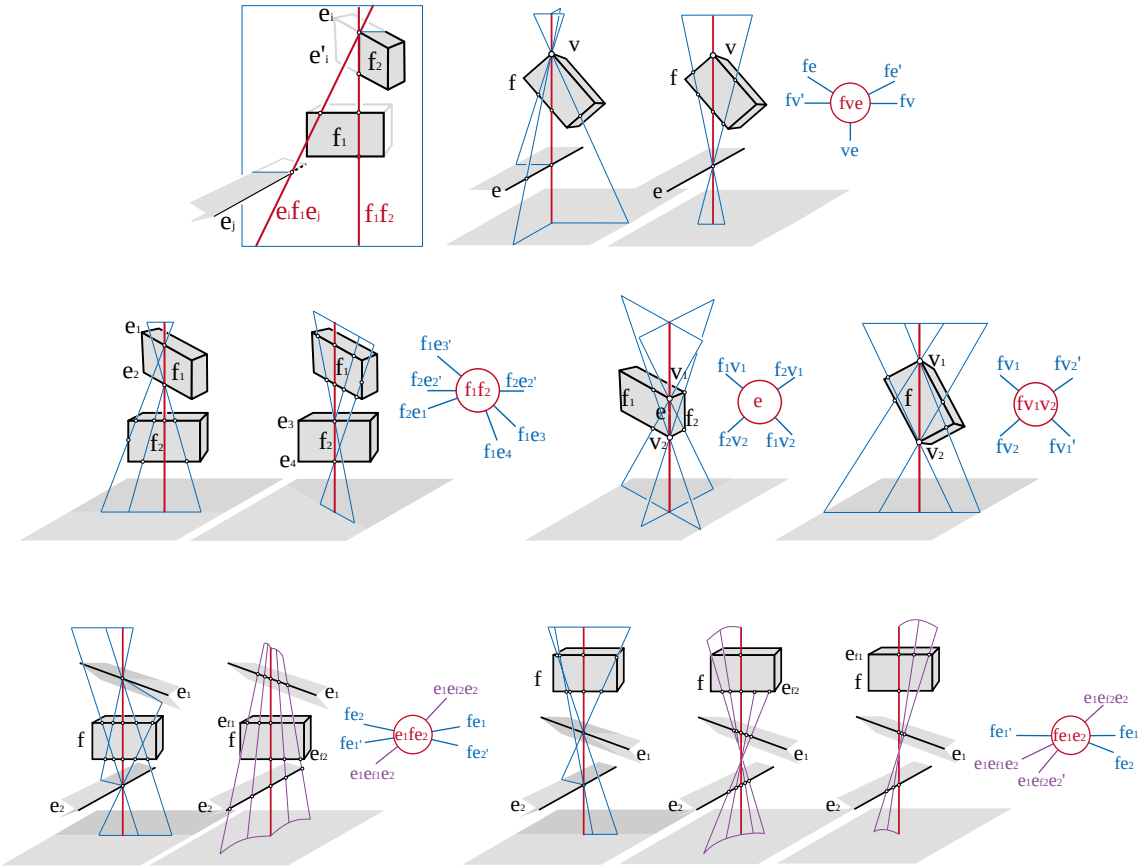


FIG. 10 – Illustrations du calcul des nœuds générés par des faces

Création des arcs La création des arcs du squelette de visibilité est effectuée simultanément à la détection des nœuds. Lorsque un nœud est crée nous traitons simultanément tous les arcs adjacents grâce au catalogue. Si l'arc a déjà été créé, nous le relions simplement au nouveau nœud. L'arc est éventuellement coupé en deux si le nouveau nœud tente de l'insérer au milieu.

Le catalogue permet de déterminer sans calculs les limites des arcs (F_{up} et F_{down}). Ces limites sont déduites de celles des nœuds.

Figure 11, un exemple d'insertion de nœuds et d'arcs.

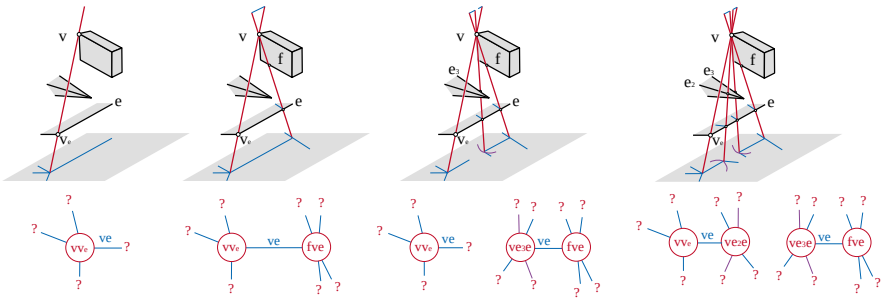


FIG. 11 – Illustrations de l'insertion des nœuds et arcs dans le squelette

Résultats Les résultats obtenus sont présentés figure 12.

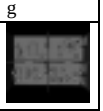
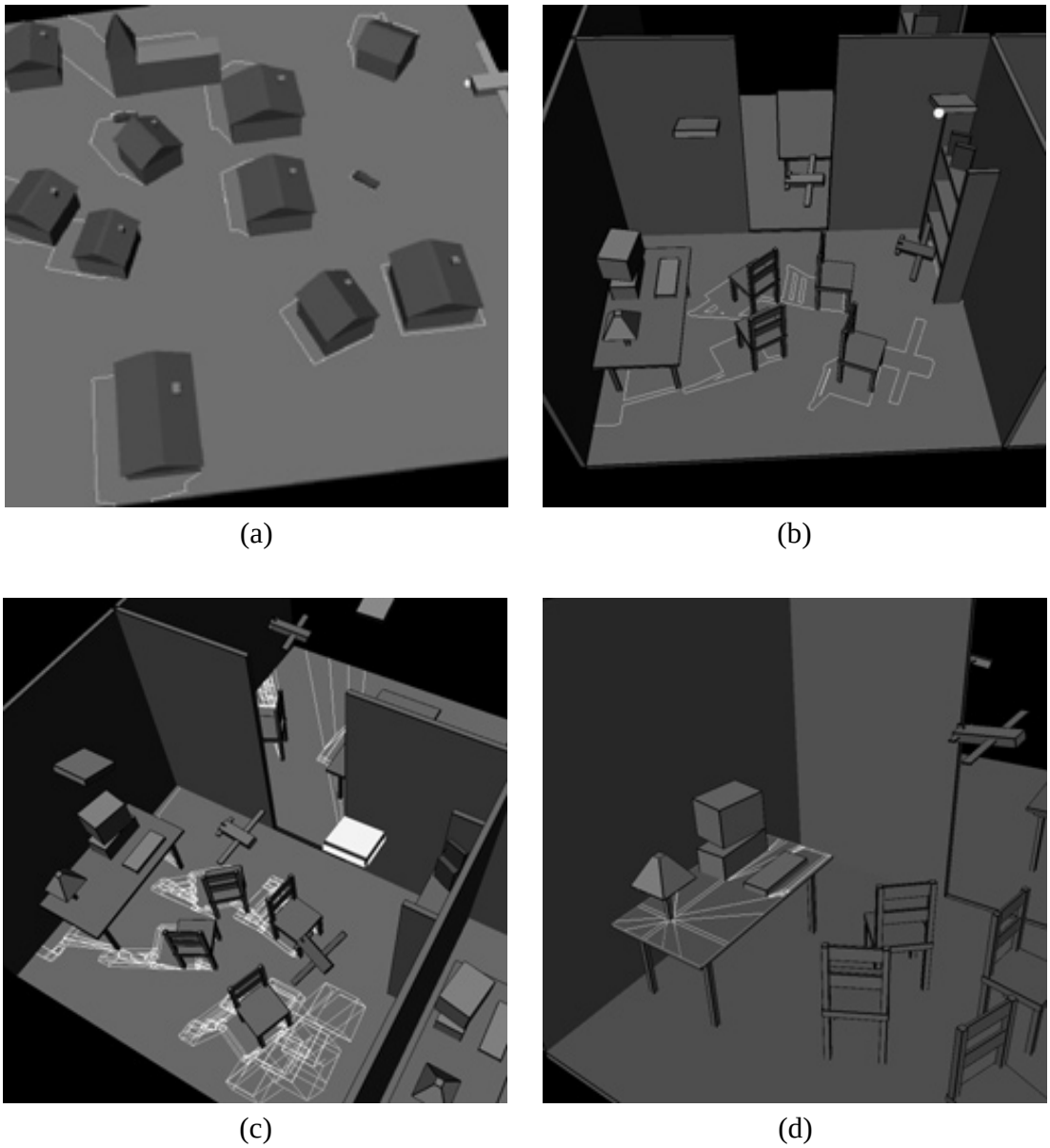
	a	b	c	d	e	f	g
Scène							
Polygones	84	168	312	432	756	1056	1488
Nœuds ($*10^3$)	7	37	69	199	445	753	1266
Arcs ($*10^3$)	16	91	165	476	1074	1836	3087
Construction	1 s 71	12 s 74	37 s 07	1 min 39 s	5 min 36 s	14 min 36 s	31 min 59
Mémoire (Mo)	1.8	9	21	55	135	242	416

FIG. 12 – résultats obtenus et présentés dans le papier [DDP97]. Machine : SGI Onyx 2 avec un R10000 à 195 MHz.

La *complexité* pratique mesurée sur ces scènes donne de l'ordre de n^2 pour le nombre de nœuds (spatiale) et $n^{2.4}$ pour le temps de calcul (temporelle).

Résultats sur les scènes considérées : figure 13.



(a) (b)

(c) (d)

FIG. 13 – (a) : Partie du sol visible depuis l'un des sommets de l'avion ; (b) : partie du sol visible depuis l'un des sommets de la source droite ; (c) : Maillage de discontinuité par rapport à la source droite ; (d) : Maillage de discontinuité créé par la lampe sur la table.

3.3 Conclusion - Discussion

Le squelette s'avère être un outil très intéressant, cependant, il souffre de deux problèmes majeurs : sa complexité spatiale (et temporelle), et son manque de robustesse. La complexité spatiale quadratique nous empêche d'envisager l'utilisation du squelette pour de grosses scènes (de l'ordre de la centaine de milliers de polygones). La plus scène la plus grosse utilisée dans les tests est composée de 1500 polygones. Les problèmes de robustesse sont liés à l'utilisation du catalogue, qui lui-même a été le moyen de rendre le squelette utilisable. Les configurations de droites poignardantes extrêmes proposées dans le catalogues ne sont pas suffisantes, en effet des dégénérescences peuvent apparaître et ne pas pouvoir être traitées ; ces dégénérescences n'étant pas en nombre fini, il n'est pas envisageable de concevoir un algorithme robuste basé sur un catalogue.

4 Motivation

4.1 Objectifs

Nous avons présenté nombre d'algorithmes de calcul de l'information de visibilité, cependant aucun ne permet de calculer de façon exacte la visibilité de n'importe quelle scène. Dans certains cas comme le *depth-buffer* ou le *shadow-map*, l'information de visibilité donnée correspond à un échantillonnage de l'information exacte. Ou encore une information approchée peut être donnée, comme dans le cas de l'*approximate visibility map*. Dans d'autres cas, l'information de visibilité est certes calculée de façon exacte, mais l'algorithme ne s'applique qu'à des cas particuliers d'environnements, comme les scènes architecturales pour les techniques à base de portails, ou à des scènes particulières et petites pour le squelette de visibilité.

Le calcul exact de visibilité est utile ! Il permet en particulier de calculer des solutions d'éclairage de qualité, indépendantes de la résolution choisie (résolution d'image ou de carte), et permet ainsi d'établir des solutions de référence. Nous allons donc nous baser sur le squelette de visibilité et tenter de le rendre applicable à tout type de scène. Pour celà, nous devons corriger les défauts du squelette de visibilité qui le rendent inutilisable en pratique. Nous devons pouvoir traiter les scènes comportant des dégénérescences géométriques c'est à dire des arêtes coplanaires, des points alignés, ou des objets en intersection. De plus, le squelette de visibilité calcule une information globale, et la taille de la sortie est de complexité plus que quadratique, ce qui rend impossible son passage à l'échelle, c'est à dire le traitement de scènes complexes (voir figure 14).

Effectuer l'ensemble des calculs de visiblité n'est pas envisageable. En effet, la complexité spatiale de la structure complète est plus que quadratique, ce qui pour des scènes satisfaisantes (200000 polygones), est trop grande pour être considéré.

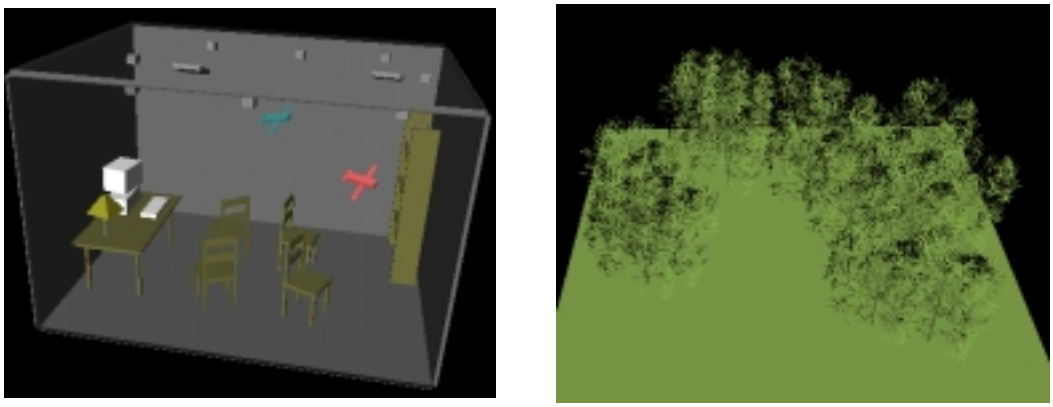


FIG. 14 – A gauche, ce qui était traité par le squelette de visibilité (1500 polygones environ). A droite, exemple type de scène que l'on souhaite traiter (200000 polygones environ).

Nous allons présenter une technique de calcul de visibilité analytique qui s'appuie sur le squelette de visibilité, mais avec un approche différente. Celle-ci prendra en

compte les dégénérescences géométriques, et permettra de traiter des scènes de grande complexité géométrique.

Les restrictions géométriques du squelette de visibilité étaient les suivantes :
- les éléments dégénérés géométriquement (alignements, coplanarités) sont traités spécifiquement, de façon non systématique.
- les normales aux faces doivent être cohérentes et les maillages doivent modéliser des surfaces d'objets solides avec information complète de connectivité.
- les objets représenter ne doivent pas entrer en contact, ni en intersection

Ces restrictions sont extrêmement difficiles à satisfaire : la plupart des scènes graphiques ne satisfont pas la première, ni la seconde restriction, souvent les objets sont alignés, comme les fenêtres d'un bâtiment. Les deux dernières ne sont pas satisfaites par beaucoup de modèles, en effet, les maillages proposés ont parfois leurs faces mal orientées et des objets en intersection.

Ces trois restrictions rendent le squelette de visibilité inutilisable dans le cas de scènes réelles.

4.2 Approche nouvelle

Nous allons proposer une nouvelle approche permettant de lever toutes ces restrictions. Elle se basera sur :
- Traitement général s'appliquant à toutes les scènes.
- Calculs de visibilité faits à un ϵ près.
- Prise en compte des intersections et des contacts.

Nous allons proposer un exemple afin de présenter les différentes techniques développées. L'exemple que nous considérons est le calcul des ombres projettées par une source directionnelle. Cet exemple nous permet de bien illustrer nos techniques car il est simple à visualiser.

Résultats escomptés On souhaite pouvoir, à partir d'une scène et d'une source directionnelle calculer à un ϵ fixé près, les ombres projetés par la source sur la scène.

La seule hypothèse faite sur la scène en entrée est qu'elle doit être décrite à l'aide de polygones ; cependant, bien que la connectivité ne soit pas nécessaire, elle permet d'optimiser les calculs.

La sortie consistera en un ensemble de polygones dont chacun est soit entièrement à l'ombre, soit entièrement éclairé. La connectivité est conservée lorsqu'elle existait dans les données en entrée.

Etapes de construction Elles sont ici données sans les optimisations (voir figure 15).

```
source : S
ConstuitNœuds
{
  pour chaque sommet V
    δ(S,V) = droite passant par V
        de direction S
    nc = Nœud candidat ({S,V}, δ(S,V))
    ValidationNœud(nc)
  pour chaque arête E1
    pour chaque arête E2
      δ(S,E,E) = droite passant par E et E
          de direction V
      nc = Nœud candidat ({S,E,E}, δ(S,E,E))
      ValidationNœud(nc)
}
ConstruitArcs
{
  pour chaque arête E
    l la liste des nœuds sur l'arête
    trie les éléments de l selon leur
        position sur E
    l = ν0, ..., νn+1
    pour i = 0 ... n
      ac = Arc candidat {νi, νi+1}
      ValidationArc(ac)
}
```

FIG. 15 – Etapes de construction

Lors de la phase de validation, on calcule le bloqueur final de chaque droite poignardante extrême, ainsi que les éventuels générateurs additionnels.

Enfin, pour chaque bloqueur de fin, on liste les arcs qui y sont bloqués, et on en déduit les discontinuités d'ombres pour la source directionnelle ; on calcule ensuite une triangulation de Delaunay contrainte pour obtenir des polygones étant soit entièrement à l'ombre soit entièrement éclairés.

Cas général En général, pour une application, on effectuera les opérations suivantes :

- Enumération - Validation des nœuds à calculer
- Enumération - Validation des arcs
- Exploitation des résultats

4.3 Structure de la suite du mémoire

Nous allons présenter dans la suite du mémoire les résultats théoriques, puis les applications du squelette de visibilité robuste.

Les résultats théoriques seront développés en six sous-parties :

- Abstraction de la méthode
- Le *presque*
- Phases de construction
- Enumération
- Validation
- Intersections

L'abstraction de la méthode décrit le raisonnement théorique développé pour la prise en compte des dégénérescences dans la description des événements de visibilité et des objets de visibilité utilisés dans le squelette.

Le *presque* exprime les outils utilisés ainsi que la démarche proposée pour faire de la visibilité à un ϵ près. Nous détaillons les techniques que nous utilisons par la suite.

Les phases de construction décrivent la méthode de construction d'un squelette complet. Chaque partie est ensuite détaillée. Cette construction complète n'est que théorique, car comme nous l'avons observé, calculer le squelette complet pour des grosses scènes n'est pas envisageable (complexité spatiale plus que quadratique).

L'énumération donne les techniques d'énumération des différents nœuds non dégénérés, la phase de validation complétant ces nœuds le cas échéant. Cette partie décrit des méthodes qui seront employées éventuellement localement pour le maillage de discontinuité par exemple.

La validation décrit la phase clé de notre algorithme : chaque nœud candidat va être éventuellement validé et s'il est dégénéré, c'est lors de cette phase que nous allons le détecter.

Les applications proposées sont au nombre de cinq. Cette liste n'est bien entendu pas exhaustive.

- Ombres projettées par une source directionnelle
- Ombres projettées par une source ponctuelle
- Tracé de sommets robuste
- Suppression d'objets cachés
- Maillage de discontinuité

Chacune des applications fait appel à des techniques développées dans la partie résultats théoriques, ainsi qu'à des techniques d'optimisation spécifiques à chaque application. Pour chacune d'entre-elles, nous allons détaillller des techniques d'optimisation et la partie du squelette utilisée (avec une méthode de construction).

Notons que seule la première application a été entièrement implantée.

5 Résultats théoriques

5.1 Abstraction de la méthode

5.1.1 Préliminaires mathématiques

Les calculs de visibilités se font sur des rayons lumineux, nous allons donc être amenés à utiliser des droites de l'espace. Pour ce faire, nous utiliserons une paramétrisation des droites proposée par Plücker [Plü65], (décrite en annexe). Cette paramétrisation est faite dans un espace vectoriel de dimension 6, ce qui nous permet d'effectuer les calculs sur ces droites en algèbre linéaire. Notons simplement qu'une droite est décrite par deux vecteurs $\delta = (u, v)$[2], avec u le vecteur directeur (les droites sont orientées), et $v = OM \times u$ [3] pour O l'origine du repère et M un point de la droite (v est indépendant du point M choisi). Nous utilisons cette représentation car c'est une paramétrisation continue des droites de l'espace monde, et que les relations qui nous intéressent sont linéaires, ainsi nous pourrons utiliser l'algèbre linéaire pour effectuer nos calculs. Plücker a introduit une forme bilinéaire donnant un prédicat de coplanarité de deux droites. Deux droites $\delta_1 = (u_1, v_1)$ et $\delta_2 = (u_2, v_2)$ sont coplanaires si et seulement si :

$$\delta_1 \odot \delta_2 = u_1 \cdot v_2 + v_1 \cdot u_2 = 0$$

Les notions d'idéaux et de variétés sont aussi utilisées [4], elles sont développées extensivement par Cox, Little et O'Shea dans *Ideals, Varieties, and Algorithms* [CLO98].

5.1.2 Couche abstraite du squelette

Nous avons déjà présenté le squelette de visibilité tel qu'il a été proposé par Durand et al. [DDP97], mais malgré le gain en performances, l'utilisation du catalogue empêche le squelette de visibilité d'être robuste face aux dégénérescences. Nous allons présenter ici une approche similaire mais plus générale en prennant en compte l'éventualité de dégérérescences. Pour ce faire, nous allons prendre un peu de recul face au squelette de visibilité et analyser ses éléments constitutifs afin de les caractériser de façon robuste.

Un exemple de droite poignardante extrême dégénérée est présenté figure 16.

Générateur La notion de générateur avait été abordée par Durand et al. dans [DDP97], un générateur était l'un des trois éléments suivant : sommet, arête ou face. Remarquons que la face est un générateur un peu particulier, en effet, si une droite poignardante extrême est générée par une face, elle est a fortiori générée par les éléments de la face (sommets ou arêtes). Le type de générateur face est donc redondant. Il avait été introduit car il s'avérait très utile pour faire face à certains problèmes de robustesse et permettait

[2] Le plus souvent, les droites seront notées δ. En l'absence de précision, δ représentera une droite quelconque.

[3] On peut rencontrer différentes versions de formulation des coordonnés de Plücker, allant de 6 variables sans lien aux glisseurs. Nous présentons la formulation que nous avons choisie.

[4] Ces notions ne sont pas nécessaires pour la compréhension globale du raisonnement, en revanche elles le sont pour la justification des résultats théoriques utilisés.

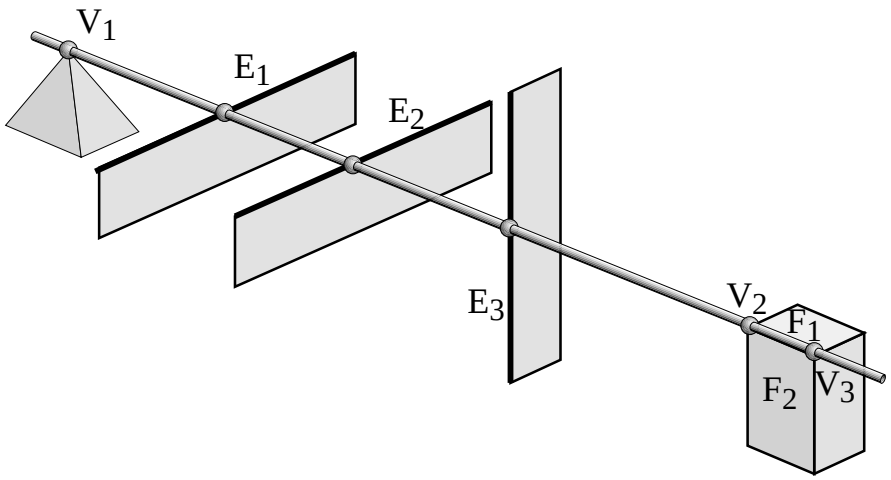


FIG. 16 – Exemple de droite poignardante extrême dégénérée. - image tirée de [Dur99]

l'établissement d'un catalogue complet. Le sommet est lui aussi redondant, en effet, il correspond à l'ensemble des arêtes qui y sont connectées. Nous verrons plus loin son utilité dans la partie énumération.

Nous pouvons dans un premier lieu nous satisfaire des deux éléments arête de la scène, ou sommet de la scène, cependant cette définition est trop restrictive. Les algorithmes et méthodes que nous allons présenter peuvent être appliqués à une définition plus générale : Un **générateur** est un élément pouvant être associé à une application linéaire dans l'espace des droites. Les droites engendrées par ce générateur sont les droites du noyau de l'application linéaire.

Dans le cas du sommet par exemple, les droites passant par V, position du sommet, sont caractérisées par $\delta = (u, v = OV \times u)$, ce qui nous donne trois équations linéaires sans second membre :

$$v_x = OV_y u_z - OV_z u_y \quad (1)$$
$$v_y = OV_z u_x - OV_x u_z \quad (2)$$
$$v_z = OV_x u_y - OV_y u_x \quad (3)$$

Pour l'arête, les équations peuvent s'écrire : $\delta \odot e = 0$, avec e la droite support de l'arête. Cette nouvelle définition de générateurs est cohérente par rapport à la précédente. Parfois, des tests supplémentaires sont nécessaires. Par exemple, le générateur arête est associé à une application linéaire dont le noyau est l'ensemble des droites coplanaires à l'arête, un test sur la position du point d'intersection entre la droite du noyau et la droite support de l'arête sont nécessaires pour dire si la droite poignarde l'arête ou non. L'ensemble des droites générées reste contenu dans le noyau de l'application linéaire : la nouvelle formulation définit algébriquement des sur-ensembles des véritables ensembles de droites poignardantes.

Bloqueur Un bloqueur est un élément géométrique de la scène. Il peut être une face, une arête ou un sommet. C'est un élément sur lequel se trouvent les points limites du segment libre maximal qui définit une droite poignardante extrême. Dans un cas générique, les bloqueurs sont des faces. Dans notre approche, nous allons conserver les éventualités improbables[5] des bloqueurs arête et sommet, nous détaillerons les techniques qui y sont liées plus loin.

Nœud Les nœuds étaient définis explicitement dans le catalogue, mais comme nous l'avons vu le catalogue ne peut plus être conservé. Nous allons donc définir les nœuds ainsi : un nœud est une droite poignardrante extrême engendrée par un ensemble de générateurs (l'ordre et le type n'interviennent pas). Cet ensemble doit suffire à la définition de la droite. De plus il est aussi défini par un bloqueur de début et de fin (éventuellement). Cette définition est plus générale et supporte les dégénérescences ; en effet, si trois sommets de la scène sont alignés, la droite passant par ces trois points est une droite poignardante extrême, et son ensemble de générateurs regroupe tous les points. Un catalogue englobant toutes les configurations possibles n'est pas envisageable, il serait infini. C'est pourquoi nous n'utilisons plus de catalogue.

Arcs Un arc est un élément reliant deux nœuds. L'information nécessaire au final pour les arcs est le couple de nœuds qu'il relie, et les bloqueurs début et fin. L'utilisation des bloqueurs est similaire au cas des nœuds car comme nous le verrons plus loin, un arc correspond à un ensemble de segments libres maximaux pour lequel les bloqueurs sont invariants. Nous ne considérons pas de générateurs pour les arcs, en effet, la donnée des bloqueurs et des nœuds nous donne rapidement (complexité $O(nlog(n))$ sur le nombre de générateurs de chaque nœud) les générateurs de l'arc. Nous verrons dans la partie applications que cette information n'est pas utilisée.

[5]Improbable est donné ici au sens probabiliste, c'est à dire événement de mesure nulle.

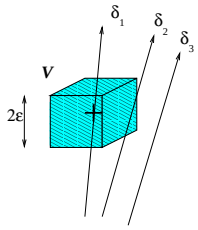
5.2 Le *presque* - Utilisation d'intervalles

5.2.1 Interactions avec les éléments de la scène

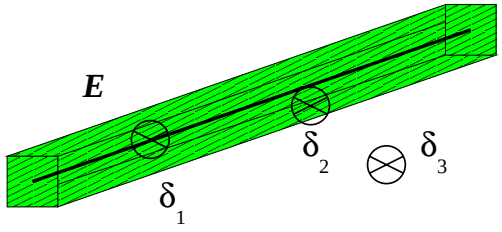
Nous sommes amenés à utiliser des droites exprimées en coordonnées de Plücker ainsi que leurs interactions avec les éléments de la scène graphique. Pour déterminer l'interaction entre une droite et un générateur, nous devons pouvoir établir le prédicat : *la droite δ passe-t-elle par le générateur g ?*. Ce prédicat n'est pas sans poser quelques problèmes de robustesse, en effet, les générateurs sont des arêtes ou des sommets, et les événements *droite passant par un sommet* et *droite passant par une arête* sont improbables, ce qui les rend difficiles à manipuler en informatique puisque la précision machine est finie.

Nous allons donc faire le choix suivant : on dira qu'une droite passe **presque** par un sommet si le sommet est à une distance [6] inférieure à un certains ϵ fixé. De même, une droite passe **presque** par une arête si le point de l'arête le plus proche de la droite est à une distance inférieure à ϵ. On a en fait grossi les éléments de la scène d'une taille ϵ. En revanche, la face n'est pas modifiée, on utilise les données en entrée comme si elles étaient une information exacte. L'interaction entre une droite et une face set la même si on l'épaissit ou non, car si l'interaction était différente, alors on aurait une arête ou un sommet poignardé (au moins).

Les interactions possibles sont donc : interaction exacte, presque ou pas d'interaction (voir figure 17). Nous reviendrons ultérieurement sur l'interprétation des interactions avec les éléments de la scène : voir phase de validation (partie 5.5).



Interaction entre un sommet et une droite : δ_1 - exact ; δ_2 - presque ; δ_3 -aucune.



Interaction entre une arête et une droite : δ_1 - exact ; δ_2 - presque ; δ_3 -aucune. Les droites sont dans la direction orthogonale au plan de projection.

FIG. 17 – Les interactions à un ϵ près.

Les interactions possibles sont résumées figure 18.

[6] N'importe quelle distance pourraît donner des résultats cohérents, cependant nous choisissons la distance *min* pour des raisons justifiées ultérieurement. Le volume de taille ϵ autout du point devient un cube aligné avec les axes d'arête 2ϵ.

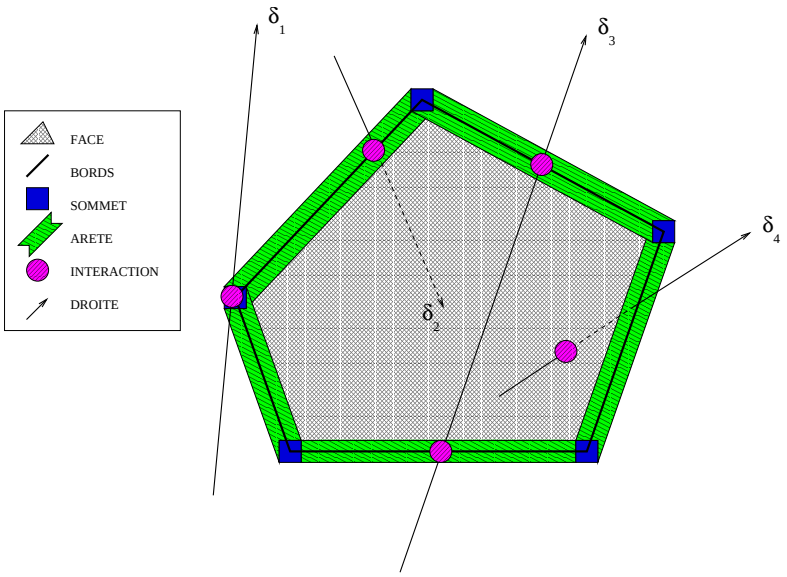


FIG. 18 – Interactions avec les éléments d'une face : δ_1 - sommet ; δ_2 - arête ; δ_3 - planaire ; δ_4 - franche (droite en intersection avec la face, approche usuelle).

5.2.2 Intervalles

Lorsque nous souhaitons déterminer l'équation de la droite passant (presque) par un ensemble de générateurs (par exemple quatre arêtes), nous devons effectuer des calculs en prenant en compte cette approche du presque contact. Pour celà, nous mettons en équation les éléments de la scène dans l'espace de Plücker à l'aide d'une arithmétique d'intervalles. C'est à dire qu'un nombre n'est plus noté par une valeur flottante mais par deux valeurs qui représentent les bornes minimale et maximale d'un intervalle. L'intervalle représente l'ensemble des valeurs valides pour la donnée représentée.

Par exemple, l'équation associée au point devient :

$$\delta = (u, v = OV \times u)$$

avec les coordonnées de V donnés en arithmétique d'intervalles : la position du sommet étant $OV = (V_0, V_1, V_2)$, les équations :

$$v_x = u_z[V_1 - \epsilon; V_1 + \epsilon] - u_y[V_2 - \epsilon; V_2 + \epsilon] \quad (4)$$

$$v_y = u_x[V_2 - \epsilon; V_2 + \epsilon] - u_z[V_0 - \epsilon; V_0 + \epsilon] \quad (5)$$

$$v_z = u_y[V_0 - \epsilon; V_0 + \epsilon] - u_x[V_1 - \epsilon; V_1 + \epsilon] \quad (6)$$

et de même pour les autres coordonnées. Il en est de même pour l'arête $[A, B]$ dont l'équation devient :

$$\delta \odot e = 0 \quad (7)$$

$$e = (u = AB, v = \frac{1}{2}(A + B) \times AB) \quad (8)$$

avec A et B donnés en comme intervalles.

Les calculs d'algèbre linéaire sont ensuite faits sur en arithmétique d'intervalles.

5.3 Phases de construction

5.3.1 Préliminaires

Nous venons de présenter une nouvelle approche théorique du squelette, cependant, la construction n'est plus intuitive. Dans le cas du catalogue, il suffisait de rechercher les instances de chaque type de nœud dans la scène, par exemple pour les nœuds du type VV, il suffisait de faire une boucle sur les paires de sommets, et on disposait de tous les nœuds VV possibles, un lancé de rayons nous donnait les bloqueurs finaux, et surtout les arcs voisins sans calculs. Cependant, en l'absence de catalogue, nous ne pouvons pas raisonnablement construire tous les nœuds possibles avec un procédé de ce type. La nouvelle méthode pourraît s'apparenter au catalogue, mais le catalogue serait infini (toute la combinatoire possible des générateurs). Bien que possible, car l'ensemble est dénombrable, et fini pour des scènes de taille finie, la complexité d'un tel procédé est très grande : la complexité est $O(2^n)$ avec n le nombre de générateurs ; il faut en effect énumérer toutes les combinaisons possibles de générateurs.

Avant de détailler une technique de construction du squellette, étudions un exemple. Supposons quatre sommets d'un maillage de coordonnées : $A = (-a, 0, 0)$, $B = (0, 0, 0)$, $C = (a, 0, 0)$ et $D = (0, 0, 1)$. Si le ϵ que l'on choisit est petit devant a, alors, les droites (D, A), (D, B) et (D, C) ne passent (au sens du presque établi précédemment) que par leurs générateurs respectifs. En revanche, si ϵ est plus grand que a, alors chacune de ces trois droite passe par au moin un autre générateur (voir figure 19).

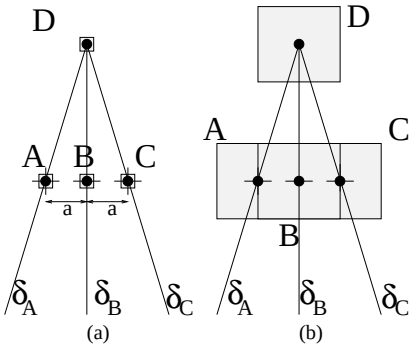


FIG. 19 – Exemple des trois points. (a) : la valeur prise par ϵ est suffisemment faible pour que le cas générique puisse encore s'appliquer, (b) : ϵ est trop grand, et les sommets sont générateurs pour plusieurs droites : A pour δ_A et δ_B, B pour δ_A, δ_B et δ_C, et C pour δ_B et δ_C.

Dans cet exemple, selon la valeur de ϵ choisie, les droites passant par des paires de sommets peuvent passer par d'autres sommets, par exemple, δ_A passe aussi par B pour une certaine valeur de ϵ. On peut donc se demander comment interpréter une telle situation dans le cadre du squelette : doit-on construire les trois nœuds, ce qui donnerait trois droites *proches* pour le squelette, ou doit-on ne conserver que quelques droites ? Cependant, si on élimine des nœuds, comment doit-on le faire ? Le choix dépendra de l'utilisation que l'on fera du squelette. Pour des applications d'ombres par exemple,

si l'on fait le choix d'une information approximative à un ϵ près, l'important est que le résultat reste cohérent ; c'est à dire que le résultat pour un ϵ donné est une approximation / simplification du résultat pour un ϵ plus petit. L'important est donc la conservation de la connectivité des nœuds en cas de suppression.

Quelque soit le choix de suppression des nœuds, le résultat, s'il est cohérent, sera valide. Nous allons donc prendre le mode de suppression correspondant à l'ordre induit par l'indexation des sommets et arêtes.

5.3.2 Générateurs V

Le générateur V est redondant, l'ensemble des droites engendrées par un sommet correspond à l'intersection des ensembles de droites engendrés par les arêtes qui y sont connectées. Cependant, le fait de le considérer comme générateur apporte plusieurs avantages : le premier est que sa mise en équation est au moins aussi simple, en effet, dès que sa valence dépasse 3, les équations des droites support des arêtes sont redondantes, et une réduction est nécessaire (qui conduit à celle du point). Par exemple, un sommet de valence 4 situé à l'origine peut être caractérisé, soit par trois équations :

$$v_x = 0 \quad (9)$$
$$v_y = 0 \quad (10)$$
$$v_z = 0 \quad (11)$$

soit par les quatres équations des arêtes de vecteurs directeurs ua, ub, uc, ud

$$v_x ua_x + v_y ua_y + v_z ua_z = 0 \quad (12)$$
$$v_x ub_x + v_y ub_y + v_z ub_z = 0 \quad (13)$$
$$v_x uc_x + v_y uc_y + v_z uc_z = 0 \quad (14)$$
$$v_x ud_x + v_y ud_y + v_z ud_z = 0 \quad (15)$$

Il est plus intéressant de considérer le premier ensemble d'équations que le second, la valence des sommets d'un maillage étant au moins 3 et le plus souvent 6 (cas régulier).

Le second, comme on le verra dans la phase d'énumération est que considérer un sommet revient à considérer simultanément plusieurs arêtes dont les droites support sont concourrantes.

5.3.3 Méthode de construction du squelette

La méthode de construction est la suivante : nous allons **énumérer** tous les ensembles de générateurs qui s'ils ne sont pas dégénérés engendrent une droite poignardante extrême, et pour chacun, chercher les générateurs touchés par cette droite ainsi que les bloqueurs début et fin. En pratique, les ensembles de générateurs non dégénérés engendrant un nœud sont de la forme : VV, VEE ou $E4$. Nous allons donc pour chaque instance de ces types d'ensemble proposer une droite poignardante extrême. La droite proposée, appelée nœud candidat, va ensuite être **validée**. La phase de validation

correspond à la recherche des générateurs supplémentaires en cas de dégénérescence, à la vérification de l'existence du nœud et à la recherche des bloqueurs début et fin.

La méthode de construction est donc la suivante : **Pour chaque instance des types de nœud VV, VEE, $E4$, valider le nœud candidat**. La phase de validation pouvant être traitée de façon générale sur tous les types de nœuds candidats, nous la séparons de la phase d'énumération pour l'exposé, cependant notons que la validation est une phase imbriquée dans l'énumération.

L'énumération des instances se fait dans l'ordre suivant. Notons que les complexités sont données à titre indicatif, ce sont des complexités théoriques du cas le pire, nous verrons des techniques pour optimiser les méthodes d'énumération :

- recherche des instances VV (complexité $O(n^2)$, n étant le nombre de sommets dans la scène)
- recherche des instances VEE (complexité $O(nm^2)$, n le nombre de sommets et m le nombre d'arêtes. ($m \simeq 3n$) avec égalité pour une triangulation régulière sans trous.)
- recherche des instances $E4$ (complexité $O(m^4)$, m le nombre d'arêtes.)

Lors de la phase de validation, nous cherchons tous les générateurs supplémentaires du nœud, pour celà, nous cherchons si un générateur touche la droite proposée, et si c'est le cas, on ajoute ce générateur à la liste des générateurs du nœud. Mais dans quelle mesure peut-on affirmer que l'on peut énumérer ainsi tous les nœuds (c'est à dire l'ensemble de générateurs de la scène) ; c'est ce que nous allons montrer dans la partie qui suit.

Justification théorique Soit un nœud ayant pour générateurs $(V_1, ..., V_n, E_1, ..., E_m)$, alors, selon les valeurs de n, il aura forcément été énuméré dans une des phases décrite précédemment : si $n \geqslant 2$, alors il a été énuméré dans la phase VV (éventuellement plusieurs fois, nous verrons comment est effectuée la gestion des redondances plus loin), si $n = 1$, alors m est forcément plus grand que 2, donc il a été énuméré dans la phase VEE, et finalement si $n = 0$, alors il a été énuméré dans la phase $E4$.

5.3.4 Gestion des redondances - suppression de certains nœuds

Dans cette méthode d'énumération, et comme nous l'avons vu dans les préliminaires avec l'exemple des trois points, certains nœuds vont être redondants par rapport à d'autres. Nous avons vu que la façon d'éliminer ces redondances importait peu tant que le squelette en sortie donnait des résultats cohérents par rapport à l'application choisie. Nous appelons générateur natif d'un nœud les générateurs considérés dans la phase d'énumération, c'est à dire les deux sommets pour les VV le sommet et les deux arêtes pour les VEE et les quatres arêtes pour les $E4$.

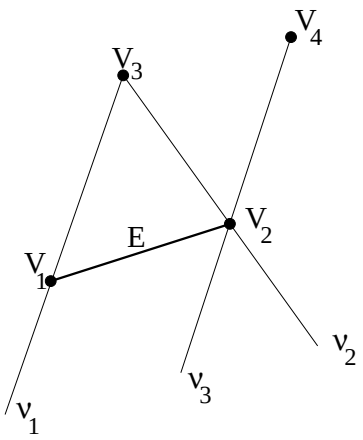
Nous allons éliminer (ne pas valider) une nœud candidat nc s'il existe un autre nœud sn, déjà calculé, tel que l'ensemble des générateurs de sn contient l'ensemble des générateurs natifs de nc. Pour ce test, on représentera le sommet par l'ensemble des arêtes qui y sont connectées, on calculera l'inclusion ou non d'un ensemble dans

l'autre. Par exemple, si sn contient comme générateurs V_1, V_2, E_1, E_2, et nc est donné par V_1, E_2, E_3, et que E_3 est arraché à V_2, alors, nc est dedondant par rapport à sn.

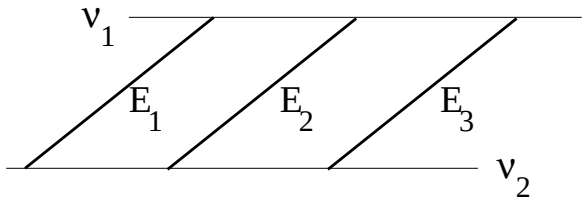
Par exemple, dans le cas des trois points, supposons que l'énumération se fasse dans l'ordre suivant : DA,DB, puis DC, alors à la création de DA, on ajoutera B comme générateur, on aura un nœud DAB dont la droite poignardante extrême sera la droite (DA), et à la proposition de DB, comme $\{D, B\} \in \{D, A, B\}$, DB ne sera pas validé. Puis on créera DC, qui lui aura en plus B comme générateur. On aura donc les deux nœuds DAB et DCB.

5.3.5 Construction des arcs

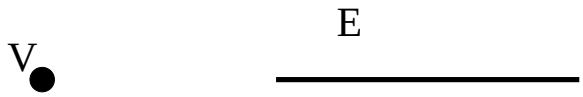
Deux nœuds sont liés par un arc si ils partagent suffisamment de générateurs. Pour définir le sens de suffisemment dans ce cas, nous devons faire appel aux idéaux et variétés algébriques.



le cas VE le plus fréquent.



un cas de EEE dont la dimension de la variété est deux.



un cas VE dégénéré : V est contenu dans la droite support de E. La dimension de la variété est 0, donc pas d'arc à priori.

FIG. 20 – deux exemples illustrant le critère de suffisance

Exemple Considérons deux exemples (figure 20). Dans le premier cas (à gauche sur la figure), les nœuds ν_1 et ν_2 partagent tous deux les deux générateurs V_3 et E. Ces deux générateurs sont suffisants à la génération d'un arc (cet arc figurait dans le catalogue). En revanche, ν_1 et ν_3 ne partagent que E, ce qui n'est pas suffisant, il en est de même pour ν_2 et ν_3 qui ne partagent que V_2.

Dans le second cas (à droite sur la figure), les trois E partagés par ν_1 et ν_2 pourraient suffire à définir un ensemble de droites critiques, s'ils étaient en position générique ; cependant, étant tous dans le même plan, l'ensemble de droites qu'ils engendrent est de dimension 2, ce qui est trop grand pour définir un arc ! Nous allons généraliser ces exemples pour donner un critère de suffisance.

Idéal des droites réelles normées Toutes les paramétrisations des droites dans l'espace de Plücker ne correspondent pas à des droites réelles, une paramétrisation ne correspond à une droite réelle que si elle est de la forme $\delta = (u, v)$ avec $u \cdot v = 0$. De plus, les paramétrisations de Plücker sont définies à une constante positive près ce qui en fait un espace projectif. Notons l'équation de normalisation : $u \cdot u = 1$, une paramétrisation vérifiant cette équation est dite normée. Ces deux équations quadratiques engendrent un idéal dont la variété des zéros est une variété algébrique de dimension 4 qui définit les paramétrisations normées des droites de l'espace. On appelle cet idéal l'idéal des droites réelles normées.

Idéaux et variétés associés aux générateurs Comme nous l'avons vu précédemment, l'ensemble des droites passant par un générateur (passant est ici pris au sens exact), peut être mis en équation. Les solutions de ces équations peuvent être interprétées commme des racines de polynômes (de degré un). Donc on peut associer à un générateur un idéal, ainsi que la variété de ses zéros.

Critère de suffisance Soit $G = g_1, ..., g_n$ un ensemble de générateurs, soient $v_i = v(g_i)$ leurs variétés associées. Un ensemble de générateurs est suffisant pour engendrer un arc si la dimension de la variété $V(G) = \bigcap_{i=1}^{n} v_i$ est égale à un. Dans les cas dégénérés, par exemple le cas des trois points exprimé précedemment, deux sommets sont partagés, d'où une dimension zéro, mais une variété non vide ; il serait de plus souhaitable de lier ces nœuds. Nous donnons donc comme critère de suffisance :

$$dim\left(\bigcap_{i=1}^{n} v(g_i)\right) \leqslant 1$$

Dans le cas du premier exemple, la variété associée à $\{V_3, E\}$ est de dimension 1, les deux générateurs sont donc suffisants et ν_1 et ν_2 sont reliés par un arc. En revanche, dans le second exemple, la variété associée à $\{E_1, E_2, E_3\}$ est de dimension 2, ce qui ne satisfait pas au critère de suffisance.

Redondance La redondance est évitée trivialement : si deux nœuds sont déjà liés, on ne les relie pas.

5.4 Enumération des nœuds et arcs du squelette

5.4.1 Les VV

Les nœuds VV sont les plus simples à énumérer : une boucle sur toutes les paires possibles est faite, et si les sommets sont à une distance supérieure à ϵ l'un de l'autre, alors on propose un nœud candidat pour le couple VV. Si ils sont à une distance inférieure à ϵ, alors deux éléments sont à noter. Le premier est que le calcul des coordonnnées de Plücker de la droite passant par les deux sommets sera d'autant plus imprécis que ϵ sera petit. Le second est que si un nœud est défini par l'un des deux sommets, alors forcément, la droite touchera presque le sommet, donc on aura un nœud dégénéré. La complexité est quadratique (et logarithmique pour la validation). Tous les couples de nœuds devant être traités, et aucune information de visibilité n'étant disponible à ce stade (c'est ce que l'on est en train de calculer), aucune optimisation n'est possible pour cette énumération.

5.4.2 Les VEE

Les nœuds VEE peuvent être énumérés de façon naïve, comme les VV, c'est à dire en testant tous les triplets (sommet, arête, arête) possibles, ce qui donne une complexité en $O(n^3 log(n))$ avec la validation. Cependant, une optimisation, proposée par Durand et al. [DDP97] peut être utilisée ici. En effet, on peut effectuer une recherche sur les paires d'arêtes et ne tester que les sommets se trouvant dans le sablier, défini par l'ensemble des droites touchant les deux arêtes. Pour ce faire, nous construisons trois grilles à deux dimensions : une dans chaque plan (x, y), (x, z), (y, z) (voir figure 21). Dans chaque cellule de ces grilles nous stockons la liste des sommets se projettant dans la cellule (plus exactement, la boîte alignée avec les axes de taille ϵ). Ensuite, nous calculons la projection des deux arêtes grossies d'une taille ϵ, et nous calculons le sablier projetté dans chacune des grilles 2D. En effet, seuls les sommets se trouvant à la fois dans chacun des sabliers des trois grilles 2D peuvent être contenus dans le sablier tridimensionnel (en fait c'est un sur-ensemble des sommets réellement contenus dans le sablier). Ensuite, pour chacun de ces sommets, nous construisons la droite passant par ces trois générateurs, et la proposons pour la phase de validation.

Une autre technique consisterait à stocker les sommets dans un arbre octaire (octree), et à découper la structure hiérarchique selon les quatres plans du sablier. Cette technique permettrait aussi de trouver l'ensemble des sommets potentiellement générateurs pour un couple d'arêtes.

Enfin une dernière technique que nous utiliserons pour le cas de la source point, consiste à boucler sur les sommets et les arêtes, puis à rechercher les arêtes en intersection avec la section angulaire définie par les droites passant par le sommet et l'arête, grâce à une structure hiérarchique sur les arêtes. Cette approche, similaire dans son concept aux deux précédentes permet cependant de ne s'intéresser qu'aux nœuds du squelette ayant un sommet donné pour générateur, ce que les autres techniques ne permettent pas efficacement.

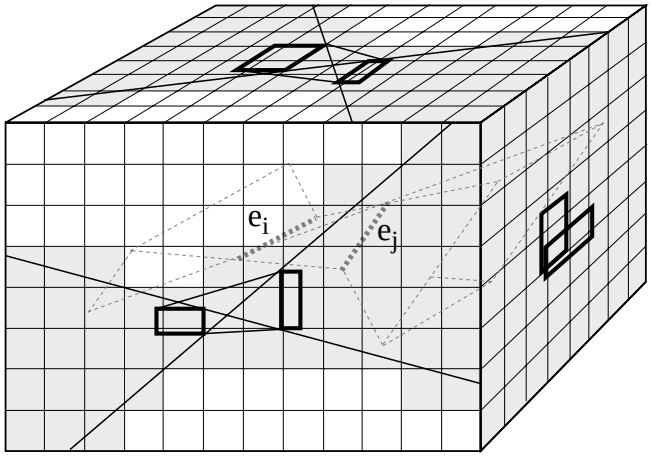


FIG. 21 – Les trois grilles à deux dimensions, optimisation proposée par Durand et al. image extraite de [Dur99]

5.4.3 Les E4

Les quadruplets d'arêtes sont énumérées naïvement par quatre boucles imbriquées sur toutes les arêtes. La complexité est donc $O(n^4 r(n))$ avec la validation, ce qui devient rapidement trop grand. Cependant, comme proposé par Durand et al. dans [DDP97], on peut utiliser une structure de sablier : on boucle sur les couples d'arêtes, on ne considère que les arêtes en intersection avec le sablier, puis on boucle sur ces arêtes, ce qui nous donne des triplets d'arêtes : (e_i, e_j, e_k). On recherche ensuite les arêtes qui sont à la fois dans les trois sabliers engendrés par les trois couples d'arêtes : $(e_i, e_j), (e_j, e_k), (e_k, e_i)$. On obtient alors des quadruplets d'arêtes.

Plusieurs méthodes de calcul de droite passant par quatre arêtes ont été proposées. Durand et al. [DDP97] ont proposé une méthode dichotomique, Devillers et Hall-Holt [DHH01] ont proposé une technique de calcul directe efficace, Teller [Tel92b] a proposé une méthode matricielle à base de décomposition en valeurs singulières. Nous proposons ici une méthode matricielle inspirée de celle de Teller, mais sans décomposition en valeurs singulières complète.

Mise sous forme matricielle du problème Afin de déterminer si les quatres arêtes engendrent une (ou deux) droites poignardantes extrêmes, nous utilisons une technique inspirée par Teller [Tel92b]. L'idée est la suivante : considérant les quatres droites $\delta_0, \delta_1, \delta_2, \delta_3$, support des quatres arêtes, une droite poignardante extrême δ engendrée par les quatres arêtes doit en particulier satisfaire :

$$\delta \odot \delta_0 = \delta \odot \delta_1 = \delta \odot \delta_2 = \delta \odot \delta_3 = 0$$

Pour une droite $\delta(u, v)$, on note $e(v, u)$ sa droite *swappée*[7]. Si δ est réelle, alors $\delta \cdot e = 0$. Les équations précédentes peuvent alors s'écrire :

[7] soit $\delta = (u, v)$ une droite, sa droite *swappée* est donnée par $\delta\prime = (v, u)$.

$$\delta \cdot e_0 = \delta \cdot e_1 = \delta \cdot e_2 = \delta \cdot e_3 = 0$$

Ces équations peuvent alors se mettre sous forme matricielle, et la droite que nous cherchons est alors une droite du noyau de l'application linéaire associée à cette matrice. Teller propose une décomposition en valeurs singulière pour trouver la dimension du noyau ainsi que deux vecteurs de base de ce noyau. En réalité, un pivot de Gauss nous permet de calculer la dimension et on déduit ensuite à l'aide d'une réduction de Gauss complète deux vecteurs du noyau. La forme matricielle est alors la suivante :

$$\begin{bmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \\ v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Calcul des droites candidates Une fois ces deux vecteurs (notés δ_a et δ_b) connus, on sait que la droite poignardante extrême, si elle existe, est de la forme $\delta = \lambda\delta_a + \mu\delta_b$, car elle est forcément un élément du noyau. Nous devons alors rechercher un couple de valeurs λ et μ tels que la droite soit réelle [8], et normée [9]. Nous allons rechercher les couples de valeurs valides.

Si les deux droites étaient réelles, alors le noyau serait de dimension deux sur les droites réelles, et donc de dimension un sur les droites normées. L'équation quadratique ne peut pas être un élément d'un idéal engendré par des arêtes car il possède un terme constant. Or si les quatre droites ne sont pas en position dégénérée, elles sont poignardées par un nombre fini de droites normées (deux au maximum). D'où l'impossibilité pour les droites du noyau d'être toutes deux réelles simultanément.

Si une droite est réelle et l'autre imaginaire, alors toute combinaison linéaire avec un coefficient non nul pour la droite imaginaire est une droite imaginaire. Il suffit donc de normer la droite réelle du noyau pour trouver la droite poignardante extrême (unique au sens près) touchant les quatres droites.

Si les deux droites sont imaginaires, alors aucun des deux coefficients de la combinaison linéaire ne peut être nul. On cherche donc les paramètres t pour lesquels la droite $\delta(t) = \delta_a + t\delta_b$ est réelle. Pour celà on calcule $\delta(t) \odot \delta(t)$ ce qui nous donne un polynôme de degré deux en t :

$$\delta_b \odot \delta_b t^2 + \delta_a \odot \delta_b t + \delta_a \odot \delta_a = 0$$

Si il existe des racines réelles (au sens usuel des nombres réels) de ce polynôme, alors, il existe des droites réelles dans le noyau. Il suffit ensuite de les normaliser pour obtenir la ou les deux droites poignardantes extrêmes.

[8] un 6-uplet de coordonnées dans l'espace de Plücker est appelé droite réelle si il correspond à la paramétrisation d'une droite de l'espace. Sinon, on dit que c'est une droite imaginaire.

[9] une droite $\delta = (u, v)$ est normée si $u \cdot u = 1$.

Gestion du presque contact dans la recherche Nous devons aussi chercher à l'aide de cette méthode les droites qui passent presque par les arêtes. Pour ce faire, nous prenons une paramétrisation des droites support des arêtes en arithmétique d'intervalles, ce qui nous permet de considérer en une seule passe toutes les droites passant presque par les quatres arêtes. La réduction de Gauss est effectuée à l'aide de pivots numériques (pas intervalles), ce qui nous donne des solutions en arithmétique flottante, cohérentes avec la position du problème. Les solutions proposées passent presque par les arêtes, et sont données en arithmétique flottante.

Validation Finalement, on vérifie si la (les) droite(s) trouvées passent effectivement par les arêtes, puis on passe à la phase de validation.

Cette méthode est intéressante car elle est robuste, et elle permet de prendre en compte notre approche à base d'ϵ. En effet, cette méthode nous permet d'utiliser des intervalles pour représenter nos données en entrée, et de calculer un résultat cohérent avec ces hypothèses.

5.4.4 Les arcs

Nous avons vu que les arcs sont déterminés par couples de nœuds. Cependant, l'énumération quadratique sur les nœuds est une opération très couteuse (complexité de l'ordre de $O(n^4)$ sans la recherche des bloqueurs. Nous allons donc proposer une technique différente. Observons tout d'abord que chaque arc s'appuie sur au moins une arête, c'est à dire que chaque arc correspond à un ensemble de droites critiques de dimension un dont toutes les droites touchent une arête donnée.

Nous allons donc effectuer une boucle sur les arêtes et pour chaque arête rechercher les arcs qui s'appuient dessus. Un arc peut s'appuyer sur plusieurs arêtes, mais comme nous l'avons vu plus haut (5.3.5), tester si un arc a déjà été crée est une opération simple. Nous allons énumérer les ensembles de droites critiques de dimension un s'appuyant sur une arête, et en déduire les arcs.

Ensembles de droites critiques - *critical line set* Afin de déterminer les ensembles de droites critiques de dimension un s'appuyant sur l'arête, nous énumérons l'ensemble des nœuds qui touchent l'arc. Puis pour chaque paire de nœuds, nous calculons l'ensemble de générateurs communs aux deux nœuds (pour ce faire, nous considérons tous les générateurs, c'est à dire que les arêtes connectées aux sommets générateurs sont aussi comptabilisées). Pour chacun de ces ensembles, nous calculons la dimension de la variété associée, c'est à dire la dimension de l'intersection des variétés associées aux générateurs de l'ensemble. Si cette dimension est inférieure à deux (strictement), alors ces générateurs définissent un ensemble de droites critiques (éventuellement de dimension zéro dans certains cas particuliers de dégénérescence). Nous obtenons donc une liste d'ensembles de droites critiques. Aucune expression des ensembles de droites critiques n'est nécessaire, il suffit de stocker les générateurs qui l'engendrent.

Arcs candidats Pour chaque ensemble de droites critiques, nous déterminons les nœuds dont les droites poignardantes extrêmes sont inclus dans l'ensemble. Pour celà, nous testons si l'ensemble des générateurs du nœud contient celui de l'ensemble de droites critiques. Si c'est le cas, la droite poignardante extrême associée au nœud est un élément de l'ensemble de droites critiques.

Le paramètre d'un nœud par rapport à une arête (génératrice de ce nœud) est défini ainsi : soit $[A, B]$ le segment de l'arête, soit I le point d'intersection entre l'arête et la droite (ou plus généralement le point de l'arête le plus proche de la droite), le paramètre t est donné par

$$I = (1 - t)A + tB$$

Nous trions la liste des nœuds de l'ensemble de droites critiques par ordre croissant de paramètre. Pour chaque paire de nœuds consécutifs, nous créons un arcs candidat. Un ensemble de droites critiques peut donner naissance à plusieurs arcs.

Validation Une fois l'arc candidat proposé, il est validé si aucun arc entre les deux nœuds n'existe. Ensuite, on recherche les bloqueurs de début et de fin. Pour celà, on établi l'ensemble des bloqueurs potentiels, c'est l'ensemble des faces connectées aux générateurs et bloqueurs des nœuds privé de l'ensemble des faces connectées aux générateurs de l'ensemble de droites critiques. Et pour chaque face, on lance un rayon au milieu entre les deux noeuds, et on teste (sans utiliser de critère à base d'ϵ) si le rayon est bloqué ou non. Ce test est effectué parmi des bloqueurs potentiels énumérés à partir des faces et arêtes connectées aux deux nœuds. On trouve ainsi les bloqueurs de début et de fin.

5.5 Validation

La phase de validation consiste à déterminer les interactions d'un rayon critique : une droite poignardante extrême, avec le reste de la scène. Nous effectuons ces calculs pour savoir si la droite définie par certains générateurs porte bien un segment libre maximal s'appuyant sur les générateurs. Si c'est le cas, l'ensemble de générateurs donnera naissance à un nœud, auquel sera associé un segment libre maximal critique.

L'algorithme de validation prend en entrée les paramètres suivants[10] :

– un nœud candidat

– un point de l'espace traversé par le nœud candidat

Le second paramètre est un point de la droite se situant entre deux générateurs du nœud.

Rappelons les éléments du nœud candidat :

– une paramétrisation de la droite dans l'espace de Plücker

– l'ensemble des générateurs qui ont engendré le candidat

L'algorithme est le suivant : on lance un rayon depuis le point donné dans chaque direction. Pour chaque face de la scéne touchée par le rayon (à un ϵ près), calculer le type d'interaction avec la face, et selon, enregistrer un élément comme générateur ou bloqueur.

Cette phase est la partie centrale de notre algorithme. Le caractère nouveau de l'approche s'exprime dans la façon d'effectuer les tests qui suivent. Cette technique nous permet de calculer à la fois les générateurs non natifs des nœuds (en dehors de ceux donnés par l'enumération), ainsi que les bloqueurs.

5.5.1 Interactions avec les faces

Les interactions possibles avec les faces sont illustrées figure 22.

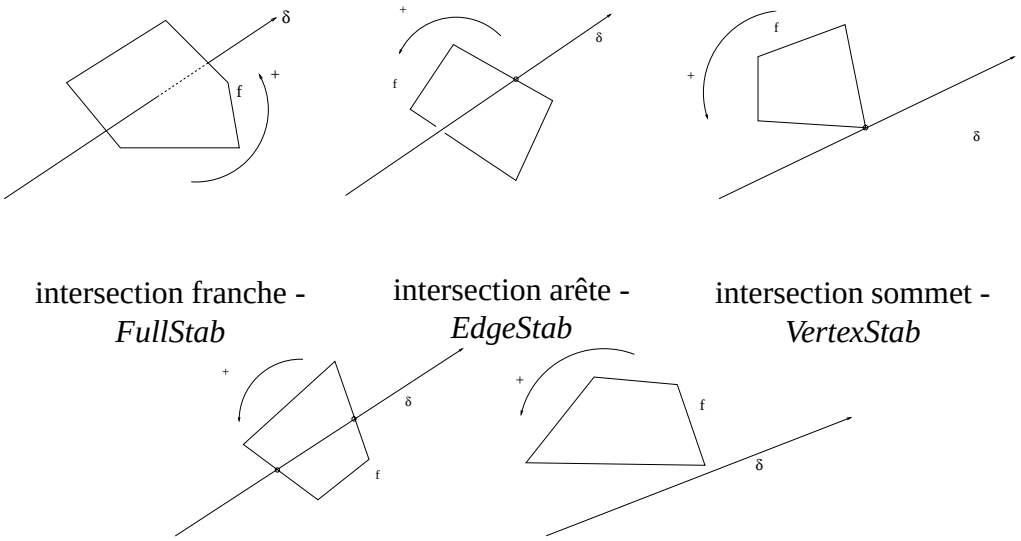
Pour chacune d'entre-elles, nous allons en donner une caractérisation ainsi que les modifications impliquées dans les listes de bloqueurs et générateurs.

Notons tout de même une nouvelle notion, celle des objets poignardés (*stabbed*). Un objet poignardé est un élément en interaction avec une droite : sommet, arête ou face. A ce stade des calculs, on ne peut savoir si un objet poignardé est bloqueur ou générateur, ou aucun des deux. Dans chaque cas, des test supplémentaires seront nécessaires.

5.5.2 Caractérisation des interactions avec les faces

Afin de calculer les interactions, nous allons mettre en équation le problème de calcul d'intersection. Pour celà, nous allons nous placer dans le cadre des coordonnées de Plücker, en arithmétique d'intervalles. La méthode que nous utilisons est inspirée de celle développée par Seth Teller dans son article [Tel92a]. La face, supposée convexe, est modélisée ici par une suite de droites orientées portant les arêtes ; l'orientation des droites étant telle que deux droites ne se dirigent pas vers le même point simultanément

[10] On suppose que la scène ne contient pas de générateurs qui ne soient pas attachés à des faces, c'est à dire des sommets seuls, ou des arêtes seules. Cette hypothèse est d'autant plus cohérente que le modèle de maillage représentant une surface ne permet pas de représenter de tels objets.



intersection franche - *FullStab*

intersection arête - *EdgeStab*

intersection sommet - *VertexStab*

intersection planaire - *FaceStab*

pas d' intersection - *NoStab*

FIG. 22 – interactions possibles entre une face et une droite

(voir figure 23 pour illustration). Cette suite de droite les appelée anneau de droites (*line-ring*). Pour la suite, on note δ la droite candidate.

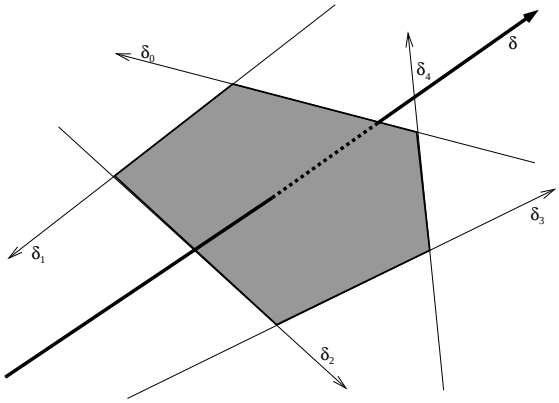


FIG. 23 – anneau de droites pour une face $\{\delta_0, \delta_1, \delta_2, \delta_3, \delta_4\}$ - *line-ring*

Chacune des droites est exprimée en coordonnées de Plücker par des intervalles. Si la forme bilinéaire de Plücker entre une droite de l'anneau et la droite candidate est nulle, alors elles sont coplanaires.

NoStab Ce cas correspond à l'absence d'interactions. Cette décision est prise par défaut, dans le cas où aucune autre ne peut être mise en évidence.

FullStab C'est l'interaction la plus simple : toutes les formes bilinéaires de Plücker doivent être du même signe, positives ou négatives selon le sens. Dans cette situation, la face est ajoutée comme un bloqueur (dans les deux directions).

EdgeStab Cette interaction peut être caractérisée d'une seule façon : seule une arête est en intersection avec la droite δ, c'est à dire que la droite portant l'arête est en intersection avec δ, et que le point d'intersection se situe sur la droite. Dans ce cas, on ajoute l'arête à la liste des objets poignardés.

VertexStab Cette interaction peut être caractérisée de plusieurs façons (redondantes) : soit δ est en intersection avec deux arêtes consécutives, alors on a un *VertexStab* avec le point partagé, soit on teste si la droite passe par le sommet, alors, on a aussi le résultat. Dans ce cas, on ajoute le sommet à la liste des objets poignardés.

MultiStab Cette interaction est caractérisée par le fait que l'ensemble des formes bilinéaires de Plücker sont nulles, et qu'il existe au moins un point de la face sur la droite. Dans ce cas, deux éléments doivent être extraits : celui de début et celui de fin d'intercation avec la droite. Ils peuvent être sommet ou arête. Les deux éléments sont ajoutés à la liste des objets poignardés.

5.5.3 Cohérence spatiale

La phase de validation s'appuie sur le principe d'un lancé de rayons. Certes le calcul de l'interaction avec une face est différent d'un tracé de rayons classique, mais il est tout de même souhaitable d'utiliser la cohérence spatiale de la scène pour les tests d'interaction. Nous allons utiliser une structure d'accélération basée sur les boîtes alignées avec les axes englobant les faces (grossies dans la direction des axes). La litterature dans le domaine du lancé de rayons est très riche. Cazals et al. [CDP95] ont proposé une technique basée sur une hiérarchie de grilles uniformes qui semble avoir des performances très intéressantes. Amanatides et Woo [AW87] ont présenté une technique efficace de parcours de grille régulière.

Nous avons implanté les structures d'accélération suivantes : grille régulière, arbre octaire, grille récursive.

5.5.4 Les objets poignardés - *stabbed*

Chaque objet poignardé interagit avec la droite et lors du calcul de cette interaction, nous allons déterminer sa nature : l'objet est soit un bloqueur soit un générateur.

Arête poignardée - *Stabbed Edge* Dans le cas d'une arête, la situation est binaire : soit elle bloque, soit elle génère ; c'est à dire que dans le premier cas, on va ajouter un bloqueur à la liste des bloqueurs, et dans le second, on ajoutera un générateur (le nœud sera dégénéré). Les deux cas sont illustrés figure 24.

Afin de caractériser ce cas, nous étudions la position de chacune des faces relativement à la droite ; pour celà, nous testons si l'arête est une arête silhouette relativement à la direction de la droite. Ce test se fait en calculant le signe du produit scalaire des normales aux faces attachées à l'arête, s'ils sont de même signe, alors l'arête est bloquante. Nous effectuons ce test de façon robuste, c'est à dire en envisageant les cas pour lesquels les calculs seraient faussés par des imprécisions numériques.

Sommet poignardé - *Stabbed Vertex* Dans le cas d'un sommet, la situation est également binaire : soit le sommet est bloquant, soit il est générateur. En revanche, la situation est plus délicate que précédemment, en effet pour l'arête, il suffisait de tester si le "dessus" et le "dessous" étaient bloqués, maintenant, il faut tester toutes les directions ! Pour celà nous allons introduire un nouvel outil : les tranches (*slices*).Nous allons comme précedemment poser au niveau du sommet un repère local, puis construire des tranches de directions autour de ce point (voir figure 25).

Une tranche est définie par deux directions (notées de façon redondante par les deux coordonnées dans le repère local x, y pour des raisons de continuité). Pour chaque face attachée au sommet V, une tranche est construite :

- on calcule des coordonnées locales de la projection de chaque arête sur le plan Π (représentée par N' sur la figure 25). Le plan Π étant le plan orthogonal à δ, passant par le sommet V.
- on détermine ensuite la position min et la position max de façon à ce que la tranche ainsi construite ne dépasse pas un angle de π. La tranche est l'ensemble des directions dans le plan Π entre la direction min et la direction max.

Remarquons que l'orientation de la face n'intervient pas ici, ce qui donne une certaine robustesse à notre algorithme. Il supporte en effet les maillages dont l'orientation des faces est mauvaise.

Ensuite, afin de déterminer si un sommet est bloquant ou non, on tente de fusionner les tranches ainsi construites. En fait, une tranche correspond à un intervalle d'angle polaire, qui sont les angles polaires occupés par la projection des faces sur la plan Π. Nous allons calculer l'union de ces intervalles, et si le résultat recouvre l'intervalle

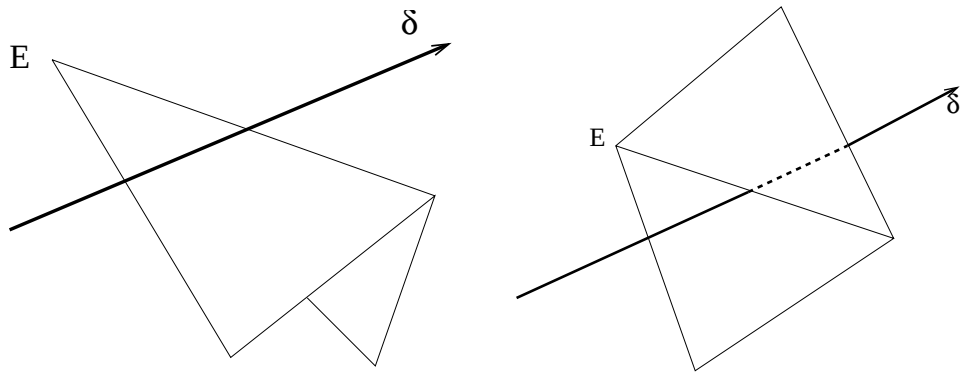


FIG. 24 – Illustration des deux cas de Edge Stab : premier : générateur, second : bloqueur

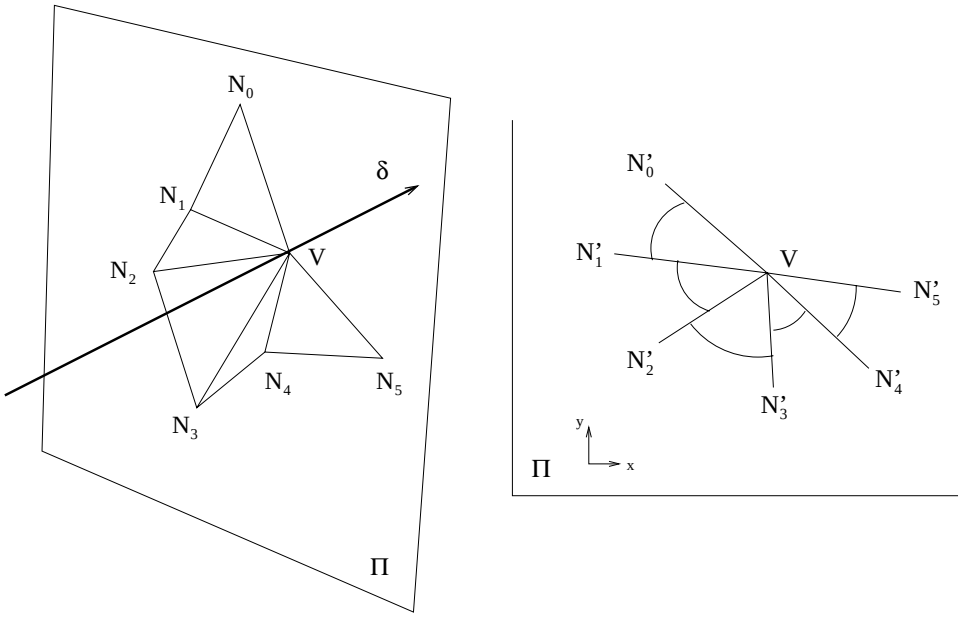


FIG. 25 – Illustration des tranches

$[0; 2\pi]$, alors, le sommet est bloquant, sinon c'est un générateur.

Pour des raisons de précision machine, nous introduisons un ϵ, qui nous permet de fusionner des tranches presque collées. Cet ϵ est très faible (de l'ordre de la précision des flottants utilisés). Les coordonnées dans le plan de projection sont normalisées pour conserver cet ϵ cohérent.

Nous pouvons donc, à l'aide de cette technique déterminer la nature d'un sommet poignardé.

Face poignardée - *Stabbed Face* Ce cas est un peu particulier. Dans une première approche, nous allons juste considérer qu'il correspond à deux objets poignardés, ceux définis précédemment : les générateurs interagissant au début et à la fin. Mais en réalité, la situation est quelque peu différente. Une étude plus correcte de ce cas est donnée lors de la présentation de la multiface (5.5.6). Notons simplement que ce cas est un autre cas d'objet poignardé.

5.5.5 Évolution de l'algorithme - performances

Quant un bloqueur de la droite est touché par un rayon, la recherche s'arête, c'est à dire que l'on ne teste pas les interactions avec les faces au delà du bloqueur. Cette approche est possible à l'aide de la grille, ainsi que l'ordre donné lors de son parcours, en effet le coût d'un calcul d'intersection entre un rayon et une face et faible devant le coût d'un calcul d'interaction. On ordonne donc l'ensemble des faces ayant une interaction possible (boîte englobante) avec le rayon (intersection du plan coupé par la boîte), et on parcours cet ensemble dans l'ordre de position sur la droite.

Un nœud candidat n’est valide que s’il a touché, lors des tests d’interaction avec les faces, tous ses générateurs.

Cette méthode de recherche est performante car elle permet de bénéficier des structures d’accélération du lancé de rayons, et elle n’effectue les tests que sur les éléments nécessaires. La complexité d’une telle méthode de recherche est logarithmique sur le nombre d’éléments de la scène (sans compter la phase de construction de la grille).

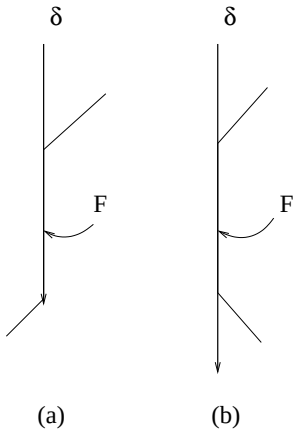
5.5.6 Gestion des Faces poignardées - *Multiface*

Position du problème Lors des calculs d'interaction entre une droite poignardante extrême candidate et une face, on a vu que l'on pouvait rencontrer des situations avec plusieurs éléments (sommet, ou arête) poignardés. Ces cas peuvent apparaître dans diverses situations :

- la droite est contenue dans le plan support de la face
- la face est de taille apparente inférieure à ϵ

Chacun de ces cas est issu d'une configuration particulière de la scène au voisinage de la face considérée.

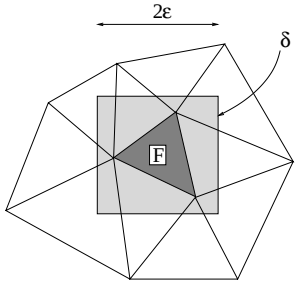
Dans le premier cas, nous devons identifier si l'un des deux éléments poignardés est bloquant (voir figure 5.5.6).



Ci-contre, coupe dans le plan contenant la droite et orthogonal à la face.
La droite est tangente à la face, chacun des élément poignardés par la droite ne serait pas, seul, bloquant, cependant l'union des deux est parfois bloquante. Dans la configuration (a), les éléments poignardés sont bloquants, dans la configuration (b), ils ne le sont pas. La face F joue donc un rôle particulier ici.

FIG. 26 – La droite est contenue dans le plan support de la face

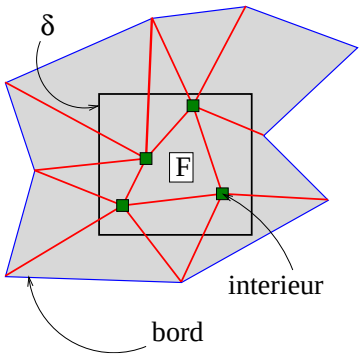
Dans le second cas, nous devons savoir comment interprêter la situation d'une face petite par rapport à l'ϵ fixé. (voir figure 5.5.6).



Ci-contre, projection dans le plan orthogonal à la droite du voisinage de la face F. La droite est supposée alignée à un axe pour la clarté de l'illustration.
La face seule ne bloque pas la droite grossie, cependant, dans le cas exposé ci-contre, la face à laquelle on adjoint ses faces voisines forme un ensemble de faces qui bloquent la droite.

FIG. 27 – La face est petite par rapport à ϵ

La multiface Au lieu de considérer les faces individuellement, nous allons les regrouper selon certains critères et les traiter collectivement. Nous allons donc procéder ainsi : si une face a une interaction multiple avec une droite, on construit une structure de type *multiface* que l'on initialise avec la face. Ensuite, on effectue une extension de la multiface au voisinage de la face selon les règles définies ci-après, et on teste si la multiface bloque la droite.



La structure de la multiface est la suivante :
– un ensemble de faces *faces*
– un ensemble d'arêtes *bord*
– un ensemble de sommets *intérieur*

La structure est construite par rapport à une droite donnée. L'ensemble *faces* est la composante connexe contenant la face ayant initié la multiface. *bord* correspond à la frontière de *faces*, et *intérieur* est l'ensemble des sommets des faces de *faces* qui ne sont pas sur *bord*. voir ci-contre pour illustration.
Les arêtes en rouge (celles qui ne sont pas au bord), sont appelées arêtes internes.

Les arêtes internes ont les propriétés suivantes :
– Elles sont touchées par la droite δ.
– Elles ne sont pas silhouette pour la droite δ.

On exclue le cas des arêtes silhouette internes car celà les empêcheraît d'être génératrices.

La construction de la multiface se fait par relaxation sur les arêtes internes : on commence avec pour liste d'arêtes internes non traitées la liste des arêtes de la face, qui vérifient les propriétés précédentes. Puis pour chaque arête interne, on ajoute à la liste des faces les faces connectées non encore explorées. Pour chaque face ainsi ajoutée, on ajoute les arêtes de la face non encore traitées dans la liste de relaxation. Puis on itère.

Chaque face de la multiface est en interaction avec la droite, mais cette interaction est déterminée dès la construction de la multiface. Les calculs liés à l'interaction ne sont donc plus à faire.

Interactions avec la multiface Une fois la multiface construite, on souhaite déterminer son interaction avec la droite. Pour celà, nous allons utiliser une technique voisine de celle du sommet : nous allons calculer la portion angulaire de la droite bloquée. Cette portion est déterminée en construisant des faces *virtuelles* entre un point de la droite et les arêtes de bord. Cette technique permet de calculer si dans la direction de la droite les arêtes de bord entourent la droite ou non. Dans la figure 28, sont exposés les deux cas : la multiface se comporte comme un générateur, ou comme un bloqueur.

On considère un point de la droite comme sommet virtuel, et on attache à ce sommet des faces virtuelles constituées chacune du sommet et d'une arête de bord. On considère alors le sommet ainsi construit comme un sommet du maillage, et on teste s'il est bloqueur ou générateur, comme décrit précédemment.

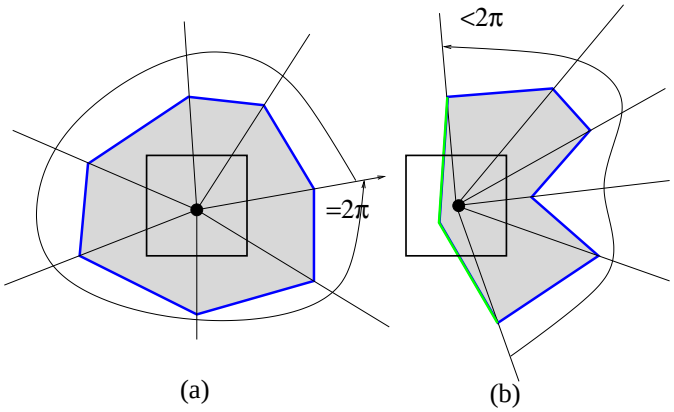


FIG. 28 – Deux configurations de multiface : (a) : la multiface est un bloqueur pour la droite, (b) : la multiface est un générateur pour la droite. Le sommet virtuel est représenté par un disque au centre de la figure.

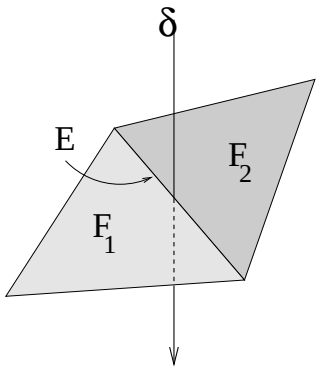
Problème de la multiface La multiface utilise fortement la connectivité. Il n'est donc pas possible de l'envisager pour des soupes de polygones tel quel. Il serait intéressant d'étudier le problème de la multiface pour des soupes de polygones, cependant nous ne l'avons pas fait ici.

5.5.7 Gestion des contacts - Eventail de bloqueurs - *Blocker Fan*

Nous avons vu que l'on souhaite pouvoir traiter tous types de scène, en particulier celles ayant des objets en contact. Mais aussi, un autre exemple de contact est constitué par les faces partageant une arête, mais dont cette information de connexion a été perdue ou jamais modélisée (soupe de polygones).

Afin de traiter ces cas, nous allons proposer un nouvel outil : l'évantail de bloqueurs. Il s'inspire de la technique proposée pour le bloquage des droites par des sommets (ou des multifaces). L'idée est de construire un éventail de bloqueurs ayant des tranches **épaisses**. Comme pour le test de bloquage avec les sommets, on va construire des tranches, auxquelles on va donner une épaisseur. On peut voir la droite comme un tube dont on enlève des tranches, si le tube est coupé, alors on a trouvé un bloqueur.

Exemple de connexion perdue Nous allons étudier un exemple dans lequel une connexion (ou presque connexion) entre deux faces a été perdue.



Prenons l'exemple de deux faces F_1 et F_2 dont la connexion a été perdue, elles partageaient une arête E. Une droite δ poignarde les deux faces sur cette arête E. Ni F_1 ni F_2 ne bloquent δ puisque pour chacune, l'arête est considérée comme étant seule, elle est silhouette. Cependant, l'arête si était partagée, elle serait bloquante.
Pour retrouver cette situation, nous construisons deux tranches (de taille π chacune car ce sont des arêtes poignardées), et nous les munissons d'une épaisseur ϵ puisque l'interaction se fait seulement au niveau de l'arête. Les deux tranches peuvent fusionner pour donner une tranche pleine, et elles partagent une position sur la droite. Elles bloquent la droite. Cette approche nous permet de retrouver le caractère bloquant de cette arête.

L'éventail de bloqueurs L'éventail de bloqueurs est une structure permettant d'accumuler les tranches épaisses et de signaler une éventuelle completion engendrant un bloquage de la droite actuellement étudiée.

Pour construire cet éventail, on ajoute une tranche épaisse pour chaque générateur de la droite. La tranche épaisse est déterminée selon le type de générateur. Nous avons trois cas possibles : le sommet, l'arête, et la multiface.

Le sommet génère une (ou plusieurs) tranche(s) d'épaisseur ϵ déterminée(s) par les faces qui y sont attachées (comme pour le test de bloquage). L'arête génère une tranche d'épaisseur ϵ et de taille π. La multiface génère une tranche de façon analogue au sommet ; son épaisseur est déterminée par l'intervalle occupé sur la droite par ses éléments internes : les arêtes et les sommets. voir figure 29 pour illustration.

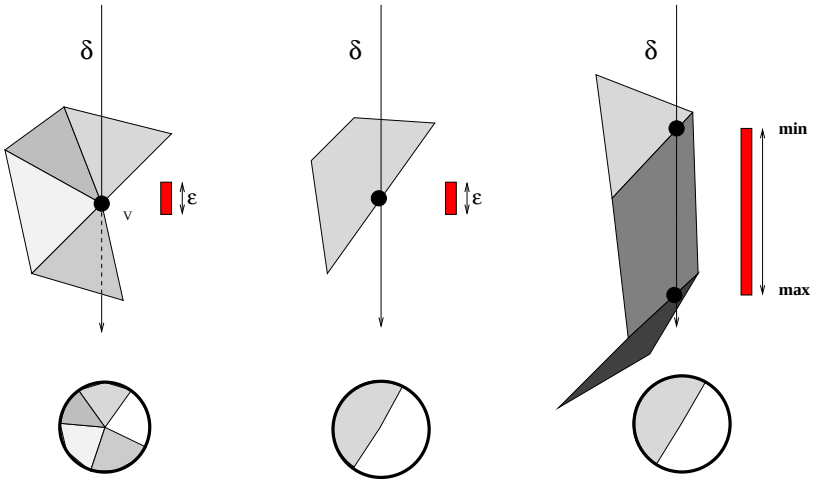


FIG. 29 – Les tranches épaisses des générateurs

Les tranches épaisses occupent des intervalles de position sur la droite (paramétrisation naturelle). Les tranches ne sont fusionnées que si les intervalles occupés entrent en intersection. Comme illustré figure 30, on partitionne la droite en segments concernés par les mêmes tranches épaisses. Dans chaque segment, on teste si les tranches forment une tranche complète, et si c'est le cas, on a alors un bloquage de la droite. Par exemple

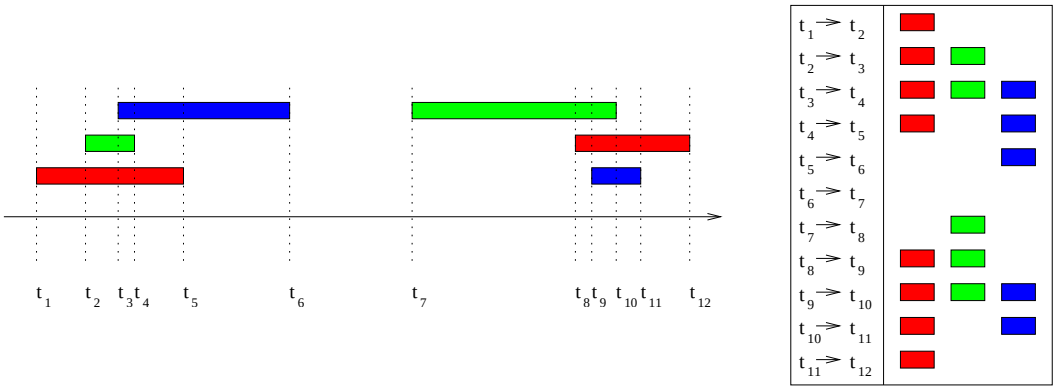


FIG. 30 – Partition de la droite selon les limites des tranches épaisses.

pour la figure 30, les tranches concernées sont les suivantes :

5.6 Intersections

5.6.1 Introduction

Lors de l'énumération des générateurs, nous avons cité le sommet (associé à une variété de dimension 2) et l'arête (associée à une variété de dimension 3). Cependant, certains éléments doivent être considérés comme générateurs.

Nous avons implicitement supposé que toutes les discontinuités des surfaces représentées par un ensemble de polygones étaient localisées sur les arêtes et les sommets, nous avons supposé que le reste de la surface était affine par morceaux. Cependant, cette hypothèse n'est pas toujours vérifiée. En effet, dans les maillages généraux, les faces peuvent entrer en intersection ! Dans ce cas, les éléments de cette intersection sont des discontinuités de la surface. Nous devons donc prendre en compte ces éléments. Pour celà, nous ajoutons deux types de générateurs les i-arêtes (*i-edges*), et les i-sommets(*i-vertices*), voir figure 31. Les i-sommets de la scène sont définis comme les points d'intersection entre les arêtes et les faces ; et les i-arêtes par les segments entre deux i-sommets contenus dans les plans des deux faces ayant généré les i-sommets.

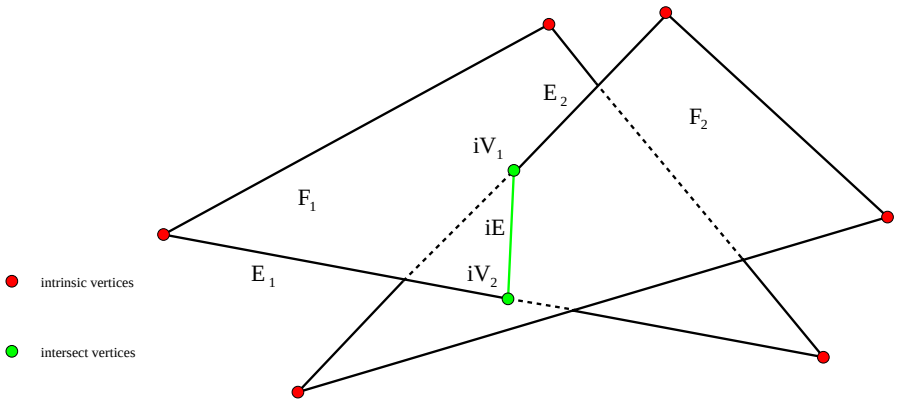


FIG. 31 – Définition des i-arêtes et i-sommets, un i-sommet est aussi appelé un sommet intersection (par opposition aux sommets intrinsèques et apparents).

5.6.2 Eléments d'intersection

Le calcul des i-sommets se fait de la façon suivante : pour chaque arête, calculer l'intersection du segment support de l'arête avec les faces de la scène. L'utilisation d'une grille et d'un lancé de rayons local (avec les faces) pour ce calcul permet de trouver efficacement les i-sommets. Nous conservons dans une structure représentant le i-sommet la face et l'arête ayant généré le i-sommet. Les i-sommets sont indexés par les faces sur lesquels ils s'appuient dans une table ainsi que par les arêtes qui les ont généré dans une autre table.

Le calcul des i-arêtes découle de l'utilisation de ces tables. Pour chaque face F, nous établissons la liste des i-sommets qu'elle a engendré $(iV_0, \ldots, iV_n)$. Pour chaque arête ayant engendré ces i-sommets $(E_0, \ldots, E_n)$, on établi la liste des faces qui y sont

attachées $(F_0, \ldots, F_m)$. Pour chaque face de cette liste, on crée une i-arête entre les i-sommets qu'elle partage avec la face F. Une i-arête est stockée à l'aide des deux sommets qui la compose.

5.6.3 Nouveaux générateurs

Au lieu de découper les arêtes en intersection avec des faces, et de subdiviser le maillage de sorte qu'il prenne en compte ces intersections, nous allons prendre ces dernières en compte, sans modifier le maillage d'entrée. Nous verrons dans les applications que toutes les intersections ne sont pas forcément utiles à calculer, ces calculs seront donc faits, lorsque celà est possible, à la volée.

Pour prendre en compte ces intersections et avoir une information de visibilité cohérente, nous allons définir deux nouveaux générateurs : le i-sommet et la i-arête. Ces deux générateurs s'apparentent au générateur sommet et arête sur plusieurs points, cependant, ils ont un comportement légèrement différent.

Le i-sommet engendre la même variété qu'un sommet placé à la même position. Cependant, ses arêtes connectées sont les i-arêtes connectées ainsi que l'arête ayant engendré le i-sommet. Ses faces connectées sont les faces connectées à l'arête l'ayant engendré ainsi que la face sur laquelle il se trouve. Un i-sommet est un générateur particulier puisqu'il est presque toujours bloquant : si la face **ou** l'arête sur laquelle il se trouve serait bloquante en l'absence d'un tel générateur, alors il est bloquant. Un nœud engendré par un i-sommet est aussi noté comme étant engendré par l'arête sur laquelle il s'appuie.

La i-arête engendre la même variété qu'une arête de même segment support. Ses faces connectées sont les faces qui l'ont engendrée. La i-arête est toujours bloquante !

5.6.4 Nœuds et arcs générés par des i-éléments

Les nœuds connectés à des i-sommets ont une phase de validation un peu particulière, en effet, on connaît presque toujours un bloqueur (début ou fin selon la position des autres générateurs). Ils sont notés comme étant aussi générés par les arêtes afin d'avoir une phase d'énumération des arcs cohérente avec les intersections.

Il en est de même pour la phase de validation pour un arc engendré par une i-arête.

Le i-sommet est considéré comme un sommet pour les phases d'énumération, ainsi que la i-arête comme l'arête.

6 Applications

6.1 Introduction

Nous allons présenter dans cette partie les résultats obtenus à l'aide de la technique présentée. Les applications sont les suivantes :

- Calcul des ombres projettées par nue source directionnelle
- Calcul des ombres projettées par une source ponctuelle
- Tracé de sommets
- Suppression d'objets cachés
- Maillage de discontinuité

Chacune des applications utilise une partie du squelette de visibilité. Nous allons détailler laquelle, puis présenter quelques optimisations pour ces techniques. Nous n'avons pas implanté toutes ces applications, en réalité, **seule la première a été faite**.

6.2 Calcul des ombres projettés par une source directionnelle

Le calcul des ombres projetées par une source directionnelle a été un problème largement abordé en graphique. Les techniques le plus souvent employées donnent des résultats certes rapides mais inexacts. Nous citerons le *shadow map* introduit par Williams [Wil78] dont on peut paramétrer la technique de projection, ce qui permet de modéliser plusieurs types de sources. Cette technique est intégrée dans les *infinite reality* de SGI et sur les cartes à base de processeurs *nVidia GeForce 3*, ce qui permet d'obtenir en temps réel (une passe OpenGL) des ombres sur les objets. Cependant, ces méthodes ont deux désavantages. Le premier est que la qualité de l'ombre dépend du point de vue, de la position de la source, et de la résolution du shadow buffer. Le second est que l'ombre est calculée lors de l'affichage, et que l'on ne peut pas utiliser cette information pour effectuer des requêtes de visibilité, elle ne donne pas de structure analytique avec informations de visibilité. La localisation des ombres nécessite d'autres calculs qui ne sont pas sans poser des problèmes de robustesse.

Nous allons donc chercher à calculer l'ombre exacte (à un ϵ près) projetée par une source directionnelle sur un objet, ou encore calculer la limite d'ombre sur les objets. Une fois cette information calculée, nous pourrons effectuer un rendu, mais aussi avoir une information fiable sur l'énergie reçue par une face.

Nous allons présenter ici une utilisation du squelette de visibilité permettant de calculer cette information de façon robuste et exacte (à un ϵ près).

6.2.1 Adaptation du squelette

Le squelette tel qu'il a été décrit précédemment ne permet pas de calculer ces ombres car pour celà il faut pouvoir restreindre l'espace des droites aux droites colinéaires à la direction de la source, cependant une petite modification peut le permettre : il faut ajouter un nouveau type de générateur, et proposer une méthode d'énumération.

Un nouveau générateur Afin de prendre en compte le caractère directionnel de la source, nous créons un nouveau type de générateur : ce générateur est similaire au sommet, mais au lieu de fixer un point de la droite, on fixe la direction de la droite. Ce générateur est aussi de codimension 2, c'est à dire que l'ensemble des droites engendrées par ce dernier est de dimension $2 = 4 - 2$, avec 4 la dimension de l'ensemble des droites et 2 la codimension du générateur. Nous allons l'utiliser de la même manière que le sommet. De plus, nous n'allons nous intéresser qu'à la partie du squelette attachée à ce générateur, donc les nœuds et arcs contenant ce générateur.

Calcul des nœuds Les nœuds sont calculés par énumération des sommets du modèle visibles (lancé de rayons avec objets grossis d'un ϵ, comme décrit dans la phase de validation), car la donnée d'une position (en plus de la direction donnée par la source) définit la droite ; et des sommets apparents, déterminés par un couple d'arêtes. Cette énumération est linéaire pour les sommets intrinsèques, et quadratique pour les sommets apparents. Le lancé de rayon en artihmétique d'intervalles permettra de prendre en

compte les dégénérescences, présentes ici aussi.

Calcul des arcs Les arcs engendrés par des arêtes non silhouette ne projettent pas de limites ombre sur la scène, il n'est donc pas intéressant de les considérer. Nous allons donc boucler sur les arêtes silhouette afin de calculer les arcs. En effet, la donnée d'une arête suffit à définir un arc car le générateur source directionnelle adjoint à une arête correspond à une variété de dimension 1 (sauf dans le cas où l'arête est dans la direction de la source auquel cas, on a un nœud dégénéré).

Une fois l'ensemble de générateurs suffisant (critère de suffisance abordé en 5.3.5) déterminé, on trie les nœuds touchant l'arête et on calcule le bloqueur de fin (voir figure 32). Si le bloqueur se trouve avant l'arête sur la droite médiane, alors il n'y a pas d'arc.

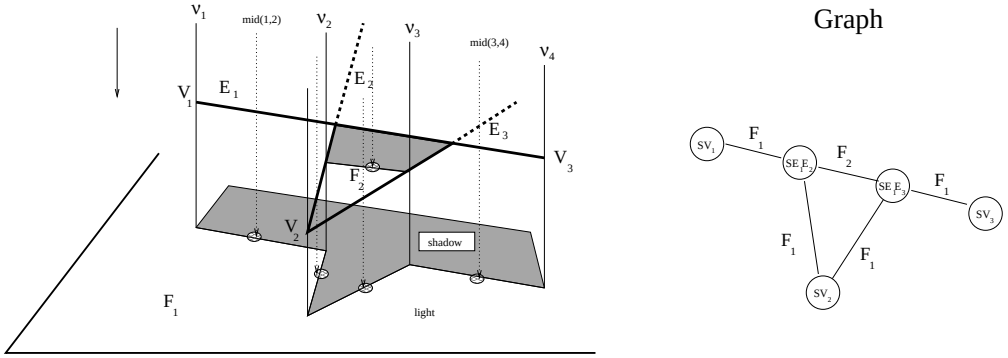


FIG. 32 – Sur la gauche, une partie de la scène ; sur la droite, le graphe construit : les nœuds sont les cercles, et les arcs les droites ; on note sur l'arc son bloqueur bas.

Optimisations La complexité naïve d'énumération des nœuds est la suivante : $O(n_f * n_e^2)$ avec n_f le nombre de faces et n_e le nombre d'arêtes. La grille tridimensionnelle sur les faces permet de passer de n_f à $r(n_f)$, avec $r(n_f)$ dépendant de distribution des faces dans la scène.

Tous les sommets et les couples d'arêtes ne sont pas parcourrus pour le calcul des nœuds candidats. En effet, comme seuls les calculs sur les arêtes silhouette sont pertinents, il n'est pas nécessaire d'énumérer tous les sommets et tous les couples d'arêtes. Pour ce faire, nous calculons l'ensemble des arêtes silhouette et énumérons uniquement les sommets de ces arêtes, ainsi que les couples d'arêtes contenant une arête silhouette au moins. La complexité de la recherche des couples d'arêtes en apparente intersection passe donc de $O(n_e^2)$ à $O(s(n_e)n_e)$ avec $s(n_e)$ le nombre d'arêtes silhouette de la scène. Pour une soupe de polygones (aucune connectivité au niveau des arêtes), la fonction est la fonction identité, pour une sphère uniformément subdivisée, la fonction est racine carrée.

La deuxième optimisation consiste à ne considérer que les couples d'arêtes qui sont en intersection apparente. Pour celà, nous utilisons une grille à deux dimensions, dans le plan orthogonal à la direction de la source, et dans chacune des cellules, nous établissons

la liste des arêtes dont la projection orthogonale est en contact avec cette cellule. Puis, dans l'énumération des couples d'arêtes, nous considérons comme couples d'arêtes possibles les arêtes silhouettes et toutes les arêtes référencées dans les cellules en contact avec l'arête silhouette. Cette optimisation permet, selon la résolution de la grille et l'allure de la scène, d'énumérer beaucoup moins de couples d'arêtes. Le nombre moyen d'intersections dépend de la forme de la scène, mais si l'on réplique des éléments dans la scène sans qu'ils interagissent, cette valeur est constante, alors que la taille de la scène augmente. On note $i(n_e)$ cette valeur.

Finalement, la complexité de l'algorithme est :

$$s(n_e) \cdot i(n_e) \cdot r(n_f)$$

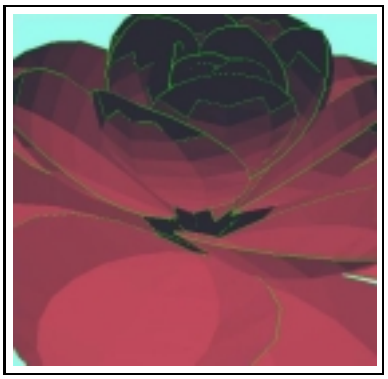
avec

- n_e le nombre d'arêtes
- n_f le nombre de faces
- $r(n)$ la fonction donnant le nombre moyen de faces testées par un rayon dans la scène
- $i(n)$ le nombre moyen d'intersections apparentes entre deux arêtes de la scène
- $s(n_e)$ le nombre d'arêtes silhouette

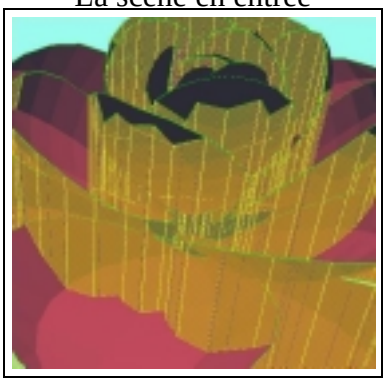
Dans la pratique, $r(n) << n$ (voir les résultats). Nous avons implanté une grille 3D récursive pour le lancé de rayons, et une grille 2D régulière pour la détection des intersections entre arêtes. Il serait sans doute intéressant d'utiliser une structure plus évoluée pour la grille 2D pour par exemple rester efficace lors de changements d'échelle, c'est à dire pour de petits objets avec beaucoup de facettes au milieu de grands objets.

Etapes de l'agorithme L'algorithme se décompose en plusieurs phases illustrées figure 33. La première consiste à calculer les arêtes silhouette de la scène (par rapport à la source). La seconde consiste à calculer les nœuds, puis les arcs du squelette. Ensuite, on calcule les intersections des arcs avec leurs bloqueurs finaux respectifs ce qui nous donne les limites d'ombres. Ensuite, on calcule une triangulation des faces de la scène, contrainte par les limites d'ombres. Enfin, pour chaque face obtenue, on teste si elle est visible ou non. Ce qui nous donne le reśultat. En illustration est aussi donnée la grille 2D utilisée pour optimiser la recherche des intersections apparentes d'arêtes.

La scène en entrée

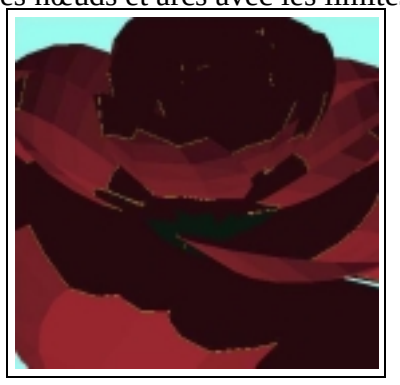


Les arêtes silhouette

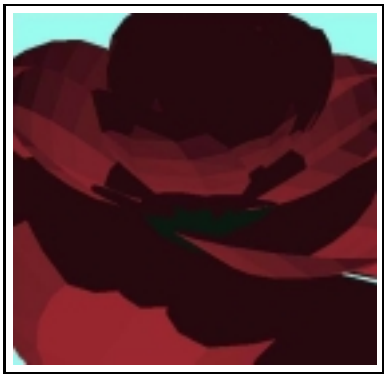


Les nœuds et arcs avec les limites

Les limites d'ombre



Le nouveau maillage



Le résultat final

FIG. 33 – Phases de calcul des ombres projettés par le soleil

Bilan Nous obtenons une partie du squelette de visibilité attachée au nouveau générateur source directionnelle qui va nous permettre par la suite d'extraire l'information dont nous avons besoin, par exemple pour calculer les limites d'ombre de la scène.

6.2.2 Intersection

Il n'est pas nécessaire de prendre en compte toutes les intersections. Si une intersection se produit dans une zone d'ombre, alors, il n'est pas nécessaire de la calculer. Pour ce faire, nous calculons l'ensemble des i-sommets. Si ils sont visibles et bloquants, alors on crée un nœud ; sinon non. Ensuite, on retrouve les i-arêtes par la méthode décrite dans la partie intersections. Les i-arêtes ainsi construites donnent nécessairement lieu à des arcs pour le squelette, c'est à dire à des limites d'ombres.

6.2.3 Calcul des limites d'ombre

Nous disposons de l'information suivante : la liste des arcs du squelette ayant pour générateur la source directionnelle, et les nœuds qui réalisent leur connexion. Dans les arcs, nous connaissons les bloqueurs, donc la dernière partie de scène visible le long d'une discontinuité de la visibilité. Cette limite correspond en fait à la limite entre les rayons issus de la source arrivant sur une face, et ceux n'y arrivant pas. Cette limite est la limite d'ombre que nous cherchons.

Le calcul des limites d'ombre dessinées sur les faces se fait alors de la manière suivante : pour chaque face, on cherche l'ensemble des arcs ayant pour bloqueur la face, et on dessine une arête dessus. On peut connaître les sommets de ces arêtes à l'aide des noeuds du graphe, plus exactement de l'intersection de ces nœuds avec ces faces. On en déduit ainsi la ligne polygonale de discontinuité d'ombre.

On peut donc sans calculs de visibilité supplémentaires, déterminer la limite d'ombre à partir de cette partie du squelette.

6.2.4 Résultats

Scènes de test Les scènes de test utilisées ont été générées par répétition aléatoire d'un objet. Trois objets ont été utilisés : un arbre, une rose, et le bunny. Chaque objet a des caractéristiques particulières :

- l'arbre est constitué de fines branches, beaucoup d'arêtes silhouette (ratio nombre d'arêtes silhouettes sur nombre de polygones = 0.73) et lors des répétitions, les arbres ont interagit, ses polygones sont allongés, et les polygones de 'arbre ont des intersections avec d'autres polygones du même arbre.
- la rose est intermédiaire, elle a moins d'arêtes silhouette (ratio = 0.39), des polygones plus réguliers, Des intersections ont été provoquées lors de la répétition de l'objet.
- le bunny a de nombreux petits polygones, mais peu d'arêtes silhouette (ratio = 0.059). Les polygones sont réguliers.

Nous allons étudier l'efficacité de notre algorithme, ainsi que les performances de nos structures d'accélération.

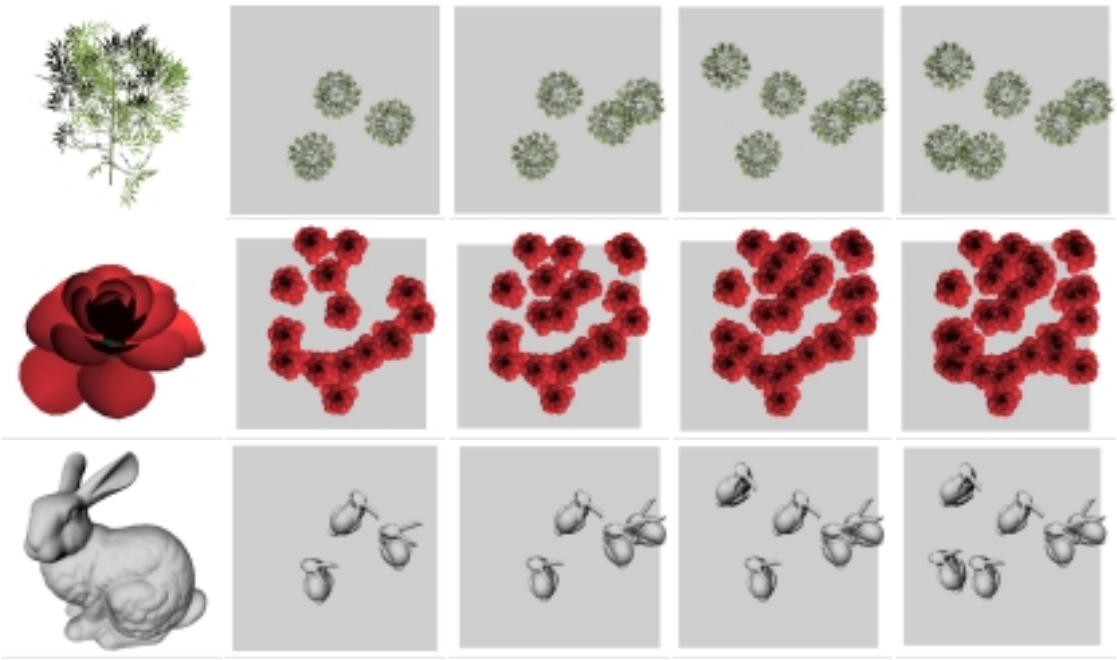


FIG. 34 – les scènes de test : en haut les arbres, au milieu les roses, en bas les bunnys

Premiers résultats On remarque que la plupart du temps est passé dans le calcul des nœuds. Nous allons donc étudier plus particulièrement les performances de cette phase.

Etude des fonctions r, s et i La fonction $r(n)$ est donnée par le nombre moyen d'interactions avec des faces calculées par rayon envoyé dans la scène. Cette fonction est très fortement liée au temps de validation puisque le temps moyen de validation est environ le temps de calcul d'une interaction multiplié par le nombre moyen d'interactions.

modèle	ratio	poly.	gr_3D	gr_2D	silh	nœuds	i-nœuds	arcs	limite
rose	0.39	17721	0.65	1.32	0.28	56.52	11.13	0.67	0.05
rose	0.39	26581	1.02	1.54	0.45	82.20	20.39	0.94	0.08
rose	0.39	35441	1.47	1.91	0.58	108.0	33.09	1.19	0.08
rose	0.39	44301	1.99	2.12	0.71	147.3	49.15	1.43	0.11
rose	0.39	53161	2.59	2.32	0.86	202.9	73.23	1.72	0.14
bunny	0.059	138947	1.52	2.44	1.89	87.57	30.86	0.72	0.05
bunny	0.059	208420	2.23	3.27	2.78	134.9	46.98	1.06	0.08
bunny	0.059	277893	3.04	4.21	3.76	180.9	62.67	1.37	0.10
bunny	0.059	347366	3.88	4.85	4.70	228.8	78.14	1.73	0.13
bunny	0.059	416839	4.56	5.75	5.57	277.3	94.22	2.60	0.16
arbre	0.73	65949	0.73	1.70	1.42	713.8	63.59	9.10	0.8
arbre	0.73	98923	1.11	2.23	2.50	1067.8	92.12	13.67	1.18
arbre	0.73	131897	1.50	2.70	2.85	1327.2	128.55	16.49	1.46
arbre	0.73	164871	1.85	3.19	3.50	1697.4	159.93	21.18	1.89
arbre	0.73	197845	2.23	3.60	4.21	2172.6	202.62	26.81	2.33

TAB. 1 – Performances (en secondes) - Sur un Processeur Pentium III cadencé à 1.0 GHz - 512 Mo de RAM.

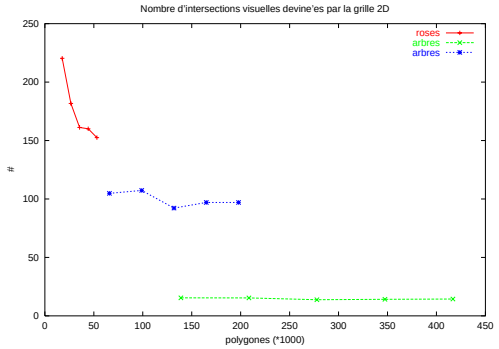
La fonction $s(n)$ dépend du modèle choisi et est constante lorsque l'on duplique le modèle. Elle est donnée pour chacun des modèles.

La fonction $i(n)$ donne le nombre moyen d'interactions possibles pour une arête. Cette fonction dépend fortement de la grille 2D utilisée. Nous ne changeons pas la résolution de la grille lors des tests et de la duplication des modèles, cette fonction est donc constante ou croissante lorsque l'on ajoute des interactions (inter-objets). Ce nombre est donné dans le tableau récapitulatif.

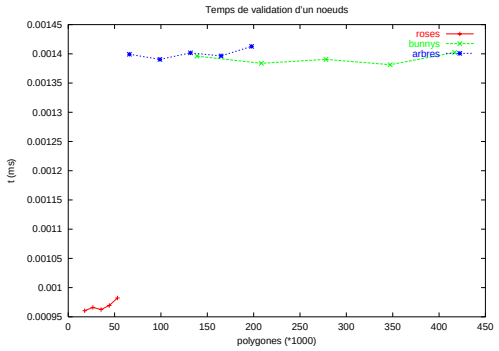
Efficacité des structures d'accélération En premier lieu, la grille à deux dimensions : la courbe donnée figure 35 en haut montre l'efficacité de la grille 2D. Elle donne le nombre effectif de sommets apparents (c'est à dire d'intersections d'arêtes vues depuis la source), en fonction du nombre d'arêtes testées à l'aide de la grille 2D.

On voit sur les courbes que la grille 2D est beaucoup plus efficace dans le cas des roses que dans le cas des arbres et qu'elle l'est de moins en moins pour les bunnys. Ceci s'interprête en observant que les arbres ont des feuilles à petits polygones qui représentent 90% de la scène, et que les polygones du bunny sont très petits. Dans ce cas, la taille d'un polygone sur la grille est inférieure à une cellule, et on retrouve beaucoup d'arêtes dans chaque cellule. Les performances peuvent être accrues en augmentant la résolution de la grille.

Notons que pour un modèle donné, la grille a une efficacité quasi constante en fonction du nombre d'éléments. Son efficacité décroît dès lors que les objets sont en interaction, voire en intersection, ce qui est le cas pour la rose et l'arbre.



nombre effectif de sommets apparents / nombre testés.



temps moyen de validation.

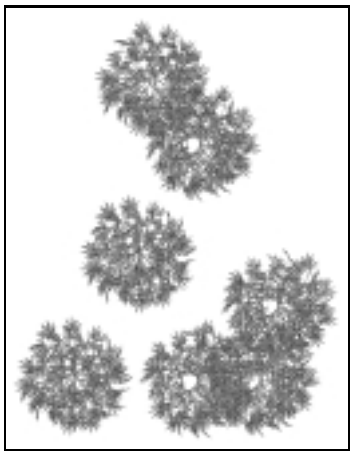


Illustration de la grille 2D : en dégradés de gris, les cellules de la grille qui contiennent les références sur les arêtes.

FIG. 35 – Efficacité des grilles

Ensuite, nous étudions l'efficacité de la grille 3D. Nous avons utilisé une grille récursive pour accélérer le lancé de rayons. Nous calculons le temps de validation d'un nœud (voir figure 35 en bas). Ce temps est directement fonction du nombre d'interactions trouvées sur un nœud candidat. Il reflète directement les performances de la structure d'accélération.

On peut voir que les courbes sont à peu près des constantes. Le temps est expérimentalement constant pour la répétition d'un motif donné.

Complexité expérimentale Etudions le temps de calcul des nœuds pour ces scènes. Ce temps est donné figure 36. Sur la gauche, le temps, sur la droite, le temps divisé par le ratio de silhouette. En effet, l'une des optimisations consiste à n'effectuer les énumérations que sur les éléments silhouette. Nous définissons le ratio silhouette comme étant le rapport entre le nombre d'arêtes silhouette et le nombre de faces.

On peut voir que la complexité est presque linéaire !

Nous venons de montrer expérimentalement que le temps de calcul des nœuds est :

$$t = i \cdot r \cdot s \cdot n$$

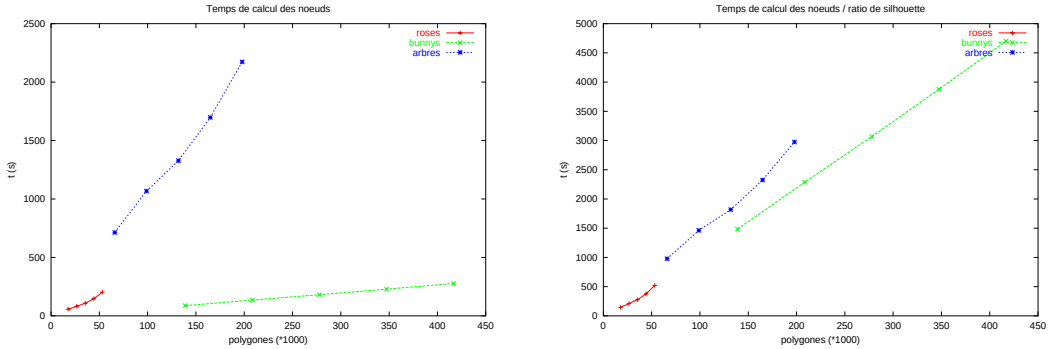


FIG. 36 – Effet du ratio de silhuoette sur le temps de calcul. Complexité linéaire.

avec i le nombre moyen d'intersections apparentes, n le nombre de polygones, s le ratio silhouette, et r le temps moyen de validation (constant par rapport à n).

Calcul des arcs Le temps de validation des arcs est donné figure 37 à gauche, Le temps moyen de validation d'un arc est donné à droite. On voit que ce temps est constant. On effectue bien la validation des arcs en temps constant !

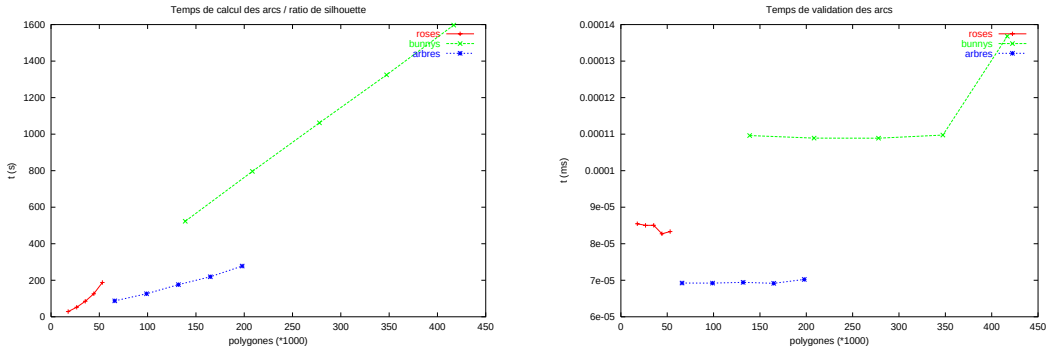
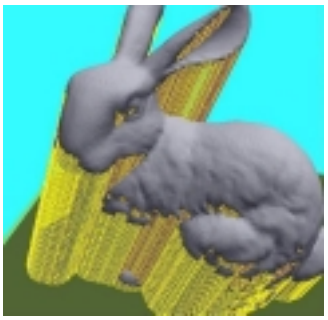


FIG. 37 – Calcul des arcs

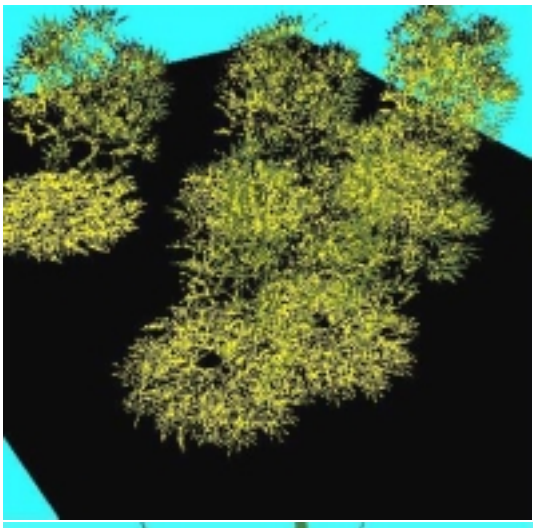
FIG. 38 – Comparaison avec le *shadow - map* : en haut, la scène en utilisant le *shadow map* (image obtenue avec l'aide de Marc Stamminger) ; en bas, notre méthode.

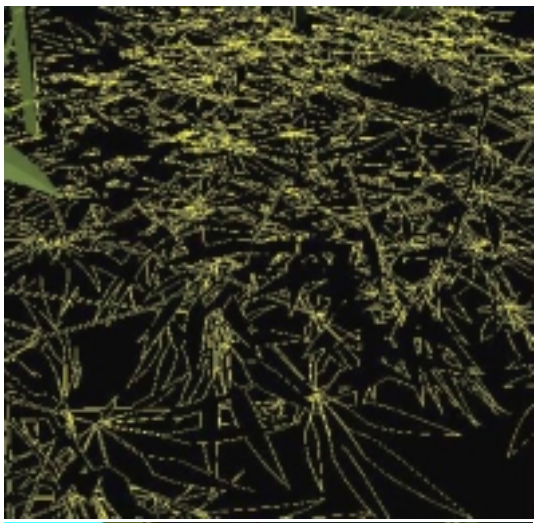


Squelette

Limites d’ombre

Résultat





6.3 Calcul des ombres projettés par une source ponctuelle

Le calcul des ombres projetées par une souce ponctuelle est similaire à celui des ombres projetées par une source directionnelle. La source ponctuelle est considérée comme un générateur se comportant comme un sommet. En revanche, le générateur n'est attaché à aucune face ni arête et ne peut bloquer une droite.

Nous allons calculer les nœuds et arcs du graphe ayant la source pour générateur. L'intersection des arcs et de leurs bloqueurs fin respectifs nous donnent les limites d'ombre.

Validation La phase de validation est similaire à celle utilisée pour la source directionnelle, c'est à dire que l'on ne considère qu'une direction pour la droite poignardante extrême, et que le point de départ est la source ponctuelle.

Optimisation Pour la source directionnelle, nous avons pu utiliser une grille à deux dimensions dans la direction orthogonale à la source, cependant, pour la source ponctuelle, nous ne pouvons pas utiliser la même technique directement. Plusieurs techniques sont envisageables : on peut utiliser plusieurs grilles 2D, une sur chaque coté d'un cube autour de la source. Mais cette structure est dépendante de la position, et si nous souhaitons intégrer plusieurs sources, ce précalcul sera à refaire. L'alternative que nous choisissons est d'utiliser un arbre octaire sur les arêtes.

Pour calculer les intersections apparentes possibles entre arêtes, on calcule le pinceau de droites porté par une arête E et le sommet de la source (voir figure 39). Alors, on cherche récursivement les cellules de l'arbre qui sont en intersection avec le pinceau. Pour déterminer si une cellule (boîte alignée avec les axes) est en intersection avec le pinceau, nous calculons trois plans, puis nous testons la position relative d'une cellule par rapport aux plans (voir figure 39) : soit au dessus, soit au dessous, soit en intersection.

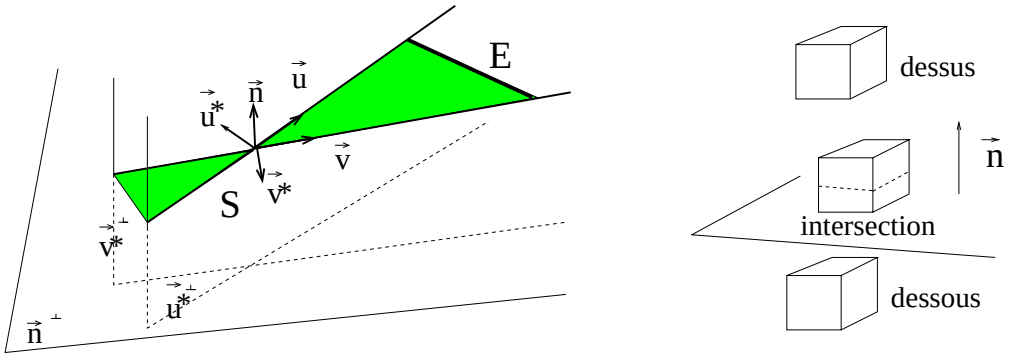


FIG. 39 – Test d'intersection entre un pinceau de droites et une cellule

Les vecteurs $\vec{n}$, $\vec{u^*}$ et $\vec{v^*}$ sont les vecteurs ainsi obtenus :

$$\vec{n} = \vec{v} \times \vec{u} \quad (16)$$

$$\vec{u^*} = \vec{n} \times \vec{u} \quad (17)$$

$$\vec{v^*} = \vec{v} \times \vec{n} \quad (18)$$

Si une cellule a des positions *(intersection, dessous, dessous)* pour les trois plans $((S, n), (S, u^*), (S, v^*))$ alors, elle est en intersection avec le pinceau de droites.

Bilan L'exploitation des résultats est similaire à celle des ombres pour une source directionnelle, en effet, le graphe contient ici la même information que dans le cas précédent.

6.4 Le tracé de sommets - *Vertex Tracing*

6.4.1 Introduction

Le tracé de sommets est une technique proposée par Stark et al. qui permet de calculer de façon exacte l'éclairage direct dû à des source étendues, ou encore à la reconstruction exacte de solutions de radiosité. Cette technique est issue de la reformulation de l'énergie lumineuse reçue en un point à partir d'informations sur les sommets visibles (depuis ce point) de la scène et des sommets apparents. Les détails sont décrits dans l'article [SR00].

L'information nécessaire au calcul de l'énergie lumineure reçue en un point, est l'ensemble des sommets intrinsèques et apparents, ainsi que la décomposition de leur voisinage en tranches (*spans*). La reconstruction finale des solutions de radiosité (ou l'éclairage direct), est alors ainsi faite : pour chaque pixel de l'image, lancer un rayon, puis calculer l'énergie lumineuse en ce point à partir du tracé de sommets. Suit le calcul de l'ensemble des sommets et de leurs voisinages, puis le calcul de la contribution à l'éclairage.

La partie critique de cet algorithme réside dans l'énumération des sommets intrinsèques et apparents visibles. Ils correspondent en fait à des nœuds du squelette de visibilité de la forme PV et PEE, ainsi que les nœuds dégénérés ; P étant le point duquel on calcule l'éclairage. Figure 40, l'illustration des trois cas rencontrés.

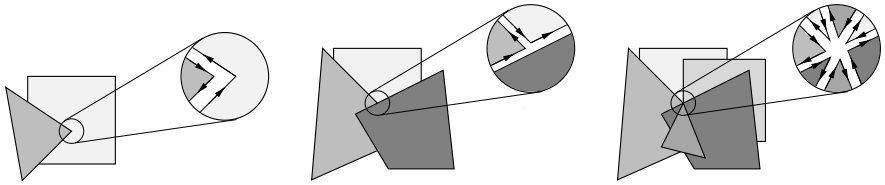


FIG. 40 – Les trois cas de sommets rencontrés : (a) : intrinsèque, (b) : apparent, (c) : dégénéré. (image tirée de l'article [SR00])

6.4.2 Utilisation du squelette

Nous allons utiliser le squelette pour énumérer les sommets intrinsèques, apparents et dégénérés de la scène vue depuis un point de vue. Ce calcul se fera par la construction de l'ensemble des nœuds attachés au générateur P. Sur chaque nœud, on connaîtra l'ensemble des générateurs, et en particulier, l'ensemble des faces qui y sont attachées. Nous utiliserons ensuite une technique voisine de l'éventail de bloqueurs pour déterminer les *spans* visibles depuis P. Finalement, ces *spans*, issus de faces, nous donneront la contribution de la face à l'eclairement du point.

La structure d'accélération utilisée pour la source point convient parfaitement ici. La liste des nœuds peut alors être obtenue comme précédemment. Ensuite pour chaque nœud, on énumère les faces qui y sont attachées et on construit des tranches pour chaque

face. Les tranches sont alors triées dans l'ordre croissant de distance au point (le long de la droite), et une projection de l'éventail est progressivement construite par intersection de tranches. Finalement, on peut intégrer la quantité d'énergie lumineuse reçue par le point à l'aide de ces tranches.

Cette approche nous donne une méthode robuste de calcul. Les dégénérescences sont mieux supportées, et le calcul peut être fait à un ϵ près. Toutes les techniques utilisées lors de la validation sont directement utilisées ici.

6.5 Suppresson d'objets cachés - *Occlusion culling*

La suppression d'objets cachés pour une région consiste à énumérer l'ensemble des objets visibles de la scène depuis au moins un point de vue de la région considérée. Pour celà, nous allons utiliser le squelette de visibilité : nous allons énumérer tous les nœuds dont la droite poignardante extrême est en intersection avec la région.

Pour chaque objet, on détermine s'il est toujours visible, potentiellement visible, ou invisible. Les objets invisibles étant supprimés de la liste d'objets à dessiner.

Objets toujours visibles S'il n'existe aucun objet entre l'objet considéré et la région (utilisation de *shaft*), alors, l'objet est visible.

Objets potentiellement visibles Si un objet est touché par un nœud (sur un bord (sommet ou arête), ou en son intérieur), alors, il est marqué comme potentiellement visible. Les nœuds sont générés à partir des objets qui touchent le *shaft* reliant l'objet considéré et la région. En effet, on vient d'exhiber une ligne de mire pour cet objet.

Objets invisibles - Idée de preuve Si un objet n'est touché par aucun nœud, alors il est invisible. En effet, si par l'absurde il avait une partie visible, alors, soit il serait entièrement visible, ce qui est facile à détecter, soit il serait partiellement visible. Soit un point r de la région depuis lequel on voit une partie de l'objet. Alors, depuis r, on voit l'objet en deux parties : la partie visible, et la partie invisible depuis r. Entre les deux existe une limite (similaire à un arc de type VE). Cette limite est une ligne polygonale dans le cas de scènes polygonales. On se place sur un sommet p de cette ligne polygonale, et on note les générateurs $(r, g_1, \ldots, g_n)$ ayant engendré cette droite poignardante extrême. On étudie ensuite l'ensemble des générateurs $(g_1, \ldots, g_n)$. Cet ensemble représente une variété de dimension au plus deux. Cette variété a des frontières de dimension 0 et 1 ; à chacune des frontières étant associé un autre générateur. On se place donc sur un point de la frontière qui correspond à une droite poignardante extrême. Alors soit cette droite poignarde l'objet et est générée par la région, ce qui le rend potentiellement visible, d'où la contradiction ; soit elle n'est pas générée par la région auquel cas elle poignarde la région, et est généré par un élément de l'objet (sommet ou arête) ; ce qui est encore une fois une contradiction.

Conclusion Afin de calculer les objets cachés pour une région, nous calculons l'ensemble des nœds dont les droites passent par la région, et nous marquons comme visibles, les éléments étant bloqueurs finaux pour une droite poignardante extrême. Ces calculs peuvent être faits localement entre la région et un polygone (ou un groupe de polygone) en utilisant le *shaft culling*.

6.6 Le maillage de discontinuité - *Discontinuity Meshing*

Calculer le maillage de discontinuité d'une scène consiste à subdiviser les éléments de la scène de façon à ce que sur chaque élément, l'aspect de la source ne change pas. Par exemple, le maillage de discontinuité pour une source directionnelle correspond à un maillage pour lequel chaque polygone est soit entièrement éclairé par la source, soit entièrement à l'ombre.

Le maillage de discontinuité peut être obtenu à partir du squelette de visibilité en calculant tous les nœuds et arcs entrant dans une des deux catégories suivantes :

– L'un des générateurs appartient à la source (sommet, arête).
– La source est un bloqueur pour le nœud ou l'arc.

Pour calculer les éléments de la première catégorie, on énumère les nœuds candidats selon les méthodes présentées dans la partie énumération (5.4), dont un des générateurs est un élément de la source. Tous les types de nœuds sont énumérés ici.

Pour ceux de la seconde catégorie, on énumère tous les noeuds candidats dont la droite passe par la source. Plusieurs techniques d'optimisation sont possibles afin de n'énumérer que les nœuds candidats dont les droites interagissent avec la source.

Les arcs sont ensuite déterminés à partir des nœuds précédemment calculés, et des arêtes de la scène.

Une fois tous les éléments du squelette concernant ce maillage calculés, on découpe les éléments de la scène selon les arcs du squelette. On obtient alors le maillage de discontinuité.

Optimisation - approche paresseuse Le calcul du maillage de discontinuité se prête naturellement à une approche paresseuse. En effet, on subdivise les éléments de la scène selon les arcs se trouvant entre la source et l'élément sur lequel on travaille. On va donc appliquer la technique suivante : pour chaque élément noté récepteur, déterminer l'ensemble des éléments de la scène entrant en interaction avec le faisceau de droites joignant la source et le récepteur. Pour ces éléments seulement, calculer les nœuds et arcs des deux catégories précédentes, et en déduire le maillage de discontinuité. L'espace mémoire utilisé par le squelette local peut alors être réutilisé pour la suite des calculs.

Conclusion

Nous avons proposé des techniques nouvelles de calcul d'information de visibilité. Elles ont été développés selon une démarche originale dans l'espoir qu'elles puissent s'appliquer au plus grand nombre de scènes. Les alogrithmes ont été pensés avec un souci constant de robustesse vis à vis de l'information en entrée.

Les optimisations et astuces développées dans les dernières parties ont rendu possible le traitement de scènes de plusieurs centaines de milliers de polygones, en un temps plus que raisonnable (de l'ordre de quelques minutes sur une station de travail). Malgré leur conception robuste, les algorithmes développés ont su rester performants.

L'idée principale de la démarche a été l'introduction et le traitement cohérent d'un ϵ, nous donnant une information visibilité *exacte à un ϵ près*. Peut-être cette notion d'ϵ-visibilité sera développée par la suite ?

Remerciements

Je souhaite tout d'abbord remercier George qui m'a encadré tout au long de ce stage, et avec qui nous avons pu développer les idées présentées dans ce mémoire. Je remercie Frédo qui m'a transmis ses connaissances du squelette et ses idées sur la robustesse. Je souhaite aussi remercier Marc pour son soutien, en particulier pour son *shadow - map* ; Xavier pour son éternelle disponibilité et Pierre pour sa touche d'humour desormais légendaire.

Je remercie aussi Mariette pour ses conseils dans l'utilisation de CGAL ; et Agnès notre assistante de REVES.

Ce stage a été financé par Visi3D : *Visibilité tridimensionnelle : théorie et applications*, Action de Recherche Coopérative de la Direction Scientifique de l'INRIA.

A L'espace des droites

Nous allons présenter ici les outils géométriques liés à la manipulation de droites dans l'espace monde (espace 3D usuel). Ces outils relèvent principalement de la géométrie euclidienne et de l'algèbre linéaire. Chaque point technique sera muni d'une illustration (exemple, figure, ...).

Dan la suite de cette partie, nous supposerons que par deux points (distincts) passe une et une seule droite (en 2D pour l'introduction et en 3D par la suite). Nous considérons aussi que les droites sont orientées.

Introduction

Un des grands problèmes soulevé par le calcul de visibilité réside dans la manipulation des droites. L'objet de cette partie est de présenter la paramétrisation des droites telles que nous l'utilisons dans notre application, et d'expliquer les raisons d'un tel choix.

Le discours adopté par la suite sera au début volontairement intuitif, et plus mathématique et rigoureux par la suite.

Dans le plan, on peut représenter une droite de plusieurs manières (cette liste n'est bien sûr pas exhaustive) :

- par son ordonnée à l'origne et un angle polaire : (y_0, θ)
- par une équation : $ax + by + c = 0$
- par un point de la droite, et son vecteur directeur : $M_0, \vec{u}$

Remarquons en premier lieu que ces paramétrisations ont un nombre de paramètres différent : deux pour la première, trois pour la seconde et quatre pour la dernière.

La première repésentation pose un problème de continuité : en effet, si la droite est parallèle à l'axe Oy, alors, l'ordonnée à l'origine n'est pas définie, il faudrait changer de paramétrisation pour ces droites spécifiques. Nous abandonnons donc ici cette paramétrisation.

Nous pouvons cependant remarquer certaines redondances dans les représentations suivantes. Pour la seconde paramétrisation, si l'on multiplie l'équation par n'importe quelle constante non nulle, l'équation obtenue représente la même droite ! Enfin, pour la dernière représentation, n'importe quel point M_0 sur la droite et n'importe quelle norme de vecteur directeur (sauf 0) peut être utilisée. Malheureusement, comme le montre la première paramétrisation, on ne peut pas se passer de cette redondance, elle est nécessaire.

La seconde et la troisième sont toutes deux aussi intéressantes dans le plan. En revanche, ce n'est pas le cas pour leur équivalent dans l'espace. En effet, la seconde paramétrisation dans le plan est intéressante car une hypersurface dans le plan est une droite, ce qui n'est pas le cas dans l'espace. Donc, il faudrait deux équations de la forme $ax + by + cz + d = 0$ pour représenter une droite dans l'espace.

La dernière représentation peut s'adapter ici : un point de la droite et un vecteur directeur. Cependant, si l'on stocke cette information pour représenter une droite, au

delà des redondances, les calculs seront allourdis : en effet, pour savoir si deux droites sont identiques, il faut comparer leur direction mais aussi savoir si un point de l'une (au moins) appartient à l'autre, ce qui impliquerait des calculs supplémentaires. De plus l'idée que la paramétrisation d'une droite soit dépendante d'un point arbitraire que l'on aurait choisi sur cette droite n'est pas satisfaisante.

Nous allons donc donner pour représenter une droite un couple de vecteurs : le vecteur directeur de la droite noté $\vec{u}$, ainsi qu'un autre vecteur indépendant du point sur la droite considéré : $\vec{v} = \vec{OM} \times \vec{u}$, avec O l'origine du repère et M un point de la droite[11].

Cette paramétrisation est en fait une paramétrisation de Plücker [Plü65], nous allons donc rappeler les principaux résultats établis.

Paramétrisation de Plücker

Paramétrisation

Considérons la droite Δ passant par les points P et Q et orientée dans le sens P vers Q. Ses deux vecteurs sont alors les suivants :

$$\vec{u} = Q - P = \vec{PQ}$$

$$\vec{v} = \vec{u} \times \vec{OP}$$

Cette paramétrisation n'est pas unique, en effet, n'importe quel couple de points de la droite représentera la droite (au sens près). Si l'on change les points de la droite, on observe que seule la norme de ces vecteurs change, elle est multipliée par un coefficient non nul (lié à la distance entre les deux points qui est en fait la norme du vecteur $\vec{u}$). Nous sommes donc en présence d'un espace projectif.

Forme bilinéaire de Plücker

Soient deux droites $\delta_1 = (\vec{u}_1, \vec{v}_1)$ et $\delta_2 = (\vec{u}_2, \vec{v}_2)$. La forme bilinéaire de Plücker notée $\odot$ par la suite est définie ainsi :

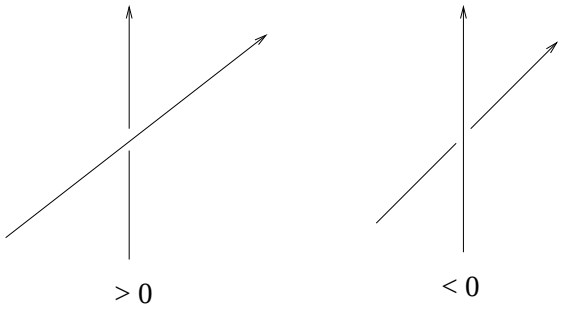
$$\delta_1 \odot \delta_2 = \vec{u}_1 \cdot \vec{v}_2 + \vec{u}_2 \cdot \vec{v}_1$$

Cette forme bilinéaire est nulle si et seulement si les deux droites sont coplanaires[12].

[11] Notons qu'une telle représentation d'une droite est en fait l'expression des éléments de réduction d'un glisseur de support la droite étudiée. Référence sur les gliseurs (cas particulier de torseurs) : [RDO79].

[12] Nous remarquerons que cette forme bilinéaire n'est rien d'autre que le comomment des deux glisseurs.

Interprétation géométrique



Si la forme bilinéaire de Plücker est positive, alors l'une des droites tourne positivement autour de l'autre, si elle est négative, la droite tourne négativement autour de l'autre.

Hypersurface de Plücker Toutes les paramétrisations ne correspondent pas à des droites réelles, en effet, par construction, nous avons pour toute droite δ, $\delta \odot \delta = 0$ (ou encore $\vec{u} \cdot \vec{v} = 0$). Cette équation définit une hypersurface dans un espace vectoriel de dimension 6. La forme bilinéaire exprimée matriciellement posséde deux valeurs propres triples : -1 et 1, nous vérifions donc que cette forme bilinéaire n'est pas un produit scalaire (non définie positive).

Nous nous intéressons donc au noyau de cette forme bilinéaire, qui représente l'ensemble des paramétrisations correspondant à des droites réelles [13].

Nous supposons par la suite que les droites que nous manipulons ont été construites à partir de points de l'espace et donc sont des éléments de cette hypersurface, le caractère réel des droites est à partir d'ici supposé acquis.

Opérations et Prédicats

Les opérations et prédicats sur les droites orientées sont les suivants :
- Droites parallèles
- Droites confondues
- Point sur la droite
- Normalisation
- Paramétrisation naturelle

Droites parallèles Deux droites sont parallèles si et seulement si leurs vecteurs directeurs le sont. C'est à dire, pour $\delta_1 = (\vec{u}_1, \vec{v}_1)$ et $\delta_2 = (\vec{u}_2, \vec{v}_2)$, δ_1 et δ_2 sont parallèles si et seulement si $\vec{u}_1 \times \vec{u}_2 = \vec{0}$.

[13] Cette hypersurface est l'ensemble des glisseurs dans l'espace des torseurs, c'est un cône.

Droites confondues L'espace des droites étant un espace projectif, deux droites sont confondues si leurs coordonnées sont égales à un coefficient multiplicatif près. Donc $\delta_1 = (\vec{u}_1, \vec{v}_1)$ et $\delta_2 = (\vec{u}_2, \vec{v}_2)$ sont confondues si et seulement si il existe λ tel que $\delta_2 = \lambda\delta_1$, ou encore la norme du vecteur $\delta_2 - \lambda\delta_1$ doit être nulle, en la calculant, on obtient un trinôme dont le discriminant réduit est $\Delta' = (\delta_1 \cdot \delta_2)^2 - (\delta_2^2 \delta_1^2)$. Ce discriminant ne peut être positif strictement, et les droites ne sont confondues que s'il est nul, d'où le prédicat.

Point sur la droite Un point M_0 : $\overrightarrow{OM_0} = (x_0, y_0, z_0)$ appartient à la droite $\delta = (\vec{u}, \vec{v})$, si on a : $\overrightarrow{OM_0} \times \vec{u} - \vec{v} = \vec{0}$. En effet, $\vec{v} = \overrightarrow{OP} \times \vec{u}$ avec P sur la droite, l'équation devient alors : $\overrightarrow{PM_0} \times \vec{u} = \vec{0}$. On en déduit le prédicat.

Normalisation L'espace des droites est un espace projectif. Comme nous l'avons vu, le premier vecteur de la représentation de la droite est le vecteur directeur de cette droite, il doit donc être non nul. Nous pouvons donc proposer une normalisation de la droite non pas pour les six coordonnées composant la droite, mais pour le vecteur directeur de la droite.

Cette normalisation nous permet d'introduire la paramétrisation naturelle de la droite.

Paramétrisation naturelle Une droite exprimée en coordonnées de Plücker possède une paramétrisation naturelle : l'ensemble des points de la droite est défini de manière unique par l'application $M : \boldsymbol{R} \to \boldsymbol{R}^3, \lambda \mapsto M(\lambda) = M(0) + \lambda \frac{\vec{u}}{|\vec{u}|}$, avec $\overrightarrow{OM(0)} = \frac{\vec{u} \times \vec{v}}{\vec{u} \cdot \vec{u}}$.

Pour une droite normalisée, la paramétrisation devient :

$$M(\lambda) = \vec{u} \times \vec{v} + \lambda \vec{u}$$

Projection orthogonale d'un point sur une droite On déduit de la parmétrisation de Plücker le résultat suivant : Soit V un point de l'espace, la paramétrisation de sa projection orthogonale sur la droite $\delta = (\vec{u}, \vec{v})$ est donnée par :

$$\lambda = \frac{\overrightarrow{OV} \cdot \vec{u}}{\vec{u} \cdot \vec{u}}$$

On en déduit le point à l'aide de la paramétrisation naturelle.

Remarque sur l'ensemble des droites Nous avons vu que seule une hypersurface (définie par un équation quadratique) correspond à des paramétrisations de droites réelles, de plus, nous pouvons réduire cet ensemble en ne considérant que les droites normalisées (ce qui correspond à une autre hypersurface définie par une équation quadratique : un cylindre), nous avons donc réduit notre ensemble de paramétrisations étudiée à l'intersection de ces deux hypersurfaces, qui est ici une variété de dimension 4.

Cette observation confirme le fait que localement (pour certaines droites), on puisse donner une paramétrisation en 4 paramètres : deux angles pour la direction, et les deux

coordonnées du point d'intersection de la droite avec le plan Oxy, cependant, la paramétrisation n'a pas les propriétés de continuité attendues sur l'ensemble de toutes les droites. D'où la paramétrisation en 6 paramètres.

B Ideaux et Variétés Algébriques

Nous allons présenter dans cette annexe quelques définitions et quelques résultats sur les idéaux de polynômes et les variétés algébriques. Nous n'allons évoquer ici que les notions nécessaires à l'établissement des résultat utilisés précédemment.

Cette annexe est très fortement inspirée du livre de David Cox, John Little et Donal O'Shea : *Ideals, Varieties, and Algorithms* [CLO98] ; spécialement de la première partie, qui introduit ces notions.

On peut supposer (mais ce n'est pas nécessaire) que le corps de base k est le corps des réels. On notera $x_1, ..., x_n$ les variables (dans notre cas, elles seront au plus au nombre de 6).

Définitions

Définition Un **monome** dans $x_1, \ldots, x_n$ est un produit de la forme

$$x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$$

où $\alpha_1, \ldots, \alpha_n$ sont des entiers non négatifs. Le **degré total** du monome est $\alpha_1 + ... + \alpha_n$.

Définition Un **polynôme** est une combinaison linéaire finie (à coefficients dans k) de monomes. On l'écrit sous la forme :

$$\sum_{\alpha} a_\alpha x^\alpha, a_\alpha \in k$$

où la somme est faite sur un ensemble fini de n-uplets $\alpha = (\alpha_1, \ldots, \alpha_n)$. L'ensemble des polynômes à coefficients dans k est noté $k[x_1, \ldots, x_n]$.

Définition Soit k un corps et $f_1, \ldots, f_s$ des polynômes sur $k[x_1, \ldots, x_n]$. Alors, soit

$$\boldsymbol{V}(f_1, \ldots, f_s) = \{(a_1, \ldots, a_n) \in k^n : f_i(a_1, \ldots, a_n)\} = 0, \quad pourtout \quad 1 \leqslant i \leqslant s\}$$

On appelle $\boldsymbol{V}(f_1, \ldots, f_s)$ une **variété affine**.

Exemple La variété affine $\boldsymbol{V}(x^2 + y^2 - 1)$ dans le plan, est le cercle de rayon 1 centré sur l'origine.

Lemme Soient V et W deux variétés affines, alors $V \cap W$ et $V \cup W$ sont aussi des variétés affines.

Définition Un sous-ensemble $I \subset k[x_1, \ldots, x_n]$ est un **idéal** s'il satisfait :
- $0 \in I$,
- Si f et g sont éléments de I alors $f + g$ est aussi élément de I,
- Si $f \in I$ et $h \in k[x_1, \ldots, x_n]$, alors $hf \in I$.

Définition - Notation Soient $f_1, \ldots, f_s$ des éléments de $k[x_1, \ldots, x_n]$. On note

$$< f_1, \ldots, f_s >= \left\{ \sum_{i=1}^{s} h_i f_i : h_1, \ldots, h_s \in k[x_1, \ldots, x_n] \right\}$$

Lemme Dans les notations précédentes, $< f_1, \ldots, f_s >$ est un idéal, on dit que c'est **l'idéal engendré par** $f_1, \ldots, f_s$.

Proposition Si $f_1, \ldots, f_s$ et $g_1, \ldots, g_t$ sont deux bases d'un même idéal sur $k[x_1, \ldots, x_n]$, c'est à dire telles que $< f_1, \ldots, f_s >=< g_1, \ldots, g_t >$, alors, $\boldsymbol{V}(f_1, \ldots, f_s) = \boldsymbol{V}(g_1, \ldots, g_t)$.

Définition Soit $V \subset k^n$ une variété affine. Alors on pose

$$\boldsymbol{I}(V) = \{f \in k[x_1, \ldots, x_n] : f(a_1, \ldots, a_n) = 0 \quad pour\ tout \quad (a_1, \ldots, a_n) \in V\}$$

Lemme Si $V \subset k^n$ est une variété affine, alors $\boldsymbol{I}(V)$ est un idéal. On l'appelle l'**idéal de** V.

Lemme Soit $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$, alors $< f_1, \ldots, f_s > \subset \boldsymbol{I}(\boldsymbol{V})$.

Utilisation des notions

Nous allons utiliser les idéaux algébriques et variétés affines de la façon suivante : nous allons chercher à caractériser par une équation le fait qu'une droite passe par un générateur, puis étudier les idéaux engendrés par de telles équations.

Variété des zéros La variété des zéros d'un idéal I est l'ensemble $V(I) \subset k^n$ vérifiant : pour tout $a = (a_1, \ldots, a_n) \in V(I)$, pour tout $f \in I$, on a $f(a) = 0$.

Propriété importante Une propriété importante des idéaux et variétés est la suivante : soient $f_1, \ldots, f_s$ et $g_1, \ldots, g_t$ deux bases, et $I_f =< f_1, \ldots, f_s >$ et $I_g =< g_1, \ldots, g_t >$ les idéaux engendrés par ces bases. Alors :

$$V(I_f + I_g) = V(I_f) \cap V(I_g)$$

Ce résultat intéressant nous amène à la conclusion suivante : chercher l'intersection de variétés revient à chercher la variété associée à la somme des idéaux des variétés.

Par exemple, si l'on modélise la réalisation d'un évenement ω par les zéros d'un polynôme (disons x réalise ω, si $f(x) = 0$), alors l'ensemble des réalisations de cet événement ($V(\omega)$) est la variété des zéros de l'idéal $I =< f >$ engendré par les polynômes caractérisant l'événement. Donc, l'ensemble des réalisations simultanées de plusieurs événements $\omega_1, \ldots, \omega_s$ modélisées par les zéros de polynômes (x réalise $\omega_1, \ldots, \omega_s$, si $f_1(x) = 0, \ldots, f_t(x) = 0$), est l'intersection des variétés des zéros de chaque idéal engendré par chaque équation, mais c'est aussi la variété des zéros de l'idéal $I =< f_1, \ldots, f_t >$ engendré par ces polynômes.

C Calcul de la dimension d'une variété de l'espace des droites engendrée par un ensemble fini de générateurs

Modélisation Rappelons brièvement le modèle : un générateur est un sommet ou une arête, il est associé à un idéal engendré par des polynômes de degré et de valuation 1. L'équation de réalité d'une droite $\delta = (\vec{u}, \vec{v}) = (x, y, z, u, v, w)$ est $\vec{u} \cdot \vec{v} = ux + vy + wz = 0$, avec l'idéal de réalité et la variété de réalité implicitement associés. Et finalement l'équation de normalisation est $x^2+y^2+z^2 = 1$, avec l'idéal de normalisation et la variété de normalisation implicitement associés.

Partie *linéaire* Afin de calculer la dimension de l'intersection de variétés associées à des générateurs, nous allons étudier la somme des idéaux engendrés par ces générateurs, puis étudier la variété de ses zéros.

A chaque générateur est associé un ensemble de polynômes de valuation et de degré 1 (applications linéaires). Puis la variété des zéros de la somme des idéaux engendrés par ces polynômes est ensuite intersectée avec la variété de normalisation (cylindre), et de réalité (cône). Nous allons calculer la dimension de l'intersection des variétés en calculant une base de Gröbner de l'idéal.

Dans une première partie, les polynômes engendrés étant des applications linéaires, une base de Gröbner de l'idéal peut être donnée par une base du noyau de l'application linéaire : soient $g_1, \ldots, g_n$, les générateurs. On note $l_1, \ldots, l_m$ les équations linéaires sans second membre qu'ils engendrent. Nous écrivons ces équations comme une application linéaire de l'espace des droites (isomorphe à R^6) dans R^m matriciellement :

$$\begin{matrix} l_{1_0} & l_{1_1} & l_{1_2} & l_{1_3} & l_{1_4} & l_{1_5} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{m_0} & l_{m_1} & l_{m_2} & l_{m_3} & l_{m_4} & l_{m_5} \end{matrix}$$

La variété que nous recherchons est en fait l'intersection du noyau de cette application linéaire (variété des zéros de cet idéal) avec les deux variétés de normalisation et de réalité. Une réduction de la matrice par pivot de Gauss nous permet de calculer la dimension du noyau de cette application linéaire. Cette variété des zéros est une intersection d'hyperplans.

Une fois la dimension de la partie *linéaire* calculée, nous devons calculer la dimension de l'intersection de la variété obtenue (intersection d'hyperplans) avec les deux variétés de normalisation et de réalité.

Intersection d'hyperplans avec la variété de réalité Dans certains cas, l'idéal de réalité est contenu dans l'idéal engendré par les applications linéaires. Par exemple, les polynômes (de degré et de valuation 1) engendrées par un générateur sommet sont de la

forme : (sommet de coordonnées (x_0, y_0, z_0), vecteur $\vec{u} = (x, y, z)$ et $\vec{v} = (u, v, w)$)

$$u + yz_0 - zy_0 \tag{19}$$
$$v + zx_0 - xz_0 \tag{20}$$
$$w + xy_0 - yx_0 \tag{21}$$

Si on effectue la combinaison linéaire (dans le module) suivante :

$$x(19) + y(20) + z(21)$$

On obtient le polynôme

$$ux + vy + wz$$

Ce polynôme engendre l'idéal de réalité. L'intersection de la variété de réalité avec l'intersection d'hyperplans est donc égale à l'intersection d'hyperplans.

Donc dans le cas d'un générateur sommet, l'idéal de réalité est contenu dans l'idéal associé à l'ensemble de générateurs. Il en est de même pour l'idéal associé à un ensemble d'arêtes concourrantes (et non toutes coplanaires), qui sont équivalentes à l'idéal associé à un sommet.

Afin de déterminer si l'idéal associé à un triplet d'arêtes contient l'idéal de réalité, nous effectuons le calcul suivant : on écrit les trois applications linéaires sous forme matricielle.

$$\begin{matrix} u_0 & v_0 & w_0 & x_0 & y_0 & z_0 \\ u_1 & v_1 & w_1 & x_1 & y_1 & z_1 \\ u_2 & v_2 & w_2 & x_2 & y_2 & z_2 \end{matrix}$$

On effectue ensuite une réduction de Gauss sur cette matrice pour la mettre sous la forme

$$\begin{matrix} 1 & 0 & 0 & m_{0,0} & m_{0,1} & m_{0,2} \\ 0 & 1 & 0 & m_{1,0} & m_{1,1} & m_{1,2} \\ 0 & 0 & 1 & m_{2,0} & m_{2,1} & m_{2,2} \end{matrix}$$

Si l'on ne peut mettre la matrice sous cette forme, alors, l'idéal de réalité n'est pas contenu dans l'idéal engendré par les trois arêtes. Sinon, on note M la matrice carrée de droite (c'est à dire $M = (m_{i,j})_{0 \leqslant i,j < 3}$). Alors l'idéal engendré par les trois arêtes contient l'idéal de réalité seulement si M est antisymétrique (c'est à dire peut correspondre à un produit vectoriel). On peut d'ailleurs lire les coordonnées du point d'intersection des trois arêtes à l'aide de la matrice M.

Si l'idéal de réalité n'est pas contenu dans celui associé aux générateurs, alors on décrémente la dimension, sinon, on ne la change pas.

Intersection d'hyperplans avec la variété de normalisation On note d la dimension de l'intersection d'hyperplans et de la variété de réalité. Si $d \geqslant 1$, alors l'intersection avec la variété de normalisation est une variété de dimension $d - 1$. En effet, l'idéal de polynômes engendré par l'intersection d'hyperplans est sans termes constants, et l'équation de normalisation ($u \cdot u = 1$, avec u vecteur directeur, en dimension 3) a un terme constant (valuation 0). L'équation de normalisation est donc ajoutée à la base de Gröbner, et la variété qui en résulte est donc de dimension $d - 1$.

Conclusion Afin de calculer la dimension de la variété associée à un ensemble de générateurs, on calcule la dimension d de la partie *linéaire* ; si elle est égale à zéro, il n'y a pas de droite passant par tous les générateurs. On teste ensuite les triplets d'équations linéaires pour savoir si l'idéal de réalité est inclus dans celui associé aux générateurs, si c'est le cas, la dimension est $d - 1$, sinon c'est $d - 2$.

Exemples Dans le cas du VE (V en dehors de la droite support de E), on a un sommet et une arête, donc quatre équations, d'où $d = 2$. Ayant un sommet, l'idéal de réalité est inclus, on a donc une dimension de 1 pour la variété, ce qui correspond bien à l'arc ! Dans le cas du EEE, on a $d = 3$ si les arêtes ne sont pas toutes dans un plan. Si l'idéal de réalité est inclus, c'est à dire que les droites sont concourrantes, on a une dimension de 2 pour la variété, sinon elle est égale à 1 (cas de l'arc EEE du catalogue). Cependant si deux arêtes sont coplanaires et non confondues, mais pas la troisième, alors, on obtient une dimension 1, mais avec un arc plan !

Bibliographie

Références

[Arv94] J. Arvo, *The irradiance Jacobian for partially occluded polyhedral sources*, ACM SIGGRAPH '94, 1994, pp. 343–350.

[AW87] John Amanatides and Andrew Woo, *A fast voxel traversal algorithm for ray tracing*, Eurographics '87, Elsevier Science Publishers, Amsterdam, North-Holland, 1987, pp. 3–10.

[Cat74] Edwin E. Catmull, *A subdivision algorithm for computer display of curved surfaces*, Ph.D. thesis, Dept. of CS, U. of Utah, December 1974.

[CDP95] Frédéric Cazals, George Drettakis, and Claude Puech, *Filtering, clustering and hierarchy construction : a new solution for ray tracing very complex environments*, Eurographics '95, 1995.

[CLO98] David A. Cox, John B. Little, and Donal O'Shea, *Ideals, varieties, and algorithms*, Springer, 1998.

[DDP96] Frédo Durand, George Drettakis, and Claude Puech, *The 3d visibility complex, a new approach to the problems of accurate visibility*, Proceedings of 7th Eurographics Workshop on Rendering in Porto, Portugal (Rendering Techniques '96) (Xavier Pueyo and Peter Schröder, eds.), Springer Verlag, June 1996, pp. 245–256.

[DDP97] _____, *The visibility skeleton : A powerful and multi-purpose global visibility tool*, ACM SIGGRAPH 97, August 1997, http ://w3imagis.imag.fr/Membres/Fredo.Durand/PUBLI/siggraph97/index.htm.

[DF94] George Drettakis and Eugene Fiume, *A fast shadow algorithm for area light sources using backprojection*, Proceedings of SIGGRAPH '94 (Andrew Glassner, ed.), Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, ACM Press, 1994, pp. 223–230.

[DHH01] Olivier Devillers and Olaf Hall-Holt, *Predicates and constructions for visibility problems*, manuscrit, 2001.

[Dur99] Frédo Durand, *3d visibility : analytical study and applications*, Ph.D. thesis, Université Joseph Fourier, Grenoble I, July 1999, http ://www-imagis.imag.fr.

[GKM93] Ned Greene, Michael Kass, and Gavin Miller, *Hierarchical z-buffer visibility*, ACM SIGGRAPH '93, 1993.

[Hec92] Paul S. Heckbert, *Discontinuity meshing for radiosity*, Eurographics Rendering Workshop 1992, Eurographics, May 1992, pp. 203–216.

[JW89] David Jevans and Brian Wyvill, *Adaptive voxel subdivision for ray tracing*, Proceedings Graphic's Interface '89, Canadian Information Processing Society, 1989, pp. 164–172.

[Kaj86] James T. Kajiya, *The rendering equation*, Computer Graphics (SIGGRAPH '86 Proceedings) (David C. Evans and Russell J. Athay, eds.), vol. 20, Computer Graphics Proceedings, Annual Conference Series, no. 4, ACM SIGGRAPH, ACM Press, Août 1986, pp. 143–150.

[KvD79] Jan J. Koenderink and Andrea J. van Doorn, *The internal representation of solid shape with respect to vision*, BioCyber **32** (1979), 211–216.

[Plü65] Plücker, *On a new geometry of space*, Phil. Trans. Royal Soc. London, 1865.

[PV96] Michel Pocchiola and Gert Vegter, *The visibility complex*, International Journal of Computational Geometry and Applications **6** (1996), no. 3, 279–308.

[RDO79] Ramis, Deschamps, and Odoux, *Cours de mathématiques spéciales*, vol. 2, Masson, 1979.

[SG94] A. James Stewart and Sherif Ghali, *Fast computation of shadow boundaries using spatial coherence and backprojections*, Proceedings of SIGGRAPH '94 (Andrew Glassner, ed.), Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, ACM Press, 1994, pp. 231–238.

[SK98] A. James Stewart and Tasso Karkanis, *Computing the approximative visibility map, with applications to form factor and discontinuity meshing*, Eurographics Rendering Workshop 1998, Eurographics, 1998.

[SR00] Michael M. Stark and Richard F. Riesenfeld, *Exact radiosity reconstruction and shadow computation using vertex tracing*, Proceedings of 11th Eurographics Workshop on Rendering, 2000.

[SR01] ———, *Reflected and transmitted irradiance from area sources using vertex tracing*, Proceedings of 12th Eurographics Workshop on Rendering, 2001.

[Tel92a] Seth J. Teller, *Computing the antipenumbra of an area light source*, Proceedings of SIGGRAPH '92, Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, ACM Press, 1992, pp. 139–148.

[Tel92b] ———, *Visibility computation in densely occluded polyhedral environments*, Ph.D. thesis, University of California, Berkeley, 1992.

[TH94] Seth Teller and Pat Hanrahan, *Global visibility for illumination computations*, Proceedings of SIGGRAPH '94, Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, ACM Press, 1994, pp. 443–450.

[TS91] Seth J. Teller and Carlo H. Séquin, *Visibility preprocessing for interactive walkthrough*, ACM SIGGRAPH '91, july 1991, pp. 61–69.

[Wil78] Lance Williams, *Casting curved shadows on curved surfaces*, Proceedings of SIGGRAPH '78, ACM SIGGRAPH, August 1978, pp. 270–274.

[ZMHI97] Hanson Zhang, Dinesh Manocha, Tom Hudson, and Kenneth E. Hoff III, *Visibility culling using hierarchical occlusion maps*, ACM SIGGRAPH '97, 1997.