UNIVERSITÉ DE MARNE-LA-VALLÉE
École doctorale "Information, Communication, Simulation, Modélisation"

THÈSE

Formation doctorale: Sciences de l'Information Géographique

présentée par
Fabien RAMOS

MODELISATION ET VALIDATION D'UN SYSTEME D'INFORMATION GEOGRAPHIQUE 3D OPERATIONNEL

Soutenue le 5 mai 2003 devant le jury composé de:

Jacqueline Argence,	Docteur en Sciences,	Examinatrice
Patrice Boursier,	Professeur à l'Université de La Rochelle,	Rapporteur
Bernard Cervelle,	Professeur à l'Université de Marne-la-Vallée,	Directeur de thèse
Jean-Paul Donnay,	Professeur à l'Université de Liège,	Rapporteur
Hervé Le Men,	Habilité à Diriger les Recherches,	Examinateur
Anne Ruas,	Docteur en Sciences,	Examinatrice
Daniel Siret,	Docteur en Sciences,	Examinateur

RESUME

De nombreux utilisateurs de données géographiques issus de domaines d'application très variés comme l'urbanisme, la sécurité civile, la défense, la téléphonie ou le transport urbain, sont demandeurs de systèmes d'information géographique (SIG) capables d'exploiter des données 3D. Malheureusement, les solutions commerciales disponibles sur le marché du SIG sont assez pauvres en matière de 3D. Dans cette thèse, nous avons modélisé puis implémenté un prototype de SIG 3D en souhaitant dépasser la simple extrusion de contours 2D et ainsi démontrer qu'il est aujourd'hui envisageable de manipuler des lots de données véritablement tridimensionnels au sein d'un système d'information géographique.
L'architecture de notre prototype s'articule autour d'un modèle de données géométrique de type « Boundary Représentation » (B-Rep), chaque objet 3D étant ainsi décrit par un ensemble de faces. Cette représentation par frontière possède l'avantage de pouvoir modéliser toute forme d'objet, aussi complexe soit-elle. Pour garantir une meilleure cohérence des données, nous associons une composante topologique « structurelle » à cette modélisation géométrique B-Rep. Une topologie de réseau optimise quant à elle une grande partie des requêtes appliquées aux réseaux. L'ensemble des données est stocké dans un système de gestion de base de données relationnel étendu, un index spatial de type R-Tree 3D ayant été mis en place au-dessus de cet SGBDRE afin de permettre un accès plus direct, et donc plus rapide, aux objets.
Un des apports essentiels de la 3D aux systèmes d'information géographique réside dans la nouvelle dimension qui est donnée aux requêtes de calcul d'intervisibilité et de recherche de trajectoire. Notre prototype offre ainsi la possibilité de constituer des cartes de visibilité tenant compte aussi bien du MNT, et donc du relief du terrain, que de la forme 3D des objets composant le sur-sol de la scène modélisée. Ce sont des contraintes qui ont été également retenues lors de l'implémentation de nos algorithmes de recherche de trajectoire optimale. D'autres éléments, tels que des critères de visibilité ou les caractéristiques de l'objet en mouvement, peuvent intervenir comme paramètres d'entrée des recherches de trajectoire.

ABSTRACT

A huge community of geographical and spatial data users like geologists, militaries, town planners or communication and utilities managers are interested in a Geographical Information System being able to handle the third dimension. Unfortunately, GIS commercial solutions are extremely weak about 3D concern. The purpose of this PhD thesis is to model and implement a 3D GIS prototype, which is able to surpass the simple 3D extrusion of 2D outlines. Our prototype demonstrates that today it is actually possible to deal with true 3D data in a Geographical Information System.
The architecture of our prototype is structured on a "Boundary Representation" (B-Rep) geometrical model. This model describes an object as a set of faces, which has the advantage of modeling any kind of forms, even very complex ones. In order to insure a better coherence of data, we have added a "structural" topological layer above the B-Rep geometrical description of the object. A "network" topology is implemented to optimize most of network's queries inside the application. Finally, the set of geographical data is stored in an extended relational database management system, and a 3D R-Tree spatial index allows a straighter and faster data access.
One of the most important contributions of 3D to the geographical information systems lies in the new standing given to visibility computing and to optimal path queries. Our prototype allows to draw visibility maps that take the 3D forms of the terrain and the 3D forms of the geographical objects into account. Those constraints are also considered by our optimal path algorithm, which can be parameterized with visibility maps or with the characteristics of the moving object.

AVANT PROPOS

Nous présentons dans ce mémoire trois années de travaux consacrées à la conception puis au développement d'un prototype de système d'information géographique tridimensionnel. Un programme d'étude amont en géographie numérique de la Délégation Générale pour l'Armement est à l'origine de ce prototype. La DGA souhaitait en effet réaliser une étude sur les apports potentiels d'un outil commun de SIG 3D pour des opérationnels militaires issus d'organismes diversifiés. Ce projet d'étude s'est rapidement transformé en appel d'offre auquel le département Geomatics de Matra Systèmes et Information (MS&I) et le laboratoire de recherche COGIT de l'Institut National Géographique (IGN) ont répondu conjointement. La complémentarité de ces deux organismes en matière de défense et modélisation 3D pour MSI (aujourd'hui intégré au sein du groupe EADS) et d'expérience en systèmes d'information géographique pour l'IGN firent que l'étude commanditée par la DGA leur fut attribuée. Ceci se concrétisa par la mise en place d'une convention CIFRE associant MS&I comme partenaire industriel et le COGIT comme laboratoire de recherche, le COGIT étant un des laboratoires d'accueil de l'école doctorale ICMS (Information, communication, modélisation, Simulation) de l'université de Marne-La-Vallée. Enfin, l'Institut Francilien des Géosciences et sa formation doctorale SIG (affiliée à l'école doctorale ICMS) ont également été associés à nos recherches.

Afin de satisfaire l'ensemble des organismes concernés par ce partenariat, nous avons dû tenir compte d'un certain nombre de contraintes, notamment industrielles, durant la conduite de notre thèse. Ainsi, le travail que nous présentons dans ce mémoire est-il, peut-être, davantage orienté vers des objectifs de développement qu'il ne l'aurait été dans un contexte plus académique. Nous ne nous sommes pas placés dans une perspective de pure recherche fondamentale mais bien dans une optique de recherche et développement.

SOMMAIRE

LISTE DES FIGURES

LISTE DES TABLEAUX

1 INTRODUCTION

Le nombre d'applications générant ou traitant de l'information géographique est depuis quelques années en constante augmentation. L'éventail des secteurs d'activité impliqués dans cet essor est à la fois vaste et diversifié : la défense, l'aménagement, le tourisme ou le jeu utilisent tous aujourd'hui des données géographiques 3D. Philippe Gerbe [GERBE 2002] recensa en 2002 quelques exemples d'utilisations de ces données en les regroupant par thème d'application dans un tableau offrant ainsi un assez bon aperçu de ce qu'il est possible de faire aujourd'hui avec de telles données (Tableau 1).

Thème	Exemple d'utilisation de données géographiques en 3D
Aménagement	· étude d'impact paysager (zone rurale ; milieu urbain ; volet paysager des permis de construire) · analyse pour l'insertion de projet dans un site (ouvrage d'art, bâtiment, station de ski …) · analyse pour des études d'urbanisme · visualisation de l'évolution de la végétation sur plusieurs années · parcours d'une future route et évaluation de son impact · aide à la cartographie des servitudes d'utilité publique (SUP) tels que les plans de servitude aéronautique
Risques et pollutions	· cartographie 3D du bruit et analyse de l'impact d'aménagements (écran antibruit …) · cartographie et étude de pollutions atmosphériques · simulation d'inondations, application de modèles d'écoulement des eaux (mise au point de plans de prévention des risques – PPR) · évaluation des populations en zone d'aléa : répartition de la population par volumétrie des bâtiments
Sous-sol ; sous-marin	· analyse de mesures d'exploration pétrolière · aide à la pose de pipe-lines sous-marins · cartographie et analyse géologique · spéléologie (représentation de réseaux souterrains)
Tourisme	· présentation de villes : villes virtuelles · présentation de régions touristiques, d'un département · reconstitution virtuelle d'un patrimoine historique (village, cathédrale, grotte …) · mise à disposition d'informations par Internet

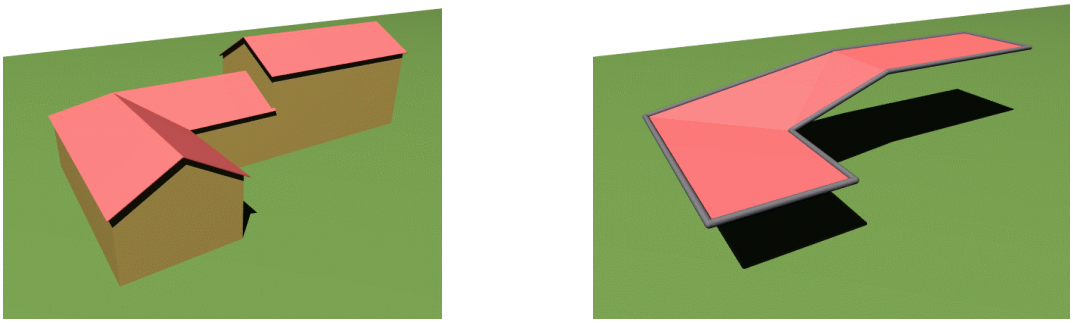
Télécoms	· aide à l'implantation d'antennes (téléphone, télévision) · calcul d'inter-visibilité entre émetteurs
Transports	· simulateurs de poste de conduite (avion, train, bateau) · simulation de trafic pour évaluer les capacités d'un nouvel aménagement routier · vision haute pour les pilotes d'avions
Jeux, films	· réalisation et visualisation de paysages virtuels
Géomarketing	· visualisation de zones d'influence ou de chalandise (dans ce cas, la 3D n'est pas indispensable, mais ajoute un " plus " dans la visualisation et aide à l'analyse des zones d'influence)
Défense	· entraînement, simulateurs de conduite (chars, avions, navires) · préparation de mission · vision haute pour les pilotes d'avions

Tableau 1 : Application 3D [GERBE 2002]

Assez paradoxalement, les outils pour manipuler cette information géographique 3D restent peu nombreux. Les principaux éditeurs de SIG proposent des solutions approchant la 3D de manière plus ou moins convaincante mais aucune ne peut être qualifiée de pleinement 3D.

Les premières approches vers la troisième dimension dans les bases de données géographiques consistèrent à ajouter un attribut altitude sur chacun des points des bases de données en deux dimensions. La principale limitation de ce type de démarche est de ne permettre qu'une seule altitude z en tout point de la scène géographique modélisée, c'est à dire pour tout couple (x,y) du plan. De ce fait, il devient impossible, contrairement à ce qui peut se faire en CAO par exemple, d'intégrer dans ces logiciels des objets géographiques aux formes concaves. Les systèmes d'information géographique se placent donc inconfortablement à mi-chemin entre la 2D et la 3D, ils sont souvent qualifiés d'applications 2,5D.

L'Institut Géographique National (IGN) constitue depuis plusieurs années une base de données en 2,5D : La Base de Donnée Topographique (BD Topo). Cette base regroupe différents thèmes (voiries de communication routière, hydrographie, bâtiments, végétations, limites administratives etc.) pour des échelles allant du 1:5 000 au 1:50 000. Les bases de données 2,5D présentent l'avantage de pouvoir être générées et gérées de manière relativement simple puisque le modèle de données 2D n'est pas remis en question. Dans le cas de la BD Topo, les bâtiments ont simplement été saisis au niveau des gouttières au lieu de l'être au niveau du sol. Malheureusement, la forme modélisée de l'objet reste parfois assez éloignée de la forme réelle de l'objet [DE LA LOSA 2000]. La Figure 1 illustre un exemple d'incohérence entre la forme d'un objet géographique et sa modélisation dans la BD Topo.



Monde réel Bâtiment dans la BD Topo

Figure 1 : Représentation des bâtiments dans la BD Topo [DE LA LOSA 2000]

Devant les insuffisances de la 2,5D, certaines équipes de développement ont alors pensé à des modèles permettant de stocker deux altitudes différentes pour un même couple (x,y) de coordonnées. Cette démarche s'apparente d'avantage à une évolution de la 2,5D plutôt qu'à une proposition de nouveau model 3D. La 2,75D permet en effet d'attribuer une certaine « épaisseur » pour de simples objets mais reste vite insuffisante dans le cas d'objets plus complexes ou décrits plus en détails.

Il peut être étonnant de constater que la communauté des géographes n'ait pas cherché à combler plus rapidement son retard en modélisation 3D. Après tout, s'il est possible de modéliser dans le détail un objet 3D en CAO, pourquoi ne serait-il pas possible de le faire au sein d'un SIG ?

Un élément de réponse réside sans doute dans la nature même des objets à modéliser. L'ingénieur industriel utilisant des logiciels de CAO travaille sur un nombre restreint d'objets qu'il clonera ensuite à l'infini. A l'opposé, un ingénieur géographe manipule de gigantesques bases de données remplies d'objets géographiques aux formes souvent différentes les unes des autres. Les contraintes d'acquisitions ne sont donc absolument pas les mêmes pour nos deux ingénieurs. Si ces contraintes sont si lourdes à gérer qu'il n'est pas envisageable pour un géographe de se constituer une base de données véritablement 3D, on comprend qu'il puisse se satisfaire de 2,5D pour manipuler ses objets géographiques.

Faut-il pour autant abandonner toute idée de solution SIG véritablement 3D ? Nous ne le pensons pas. Les progrès réalisés en matière d'acquisition de données sont tels qu'il est aujourd'hui réaliste d'envisager qu'à moyen terme les techniques seront suffisamment avancées pour automatiser les processus de saisie, reconstitution et modélisation de formes 3D. Il serait dommage que dans cinq, dix ou quinze ans, le jour où la télédétection nous aura donné les moyens de nous constituer rapidement et simplement une base de données géographique 3D finement détaillée, aucun SIG ne soit capable d'exploiter pleinement le potentiel de ces nouvelles données.

Aussi avons-nous volontairement placé notre travail de thèse sur les SIG 3D en aval des processus d'acquisition. Nos recherches ont été menées autour des trois axes majeurs que sont la modélisation géométrique et topologique, les calculs de visibilité et la recherche de trajectoire. L'objectif final de nos travaux reste de mettre ces recherches à profit afin de réaliser un prototype démontrant la faisabilité et le potentiel d'un système d'information géographique tridimensionnel.

Le premier chapitre de notre rapport est consacré aux modèles de données. Nous nous sommes inspirés des travaux antérieurs menés en sciences géographiques mais aussi en CAO pour proposer notre propre modèle. Celui-ci intègre à la fois les caractéristiques géométriques et les caractéristiques topologiques des objets géographiques. Nous avons essayé de trouver un compromis pour aboutir à une modélisation simple mais fonctionnelle.

Le second volet de ce document traite de l'étude théorique des fonctionnalités d'analyse indispensables à un SIG 3D. La consultation de données attributaires, les requêtes géométriques, l'utilisation d'index spatiaux font classiquement partie de ces fonctionnalités. Cependant, ce seront essentiellement les problèmes de visibilité et de recherche de trajectoire qui attireront notre attention.

Enfin, la dernière partie de ce mémoire reprendra les grandes lignes de l'implémentation de notre prototype baptisé TriGO (Tridimensionnal Geographical Objects). Ce prototype a été développé au département Géomatics de EADS S&DE. De conception objet, il s'appuie sur des composants externes de visualisation et de stockage développés notamment en interne par EADS. Bien qu'il fallut parfois s'adapter aux contraintes industrielles ainsi qu'au contexte de l'étude de la DGA, TriGO reprend pour l'essentiel les idées avancées dans les deux premiers chapitres du rapport. Expérimenté auprès d'une communauté d'utilisateurs militaires potentiellement intéressés, il reçut un accueil plutôt favorable.

2 MODELISATION

2.1 Introduction

Où il est dit que tout n'est pas 3D dans la 3D, où l'on parle de géométrie et de topologie et où l'on s'entend sur des noms de primitives.

Cette partie se propose de faire le tour d'horizon des différentes techniques et solutions envisageables pour modéliser un objet en trois dimensions. Notre présentation privilégie l'univers des systèmes d'information géographique sans pour autant négliger d'autres domaines applicatifs fortement demandeurs de 3D tels que la CAO, la synthèse d'image ou les jeux. Nous n'aborderons pas dans le détail les modèles 2D, un état de l'art sur ces derniers pourra être trouvé dans [DE LA LOSA 2000] ou [DE CAMBRAY 1994].

Les modèles abordés dans ce chapitre peuvent être classés en 3 grands types :

- **Les modèles 2,5D** : ce type de modélisation se retrouve notamment chez les éditeurs de SIG ayant construit leur notoriété autour d'une gamme de logiciels et d'applications 2D et souhaitant aujourd'hui offrir de nouvelles possibilités 3D aux utilisateurs de leurs produits. Leurs modules dits 3D se limitent très souvent au stockage d'une unique altitude z par couple de coordonnées (x,y).

- **Les modèles 3D purement géométriques** : Ces modèles sont exploités depuis de nombreuses années en CAO, synthèse d'image ou dans les jeux. Trois principes de modélisation sont couramment utilisés en la matière : la modélisation volumique, la modélisation surfacique et la modélisation par balayage.

- **Les modèles 3D topologiques** : A l'image de ce qui existe en 2D, certains laboratoires et instituts de recherche proposent de dépasser le simple descriptif géométrique de l'objet 3D et d'ajouter une description spécifique des relations topologiques entre primitives ou objets [TROTT 1999], [DE LA LOSA 2000], [MOLENAAR 1990], [SHAW 1998].

Nous commencerons donc par passer en revue les différentes modélisations existantes, puis nous consacrerons la seconde partie de ce chapitre à une présentation détaillée du modèle conceptuel de données retenu dans le cadre de notre travail de thèse.

Avant toute chose et par souci de clarté, il convient de s'entendre sur le vocabulaire utilisé pour nommer les primitives géométriques ou topologiques utilisées dans les modèles que nous allons aborder. En effet, selon les auteurs des modèles conceptuels, des noms différents sont employés pour décrire des primitives aux caractéristiques et propriétés similaires. Aussi proposons-nous de réserver les mots Nœud, Arc, Face et Volume à la désignation de primitives topologiques et d'utiliser les mots Sommets, Arête, Surface (ou Facette) et Solide pour leurs équivalents géométriques.

2.2 Les techniques de modélisation

Dans lequel un œil critique est porté sur les techniques de modélisation 3D les plus en vogue, la 2,5D n'ayant pas la cote.

2.2.1 La modélisation 2.5 D

Il n'existe pas aujourd'hui, parmi les systèmes d'informations géographiques les plus répandus, de logiciels capables de manipuler toute forme de données 3D. Lorsqu'un éditeur met en avant ses solutions dites 3D, celles-ci ne prennent en charge qu'une 3D approximative, certes suffisante pour un nombre non négligeable d'applications, mais pouvant se révéler trop restrictive avec la tendance actuelle et sans doute irréversible d'un besoin de détail sans cesse croissant. De manière générale, nous qualifions de 2,5D ces modélisations dont la caractéristique essentielle est de faire correspondre à un couple (x,y) de coordonnées géographiques une et une seule altitude Z. Il existe quelques variantes autour de ce type de modélisation résumées dans le tableau comparatif suivant [GERBE 2002].

Dimension	Description	Exemple de SIG
2D ¼	Le Z des objets est géré comme un attribut des objets, ce qui le limite à quelques valeurs possibles par objet (Zmin, Zmax par exemple). Les fonctions topologiques, d'analyse ou de visualisation n'utilisent pas le Z.	MapInfo (MapViewer)
2D ½	Tous les points de chaque objet sont stockés avec leurs trois coordonnées : X, Y, Z. Le SIG manipule des points et des lignes 3D, mais contient différentes restrictions : · pas de surface 3D (mais possibilité d'avoir des surfaces 2D avec des limites en 3D) · un seul Z par point 2D · primitives géométriques 3D mais moteur topologique 2D (lignes 3D mais les intersections ou les distances sont calculées en 2D)	ArcGis (3D analyst) GeoConcept (Virtual Geo)
2D ¾	Gère les surfaces 3D et le moteur topologique inclus des fonctions 3D. Les calculs de distances et d'intersections restent en 2D.	Cadcorp (SIS)
3D	Gère des données 3D (jusqu'aux volumes tels que les sphères) y compris la sémantique. A des capacités d'analyse 3D.	?

Tableau 2 : De la 2D à la 3D [GERBE 2002]

De la 2D à la 3D, la complexité de modélisation s'échelonne sur 4 niveaux. Une application de niveau complexe proposera des fonctionnalités avancées que d'autres applications aux modèles de données plus simples ne pourront pas mettre en œuvre.

Les solutions proposées autour de la thématique 2,5D sont aujourd'hui légions et s'étendent à travers de nombreux domaines d'applications. L'U.S. Army Topographic Engineering a produit en juin 2002 un document de 710 pages recensant quelques 600 solutions commerciales disponibles traitant de visualisation dîtes 3D [JORGENSEN 2002]. Toutes les branches de la 3D y sont abordées ainsi que

tous les domaines d'applications, mais les solutions dédiées à l'information géographique sont en nombre très limité, et d'autant plus limité si on écarte les applications spécifiquement raster. Il ne reste alors plus que quelques dizaines de logiciels pouvant être classés en 3 catégories que nous détaillons ci-dessous, avec une liste non exhaustive d'exemples de logiciels pour chacune d'entre elles.

Logiciels de visualisation 3D :

- Logiciel World Construction Set, 3D Nature : Modélisation et la visualisation de terrains photo-réalistes.
- Logiciel Virtual Forest et RAPID Surfing, Pacific Meridian Resources : Visualisation en foresterie et survol de modèle de terrain sous Arc/Info
- Logiciel Vertical Mapper, Northwood Geoscience : Extension pour la visualisation 3D du logiciel MapInfo.

Logiciels de visualisation et navigation 3D :

- Logiciel Fly!, PCI Geomatics : Outil de visualisation de terrain et de création de scènes 3D texturées ;
- Logiciel ArcView 3D Analyst, ESRI : Extension du SIG ArcView pour la visualisation et la navigation dans un environnent 3D ;
- Module 3D for Geoconcept, Geoconcept SA : Extension 3D de GeoConcept pour la création de modèles numériques de terrain (MNT), scènes 3D ou profils en long.

Logiciels de navigation, visualisation et possédant des fonctions d'analyses :

- Logiciel Environmental Visualization System EVS, C Tech Development Corp. : Visualisation 3D et analyse géostatistique d'environnements miniers et sites contaminés ;
- Erdas Logiciels IMAGINE Virtual GIS, ERDAS : Outil de visualisation et d'analyse supportant des fonctionnalités SIG dans un environnement 3D ;
- Logiciel SIS , Cadcorp : Logiciel de SIG présentant des possibilités 3D et notamment utilisé par l'Institut Géographique National français.
- Virtual Geo, Communication & Systemes : Logiciel offrant des fonctionnalités de navigation, visualisation et de SIG. Initialement utilisé comme extension de GeoConcept, il peut s'adapter aux autres SIG classiques.

L'objectif de notre travail n'est pas de développer une application 2,5D : les techniques de modélisation 3D étant radicalement différentes de ce qui se fait en 2,5D, il ne nous apparaît pas indispensable de traiter plus en profondeur la problématique 2,5D. Nous conseillons au lecteur intéressé de se reporter aux sites web référencés à la fin du mémoire pour obtenir davantage de précisions sur les modèles 2,5D.

2.2.2 La modélisation 3D géométrique

Dans ce chapitre, nous recensons les principaux modèles couramment utilisés dans des domaines d'application 3D tels que la conception assistée par ordinateur (CAO) ou le secteur du jeu, souvent innovateurs en la matière. Nous traitons ici de modélisation purement géométrique en ne faisant aucune référence à la topologie.
Les modèles que nous présentons sont répartis en trois grands types : les modèles volumiques, les modèles surfaciques et la modélisation par balayage.

2.2.2.1 Modélisation par balayage

Le principe de cette modélisation repose sur l'association d'une surface 2D et d'une trajectoire. Le déplacement de la surface 2D le long de la trajectoire permettant d'aboutir à une forme 3D. On distingue principalement quatre types de balayage selon les caractéristiques de la trajectoire :

- le balayage translationnel ou linéaire, le plus simple (Figure 2)
- le balayage rotationnel, bien adapté au monde de la mécanique industrielle (Figure 3)
- le balayage hybride, permettant de représenter des objets plus complexes grâce à l'union de plusieurs balayages simples
- le balayage généralisé, offrant la possibilité de modifier la surface balayée le long de la trajectoire

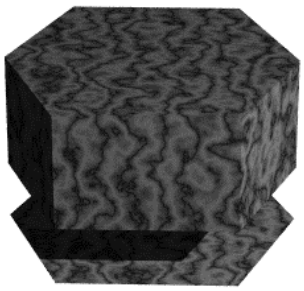


Figure 2 : Balayage linéaire

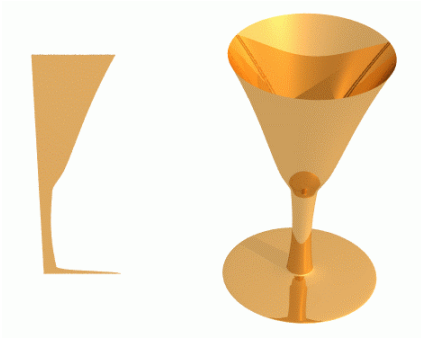


Figure 3 : Balayage de révolution

Notons que la simple union de deux objets balayés ne forme par forcément un objet balayé (Figure 4). Pour cette raison, certains outils de modélisation proposent cette technique pour construire des objets, mais les transforment ensuite dans une autre représentation pour pouvoir mieux les manipuler.

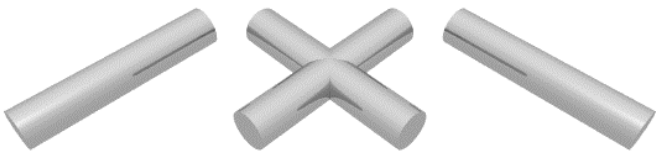


Figure 4 : Combinaison de deux objets balayés

Le balayage linéaire peut être très utile pour représenter des objets géographiques. En effet, sur une image aérienne, il est facile de saisir le contour d'un bâtiment et de faire ensuite « descendre les murs ». De la même manière et à l'aide d'un MNT, on peut transformer la BD Topo de l'IGN en données 2,5D. Les objets ainsi créés sont une bonne approximation du monde réel.

2.2.2.2 Modélisations surfaciques

Bien qu'offrant une modélisation 3D à part entière, les modèles surfaciques ne modélisent pas le volume d'un objet, ils modélisent le contour ou l'enveloppe de cet objet.

2.2.2.2.1 Modèle fil de fer

Le modèle fil de fer fut le premier modèle vecteur en trois dimensions. Ce modèle ne conserve que les coordonnées(x ,y, z) des sommets et les arrêtes les joignant. C'est donc un modèle très simple. Il nécessite un minimum de puissance de calcul et de place mémoire sur ordinateur. Cette modélisation est cependant ambiguë puisqu'elle ne permet pas, d'une part, de distinguer le vide du plein et peut, d'autre part, poser des problèmes de visualisation (Figure 5).

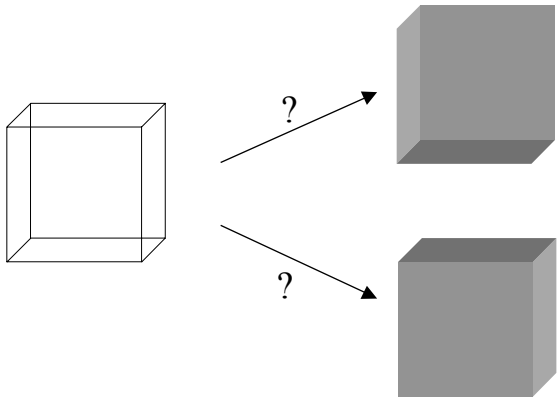


Figure 5 : Ambiguïté de visualisation avec la modélisation Fil de Fer

2.2.2.2.2 Boundary representation (B-Rep)

La modélisation par Boundary Representation, qui succède à la modélisation fil de fer, ne s'intéresse qu'à l'enveloppe, c'est-à-dire à la frontière des objets. Cette frontière comprend des facettes, des arêtes et des sommets. Suivant les cas, les facettes peuvent être des polygones plans (voir même de simples triangles) ou des surfaces courbées plus ou moins complexes utilisant notamment des quadratiques. La Figure 6 montre un galion représenté en B-Rep. A droite, on peut voir ce même galion auquel on a retiré quelques facettes sur la proue.

Figure 6 : Exemple de B-Rep

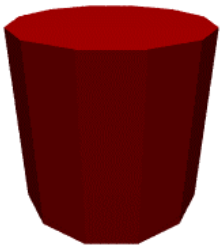


Figure 7 : Cylindre modélisé avec des faces planes

Dans le cas où les facettes sont obligatoirement planes, la représentation des surfaces courbes est délicate car il faut les discrétiser. Pour avoir un rendu réaliste, le nombre de facettes doit être très important.

Une méthode couramment employée pour contourner le défaut de forme consiste à associer des normales aux sommets des facettes. Ces normales sont calculées à partir de la surface courbe réelle et sont utilisées pour lisser le rendu en modifiant l'incidence de la lumière.

Cette approche présente cependant des faiblesses puisque, chaque facette étant définie indépendamment des autres, elles peuvent se croiser ou se joindre approximativement. Afin de contourner ce problème, on peut envisager d'ajouter un niveau de topologie « locale » à un objet garantissant la cohérence de la représentation et des déformations de l'objet. Un objet n'est plus alors un ensemble de facettes élémentaires mais un ensemble cohérent de nœuds, arcs et faces (voir 2.2.3). La modélisation ne s'intéressant qu'à la frontière de l'objet, on n'a pas de notion de volume. Cette notion peut cependant être introduite en utilisant des faces orientées.

2.2.2.3 Modélisations volumiques

Les modélisations volumiques s'appuient sur des associations et des combinaisons de primitives de dimension trois (volumiques). Les objets ne sont plus représentés par leurs contours, ils sont représentés par leurs volumes.

2.2.2.3.1 Notion de Regularized Boolean Set Operation (RBSO)

Quelle que soit la manière adoptée pour représenter des objets 3D, une problématique générale consiste à pouvoir décrire des objets complexes sous la forme de combinaisons d'objets plus simples. Pour ce faire, une des méthodes les plus intuitives consiste à décrire ces objets complexes sous la forme de combinaisons d'opérations de type booléen sur des objets simples : union, différence, intersection. Sur la base de ces opérations unitaires, un ensemble d'opérateurs appelés les « opérateurs booléens régularisés » (Regularized Boolean Set Operations) permet de garantir que les résultats

d'opérations booléennes sur des solides aboutissent dans tous les cas à un solide. En effet, le résultat d'une intersection standard (Figure 8) peut être un volume (a), un plan (b), une droite (c), un point (d) ou vide (e). Le résultat de l'intersection régularisée est toujours un volume. Le résultat est vide dans les cas (b), (c) et (d).

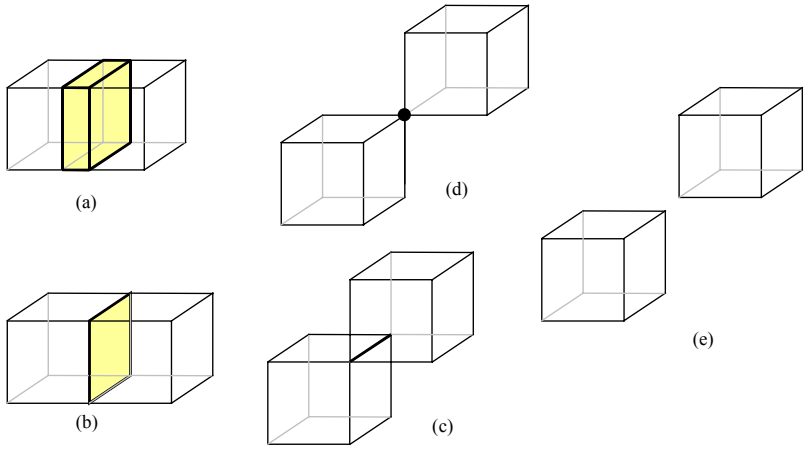


Figure 8 : résultat d'intersections standards

Cette méthode de combinaison d'objet peut être utilisée dans les techniques de modélisation décrites ci-dessous.

2.2.2.3.2 Constructive Solid Geometry (CSG)

Le modèle CSG combine des primitives simples à l'aide d'opérateurs RBSO qui font alors partie intégrante de la modélisation. La Figure 9 présente les résultats de la différence, l'intersection et l'union d'un cube et d'un cylindre. Le pouvoir de représentation d'une telle technique est d'autant plus important que la panoplie de primitives est riche. En effet, un premier niveau de CSG pourrait n'avoir que le cube comme primitive. Un deuxième niveau pourrait également intégrer la sphère, etc. On peut envisager d'utiliser des primitives non-bornées comme par exemple des demi-espaces pour pouvoir couper une primitive par un plan. L'utilisation de demi-espaces peut poser des problèmes puisque certaines opérations ne produisent pas des solides bornés.

Figure 9 : Différence, intersection et union

Un objet réalisé par la technique CSG est constitué d'un arbre dont les nœuds sont des opérateurs booléens ou des transformations géométriques et les feuilles des primitives simples. Pour déterminer les propriétés physiques d'un objet CSG, il faut combiner les propriétés des feuilles de l'arbre en l'explorant en profondeur.

Il est à noter que la CSG ne permet pas de représenter un solide donné d'une manière unique. La Figure 10 montre deux représentations différentes d'un même objet. Le premier cas est la conséquence d'une soustraction entre deux objets, alors que le second cas est issu d'une union.

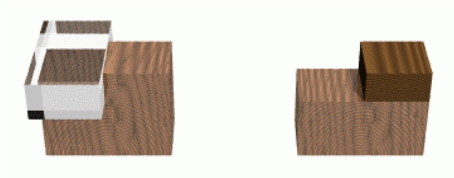


Figure 10 : Deux manières de représenter le même objet par CSG

2.2.2.3.3 Spatial Partitioning Representations (SPR)

Le modèle SPR décompose un solide en un ensemble de solides élémentaires voisins les uns des autres et ne s'intersectant pas.

Une méthode de décomposition dérivée du SPR est l'énumération spatiale. Cette dernière procède au découpage de l'espace 3D en petits cubes élémentaires appelés « voxels » (pour « volume element » et par analogie avec le terme « pixel »). La délimitation de l'emprise volumique d'un objet dans l'espace 3D s'effectue en précisant, pour chaque voxel, s'il est occupé par le solide ou pas. Cela correspond à l'extension directe des images raster à la 3D.

La Figure 11 présente un tore modélisé suivant cette procédure.

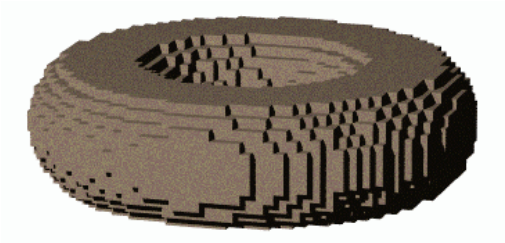


Figure 11 : Construction d'un tore à partir de cubes par SPR

Cette méthode présente plusieurs avantages :

- Il est très simple de distinguer l'intérieur et l'extérieur d'un solide ;
- Le calcul du volume du solide peut se faire (d'une manière approchée) quelque soit la forme de ce solide en comptant le nombre de cellules occupées.

L'énumération spatiale souffre cependant de plusieurs inconvénients :

- La modélisation des objets ne peut être qu'approximative. Malgré le nombre de cubes utilisés pour modéliser le tore de la Figure 11, les voxels apparaissent nettement ;

- Le nombre de voxels nécessaires pour représenter une scène peut devenir rapidement important.

2.2.2.3.4 Modélisation par Octree

Une variante de la technique de modélisation SPR permet de palier le défaut du volume des informations nécessaires à la modélisation d'un objet. Il s'agit de la technique de modélisation par Octree. Cette technique est une simple extension 3D des quadtree et permet de remplir l'espace avec des cubes de différentes tailles (Figure 12). Chaque cube étant une puissance de deux du cube élémentaire, ce type de modélisation est d'autant plus adapté au stockage numérique.

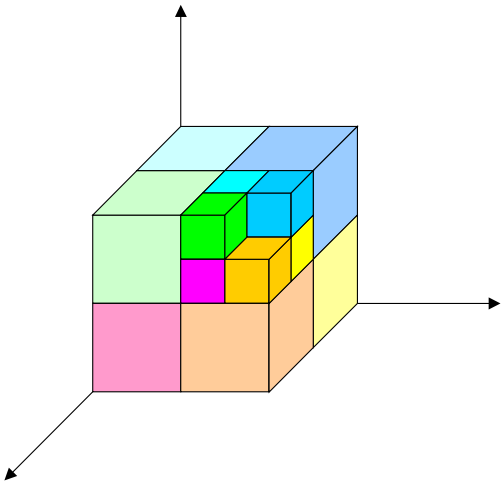


Figure 12 : Octree

2.2.2.3.5 Instanciation de primitives

Avec la modélisation par instanciation de primitives, l'utilisateur dispose d'une bibliothèque d'objets graphiques correspondant à son domaine d'application qu'il peut utiliser à souhait pour construire sa scène 3D. Par exemple, un photo-interprète pourrait avoir des objets de type : maison, hangar, bâtiment, arbre,… Il peut également importer des objets provenant d'autres applications ou d'une banque de données d'objets 3D.

Ces objets peuvent être paramétrés pour correspondre au besoin de l'utilisateur. Une maison peut ainsi être paramétrée par sa largeur, sa longueur, ou son nombre d'étages. La Figure 13 présente un exemple d'instanciation de primitives dans le domaine de la photo-interprétation. Dans cet exemple, on utilise deux primitives : maison et arbre.

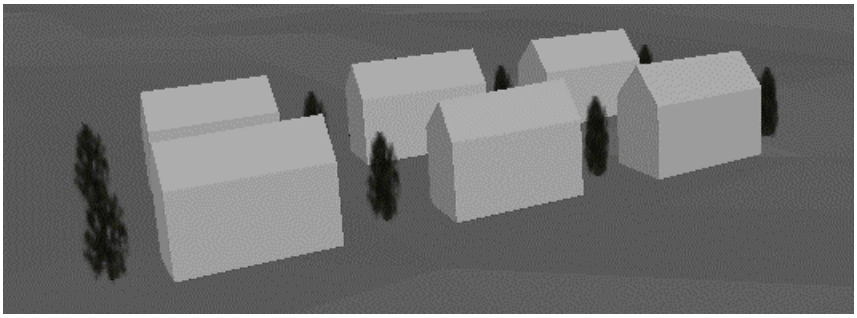


Figure 13: Instanciation de primitives

L'instanciation de primitives ne permet pas la combinaison d'objets (en utilisant les RBSO par exemple) sans conversion dans un autre modèle. La seule manière d'ajouter de nouveaux objets est de les créer de toute pièce par programmation. De plus, chaque objet doit porter ses propres fonctions de dessin et de calcul de propriétés (volume, masse…).

2.2.2.4 Conclusion sur les modèles géométriques

Nous avons vu au travers de ce chapitre que les possibilités de modélisation 3D sont nombreuses et diversifiées. Cependant, ces modèles ne sont pas tous adaptés aux besoins spécifiques d'un système d'information géographique tridimensionnel et nous verrons au paragraphe 2.3.1 les critères qui nous ont amenés à choisir le modèle B-Rep. Auparavant, nous allons nous intéresser aux problématiques topologiques ainsi qu'à l'utilité d'une topologie dans un SIG 3D.

2.2.3 Modélisation 3D Topologique

2.2.3.1 Définition de la topologie adaptée au besoin des SIG

La topologie est la branche des mathématiques s'intéressant aux propriétés des objets invariantes par homéomorphisme (bijection continue dont la réciproque est continue), ou plus simplement invariantes par déformation, rotation, translation, étirement mais sans déchirement.

En termes d'opérations ensemblistes, une définition formelle de la topologie est la suivante (topologie définie par les ouverts):

Soit E un ensemble, et soit P(E) l'ensemble des parties de E. On dit qu'un sous-ensemble O de P(E) est un ensemble d'ouverts de E si O vérifie les axiomes suivants :

1. *Le sous ensemble trivial E et l'ensemble vide ∅ sont éléments de O.*
2. *Toute réunion d'éléments de O est un élément de O.*
3. *Toute intersection finie d'éléments de O est un élément de O.*

L'ensemble E muni d'une famille d'ouverts est appelé espace topologique, l'ensemble des ouverts O définit la topologie de E ; on note (E,O) l'espace topologique ainsi défini.

Bien que cette définition de la topologie soit exacte et rigoureuse, elle n'est finalement que peu utilisée par la communauté des géomaticiens. Lorsqu'en géomatique on parle de topologie, on se réfère d'avantage aux concepts véhiculés par la théorie des graphes. Un graphe est une relation binaire à l'intérieur d'un ensemble. Si on appele nœuds les éléments de cet ensemble et arcs les couples de nœuds en relation, on peut dire qu'un graphe est aussi la donnée d'un couple (N,A) où N est l'ensemble des nœuds et A l'ensemble des arcs, formés de couples de nœuds (on a donc A est inclus dans NxN). En d'autres termes, un graphe est un couple d'ensemble (N,A) où A est une famille de couples (orientés ou non) d'éléments de N.

Encore une fois, il convient d'être prudent face à cette définition et veiller, d'après [LANGLOIS 1994], à ne pas confondre le concept de graphe avec le dessin qui le symbolise, d'autant qu'en géomatique le graphe porte sur des éléments géométriques.

Afin de satisfaire les considérations géomatiques, sauf exceptions pour ne pas être en contradiction avec les auteurs lors de la description de certains modèles topologiques, nous entendrons par topologie l'ensemble des relations d'adjacences reliant les primitives de forme entres elles. Ces primitives sont au nombre de quatre et leurs primitives topologiques et géométriques associées sont présentées dans le Tableau 3. Nous excluons les relations d'ordre ou les relations directionnelles qui ne peuvent être considérées comme des relations topologiques.

Dimension de la Primitive	Primitive Géométrique Associée	Primitive Topologique Associée
0	Sommet	Nœud
1	Arête	Arc
2	Facette	Face
3	Solide	Volume

Tableau 3 : Primitives Géométriques et Primitives Topologiques

2.2.3.2 Intérêt de la topologie

Que nous nous placions en 2D ou en 3D, la topologie est utile en de nombreux points lorsqu'elle vient compléter une modélisation géométrique. Voici quelques arguments avancés en faveur d'un modèle topologique :

- Compte tenu du fait que certains calculs géométriques d'inclusion, d'adjacence, de frontière ou de parcours de réseaux sont extrêmement gourmands en ressource système, l'organisme international pour la standardisation (ISO) préconise de s'appuyer sur les structures combinatoires des complexes topologiques pour traduire des calculs géométriques coûteux en algorithmes combinatoires [ISO/TC211 2001].

- La modélisation topologique permet de garantir une cohérence au sein des données décrites. Dans un modèle purement géométrique, chaque primitive composant l'objet 3D est indépendante de ses voisines, ce qui peut être source d'aberrations de forme en cas d'approximation, ou d'incertitude dans la saisie ou la modification des objets.

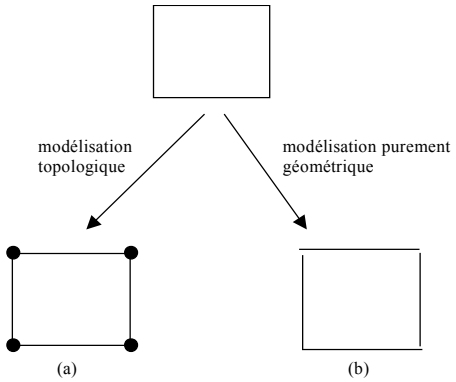


Figure 14 : La topologie pour une meilleure cohérence des données

La Figure 14 illustre ce propos et montre qu'en imposant des contraintes sur les arêtes et les sommets, l'utilisation de la topologie réduit considérablement les risques d'aberrations lors de la modélisation d'une forme (ici un trivial rectangle).

- Karine Zeitouni affirme dans le cadre de ses travaux [ZEITOUNI 1995] que le premier intérêt de la topologie réside dans sa sémantique spatiale. Elle considère que la façon dont les objets sont connectés dans un espace donné constitue un modèle naturel, plus explicite qu'un modèle géométrique pour les utilisateurs de SIG. En effet, dans le langage commun, les objets ne sont pas désignés par leurs coordonnées géométriques mais par leurs positions relatives les uns par rapport aux autres. Elle en conclue donc que ces propriétés de « topologie sémantique » sont plus importantes qu'une pure description géométrique.

- Parfois la topologie est présentée comme un moyen de réduire le volume de stockage des données. C'est un point à nuancer, puisque s'il est vrai que la topologie permet d'éviter de nombreuses redondances induites par la duplication des primitives géométriques, la description détaillée des relations topologiques implique tout de même un surcoût de stockage.

2.2.3.3 Les modèles topologiques 3D

Les précédents travaux en matière de modélisation topologique 3D semblent tous convenir qu'une modélisation géométrique par frontière est la plus adaptée à l'ajout d'une information topologique. Cependant deux approches assez différentes sont généralement proposées [RAMOS 2002].

- Une première approche classique décompose l'espace topologique en quatre types de primitive : les nœuds, les arcs, les faces et les volumes. Cette approche est la plus couramment utilisée.

- Une seconde approche s'inspire du concept de carte combinatoire topologique (Winged-edge data structure et DCEL [DAVID 1991]). Cette approche introduit un nouveau type de primitive né de l'association de la face, primitive non orientée, et de l'arc, primitive orientée.

Dans ce chapitre nous ne rentrerons pas dans le détail de chaque modèle topologique 3D, nous nous contenterons d'en faire ressortir les grandes lignes tout en sachant qu'elles peuvent être déclinées différemment selon les objectifs à atteindre.

2.2.3.3.1 La modélisation classique

2.2.3.3.1.1 Le squelette

L'approche par modélisation classique de la topologie est assez intuitive, c'est celle adoptée par de nombreux auteurs parmi lesquels nous retrouvons Martien Molenaar, un des pionniers sur cet axe de recherche [MOLENAAR 1990]. Ce modèle conceptuel repose sur les quatre primitives de base que sont le nœud, l'arc, la face et le volume, reliées entre elles par des relations de construction et d'inclusion.

La Figure 15 représente le squelette général qui se retrouve dans chacun des modèles issus de cette approche conceptuelle.

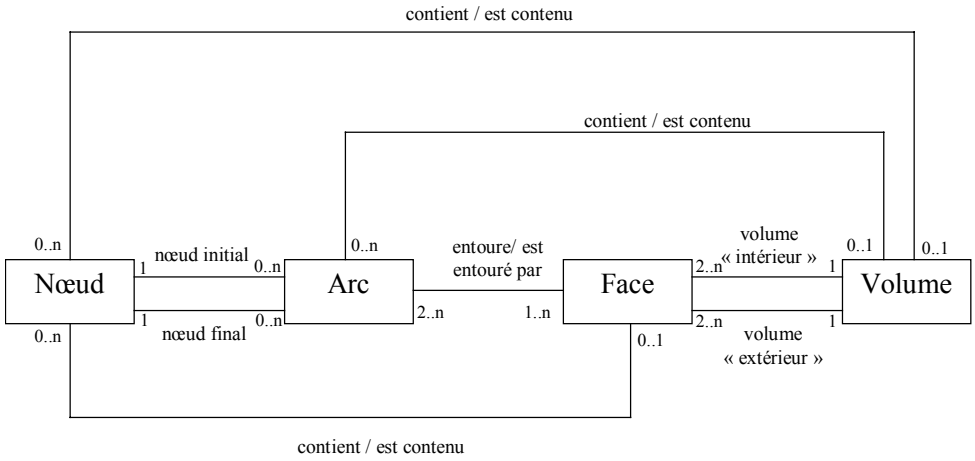


Figure 15 : Diagramme UML schématisant le squelette général d'une modélisation topologique 3D classique

Ce modèle s'appuie sur des principes de modélisation 2D classiques, les primitives géométrico-topologiques Nœud, Arc, Face sont issues de standard 2D, on les retrouve par exemple dans le modèle de donnée VPF (Vector Product Format). Elles ont simplement été étendues à un contexte tridimensionnel.

2.2.3.3.1.2 Définition des primitives

Le Nœud :

Le nœud est l'équivalent topologique d'une primitive géométrique « sommet » de dimension 0. Il est généralement associé à un triplet de coordonnées. Une distinction est parfois faite entre les nœuds reliés à d'autres primitives et les nœuds isolés.

L'Arc :

L'arc est l'équivalent topologique d'une primitive géométrique « arête » de dimension 1. Il est orienté par son nœud initial et son nœud final. Une chaîne ordonnée d'arc délimite la frontière d'une face. Comme pour les nœuds, certains arcs peuvent se trouver isolés au sein de volumes. Quelques modèles acceptent les boucles, c'est à dire les arcs ayant leurs nœuds initiaux et finaux confondus. Les arcs ne peuvent ni se superposer ni s'intersecter en dehors des nœuds, on parle alors de topologie planaire. Les primitives géométriques associées à la primitive topologique arc sont plus ou moins complexes d'un modèle à l'autre. Certains n'accepteront qu'une association avec un segment, d'autres supporteront les lignes brisées, voire même des portions de courbes aux équations plus complexes.

La Face :

La Face est l'équivalent topologique d'une primitive géométrique « surface » de dimension 2. Une face est délimitée par les arcs qui l'entourent. Elle peut également être orientée par ces mêmes arcs. Les contraintes de superpositions et d'intersection s'appliquent également à la face, une face ne peut donc en intersecter une autre que par l'intermédiaire d'un arc ou d'un sommet (dans ce dernier cas on parle alors de structure non eulerienne). Certains modèles admettent les faces à « trous », d'autres ne les supportent pas. De même que pour les arcs, la primitive géométrique associée à une face peut être plus ou moins complexe, à savoir une simple surface planaire, une surface triangulée ou une surface décrite par une équation complexe de type surface de Bézier ou NURBS. Enfin une face peut être isolée au sein d'un volume.

Le Volume :

Le Volume est l'équivalent topologique d'une primitive géométrique « solide » de dimension 3. Un volume est délimité par un ensemble de faces constituant son enveloppe. Certaines modélisations définissent une partition de l'espace, auquel cas les volumes ne peuvent s'intersecter et un volume « universel » est introduit. Ce volume universel est la partie de l'espace privée de l'ensemble des unions de volumes définis par leurs enveloppes. Par analogie avec les trous des faces, quelques modèles introduisent le concept de cavité.

2.2.3.3.1.3 Quelques variantes autour du modèle

Nous présentons ici quatre variantes de modèles topologiques inspirés du squelette présenté en 2.2.3.3.1.1. Les diagrammes UML de ces modèles pourront être trouvés en Annexe à la fin du mémoire.

- Le modèle proposé par M. Molenaar en 1990 est souvent considéré comme un des premiers modèles aboutis en termes de modélisation topologique 3D. Le modèle de données FDS (Formal Data Structure) présenté dans [MOLENAAR 1990] est accompagné de douze conventions dont certaines sont sujettes à quelques critiques, notamment la convention 8 qui stipule que les faces doivent être planaires et qui est souvent considérée comme beaucoup trop limitative. Nous pensons pourtant qu'il s'agit là de critiques mineures sur des points qui peuvent être revus et adaptés selon les objectifs de chacun. Le concept et la philosophie de ce modèle conceptuel, et plus particulièrement dans ses aspects topologiques, sont quant à eux très largement repris dans de nombreux travaux.

- L'Organisme International de Standardisation (ISO) préconise pour sa part l'utilisation d'un modèle conceptuel de donnée très proche du modèle classique, mais recommande fortement de rendre indépendantes la géométrie et la topologie, tout en instaurant un système de passerelles entre les deux. Pour une présentation détaillée des modèles géométriques et topologiques préconisés par l'ISO, le lecteur pourra consulter le document [ISO/TC211 2001].

- Le DARPA image understanding, organisme américain, a développé un concept intéressant de k-chaîne où k est la dimension des éléments liés sous forme de chaîne. Ainsi une 0-chaîne est une séquence de points, une 1-chaîne une séquence d'arcs et une 2-chaîne une séquence de faces. Ce concept est repris par le moteur topologique disponible dans TargetJr [TargetJr 1997].

- K. Trott reprend partiellement cette idée de k-chaîne dans sa proposition d'extension 3D du format VPF. Il nomme « ring » les 1-chaînes, c'est à dire les séquences d'arcs et « shell » les 2-chaînes, c'est à dire les séquences de faces [TROTT 1999]. Contrairement au moteur topologique de TargetJr, la notion d'entité volumique est bien présente sous le modèle VPF 3D. TargetJr s'arrête à la notion de contour de volume alors que VPF 3D établit une partition spatiale de l'espace.

2.2.3.3.2 Extension 3D des cartes topologiques

Les modèles topologiques classiques, qu'ils soient 2D ou 3D, ne traitent que de relations de composition entre primitives. Les cartes topologiques 2D ont été introduites pour ajouter une notion de cycle d'arcs autour d'un nœud. Cette notion de cycle d'arcs est reprise dans plusieurs modèles 2D dont la structure DCEL [DAVID 1991] ou la structure de données « winged-edge ».

L'idée d'une extension 3D de ce concept de carte topologique a été abordée indépendamment mais de façons similaires à l'université de la Nouvelle Orléans par R. Ladner [LADNER 1998] et K. Shaw [SHAW 1998] et à l'IGN par A. De La Losa et B. Cervelle [DE LA LOSA 1999]. Tous proposent d'articuler la description des objets 3D autour des arcs en introduisant une nouvelle primitive associant l'arc (primitive orienté) et la face (primitive non-orienté). Suivant les auteurs, cette primitive porte le nom de EFace (Ladner-Shaw) ou de couple (Arc,Face) (De La Losa-Cervelle).

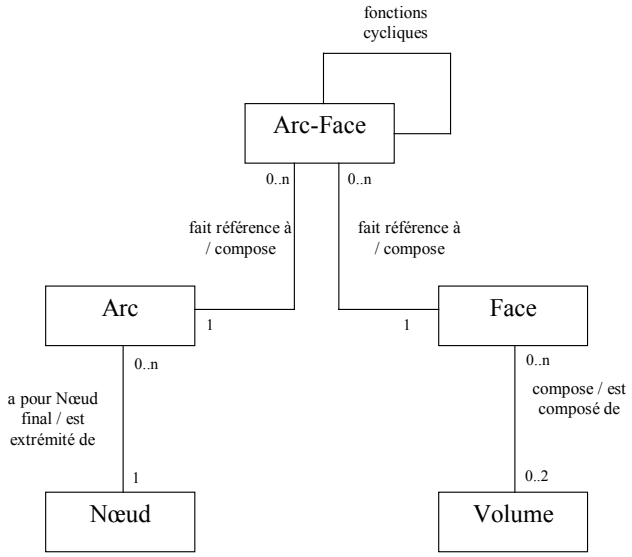


Figure 16 : Diagramme UML schématisant une extension 3D des cartes topologiques

La Figure 16, qui résume le schéma général de ces modèles inspirés de cartes topologiques, montre clairement que les primitives arc et face ne sont plus directement liées. La primitive (Arc,Face) est une véritable clef de voûte autour de laquelle s'organisent les relations entre primitives. Ce type de modélisation offre des possibilités de navigation de proche en proche entre primitives, possibilités supérieures aux modèles s'appuyant simplement sur des relations de compositions.

2.2.3.3.3 D'autres tentatives de modélisation topologique

Les deux approches que nous venons de présenter, à savoir l'approche classique et l'approche inspirée des cartes topologiques, représentent les deux principales tendances en modélisation topologique aujourd'hui. Cependant d'autres propositions plus originales, aux caractéristiques intéressantes, méritent d'être citées.

2.2.3.3.3.1 L'hypergraphe d'adjacences de face

Dans l'hypergraphe proposé par Leila De Floriani [DE FLORIANI 1988], seules les relations d'adjacences de faces sont prises en compte, tout en différentiant les adjacences de faces le long d'arcs et les adjacences de faces par les nœuds.

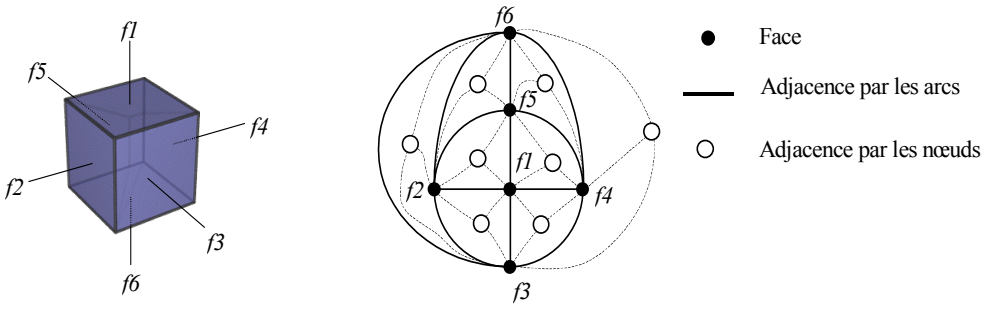


Figure 17 : Les hypergraphes d'adjacences de faces (figure empruntée à [DE LA LOSA 2000])

2.2.3.3.3.2 Le modèle 3DGT

Karine Zeitouni [ZEITOUNI 1995] pense qu'il est utile de distinguer les éléments verticaux et les éléments horizontaux dans la modélisation de l'information géographique. D'autre part, elle insiste sur la nécessité d'une relation de type « poser sur » [DE CAMBRAY 1994] pour notamment assurer une cohérence entre le MNT et le « sur-sol ».

2.2.3.4 Conclusion sur les modèles topologiques

Ce chapitre nous a permis de recenser quelques-uns des principaux modèles topologiques. Ces derniers ne se posent pas en alternative des modèles géométriques vus au chapitre précédent, en revanche ils peuvent leur être complémentaires.

Devant tous ces modèles, il nous a fallu faire des choix que nous allons maintenant exposer. Ces choix sont d'ordres topologiques et géométriques, ils constituent le cœur de notre prototype de système d'information géographique tridimensionnel.

2.3 Choix d'une modélisation des données

Où des choix sont faits pour donner naissance à notre modèle conceptuel. Loin d'être universel, il sera, on l'espère, apte à servir nos besoins.

Le modèle de données est le cœur d'un système d'information géographique. Le chapitre que nous entamons présente et justifie nos choix de modélisation tant sur le plan géométrique que topologique. Ces choix concernent principalement les objets du sur-sol, pris séparément ou associés dans un réseau. Bien que ce soit de manière sommaire, nous abordons également dans cette partie les différentes solutions envisageables en matière de Modèles Numériques de Terrain.

2.3.1 Choix du modèle géométrique

2.3.1.1 Liste des critères

Le choix d'un modèle géométrique nécessite de définir un certain nombre de critères de comparaison. La liste qui suit présente les critères que nous avons retenus pour évaluer les différents modèles géométriques [BERNARD 2000] :

- Etendue du domaine : le domaine d'une représentation doit être suffisamment large pour permettre à un ensemble utile d'objets physiques d'être représenté.

- Complétude : une représentation est complète si elle est bi-univoque, c'est-à-dire s'il n'existe aucune équivoque possible sur la nature de ce qui est représenté, et si une représentation donnée correspond à un objet et un seul.

- Unicité : une représentation est unique si elle peut être utilisée de manière unique pour décrire un objet donné : si la représentation est unique, les opérations de type « égalité » entre deux objets sont alors faciles à mettre en œuvre et peu coûteuses en temps de calcul.

- Exactitude : une représentation est exacte si un objet quelconque peut être représenté sans approximation.

- Validité : une méthode de représentation valide est une méthode de représentation telle qu'il soit impossible de créer une représentation qui corresponde à un objet n'ayant pas une réalité physique (objet « absurde »).

- Compacité : une représentation compacte permet d'optimiser l'espace mémoire nécessaire pour décrire l'objet.

- Efficacité : une représentation efficace est une représentation qui permet l'utilisation d'algorithmes efficaces pour calculer les propriétés physiques des objets et générer les images de synthèse.

- Visualisation statique : une représentation répond à ce critère s'il existe des composants permettant de générer une image de synthèse des objets 3D décrits sous cette forme.

- Visualisation dynamique : une représentation répond à ce critère s'il existe des composants permettant de générer une image de synthèse des objets 3D décrits sous cette forme, ceci en moins de 1/25ème de seconde.

- Potentiel topologique : ce critère évaluera la capacité d'une représentation à permettre d'effectuer des opérations topologiques sur les objets 3D décrits sous cette forme.

- Potentiel multi-échelle : une représentation répond à ce critère si elle permet facilement de gérer plusieurs types de représentations (niveaux de détails) des objets 3D décrits sous cette forme.

- Adaptativité à la production opérationnelle de données : une représentation répond à ce critère si elle est adaptée aux contraintes opérationnelles liées aux techniques actuelles de production de données 2D et 3D par extraction d'informations sur images sources.

2.3.1.2 Comparaisons et choix du modèle

A partir de la liste de critères que nous venons d'établir, nous avons comparé cinq des modèles géométriques 3D les plus couramment utilisés. Le tableau ci-dessous présente de manière synthétique le résultat de cette étude comparative.

	CSG	Instanciation de Primitives	B-Rep	SPR	Balayage
Domaine	=	-	+	-	-
Complétude	+	-	+	-	-
Unicité	-	+	=	+	=
Exactitude	+	=	-	-	-
Validité	+	+	-	+	+
Compacité	=	+	-	-	+
Efficacité	-	=	+	+	=
Visu Statique	+	+	+	+	+
Visu Dynamique	-	=	+	-	=
Potentiel Topologique	=	=	+	-	=
Potentiel Multi-échelle	-	=	+	-	=
Adaptabilité au besoin	=	+	+	-	+

Tableau 4 : Comparaison des techniques de modélisation géométrique

2.3.1.2.1 La CSG

La Constructive Solid Geometry (CSG) est une technique très riche. Si on utilise des primitives très petites, on peut couvrir un domaine très étendu et exact mais au prix d'une faible compacité et d'une grande difficulté de modélisation. Cette modélisation ne permet pas de garantir l'unicité de la représentation. En effet, un même objet peut être obtenu de plusieurs manières différentes. Les opérations du CSG sont du type Regularized Boolean Set Operations (RBSO). Ces opérateurs garantissent que le résultat sera toujours un solide (ou un objet vide). La représentation CSG permet

donc de ne faire que des objets valides. A l'heure actuelle, il n'existe pas d'outils permettant de faire de la visualisation dynamique d'un modèle CSG sans le convertir en BRep. En revanche, la plupart des outils de lancer de rayon (PovRay par exemple) permettent de faire de la visualisation statique de très bonne qualité.

2.3.1.2.2 L'instanciation de primitive

L'instanciation de primitive ne permet pas de modéliser une grande variété d'objets, puisqu'il s'agit d'instancier des primitives prédéfinies. Cependant, si le jeu de primitives couvre le besoin fonctionnel, il permet de couvrir l'étendue du domaine. Dans le domaine de la géographie numérique 3D, le jeu de primitives, aussi riche soit-il, ne peut pas couvrir l'intégralité du besoin. Il peut en revanche faciliter le travail du photo-interprète si les primitives sont judicieusement choisies. Il permet, de plus, de garantir l'unicité de la représentation : deux objets identiques dans le monde réel seront identiques dans le monde modélisé. En outre, c'est un modèle qui peut être très compact et un objet, aussi complexe soit-il, sera décrit par un ensemble réduit de paramètres. Une maison sera par exemple décrite par sa largeur, sa longueur et le nombre d'étages au lieu d'un ensemble de murs, de fenêtres et de portes.

2.3.1.2.3 La modélisation SPR

La modélisation par Spatial Partitioning Representations (SPR) est surtout utilisée dans les domaines scientifiques de simulation (modélisation scientifique par élément fini par exemple) et elle a également été utilisée dans les jeux. Elle souffre cependant de gros défauts : elle ne permet pas d'atteindre une grande complétude car l'élément de base est un cube, elle est très gourmande en espace mémoire et n'est pas très compacte, elle est mal adaptée pour une visualisation dynamique.

2.3.1.2.4 La modélisation par balayage

La modélisation par balayage n'a pas un très grand domaine d'application, tous les objets du monde réel ne peuvent pas être modélisés de cette façon. Elle permet cependant de toujours obtenir des objets valides et une grande compacité. Un volume est décrit par une surface 2D et une courbe de balayage. Elle peut cependant être utile dans notre domaine d'application sous une forme simplifiée très courante : une surface 2D horizontale et un balayage vertical. Cette technique permet de modéliser rapidement tous les bâtiments à toit plat et « d'approximer » beaucoup d'autres bâtiments. Elle peut être une aide à une saisie efficace.

2.3.1.2.5 Notre choix : La modélisation par frontière surfacique (B-Rep)

La modélisation par frontière surfacique (B-Rep), permet de couvrir un très grand domaine puisque la plupart des solides peuvent être représentés par leur surface. L'ajout d'artefacts comme des normales et une orientation des faces peut améliorer la visualisation dynamique et l'efficacité. Si on utilise des faces orientées, on pourra alors calculer facilement le volume du solide. Cette représentation n'est cependant pas unique, un même objet réel peut être modélisé par plusieurs représentations par surfaces. Cette technique a un très bon potentiel d'utilisation en multi-échelle, il existe en effet des algorithmes de simplification automatique sur ce genre de modélisation. Cette modélisation peut également être facilement connectée à un modèle topologique et est ouverte à l'utilisation de la texture. En revanche, elle est très peu compacte puisque la modélisation d'un objet nécessite de décrire un grand nombre de faces. Cependant, ceci est relativement peu pénalisant pour les phases d'affichages, les outils de visualisation dynamique utilisant principalement des ensembles de faces : cette modélisation est donc très bien adaptée à ce mode de visualisation. La représentation par frontière surfacique BREP couvre donc la plupart des besoins en systèmes d'information géographique 3D.

C'est donc tout naturellement que nous nous avons choisi d'intégrer ce modèle géométrique au cœur de notre prototype.

Notons que les modèles utilisés dans un processus de saisie ne seront pas nécessairement du B-Rep. Par exemple, il peut être intéressant d'utiliser les opérateurs booléens CSG, un jeu de primitives prédéfinies ou encore du balayage vertical sur des formes 2D pour réaliser des bâtiments comme celui de la Figure 18.

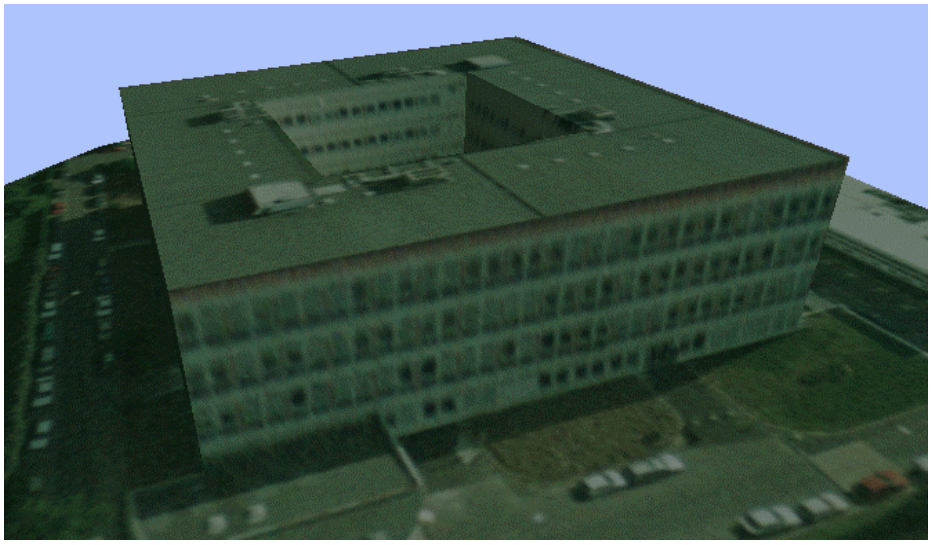


Figure 18 : Exemple d'utilisation de la CSG

2.3.2 Les choix topologiques

2.3.2.1 Nos orientations topologiques

Au vu des nombreux avantages apportés par la topologie (cf. 2.2.3), nous avons choisi d'intégrer une composante topologique à notre modèle conceptuel de données. Dans le cadre de nos travaux, il est cependant important de ne pas alourdir et complexifier nos structures de données par cette entrée de la topologie au cœur de notre prototype. Afin d'éviter que cette dernière n'engendre des complications quant à la gestion de notre futur SIG, nous avons pris soin de définir un modèle topologique en harmonie avec nos besoins, c'est à dire :

- Disposer d'un outil permettant de garantir une cohérence dans la modélisation des objets tridimensionnels ;
- Optimiser l'analyse, la gestion et les requêtes sur les réseaux ;
- L'ajout d'une topologie ne doit pas nuire exagérément aux autres aspects de l'application.

Ce dernier point est crucial, il est impératif de ne pas se perdre dans des difficultés de maintenance de données ou dans des lourdeurs de calculs dues à la mise en place d'une topologie couvrant certes toute situations spatiale aussi exceptionnelle soit elle, mais au détriment d'une modélisation complexe et encombrante.

En effet, lorsqu'on établit une relation topologique entre deux primitives, on établit une relation de dépendance entre ces primitives. Plus le modèle topologique est exhaustif, plus il établit de liens entre les diverses primitives, ce qui a pour effet d'augmenter les dépendances entre les données. L'intégration ou la suppression de donnée devient plus délicate, donc plus coûteuse, puisque le nombre de relations entre objets croît. De même, le stockage devient problématique, notamment sous des systèmes de gestion de base de données relationnels où l'unique moyen de stocker ces relations consiste à créer de nouvelles tables. La taille de la base augmente et les temps d'accès à l'information, qui se fait nécessairement par des jointures de tables, peuvent s'en trouver augmentés.

Il s'agit là d'un problème auquel se sont heurtés des organismes comme la NIMA (National Imagery and Mapping Agency). Après avoir recherché pendant des années une sorte « d'exhaustivité topologique » pour leurs bases de données géographiques, ils sont aujourd'hui contraints de revoir leurs ambitions en simplifiant leurs modèles.

Conscients de ces difficultés, nous avons cherché à limiter l'utilisation de la topologie. Il n'est pas nécessairement utile de décrire tous les éléments d'une scène géographique avec la même précision topologique. Nous estimons que dans un univers géographique le nombre d'objets suffisamment intéressants aux yeux de l'utilisateur pour justifier une description minutieuse est assez restreint. La topologie est par exemple rarement utile pour décrire le relief d'un paysage. Nous préconisons donc de n'employer les primitives topologiques que là où elles sont indispensables.

D'autre part, puisque le défi est de décrire topologiquement deux structures aussi différentes qu'une structure réseau et une structure de type bâtiment 3D, tout en gardant un modèle simple, nous nous sommes résolus à utiliser deux niveaux de modélisation topologique (Figure 19).

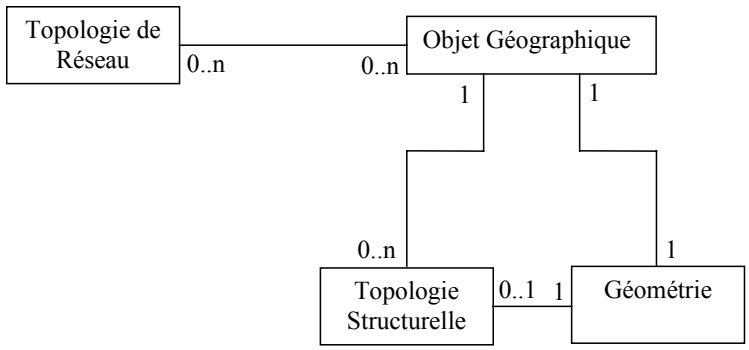


Figure 19 : Modélisation Topologique et Modélisation Géométrique

Tout objet géographique fait l'objet d'une description géométrique, mais au-delà de sa géométrie il peut être topologiquement décrit dans sa structure interne (Topologie Structurelle). Il peut aussi faire partie intégrante d'un réseau et être alors associé à une primitive réseau (Topologie de Réseau). Dans le cas d'un bâtiment, la topologie structurelle permettra par exemple de le décrire par ses murs, son toit, son sol, alors que la topologie de réseau permettra de l'associer éventuellement à un nœud du réseau routier s'il est le point de départ ou le but d'une recherche d'itinéraire. Pour un même objet géographique, nous avons donc trois niveaux de représentation :

- Tout objet géographique de la scène possède obligatoirement sa propre modélisation géométrique 3D. Le modèle géométrique que nous avons choisi est le B-Rep (cf. 2.3.1).

- Un objet géographique peut éventuellement être décrit dans sa topologie par le modèle de topologie structurelle. Nous décrivons ce modèle en 2.3.2.2. Cette modélisation topologique structurelle vient se greffer au-dessus de l'information géométrique.

- Enfin, si cet objet géographique s'insère dans un réseau, il peut être associé à une primitive topologique de réseau. Le modèle topologique de réseau est totalement indépendant du modèle géométrique et du modèle de topologie structurelle. Nous le détaillons en 2.3.2.3.

Dans notre proposition de modélisation, une scène 3D peut être vue comme un ensemble essentiellement décrit par la géométrie avec à certains endroits des îlots d'objets, particulièrement intéressants, également décrits par un modèle topologique (topologie structurelle). Si des relations existent entre plusieurs de ces « îlots d'intérêt », elles peuvent être décrites par notre topologie de réseau.

2.3.2.2 Topologie structurelle

2.3.2.2.1 Le modèle

La topologie structurelle a pour objectif de décrire la structure interne d'un objet géographique et le positionnement relatif des différents éléments de cette structure. Cette Topologie Structurelle est organisée suivant le modèle présenté plus loin sur la Figure 21.

Un objet topologique correspond à la représentation topologique d'un objet géographique. L'objet géographique appartient au monde réel, l'objet topologique n'est qu'une représentation parmi d'autres de cet objet géographique. Dans notre modèle de topologie structurelle, cet objet topologique sera au final décrit par les quatre primitives topologiques classiques que sont le nœud, l'arc, la face et le volume. Les relations entre ces primitives permettront notamment de garantir une cohérence dans la modélisation de l'objet. Puisque ces quatre primitives décrivent la structure topologique de l'objet, nous les appelons « Primitives Structurelles ».

Un objet géographique se compose souvent de plusieurs entités discernables. Un hangar est par exemple bâti autour de trois structures qui sont son toit, son sol et ses murs. Il nous a semblé opportun de pouvoir tenir compte de ces décompositions au niveau de la modélisation topologique. Nous avons donc intégré dans notre modèle une primitive intermédiaire que nous avons appelée « Structure Constitutive ». Cette structure constitutive est une composition de primitives structurelles et d'autres structures constitutives. La décomposition d'un objet en structures constitutives organise la description topologique de celui-ci. Un avantage de cette organisation est qu'elle facilite et accélère l'accès aux primitives structurelles.

La Figure 20 reprend l'exemple de décomposition topologique dans la modélisation d'un hangar.

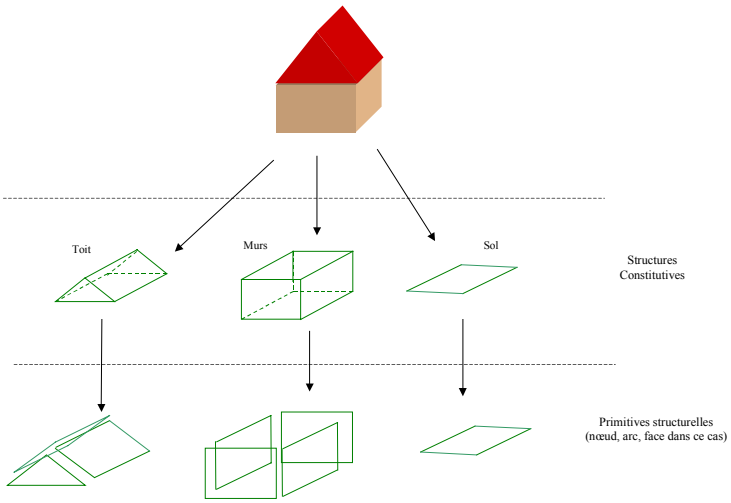


Figure 20 : Décomposition topologique d'un hangar

L'objet Hangar est topologiquement modélisé par trois structures constitutives représentant le toit, les murs et sol du hangar. Ces trois structures constitutives sont ici directement décrites par des primitives structurelles. Une modélisation plus détaillée aurait pu consister à décomposer la structure Mur en trois autres structures constitutives séparant les portes, fenêtres et la partie façade du mur.

Le diagramme UML de la Figure 21 formalise notre modèle topologique.

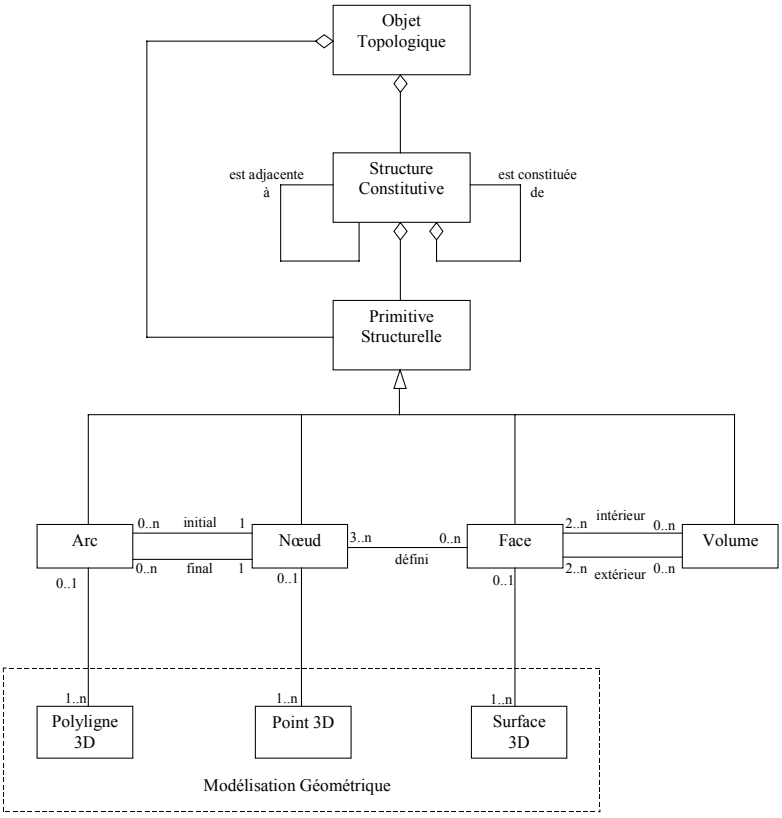


Figure 21 : Modèle de Topologie Structurelle

Ce modèle est dans la lignée des modèles classiques les plus simples (vu en 2.2.3.3.1). Bien que des relations d'adjacence soient possibles entre structures constitutives, l'essentiel de la topologie est porté par les relations entre primitives structurelles. Ces structures primitives font également le lien avec le modèle géométrique.

L'originalité du modèle tient en ce que la face n'est pas ici décrite par un contour d'arc mais directement par un contour de nœuds. Ainsi l'arc n'est plus un intermédiaire nécessaire et indispensable pour décrire un objet par ses faces. En revanche, l'arc reste toujours présent dans le modèle pour pouvoir décrire des objets linéaires.

En matière de description 3D d'objets géographiques, l'arc en tant que primitive intermédiaire entre le nœud et la face n'est que très peu utilisé. En effet, la primitive qui délimite le contour des objets est essentiellement la face et non plus l'arc comme c'est le cas en 2D. Comme la face peut être directement décrite par une liste ordonnée de nœuds, l'arc devient secondaire (sauf pour décrire des objets linéaires). En supprimant cet intermédiaire arc entre la face et le nœud, nous simplifions la description des objets et, de ce fait, nous limitons les coûts de stockage. Si nécessaire, les arcs peuvent être recalculés puisqu'ils ne sont rien d'autre que des couples de nœuds.

L'objectif premier de notre modèle de topologie structurelle est d'assurer une meilleure cohérence dans la description des objets 3D. La suppression de l'intermédiaire arc ne nuit en rien à cet objectif puisque le maintien de cette cohérence reste assuré par les primitives nœuds et faces. D'autre part, cette modélisation n'a pas pour vocation de pouvoir décrire l'ensemble des situations spatiales imaginables dans un espace 3D quelconque, elle se contente de proposer une solution pour couvrir les principales caractéristiques topologiques nécessaires à la description 3D d'un univers géographique. Ainsi certains cas particuliers ou certaines situations marginales (du type bouteille de Klein, ceinture de Moebius, anses etc …) sont écartés, afin de traiter plus efficacement les situations classiques. Le modèle que nous proposons est adapté au besoin d'un utilisateur de SIG, mais ne prétend en aucune manière convenir par extension à la description de tout espace topologique 3D.

2.3.2.2.2 Comparaison avec d'autres modèles

L'objectif principal du modèle que nous venons de présenter n'est pas d'optimiser la navigation de proche en proche entre primitives topologiques, pourtant celle-ci est bien supportée par notre modélisation topologique structurelle. Afin d'évaluer les performances de notre modèle sur ce point, nous avons comparé nos temps d'accès théoriques aux primitives topologiques avec ceux de deux autres modèles. Ces deux autres modèles sont le modèle GEO 3D d'Arnaud De La Losa qui représente le courant des extensions de cartes topologiques [DE LA LOSA 2000] et le modèle VPF 3D de Kevin Trott qui représente le courant des modèles topologiques classiques.

Les temps d'accès théoriques ont été établis en dénombrant les accessions aux primitives nécessaires pour obtenir le résultat recherché. Supposons par exemple que nous cherchions à déterminer le temps d'accès théorique pour récupérer, à partir d'un nœud donné, l'ensemble des nœuds reliés par un arc à notre nœud initial. Dans le cas de notre proposition de modèle, cette recherche passe dans un premier temps par la récupération de l'ensemble des arcs entourant le nœud initial, puis, pour chacun de ces arcs, nous nous intéressons au nœud complémentaire de notre nœud initial.

Le résultat est exprimé à partir d'un certain nombre de constantes dont voici les définitions :

k : temps moyen pour accéder à une information à partir de son identifiant.

An : nombre moyen d'arcs autour d'un nœud

Af : nombre moyen d'arcs autour d'une face

Nf : nombre moyen de nœuds autour d'une face

Fn : nombre moyen de faces autour d'un nœud

Fa : nombre moyen de faces autour d'un arc

Pour notre exemple, le temps moyen et théorique d'accès à notre liste de nœud est 3k*An et non pas 2k*An. En effet bien que seul le nœud complémentaire soit l'objet de notre convoitise, ce nœud ne peut pas être directement atteint, il nous faut récupérer les 2 nœuds extrémités de l'arc et ne garder que celui qui n'est pas notre nœud initial. Pour cet exemple le détail du calcul est donc le suivant :

$$T_{\text{nœud-nœud}} = k*An + An*2k = 3k*An$$

Chaque ligne du Tableau 5 correspond à une requête désignée par une abréviation du type « PrimitiveDepart-PrimitiveResultat » qu'on peut traduire par « A partir d'une PrimitiveDepart, retrouver l'ensemble des PrimitivesResultats liées à la PrimitiveDépart ».

Les modèles GEO 3D et VPF+ sont présentés en Annexe à la fin du mémoire.

	GEO 3D	VPF 3D	SIG 3D
Nœud-Nœud	4k*An	**3k*An**	**3k*An**
Nœud-Arc	2k*An	**k*An**	**k*An**
Nœud-Face	k*An*(1+3Fa)	k*An*(1+Fa)	**k*Fn**
Arc-Nœud	4k	**2k**	**2k**
Arc-Arc	4k*(1+An)	**2k(1+An)**	**2k*(1+An)**
Arc-Face	k*(1+3Fa)	**k*Fa**	2k*(1+Fn)
Face-Noeud	k*(1+3Af)	k*(1+2Af)	**Af*k**
Face-Arc	k*(1+3Af)	**k*(1+Af)**	k*(An+Af)
Face-Face	k*(1+Af*(3Fa-1))	k*(1+Af*(1+Fa))	Af*k*(1+Fn)

Tableau 5 : Comparaison GEO 3D, VPF 3D et SIG 3D

Il apparaît clairement que le modèle GEO 3D n'offre pas les meilleurs temps d'accès aux primitives. Ceci s'explique par la philosophie du modèle qui n'est pas d'optimiser les temps d'accès mais plutôt de pouvoir modéliser absolument tout type de relation topologique 3D entre les 4 primitives de base que sont le Nœud, l'Arc, la Face et le Volume. Notons à ce sujet que le modèle proposé par Arnaud De La Losa [DE LA LOSA 2000] valide l'ensemble des relations topologiques définies par le modèle aux 9 intersections d'Egenhofer [EGENHOFER 1990].

En revanche, les deux autres modèles offrent des temps d'accès bien meilleurs et souvent assez comparables voire identiques. Les distinctions ne se font que sur des requêtes impliquant la primitive Face et de manière générale le modèle SIG 3D favorise les relations Face-Nœud alors que le modèle VPF 3D préfère les relations Face-Arc. La comparaison des performances pour les relations Face-Face est un peu moins triviale que pour les autres. Si on développe le résultat on obtient (en simplifiant par k):

Pour VPF 3D : 1 + Af + Af*Fa

Pour SIG 3D : Af + Af*Fn

Rappelons que Fa représente le nombre de faces autour d'un arc alors que Fn représente le nombre de faces autour d'un nœud. Ici les arcs sont considérés amputés de leurs deux extrémités. Comparer le temps d'accès pour retrouver toutes les faces entourant une face revient donc à comparer le nombre de faces autour d'un arc et le nombre de faces autour d'un nœud. Un nœud est entouré au minimum du nombre de faces entourant l'arc (ou les arcs) qui le supporte, le nombre moyen de faces autour d'un nœud est donc plus élevé que le nombre moyen de faces autour d'un arc d'où $Fn \geq Fa$. En conclusion, le modèle VPF 3D est plus efficace pour retrouver les relations entre faces. Ceci est peu handicapant pour notre prototype, la différence de temps d'accès restant relativement faible.

Ce modèle de topologie structurelle est particulièrement bien adapté pour modéliser la topologie d'un objet géographique 3D. Il est toutefois trop lourd pour gérer les applications réseaux d'un SIG, aussi proposons nous un second modèle topologique dédié à ce style d'applications.

2.3.2.3 Topologie de réseau

Les besoins topologiques en matière de réseau diffèrent fondamentalement de ceux utiles à la description d'un objet 3D ou d'un ensemble d'objets 3D. L'utilisation de la topologie est avantageuse pour décrire un objet 3D, elle garantit, entre autre, une meilleure cohérence des données. En revanche, la topologie devient indispensable lorsqu'il s'agit d'assurer correctement la gestion d'un réseau. Un simple modèle géométrique aura beaucoup plus de mal à modéliser les phénomènes de propagation et de diffusion à l'intérieur d'un réseau. Les apports de la topologie dans l'analyse des réseaux sont connus depuis longue date dans le monde des SIG et la plupart d'entre eux offrent des solutions très bien adaptées en la matière. Nous nous sommes donc contenté de reprendre le schéma général d'un modèle de réseau topologique classique pour l'intégrer dans notre modèle conceptuel général (Figure 22). Ce modèle est beaucoup plus simple que le modèle de topologie structurelle, puisque seules les primitives arcs et nœuds sont utiles à la description d'un réseau.

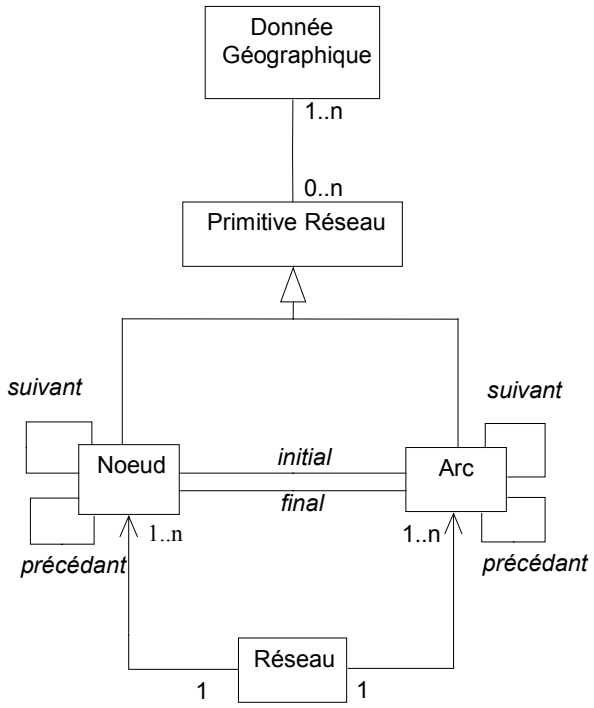


Figure 22 : Modèle de Topologie Réseau

Ce modèle s'appuie sur un graphe planaire, c'est à dire un graphe ne comportant pas d'intersections entre arcs en dehors des nœuds. Il est particulièrement bien adapté au parcours de graphes et donc aux applications des sciences de l'information géographique portant sur des recherches d'itinéraires ou sur des extractions de sous-réseaux (recherche de réseaux amonts ou avals par exemple).

Une donnée géographique peut être associée à une ou plusieurs primitives topologiques. Chaque primitive appartient à un réseau unique. Si une donnée géographique doit être intégrée dans deux réseaux différents, elle devra obligatoirement être associée à deux primitives différentes. Ce sera, par exemple, le cas d'un pont surplombant un réseau autoroutier et sur lequel passe une route départementale. Ce pont sera associé d'une part à un nœud du réseau « autoroutes » et d'autre par à un nœud du réseau « départementales » Une primitive topologique de réseau ne peut pas appartenir à plusieurs réseaux. En revanche une primitive nœud peut être associée à plusieurs objets géographiques. Dans une application du type recherche d'itinéraire tous les bâtiments à proximité d'un nœud du réseau routier pourront être associés à celui-ci.

2.3.2.4 Synthèse sur la topologie

En guise de synthèse nous présentons à titre d'exemple en Figure 23 une scène 3D simplifiée, modélisée selon nos choix topologiques. Cette scène est constituée d'un hangar, d'un bloc d'immeuble, de routes les reliant et formant par endroit des carrefours.

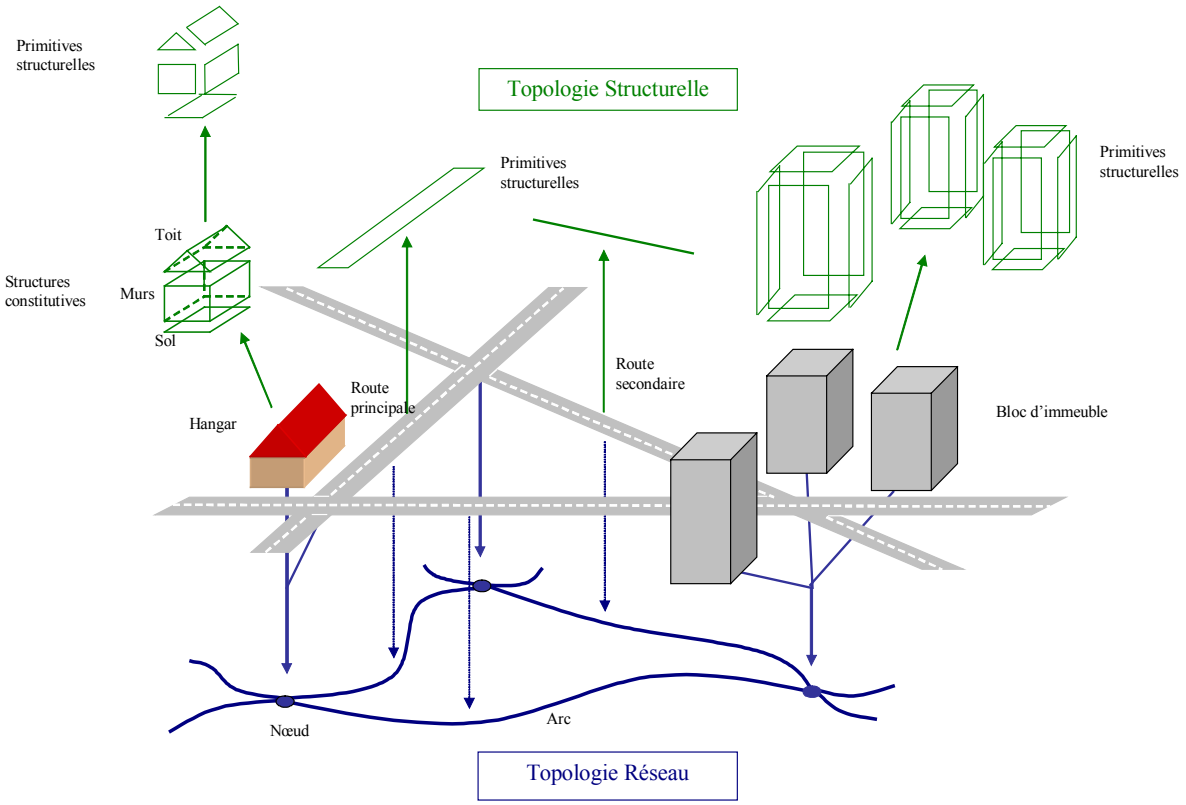


Figure 23 : Exemple de modélisation

Topologie Structurelle

Commençons par la description des objets dans leurs structures internes, c'est-à-dire suivant le modèle de topologie structurelle.

Le hangar : supposons que le hangar ait une importance particulière dans la scène, à ce titre il bénéficie d'une représentation soignée. Une modélisation possible de l'objet topologique est donc de le décomposer en trois structures constitutives : Toit, Mur et Sol. Chacune de ces structures constitutives est elle-même composée de faces et de nœuds, c'est à dire de primitives structurelles, briques de base de la modélisation.

Le bloc d'immeubles : supposons que l'importance du bloc d'immeuble soit moindre que celle du hangar. La représentation des immeubles sera donc moins détaillée. Une modélisation envisageable est de considérer le bloc formé d'immeubles comme un seul objet topologique et chaque immeuble pris séparément joue le rôle de structure constitutive.

Les routes : dans notre exemple, les routes sont elles aussi décrites par une topologie structurelle. Simple à modéliser, elles sont représentées directement par des primitives structurelles sans utiliser de composition en structures constitutives. Les carrefours ne font pas ici l'objet d'une description propre, ils sont simplement la conséquence du croisement de deux routes. Les routes principales sont, par exemple, modélisées par des faces alors que les routes secondaires ne sont modélisées que par des arcs.

Topologie Réseau

Les objets que nous venons de décrire sont, dans notre exemple, reliés entre eux par un réseau routier. Le réseau routier est modélisé par une topologie de réseau constituée d'un ensemble d'arcs et de nœuds interconnectés. Les objets de type route sont associés aux arcs du réseau. Chaque intersection de route, donc chaque carrefour, est associé à un nœud du réseau. Le réseau routier constitue un graphe sur lequel vont pouvoir s'appliquer des algorithmes de recherche de trajectoire. Pour que le hangar et les immeubles puissent faire office de point de départ et de point d'arrivée dans notre recherche de trajectoire, on associe chacune de ces primitives à un nœud du réseau. L'association se fait ici par critère de proximité, chaque bâtiment est associé au nœud représentant le carrefour le plus proche.
Dans cet exemple nous avons proposé une modélisation des objets composant ce que l'on appelle le sur-sol d'une scène géographique, c'est-à-dire principalement les bâtiments, la végétation et l'ensemble des infrastructures réalisées par l'homme. Pour que la description de la scène soit complète, il convient de modéliser le terrain sur lequel repose ces objets du sur-sol.

2.3.3 Modélisation du terrain

2.3.3.1 Définition des Modèles Numériques de Terrain

La modélisation du terrain dans un système d'information géographique 3D est une problématique à part entière. Un modèle numérique de terrain (MNT) est une représentation numérique simplifiée de la surface d'un territoire, en coordonnées altimétriques (le plus souvent exprimées en mètres par rapport au niveau de la mer) et planimétriques, calées dans un repère géographique.

Il existe, parmi une grande variété de représentations possibles d'une surface, deux modes largement utilisés pour les MNT qui s'apparentent aux deux modes de représentation de l'information géographique plane : le mode vecteur (polygones) ou le mode raster (pixels). Dans le cas particulier du

relief, les polygones utilisés sont des triangles (le polygone le plus simple pour représenter un élément de surface orienté dans l'espace). Les caractéristiques de ces deux types de MNT ont des incidences sur leurs méthodes de production ainsi que sur les contraintes techniques de leurs utilisations.

Les MNT Raster

Un MNT sous forme raster est aussi appelé matrice d'altitudes ou grille régulière. Il s'agit d'un ensemble de valeurs numériques représentant des altitudes, régulièrement espacées et ordonnées selon un balayage du terrain (par exemple d'ouest en est et du nord au sud). Chaque altitude ainsi positionnée correspond à ce qui doit être alors considéré comme une altitude moyenne d'un élément de surface du terrain.

Cette distribution régulière de points définit un maillage de la surface du terrain, les dimensions de la maille (de fait, rectangulaire ou carrée) définissant ce qu'on appelle la résolution spatiale planimétrique du MNT. Chaque point est au centre d'une maille. Plus l'espacement des points est serré, plus la résolution est grande, plus le MNT est fin et riche en détails topographiques.

Le MNT Vecteur

La seule alternative au mode raster est d'utiliser un pavage de la surface du terrain à l'aide de triangles. On parle simplement de triangulation. Cette méthode est utilisée pour des MNT depuis les années 70, c'est à dire dès l'arrivée des premiers systèmes d'information géographique vectoriels. De tels MNT ont pour caractéristique principale de s'appuyer sur un semis de points de mesure le plus souvent disposés de façon irrégulière et dont la densité peut augmenter en fonction de la complexité du relief et de la précision recherchée.

Chaque point est relié à deux voisins pour former un réseau de triangles. Ce réseau ne doit laisser apparaître aucun " trou " et est tel qu'aucun des triangles ne se superpose, même partiellement, à un autre. On parle alors de « TIN » de l'anglais Triangular Irregular Network.

2.3.3.2 Le Format MNT utilisé

La représentation sous forme d'un réseau irrégulier de triangles est bien adaptée au type de modélisation pour lequel nous avons opté puisqu'une modélisation de type B-Rep est elle-même une modélisation par facette. Les triangles composant le MNT sont autant de facettes pour notre modèle de données. Comme nous le verrons plus tard, cette décomposition en facette, est particulièrement bien adaptée à nos problèmes d'intervisibilité et nous permet dans ce cas de traiter le MNT de la même manière qu'un quelconque objet du sur-sol. Le format TIN possède en outre l'avantage d'être efficacement traité dans les processus de visualisation et, plus encore, si on lui associe des niveaux de détails.

2.4 Synthèse et conclusion sur la modélisation

Où le point est fait sur nos orientations en terme de modélisation.

Le modèle de donnée est au cœur d'un système d'information géographique. Il constitue la structure sur laquelle vont être appliqués les algorithmes de traitement, d'analyse ou de visualisation de données. Nous ne pensons pas qu'il puisse exister un modèle de donnée universel parfaitement adapté à toutes les applications envisageables autour d'un SIG. Choisir un modèle, c'est trouver un compromis sur des critères d'efficacité, d'exhaustivité, de faisabilité ou de performance.

La modélisation que nous avons retenue tient compte de la diversité des fonctionnalités que notre prototype aura à mettre en œuvre. Notre objectif est de modéliser trois types de données :

- Les formes 3D des objets du sursol ;
- Les réseaux ;
- Le terrain.

Pour décrire la forme 3D des objets, nous avons adopté une modélisation géométrique de type B-Rep (représentation de la frontière des objets). Ce modèle de représentation surfacique permet de modéliser de manière simple des formes 3D aussi complexes soient-elles. Il est efficacement pris en charge par la majorité des outils de visualisation et peut facilement être associé à une modélisation topologique. Afin de supporter une complexité croissante des représentations d'objet 3D et pour assurer au mieux des objectifs de cohérence nous avons effectivement décidé de mettre en place une modélisation topologique structurelle au dessus du modèle géométrique. Bien qu'inspirée par des modèles classiques (tels que les modèle VPF+ de Kevin Trott ou le modèle FDS de Martien Molenaar), notre proposition de topologie structurelle privilégie les relations entre la face et le nœud. Nous considérons en effet que dans une modélisation 3D l'importance de l'arc est diminuée, d'autant plus lorsque la topologie est d'abord utilisée pour garantir la cohérence du modèle.

La relative complexité de notre modèle de topologie structurelle n'est pas adaptée à la gestion des réseaux. Aussi avons-nous choisi de décrire ces derniers par un modèle topologique plus simple dit de réseau. Ce modèle, comparable à ceux classiquement utilisés dans les SIG 2D, s'appuie sur une représentation de type graphe planaire. Il sera notamment utilisé pour traiter les calculs de trajectoire, mais aussi pour étudier les phénomènes de propagation (réseau d'eau, réseau fluvial, réseau électrique, etc.)

Enfin, en ce qui concerne la modélisation du terrain, nous avons choisi d'utiliser un maillage de triangles irréguliers (TIN). Sous cette forme, le sol pourra être facilement incorporé aux autres objets de la scène pour être pris en compte dans des processus tels que les tests d'intervisibilité.

3 ANALYSE ET EXPLOITATION

3.1 Introduction

Là où nous présentons les fonctionnalités d'un système d'information géographique 3D : requêtes, calcul d'intervisibilité et de trajectoire.

Nous avons consacré le premier chapitre de notre mémoire au squelette d'un système d'information géographique 3D, c'est à dire sa structure de donnée, sans véritablement mettre l'accent sur la finalité d'un tel système. Le lecteur familiarisé à l'utilisation de SIG 2D classiques connaît le potentiel d'un tel système, et par expérience a-t-il peut être déjà son idée sur les nouvelles fonctionnalités que peut offrir un SIG traitant de vraie 3D, voire même les façons d'aborder le problème. Nous verrons donc dans cette partie quelques outils de requête et d'analyse spatiale permettant une exploitation avancée d'une modélisation tridimensionnelle. Evidemment, parmi les fonctionnalités présentes dans notre prototype, nous retrouvons quelques grands standards implémentés dans les SIG 2D conventionnels depuis longue date. Il s'agit là principalement de requêtes sémantiques, géométriques, géographiques ou topologiques qui nous permettent, par exemple, d'obtenir les noms de tous les propriétaires des terrains de plus de 10 hectares situés en périphérie d'une zone industrielle donnée. Nous aborderons ici quelques principes de cette analyse spatiale que l'on peut qualifier de classique sans pour autant en faire l'essentiel de notre propos.

Le calcul d'intervisibilité et la recherche de trajectoire sont en revanche les deux grands thèmes sur lesquels nous allons nous attarder. Nous ne prétendons pas qu'il s'agit là de deux aspects totalement novateurs dans le domaine des SIG, mais nous allons les aborder sous le jour nouveau de la vraie 3D géographique. En matière d'intervisibilité, nous avons introduit des techniques empruntées au monde de la synthèse d'image puisque ce sont des lancers de rayons qui statueront de manière discrète sur la visibilité d'une forme 3D complexe. Une deuxième technique procédant à des calculs d'intersections entre polyèdres permet d'obtenir un résultat exact au détriment du temps de calcul. Ces techniques s'appliquent aussi bien sur les éléments du sur-sol, bâtiment ou autre ouvrage d'art, que sur le MNT caractérisant le relief de la zone étudiée. Les calculs de trajectoires sont quant à eux fortement paramétrables puisqu'ils font intervenir des variables représentatives du relief, de la nature du sol, de la nature du véhicule ou du gabarit du véhicule. Ces trajectoires sont calculées aussi bien pour des engins terrestres que pour différents types d'aéronefs. Nous distinguons les calculs d'itinéraires classiques s'appuyant sur un réseau préétabli auquel on applique des algorithmes de type Dijkstra et les calculs de trajectoires dites « libres » pour lesquelles il est nécessaire au préalable de reconstruire une matrice de coût de déplacement sur laquelle pourront être appliqués les algorithmes *ad hoc*.

3.2 Requêtes communes

Dans lequel on part à la recherche des informations géométriques et sémantiques et où l'on décortique les index spatiaux qui optimiseront nos requêtes.

De manière générale, l'information est stockée par le SIG sous une forme non intelligible pour l'utilisateur. Les outils d'analyse et d'exploitation sont les intermédiaires entre l'information stockée sous sa forme brute et l'utilisateur. Ce n'est qu'une fois traduite que l'information peut ensuite être utilisée pour des prises de décisions futures.

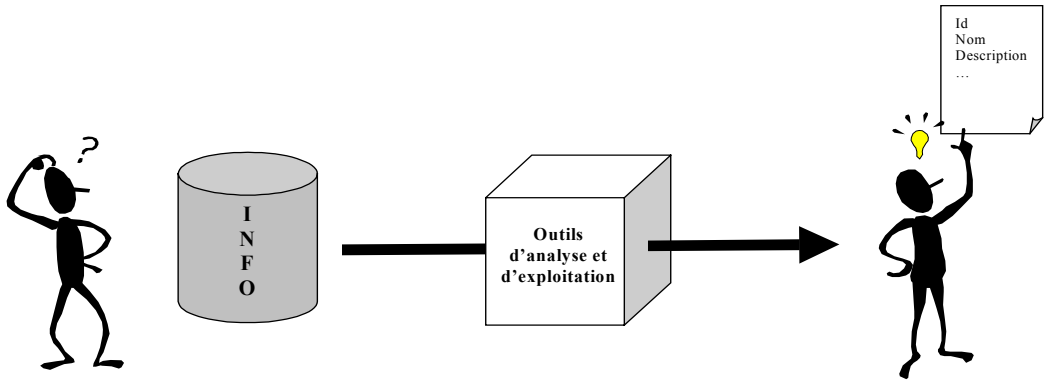


Figure 24 : Outils d'analyses et d'exploitations

Un SIG véhicule trois niveaux d'information sur les données lorsque celles-ci sont prisent individuellement :

- Un niveau d'information sémantique.
- Un niveau d'information géométrique renseignant sur la forme de la donnée.
- Un niveau d'information géographique renseignant sur le positionnement de la donnée.

En outre, un SIG va bien au-delà de la simple accumulation de données individuelles, il tisse un réseau de relations diverses entre ces données qui peut être vu comme un quatrième niveau d'information. Il doit permettre d'accéder à chacun de ces niveaux d'information et doit être capable de modifier ces informations lorsque le demande l'utilisateur autorisé.

Ces notions seront abordées dans la suite de notre propos. Nous commencerons par faire un rappel sur les mécanismes de consultations de l'information sémantique dans une base de donnée à partir de langage de requête. Nous nous intéresserons ensuite aux processus mis en œuvre pour réaliser une analyse géométrique pertinente. Nous terminerons enfin par les aspects liés à la localisation géographique et, plus précisément, à l'optimisation de cette localisation via l'utilisation d'index spatiaux.

3.2.1 Restitution et modification de l'information sémantique

Notre propos n'est pas d'étudier les systèmes de gestion de base de données, nous pensons que les fonctionnalités d'un système d'information géographique tirent avantage à être indépendantes du système de stockage. Ce point de vue est discutable, certains argumenteront que pour optimiser les performances d'un SIG, l'architecture de celui-ci doit être pensée pour un système de stockage particulier. Ce débat récurrent dans la mise en œuvre d'un SIG est celui de l'opposition polyvalence-performance : or, sur cette question du stockage, nous privilégions la polyvalence.

Dans ce chapitre nous proposons plutôt d'aborder la restitution des valeurs attributaires via le recueil, direct ou indirect, de l'information désirée au sein du système de stockage. Cet accès à la base de donnée se fait par des langages et protocoles spécifiques et dépendants de la logique objet ou relationnelle de la base.

3.2.1.1 Rappels sur les langages de requête

Un langage de requête ne se limite pas à la simple recherche des données. Il permet également de créer, modifier ou organiser une base de données. Pour assurer toutes ces fonctionnalités, nous distinguons trois niveaux de langage :

- Le **langage de définition des données** (LDD) qui prend en charge l'organisation des données.
- Le **langage de description physique** (LDP) qui spécifie l'organisation physique des données sur les différents supports de stockage.
- Le **langage de manipulation de données** (LMD) permettant de créer, modifier et interroger les bases de données.

Dans les prochains paragraphes, nous présentons succinctement le LDD et le LMD pour le langage SQL (Standard Query Language) qui permet d'attaquer les bases relationnelles, et pour le langage OQL (Object Query Language) qui attaque les bases objets. Le LDP est davantage un langage d'optimisation de la base de données, il ne sera pas traité dans notre optique d'exploitation et d'analyse.

3.2.1.1.1 Standard Query Language

3.2.1.1.1.1 Historique

Le langage SQL a été créé en 1981 par IBM. Il s'agit du langage standard pour la gestion des données dans un Système de Gestion de Base de Données Relationnel (SGBDR). Depuis sa création, le langage SQL a subi plusieurs mutations :

- Sa première version standardisée est SQL 86 ou SQL-1 qui définit le cœur du langage.
- Cette version a été améliorée par SQL 89 qui ajoute des contraintes d'intégrité sur les données (valeurs par défaut, contraintes de domaines ou contraintes référentielles).
- SQL 92 (ou SQL-2) correspond aux normes "ANSI X3.135-1992" et "ISO/IEC 9075:1992" et ajoute de nouvelles fonctionnalités.

- SQL 98 (ou SQL-3) ajoute l'accès à un modèle objet.

3.2.1.1.1.2 Langage de Définition des Données

La commande principale pour la définition des données est la commande "CREATE TABLE". Elle permet de définir une table dans laquelle seront insérées les données. Suivant les éditeurs de SGBD, il est possible de définir plus ou moins de paramètres liés au stockage physique de la table (taille initiale, extension de la table, création d'index…) ou des contraintes d'intégrité.

La création d'une table se fait donc par la commande :

```
CREATE TABLE clients
(
     numClient    INTEGER,
     nom          CHAR(30),
     prenom       CHAR(30)
);
```

Sous ORACLE, la création de la table serait de la forme :

```
CREATE TABLE clients
(
     numClient    INTEGER NOT NULL,
     nom          VARCHAR2(30) NOT NULL,
     prenom             VARCHAR2(30) NOT NULL,
CONSTRAINT pk_numClient PRIMARY KEY(numClient)
) TABLESPACE espace_commandes;
```

Cette commande ajoute des contraintes d'intégrité (en rouge) et une contrainte de stockage physique (en bleu). La contrainte "PRIMARY KEY" permet d'indiquer que la colonne "numClient" fait office de clé primaire. C'est-à-dire qu'elle permet d'identifier une ligne et une seule.

D'autres commandes permettent également de modifier la définition des données (modification d'une table par ajout de colonne, suppression d'une table,…) ou encore la création d'index.

3.2.1.1.1.3 Langage de Manipulation des Données

Il existe quatre types de commande permettant de créer, modifier, supprimer ou lire des données.

Création d'une ligne

L'ajout d'une ligne dans la table "clients" se fait en utilisant la commande "INSERT".

Par exemple :

```
INSERT INTO clients VALUES (1, 'Lagaffe', 'Achile');
```

Si, lors de l'insertion, une contrainte d'intégrité est violée (par exemple un nom vide ou un numéro de client existant déjà), la ligne n'est pas insérée.

Modification d'une ou plusieurs lignes

La modification d'une ou plusieurs lignes se fait en utilisant la commande "UPDATE table SET attr = val WHERE condition".

Par exemple :

```
UPDATE clients SET prenom = 'Gaston' WHERE nom = 'Lagaffe';
```

Cette commande permet de modifier le prénom de Lagaffe.

Suppression d'une ou plusieurs lignes

La suppression d'une ou plusieurs lignes se fait en utilisant la commande "DELETE FROM table WHERE condition".

Par exemple :

```
DELETE FROM clients WHERE nom LIKE 'L%';
```

La commande précédente permet de supprimer tous les clients dont le nom commence par 'L'.

Lecture de données

Cette opération est sans doute la plus riche du langage SQL. Elle se construit autour d'une commande de type "SELECT attr1, attr2,… FROM table WHERE condition". Elle permet de réaliser tous les opérateurs de l'algèbre relationnel (la projection, la sélection, la jointure, l'union, l'intersection, et le produit cartésien).
Les requêtes de lecture de données peuvent être très complexes et comporter plusieurs niveaux d'imbrication. Nous nous limitons ici à quelques exemples simples (Figure 25). La première commande est une projection de la table "commandes", la deuxième est une restriction de la table clients à tous les clients dont le nom commence par 'C', la troisième est une jointure entre les tables clients et relations, la dernière est une sélection imbriquée permettant de déterminer les numéros des commandes passées par le Profeseur Tournesol.

```
SELECT libelle, montant FROM commandes;

SELECT * FROM clients WHERE nom LIKE 'C%';

SELECT c.numClient, r.numCommande FROM clients
c, relations r WHERE c.numClient = r.numClient;

SELECT numCommande FROM relations
   WHERE numClient IN (
      SELECT numClient FROM clients
         WHERE nom = 'Tournesol');
```

Figure 25 : Exemples d'utilisation de la commande SELECT

3.2.1.1.2 Object Query Language

3.2.1.1.2.1 Présentation

OQL (Object Query Language) est un langage orienté objet de requête. Sa syntaxe, pour certaines requêtes, est proche de SQL. OQL est le langage de requête des Systèmes de Gestion de Base de Données Objets (SGBDO) défini par l'ODMG-93.

Dans la suite de ce chapitre, on utilise l'exemple de la gestion des clients et des commandes et on l'adapte à un SGBDO.

3.2.1.1.2.2 Langage de Définition des Données

L'organisation des données dans un SGBDO est différente de celle utilisée dans un SGBDR. Dans un SGBDO, il s'agit de stocker des objets (instances de classe) sous forme d'un graphe d'objets. Ce graphe possède une ou plusieurs racines (appelées racines de persistance).

Création du schéma de la base

Pour pouvoir stocker des objets dans un SGBDO, il est nécessaire de définir le schéma des données, c'est-à-dire décrire les classes manipulées dans le SGBD.

```
class Commande public type
     tuple(
           numCommande: integer,
           libelle: string,
           montant: real,
           dateLiv: string
           )
end

class Client public type
      tuple(
           numClient: integer,
      nom: string,
      prenom: string
      commandes: List(Commande)
      )
end
```

Figure 26 : Définition des classes Commande et Client

La Figure 26 présente la définition des classes "Commande" et "Client". Ces classes possèdent des attributs correspondants aux colonnes du modèle relationnel. Dans un SGBDO, il n'est plus nécessaire d'avoir une table de relation. Pour gérer la relation client-commande, la classe "Client" comporte une liste de commandes.

Création des racines de persistance

Une fois le schéma de la base de données défini, on peut créer les racines de persistance. La solution la plus naturelle dans notre exemple consiste à créer deux racines de persistance correspondant à :

- une liste de clients ;

```
name clients: List(Client);
```

- une liste de commandes.

```
name commandes: List(Commande);
```

A ce stade, la base de données est prête à recevoir des données.

3.2.1.1.2.3 Langage de Manipulation des Données

Le LMD d'OQL est un sur-ensemble de SQL. Il ajoute toutes les possibilités objets comme l'héritage ou le polymorphisme. Il permet également d'appeler les méthodes des objets définis et d'utiliser le résultat de ces appels dans la requête.

Les SGBDO doivent être vus comme une mémoire persistante. Ainsi, la création, la modification et la suppression peuvent se faire par le langage OQL mais ce n'est pas le mode privilégié de modification.

Une application utilisant un SGBDR va écrire des requêtes SQL pour créer, modifier, supprimer ou lire des données. Une application utilisant un SGBDO va directement créer des données dans son espace d'adressage normal et cette donnée sera automatiquement ajoutée à la base de données objet. Il en va de même pour les modifications et les suppressions.

OQL est donc surtout utilisé pour des requêtes de lecture en utilisant la forme "SELECT … FROM collection WHERE …" standard. Au lieu de travailler sur une table, cette requête travaille sur une collection de données. Ainsi, les requêtes SQL décrites Figure 25 restent-elles valables en OQL.

3.2.1.2 Requête objet sur SGBDRE

3.2.1.2.1 Problématique

Aujourd'hui, nous vivons une situation assez paradoxale puisque nous assistons à une croissance presque exponentielle du nombre d'applications empruntant les technologies objets tout en restant fortement ancrées dans le monde du relationnel en matière de stockage de données. Le défi est donc de pouvoir interroger « naturellement » une base de données relationnelle (éventuellement étendue) à partir d'un langage de programmation objet. Ceci se fait généralement par le biais d'interfaces de programmation qui gèrent le passage du modèle objet au modèle relationnel et vice versa.

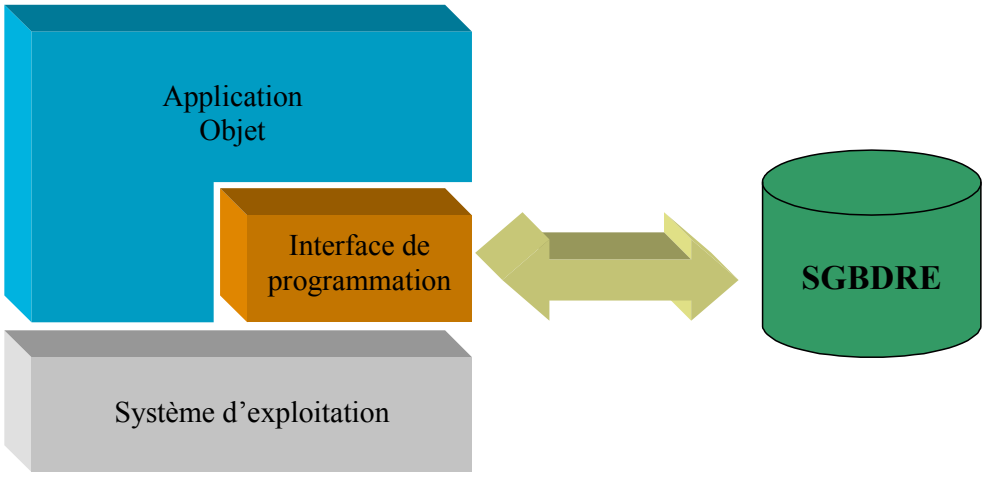


Figure 27 : D'une application objet vers un SGBDRE

Parmi ces interfaces, certaines sont génériques et ne dépendent pas du SGBD qu'elles attaquent. Les interfaces génériques les plus connues sont ODBC de Microsoft et JDBC pour les fonctions écrites en Java. Cependant, à l'occasion de la mise en place de notre prototype, ces interfaces génériques se sont avérées bien trop limitées puisqu'elles ne permettent pas l'accès aux fonctionnalités liées aux SGBD relationnels étendus et notamment à leurs fonctions spatiales.

Oracle, Postgres et la majorité des éditeurs de SGBD fournissent une interface « bas-niveau » permettant d'accéder à la totalité des fonctions du SGBD. Cette interface s'appelle par exemple OCI (Oracle Call Interface) pour Oracle ou libPq pour Postgres. Elles consistent en une collection de fonctions C offrant la possibilité de définir des requêtes ou de lire des données. Malheureusement, ces interfaces « bas-niveau » sont « propriétaires » et impliquent donc une grande dépendance vis à vis du SGBD utilisé.

Partant du principe qu'il est regrettable de perdre les fonctionnalités spatiales d'un SGBDRE et qu'il est tout aussi regrettable d'être dépendant d'un SGBD particulier, le département géomatique de la société Matra Systèmes et Information, aujourd'hui renommée EADS S&DE, a développé en interne le composant IDA (Immediate Data Acces). Cette interface de programmation affranchit le développeur des contraintes propriétaires imposées par les modules des grands éditeurs de bases de données, tout en permettant de pleinement exploiter l'ensemble des fonctionnalités d'un SGBDRE.

3.2.1.2.2 Quelques mots sur l'interface IDA

L'architecture générale du composant logiciel IDA se compose de neuf packages présentés Figure 28, les flèches symbolisent ici les dépendances entre packages.

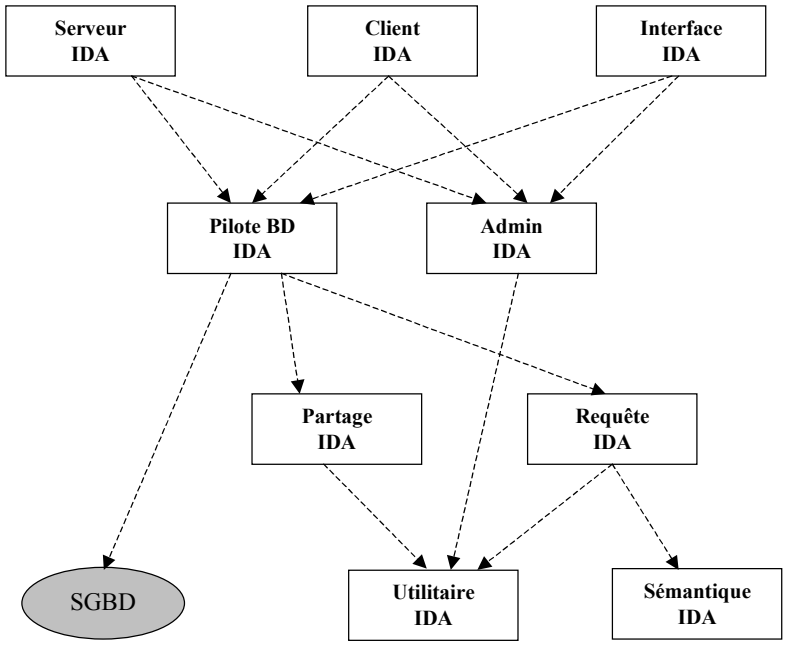


Figure 28 : Architecture générale de l'interface de programmation IDA

Chaque package joue un rôle bien précis au sein du composant IDA :

- Le package « Utilitaire » contient les classes bas niveaux utilisées dans les autres packages.
- Le package « Sémantique » contient le méta-modèle conceptuel de données utilisé dans IDA dont le rôle est de faire évoluer le modèle conceptuel de données. Ainsi, il gère l'ajout de nouveaux types ou la modification d'un type existant (ajout de nouveaux attributs).
- Le package « Requête » permet de décrire une requête sur des données indépendamment de tout langage de requête. Il offre la possibilité de combiner différentes conditions sur des attributs.
- Le package « Partage » contient les classes nécessaires à la gestion de la concurrence d'accès aux données. Il contrôle les notifications de changement et le verrouillage des données.
- Le package « Pilote » contient le pilote d'accès aux données, un pilote pour stocker les requêtes et un pilote pour faire évoluer le modèle conceptuel de données (MCD). Il contient également un convertisseur pour écrire des requêtes SQL. Enfin, il gère les relations entre plusieurs objets.

- Le package « Admin » supervise les sessions de travail et les transactions.

- Le package « Interface » contient les classes d'interface du composant IDA. Les différentes implémentations de ces interfaces délèguent des opérations aux autres classes IDA.

3.2.1.3 Conclusion

La quantité d'information traitée par un SIG rend indispensable l'utilisation de base de données. Les bases de données objets n'étant pas aujourd'hui suffisamment matures, l'essentiel de l'information est généralement stockée dans des bases relationnelles. Or notre prototype est de conception objet, nous avons donc besoin d'une interface pour que notre application objet puisse communiquer avec une base de données relationnelle. Nous avons choisi d'utiliser le composant IDA pour assumer cette interfaçage. Comme nous le verrons dans la partie 4 consacrée à l'implémentation du prototype, IDA nous permet de stocker notre modèle de donnée objet dans le SGBDRE Oracle 8i.

3.2.2 L'analyse géométrique

3.2.2.1 Opérations et transformations géométriques

Que la donnée géographique soit modélisée sous forme vectorielle ou sous une forme raster, un des grands intérêts des systèmes d'information géographique est de pouvoir appliquer à celle-ci toute une série de transformations ou de filtres géométriques permettant d'associer, de combiner les données de diverses origines pour en extraire de nouvelles informations. On peut facilement classer les transformations géométriques appliquées aux données en deux grands groupes. Les opérations unaires portant sur une seule donnée ou sur un unique ensemble de données définissent le premier groupe, et les opérations binaires portant sur deux données ou deux ensembles de données définissent le second.

- Dans le premier groupe nous retrouvons les translations, les rotations, les homothéties mais aussi les calculs de longueur, de surface, de volume, les recherches de barycentre, la décomposition en primitives fondamentales, l'analyse de la forme ou encore le calcul d'englobant.

- Le second groupe comprend quant à lui des opérations comme l'intersection, le recouvrement, le regroupement, l'inclusion, la fusion, le calcul de distance ou la localisation relative.

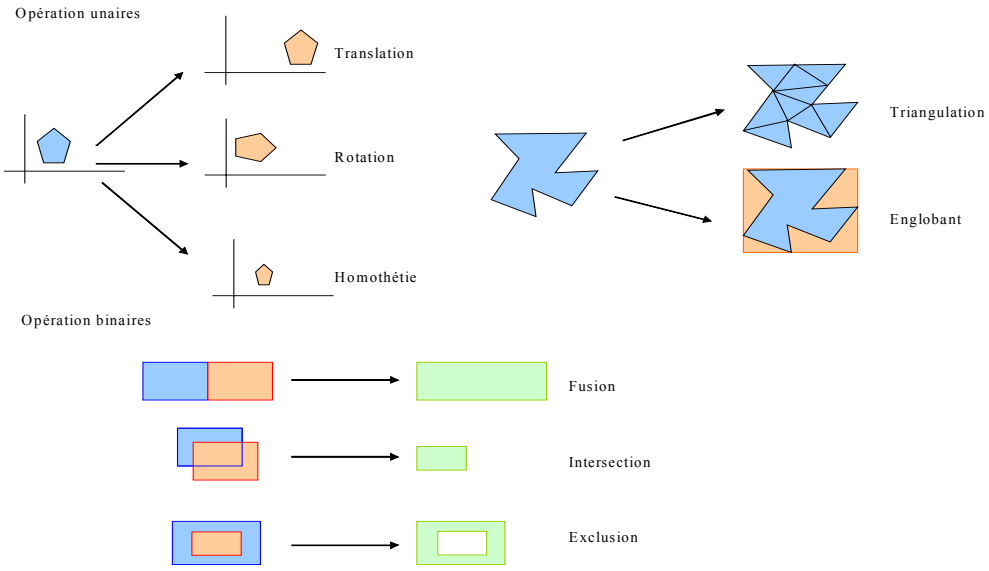


Figure 29 : Quelques transformations géométriques

3.2.2.2 Partage des taches géométriques

Ces fonctionnalités géométriques sont assurées par la grande majorité des SIG commerciaux. Aujourd'hui, elles apparaissent même de plus en plus souvent au niveau du composant de stockage. Les SGBDR font ainsi appel à des extensions spatiales pour prendre en charge une partie de ces opérations géométriques. Voici à titre d'exemple les extensions spatiales proposées par deux des acteurs principaux du marché des SGBDR : Postgres et Oracle.

3.2.2.2.1 Les extensions spatiales sous Postgress

Postgres propose un certain nombre d'opérateurs géométriques permettant d'interroger ou de modifier des données géométriques. Ces opérateurs sont listés dans le Tableau 6. Postgres permet d'étendre ces opérateurs relativement facilement en utilisant des fonctions SQL ou des fonctions C dynamiquement introduites dans le noyau.

Opérateur	Description
+	Translation
*	Changement d'échelle/rotation
/	Changement d'échelle/rotation
#	Intersection
#	Nombre de points du polygone
##	Point le plus proche
&&	Superposé?
&<	Superposé par la gauche?
&>	Superposé par la droite?
<->	Distance
<<	A gauche de ?
<^	Est dessous?
>>	A droite de?
>^	Est dessus?
?#	Intersecte ou est superposé ?
?-	Est horizontal ?
?-|	Est perpendiculaire ?
@-@	Longueur ou circonférence
?|	Est vertical?
?||	Est parallèle ?
@	Contient ou sur
@@	Centre de
~=	Semblable à

Tableau 6 : Les opérateurs géométriques de Postgres

Les opérateurs "<<", "&<", "&>", ">>", "@", "~=" et "&&" utilisent l'index spatial de type RTree de Postgres pour accélérer le traitement. La comparaison se fait ensuite sur la géométrie et le résultat est donc exact (par opposition à un filtre utilisant uniquement l'index spatial).

A ces opérateurs géométriques s'ajoutent des fonctions SQL qui sont listées dans le Tableau 7.

Fonction	Description
area(object)	Surface de l'objet
box(box,box)	Boîte résultant de l'intersection de deux boîtes.
center(object)	Centre de l'objet
diameter(circle)	Diamètre du cercle
height(box)	Hauteur d'une boîte
isclosed(path)	Est un chemin fermé?
isopen(path)	Est un chemin ouvert?
length(object)	Longueur de l'objet
pclose(path)	Ferme le chemin
npoint(path)	Nombre de points
popen(path)	Ouvre le chemin
radius(circle)	Rayon du cercle
width(box)	Largeur d'une boîte

Tableau 7 : Liste des fonctions géométriques SQL de Postgres

3.2.2.2.2 Les extensions spatiales sous Oracle

Le produit ORACLE 8i Spatial (version 8.1.5) distingue, dans le modèle objet relationnel, deux catégories de fonctions liées à la consultation et au traitement de données géographiques 2D (voire 2D ½). Elles agissent sur des objets stockés en base de données ou construits dynamiquement.

- Les « spatial operators » : ils utilisent l'index spatial pour les recherches. Ce sont essentiellement des prédicats utilisés dans une clause « where » rendant une valeur logique. Ils font interagir une collection d'objets, issue d'un ordre « select », et un objet donné en passant par l'index spatial.

Nom	Usage
SDO_FILTER	Prédicat utilisé dans une clause « Where » qui détermine si deux objets interagissent. Il travaille sur l'index mais ne rend que l'information déduite par le filtre primaire. Deux objets interagissent s'ils recouvrent certaines mêmes mailles de l'index. Il est généralement utilisé par les applications clientes qui réalisent elles-mêmes le filtre secondaire.
SDO_RELATE	Prédicat utilisé dans une clause « Where » qui teste la relation entre deux objets suivant un critère de type inclusion, intersection ou exclusion. Il réalise un calcul exact par application de filtres primaires et secondaires.
SDO_WITHINDISTANCE	Prédicat utilisé dans une clause « Where » qui teste si deux objets sont distants au maximum d'une valeur.

Tableau 8 : Les "Spatial Operators" d'Oracle

- Les « geometry functions » : ce sont des utilitaires géométriques de calcul ou de comparaison. Certaines fonctions sont des équivalents fonctionnels de « spatial operators », mais n'agissent que sur deux objets désignés sans passer par l'index spatial.

Nom	Usage
SDO_GEOM_AREA	Fonction qui calcule la surface d'un polygone.
SDO_GEOM_LENGTH	Fonction qui calcule la longueur ou le périmètre d'une ligne.
SDO_GEOM.RELATE	Fonction logique testant une ou des situations d'inclusion, d'exclusion ou d'intersection entre deux objets.
SDO_GEOM.SDO_BUFFER	Fonction qui génère une enveloppe parallèle (buffer) à un objet.
SDO_GEOM.SDO_POLY_DIFFRENCE	Fonction qui retourne la différence (exclusion ordonnée) entre deux objets. Le résultat est fourni sous forme d'objet.
SDO_GEOM.SDO_POLY_INTERSECTION	Fonction qui retourne l'intersection entre deux objets. Le résultat est fourni sous forme d'objet.
SDO_GEOM.SDO_POLY_UNION	Fonction qui retourne l'union entre deux objets. Le résultat est fourni sous forme d'objet.
SDO_GEOM.SDO_POLY_XOR	Fonction qui retourne l'exclusion symétrique entre deux objets. Le résultat est fourni sous forme d'objet.
SDO_GEOM.VALIDATE_GEOMETRY	Fonction logique qui teste si un objet est cohérent géométriquement.
SDO_GEOM.WITHIN_DISTANCE	Fonction logique qui teste si deux objets sont distants au maximum d'une valeur.

Tableau 9 : Les "Geometry Functions" d'Oracle

3.2.2.3 Spécificités 3D

En matière de 3D, aucun éditeur de SGBD ou de SIG n'était en mesure, en 2002, de proposer une panoplie d'outils d'analyse 3D comparable à ce qui existe en 2D. Et pour cause, l'attention portée aux données véritablement 3D étant relativement récente, les concepteurs de SIG et de SGBD n'ont pas encore pu se pencher en profondeur sur les problèmes d'analyse géométrique dans un environnement 3D. Cependant, il existe d'autres domaines ou l'existant est riche sur ces problématiques et en particulier celui de la conception assistée par ordinateur (CAO). La librairie Open Cascade, moteur du logiciel de CAO Euclid développé jusqu'à la fin des années 90 par Matra Datavision [OPENCASCADE 1999], offre de nombreuses fonctionnalités géométriques dont d'intéressantes opérations booléennes sur les polyèdres. Ce n'est pas la seule librairie à offrir de telles fonctionnalités, mais elle possède l'avantage d'être aujourd'hui en « open source ».

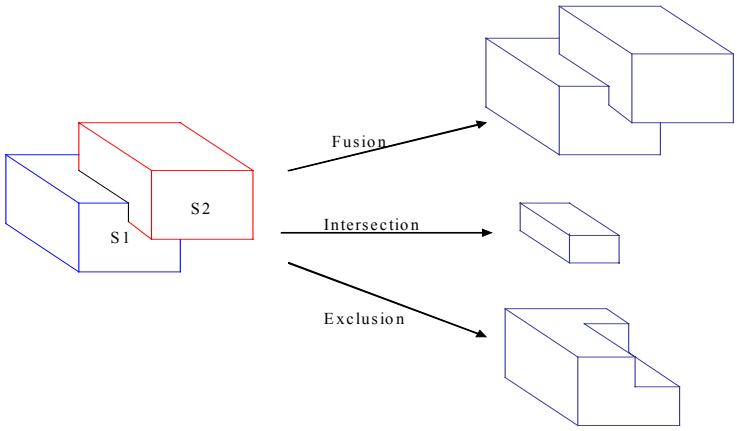


Figure 30 : Opérations booléennes 3D

3.2.2.4 Sur la route d'un prototype

La palette des opérations géométriques 3D intégrables comme outils d'analyses et d'exploitations dans un système d'information géographique est vaste. Cependant il est des domaines où certaines opérations complexes passeront pour secondaires. Les opérations booléennes 3D peuvent s'avérer d'excellents outils pour un géologue souhaitant manipuler des volumes représentatifs de couches géologiques, cavités ou inclusions et jouer des scénarios d'évolution du sous-sol. En revanche, dans le cadre de l'étude qui a servi de fil directeur à cette thèse, ces mêmes opérations booléennes ne présentent que peu d'intérêt, d'où notre choix de ne pas les intégrer au stade du prototype.

Nous verrons dans la partie consacrée à la réalisation du prototype (chapitre 4) que nous avons implémenté quelques fonctionnalités géométriques 3D comme le calcul de distance, la rotation et la translation d'objets, ou encore la création de buffer 3D. Cependant, il ne s'agit pas d'un axe principal de notre travail de recherche. En effet, l'adaptation au 3D des algorithmes d'opérations géométriques 2D pose rarement de réels problèmes et se résume bien souvent au simple rajout d'une coordonnée z.

3.2.3 L'index spatial, une optimisation pour la recherche de données géographiques

3.2.3.1 Présentation

De par la nature des informations traitées (vecteurs complexes, images …) les bases de données géographiques sont volumineuses. Elles le deviennent de plus en plus au fur et à mesure de l'amélioration des processus d'acquisition et de la disponibilité des bases référentielles. Dans ce contexte, les systèmes d'information les utilisant ne peuvent plus raisonner sur l'ensemble de l'information, ils doivent opérer une sélection en amont afin de ne garder que l'information utile. Les index spatiaux ont pour fonction première de fournir une structure afin d'optimiser les accès mémoires (temporaires ou permanentes) lors de requêtes définies sur critères spatiaux.

Cette optimisation se fait au travers de deux objectifs :

- La définition d'une structure de données (ou index) qui limite les lectures successives et offre un accès le plus direct possible à l'information.

- Le regroupement d'informations (« clustering ») qui part du principe que des informations qui sont proches dans la réalité doivent l'être aussi en mémoire. Les regroupements sont matérialisés par les éléments terminaux (ou feuilles) des arbres de structure.

Les techniques d'index spatiaux sont assez nombreuses. Leurs fondements théoriques sont apparus essentiellement pour étudier l'espace 2D. Pourtant chacun des index décrits dans ce chapitre peut être transcrit en 3D. Nous présentons ici six des techniques les plus utilisées accompagnées de pistes pour les étendre à la 3D.

3.2.3.2 Grilles

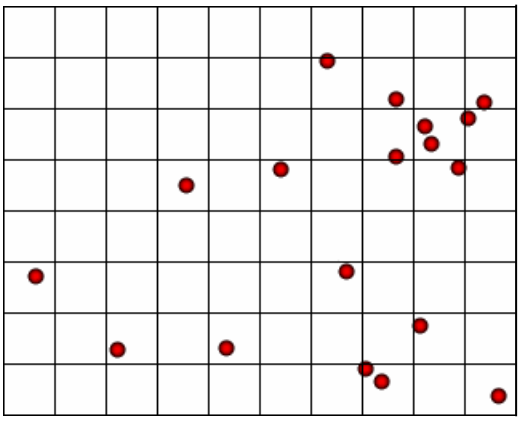


Figure 31 : Grille fixe de points

Le principe des grilles consiste à diviser l'espace en rectangles identifiables par un indice dans chaque dimension. Les grilles ne sont pas des structures hiérarchiques. A chaque cellule ou maille est associé l'ensemble des objets qu'elle contient.

L'exemple de la Figure 31 traite d'une indexation de points. Dans ce cas, il n'y a pas d'ambiguïté et un point appartient à un et un seul élément de grille. Cette technique peut aussi être utilisée pour des lignes ou surfaces. Dans ce cas, la difficulté de gestion consistera dans le multi-référencement possible des objets (objet touchant plusieurs mailles).

Plusieurs variantes existent :

Dans la variante « Fixed Grid » [BENTLEY 1979], les grilles sont régulières et toutes les cellules sont de même taille.

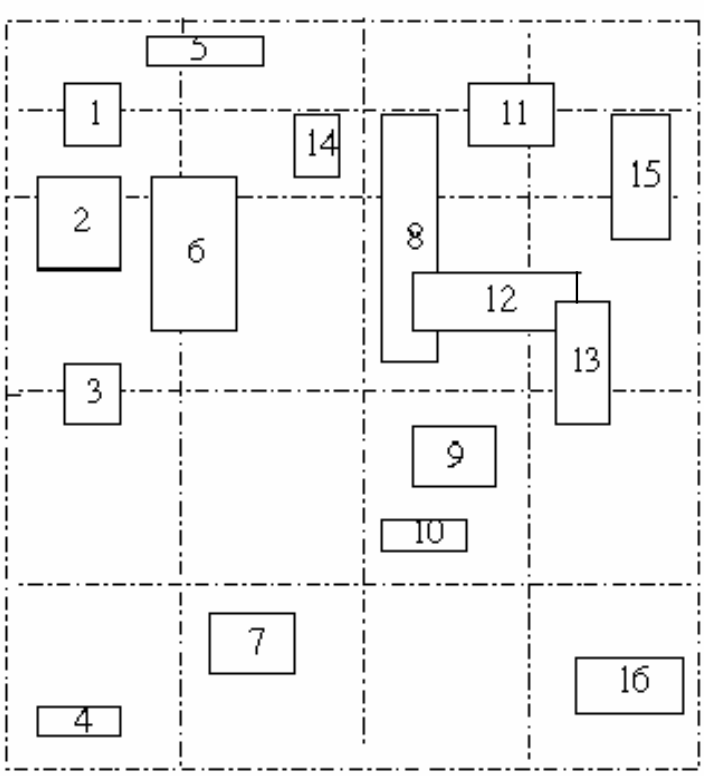


Figure 32 : "Grid File"

Dans la variante « Grid File » [NIEVERGELT 1984], la partition s'adapte à la densité locale des objets en introduisant des tailles variables pour les mailles.

Le « Field Tree » est une solution à base de grilles superposées. Chaque grille a un pas fixe. Plusieurs principes peuvent être retenus pour le maillage. Un de ceux-ci consiste à décaler les origines des grilles et à calculer les tailles de mailles de manière à ce qu'un objet se rattache à la grille la plus proche qui le contient entièrement (ce qui évite le multi-référencement), et que le niveau de grille correspondant soit proche de la taille de l'objet.

Adaptation à la 3D :

L'extension 3D de grilles 2D est triviale. Il suffit de partitionner l'espace 3D en cellules parallélépipédiques.

3.2.3.3 Du Quadtree à l'Octree

Les arbres quaternaires ou quadtrees réalisent une décomposition récursive de l'espace en quatre quadrants. Plusieurs méthodes à base d'arbre quaternaire existent [SAMET 1990].

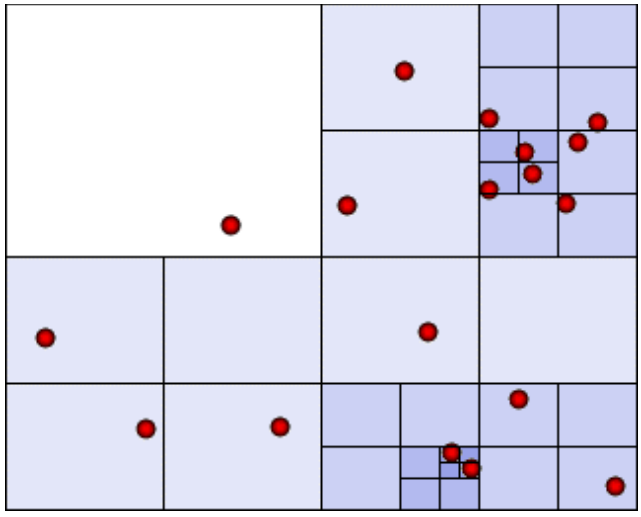


Figure 33 : Quadtree de points

L'une des plus simples est l'arbre Q. Dans l'arbre Q, chacun des quadrants est récursivement décomposé jusqu'à ce que le nombre d'objets qu'il contient soit inférieur ou égal à un seuil préalablement fixé. Pour les lignes et surfaces le multi-référencement doit aussi être géré, ce qui est illustré par la figure suivante.

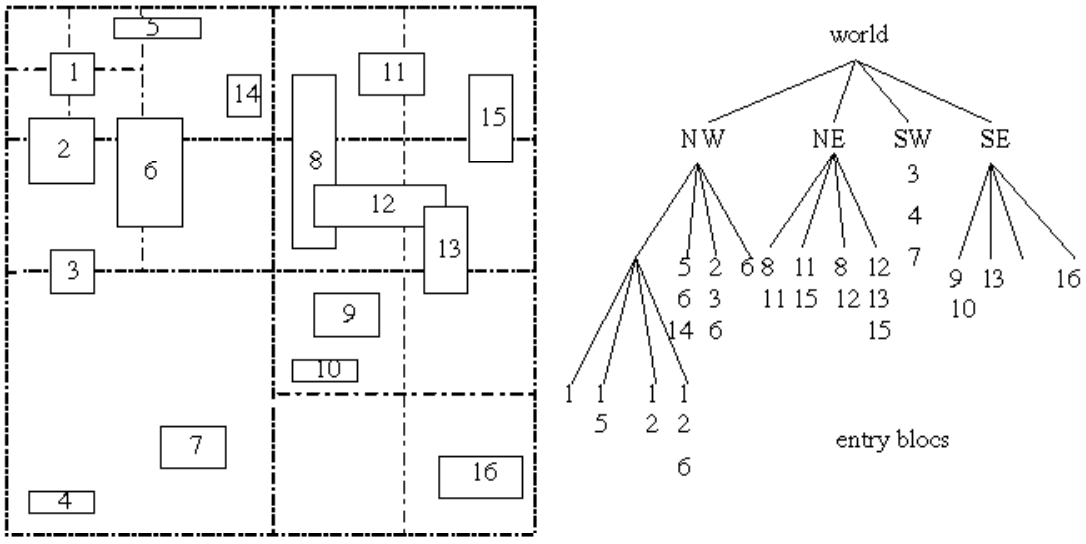


Figure 34 : Quadtree de rectangles avec recouvrement

Le grand avantage des arbres de hiérarchisation est de permettre d'élaguer des branches entières en déterminant au niveau des nœuds une propriété qui sera vérifiée pour toutes les feuilles. Ainsi, les quadtrees s'adaptent bien à des couvertures géographiques de densité non uniforme. Cependant, il faut veiller à limiter le niveau de découpage minimal afin de ne pas grossir exagérément l'arbre.

Les stratégies d'insertions peuvent être guidées par la taille des objets ou le nombre maximum d'éléments dans une maille.

Les quadtrees sont utilisés dans de nombreux domaines de l'informatique : SIG, astronomie, maillage de structure, compression d'image etc.

Dans ce dernier domaine, on utilise généralement une structure de type « region quadtree » qui permet de définir une approximation raster d'un polygone. Le principe consiste à déterminer les éléments de l'arbre entièrement couverts par le polygone en commençant par les mailles les plus grosses. La représentation de la couverture du polygone peut alors se faire par un codage de chaque nœud couvert. Le code résultant représente une compression de l'information initiale.

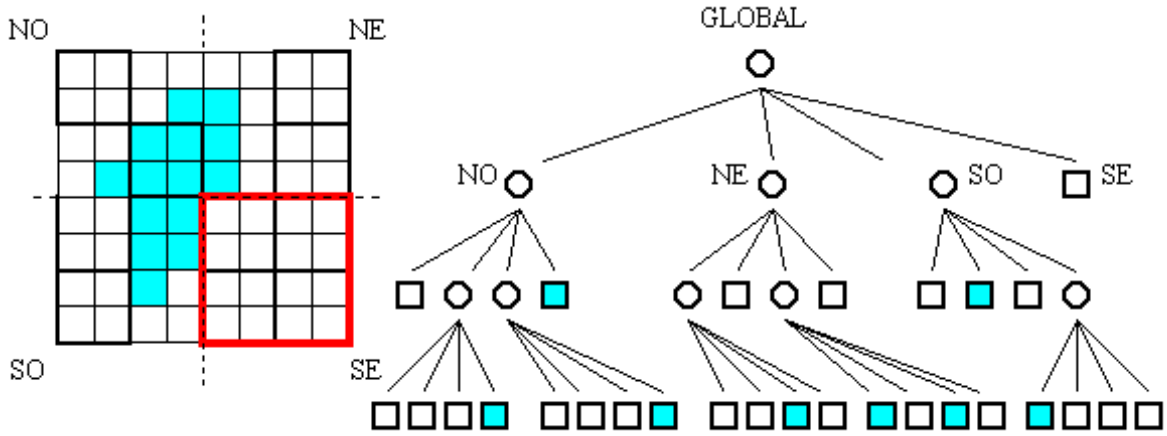


Figure 35 :Compression d'image par "Region Quadtree"

Adaptation à la 3D :

L'Octree est l'extension 3D du quadtree. Il est une simple généralisation de cette hiérarchisation à l'espace 3D, avec cette fois-ci huit sous-espaces à chaque nœud.

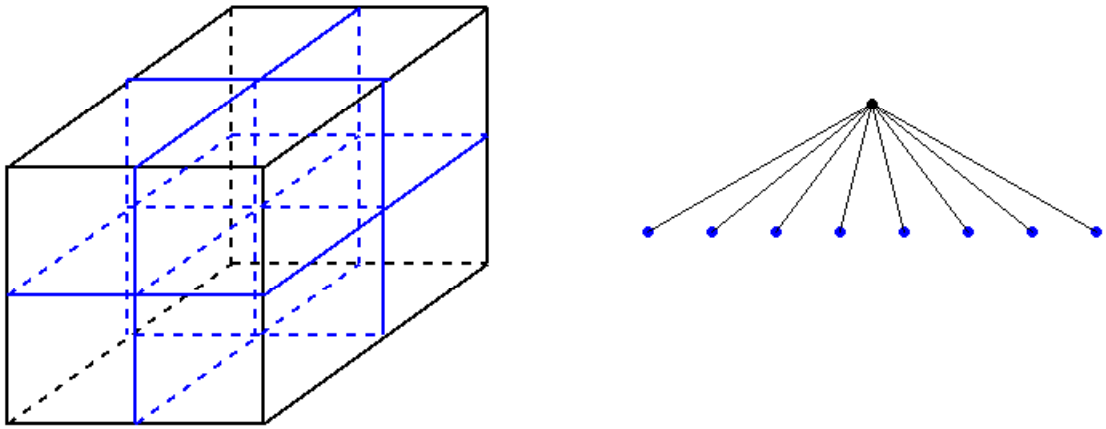


Figure 36 : Représentation du premier niveau Octree

3.2.3.4 Les courbes mathématiques

L'étude des transformations de l'espace 2D (ou 3D) vers un espace à une dimension n'est pas une préoccupation récente. De nombreux mathématiciens se sont penchés sur cette problématique et ont découvert des courbes aux propriétés intéressantes :

- la courbe continue passe une seule fois à travers tous les points de l'espace,
- les points voisins dans la courbe le sont généralement dans l'espace,
- la courbe constitue une transformée entre elle-même et l'espace à n dimensions.

Les courbes les plus fréquemment utilisées sont celles de Peano (modifiée peu de temps après par Hilbert). On retrouve également les courbes de Gray, Cantor ou Sierpinski. Ces courbes permettent d'établir un codage qui ordonne et repère les régions de l'espace.

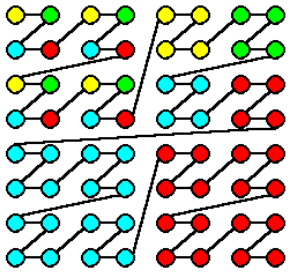


Figure 37 : Courbe de Peano

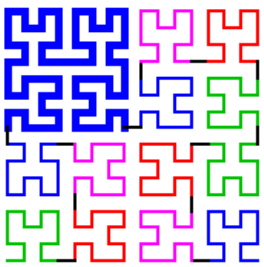


Figure 38 : Courbe de Hilbert

En 1966 Morton développa pour le système d'information géographique canadien (SIGC) un système de codage qui entrelace les bits de la représentation binaire des coordonnées géographiques. La clé résultante est appelée clé de Morton (ou clé de Peano ou « Z-Order »).

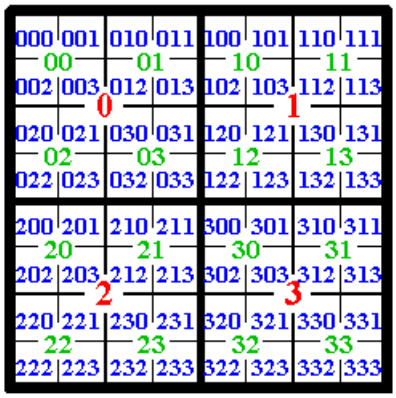


Figure 39 : Codage de Morton

Ce codage est particulièrement adapté aux structures de Quadtree et a donné naissance à la famille des «Quadtrees linéaires». Cette structuration ne propose pas la construction d'un arbre mais consiste uniquement à rattacher un ou plusieurs codes de ce type à chaque objet.

Adaptation à la 3D :

La Figure 40 présente l'adaptation du codage de Morton à la 3D. Un codage de Morton 2D ajoute 2 bits à chaque niveau de subdivision de l'espace. Un codage 3D ajoute quant à lui 3 bits à chaque niveau de subdivision. Elle montre le codage binaire de la clé en haut et le codage décimal en bas.

Lorsqu'on subdivise la cellule "011", on fait donc un décalage de 3 bits vers la gauche et on ajoute la clé de la subdivision locale. On obtient ainsi les clés "011000" à "011111" (soit de 24 à 31 en décimal).

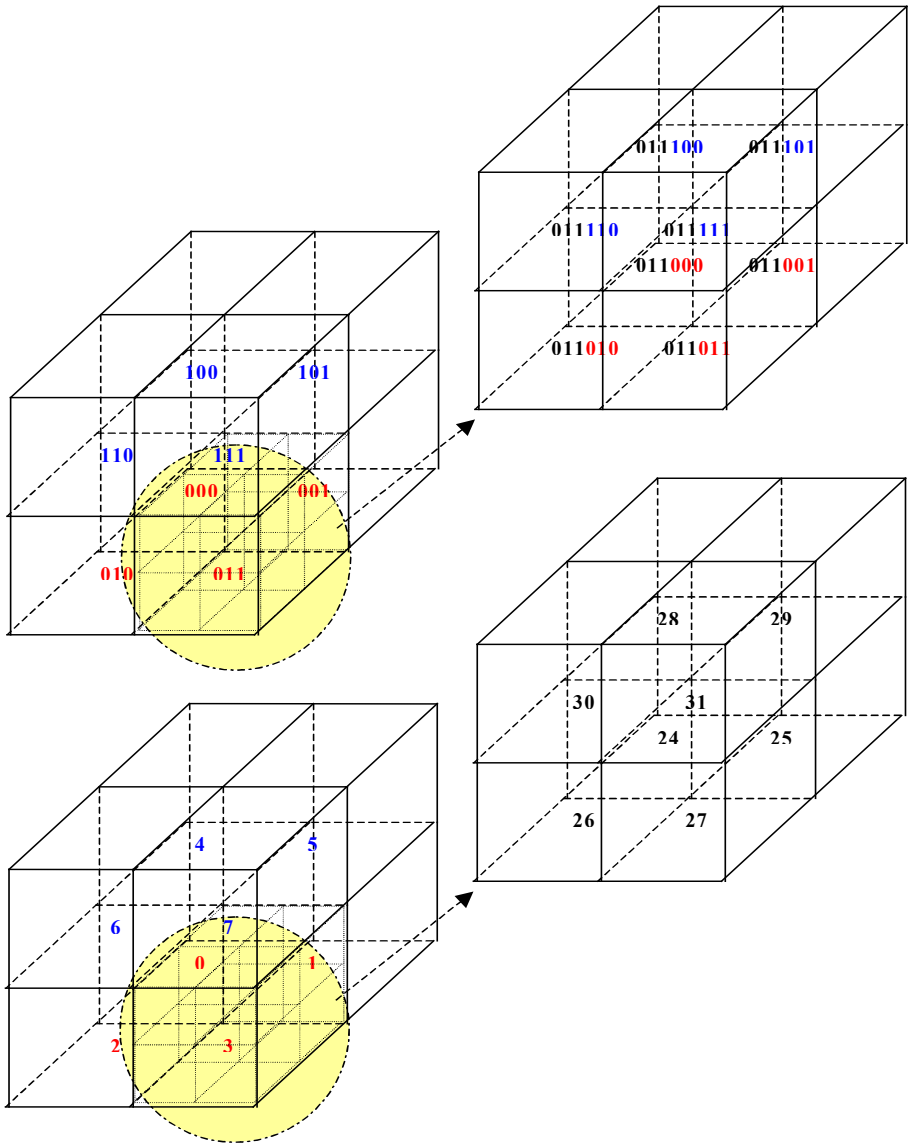


Figure 40 : Codage de Morton 3D

3.2.3.5 Arbres KD ou KD-tree

La structure de « KD-Tree » (Figure 41) a été initialement conçue pour stocker des points [BENTLEY 1975]. L'espace est divisé en deux parties en utilisant alternativement la coordonnée x ou y du point suivant la parité du niveau de l'arbre. Un nouveau point est inséré par parcours de l'arbre jusqu'à ce que soit trouvé un nœud sans descendance.

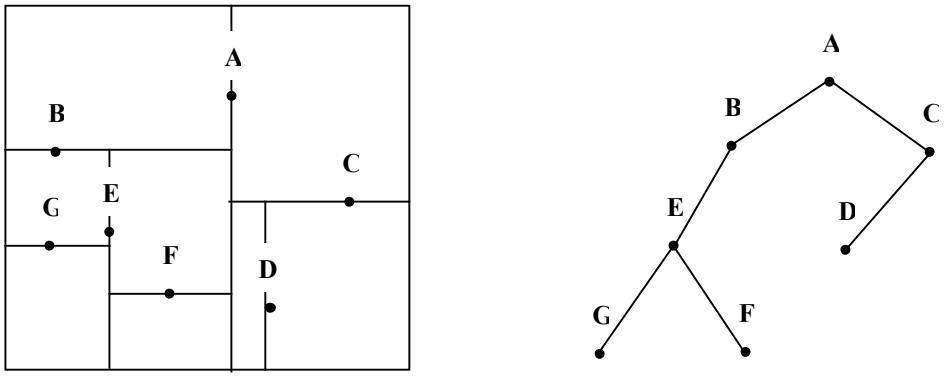


Figure 41 : KD-Tree

Le désavantage important de ce KD-Tree est que la partition de l'espace dépend de l'ordre d'insertion des points. Dans le pire des cas, il existe autant de niveaux d'arbre que de nœuds.

Le « adaptative KD-Tree » [BENTLEY 1979] résout ce problème en choisissant une droite qui découpe l'espace en deux groupes de même nombre de points (à l'unité près). Cette solution a le désavantage de ne pas être dynamique et nécessite des opérations complexes pour l'insertion et la destruction de points.

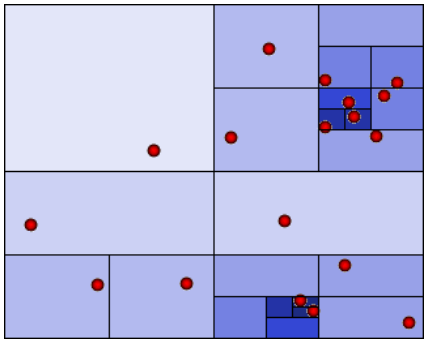


Figure 42 : KD-Tree « Bintree »

Une autre variante de KD-Tree est le « bintree » (Figure 42) [TAMMINEN 1984]. La décomposition de l'espace 2D d'un «bintree» est alors très semblable à une décomposition en quadtree à ceci près que l'on ne divise pas l'espace en 4 parties à chaque nœud mais en 2 parties égales par rapport à une droite (à un plan en 3D) perpendiculaire à chacun des axes alternativement.

Lors de la construction de l'arbre, si la condition d'arrêt de la subdivision est basée sur la complexité des objets du sous-espace, les KD-Tree auront tendance à générer moins de feuilles et donc moins de sous-espaces qu'un quadtree. D'un autre côté, pour un même niveau de subdivision de l'espace, le KD-Tree donnera un arbre de profondeur supérieure.

Adaptation à la 3D :

L'extension naturelle des KD-Tree au 3D consiste à subdiviser l'espace par des plans perpendiculaires à chacun des trois axes alternativement.

3.2.3.6 Arbre BSP

L'arbre BSP ou « Binary Space Partitioning » est à l'origine un index spatial adapté à la 3D. Nous ne parlerons pas ici de ses applications 2D puisqu'aujourd'hui il est essentiellement utilisé dans le domaine des jeux 3D où il constitue une méthode efficace pour réaliser des traitements de type « parties cachées».

L'arbre BSP (Figure 43) conserve cette idée de subdiviser récursivement l'espace 3D par rapport à un plan, mais en autorisant un plan quelconque à chaque nœud. On peut choisir cet élément en fonction de la scène, en particulier si elle est constituée de polygones. En fait, chaque plan dans lequel se situe un polygone est un bon candidat. A chaque nœud on stocke donc les informations sur le plan de division et l'ensemble des polygones contenus dans ce plan.

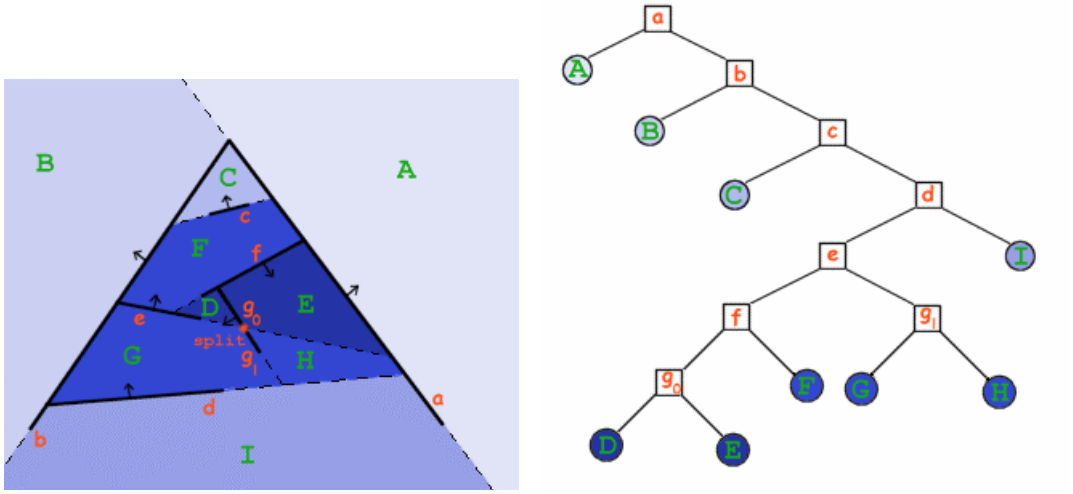


Figure 43 : Partition BSP et l'arbre correspondant

Comme avec les autres structures de partition de l'espace, les polygones se situant à cheval sur une limite doivent être traités avec égard. Deux solutions : les placer dans les deux espaces ou les couper en deux polygones, un par espace. Dans la pratique, cette deuxième solution est souvent préférée, en particulier parce qu'elle permet d'utiliser le BSP pour trier les faces à l'affichage.

Comme dans le KD-tree, les feuilles d'un BSP représentent des sous-espaces, et elles constituent dans leur ensemble une partition de l'espace. Mais si dans le cas d'un octree ou d'un kd-tree 3D le volume occupé par un nœud est simple (boîte alignée sur les axes), dans le cas d'un BSP c'est un polyèdre dont la seule propriété est d'être convexe. Afin de permettre un élagage facile de l'arbre, il est intéressant de calculer un volume englobant grossier mais plus simple à traiter, comme par exemple une boîte englobante alignée sur les axes.

Malheureusement, la construction de l'arbre engendre souvent un grand nombre de polygones et, une fois construit, l'arbre est difficilement modifiable. Cette technique doit donc être réservée au traitement de lots de données statiques.

3.2.3.7 Arbre R ou R-Tree

Le principe des arbres R est de regrouper les objets spatiaux dans les feuilles selon un critère de proximité. Ils sont l'adaptation au domaine spatial de l'arbre B (Figure 44).

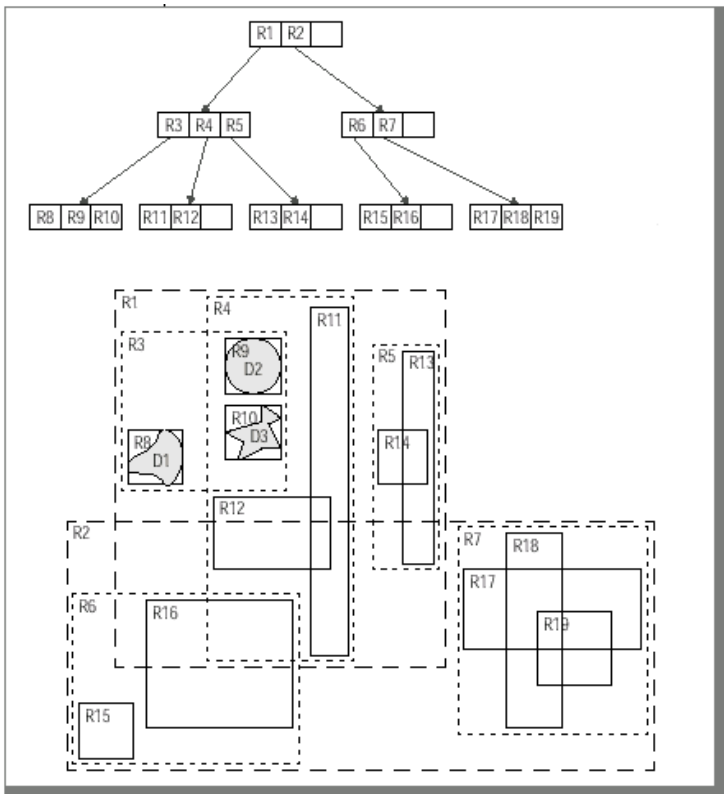


Figure 44 : R-Tree de rectangles et les niveaux de l'arbre

L'arbre B est une structure arborescente permettant de stocker des index répartis suivant un critère déterminé (exemple : inférieur et supérieur). L'arbre est équilibré, c'est-à-dire qu'il comprend le même nombre de niveaux dans chaque branche ; une recherche dans ce type d'arbre a donc toujours la même durée. C'est un des index les plus utilisés dans les SGBD.

Dans le R-Tree de base [GUTTMAN 1984], un objet spatial est inséré dans une seule feuille mais les nœuds internes, représentants des portions de l'espace, peuvent se recouvrir. Ainsi, une recherche spatiale nécessite parfois le parcours de plusieurs sous-arbres.

Comme dans le cas de l'arbre B, les feuilles sont à la même profondeur. Chaque nœud a un nombre d'entrées compris entre m (minimum) et M (maximum), $0<m<M/2$

L'arbre R est une structure qui s'adapte à des données évolutives et de distribution quelconque. Elle est dynamique dans la mesure où il n'est pas nécessaire de connaître a priori la taille et la répartition de densités de l'espace. En revanche, les arbres de ce type sont difficiles à mettre en œuvre (ce qui n'est cependant pas un problème utilisateur) et les stratégies d'insertion déterminent significativement les performances.

Plusieurs variantes existent :

- Le R+-Tree [SELLIS 1987] interdit le recouvrement des nœuds (Figure 45). Une recherche sur un point correspond alors à une seule traversée de l'arbre. En revanche, il est nécessaire de dupliquer certains éléments dans plusieurs nœuds. Cette méthode est efficace pour accélérer les temps de recherche dans un espace uniforme. Cependant, plus la densité des éléments est grande et plus il y a de duplications d'éléments.

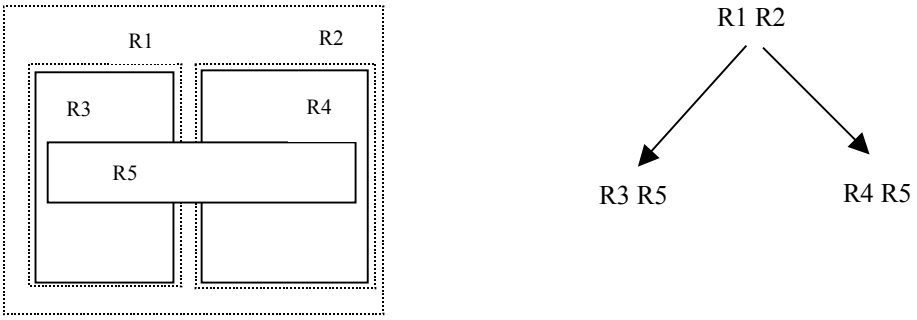


Figure 45 : R+ Tree

- Le R*-Tree [BECKMANN 1990] est basé sur la même structure que le R-Tree de base. Cependant, la stratégie d'insertion est modifiée. Quand un nœud est trop rempli, le découpage n'est pas réalisé immédiatement mais l'algorithme attend de nouvelles entrées (dans une limite fixée) pour décider si d'autres réorganisations ne sont pas plus judicieuses. Par ailleurs, l'algorithme essaye d'arbitrer entre deux stratégies parfois contradictoires : minimiser les recouvrements ou minimiser la surface des nœuds.

Cette problématique est illustrée par la Figure 46 et la Figure 47 :

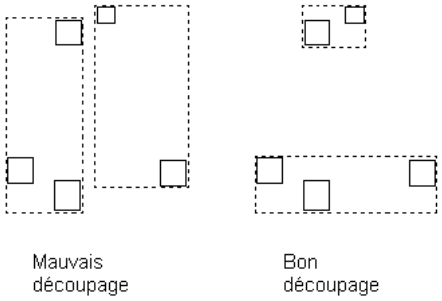


Figure 46 : Stratégie de non-recouvrement

Dans ce cas, la stratégie de non-recouvrement permet aussi de limiter la surface.

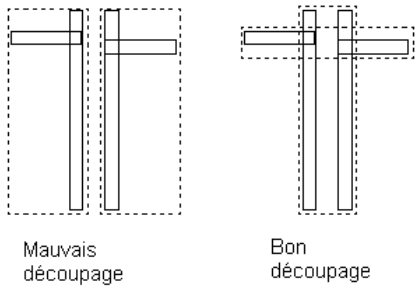


Figure 47 : Stratégie de minimisation de la surface

Dans cet exemple, il est préférable de minimiser la surface plutôt que d'interdire le recouvrement.

- Le « Hilbert-Tree » combine les principes de l'arbre R et un codage de Hilbert qui caractérise la taille des nœuds. Le codage est utilisé pour les insertions et destructions. La recherche est faite au travers de l'arbre.

- Le « Sphere-Tree » [OOSTEROM 1990] est similaire au R-Tree en remplaçant les rectangles par des cercles (Figure 48).

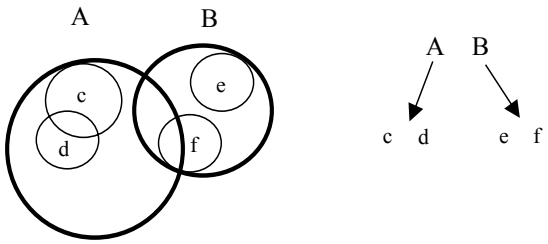


Figure 48 : Le Sphere-Tree

Adaptation à la 3D :

Un R-Tree 3D repose sur les mêmes principes qu'un R-Tree 2D, il peut se décliner suivant les mêmes variantes. Les rectangles englobants sont simplement remplacés par des boîtes parallélépipédiques englobantes (Figure 49) [KOFLER 1998].

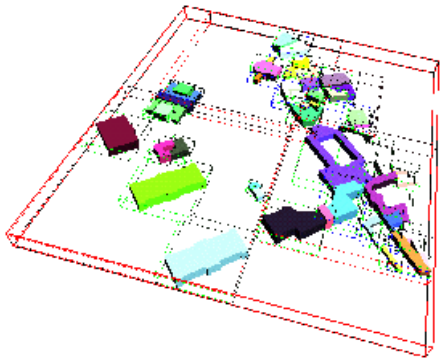


Figure 49 : R-Tree 3D [KOFLER 1998]

3.2.3.8 Notre choix final : le R-Tree 3D

Les structures conçues pour optimiser la sélection de données sur critères spatiaux sont nombreuses et le choix est vaste. Il n'y a pas vraiment d'index spatial meilleur que les autres, en revanche certains sont mieux adaptés que d'autres pour un domaine d'application précis. Le système de gestion de base de donnée Oracle (version 8i et précédentes) indexe les données qu'il stocke par un Quadtree, les concepteurs de jeux 3D utilisent très souvent des arbres BSP, quant à nous, nous pensons qu'un R-Tree est la structure qui convient le mieux pour notre prototype.

L'avantage du R-Tree tient essentiellement par ses aspects dynamiques. L'ajout et la suppression de données sont parfaitement gérés par un R-Tree, ce qui n'est pas toujours vrai pour des structures plus statiques comme l'arbre BSP.

Bien que les objets d'une scène géographique soient essentiellement étalés sur le plan horizontal et très peu en altitude, nous avons choisi d'implémenter un R-Tree 3D plutôt qu'un R-Tree 2D. Notons que l'intégration de la troisième dimension dans un R-Tree n'engendre aucune difficulté de développement supplémentaire.

3.3 Calcul de visibilité

On m'voit…. on m'voit plus… on m'voit un peu … on m'voit plus… (Chantal Lobby alias « Cartapus », Asterix Mission Cléopatre 2001).

Nombreux sont les domaines d'applications suivant avec grand intérêt les progrès réalisés sur les fonctionnalités de calcul de visibilité 3D. Les demandes sont diverses, certaines s'appliquent à de vastes MNT alors que d'autres se résumeront par une requête ponctuelle sur un objet de la scène proche de l'observateur. Certes il s'agit là de deux requêtes d'intervisibilité, mais les moyens et méthodes à mettre en œuvre pour y répondre ne sont pas les mêmes. Dans ce sous-chapitre, nous aborderons les problèmes de visibilité en deux temps pour d'abord s'intéresser aux larges étendues où l'importance du sur-sol par rapport au relief est minimale, puis nous verrons le cas inverse où les objets composant le sur-sol deviennent les obstacles les plus probables à la propagation du signal visuel. Les problèmes de visibilité sur MNT ont fait l'objet de nombreuses études et nous présentons ici essentiellement une synthèse de ces travaux. En revanche, il est beaucoup plus difficile de trouver un existant sur l'étude de visibilité des objets du sur-sol dans un système d'information géographique, aussi proposons nous nos propres solutions sur ce point.

3.3.1 Notion d'intervisibilité par rapport à un Modèle Numérique de Terrain

3.3.1.1 Problèmes de visibilité nécessitant une faible résolution

Pour des scénarios de type préparation de missions aéroportées, qui ne nécessitent qu'une résolution de données assez faible, il peut être intéressant de traiter les problèmes de visibilité relativement à la structure générale du terrain et non pas dans le détail des objets 3D. Pour d'autres applications comme le positionnement d'observateurs militaires, la détermination d'itinéraires restant invisibles depuis certains points d'observation, la mise en place de relais radio, ou encore la localisation de tours incendies, il peut même être envisagé de fusionner MNT et objets 3D dans un Modèle Numérique d'Elévation (MNE) modélisant l'altitude maximale (sur-sol compris) pour tout point de coordonnées (x,y).

3.3.1.2 Note sur les Modèles Numériques de terrains

Une surface topographique σ peut être vue comme l'image d'une fonction f bivariante définie sur un domaine D du plan Euclidien :

$$\sigma = \{(x,y,f(x,y))/(x,y)\in D\}$$

Construit à partir d'un ensemble fini de mesures effectuées sur le terrain, un modèle numérique de terrain (MNT) représente une telle surface. Soit S le sous-ensemble fini de points du domaine D, les points de S peuvent être dispersés ou former un maillage régulier. Deux types de MNT sont généralement utilisés dans les contextes de visibilité associés aux SIG (paragraphe 2.3.3) :

- Les TIN (Triangulated Irregular Networks) qui sont définis par une triangulation du domaine D avec comme sommets des triangles les points du sous-ensemble S. La fonction f qui définit la surface est alors linéaire pour chaque triangle. Un TIN est donc une partition de surfaces

triangulaires planes. On lui associe souvent une triangulation de Delaunay (les cercles circonscrits aux triangles ne contiennent aucun autre sommet), qui présente l'avantage de faciliter les interpolations.

- Un RSG (Regular Square Grid) est défini par un partitionnement en rectangles (appelés cellules ou pixels) du domaine D, formant ainsi une grille régulière sur D. Les fonctions utilisées sur de telles partitions dépendent du degré de continuité désiré pour la surface. Généralement, il s'agit soit de fonctions bilinéaires, soit de fonctions constantes pour chaque cellule de la grille. Notons qu'il existe des méthodes de conversion permettant de passer d'un RSG à un TIN ou vice versa. Le type « raster » vu dans le chapitre sur la modélisation du MNT (2.3.3) s'apparente à un RSG.

3.3.1.3 Structure de visibilité pour les MNT

Deux points A et B sont dits visibles si et seulement si l'intérieur du segment joignant les deux points se trouve strictement au-dessus du MNT ; un tel segment ne pouvant toucher la surface du MNT qu'en ses extrémités A et B. Tout point A se situant sur ou au-dessus du MNT peut être choisi comme point de vue.

Les structures de visibilités permettent de stocker de l'information sur la visibilité des surfaces composant un MNT et fournissent ainsi une aide précieuse pour répondre aux requêtes de visibilité. Nous abordons ici plus en détail les principales structures utilisées pour représenter la visibilité d'un terrain par rapport à un ou plusieurs points de vue [DE FLORIANI 1998].

3.3.1.3.1 Présentation des structures

Le « viewshed »

La structure de visibilité de base est le « viewshed », littéralement l'étendue de vision. Cette structure est une carte de visibilité qui regroupe, pour un point de vue V, l'ensemble des points du MNT (ou situés à une hauteur h au-dessus du MNT) visibles depuis V, c'est-à-dire :

$$viewshed(V)=\{W\in D/W \text{ est visible depuis } V\}$$

L'horizon

Une autre structure de visibilité utile est l'horizon d'un point de vue V. L'horizon de V correspond au contour externe du viewshed, c'est-à-dire à l'ensemble des points les plus éloignés visibles depuis A :

$$Horizon(V)=\{W\in D/\ W \text{ est visible depuis } V \text{ et } \forall Q\in D \text{ si } W\in [VQ] \text{ alors } Q \text{ est invisible depuis } V\}$$

Multi-visibilité

Les structures de visibilité pour plusieurs points de vue peuvent être définies par différentes combinaisons des viewsheds en ces points :

- Le recouvrement : le terrain est partitionné en régions de sorte que chaque région soit visible depuis un ensemble donné de points de vue.

- Les opérateurs booléens : il s'agit principalement d'intersections ou d'unions de viewsheds permettant de déterminer les zones visibles depuis tous les points de vue ou par au moins un point de vue.

- Les opérateurs de comptage : une surface peut être partitionnée en région pour que chacune des régions puisse être vue par le même nombre de points de vue.

L'ensemble des points de vue est généralement un sous-ensemble des sommets d'un TIN ou des cellules d'un RSG.

Indice de visibilité

Enfin, l'indice de visibilité présenté par Randolph Franklin [FRANKLIN 2000] offre la particularité de pouvoir comparer rapidement la visibilité de plusieurs points de vue. Pour un observateur O, l'indice de visibilité de O correspond à la fraction du disque de rayon R centré en O qui est visible depuis O. Franklin considère que quelle que soit la méthode employée, il y a toujours une part d'approximation dans la détermination des viewsheds. Aussi préconise-t-il de ne pas rechercher à tout prix l'étendue exacte d'un viewshed pour chaque observateur, mais de plutôt s'intéresser aux étendues relatives des viewsheds.
Une des applications possible est le positionnement d'antennes de transmission radio. Rechercher quelles sont, parmi plusieurs localisations d'antennes possibles, celles qui offrent relativement les meilleures indices de visibilités est plus rapide que de déterminer le viewshed exact de chaque emplacement sélectionné.

3.3.1.3.2 Stratégie d'implémentation des structures

Une structure de visibilité peut être implémentée de manière continue ou discrète. Une implémentation continue se révélera toujours plus précise qu'une implémentation discrète, mais souvent au détriment des temps de calculs. Les complexités des algorithmes sont issues de [DE FLORIANI 1998].

3.3.1.3.2.1 Les viewsheds

Pour les viewsheds, une implémentation continue dresse une carte de visibilité continue en établissant un découpage du MNT selon le critère de visibilité. Certains triangles ou polygones partiellement visibles seront alors subdivisés. Cette technique est principalement utilisée dans le cas des TIN car il s'agit là de MNT beaucoup plus légers d'un point de vue stockage et traitement que les RSG (un RSG ne permet pas d'interpolation linéaire et comporte un nombre important de sommets). La carte de visibilité continue d'un TIN possédant n sommets a une complexité au pire de $O(n^2)$.

Dans le cas des RSG, on utilise habituellement une implémentation discrète en marquant chaque cellule ou sommet de la grille comme visible ou invisible. Le tableau de booléen résultant est nommé carte de visibilité discrète. Les RSG sont généralement denses, aussi les cartes de visibilité discrète sont-elles assez précises et régulièrement utilisées. La complexité d'un calcul de visibilité pour un RSG de n sommets est au pire de O(n).

3.3.1.3.2.2 Les horizons

Une représentation continue de l'horizon d'un point de vue V est une suite ordonnée d'arêtes autour de V. Comme pour la carte de visibilité continue, ce type de représentation est principalement utilisé pour

les TIN. La complexité d'un horizon pour un TIN de n sommets est O(n.α(n)), où α est une fonction croissant lentement (inverse de la fonction Ackermann).
Dans le cas de RSG, les horizons peuvent être représentés de manière discrète par une collection de cellules de la grille. Cependant, le gain de complexité par rapport au viewshed étant mince, on utilise très rarement une représentation discrète de l'horizon.

3.3.1.3.2.3 Multi-visibilité

Les structures de visibilité relatives à un ensemble de points peuvent être implémentées de manières diverses. De par la dimension imposante d'une implémentation continue, les méthodes discrètes sont les plus utilisées quand un nombre important de points de vue sont à prendre en compte. Le graphe de visibilité proposé par Puppo et Marzano [PUPPO 1997] représente un recouvrement des cartes de visibilité discrète pour chaque point de vue. Dans ce graphe de visibilité, deux nœuds visibles (cellule ou sommet) sont joints par un arc. La complexité d'un tel graphe est $O(n^2)$.

La carte d'intervisibilité entre deux régions de Mills [MILLS 1992] indique pour chaque point d'une région destination le nombre de points de la région source d'où ils sont visibles. La taille d'une telle carte est fonction du nombre de points de la région de destination.

3.3.1.3.2.4 L'indice de visibilité

Cet indice est obtenue en lançant pour chaque observateur un nombre fixe de rayons (par exemple 36 rayons tous séparés de 10 degrés). Le long de chaque rayon, un sous-ensemble de N points est individualisé et un algorithme mono-dimensionnel de type LoS (line of sight) est appliqué. Le nombre K de points visibles divisé par le nombre N de points testés détermine l'indice de visibilité K/N avec comme écart type :

$$\sigma = \sqrt{\frac{K}{N^2}(1-\frac{K}{N})}$$ [FRANKLIN 2000].

3.3.1.4 Algorithmes pour calculs de visibilité

3.3.1.4.1 Les algorithmes applicables aux TIN

Comme nous l'avons vu précédemment, la plupart des implémentations à partir de TIN utilisent des algorithmes de visibilité continue qui exploitent les surfaces polygonales d'un TIN.

3.3.1.4.1.1 Détermination des viewshed

Le problème de détermination des cartes de visibilité continues est étroitement lié au problème plus général de détermination des surfaces cachées. Une approche commune aux calculs de visibilité sur des TINs, dite « front-to-back », est de commencer par trier chaque face depuis le point de vue. A partir d'un point de vue V, une cellule c1 d'un MNT est dite en avant d'une autre cellule c2 si un rayon issu de V intersecte c1 avant c2. Un MNT sera dit trié d'avant en arrière si on peut déterminer pour chacune de ses cellules si elle se situe en avant ou en arrière de n'importe quelle autre cellule. Malheureusement, du fait de leur structure irrégulière, tous les TIN ne sont pas ainsi ordonnables. Cependant, si on se restreint aux triangulations de Delaunay, un TIN peut être ordonnable [DE

FLORIANI 1991] et tout TIN non ordonnable peut être rendu ordonnable par décomposition de certains de ses triangles.

Cette approche « front-to-back » exploite le fait qu'un triangle ne peut être caché que par des triangles se trouvant devant lui. A chaque étape, un horizon courant est utilisé pour déterminer la visibilité des nouveaux triangles. Cette méthode fut développée et implémentée [DE FLORIANI 1989] puis [DE FLORIANI 1994]. D'autres nombreux auteurs ont développé des algorithmes dérivant de ce concept de base. On peut notamment citer parmi eux Reif et Sen [REIF 1988], Katz [KATZ 1992], Overmars et Sharir [OVERMARS 1992] ou d'un point de vue plus théorique Preparata et Vitter [PREPARATA 1992] ou Edelsbrunner [EDELSBRUNNER 1989].

En préférant des traitements moins précis mais plus légers et toujours en s'inspirant de cette approche « front-to-back », Lee [LEE 1991] a développé un algorithme discret pour lequel un triangle est visible si ses 3 cotés sont totalement visibles.

3.3.1.4.1.2 Détermination des horizons

L'horizon peut être vu comme un sous-produit de la carte de visibilité, ou bien peut être calculé directement comme l'enveloppe de l'ensemble des segments constitués par la projection des côtés des triangles visibles du MNT sur une image plane. Pour obtenir cette enveloppe, une façon de procéder est d'utiliser une approche incrémentale [DE FLORIANI 1998].

3.3.1.4.2 Les algorithmes applicables aux RSG

Il s'agit principalement d'algorithmes de détermination discrète de structures de visibilité puisque les algorithmes de détermination continue sont trop lourds pour un RSG.
La méthode de Shapira [SHAPIRA 1990] consiste à tracer un rayon depuis V vers tous les autres points P du MNT ; on parcourt ensuite le rayon de V vers P jusqu'à ce qu'une intersection entre le rayon et une arête du terrain soit trouvée.

Cependant, cette méthode engendre des calculs inutiles. Pour éviter ces derniers, [DONNAY 1992] propose un algorithme fondé sur la recherche des intersections potentielles entre le relief et tous les rayons menés depuis V vers les cellules situées en bordure de la zone couverte par la carte de visibilité.

Aujourd'hui, les algorithmes appliqués aux RSG différent les uns des autres essentiellement par leurs techniques d'interpolation à partir des sommets de la grille pour déterminer les intersections entre les rayons et les arêtes des cellules. Dans un état de l'art, Fisher répertorie en 1993 trois tendances que résume la Figure 50 [FISHER 1993]. Sur cette figure, le nombre de points utilisés pour interpoler puis rechercher les intersections entre les rayons issus du point de vue et les cellules augmente de (a) vers (c). Une dernière procédure consiste à attribuer une élévation constante pour chaque cellule de la grille.

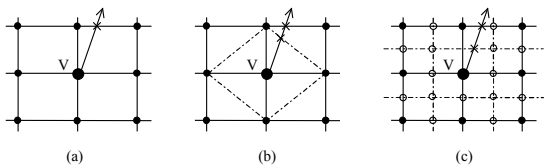


Figure 50 : Différentes possibilités d'interpolation des intersections entre les rayons

Fisher a ainsi comparé les cartes de visibilité sur deux régions tests obtenues à partir de logiciels (dont Arc/info visibility et GRASS) implémentant différentes méthodes d'interpolation et de calcul de hauteur. Les résultats montrent que le nombre de cellules composant les cartes de visibilités peut varier du simple au double suivant le logiciel utilisé. Malheureusement, les logiciels commerciaux communiquent rarement sur les algorithmes d'intervisibilité qu'ils utilisent.

3.3.1.5 Algorithmes applicables sur les modèles «multi-résolution»

Parce que les erreurs en élévation près du point de vue sont amplifiées proportionnellement à la distance, il est important de représenter la topographie entourant le point de vue avec la plus grande précision possible, alors qu'une résolution moindre suffit pour le reste de la surface.
Pour introduire ce concept dans les algorithmes de visibilité, la principale difficulté réside dans le maintien d'une continuité entre les fragments de surfaces décrits par le modèle « multi-résolution ».
Puppo [PUPPO 1996] traite le problème d'extraction d'un MNT « multi-résolution » en s'appuyant sur un graphe, dit d'interférences, décrivant les relations entre les fragments de MNT de résolutions différentes. Un parcours transversal du graphe identifie le nombre minimal de fragments à extraire pour avoir la résolution souhaitée.
Les principes de résolution des problèmes de visibilité à l'aide de MNT « multi-résolution » s'apparentent à l'utilisation d'un index spatial lors de requêtes plus traditionnelles. Ensuite, une fois le MNT reconstitué, il suffit de lui appliquer un quelconque algorithme de calcul de visibilité

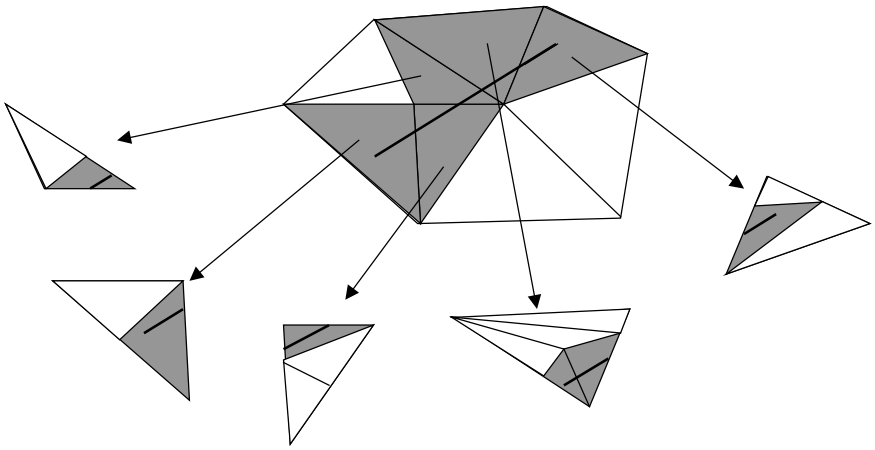


Figure 51 : Décomposition d'un segment dans un modèle "multi-résolution" [DE FLORIANI 1998].

La mise à jour des structures de visibilité pour un changement de résolution locale est également un problème de visibilité récurrent sur des MNT « multi-résolution ». Cela arrive souvent, par exemple, dans le cas de simulation de vol où l'attention se focalise rapidement d'un endroit à un autre et le maximum de résolution est alors requis pour divers endroits à différents moments. Généralement, on procède à un nouveau calcul de visibilité où on applique des algorithmes dynamiques qui possèdent une structure permettant de retrouver rapidement les zones du MNT affectées par une mise à jour [DE FLORIANI 1998].

3.3.2 Notion d'intervisibilité dans un environnement 3D

Jusqu'à présent, nous avons traité de structures et calculs de visibilité applicables aux modèles numériques de terrain. Ces techniques peuvent être applicables aux modèles numériques d'élévations (MNE), c'est-à-dire aux modélisations 2,5 D. En revanche pour procéder à des tests d'intervisibilité sur des objets véritablement 3D du sur-sol, il convient d'utiliser d'autres approches. Nous détaillons ici une approche possible pour traiter des problèmes d'intervisibilité sur objets 3D dans un système d'information géographique.

3.3.2.1 Définitions

La caractéristique de visibilité entre deux objets est à la fois propre à la géométrie de ceux-ci et sujette à interprétation ou approximation. Nous définissons ici quelques notions utilisées au cours du chapitre.

Fenêtre de visibilité :

Nous appellerons « fenêtre de visibilité » la forme 3D englobant l'ensemble des segments reliant tous les points composant l'objet observateur à tous les points composant l'objet observé. Ainsi la forme géométrique de la fenêtre de visibilité sera :

- Entre 2 points : un segment
- Entre un point et un segment : une face triangulaire
- Entre un point et une courbe 3D : une développée
- Un volume complexe dans tous les autres cas

La fenêtre de visibilité entre un observateur O et une cible C pourra être notée $\varphi(O,C)$

Etat d'intervisibilité :

Nous appellerons « état d'intervisibilité » l'approche logique à trois états suivante (Figure 52):

- L'état d'intervisibilité entre un observateur et une cible est défini comme totalement visible si aucun objet du sol ou du sur-sol n'intersecte la fenêtre de visibilité entre l'observateur et la cible.

- L'état d'intervisibilité entre un observateur et une cible est défini comme totalement invisible si la fenêtre de visibilité entre l'observateur et la cible est totalement obstruée par un ou plusieurs objets du sol ou sur-sol.

- L'état d'intervisibilité entre un observateur et une cible est défini comme partiellement visible si la fenêtre de visibilité entre l'observateur et la cible est partiellement obstruée par un ou plusieurs objets du sol ou sur-sol.

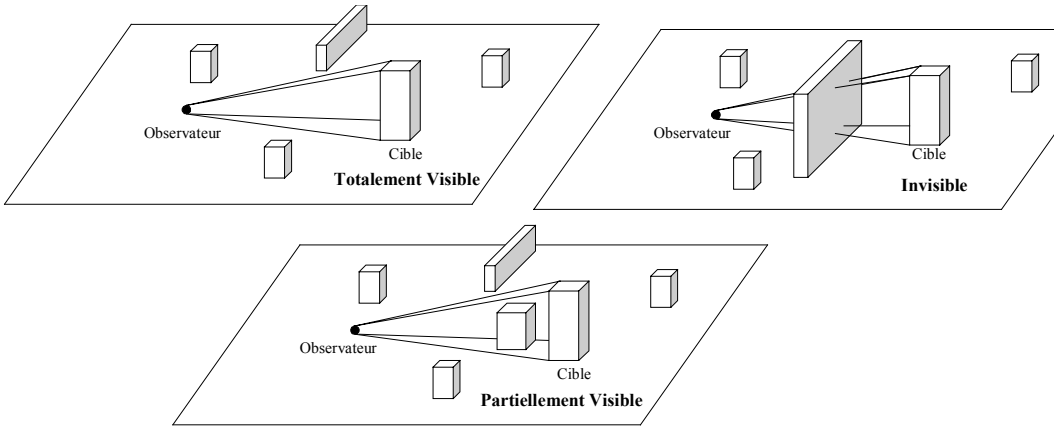


Figure 52 : Etat d'intervisibilité

3.3.2.2 Primitives géométriques et relations d'intervisibilité

La notion d'intervisibilité est ici liée à la notion de propagation du signal visuel. Nous assimilons ce signal à un rayon lumineux rectiligne partant de l'observateur O en direction de la cible C. Aussi avons-nous une certaine symétrie entre observateur et observé puisque si C est visible depuis O alors C voit O. Voyons maintenant les caractéristiques des fenêtres de visibilité en fonction des primitives géométriques impliquées.

3.3.2.2.1 Relation d'intervisibilité entre 2 objets ponctuels

3.3.2.2.1.1 Présentation

C'est la relation la plus exploitée dans les applications SIG traitant l'intervisibilité, c'est aussi le cas le plus simple. La fenêtre de visibilité est un segment (Figure 53).

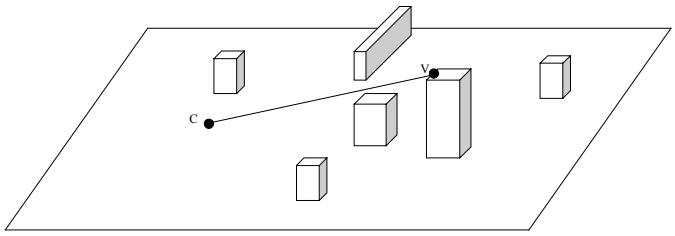


Figure 53 : Intervisibilité point à point

3.3.2.2.1.2 Algorithmes

L'algorithme à mettre en œuvre pour résoudre ce cas est simple. Il se contente de vérifier que le segment [VC] n'intersecte aucun objet de la scène. On utilisera un index spatial pour optimiser cette recherche d'intersection. Le résultat retourné est exact.

3.3.2.2.1.3 Exemples d'applications

De tels algorithmes peuvent être utilisés afin de s'assurer qu'un observateur ou une sentinelle est idéalement placée pour surveiller une cible ponctuelle donnée. Dans le domaine des télécommunications, ces algorithmes sont très utiles pour placer les antennes relais de nos téléphones portables. Que ce soit pour des applications civiles ou militaires, les calculs point à point sont sans doute les plus utilisés parmi les calculs d'intervisibilité.

3.3.2.2.2 Relation d'intervisibilité entre un objet ponctuel et un segment

3.3.2.2.2.1 Présentation

La fenêtre de visibilité est ici un triangle ayant pour base le segment et pour sommet l'objet ponctuel (Figure 54).

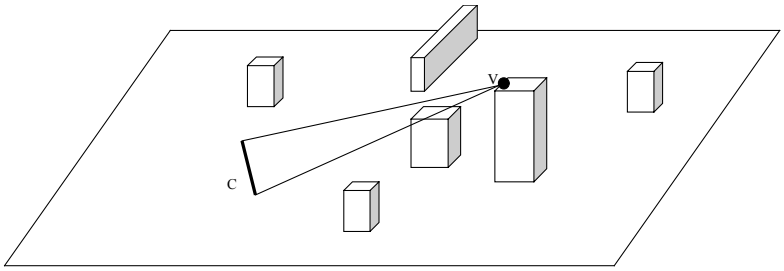


Figure 54 : Intervisibilité entre un objet ponctuel et un segment

3.3.2.2.2.2 Algorithmes

Visibilité totale :

Il s'agit d'un cas simple où il suffit de rechercher une éventuelle intersection entre le triangle représentant la fenêtre de visibilité et un objet O_i de la scène.

Visibilité partielle :

La recherche de visibilité partielle peut se faire de manière continue ou de manière discrète.

- Dans le premier cas on recherche l'intersection entre l'objet O_i et le triangle représentant la fenêtre de visibilité $\varphi(V,C)$, puis on détermine si cette intersection masque partiellement ou totalement le segment C. Le résultat retourné est exact.

- L'approche discrète consiste à discrétiser le segment C en un ensemble de points pour lesquels on effectue un test de visibilité de type point à point (voir paragraphe précédent). Si au moins un segment n'est pas intersecté, le segment C est partiellement visible depuis V. Le résultat retourné n'est pas exact et sa validité dépend du pas de discrétisation choisi pour le segment C.

3.3.2.2.2.3 Exemples d'applications

L'intervisibilité entre un objet ponctuel et un segment est utile pour la surveillance de chemins d'accès depuis un point fixe, ou encore pour la surveillance d'un point précis le long d'un chemin de ronde. Les organismes régissant les systèmes de communication au sein des transports publics urbains de surface sont capables, avec ce type de calcul d'intervisibilité, de déterminer l'endroit où placer les relais de communication pour que les bus puissent avoir une liaison radio avec leurs postes de contrôles tout au long de leurs trajets.

3.3.2.2.2.4 Extension à la polyligne ou à la courbe

La recherche des relations d'intervisibilité entre un objet ponctuel et un segment s'étend facilement au cas de la polyligne puisque celle ci peut être considérée comme une suite de segments mis bout à bout. Il suffit alors de réitérer n fois l'algorithme décrit précédemment pour une polyligne à n segments.

Le cas d'une ligne courbe est en revanche un peu plus complexe et une méthode discrète semblable à 3.3.2.2.2.2 est sans doute préférable à une méthode continue s'appuyant sur une recherche d'intersection pour une surface complexe de type développée.

3.3.2.2.3 Relations d'intervisibilité entre un objet ponctuel et un objet surfacique

3.3.2.2.3.1 Présentation

Dans ce cas la fenêtre de visibilité prend une forme pseudo conique ou pseudo pyramidale avec comme base l'objet surfacique et comme sommet principal l'objet ponctuel (Figure 55).

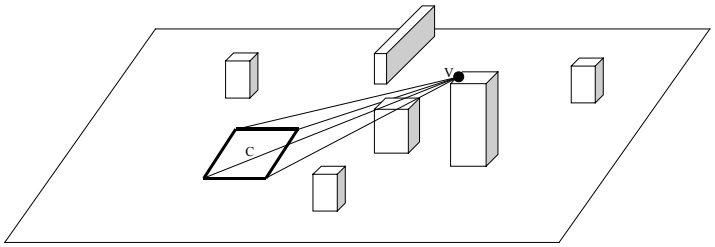


Figure 55 : Intervisibilité entre un objet ponctuel et un objet surfacique

3.3.2.2.3.2 Algorithmes

Les opérations de calculs d'intersection sur les volumes peuvent s'avérer extrêmement complexes et mettre en œuvre des algorithmes génériques permettant d'obtenir un résultat exact sur les calculs d'intervisibilité entre volumes quelconques dépasse le cadre de nos travaux.
Nous préconisons la recherche de solutions approchées qui offrent un bon compromis entre la simplicité de l'algorithme, le temps de calcul et la qualité du résultat retourné. L'algorithmique employée en infographie et synthèse d'image est tout a fait adaptée à notre besoin.

3.3.2.2.3.3 Exemples d'applications

Chez les militaires, ce type de calcul sert, entre autres, à placer des sentinelles afin de surveiller une zone spécifique. Dans le domaine civil se sont surtout les applications touristiques qui sont concernées par ce type d'intervisibilité, avec notamment la sélection de points de vue offrant les meilleurs panoramas.

3.3.2.2.4 Relations d'intervisibilité entre deux objets linéaires

3.3.2.2.4.1 Présentation

Si les deux segments sont dans le même plan, la fenêtre de visibilité est un quadrilatère sinon elle prend une forme tétraédrique (Figure 56).

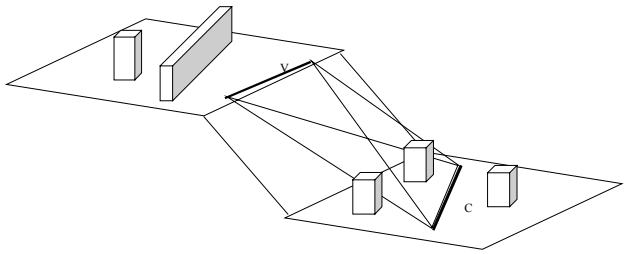


Figure 56 : Intervisibilité entre deux segments

3.3.2.2.4.2 Algorithmes

Si les deux segments appartiennent au même plan, on se retrouve dans un cas proche du 3.3.2.2.2.2 avec non plus des intersections entre volumes et triangles mais entre volumes et quadrilatères. Les algorithmes discrets de calculs de visibilité partielle auront à discrétiser non plus un mais deux segments.

Si les deux segments n'appartiennent pas au même plan, on se retrouve là dans un cas d'intersection entre volumes plus complexes. Les algorithmes utilisés pour résoudre ce cas seront donc du même ordre que ceux abordés en 3.3.2.2.3.2.

3.3.2.2.4.3 Exemples d'applications

Les applications sont ici moins nombreuses et moins évidentes. Ce sont des applications impliquant des mobiles et des contrôles de visibilité le long de leurs trajectoires. On peut par exemple penser à la mise en place de systèmes de surveillance et de sécurité lors de manifestations sportives, culturelles ou sociales.

3.3.2.2.5 Relations d'intervisibilité entre un objet linéaire et un objet surfacique

3.3.2.2.5.1 Présentation

La fenêtre de visibilité est un volume plus ou moins complexe en fonction de la forme et de l'orientation de l'observateur et de l'observé (Figure 57).

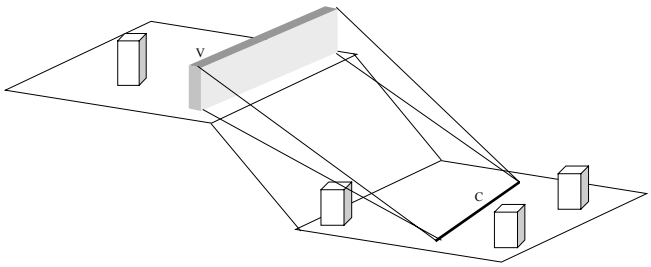


Figure 57 : Intervisibilité entre un objet surfacique et un segment

3.3.2.2.5.2 Algorithmes

Ici nous nous retrouvons dans un cas d'intersection entre volume plus ou moins complexes en fonction des orientations relatives de la surface et du segment. Les algorithmes utilisés seront donc proches de ceux abordés en 3.3.2.2.3.2.

3.3.2.2.5.3 Exemples d'applications

Ici une application envisageable consisterait à estimer les zones couvertes par le trajet d'un avion pour une campagne de photographies aériennes sur un secteur donné.

3.3.2.2.6 Relations d'intervisibilité entre 2 objets surfaciques

3.3.2.2.6.1 Présentation

La fenêtre de visibilité prend à nouveau la forme d'un volume complexe.

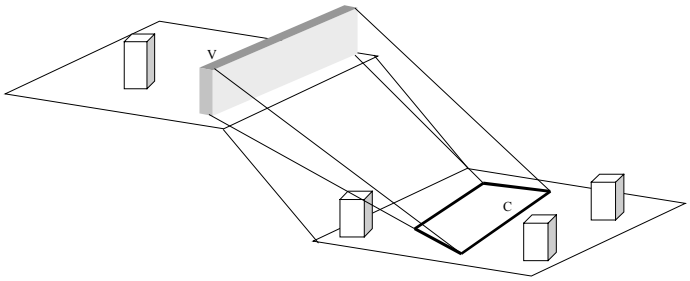


Figure 58 : Intervisibilité entre deux objets surfaciques

3.3.2.2.6.2 Algorithmes

Une fois encore il s'agit là d'intersections entre volumes complexes. Les algorithmes utilisés seront donc à nouveau très proches de ceux abordés en 3.3.2.2.3.2.

3.3.2.2.6.3 Exemples d'applications

Certaines grandes manifestations font appel à d'immenses écrans vidéos pour les spectateurs un peu trop éloignés de la scène principale. Une application envisageable de calculs d'intervisibilité entre surfaces est celle du positionnement d'un tel écran géant afin qu'il soit visible pour la plus grande partie de la foule.

3.3.2.2.7 Relations d'intervisibilité impliquant un ou plusieurs objets volumiques

3.3.2.2.7.1 Présentation

La définition de l'état de visibilité sur un objet volumique est problématique. Quel sens donner à l'affirmation « l'observateur voit totalement le bâtiment » ?

Il est évident que quelle que soit sa position, un observateur ne peut pas voir chaque mur composant le bâtiment. L'affirmation précédente doit plutôt être interprétée comme « l'observateur voit tous les murs du bâtiments qu'il pourrait voir si aucun obstacle visuel ne se dressait entre lui et le bâtiment ».

Cette dernière affirmation évite sans doute des problèmes d'interprétation sur des questions de visibilité globale, elle n'en reste pas moins floue quant il s'agit de visibilité partielle. Lorsqu'une application a besoin de connaître la proportion visible d'un objet, on parle de « pourcentage de visibilité ». Cette notion de pourcentage de visibilité est intuitive pour des objets de dimensions inférieures ou égales à 2, mais l'est beaucoup moins pour des objets de dimension 3, c'est à dire des volumes. Suivant les domaines d'application, un bâtiment pourra être dit visible à 50 % tantôt si 2 de ses 4 murs sont entièrement visibles, tantôt si 3 de ses 6 faces (sol et toit compris) sont entièrement visibles, ou encore si l'observateur voit effectivement la moitié de la surface du bâtiment qu'il pourrait potentiellement voir dans le cas ou aucun obstacle ne se dresserait entre lui et le bâtiment. Il est difficile de trancher en faveur de l'une ou l'autre de ces définitions, chacune pouvant satisfaire sa propre communauté d'utilisateurs.

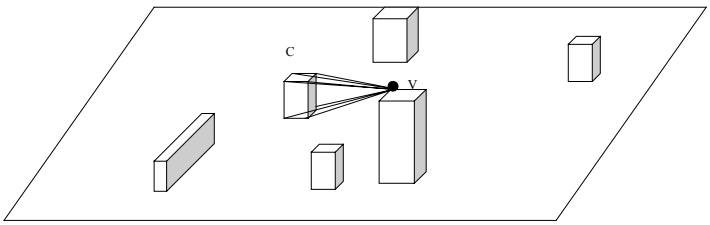


Figure 59 : Intervisibilité impliquant des volumes

3.3.2.2.7.2 Algorithmes

De manière générale, les recherches d'intervisibilité sur les volumes passent par des recherches d'intervisibilité sur les faces composant ce volume. En fonction de la position de l'observateur, une première sélection des faces potentiellement visibles doit être faite, puis les résultats de tests de visibilités sur ces faces doivent être combinés pour déterminer l'état de visibilité du volume. Les algorithmes permettant de déterminer l'état de visibilité sur un volume reprennent donc en grande partie les algorithmes de calculs d'intervisibilité pour les éléments surfaciques.

3.3.2.2.7.3 Exemples d'applications

La surveillance de bâtiments et le placement de sentinelles ou de vigiles pour en surveiller toutes les issues sont pour les militaires et les garants de la sécurité publique des applications intéressantes de calculs de visibilité sur entités volumiques.

3.3.3 Conclusion sur l'intervisibilité

Les calculs d'intervisibilité font partie des apports majeurs d'un système d'information géographique tridimensionnel. Nous proposons d'aborder l'intervisibilité sous deux angles :

- L'intervisibilité partielle où l'objet est déclaré visible si une partie de celui-ci est effectivement vue par l'observateur.

- L'intervisibilité totale où l'objet est déclaré totalement visible s'il est entièrement vu par l'observateur, c'est-à-dire si aucun obstacle ne traverse la fenêtre de visibilité formée par l'observateur et la cible.

Les algorithmes de calculs de visibilité partielle sont de nature discrète. La cible est assimilée à un ensemble de points dont on teste tour à tour la visibilité depuis l'observateur par des lancers de rayons. Les algorithmes de calculs de visibilité totale peuvent s'appuyer sur la même technique de discrétisation ou sur une stratégie continue. Cette dernière réalise des intersections entre volumes, et non plus de simples lancers de rayons.

Intégré à notre prototype de SIG 3D, ces algorithmes permettront d'effectuer des tests de visibilité directs entre deux objets. Une autre application envisageable est de dresser des cartes de visibilité en déterminant l'état de visibilité de chaque objet de la zone concernée par rapport à un observateur donné.

Les cartes de visibilité peuvent, par exemple, intervenir comme paramètres en entrée des algorithmes de recherche de trajectoire que nous allons maintenant étudier.

3.4 Recherche de trajectoire

Dans lequel il est question de retrouver son chemin en associant l'univers géographique à un graphe.

La détermination du chemin optimal pour rejoindre deux points est une fonctionnalité couramment présentée par les systèmes d'information géographique commerciaux. L'implémentation de cette fonctionnalité est généralement peu évoluée, voire simpliste, à l'exception de quelques produits proposant des algorithmes de recherche de trajectoire plus sophistiqués. Les principes algorithmiques s'appuient sur la recherche de l'accumulation minimale d'une valeur de coût représentative du type souhaité de trajectoire. Il peut s'agir de la trajectoire la plus courte, la plus rapide, la plus économique ou encore la moins risquée.

Lorsque le mobile pour lequel on cherche à déterminer une trajectoire optimale est contraint à un réseau linéaire, comme un réseau routier ou un réseau de couloirs aériens, une approche classique est de modéliser ce réseau par un graphe pondéré auquel on applique des algorithmes classiques de détermination de chemin optimal [STEFANKIS 1995].

L'étude des trajectoires dites « libres », c'est à dire sans que le déplacement du mobile ne soit assujetti à aucun réseau, est un problème autrement plus complexe qui trouve nombre d'applications aussi bien dans le civil que dans le militaire. Les paramètres à prendre en compte pour ces calculs de trajectoire sont nombreux. Le mobile possède entre autres une vitesse moyenne et une vitesse de pointe, il se déplace plus facilement sur certaines surfaces, il possède son propre gabarit. Le relief du MNT est également influant sur le choix d'une trajectoire optimale et des contraintes supplémentaires, telles que des contraintes de proximité ou de visibilité, peuvent être ajoutées. Enfin, puisque nous sommes dans un contexte 3D, nous devons être capables de rechercher des trajectoires « terrestres » plaquées au sol tout comme des trajectoires aériennes.

Après quelques rappels sur les graphes indispensables à la compréhension des algorithmes de recherche de trajectoire, nous aborderons les principaux algorithmes de recherche de trajectoire et nous proposerons une méthodologie pour reconstruire, à partir d'une scène 3D, un graphe sur lequel seront applicables ces algorithmes. La recherche de trajectoire « libre » s'appuie en effet sur la reconstruction d'un tel graphe. Pour faciliter la compréhension de notre discours, nous commencerons par quelques rappels sur la théorie de graphes.

3.4.1 Notion de graphe

Un graphe orienté G= (X,U) peut être considéré comme :

a) un ensemble fini $X=\{x_1,x_2\ldots,x_n\}$ d'éléments appelés sommets. Si X possède n éléments distincts, le graphe sera dit d'ordre n.
b) un ensemble fini $U=\{u_1,u_2\ldots,u_n\}$ dont les éléments sont des couples ordonnés de sommets appelés arcs.

Un graphe est un triplet G=(X,U,p) où p est une valuation des arêtes de G (par des entiers, par exemple).

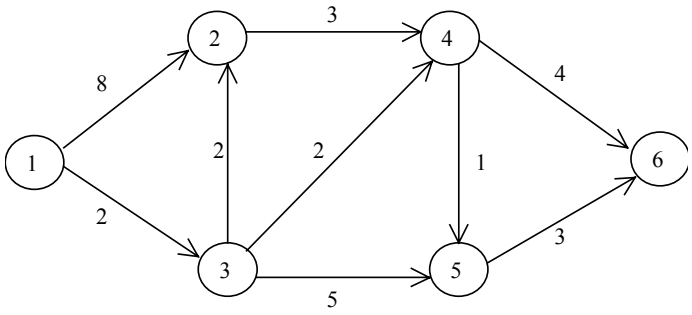


Figure 60 : Notion de graphe orienté et valué

D'autre part voici quelques définitions relatives aux arcs et sommets d'un graphe orienté [DROESBEKE 1987] :

- *Extrémités d'un arc* : un arc u = (x,y) possède une extrémité initiale x et une extrémité finale y.

- *Boucle* : arc (x,x) dont les extrémités coïncident.

- *Arcs adjacents* : arcs ayant au moins une extrémité en commun.

- *Successeur* : on dit que y est un successeur de x s'il existe un arc ayant x comme extrémité initiale et y comme extrémité finale.

- *Prédécesseu*r : on dit que y est un prédécesseur de x s'il existe un arc ayant y comme extrémité initiale et x comme extrémité finale.

- *Sommet adjacent* : un sommet x est adjacent à un sommet y s'il est prédécesseur ou successeur de y.

- *Degré* : soit u = (x,y) un arc (qui n'est pas une boucle) de G. Il est dit incident à x vers l'extérieur et incident à y vers l'intérieur. Le nombre d'arcs incidents à x vers l'extérieur s'appelle le demi-degré extérieur et par analogie le demi-degré intérieur représente le nombre d'arcs incidents à x vers l'intérieur. Le degré de x est la somme des demi-degrés intérieurs et extérieurs.

- Un *p-graphe* est un graphe dans lequel il y a au maximum p arcs de la forme (x,y) entre deux sommets x et y.

- *Sous-graphe* de G=(X,U) : étant donnée un sous-ensemble $A \subset X$ de sommets de G, le sous-graphe G_A engendré par A est défini par $G_A = \{A, U_A\}$,où U_A est l'ensemble des arcs de U ayant leurs extrémités en A.

- *Graphe partiel* de G = (X,U) : étant donné un sous-ensemble d'arc $V \subset C$, le graphe partiel engendré par V est défini par (X,V).

- *Chaîne* : une chaîne de cardinalité q est une séquence de q arcs de U $\sigma=\{u_1, \ldots,u_q\}$ telle que chaque arc u_k de σ ($1<k<q$) ait une extrémité en commun avec l'arc précédent u_{k-1} et l'autre extrémité en commun avec l'arc suivant u_{k+1}, les extrémités x de u_1 et y de u_q qui ne sont

adjacents à aucun autre sommet dans σ sont les extrémités de la chaîne. D'autre part, la chaîne est dit élémentaire si en la parcourant on ne rencontre jamais deux fois le même sommet.

- *Chemin* : un chemin est une chaîne dont tous les arcs sont orientés dans le même sens.
- *Cycle* : un cycle est une chaîne dont les extrémités coïncident.
- *Circuit* : un circuit est un chemin dont les extrémités coïncident.
- *Connexe* : Un graphe est dit faiblement connexe si pour tout couple de sommet x_i, $x_j \in X$ ($i \neq j$), il existe une chaîne rejoignant x_i à x_j. Un graphe est dit fortement connexe si pour tout couple de sommet x_i, $x_j \in X$ ($i \neq j$), il existe un chemin allant de x_i à x_j.
- *Complet* : Un graphe G = (X,U) est dit complet si, quelque soit $x_i, x_j \in X$ avec $x_i \neq x_j$, $(x_i,x_j) \in U$.
- *Transitif* : Un graphe G = (X,U) est dit transitif si $(x_i,x_j) \in U$ et $(x_i,x_k) \in U$ implique $(x_j,x_k) \in U$, quelque soit $x_i, x_j, x_k \in X$.

3.4.2 Chemin de plus faible coût de déplacement

3.4.2.1 Formulation du problème

Les calculs de trajectoires optimales assujetties à un réseau peuvent facilement être assimilés au calcul de chemin minimal dans un 1-graphe orienté G= (X,U) sans boucle et comportant n sommets. A tout arc (x_i,x_j) de U on associe un nombre réel ω_{ij} représentatif de la contrainte de déplacement et appelé coût de déplacement de l'arc. Le coût de déplacement cumulé pour un chemin μ quelconque, noté c(μ), est alors défini comme la somme des coûts de déplacement des arcs qui le composent :

$$c(\mu) = \sum_{(x_i,x_j)\in\mu} \omega_{ij}$$

Un chemin joignant deux sommets du graphe est dit optimal s'il minimise le coût de déplacement c(μ) dans l'ensemble de tous les chemins μ joignant ces deux sommets.

Classiquement, on distingue deux grandes familles d'algorithmes de recherche de chemin optimal [BURTON 1993]. La première d'entre elle, méthode dite du « label-correcting », s'applique sur tout type de graphes pondérés par des valeurs positives ou négatives, alors que la méthode dite de « label-setting » ne s'applique qu'aux graphes pondérés avec des valeurs positives.

Les algorithmes qui suivent s'appliquent à un graphe G=(X,U) ou X représente l'ensemble des sommets et U l'ensemble des arcs, associé à ω un ensemble de coûts ω_{ij} non négatifs associés aux arcs. C(i) représente le coût de déplacement cumulé courant pour rejoindre le sommet i, pred(i) stocke le sommet précédent i sur le chemin du plus faible coût, « src » représente le sommet source et « dest » le sommet destination. D'autre part, on ne considère ici que les graphes G ne comportant pas de circuits négatifs pour s'assurer que le problème de recherche du chemin optimal admet une solution finie.

3.4.2.2 Principe du « label-correcting »

Le premier algorithme de ce type fut proposé par Ford [FORD 1956] puis détaillé par Bellman [BELLMAN 1958] et [MOORE 1959]. L'algorithme affecte des marques (ou labels) temporaires aux sommets en fonction du coût nécessaire pour rejoindre ces sommets depuis le sommet source. L'algorithme ne se contente pas de retourner le chemin le moins coûteux entre un sommet source et un sommet destination, il calcule les chemins les moins coûteux entre le sommet source et tous les sommets du graphe. La complexité de l'algorithme (Figure 61) est de l'ordre O(nm) ou $O(n^3)$, avec n représentant le nombre de sommets du graphe et m le nombre d'arcs du graphe.

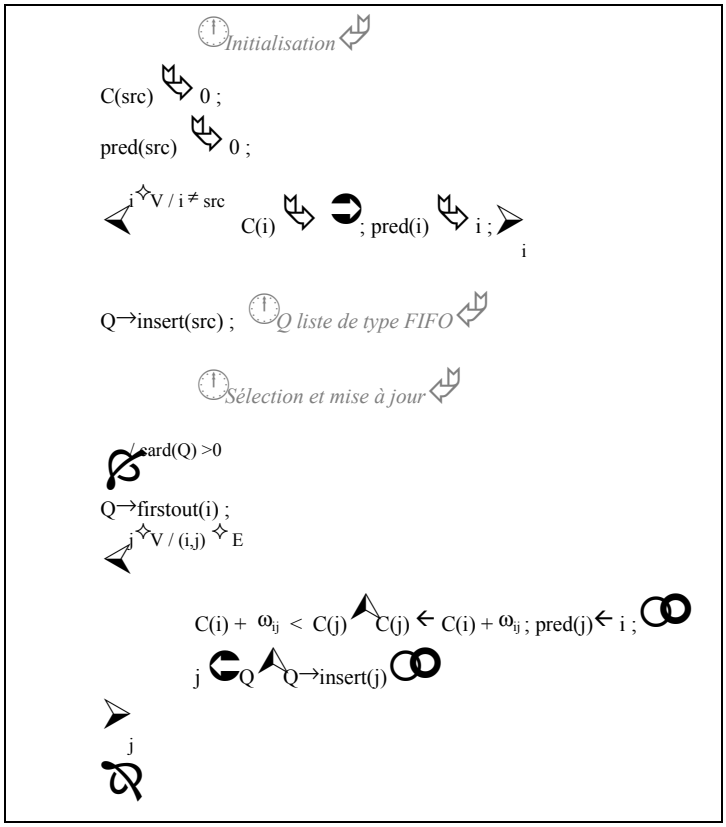


Figure 61 : Algorithme de Ford-Bellman-Moore

Cet algorithme a pour avantage de manipuler des arcs aux pondérations positives ou négatives ω_{ij}. En revanche, pour calculer le chemin de moindre coût entre un sommet source et un seul sommet destination, il est indispensable de faire tourner l'algorithme jusqu'à son terme, c'est à dire jusqu'à ce que tous les sommets du graphe soient traités. Il existe cependant quelques variantes de l'algorithme de Ford-Bellman-Moore permettant de s'adapter à des cas particuliers d'utilisation, notamment lorsque le graphe G peut être partitionné en plusieurs sous-graphes.

3.4.2.3 Principe du « label-setting »

Les premiers algorithmes de label-setting ont été publiés par Dijkstra [DIJKSTRA 1959] et Moore [MOORE 1959]. Depuis, les algorithmes de label-setting découlent tous plus ou moins de l'implémentation initiale de Dijkstra. Ce type d'algorithme fonctionne uniquement à partir de pondérations positives sur les arcs mais cette contrainte ne se révèle pas véritablement handicapante dans la pratique. La caractéristique principale de l'algorithme est qu'il choisit le sommet au plus faible

coût de déplacement cumulé comme prochain sommet à traiter. Cela permet de ne traiter qu'une et une seule fois chaque sommet et donc d'obtenir un chemin minimum entre la source et un des sommets à chaque itération. Pour l'ensemble des sommets le calcul prendra au pire n-1 itérations. Plus intéressant encore, le calcul pourra être arrêté dès que le sommet destination est le prochain sommet à traiter.

Le point critique de l'algorithme réside d'ailleurs dans cette recherche du sommet au coût minimal devant être traité à l'itération suivante. La Figure 62 présente une version de base de l'algorithme tournant en $O(n^2)$; cet algorithme peut être optimisé en accélérant la recherche du prochain sommet grâce à un tri des sommets à l'intérieur de listes [POLLITT 1995] :

- Une première méthode consiste à ranger les sommets de même coût cumulé dans une liste commune. Il y aura donc autant de listes que de coûts cumulés différents, ces listes sont elles mêmes rangées dans un tableau. Cette variante est connue sous le nom de « Integral Bucket Structure ». Si C est le coût maximal pour rejoindre 2 sommets du graphe, la complexité de l'algorithme de Dijkstra passe à O(m+n*C) avec cette méthode.

- Une seconde méthode limite le nombre de listes en rangeant les sommets dans des listes caractérisées par une fourchette de coût et non plus un coût exact. On parle alors de « Approximate Bucket Structure ». Notons que le nombre de listes utilisées sera fonction du type de graphe à traiter. Si B est le nombre de listes, la complexité de l'algorithme de Dijkstra passe à O(m*B+n(B+C/B)) avec cette méthode.

- Enfin, une dernière technique regroupe les deux précédentes pour ranger les sommets sur deux niveaux : suivant la valeur de leur coût de déplacement cumulé, les sommets sont classés dans des listes bas niveaux de type « integral bucket » ou des listes de hauts niveaux de type « approximate bucket ». Le sommet à traiter est toujours choisi dans la liste de bas niveau regroupant les sommets ayant le plus faible coût cumulé. Les listes de haut niveau approvisionnent les listes de bas niveau lorsque ces dernières se retrouvent vides. Si B est le nombre de listes de bas niveaux et C le coût maximal pour rejoindre 2 points du graphe, la complexité de l'algorithme de Dijkstra passe à O(m+n(B+C/B)) avec cette méthode.

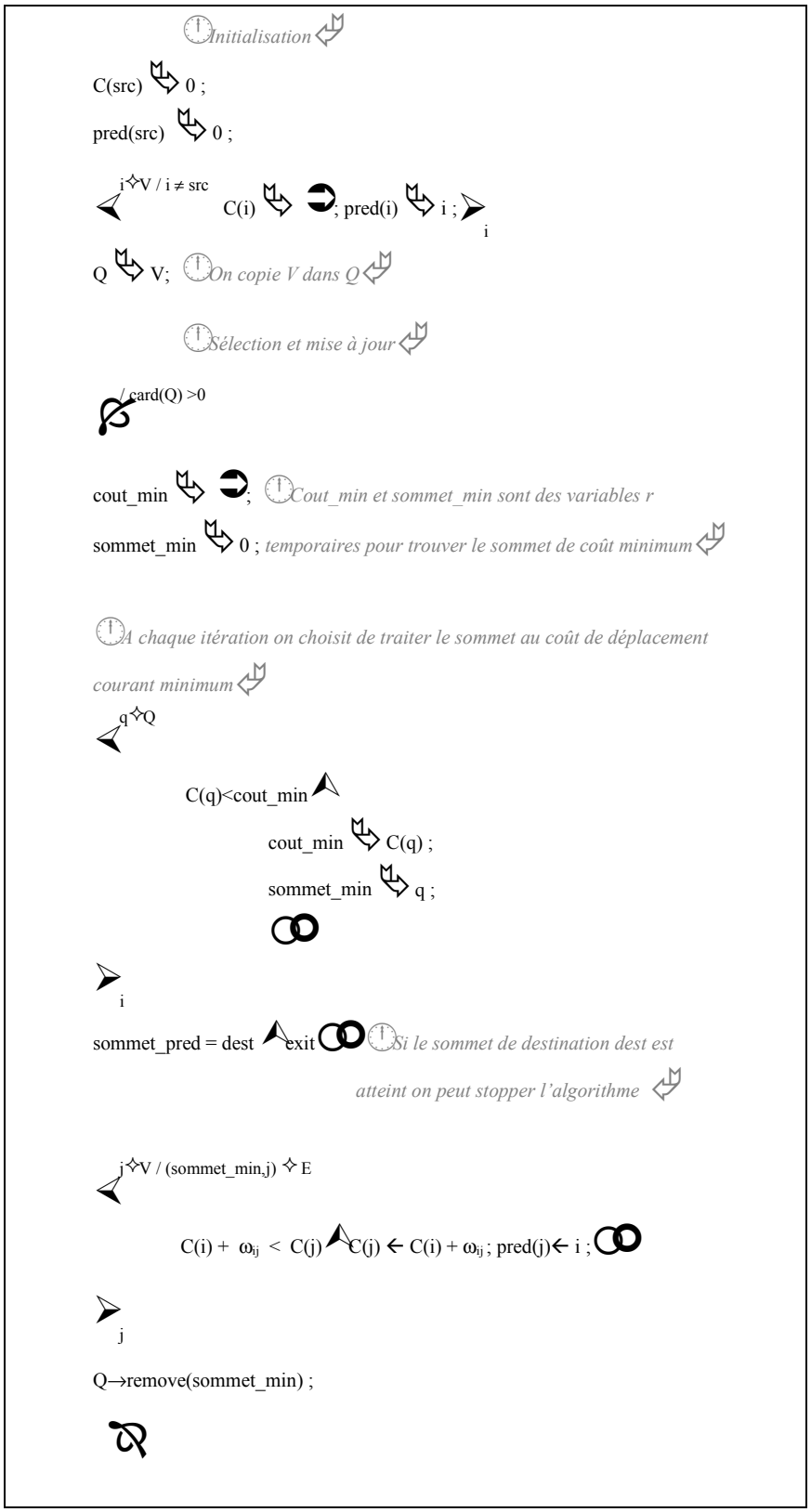


Figure 62 : Algorithme de Dijkstra

3.4.3 Recherche de trajectoires libres

La recherche d'une trajectoire libre s'effectue en cinq étapes [STEFANKIS 1995]. Elle débute par l'établissement d'un graphe connectant un ensemble fini de positions potentielles du mobile avant de procéder au calcul effectif du chemin optimal.

3.4.3.1 Détermination d'un nombre fini de situations spatiales.

Pour décrire l'espace et ainsi limiter à un nombre fini les positions ou situations spatiales que peut adopter un objet, une solution simple est de découper l'espace en une grille régulière de type RSG (Regular Square Grid) (voir 3.3.1.2). Ce type de grille est utile lorsque l'espace modélisé est de nature très hétérogène. Dans le cas d'une recherche de trajectoire d'avion où le mobile se déplace dans un milieu relativement homogène, l'utilisation d'autres outils d'indexation spatiale de type Octree ou R-Tree pourra s'avérer plus performante [SOYEZ 1994] (voir 3.2.3).

3.4.3.2 Détermination du graphe

Il est facile d'associer à une grille RSG un graphe ayant pour sommets les centres des cellules de la RSG. L'ensemble des arcs est constitué des couples de cellules adjacentes. Le graphe de mouvement ainsi formé relie tous les centres de cellules adjacentes par des arcs, constituant ainsi un support pour le déplacement du mobile (Figure 63).

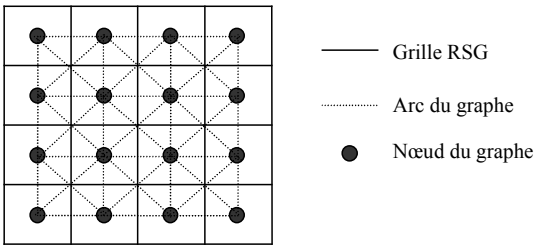


Figure 63 : Construction d'un graphe à partir d'un RSG

3.4.3.3 Pondération des nœuds du graphes

Chaque cellule de la grille, et donc chaque nœud du graphe, reçoit ensuite une valeur qui sera la résultante d'une combinaison de paramètres divers et variés prenant notamment en compte la nature du terrain associé à chaque cellule, l'encombrement par des objets se trouvant dans les cellules, le dénivelé du terrain, le type du mobile, la vitesse du mobile ou le gabarit du mobile (Figure 64). Les algorithmes calculant les valeurs ou coefficients de pondération peuvent aussi être affectés par des contraintes de visibilité de type viewshed ou autre (voir 3.3.1.3).

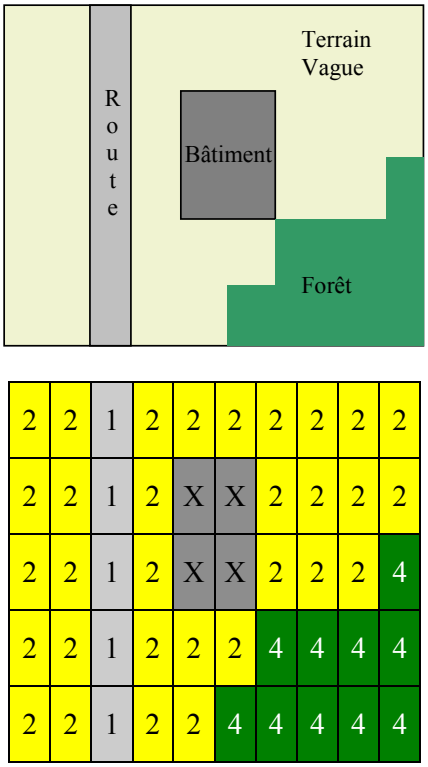


Figure 64 : Exemple de pondération des cellules d'un RSG suivant la nature du terrain

3.4.3.4 Application d'un algorithme du plus court chemin

Dans le graphe obtenu à la fin de l'étape 3.4.3.3, ce ne sont pas les arcs qui sont pondérés mais les sommets. Pour pouvoir appliquer à ce graphe un algorithme tel que celui de Dijkstra, il convient de retrouver le poids de chaque arc. A cette fin, nous proposons d'affecter à l'arc la moyenne des poids de ses sommets initiaux et finaux. Nous tenons également compte de l'adjacence des cellules. Ainsi, 2 cellules adjacentes par les côtés auront leurs centres plus proches que 2 cellules adjacentes par leurs sommets, un facteur $\sqrt{2}$ sera appliqué sur le coût d'un arc reliant deux nœuds représentant les centres de cellules voisines par leurs sommets (Figure 65).

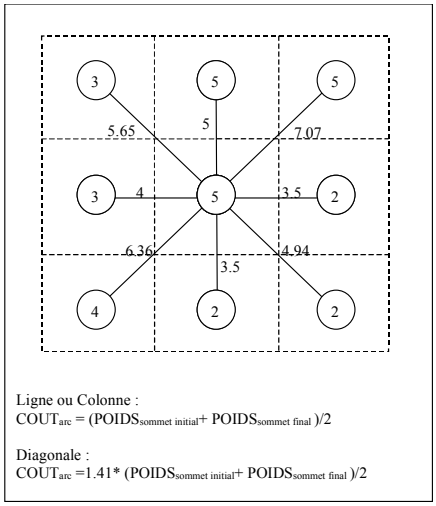


Figure 65 : Exemple de pondération des arcs d'un graphe à partir d'une structure RSG associée

Dans un espace 3D nous distinguerons 3 types d'adjacences entre cellules d'une RSG 3D, à savoir l'adjacence par face, l'adjacence par coté et l'adjacence par sommet. Nous appliquerons un facteur $\sqrt{2}$ à un arc joignant 2 cellules voisines par leurs cotés et un facteur $\sqrt{3}$ à un arc joignant deux cellules voisines par leurs sommets.

A partir d'un graphe correctement reconstitué, il ne reste plus qu'à enchaîner sur un des algorithmes vu en 3.4.2.

3.4.3.5 Reconstruction du chemin optimal

A partir du point final F, cette dernière étape sélectionne de proche en proche les cellules adjacentes ayant le plus faible « coût total de déplacement ». Lorsque deux cellules ont la même valeur de « Cumul de coût de déplacement », la première sélectionnée est choisie. On obtient ainsi un des chemins optimum parmi plusieurs. Le résultat est illustré sur la Figure 66 ; on constate que les résultats peuvent être radicalement différents selon la précision imposée à la modélisation de la scène. En effet, dans notre exemple, la matrice A a été obtenue en ne modélisant que les adjacences par côté alors que la matrice B prend également en compte les adjacences par les sommets.

Matrice A :

6	4	3	5	7	9	11	13	15	17
4	2	2	4	X	X	13	15	17	19
2	**0**	**1**	3	X	X	**14**	**16**	**18**	**22**
4	2	**2**	**4**	**6**	**8**	**12**	16	20	24
6	4	3	5	7	11	15	19	23	27

Matrice B :

3.4	2.8	2.4	2.8	**3.8**	**5.8**	**7.8**	**9.8**	11.8	13.8
2.8	2	1.4	**2.4**	X	X	8.6	10.6	**12.6**	14.6
2	**0**	**1**	2	X	X	9.6	11.4	13.4	**18.2**
2.8	2	1.4	2.4	4.8	6.8	10.8	15.2	17	19
3.4	2.8	2.4	3.8	5.2	10.4	12.4	16.4	20.8	22.6

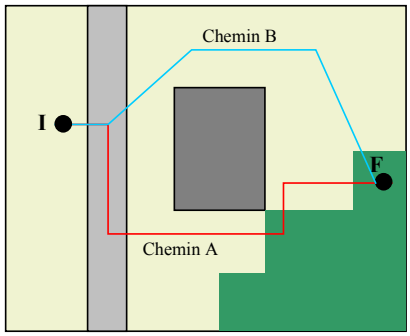


Figure 66 : Détermination de trajectoire à partir de deux modélisations différentes d'adjacences entre cellules de RSG.

3.4.4 Synthèse sur la recherche de trajectoire

Avec les fonctionnalités liées aux calculs d'intervisibilité, la recherche de trajectoire optimale constitue un deuxième point clé de notre système d'information géographique 3D.

Parmi les applications liées à la recherche de chemin optimal, nous distinguons celles qui s'effectuent sur des graphes prédéterminés de celles où le graphe est à reconstruire. Dans la première de ces deux catégories, nous rangeons tout ce qui concerne les calculs d'itinéraires sur réseau (réseau routier par exemple). Dans la seconde catégorie, nous retrouvons ce que nous nommons la recherche de trajectoire « libre ».

Dans un cas comme dans l'autre, ce sont des algorithmes de type Dijkstra qui sont utilisés pour rechercher les chemins optimaux. Cependant, dans le cas des trajectoires « libres », l'application de ces algorithmes est précédée d'une étape de reconstruction du graphe à traiter. Cette étape prend en compte un nombre important de contraintes applicables au déplacement du mobile, à savoir :

- Les objets du sur-sol (ex : bâtiment, pont, route) ;
- La nature des terrains traversés (ex : plaine, forêt, marécage, secteur goudronné) ;
- Le relief ;
- Les caractéristiques du mobile (poids, hauteur, largeur, longueur) ;
- Des critères de visibilité.

Tous ces calculs de trajectoire tiennent compte, bien entendu, de la forme tridimensionnelle des objets.

4 LE PROTOTYPE

4.1 Contexte et contraintes de développement

Où le partenariat entre acteurs industriel et universitaire implique d'orienter mes recherches de manière à satisfaire les deux parties.

La réalisation de TriGO (Tridimensionnal Geographical Object), prototype d'un Système d'Information Géographique 3D, a dû tenir compte d'un certain nombre de contraintes inhérentes au contexte dans lequel se sont effectués nos travaux de recherche. En effet, le financement de ces derniers fut assuré par une convention CIFRE qui établit un partenariat entre un acteur industriel et un acteur universitaire. Il incombe donc au bénéficiaire de ce type d'allocation d'orienter ses recherches de manière à pouvoir satisfaire chacune des parties.

Le contexte industriel implique une obligation de résultat, peut être plus contraignante qu'en milieu universitaire, et dont il a fallu tenir compte lors du développement de notre prototype. Notre travail de recherche s'est effectué parallèlement à une étude, menée par EADS pour le compte de la DGA, sur le potentiel de la 3D dans les systèmes d'informations géographiques. Cette étude amont portait sur la conception de systèmes permettant la gestion et l'utilisation de données géographiques 3D dans des domaines d'applications proches du combat urbain. De cette étude est né un démonstrateur prenant en compte les impératifs industriels et opérationnels émis par la DGA. L'objectif était de démontrer la faisabilité de certains choix techniques en matière d'architecture et de fonctionnalités 3D mis en avant au cours de l'étude. Ce démonstrateur fut soumis au jugement de différents corps d'armée au cours de phases d'expérimentations. Celles-ci ont permis de confronter des points de vue d'utilisateurs variés, et ainsi de contribuer à l'élaboration d'un document de spécifications préliminaires tenant compte des opinions et besoins de chacun. Ce document de spécifications préliminaires jette les bases non plus d'un prototype mais d'un système d'information géographique véritablement opérationnel.

Pour ne pas réinventer la roue et afin d'obtenir un prototype à peu près stable dans les temps impartis, nous avons choisi de ne pas « re-développer » chacune des composantes d'un SIG. Dans la mesure du possible, nous avons préféré nous appuyer sur des briques logicielles déjà existantes, tout en les adaptant à notre problématique. Sur ce point, notre souci fut toujours de favoriser les solutions garantissant une certaine pérennité. L'étude menée pour la DGA étant censée voir à long terme, nous ne pouvions préconiser des solutions logicielles issues de technologies peu matures et risquant à tout moment de s'écrouler. A titre d'exemple, nous avons préféré partir sur une solution de stockage relationnelle-objet plutôt que sur une solution strictement objet, pourtant tout à fait justifiable d'un point de vue théorique. Il est probable que dans un contexte purement universitaire notre choix fut différent, mais le caractère industriel de l'étude nous a amené à minimiser les risques. D'autant que les désillusions liées à l'échec commercial de la base de données objet O2 étaient encore fraîches dans les esprits.

Dans la suite de ce chapitre, nous présentons l'architecture du prototype, les principaux algorithmes utilisés, ainsi que quelques conclusions sur des expérimentations tant quantitatives que qualitatives.

4.2 Architecture du prototype TriGO

Où il est question de bâtir un prototype autour de notre modèle de données et de quelques composants externes.

4.2.1 Plate-forme de développement

Le contexte et les contraintes de l'étude, menée parallelement à cette thèse, nous ont conduit à développer TriGO sur une plate-forme PC équipée du système d'exploitation Windows NT 4/2000 et d'une carte graphique compatible avec la librairie graphique OpenGL.

TriGO respecte les standards d'interface utilisateur Windows puisque l'IHM (Interface Homme Machine) a été conçue à l'aide des classes C++ MFC proposées par Microsoft. Les MFC (Microsoft Foundation Classes) constituent un ensemble de classes prédéfinies autour desquelles s'articule la programmation Windows via Visual C++.

Les données sont stockées dans une base de donnée relationnelle étendue Oracle 8i. Cette base de données offre des extensions intéressantes et permet, dans une certaine mesure, de définir des types abstraits de données. Enfin, elle procure indirectement quelques possibilités de stockage de formes 3D complexes.

L'affichage 2D est pris en charge par Ilog Views qui fournit une bibliothèque de classes graphiques dédiée au primitives 2D.

L'affichage de scène 3D est assurée par OpenInventor qui est une bibliothèque de développement spécialisée dans les applications graphiques et particulièrement bien adaptée au 3D.

Lors du développement de notre prototype, la ligne de conduite générale fut de construire une architecture objet permettant d'être le plus modulaire possible, afin de pouvoir remplacer chaque brique du système avec un nombre limité de contraintes d'intégration.

4.2.2 Architecture générale

TriGO est un système architecturé autour d'un noyau supportant le modèle de données (Figure 67). Ce modèle s'appuie sur les propositions faites précédemment (paragraphe 2.3.1.2.5) et inclus donc une couche topologique en plus d'une modélisation géométrique classique de type B-Rep.

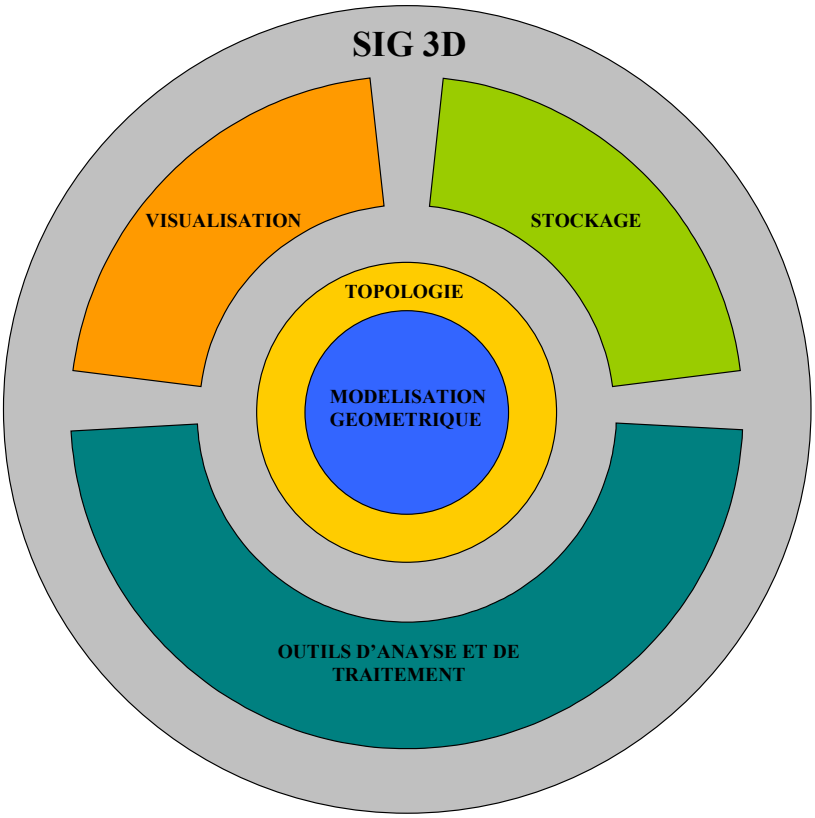


Figure 67 : Architecture générale de l'application TriGO

Les services de visualisation, stockage et traitement de données sont les trois grandes composantes fonctionnelles du prototype :

- L'objectif majeur de cette thèse est de proposer une gestion efficace de la 3D dans un système d'information géographique. Cependant, le module de visualisation que nous avons intégré propose des affichages 3D mai aussi 2D. Plus précisément, ce sont deux sous modules qui gèrent cet affichage. Ils ont été développés à partir des bibliothèques graphiques Ilog Views pour la 2D et OpenInventor pour la 3D

- Le nombre de données à stocker sous SIG 3D est important. Certaines données, comme les textures ou le MNT, sont directement stockées sous forme de fichiers sur le disque, d'autres, comme les objets géographiques, font appel à un SGBD, en l'occurrence Oracle 8i.

- Le dernier module regroupe les classes nécessaires à l'analyse et au traitement des données. Il répond aux requêtes sémantiques, aux requêtes géométriques et à des calculs plus complexes d'intervisibilité ou de recherches de trajectoires.

4.2.3 Dépendance des modules

L'interfaçage entre l'utilisateur et le prototype est réalisé par l'IHM (Figure 68).

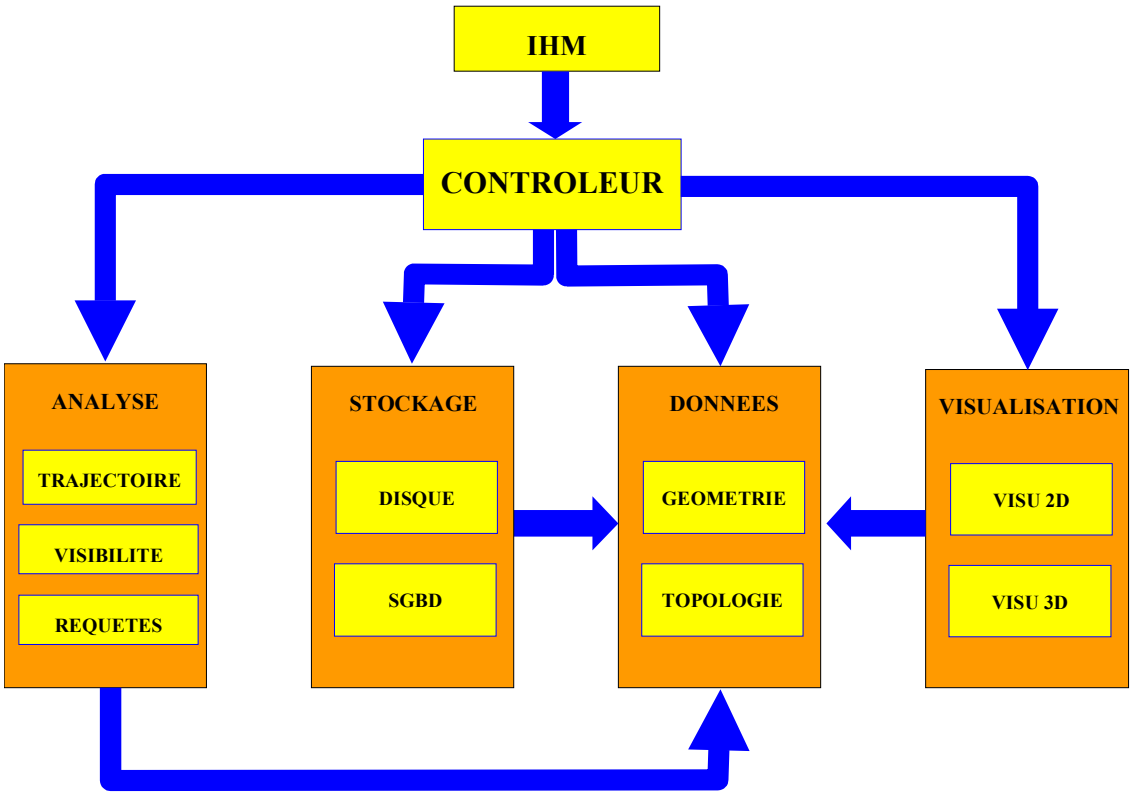


Figure 68 : Dépendance des modules de TriGO

L'IHM reste totalement indépendante du reste du prototype et ne communique avec les autres modules que par l'intermédiaire d'un « contrôleur ». Cette couche « contrôleur » gère toute la logique applicative qui permet d'enchaîner les traitements, les boites de dialogues ou les algorithmes géographiques.

Les composantes visualisation, stockage, et analyse sont indépendantes les unes des autres. Elles reposent toutefois sur la composante « données » qui abrite la modélisation géométrique et topologique des objets de la scène géographique modélisée.

Tout au long de la phase de développement de ce prototype, nous nous sommes appliqués à essayer de rendre chaque composant de SIG 3D le plus indépendant possible du reste du système. Cette philosophie de développement objet permettra par la suite de modifier ou mettre à jour un quelconque module du prototype sans qu'il y ait de répercutions sur le reste du logiciel. De même, l'application doit pouvoir intégrer facilement d'éventuels nouveaux modules sans remettre en cause l'existant.

La Figure 69 présente les principaux packages développés pour notre prototype et leurs dépendances vis à vis des composants externes intégrés dans ce prototype.

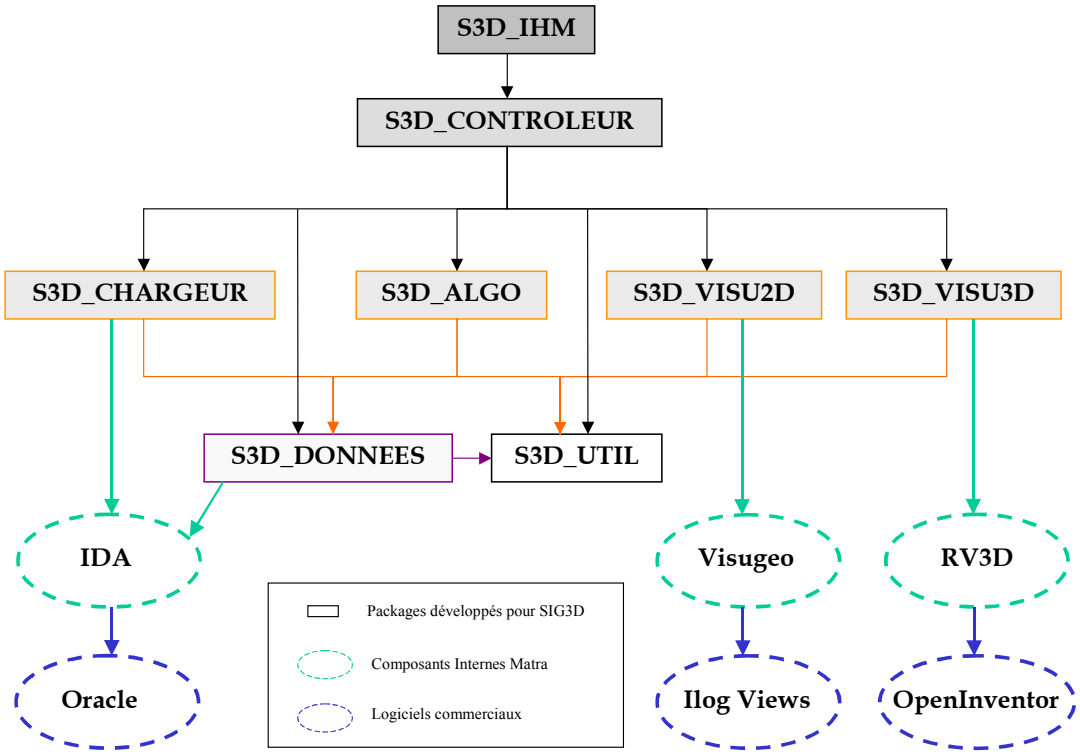


Figure 69 : Présentation des packages de TriGO et des composants externes

- Le package *S3D_IHM* regroupe l'ensemble des fenêtres et boîtes de dialogues formant l'interface entre l'utilisateur et l'application.
- Le package *S3D_CONTROLEUR* est la couche applicative s'occupant des échanges entres les différents modules de l'application.
- Le package *S3D_CHARGEUR* gère la création de la base de données et apporte un minimum de fonctions d'imports et d'exports.
- Le package *S3D_ALGO* regroupe les classes utiles pour effectuer des traitements et de l'analyse géographique sur les données.
- Le package *S3D_VISU2D* permet l'affichage 2D des objets géographiques composant la scène étudiée.
- Le package *S3D_VISU3D* permet l'affichage 3D de cette même scène.
- Le package *S3D_DONNEES* se charge de la modélisation géométrique et topologique des données.
- Le package *S3D_UTIL* regroupe des classes utilitaires utilisées par l'ensemble des autres packages.

Certains de ces packages font appels à des composants développés en interne au sein de EADS S&DE :

- Le composant *IDA,* qui gère les accès à la base de données (ici Oracle). Cette interface permet à une application objet de communiquer de manière transparente avec une base de donnée relationnelle étendue.

- Le composant *Visugeo,* dont le rôle est d'assurer l'affichage 2D d'une scène. Les primitives de dessin utilisées par Visugeo sont celles contenues dans la bibliothèque graphique Ilog Views.

- Le composant *RV3D,* qui affiche à l'écran les données géographiques sous forme de scène 3D. RV3D fait appel à la bibliothèque graphique OpenInventor.

4.2.4 Implémentation du modèle de données sous TriGO

4.2.4.1 Présentation

Au sein de TriGO, la gestion des données est en grande partie assurée par les classes du package S3D_DONNEES. Ces dernières se nomment :

- S3D_DoDonneeGeo
- S3D_DoImage
- S3D_DoLocCarto
- S3D_DoMNT
- S3D_DoRef
- S3D_DoRef3D
- S3D_DoRefCarto
- S3D_DoSite
- S3D_ToETArc
- S3D_ToETNoeud
- S3D_ToETPrimitive
- S3D_ToETReseau

La classe centrale de ce package est la classe S3D_DoDonneeGeo (Figure 70). Elle centralise toutes les modélisations prises en charge par le prototype, à savoir :

- La modélisation de la forme 3D des objets géographiques ;
- La modélisation des relations entre objets géographiques, c'est-à-dire leur intégration à un réseau ;
- La modélisation de la sémantique attributaire des objets géographiques.

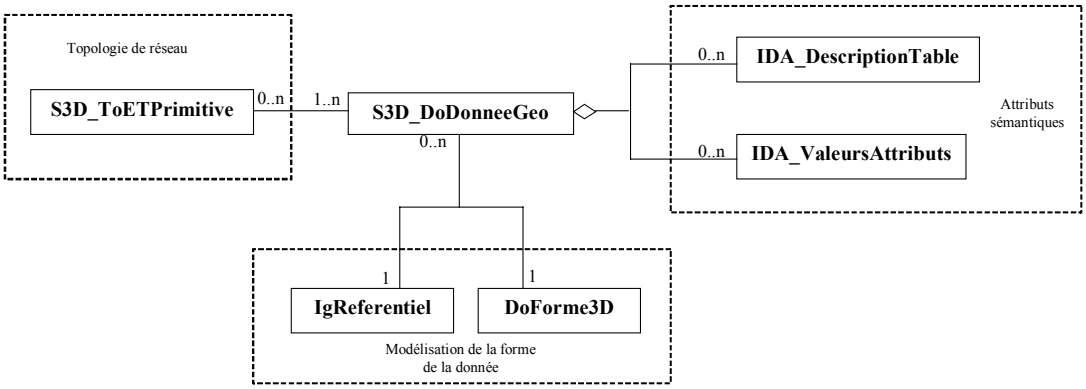


Figure 70 : Organisation des données sous le SIG 3D TriGO

De surcroît, nous verrons que la classe S3D_DoDonneeGeo est également impliquée dans la modélisation du terrain et plus généralement dans la modélisation de l'ensemble d'une scène 3D géographique.

4.2.4.2 Modélisation de la forme 3D des objets géographiques

4.2.4.2.1 Modélisation géométrique

La modélisation géométrique de la donnée est portée par les classes héritant de la classe DoForme3D. Cette DoForme3D est une classe générique du composant RV3D. Parmi les classes dérivant de DoForme3D celles qui nous intéressent sont DoPolyligne3D et DoFormeAFaces.

DoPolyligne 3D nous permet de modéliser les éléments linéaires. DoFormeAFaces permet de modéliser une forme 3D quelconque, aussi complexe soit-elle, par un ensemble de faces (technique de modélisation par frontière B-Rep, voir 2.3.1.2.5). La Figure 71 représente un peu plus en détail l'organisation de ces deux classes.

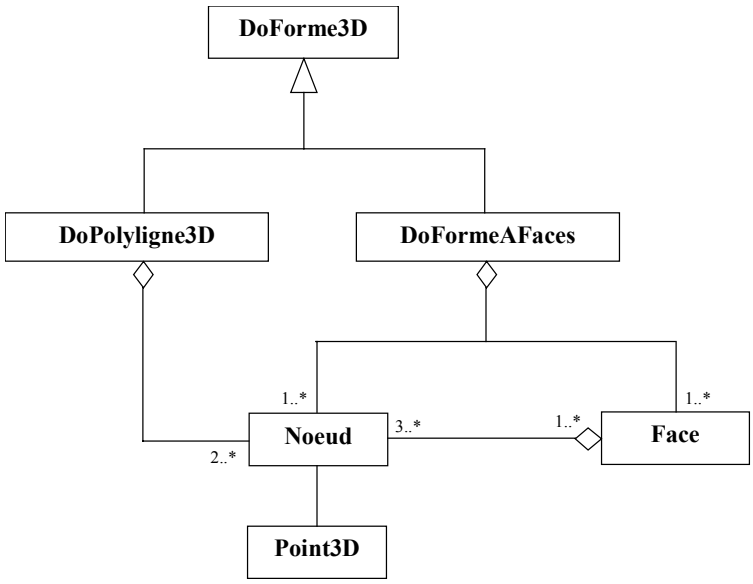


Figure 71 : Modélisation d'une forme 3D

4.2.4.2.2 Topologie structurelle

Le modèle de topologie structurelle que nous avons proposé en 2.3.2.2 n'est que partiellement repris dans l'implémentation de TriGO. Il a fallu nous adapter aux contraintes imposées par les composants logiciels que nous utilisions (tel que RV3D). Nous avons également fait des simplifications volontaires sur des points qui n'auraient pas été mis en valeur lors des applications envisagées pour notre prototype.

Le cœur du modèle, c'est-à-dire la relation topologique liant la face et le nœud, est bien là. Une DoFormeAFaces est décrite par une liste de noeuds et une liste de faces indexées par références sur la liste de noeuds. Ainsi, toute modification sur les nœuds se répercute sur les faces.

Une DoPolyligne3D est décrite par une liste de noeuds. Les arcs ne sont pas explicitement décrits, ils peuvent cependant se retrouver à partir de la liste de nœuds.

En ce qui concerne les volumes, les applications envisagées pour ce prototype ne nécessitent pas de partitionner l'espace en entités volumiques identifiables, seul les contours de volumes sont utiles. Nous n'avons pas intégré la modélisation des volumes dans notre implémentation.

4.2.4.2.3 Gestion des référentiels sous TriGO

TriGO permet d'afficher ensemble des données exprimées à l'origine dans des systèmes de coordonnées différents. Chaque objet de la classe S3D_DoDonneeGeo est associé à un référentiel par l'intermédiaire de la classe IgReferentiel.

La classe IgReferentiel est issue du package Visugeo. Nous avons implémenté trois classes héritant de IgReferentiel et dédiées au géoréférencement des données géographiques.

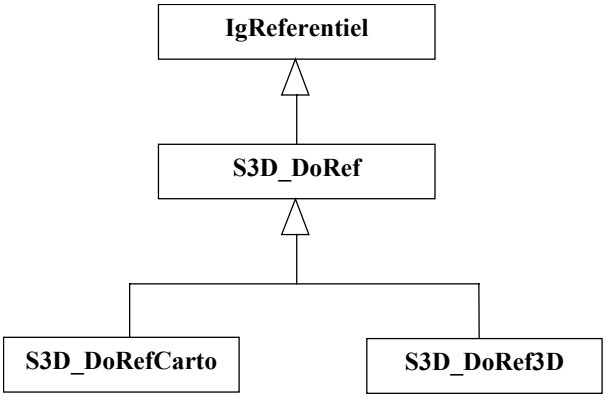


Figure 72 : Classes de référentiels utilisées dans le SIG 3D TriGO

La classe S3D_DoRef possède les méthodes permettant le stockage des référentiels dans la base de données.
La classe S3D_DoRefCarto hérite de S3D_DoRef, elle est spécialisée dans les référentiels cartographiques 2D, alors que la classe S3D_DoRef3D est spécialisée dans les référentiels 3D.

4.2.4.3 Les réseaux topologiques

Pour parcourir plus efficacement les réseaux, nous avons implémenté un ensemble de classes qui prend en charge la description topologique de ces réseaux. Chaque S3D_DoDonneeGeo peut être ainsi associée à un nœud ou à un arc d'un quelconque réseau. Dans ce prototype, nous nous limitons à un seul réseau par donnée géographique. En dehors de cette restriction, le modèle topologique de réseau que nous avons implémenté dans TriGO (Figure 73) correspond précisément à celui que nous avons présenté en 2.3.2.3 (Figure 22). La destruction du réseau entraîne la destruction des primitives arcs et nœuds qui le composent, un nettoyage est effectué sur les pointeurs vers les S3D_DoDonneeGeo concernées pour éviter des références hasardeuses.

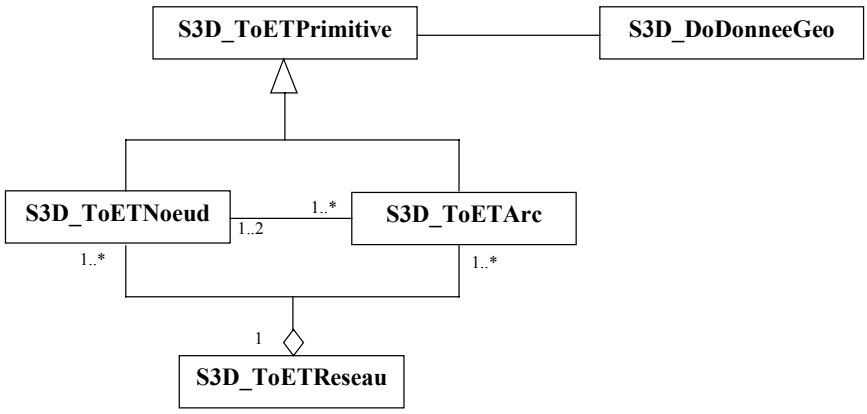


Figure 73 : Modélisation des réseaux topologiques

La classe S3D_ToETPrimitive est la classe mère possédant comme attribut une référence vers la donnée qui lui est associée et une référence vers le réseau auquel elle appartient.

La classe S3D_ToETArc dérive de S3D_ToETPrimitive. Elle pointe vers son nœud initial et son nœud final. Elle porte deux variables membres particulières qui permettent d'affecter un coût de déplacement à chaque sens de parcours.

La classe S3D_ToETNoeud dérive de S3D_ToETPrimitive. Un nœud peut être relié à un nombre quelconque d'arcs (hormis zéro, les nœuds isolés ne sont pas traités ici). Le nœud porte un triplet de coordonnées (x,y,z) pour simplifier sa localisation géographique.

La classe S3D_ToETReseau comporte deux listes référençant, d'une part les nœuds et, d'autre part, les arcs qui composent cette classe. Ce sont les méthodes de cette classe qui permettent d'appliquer des algorithmes de parcours de graphe au réseau.

4.2.4.4 Gestion des attributs sémantiques

Les attributs sémantiques des objets sont stockés dans une base de données relationnelle. Pour que l'utilisateur puisse accéder à ces attributs, la classe S3D_DoDonneeGeo fait appel à deux classes dédiées à leur gestion (Figure 74).

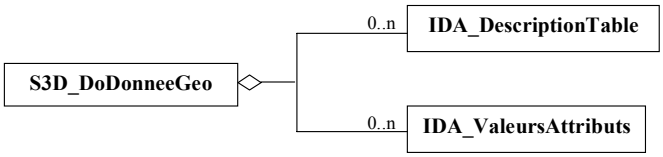


Figure 74 : Gestion des attributs sémantiques

A chaque type de donnée de l'application correspond une table d'attributs sémantiques stockée dans la base de données. La classe IDA_DescriptionTable décrit cette table, c'est-à-dire le nombre d'attributs attachés à la donnée, leurs noms, leurs types, les valeurs qu'il peuvent prendre, etc.

La classe IDA_ValeursAttributs stocke quand à elle les valeurs des attributs sémantiques de la donnée.

La Figure 75 présente le modèle conceptuel sémantique que nous avons suivi pour organiser l'information sémantique dans TriGO.

La classe de base de la hiérarchie sémantique est la classe « objet ». De cette classe « objet » vont dériver les classes sémantiques « MNT », « Image », « Site » d'une part, et les classes concernant plus particulièrement les objets du sur-sol, d'autre part. Ces objets du sur-sol sont classés en deux catégories selon leur dimension spatiale. Tous les objets sont exprimés dans un référentiel 3D, mais certains sont représentés par une forme volumique, de dimension 3, et sont considérés comme objets 3D. D'autres ont une modélisation uniquement surfacique ou linéaire, donc de dimension strictement inférieure à 3, ils sont classés comme objets 2D.

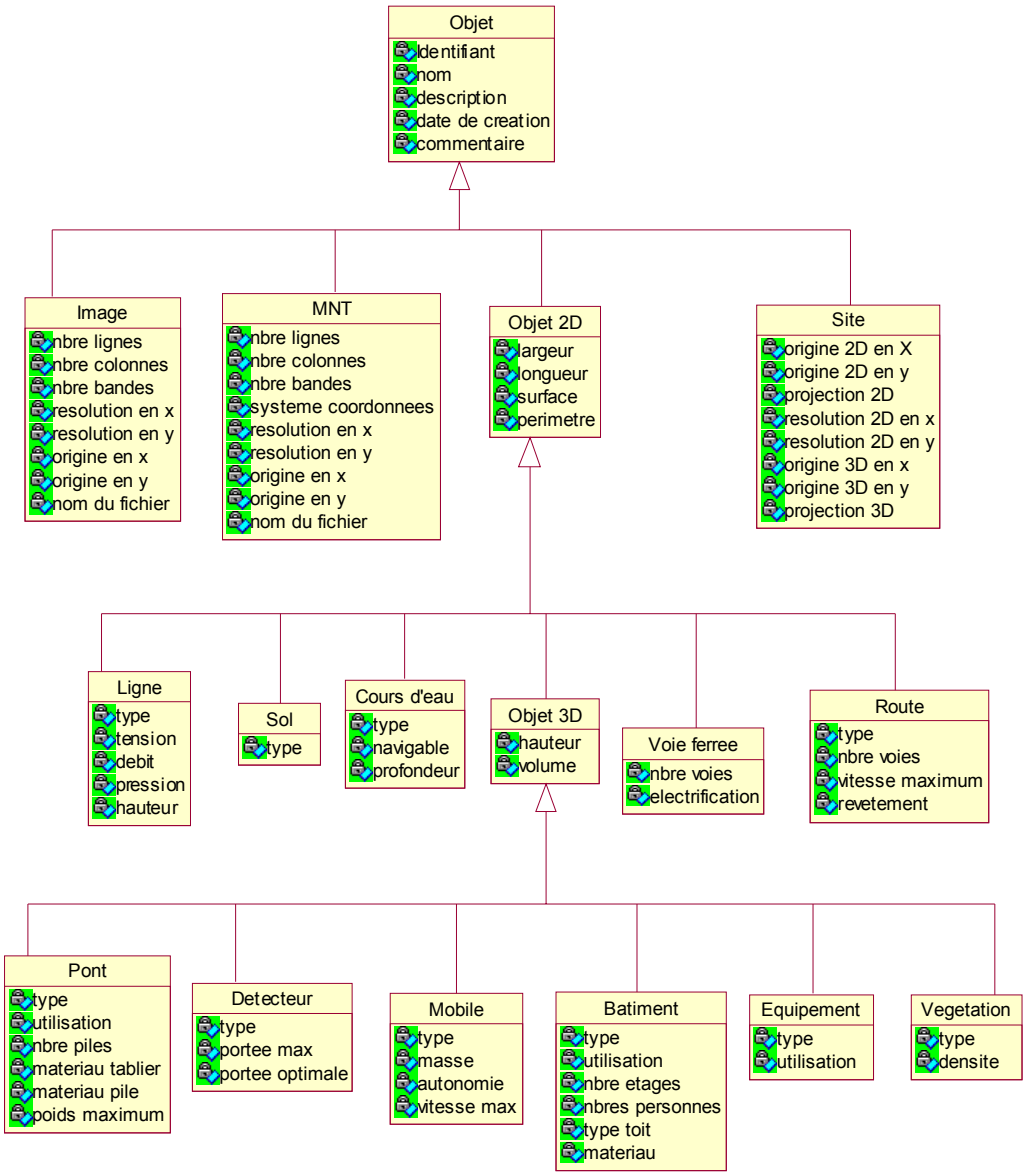


Figure 75 : Modèle Sémantique du SIG 3D TriGO

La liste des classes retenue ici n'est absolument pas exhaustive. Son seul mérite est de rendre le prototype un peu plus démonstratif pour nos expérimentations auprès d'utilisateurs recherchant des exemples d'applications. Ce modèle sémantique est évolutif et a été conçu pour qu'il soit aisé de retirer, modifier ou ajouter à loisir de nouveaux types sémantiques.

4.2.4.5 Modélisation des MNT

La classe S3D_DoMnt est dédiée à la gestion des modèles numériques de terrain utilisés dans TriGO. Elle hérite de la classe S3D_DoDonnéeGéo et possède une référence vers le MNT.

Bien que d'autres formes de MNT entrent dans le cadre du développement de ce prototype, nous nous sommes restreints à ne prendre en entrée que des MNT standardisés au format BIL, notre application supporte d'autre formats tel que le DTED ou le DEM.

TriGO traite les MNT sous deux formes :

- Sous sa forme matricielle, pour rechercher l'altitude en un point quelconque du MNT. Cette opération peut requérir une précision importante : moins le MNT sera simplifié, meilleur sera le résultat de la recherche.

- Sous sa forme vecteur pour l'affichage. L'affichage d'une scène 3D est un processus lourd et complexe, il est donc avantageux de simplifier un MNT de type RSG (Regular Square Grid) avant de l'afficher. Le rendu visuel du MNT n'est alors qu'imperceptiblement dégradé, alors que les temps d'affichage sont très nettement améliorés, le gain en confort d'utilisation est donc notable. C'est la raison pour laquelle nous convertissons le MNT en TIN (Triangulated Iregular Network) avant de le passer en entrée des processus d'affichage.

Nous offrons la possibilité d'habiller le MNT avec une image ortho-rectifiée, comme par exemple avec une image satellitaire ou une photographie aérienne. C'est cette même image que nous affichons dans la vue 2D. A l'instar de la classe S3D_DoMNT, la classe S3D_DoImage hérite de S3D_DoDonneeGeo et possède un pointeur vers l'orthophoto (Figure 76). Les processus de plaquage de la texture sur le MNT sont pris en charge par la bibliothèque graphique RV3D.

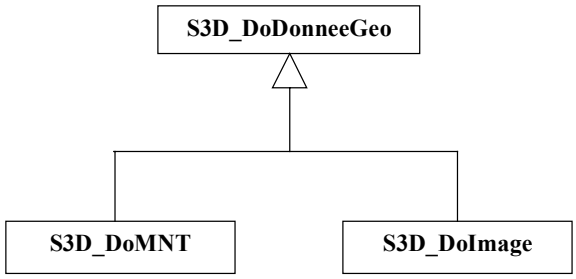


Figure 76 : Les classes dédiées au MNT

4.2.4.6 La notion de Site

Un Site correspond à l'ensemble des données concernant un même secteur géographique. Chaque secteur géographique regroupe :

- Un Modèle Numérique de Terrain ;
- Un image ortho-rectifiée d'origine aérienne ou satellitaire ;
- Un lot d'objets caractérisant le sur-sol.

Un site est lui-même une donnée de type S3D_DoDonneeGeo, la classe S3D_DoSite hérite donc de S3D_DoDonneeGeo (Figure 77).

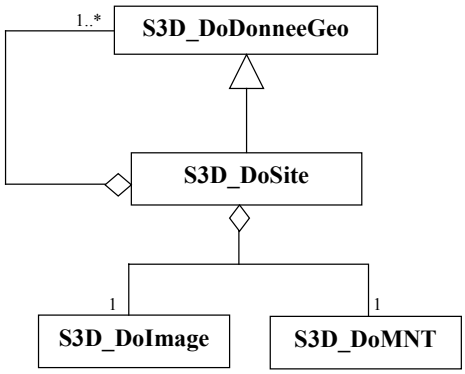


Figure 77 : Composition d'un S3D_DoSite

4.2.5 Stockage des données

Dans les paragraphes précédents, nous avons rapporté comment l'information géométrique, topologique, sémantique liée aux données est organisée en mémoire centrale. Dans ce qui suit, nous allons nous intéresser à son stockage.

4.2.5.1 Le stockage du modèle

Oracle étant un système de gestion de base de données relationnelle (SGBDR), nos données, pourtant représentées par des classes objets en mémoire, sont stockées dans Oracle sous forme de tables relationnelles. Le composant logiciel qui permet de réaliser la conversion objet-relationnel a été développé par Matra Systèmes & Informations et se nomme IDA (Immediate Data Access) [RHIN 1997] (voir 3.2.1.2.2). Dans le cadre de nos travaux, nous n'avons pas exploité tout le potentiel de cette bibliothèque, nous nous sommes contentés d'en faire une utilisation simple de manière à stocker proprement nos données sous Oracle.

IDA associe à chacune des 17 classes du modèle sémantique illustré en Figure 75 une table dans le SGBD. Les colonnes de ces 17 tables « données » correspondent aux attributs des classes et chaque tupple stocke une instance de la classe correspondant à la table. La première de ces tables est la table OBJET suivante :

OBJET :

ID	IDREF	IDPAL	ENGLOBANT	FORME	PROPRIETAIRE	GROUPE	DROIT	NOM	DESC	DATE	COMMENTAIRE

Sont entre autres stockés dans cette table :

- Un identifiant unique sur toute la BD ;
- Une référence vers le référentiel dans lequel est stockée la forme ;
- Une relation avec la palette utilisée par la forme (les textures et matériaux appliqués sur la forme) ;
- La forme 3D de la donnée qui est stockée dans un BLOB, c'est à dire un tableau d'octets ;

- L'englobant de la forme ;
- Quelques attributs de gestion de la donnée.

Les 16 autres tables reprennent les champs de la table objet, auxquels elles ajoutent les champs des attributs qui sont propres à la nature de l'objet qu'elles stockent. Ci-dessous l'exemple de la table stockant les objets de type « bâtiment » :

BATIMENT :

ID	…	TYPE_BATIMENT	NBRE ETAGES	NBRE PERSONNES	MATERIAU	TYPE TOIT

En dehors de ces tables stockant le contenu de la donnée, d'autres ont un rôle dans la gestion et l'administration du stockage, les voici :

IDA_DESCRIPTIONTABLES

La table qui référence toutes les tables impliquées dans le modèle conceptuel de données se nomme « Ida_descriptiontables ».

IDDESCTABLE	NOM	DESCRIPTION	TYPE	VERSION	TAILLEDONNEE

IDA_DESCRIPTIONTABLEHERITAGE

La table « Ida_descriptiontableheritage » stocke les relations d'héritage entre classes du MCD (Model Conceptuel de Données). Chaque tupple de la table correspond à une relation « hérite de » qui s'exprime à l'aide des identifiants de la table Mère et de la table Fille.

NUMRELATION	IDDESCTABLEFILS	IDDESCTABLEPERE

IDA_DESCRIPTIONATTRIBUTS

Le type des attributs de chaque classe du MCD est décrit par une table nommée « Ida_decriptionattributs » dans laquelle sont, entre autres, précisés le type de l'attribut, l'unité utilisée ou encore les valeurs minimales et maximales que peut prendre l'attribut. Plus généralement il s'agit d'une description de l'ensemble des propriétés d'un attribut.

IDDESCATTR	IDDESCTABL	NOM	DESCRIPTION	TYPE	BORNEMIN	BORNEMAX	DEFAUT	UNITE	ETAT	ORDRE

IDA_DICTIONNAIRE

Les dictionnaires référencent les valeurs que peut prendre un attribut qui est de type énumération (enum). Leurs persistances sont assurées par la table « Ida_dictionnaire ». Chaque tupple de la table correspond à une des valeurs possibles pour un attribut donné.

IDDESCATTR	VALEUR	NOM

IDA_RELATIONS

Une base de données relationnelle est bien adaptée au stockage des différents attributs d'une classe. En revanche, dans cet univers relationnel, il n'y a pas de structures dédiées à la persistance des liens entre classes du modèle objet. Une façon de contourner le problème est de créer des tables supplémentaires « Ida_Ralations » qui ont pour tâche de faire la correspondance entre les identifiants des deux données et les identifiants de leurs tables d'appartenance. Il s'agit donc d'une correspondance entre deux tupples de tables éventuellement différentes. L'attribut « verrouillé » est un attribut booléen protégeant la destruction des données sources.

IDDESCTABLEP	IDPERE	IDDESCTABLEF	IDFILS	VERROUILLE	TYPE

IDA_REFERENTIELS

La table « Ida_referentiels » s'occupe du stockage des différents référentiels utilisés dans le prototype. Notons que la valeur du référentiel est stockée sous forme de BLOB (Binary Large Object).

IDREFERENTIEL	NOM	DESCRIPTION	VALEUR

La Figure 78 reprend les points clés de ces tables et met en avant les relations entre tables.

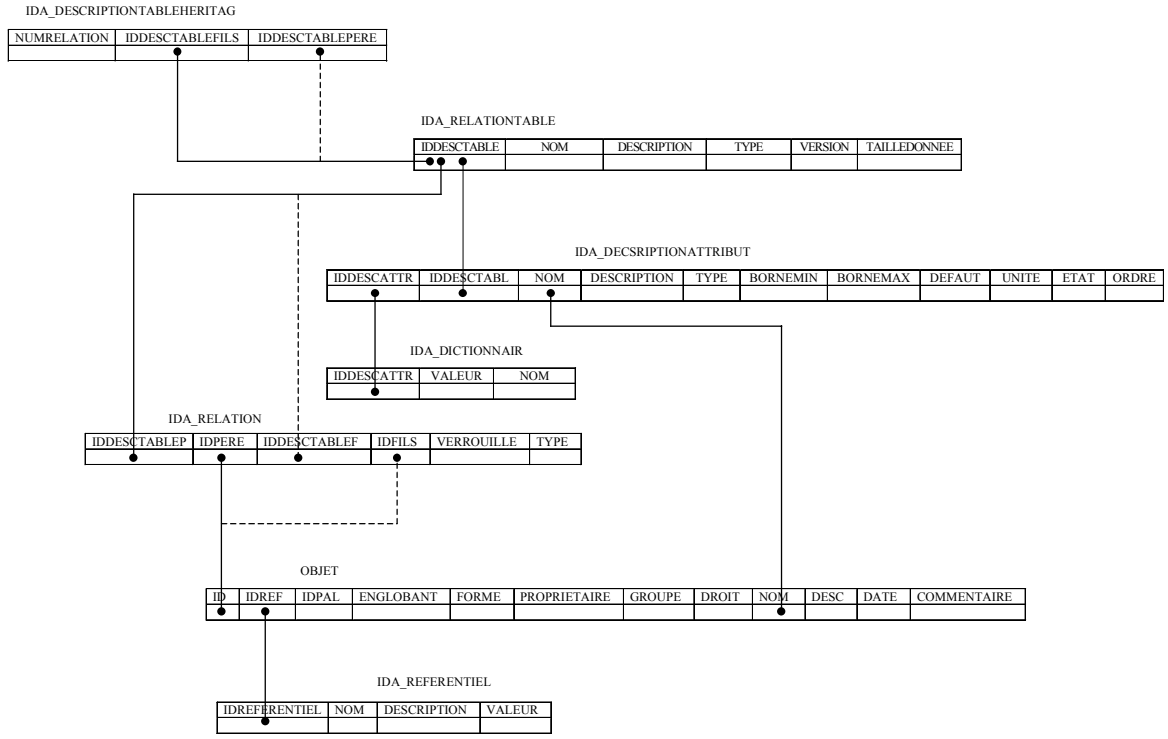


Figure 78 : Persistance des données sous TriGO

4.2.5.2 Les chargeurs de données

Les classes du package S3D_CHARGEUR nous ont servi pour créer la base de données. Elles sont au nombre de cinq :

- La classe S3D_ChMCD dont les méthodes sont consacrées à la création du modèle conceptuel de données (MCD) de la base de données.

- La classe S3D_ChChargeurAsc, qui permet d'importer des données sous format Asc dans la base de données du prototype. Le format Asc est un format du logiciel SIG APIC.

- La classe S3D_ChChargeurDxf, qui permet d'importer des données au format dxf dans la base de données du prototype. Le format dxf est un format qui tire son origine d'Autocad et qui est devenu aujourd'hui un format standard et incontournable en modélisation de données 2D ou 3D.

- La classe S3D_ChCreateurAsc, qui permet d'exporter des données au format Asc.

- Et enfin la classe S3D_ChChargeurCreateurBd, qui crée véritablement la base de données.

4.2.6 Synthèse sur l'architecture générale de TriGO

TriGO (TRIdimensionnal Geographical Object) est donc un prototype de système d'information géographique tridimensionnel conçu autour d'une architecture objet et fonctionnant sur une plateforme PC Standard. Pour réaliser ce prototype, nous nous sommes appuyés sur des composants logiciels préexistants, certains sont développés en interne au sein du département géomatique de EADS S&DE, d'autres sont des solutions commerciales courantes. Ces composants externes assurent principalement les taches de visualisation (2D et 3D) et la gestion des données persistantes.

L'essentiel des points de modélisation abordés au chapitre 2.3 ont été implémentés dans TriGO :

- La forme 3D des objets géographiques est en effet modélisée par des « formes à faces » correspondant à une modélisation par frontière de type BRep.

- Sur cette modélisation géométrique est greffée une modélisation de topologie structurelle partielle.

- Un modèle de topologie de réseau a également été implémenté pour faciliter la navigation au sein de ces derniers.

Le stockage des objets géographiques est pris en charge par le système de gestion de base de données relationnelle étendu Oracle 8i. Le composant IDA (Immediate Data Access) gère l'étape de transformation de notre modèle de données objet en tables relationnelles pouvant être stockées dans Oracle.

4.3 Principales fonctionnalités

Dans lequel on parle de l'implémentation d'un index spatial, de calculs de visibilité et de recherches de trajectoire.

4.3.1 L'index spatial R-Tree

Que ce soit en matière d'intervisibilité, de calcul de trajectoire ou de requêtes géométriques, la manipulation de formes 3D est coûteuse, aussi convient-il de minimiser le nombre d'objets géographiques à traiter. La sélection préalable des objets géographiques peut elle-même se révéler lourde si elle ne repose pas sur une structure de recherche efficace. Comme annoncé dans notre synthèse sur les index spatiaux (paragraphe 3.2.3.8), nous avons choisi d'intégrer dans TriGO un RTree pour optimiser les recherches d'objets sur critères spatiaux.

4.3.1.1 Choix du R-Tree

Nous l'avons vu au chapitre 3.2.3, un R-Tree a pour fonction essentielle de permettre un accès plus rapide aux données géographiques en les organisant sous forme d'une hiérarchie de boîtes englobantes. Parmi les différents modèles de R-Tree recensés dans la littérature, nous avons optés pour un R-Tree classique avec recoupement des boîtes englobantes mais sans duplication des entrées du R-Tree, ce qui offre un bon compromis pour nos applications

La Figure 79 illustre la structure d'un R-Tree classique comparable à celui que nous utilisons dans notre SIG 3D TriGO. Notons cependant, que pour des raisons de clarté, le R-Tree est représenté ici en 2D, bien que nous ayons implémenté un R-Tree 3D dans notre application. Le R-Tree 3D repose sur les mêmes principes que le R-Tree 2D et l'ajout d'une troisième dimension ne pose aucune difficulté majeure.

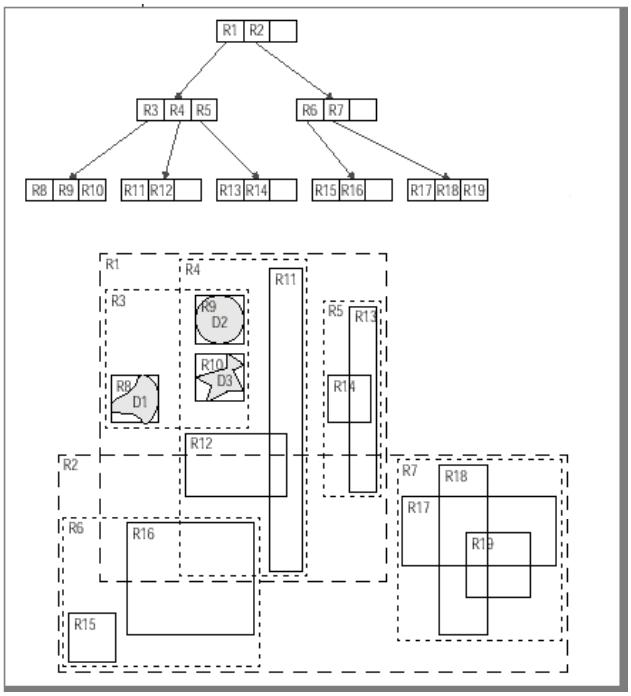


Figure 79 : R-Tree Classique

4.3.1.2 Implémentation du R-Tree

4.3.1.2.1 Organisation structurelle

Les classes impliquées dans l'architecture de l'index spatial sont au nombre de quatre :

- La classe S3D_AlRTNoeud
- La classe S3D_AlRTBranche
- La classe S3D_AlRTFeuille
- La classe S3D_AlRTree3D

Comme son nom l'indique, l'index spatial R-Tree peut être vu comme un arbre. La structure principale de notre arbre (Figure 80) est constituée par la classe S3D_AlRTree3D. Les ramifications de cet arbre sont de type « branche » avec la classe S3D_AlRTBranche ou de type « feuille » avec la classe S3D_AlRTFeuille. La classe S3D_AlRTFeuille porte la donnée géographique alors que la classe S3D_AlRTBranche est un agrégat de feuilles ou d'autres branches. Enfin, les feuilles et les branches de l'arbre ont une origine commune (assimilable au bourgeon) modélisée dans notre index spatial par la classe S3D_AlRTNoeud.

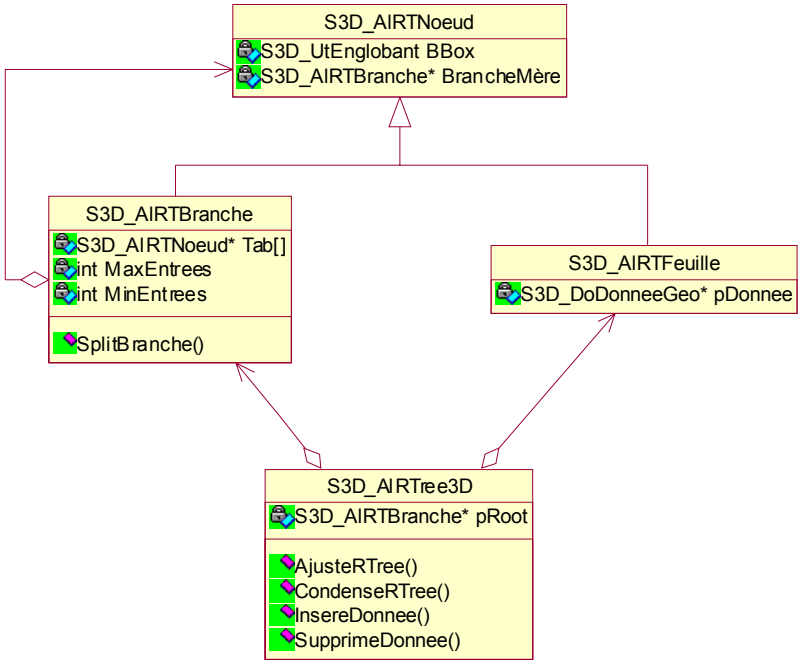


Figure 80 : Diagramme UML des relations entre classes composant le R-Tree

4.3.1.2.2 Quelques méthodes implémentées

Nous ne détaillons pas ici l'ensemble des méthodes implémentées autour de notre index spatial, nous nous contentons d'apporter quelques précisions sur les plus importantes d'entre elles.

4.3.1.2.2.1 Les méthodes d'accès au R-Tree

Insertion

La méthode S3D_AlRTree3D : :insere_donnee(…) permet d'insérer une donnée dans le R-Tree. Cette insertion peut se faire à une profondeur quelconque du R-Tree, par défaut les nouvelles données étant toutes insérées pour une profondeur maximale (ainsi, toutes les feuilles sont situées en bout d'arborescence). Si nécessaire, des branches de l'arbre sont scindées en 2 pour accueillir de nouvelles branches ou de nouvelles feuilles (méthode S3D_ARTBranche : :SplitBranche(…))

Suppression

La suppression de données est prise en charge par la méthode S3D_AlRTree3D : : supprime_donnee(…). Une fois la donnée supprimée, un appel à la méthode S3D_AlRTree3D : :Condense(…) est effectué pour éliminer les branches insuffisamment remplies du R-Tree.

Recherche de données dans une zone précise

Nous proposons plusieurs méthodes consacrées à la recherche géographique de données :

- S3D_AlRTree3D : :cherche_donnees_contenant_point(…) retourne les données dont la boîte englobante recouvre un point donné.
- S3D_AlRTree3D : :cherche_donnees_dans_englobant(…) retourne les données strictement incluses dans un englobant prédéfini.
- S3D_AlRTree3D : :cherche_donnees_intersectant_englobant(…) retourne les données en partie incluses dans un englobant prédéfini.

4.3.1.2.2.2 Les méthodes de maintenance du R-Tree

Dichotomie

Lorsqu'une branche est saturée et ne peut plus recevoir de nouvelles feuilles en entrée, il est alors nécessaire de la scinder en deux nouvelles branches pour augmenter la capacité d'accueil du RTree. Cette opération est effectuée par la méthode S3D_AlRTBranche : :SplitBranche(…) de notre index spatial. Elle implique la création d'une nouvelle branche, et les feuilles ou branches descendantes d'une branche saturée sont ensuite redistribuées, en fonction de leur localisation, entre la nouvelle branche et la branche anciennement saturée.

Ajustement

Il est fréquent de devoir propager de proche en proche l'opération de dichotomie décrite précédemment aux branches mères de la ramification, jusqu'à éventuellement créer une nouvelle racine. Nous avons implémenté à cet effet la méthode S3D_AlRTree3D : :AjusteArbre(…).

Condensation

Suite à une suppression de donnée, une branche du R-Tree peut se retrouver vide, ou insuffisamment remplie, auquel cas il convient de supprimer les ramifications inutiles. S3D_AlRTree3D : :CondenseRTree(…) est appelée après chaque suppression d'élément afin de maintenir la compacité de l'index spatial et de garantir ainsi une distribution homogène.

La Figure 81 illustre quelques situations d'insertions et de suppressions d'objets dans un R-Tree. Les cas a) et b) sont triviaux, en revanche le cas c) implique un appel aux méthodes SplitBranche(…) et AjusteBranche(…) et le cas d) met en évidence le traitement effectué par la méthode CondenseRTree(…).

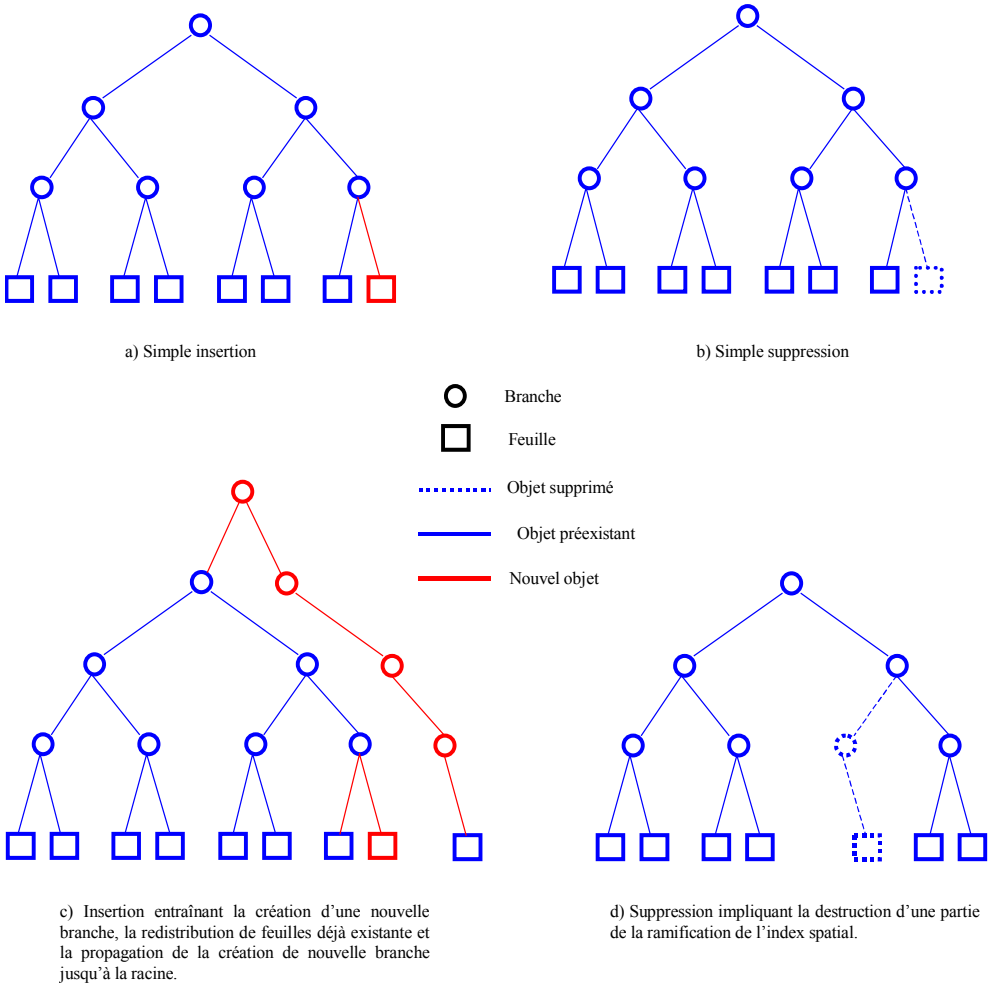


Figure 81 : Insertion et suppression d'objets dans un R-Tree

4.3.1.3 Compte rendu d'utilisation

L'arbre R-Tree que nous avons mis en place nous a effectivement permis de gagner un temps non négligeable sur des requêtes impliquant une sélection spatiale des objets. La Figure 82 illustre les temps d'accès aux objets contenus dans un cercle de rayon donné. Nous constatons que l'utilisation d'un R-Tree permet de diviser, en moyenne, les temps d'accès par un facteur proche de deux.

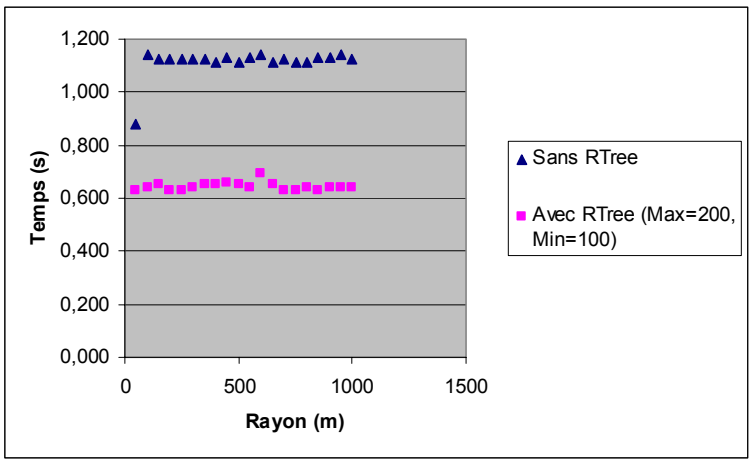


Figure 82 : Comparaison des temps d'accès aux données avec et sans l'utilisation R-Tree

Le R-Tree utilisé ici est un R-Tree dont les boîtes englobantes comptent au maximum 200 références et au minimum 100 références. Le nombre d'objets total composant la scène est de 2 millions, l'arbre s'étend donc sur 3 niveaux de profondeur.

Il est à noter que les caractéristiques des boîtes englobantes du R-Tree déterminent l'ordre dans lequel sont retournés les objets sélectionnés. Ceci peut avoir une influence sur les traitements venant en aval de la sélection spatiale. Nous avons été confrontés à ce problème lors de nos tests d'intervisibilité : nous constations des disparités importantes dans les temps de traitements pour des R-Tree utilisant des boîtes englobantes aux caractéristiques très proches. Ceci était dû à l'ordre dans lequel étaient traités les objets retournés par les appels au RTree. En effet, chaque objet est composé d'un certain nombre de faces, plus ce nombre de faces est important, plus les tests de visibilités qu'on leur applique sont longs à effectuer. En triant systématiquement les objets retournés par le R-Tree selon l'ordre croissant du nombre de ses faces, nous avons pu éviter des calculs inutiles et ainsi gagner en temps de traitement. Ce sont ces calculs d'intervisibilité que nous allons maintenant aborder.

4.3.2 Calculs d'intervisibilité

4.3.2.1 Introduction

Les calculs d'intervisibilité font partie des fonctionnalités les plus intéressantes pour un SIG 3D et les applications en sont nombreuses : télécommunications, défense, video-surveillance, urbanisme, etc. Comparé à une extension 2.5D classique, le modèle de donnée véritablement 3D utilisé dans notre prototype permet d'affiner le résultat retourné lors des tests de visibilité.

Pour notre étude sur les calculs d'intervisibilité, nous nous sommes limités à des signaux rectilignes entre l'observateur et la cible. Cette approximation reste acceptable pour la grande majorité des applications tirant un intérêt de ces calculs. L'amplitude du signal ondulatoire est très souvent négligeable par rapport à la distance entre l'observateur et la cible ou les éventuels obstacles.

Nous ne gérons pas non plus les phénomènes de réflexion ou de réfraction. Par manque d'information sur les données, celles-ci ne sont pas modélisées assez précisément pour prendre en compte ces phénomènes.

Puisque nous assimilons le signal à un rayon, les calculs d'intervisibilité se réduisent à des calculs d'intersection entre le ou les rayons visuels issus de l'observateur, et les obstacles qui peuvent se dresser entre l'observateur et la cible. Ces calculs d'intersection sont implémentés en tant que méthodes de la classe S3D_AlIntersection.

TriGO implémente deux types de calcul d'intervisibilité. Le premier est un calcul d'intervisibilité partielle qui stipule si l'observateur voit au moins une partie de la cible, et le second est un calcul de visibilité totale qui vérifie si la cible est entièrement visible depuis l'observateur. Les techniques à mettre en œuvre selon la nature du résultat souhaité ne sont pas les mêmes, c'est ce que nous allons évoquer maintenant.

4.3.2.2 Calculs de visibilité partielle

4.3.2.2.1 Principe de l'algorithme

Dans le cas d'un calcul de visibilité partielle, l'objectif est de savoir si au moins une partie de l'objet cible est vue depuis la position de l'observateur. Pour expliquer les principes de l'algorithme, nous traiterons de visibilité entre un observateur ponctuel et une cible volumique.

Les paramètres en entrée sont donc un point 3D représentant l'observateur et une forme 3D de type FormeAFaces (cf. 4.2.4.2.1) représentant la cible. L'algorithme raisonne de manière discrète en lançant des rayons depuis le point observateur et en direction de la cible (Figure 83).

La première étape consiste à discrétiser la forme cible en une grille régulière de points. Il est possible d'agir sur la précision du calcul en paramétrant le pas de discrétisation. Chacun des points de cette grille forme avec le point observateur un rayon. Tour à tour on vérifie que chacun des rayons ainsi formés n'est intersecté par aucun autre objet de la scène. Dés qu'un rayon atteint la cible sans être intersecté, la cible est déclarée partiellement visible. Le résultat retourné s'exprime sous la forme d'un booléen.

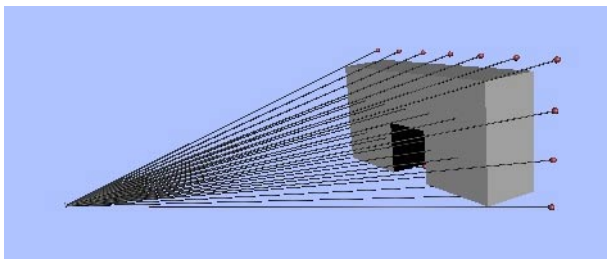


Figure 83 : Discrétisation puis lancer de rayon

4.3.2.2.2 Discrétisation de la forme 3D en une grille régulière de points

Cette discrétisation ramène les calculs de visibilité sur formes complexes à des calculs de visibilité sur un ensemble fini de points. Nous présentons ici le squelette de l'algorithme utilisé (Figure 84).

Soit V un observateur ponctuel et C la cible de forme quelconque dont on veut étudier la visibilité depuis V.
Soit S l'ensemble des sommets s de C.

On détermine dans un premier temps le barycentre G des sommets de S en attribuant la même valeur pondérée de 1 à chaque sommet s ∈ S.

Soit V(VG,VN,VZ) un nouveau repère orthogonal formé à partir de l'axe VG.

Dans ce nouveau repère on calcule la boîte englobante B de la forme associée à la cible C.

Soit P le plan portant la face de l'englobant B la plus éloignée de V et orthogonal à VG.

Pour chaque point s ∈ S, on détermine la projection s' de s sur P par la projection centrale de centre V.

Soit R le rectangle englobant des points s' de P. La dernière étape de cet algorithme consiste à discrétiser la surface de ce rectangle en un certain nombre de points s'' ∈ S'' suivant un pas de discrétisation donné. C'est cet ensemble de points S'' qui est retourné.

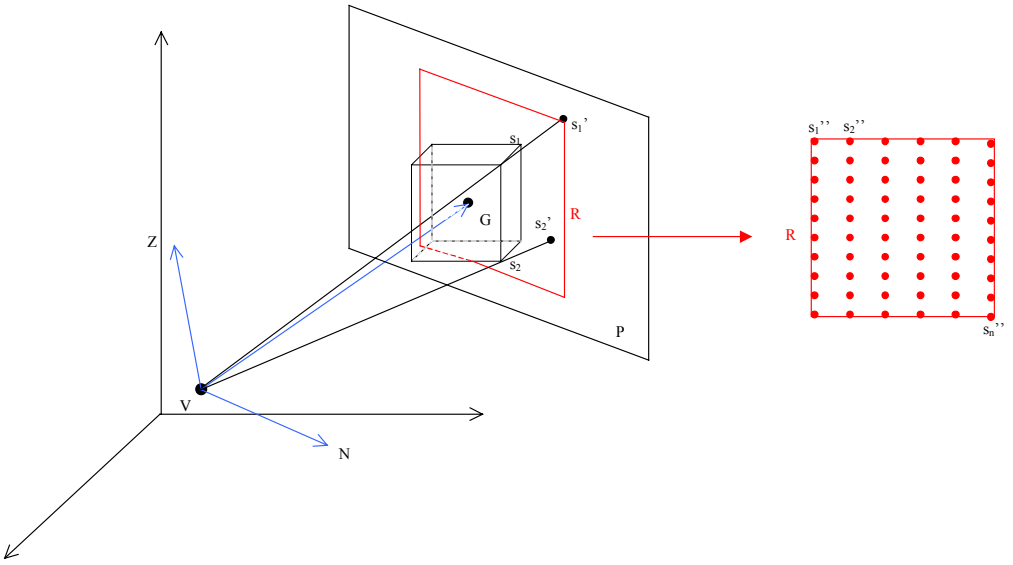


Figure 84 : Discrétisation d'une forme en une grille de points en vue des tests d'intervisibilité

4.3.2.2.3 Recherche des intersections

Après l'étape de discrétisation de la forme en une grille régulière de points, vient l'étape de recherche d'intersections. Associés à un observateur ponctuel, les nouveaux points issus de la discrétisation forment les segments pour lesquels on recherche des intersections avec les objets de la scène.

4.3.2.2.3.1 Sélection des obstacles éventuels

Il est évident que seules les formes à proximité immédiate des segments seront éventuellement impliquées en tant qu'obstacles dans les intersections qui nous intéressent. Aussi est-il indispensable de pré-sélectionner les candidats aux intersections par l'utilisation du R-Tree présenté en 4.3.1.

La recherche de visibilité sur une forme de la scène 3D implique des calculs d'intersection sur un nombre non négligeable de segments (60 segments en moyenne par forme testée dans notre prototype). Il convient ici de faire un choix entre, d'une part, limiter le nombre de formes candidates

au rôle d'obstacle, et d'autre part, limiter le nombre d'appels au R-Tree. Pour limiter au maximum les candidats obstacles, une solution est d'utiliser la boîte englobante de chaque segment impliqué dans les tests d'intervisibilité partielle et d'indiquer cette boîte englobante en paramètre d'appel au R-Tree. Cependant, cette méthode effectue parfois un trop grand nombre d'appels au R-Tree pour des segments proches les uns des autres. Pour limiter ces appels, nous recherchons non plus n englobants pour n segments, mais l'englobant de l'ensemble des segments. De ces deux solutions de filtrage, il est bien difficile de dire quelle est la plus judicieuse. Leurs efficacités peuvent considérablement varier d'une scène à l'autre, ou même d'un test d'intervisibilite à l'autre, en fonction du nombre et de la forme des objets présents dans la scène, mais aussi en fonction de la précision du résultat souhaité.

4.3.2.2.3.2 Tests d'intersection

Une fois les candidats obstacles sélectionnés, la suite n'est guère subtile, puisqu'après un test préliminaire sur la boîte englobante de l'obstacle, les faces de la forme sont testées une à une afin de vérifier si le segment traité les intersecte. Pour cette vérification, on commence par rechercher une intersection entre le segment et le plan supportant la face. Si une telle intersection existe, il convient ensuite de s'assurer qu'elle a bien lieu à l'intérieur du contour délimitant la face. Ces deux étapes sont détaillées en 4.3.2.4.

Notons tout de même que nous tenons compte de l'agencement des objets dans l'espace. En effet, lorsque nous lançons nos rayons, nous commençons toujours par les rayons les plus en altitude puisque la partie supérieure d'un objet est la plus souvent dégagée de tout obstacle. En revanche, lorsque nous effectuons des tests de visibilité totale (« la cible est-elle vue dans son intégralité ? »), nous commençons toujours par la partie inférieure de l'objet qui, statistiquement, est la plus masquée à la vue de l'observateur.

4.3.2.3 Calculs de visibilité totale

Un test de visibilité totale vérifie si toutes les parties d'un objet potentiellement visibles depuis l'observateur le sont effectivement. Notre prototype propose deux implémentations pour ce test de visibilité. La première est une implémentation directement inspirée du test de visibilité partielle. On ne vérifie cependant plus qu'au moins un rayon issu de l'observateur atteint la cible sans être intercepté par un obstacle, mais que tous les rayons issus de l'observateur atteignent la cible. Cette implémentation retourne un résultat qui ne peut être qu'approximatif, aussi proposons-nous un deuxième type d'implémentation qui ne repose plus sur des intersections de segments, mais sur des intersections de surfaces et de volumes. Le principe est de rechercher les intersections entre ce que nous appelons un volume de visibilité (ou frustum) et d'éventuelles formes 3D obstacles. Cette nouvelle approche amène son lot de méthodes et d'algorithmes que nous détaillons ici. Les paramètres d'entrée sont la forme 3D de l'observateur et la forme 3D de la cible. Le résultat retourné est booléen. Que l'observateur soit ponctuel ou pas n'a que peu de répercussions sur la mise en œuvre et la complexité de l'algorithme, seule la forme du frustum change.

4.3.2.3.1 Création du frustum

Un frustum est une forme géométrique utilisée en infographie pour décrire un volume de visibilité. Dans le cadre de nos tests de visibilité totale, nous appellerons frustum le volume de visibilité entre l'observateur et la cible, c'est-à-dire la forme observée. En infographie, un frustum apparaît comme une pyramide tronquée, ici sa forme, bien que d'allure pyramidale, est un peu plus complexe de part sa base (Figure 85).

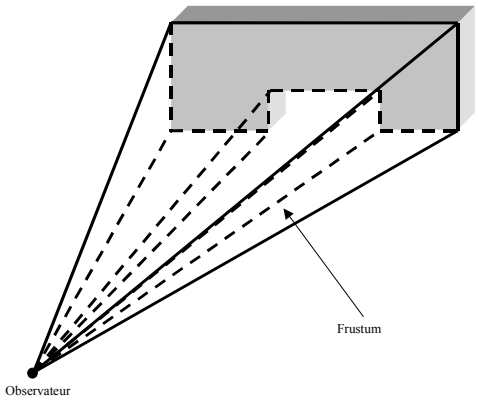


Figure 85 : Exemple de frustum dans SIG3D

Ces frustum sont simples de création : à partir d'une forme donnée, il suffit de récupérer les faces et les sommets utiles aux tests de visibilité totale. Ces derniers constituent la base du frustum et l'ajout de l'observateur comme sommet permet de reconstituer une forme à face pseudo-pyramidale représentative du frustum. Si l'observateur n'est pas ponctuel, le frustum prend une forme plus quelconque et plus complexe, mais les principes de l'algorithme restent les mêmes.

4.3.2.3.2 Recherche d'intersections entre le frustum et les formes 3D obstacles

Un objet est dit totalement visible depuis l'observateur si les faces potentiellement visibles par ce dernier sont effectivement vues dans leurs totalités. Les tests de visibilité totale recherchent donc des intersections entre les frustums des faces concernées et les formes à faces des objets « obstacles » de la scène. Tout comme pour les tests de visibilité partielle, un premier filtrage des objets candidats à l'intersection est fait par l'intermédiaire de l'index spatial.

Les tests d'intersection entre forme à faces et frustum sont d'algorithmie simple mais assez gourmands en calculs. Ils recherchent des intersections entre faces. En effet, les frustum et les formes obstacles sont des formes à faces, or une forme à face est un ensemble de faces : il y a donc intersection entre une forme à faces A et une forme à faces B si une face de A intersecte une des faces de B.

L'algorithme que nous avons utilisé pour rechercher les intersections entre faces s'inspire de celui proposé par Moller [MOLLER 1997]

Cet algorithme procède en deux temps :

- Le premier temps est consacré à la recherche des cas où la non-intersection est évidente, c'est-à-dire les cas où tous les sommets de la première face sont situés du même coté du plan portant la seconde face et vice versa.

- Les autres cas sont traités dans un second temps. La technique employée se rapproche de celle que nous présenterons au 4.3.2.4.2, les segments issus de l'intersection entre une face et le plan porteur de l'autre face sont projetés sur un des axes du repère. Les cas d'intersections entre les deux faces apparaissent en étudiant les recouvrements de ces projections.

Notons le cas particulier où la face est entièrement contenue dans la forme à faces. En fonction des objectifs de l'algorithme, ce cas peut être considéré comme un cas d'intersection (la forme à faces est alors dite « pleine ») ou comme un cas de non-intersection (la forme à faces est dite « vide »).

L'orientation des faces de la forme à faces est utile pour identifier ce cas particulier. Une technique possible est de lancer un rayon depuis un des sommets de la forme A dont on veut savoir si elle est à l'intérieur d'une autre forme B. On comptabilise les intersections entre le rayon et les cotés « intérieurs » des faces. Si le nombre de faces ainsi touchées est impair, la forme A est à l'intérieur de la forme B.

4.3.2.4 Utilitaires pour calculs d'intervisibilité

Les algorithmes présentés ici sont utilisés par les calculs d'intervisibilité. Ils sont couramment utilisés en synthèse d'image et notamment dans la technique dite de lancer de rayon [FOLEY 1994].

4.3.2.4.1 Intersection entre un segment et un plan

Soit (x0,y0z0) et (x1,y1,z1) les extrémités du segment. L'équation paramétrique de ce dernier est :

$$\begin{aligned} x &= x0 + t(x1-x0) \\ y &= y0 + t(y1-y0) \quad (1) \\ z &= z0 + t(z1-z0) \end{aligned}$$

D'autre part, l'équation d'un plan s'exprime à partir d'un vecteur normal au plan et d'un point du plan, elle est de la forme :

$$Ax + By + Cz + D = 0 \quad (2)$$

En substituant (1) dans (2) on obtient :

$$t = \frac{(Ax_0 + By_0 + Cz_0 + D)}{(A\Delta x + B\Delta y + C\Delta z)}$$

avec $\Delta x = x_1 - x_0$, $\Delta x = y_1 - y_0$, $\Delta x = z_1 - z_0$

Le segment intersecte le plan si et seulement si $0 < t < 1$.

4.3.2.4.2 Appartenance d'un point à une face

Une technique classique et pratique utilisée ici consiste à projeter les arêtes de la face et le point d'intersection entre le plan et le segment sur l'un des trois plans de projection du système de coordonnées. Couramment, on utilise l'axe de projection qui conduit à la plus grande projection. Il suffit alors d'abandonner la coordonnée correspondante pour les points extrémités des arêtes de la face et pour le point d'intersection.

On se retrouve ainsi dans une des situations 2D suivantes (Figure 86):

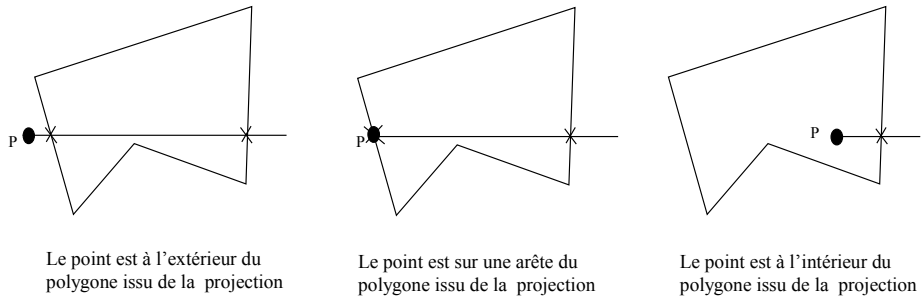


Figure 86 : Appartenance d'un point à un polygone

Pour distinguer les trois situations, un méthode efficace est de comparer la position de chacune des arêtes du polygone par rapport à un axe quelconque issu du point P (en pratique on utilise l'axe i du repère courant pour des raisons de simplicité et d'efficacité). Pour ce faire, on comptabilise le nombre d'intersections rencontrées le long de l'axe. Si ce nombre est pair, alors le point P est extérieur au polygone, sinon il lui est intérieur.

4.3.2.4.3 Intersection avec des volumes simples

Il est très utile de posséder des méthodes de calculs d'intersection sur des volumes simples. Il est parfois plus judicieux de procéder à un filtrage préliminaire des objets géographiques par des tests sur des primitives approchant de manière large la forme réelle de l'objet. Les coûteux algorithmes ne sont alors appliqués qu'aux formes complexes des objets issus de cette première sélection.

4.3.2.4.3.1 Intersection entre un segment et une sphère

La sphère est une primitive volumique très souvent utilisée en synthèse d'image. Cette popularité tient sans doute au fait qu'il s'agit là d'une forme somptueuse mêlant douceur et perfection. Poésie mise à part, il se trouve que, pour les techniques de lancers de rayons, les calculs d'intersections entre segments et sphères sont parmi les moins coûteux.

L'équation de la sphère est du type :

$$(x-a)^2 + (y-b)^2 + (z-c)^2 = r^2 \qquad (3)$$

En développant et en substituant (1) dans (3) on obtient :

$$(\Delta x^2 + \Delta y^2 + \Delta z^2)t^2 + 2t[\Delta x(x_0 - a) + \Delta y(y_0 - b) + \Delta z(z_0 - c)] + (x_0 - a)^2 + (y_0 - b)^2 + (z_0 - c)^2 - r^2 = 0$$

C'est une équation quadratique en t, dont les coefficients constants sont déduits de l'équation de la sphère et de celle du rayon visuel et qui peut donc être résolue comme une équation du second degré classique.

4.3.2.4.3.2 Intersection entre un segment et une « bounding-box »

Une « bounding-box » est une forme parallélépipédique dont les arêtes sont parallèles aux axes du repère. Elles sont très utilisées comme boîtes englobantes pour approcher les formes 3D quelconques. Les tests d'intersections sur ces primitives sont là aussi très rapides. L'algorithme utilisé compare la position des extrémités du segment par rapport à chacun des plans composant la bounding box.

4.3.2.5 Autres intersections

En dehors des calculs d'intersections directement impliqués dans la résolution des problèmes de visibilité, nous avons également implémenté quelques méthodes d'intersections supplémentaires classiques et utiles pour répondre à certaines requêtes géométriques.

Parmi ces méthodes nous retrouvons des calculs d'intersections sur une sphère :

- intersection d'une sphere et d'un englobant
- intersection d'une sphere et d'une face
- intersection d'une sphere et d'une forme à faces

mais aussi des calculs de distance entre deux points, entre un point et une forme à faces, entre deux segments, entre un point et un segment ou encore entre deux formes à faces.

4.3.2.6 Evaluation qualitative des algorithmes de calculs d'intervisibilité

Nous avons cherché à évaluer le comportement de nos algorithmes d'intervisibilité à l'aide de quelques tests de performances. Les deux premiers tests opèrent sur des cartes de visibilité partielle. Le premier compare le temps de traitement de l'algorithme par rapport à la surface de la carte de visibilité ou au nombre d'objets testés. Le second établit le pourcentage de temps passé dans les principales étapes de l'algorithme. Le troisième test est un comparatif entre les calculs d'intervisibilité partielle et ceux d'intervisibilité totale.

4.3.2.6.1 Tests sur cartes de visibilité partielle

La série de tests que nous avons mis en œuvre concerne la place des Vosges à Paris. Nous avons dressé des cartes de visibilité pour deux observateurs dont les positionnements sont indiqués par la Figure 87.

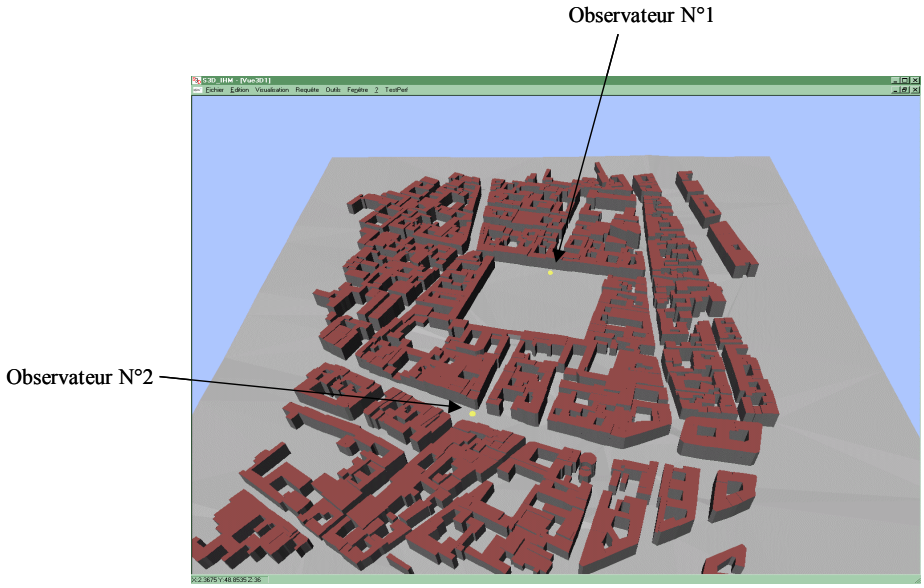


Figure 87 : Positionnement des Observateurs

Notons tout de suite que la densité de bâtiment n'est pas la même autour des deux observateurs. L'observateur numéro 1 est placé à l'intérieur de la place des Vosges, qui représente une surface d'environ 22500 m² vierge de tout bâtiment. Nous n'avons pas modélisé ici les arbres présents dans cette place, ni la statue de Louis XIII à cheval trônant en son centre. L'observateur numéro 2 est au centre de la rue Saint-Antoine et donc plus directement bordé par des bâtiments.

4.3.2.6.1.1 Temps de calculs pour établir une carte de visibilité partielle

Le premier des tests donne une idée des temps de calculs pour dresser une carte de visibilité d'une surface donnée. Nous avons, pour chaque observateur, dressé successivement dix cartes de visibilité partielle pour des surfaces allant de 2500 à 250000 m² (soit des carrés centrés sur l'observateur de 100 à 1000 mètres de coté). Les temps nécessaires pour établir chacune de ces cartes sont présentés Figure 88.

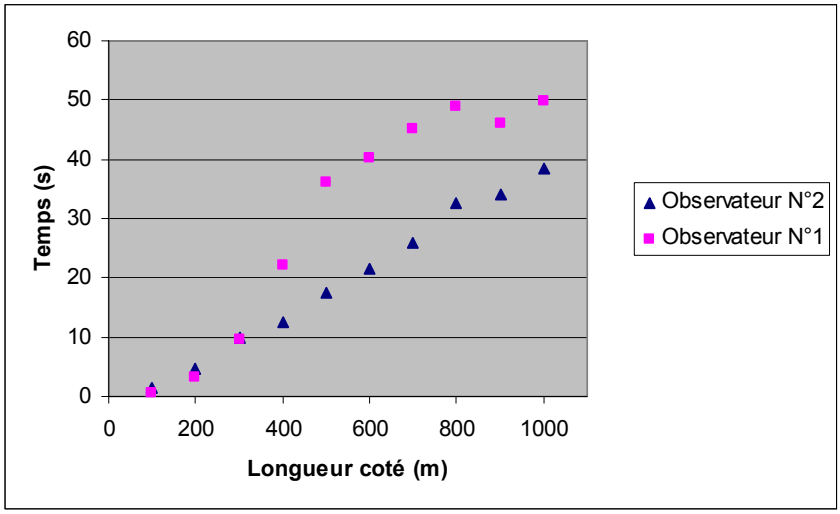


Figure 88 : Temps nécessaires à l'établissement de cartes de visibilité de surfaces variées

La différence de densité de bâtiment autour des observateurs est clairement visible sur cette figure :

La forme du nuage de points pour l'observateur n°1 connaît une cassure pour des cartes de longueur de coté supérieure à 400 m. La première partie de la courbe ressemble à celle d'une fonction exponentielle alors que la seconde partie retrouve une certaine linéarité. Nous expliquons ce comportement par la diminution de croissance de la surface vide de bâtiment prise en compte (la place des Vosges) au fur et à mesure que la taille de la carte de visibilité grandit. Pour des valeurs de coté supérieures à 400 m, la surface de la place des Vosges est entièrement prise en compte, le comportement du nuage de point devient linéaire.

La forme du nuage de points pour l'observateur n°2 ne connaît pas de telle cassure. La prise en compte de la place des Vosges arrive lorsque le nombre d'objets à tester est suffisamment grand pour masquer l'influence de cet écart de densité.

Quand nous comparons les temps non plus en fonction des surfaces des cartes de visibilité mais du nombre d'objets testés, les nuages de points pour les deux observateurs ont alors des formes beaucoup plus similaires (Figure 89).

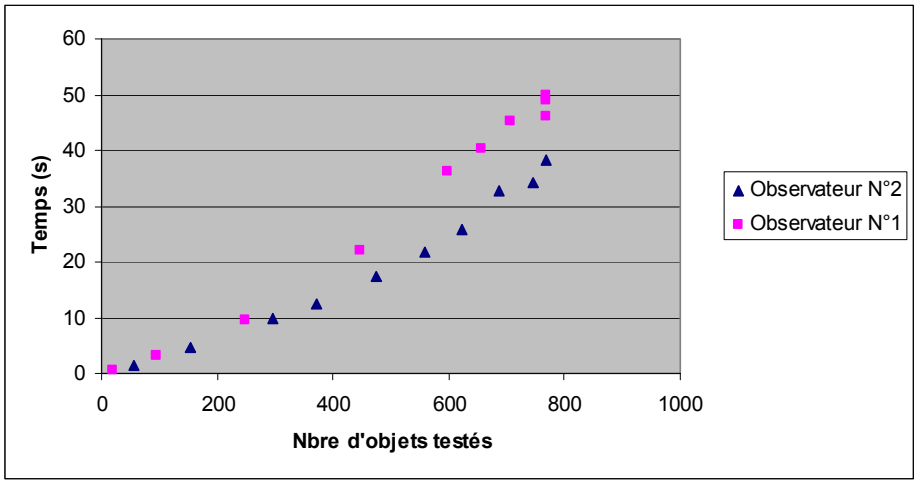


Figure 89 : Temps nécessaire à l'établissement de cartes de visibilité en fonction du nombre d'objets testés

4.3.2.6.1.2 Répartition du temps de calcul

Les trois étapes principales de l'algorithme de visibilité partielle que nous avons présentés sont la discrétisation des objets cibles, la recherche des objets pouvant être des obstacles et les lancers de rayons en eux mêmes. La Figure 90 résume la répartition du temps de calcul entre ces trois étapes. Les mesures ont été obtenues lors de la détermination de cartes de visibilité de 100 et 1000 m de coté.

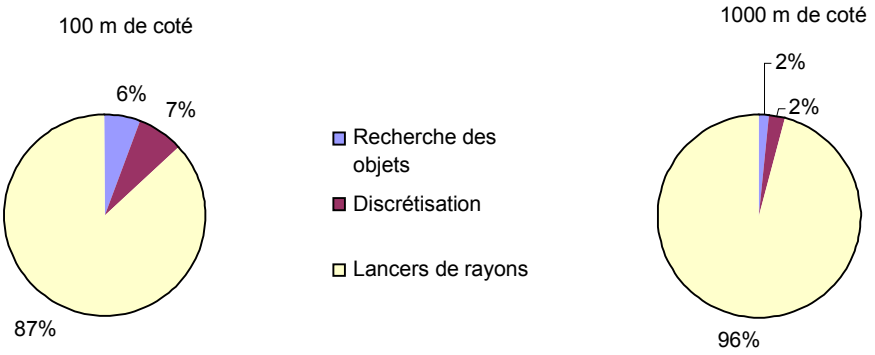


Figure 90 : Répartition des temps de calcul entre les différentes étapes de l'algorithme de calcul de visibilité partielle

Les lancers de rayons occupent l'immense majorité du temps de calcul. Les temps consacrés à la discrétisation et à la recherche des objets éventuellement obstacles sont très réduits. La différence d'une carte de visibilité à l'autre réside dans le pourcentage d'objets vus sur le nombre total d'objets testés. Le nombre de rayons lancés est moins important pour un objet vu que pour un objet non vu puisque le test se termine dès qu'un rayon atteint la cible sans être intersecté.

4.3.2.6.2 Comparaison visibilité partielle/visibilité totale

Les algorithmes de visibilité partielle travaillent sur des lancers de rayons, ceux de visibilité totale s'appliquent sur des intersections entre polyèdres. Nous pourrions penser que les premiers sont toujours plus rapides que les seconds, or la Figure 91 apporte une nuance.

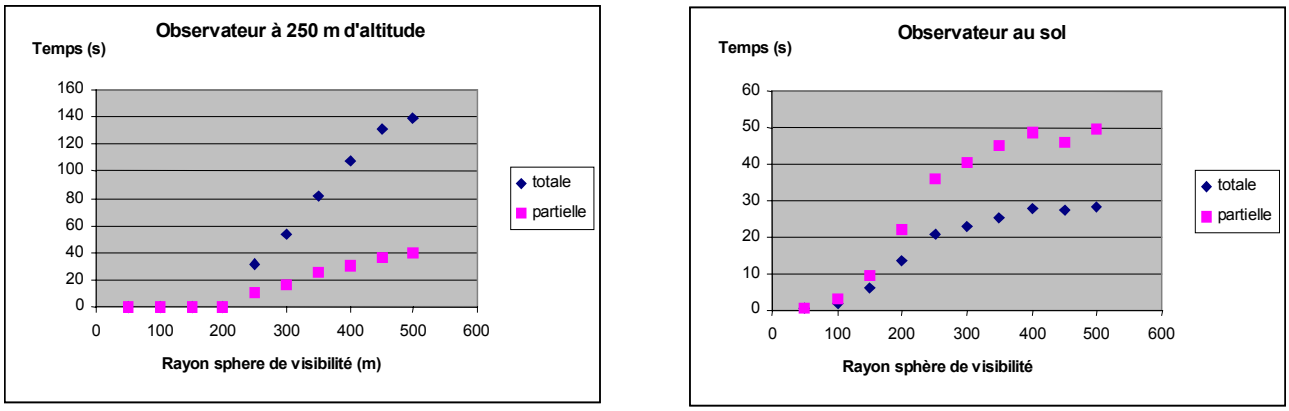


Figure 91 : Comparatif visibilité partielle / visibilité totale

Les deux graphes de la Figure 91 portent sur l'établissement de cartes de visibilité toujours autour de la place des Vosges. Le graphe de droite compare les temps nécessaires à la mise en place de la carte de visibilité totale et partielle pour un observateur à 250 m d'altitude, le graphe de gauche fait la même comparaison pour un observateur au sol.

Les conclusions des deux comparatifs sont radicalement différentes. Il est plus rapide de dresser une carte de visibilité totale au sol alors qu'une carte de visibilité partielle sera plus rapide à établir pour un observateur à 250 m d'altitude. L'explication en est simple :

- Lors de tests de visibilité partielle en altitude, presque tous les bâtiments sont visibles, le nombre de rayons lancés est moins important et les temps de traitements sont donc plus courts. Au sol les bâtiments non visibles sont majoritaires, pour chaque bâtiment un nombre maximal de rayons sont lancés.

- Lors de tests de visibilité totale au sol, les bâtiments étant tous côte à côte, il est très rare de pouvoir voir des bâtiments dans leur intégralité, ils sont presque toujours partiellement masqués par une partie du bâtiment voisin. Les tests de visibilité totale sont donc très vite interrompus puisqu'ils s'arrêtent dès la première intersection entre le frustum et un objet de la scène (chapitre 4.3.2.3.2). Lorsque l'observateur surplombe la scène, la vue est plus dégagée et le nombre d'objets à tester est plus important avant de trouver une intersection avec le frustum.

Ce sont essentiellement des tests de visibilité partielle qui seront pris en compte comme contraintes appliquées sur les calculs de recherches de trajectoire que nous allons maintenant évoquer.

4.3.3 Calculs de trajectoires libres

4.3.3.1 Introduction

Puisque notre prototype de système d'information géographique est tridimensionnel, nous avons choisi d'étendre la recherche de trajectoire à la troisième dimension et de ne plus nous contenter de déplacements assujettis au sol. De ce fait, notre prototype se démarque fortement des outils conventionnels proposés par les applications SIG courantes et ouvre la voie vers de nouvelles applications, par exemple la préparation d'approche héliportée.

Le calcul de trajectoire libre s'effectue en deux étapes. La première étape permet de modéliser l'ensemble de la scène 3D en une grille. Chaque cellule de la grille représente une portion de l'espace, cette portion d'espace peut être vide ou occupée par un objet géographique. Nous affectons à chaque cellule de la grille une contrainte dépendant de l'état de la portion d'espace qu'elle représente, c'est-à-dire de la nature de la donnée géographique occupant la portion d'espace associée. La valeur de cette contrainte est également liée au type de mobile utilisé, au relief du terrain ou encore à quelques critères de visibilité.
La recherche de trajectoire est ensuite effectuée à partir de cette grille valuée. Cette grille est assimilable à un graphe (cf. 3.4.3) sur lequel s'appliquent les algorithmes classiques de recherche de chemin optimal.

C'est la classe S3D_AlSiteContraint qui au sein de notre prototype est consacrée à l'implémentation des méthodes dédiées à la recherche de trajectoires libres sur sites contraints.

4.3.3.2 Création d'une grille des contraintes

4.3.3.2.1 Les contraintes prises en compte

Les facteurs qui influencent le temps de parcours d'un objet mobile, tel un piéton, un véhicule ou un aéronef, sont des facteurs liés au mobile en déplacement ou liés à la nature des zones traversées par le mobile.

Parmi les caractéristiques du mobile, nous avons retenu :

- Le poids
- La vitesse maximale
- La longueur
- La largeur
- La hauteur

Tous ces paramètres sont modifiables à loisir, mais pour faciliter la prise en main du prototype nous proposons quelques mobiles types aux caractéristiques prédéfinies et directement utilisables pour la recherche de trajectoire :

- Piéton
- Voiture
- Camion
- Tank
- Hélicoptère
- Avion

De plus, chaque mobile possède une variable de contrainte de déplacement caractéristique du type de terrain traversé. Les types de terrains aujourd'hui pris en compte sont :

- Eau
- Route
- Marécage
- Neige
- Champs
- Forêt

Les éléments du sur-sol sont d'éventuels obstacles dont il faut tenir compte :

- Bâtiment
- Pont
- Chemin de fer
- Végétation

Enfin nous tenons compte du relief. Les pentes sont calculées à partir du MNT et prises en compte dans les algorithmes, chaque mobile possède une variable représentative de sa facilité à se mouvoir le long de pentes.

Les scénarios abordés durant la phase d'expérimentation du prototype ne mettent pas en œuvre l'ensemble de ces contraintes potentielles, mais les algorithmes proposés les prennent toutes en compte. L'ajout de contraintes supplémentaires n'entraîne que des modifications mineures dans l'implémentation de la classe S3D_AlSiteContraint.

D'autre part, afin de limiter les temps de calcul, il est possible de restreindre la zone de recherche sur laquelle sont effectués les calculs de trajectoire. Toujours dans cette optique de réduction des temps de calculs, nous permettons également de paramétrer la précision du résultat recherché. Il est inutile de rechercher des trajectoires au mètre près lorsque les données en entrée sont modélisées avec une résolution supérieure à la dizaine de mètres.

4.3.3.2.2 Discrétisation des objets contraignant le déplacement

La recherche de trajectoire a donc pour support une grille, 2D ou 3D, représentant l'ensemble de la scène géographique. Cette grille, dite grille des contraintes, est construite en discrétisant les objets (ponctuels, linéaires, surfaciques ou volumiques) de la scène en cellules de la grille. Ces objets constituent autant de contraintes de déplacement à respecter par le mobile. Les cellules de la grille sont affectées de coefficients tenant compte de la nature des objets et du type de mobile.

4.3.3.2.2.1 Discrétisation des éléments linéaires

Le traitement des éléments linéaires attribue une valeur de contrainte spécifique à chaque cellule de la grille traversée par l'élément linéaire. La détermination des cellules traversées par une ligne au sein d'une grille est une technique depuis longtemps maîtrisée par les infographes. Ils l'emploient notamment pour retrouver les pixels impliqués dans le dessin d'un segment de droite. Nous nous sommes donc inspirés de leurs travaux pour traiter de nos problèmes de discrétisation d'éléments linéaires. Les routes sont l'exemple même d'objets pouvant être modélisés par des éléments linéaires, et dont on se doit de tenir compte dans un calcul de trajectoire.

En 2D, la stratégie la plus simple de discrétisation d'un segment de droite est de représenter la pente m en $\partial x/\partial y$, d'incrémenter la valeur de x par 1 en commençant par le point d'extrémité gauche, de calculer la valeur y par $y_i = mx_i + B$ pour chaque x_i, et de marquer les cellules aux points (x_i, Arrondi(y_i)), où Arrondi(y_i) est la partie entière de ($0,5 + y_i$). Cette stratégie de discrétisation utilise les cellules les plus proches, c'est-à-dire les cellules dont la distance au segment de droite idéal est la plus petite. Cette stratégie simpliste est inefficace, car chaque itération nécessite une multiplication réelle, une addition, et un calcul d'arrondi. On peut éliminer la multiplication en utilisant la relation suivante :

$y_i+1 = mx_i+1 + B = m(x_i+dx) + B = y_i + mdx$

Et si $dx = 1$, alors $y_i+1 = y_i + m$.

Les valeurs de x_i et y_i sont définies en fonction de leurs valeurs précédentes. C'est ce qui définit un algorithme incrémental, qui, à chaque étape, fait un calcul fondé sur le résultat de l'étape précédente [FOLEY 1994] (Figure 92).

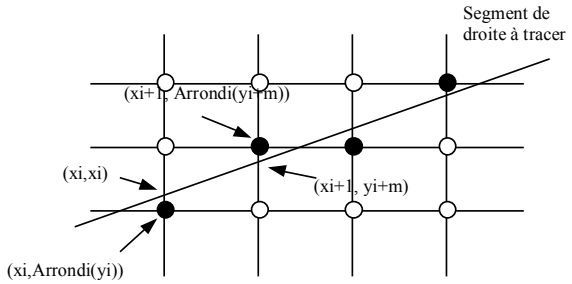


Figure 92 : Calcul incrémental (une cellule est identifiée par son coin inférieur gauche) [FOLEY 1994].

Les inconvénients de cet algorithme sont dus au temps d'exécution de la fonction arrondi et au fait que les variables y et m sont de type réel ou de type fraction binaire. Bresenham [BRESENHAM 1965] a développé un algorithme classique intéressant qui utilise uniquement l'arithmétique entière et évite ainsi l'utilisation de la fonction arrondi. Il calcule de façon incrémentale (x_i+1, y_i+1) en utilisant les

résultats du calcul de (x_i, y_i). De ce modèle fut dérivée une technique légèrement différente qui est la technique selon les demi-points, proposée d'abord par Pitteway [PITTWAY 1967], puis adaptée par Van Aken [VAN AKEN 1984] et d'autres chercheurs.

On suppose dans l'explication qui suit que la pente du segment est comprise entre 0 et 1, les autres pentes pouvant être traitées par symétrie suivant les axes. Prenons le segment représenté en Figure 93, où la cellule précédemment choisie est notée par un cercle noir et les deux cellules candidates pour l'étape suivante sont représentées par des cercles vides. Supposons que nous venions de choisir la cellule P de coordonnées (x_p, y_p). A présent nous devons choisir entre la cellule située un pas à droite (appelée cellule Est, E) et la cellule située un pas à droite et un pas vers le haut (appelée cellule Nord-Est, NE). Soit Q le point d'intersection du segment de droite idéal avec la droite $x = x_p + 1$. Dans le modèle selon les demi-points, on détermine de quel côté du segment se trouve le demi-point M. Il est facile de voir que si le demi-point se situe au-dessus du segment, la cellule E est la plus proche du segment, et que si le demi-point se situe par contre en dessous du segment, la cellule NE est plus proche du segment. Le segment peut soit passer entre E et NE, soit laisser les deux cellules du même côté, mais dans tous les cas, le test du demi-point choisit la cellule la plus proche. De plus, l'erreur, c'est-à-dire la distance verticale entre cellule choisie et le segment idéal, est inférieure ou égale à ½.

Ces optimisations algorithmiques ne s'appliquent que dans le cas d'un calcul de trajectoire assujettie au sol, c'est-à-dire à partir d'une grille de contrainte 2D et non pas 3D (même si le relief peut être pris en compte).

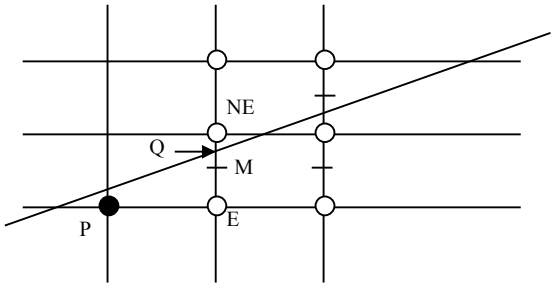


Figure 93 : Détermination des cellules par l'algorithme des demi-points [FOLEY 1994].

4.3.3.2.2.2 Discrétisation des éléments surfaciques et volumiques

La discrétisation des formes volumiques ou surfaciques fait à nouveau appel aux techniques de lancer de rayon. Nous faisons correspondre à l'espace géographique étudié une grille régulière tridimensionnelle. Discrétiser une forme 3D quelconque revient à déterminer quels sont les cubes élémentaires (ou cellules) de cette grille occupés par une partie de la forme 3D. Un moyen simple pour y parvenir est de lancer des rayons dont les vecteurs directeurs sont colinéaires à un des axes de la grille régulière 3D (Figure 94).

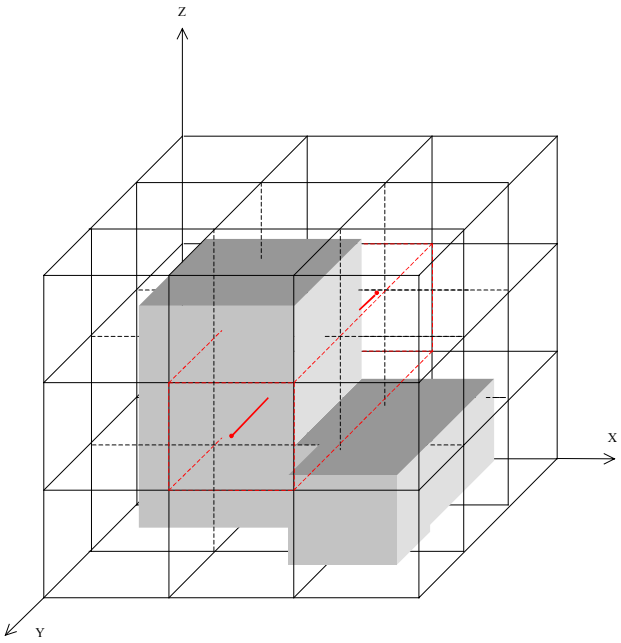


Figure 94 : Discrétisation d'une forme quelconque dans une grille régulière 3D

Les lancers de rayon sont des opérations relativement coûteuses, il convient donc d'en limiter le nombre. Pour ce faire, nous nous limitons à lancer des rayons dans la zone facilement déductible de la boîte englobante de la forme. Les rayons sont lancés le long du plus grand côté de la boîte englobante afin de diminuer davantage encore le nombre de lancers à effectuer.

Pour chaque rayon lancé dans la zone de recherche, le calcul d'intersection repose sur les algorithmes d'intersection entre segment et face présentés en 4.3.2.2.3.2. Les intersections sont stockées et triées dans une liste. Le parcours de cette liste permet d'en déduire les cellules de la grille se trouvant à l'intérieur de la forme et celles se situant à l'extérieur de celle-ci.

Si la zone de la grille tridimensionnelle correspondant à l'englobant de la forme possède NX*NY*NZ cellules avec NZ le nombre de cellules suivant l'axe vertical, alors le processus que nous venons de décrire est réitéré NZ*Min(NX,NY) fois.

4.3.3.2.3 Les critères de visibilité

Nos recherches de chemin optimal offrent trois possibilités vis-à-vis des critères de visibilité :

- La première et la plus simple ignore les contraintes de visibilité.
- La seconde propose de ne retourner que des trajectoires le long desquelles le mobile en mouvement n'est jamais vu depuis le ou les observateurs désignés.
- La troisième retourne uniquement les trajectoires pouvant être entièrement surveillées par les observateurs.

Pour tenir compte de ces critères de visibilité, nous utilisons des structures appelées Viewshed (paragraphe 3.3) qui permettent de recenser dans une scène 3D tous les points visibles depuis l'observateur. L'obtention de ces structures se fait par le biais d'algorithmes triviaux mais qui ont le mérite de tenir compte de la forme réelle des objets de la scène. Nous ne nous limitons pas à de simples modèles numériques d'élévation.

Concrètement, des tests d'intervisibilité sont effectués sur chaque cellule de la grille tridimensionnelle associée à l'espace étudié. Le centre d'une cellule forme avec l'observateur un segment dont on a vérifié qu'il n'est pas intersecté par un objet de la scène. Ce sont des algorithmes relativement lourds, il est certainement possible d'optimiser ces derniers pour limiter le nombre de tests en s'inspirant des techniques déjà existantes en matière de reconstitution de structures de visibilité sur MNT, c'est-à-dire en tenant compte des zones d'ombre en arrière d'une forme 3D par rapport à l'observateur. Malheureusement, nous n'avons pas eu le temps nécessaire, au cours de cette thèse, de développer davantage cette piste.

4.3.3.3 Calcul d'une trajectoire optimale

4.3.3.3.1 Création d'une grille des coûts cumulés

A partir de la grille des contraintes obtenues après les étapes de discrétisation et d'intégration des structures de visibilité, nous nous appuyons sur une nouvelle grille, la grille des coûts cumulés, qui est le support direct permettant de reconstituer la trajectoire optimale tant recherchée. Prenant comme référence une cellule initiale, cette grille de coûts cumulés indique le coût minimal nécessaire pour relier chaque cellule de la grille à la cellule initiale.

Les grilles sont assimilées à des graphes (Figure 95), la valeur d'un arc joignant deux nœuds est calculée en fonction du type rencontré d'adjacence.

Dans le cas de recherche d'une trajectoire 2D plaquée au sol :

Si les nœuds sont représentatifs de cellules adjacentes par les côtés, alors l'arc pour rejoindre les deux nœuds porte la valeur de coût de déplacement suivante :

$$Coût_{ARC} = (Contrainte_{SI} + Contrainte_{SF})/2$$

Si les nœuds sont représentatifs de cellules adjacentes par les sommets, alors l'arc pour rejoindre les deux nœuds porte la valeur de coût de déplacement suivante :

$$Coût_{ARC} = 1.4*(Contrainte_{SI} + Contrainte_{SF})/2$$

Dans le cas de recherche de trajectoire 3D :

Si les nœuds sont représentatifs de cellules adjacentes par les faces, alors l'arc pour rejoindre les deux nœuds porte la valeur de coût de déplacement suivante :

$$Coût_{ARC} = (Contrainte_{SI} + Contrainte_{SF})/2$$

Si les nœuds sont représentatifs de cellules adjacentes par les côtés, alors l'arc pour rejoindre les deux nœuds porte la valeur de coût de déplacement suivante :

$$Coût_{ARC} = 1.4*(Contrainte_{SI} + Contrainte_{SF})/2$$

Si les nœuds sont représentatifs de cellules adjacentes par les sommets, alors l'arc pour rejoindre les deux nœuds porte la valeur de coût de déplacement suivante :

$$Coût_{ARC} = 1.75*(Contrainte_{SI} + Contrainte_{SF})/2$$

Sur ce graphe est appliqué un algorithme de Dijkstra classique pour établir la grille des coûts cumulés. Les nœuds du graphe, et donc les cellules de la grille des coûts cumulés, portent le coût total nécessaire pour les rejoindre depuis la cellule de départ. La Figure 95 illustre le passage d'une grille des contraintes 2D à une grille des coûts cumulés 2D. Notons que nous avons choisi ici comme cellule de départ la cellule centrale. Dans la grille des coûts cumulés, nous avons attribué une valeur initiale et arbitraire de 10 à cette cellule pour illustrer le fonctionnement de l'algorithme. En pratique, une valeur de 0 est toujours associée au point de départ sur cette grille des coûts cumulés.

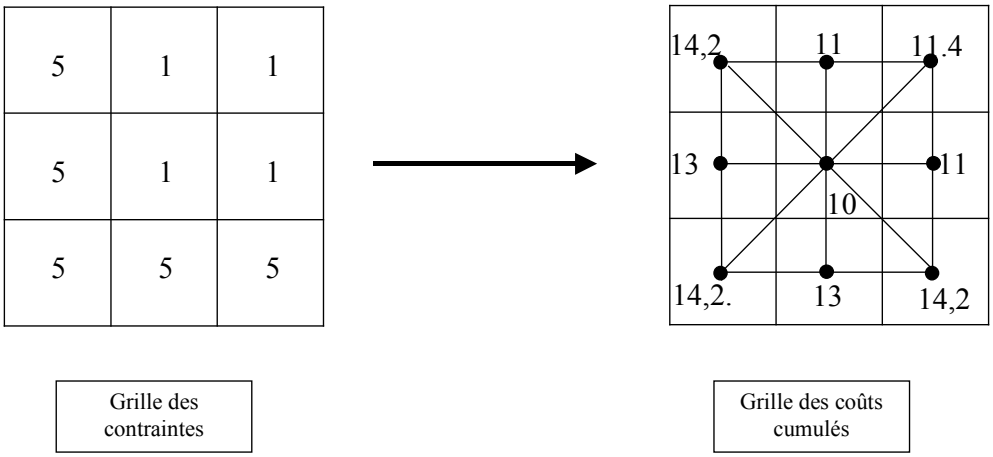


Figure 95 : Transformation d'une grille des contraintes en grille des coûts cumulés

Comme expliqué en 3.4.2.3, le point critique de l'algorithme réside dans le choix du nœud suivant à traiter. Nous nous sommes contentés ici d'une implémentation classique.

4.3.3.3.2 Prise en compte du relief

C'est à ce stade que les contraintes liées au relief sont prises en compte pour le calcul d'une trajectoire assujettie au sol. Il en effet nécessaire de connaître le point de départ de la trajectoire pour déterminer de quelle manière celle-ci va être influencée par le relief. Pour ce faire, nous déterminons l'altitude du relief au centre de chaque cellule de la grille dans laquelle a été discrétisée la scène 3D. La valeur des arcs du graphe établi durant le processus expliqué au paragraphe précédent est modifiée en fonction de la différence d'altitude entre les deux centres de cellules associés aux nœuds initiaux et finaux des arcs. En d'autres termes, la pente entre deux cellules de la grille permet d'ajuster la valeur du coût pour joindre ces deux cellules. Le graphe associé à la grille devient alors orienté.

TriGO gère cet ajustement dû au relief de la manière suivante :

- Si la pente est inférieure à un seuil minimal, alors le relief n'influence pas le coût de déplacement.
- Si la pente est comprise entre un seuil minimal un seuil maximal, alors le relief est pris en compte en affectant un facteur de ralentissement au coût de déplacement.
- Si la pente est supérieure au seuil maximal, le passage d'une cellule à l'autre (d'un nœud à l'autre) n'est plus possible.

Les seuils et le facteur de ralentissement sont dépendants de la nature du mobile. Notre prototype utilise des constantes pour chacun de ces paramètres; cependant, l'utilisation de fonctions, linéaires ou non linéaires, peut être envisagée pour obtenir des résultats plus réalistes.

4.3.3.3.3 Extraction de la trajectoire optimale

Cette étape est la plus simple du processus de recherche de trajectoire libre. A partir de la grille des coûts cumulés reconstituée, il suffit de partir de la cellule finale, objectif de notre recherche de trajectoire, et de reprendre en sens inverse le chemin à parcourir jusqu'à la cellule initiale de proche, en proche en choisissant à chaque fois la cellule de moindre coût cumulé.

4.3.3.4 Evaluations autour des recherches de trajectoire

Trois grandes étapes jalonnent un processus de recherche de trajectoire complet. La première d'entre elles est la création de la grille des contraintes (4.3.2.2.2), la deuxième est la prise en compte des critères de visibilité (viewshed) et la troisième est l'application des algorithmes de recherche d'un chemin optimal sur le graphe formé à partir des deux étapes précédentes.

Nous avons cherché à connaître le temps consacré à chacune de ces trois étapes, pour ce faire nous avons effectué des recherches de trajectoires dans le secteur de la place des Vosges (Figure 96). La zone d'intérêt a une longueur d'approximativement un kilomètre de long pour une largeur d'environ sept cents mètres. Les pas de discrétisation utilisés vont de 1 à 5 m en largeur comme en longueur. Les critères de visibilité pris en compte dans ce test concernent un champ de 50 m de rayon autour d'un unique observateur.

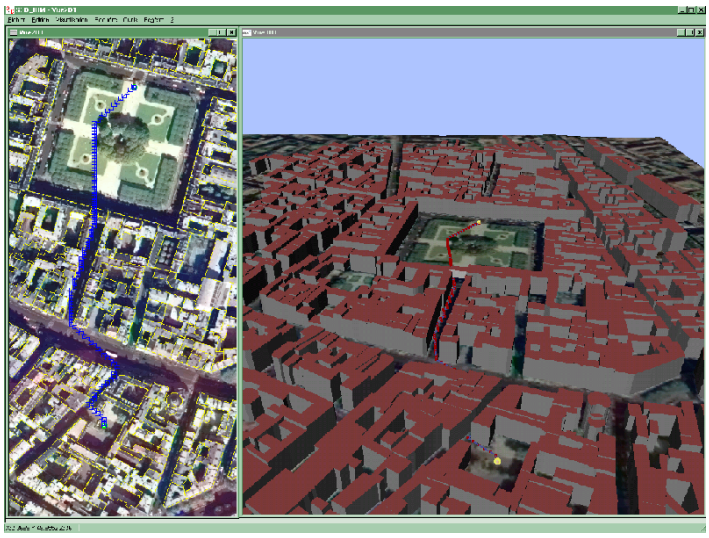


Figure 96 : Recherche d'un parcours piéton autour de la place des Vosges

La Figure 97 donne un aperçu des temps consacrés à chacune de ces étapes.

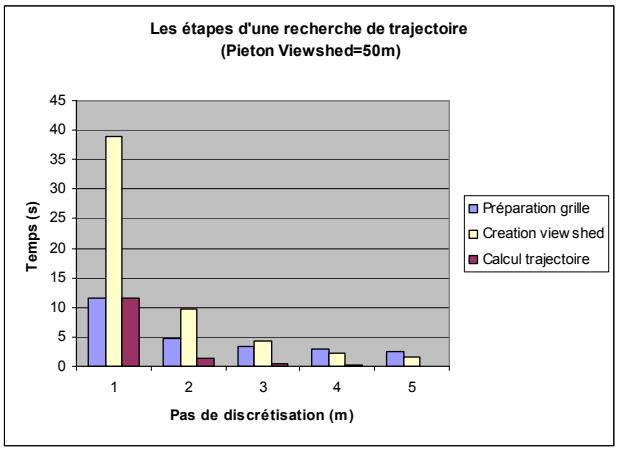


Figure 97 : Les étapes d'une recherche de trajectoire

Sur cette figure nous constatons que plus on tend vers des pas de discrétisations élevés, plus les temps consacrés à la prise en compte des critères de visibilité prédominent. C'est l'étape de discrétisation en une grille des contraintes qui croît le moins rapidement avec la résolution du calcul et donc avec le nombre de cellules des grilles des contraintes (4.3.2.2.2) et des coûts cumulés (4.3.3.3.1).

Le diagramme Figure 98 montre que chacune des étapes consacrées à la recherche de trajectoire évolue linéairement en fonction de la complexité des grilles des contraintes et des grilles des coûts cumulés associées.

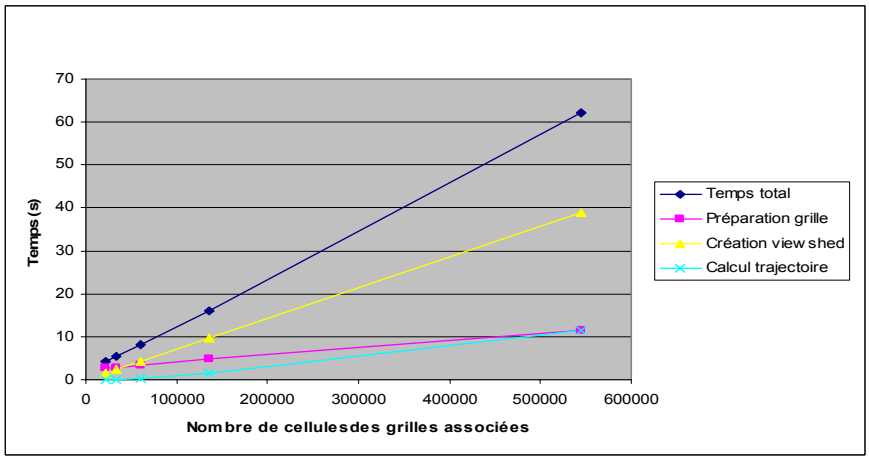


Figure 98 : Temps pour établir une trajectoire en fonction de la complexité des grilles des contraintes et des grilles des coûts cumulés associées à la zone de recherche

Les temps que nous indiquons sur nos graphes, établis à partir d'un pentium III 800 Mhtz, ne sont là que pour donner un ordre de grandeur et un indicateur de comparaison aux processus évalués dans ce chapitre. Les temps de calculs sont trop dépendants du type de matériel utilisé pour pouvoir raisonner sur leurs valeurs absolues.

Dans le prochain chapitre que nous consacrons à l'expérimentation et à la validation du prototype TriGO, nous avons davantage cherché à savoir si les besoins des utilisateurs étaient couverts par les fonctionnalités implémentées dans notre prototype plutôt que de nous focaliser sur des critères de performances pures. Cependant, l'ergonomie et la réactivité du logiciel étant des considérations importantes aux yeux de futurs opérateurs, nous en avons naturellement tenu compte dans la mise en œuvre de notre prototype.

4.4 Expérimentation et validation

Où l'œil avisé d'expérimentateurs avertis
eut à juger de la validité du prototype.

4.4.1 Présentation

Du 2 janvier au 31 mai 2002 nous avons eu l'occasion de soumettre notre prototype au jugement de divers utilisateurs potentiels intéressés par la mise en œuvre d'un système d'information géographique 3D.

Notre prototype TriGO est né de la formulation par l'ancienne CEGN (Cellule d'Etude en Géographie Numérique de la Délégation Générale pour l'Armement) d'un besoin commun à plusieurs départements militaires en matière d'outil de visualisation et d'analyse 3D. Chargé par la CEGN de la réalisation du prototype d'un tel outil, EADS S&DE à mis en place en fin de développement une campagne d'expérimentation afin de valider auprès des organismes initialement demandeurs les options et fonctionnalités privilégiées au cours de l'étude.

Les profils des personnes qui ont évalué TriGO sont assez hétérogènes puisque nous nous sommes adressés à des agents opérationnels comme à des officiers d'états majors, à des adjudants chefs comme à des colonels. Les organismes impliqués dans cette phase d'expérimentation furent, entre autres, le 13ème Régiment de parachutistes (RDP), le Service Technique de l'Armée de Terre (STAT), le Commandement des Opérations Spéciales (COS), le Groupement d'Intervention de la Gendarmerie Nationale (GIGN) ou encore les Forces Nucléaires. Pour des raisons de confidentialité, nous ne pouvons pas citer ici l'ensemble des organismes ayant participé à nos expérimentations et nous resterons volontairement imprécis sur la provenance des remarques et intérêts suscités par la présentation de notre prototype.

4.4.2 Contexte des expérimentations

4.4.2.1 Matériel utilisé

Durant les expérimentations, le démonstrateur SIG 3D V1 (baptisé TriGO) était installé sur un PC de bureautique standard. Les caractéristiques de cet ordinateur sont :

- Processeur Pentium III 800MHz
- 256Mo de mémoire vive
- Carte vidéo standard
- Système d'exploitation Windows NT4
- Le SGBD Oracle 8i est installé en local sur le PC.

La configuration de test est volontairement standard pour montrer que ce type de logiciel peut être utilisé sur des configurations modestes.

4.4.2.2 Sites utilisés

4.4.2.2.1 Scénario sur Mulhouse

Le jeu de données sur la ville de Mulhouse est composé de :

- une orthoimage SPOT rectifiée en projection UTM 32 nord avec une résolution de 10 m ;
- un MNT DTED rectifié en projection UTM 32 nord avec une résolution de 100 m ;
- un lot de données provenant de la BD Topo. Ce lot de données est transformé en 3D en extrudant les toits des bâtiments jusqu'au sol.

Il s'agit d'un jeu de données à basse résolution adapté aux scénarios de préparation de missions aéroportées ou pour l'aide au commandement.

La différence de résolution entre les données du sur-sol en 3D et les données raster (MNT et Image) ne permet pas d'utiliser les fonctions 3D telles que la recherche de trajectoire. Les données n'étant pas suffisamment précises, les algorithmes ne donnent pas de résultats exploitables.

4.4.2.2.2 Les sites sur Mulhouse

Forêt d'Ammerschwihr

Ce site contient un MNT texturé sur la forêt d'Ammerschwihr. Le MNT est très vallonné et la texturation permet un rendu réaliste d'une vue d'avion.

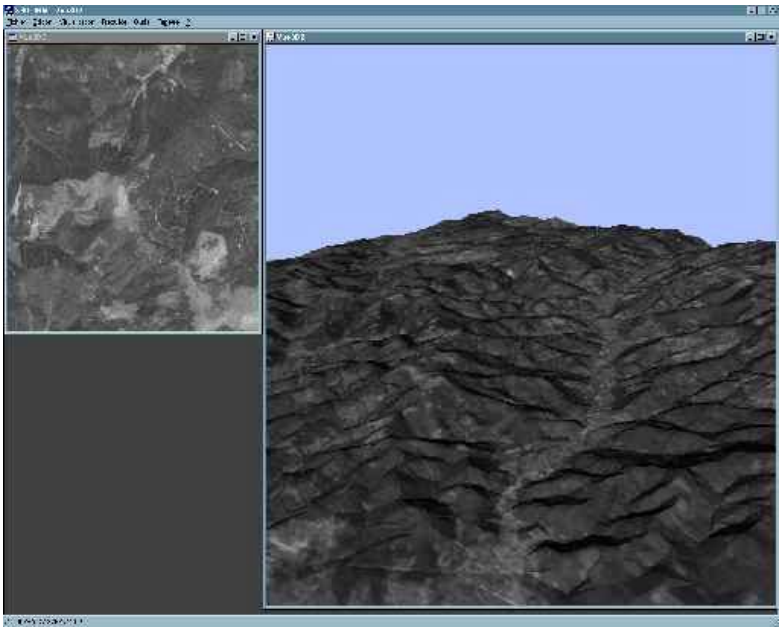


Figure 99 : Forêt d'Ammerschwihr

Mulhouse et aéroport de Colmar

Ce scénario contient le MNT texturé par une image SPOT entre la ville de Mulhouse et l'aéroport de Colmar. Il contient également quelques bâtiments de Mulhouse.

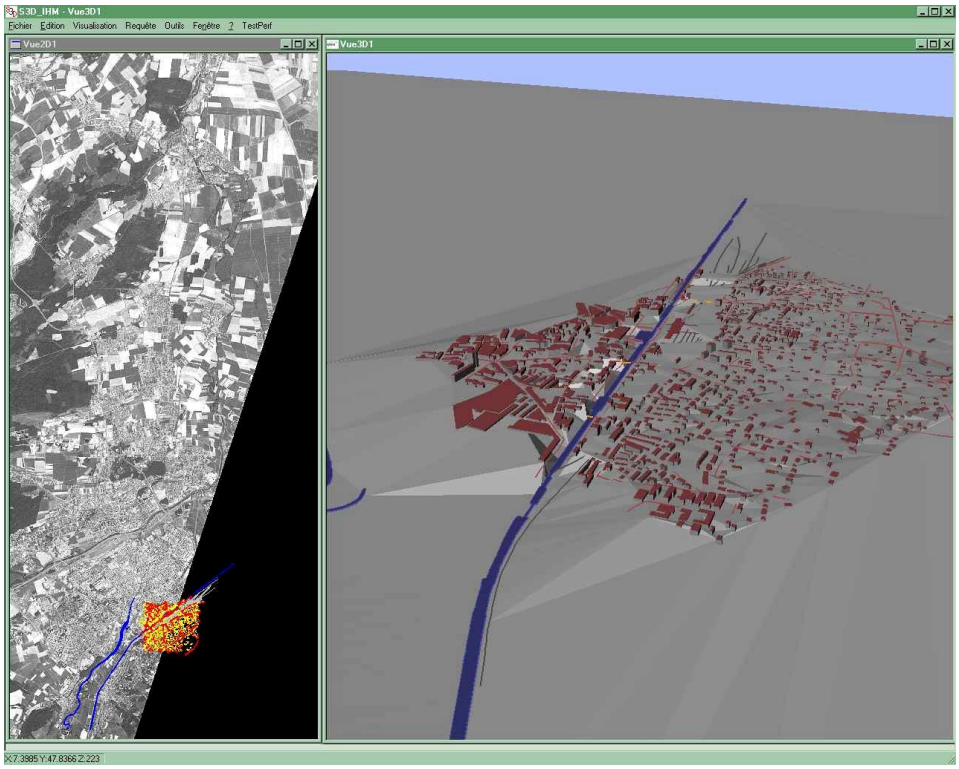


Figure 100 : Mulhouse

4.4.2.2.3 Scénario sur Paris

Le jeu de données sur la ville de Paris est composé de :

- une orthoimage ISTAR couleur en projection UTM 31 nord avec une résolution de 50 cm ;
- un MNT ISTAR rectifié en projection UTM 31 nord avec une résolution de 1m ;
- l'emprise des toits des bâtiments 2D½ exprimés en UTM31 nord. Ce lot de données est transformé en 3D en extrudant les toits jusqu'au sol.

Il s'agit d'un jeu de données à haute résolution adapté aux scénarios de préparation de missions héliportées ou à la simulation.

4.4.2.2.4 Les sites sur Paris

L'île St Louis

Ce site est le plus complet. Il contient des bâtiments, des cours d'eau, des routes et des ponts. Il permet donc d'utiliser les fonctions :

- de recherche de trajectoire sur un graphe routier ;
- de recherche de trajectoire sans graphe avec des contraintes de visibilité ;
- de calculs d'intervisibilité.

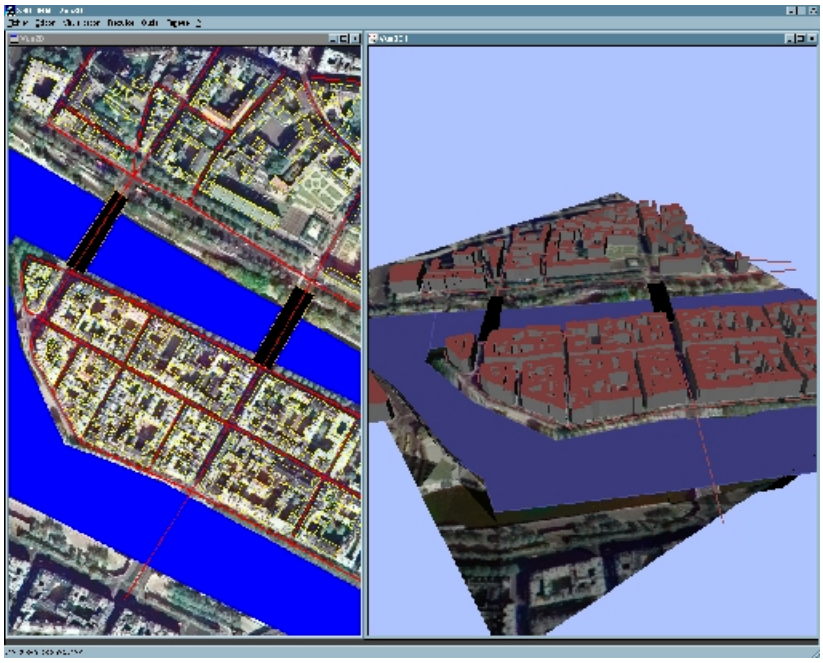


Figure 101 : Ile Saint Louis

La place des Vosges

Ce site contient les bâtiments situés autour de la place des Vosges. Nous y avons également inclus un bâtiment de type arche. Il permet donc d'utiliser pleinement les fonctions d'intervisibilité et de recherche de trajectoire sans support de graphe et en vraie 3D.

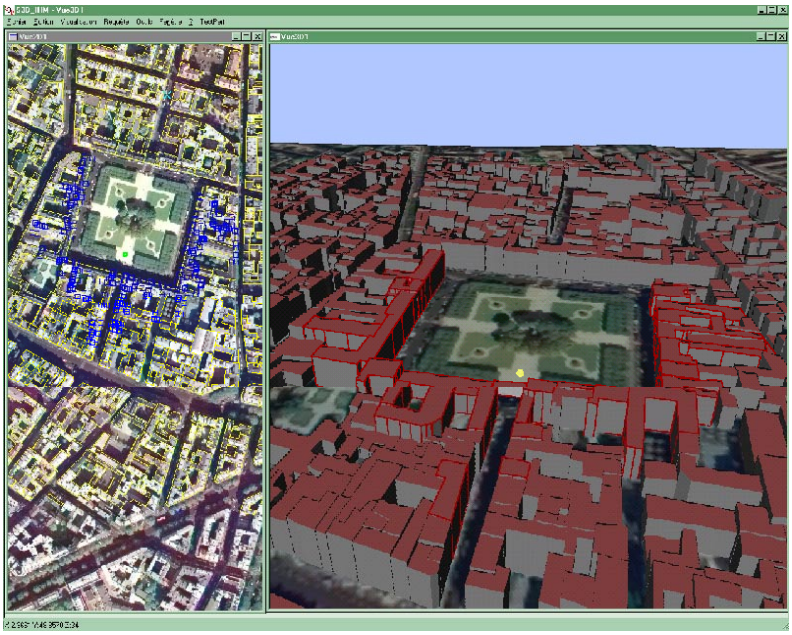


Figure 102 : Etablissement d'une carte de visibilité de 100 m de rayon sur la place des Vosges

4.4.3 Déroulement des expérimentations

4.4.3.1 Scénarios d'expérimentation

Le développement du prototype s'est effectué dans un cadre militaire, les scénarios utilisés durant les phases d'expérimentations sont donc à connotations militaires.

Nos démonstrations se sont déroulées en 4 étapes :

- Démonstration d'une préparation de mission de type commando ;
- Démonstration d'une préparation de mission aéroportée;
- Démonstration d'une aide au commandement ;
- Démonstration plus libre en fonction des souhaits exprimés par les personnes présentes durant l'expérimentation.

4.4.3.1.1 Préparation de mission commando

L'objet de ce scénario fut de valider les fonctionnalités proposées par le démonstrateur sur un scénario de préparation de mission commando. Ce type de scénario met en œuvre des données hautes résolutions en milieu urbain.

La thématique abordée était celle de la préparation d'une mission d'intervention dans un bâtiment de la place des Vosges à Paris. Les grandes lignes de la présentation s'articulaient autour de 5 actions :

- positionner des tireurs ayant pour tâche de sécuriser la zone d'intervention ;
- trouver un itinéraire permettant aux tireurs de rejoindre leurs postes ;
- empêcher toute évasion du bâtiment d'intervention ;
- surveiller et bloquer l'arrivée de renforts ennemis ;
- acheminer le commando.

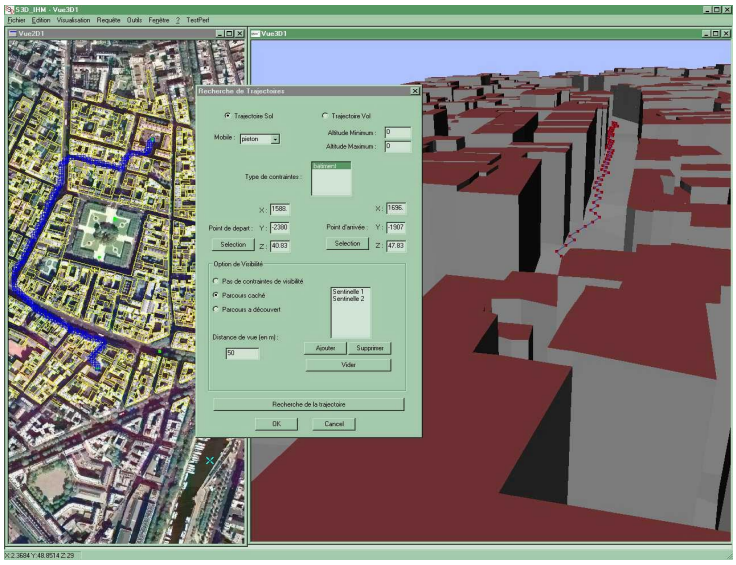


Figure 103 : trouver un itinéraire permettant aux agents de prendre leurs postes

4.4.3.1.2 Préparation de mission aéroportée

L'objet de ce scénario fut de vérifier les fonctionnalités proposées par le démonstrateur sur une préparation de mission aéroportée. Ce type de scénario met en œuvre des données à basse résolution ; en effet, un avion naviguant à des altitudes élevées, il est inutile qu'il connaisse dans le détail la morphologie des objets qu'il survole. En revanche, des données 3D de haute précision peuvent être utiles à l'approche de l'objectif final pour évaluer le résultat d'une opération.

Durant ce scénario, nous avons simulé le parcours d'un aéronef depuis son aéroport de décollage jusqu'à Mulhouse, zone d'intervention supposée. Les grandes lignes de la présentation sont :

- trouver une trajectoire aérienne à altitude constante ;
- trouver une trajectoire aérienne à altitude variable ;
- positionner les Radar ou la DCA.

Nous avons ensuite complété notre démonstration par des recherches de trajectoires hélicoptères sur le site de la place des Vosges à Paris.

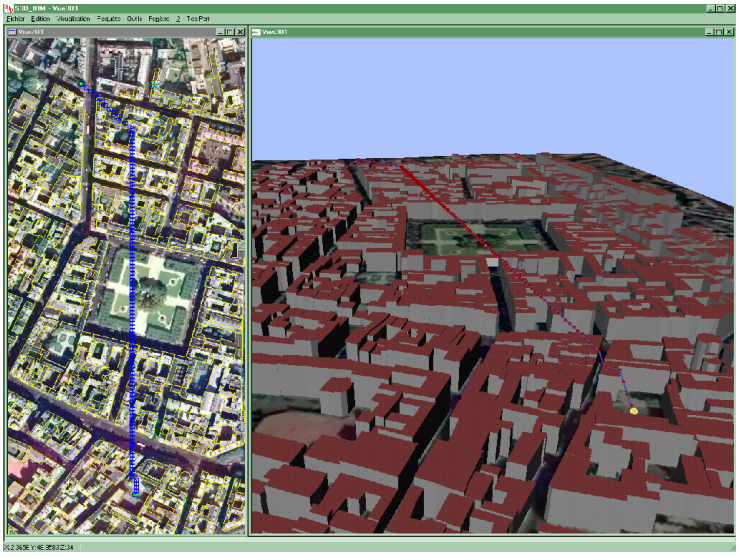


Figure 104 : Recherche de trajectoire hélicoptère sur Paris

4.4.3.1.3 Aide au commandement

L'objet de ce scénario fut de valider les fonctionnalités proposées par le démonstrateur pour une aide au commandement. Il s'agit d'afficher un théâtre d'opération contenant des données 2D et 3D présentant les différentes forces en présence et la nature du terrain.

L'utilisateur veut pouvoir déterminer ou positionner ses forces pour améliorer ses chances de réussite. Il veut déterminer où placer l'artillerie et les antennes radio pour couvrir au mieux le champ de bataille.
Il lui faut déterminer les voies d'accès utilisables par des engins blindés en tenant compte de la nature du terrain et du relief.
Nous avons utilisé pour ce scénario une large zone vallonnée entourant la forêt d'Ammerschwihr. Les grandes lignes de la présentations sont :

- analyser le théâtre d'opération ;
- positionner des antennes et l'artillerie ;
- trouver un itinéraire de convoyage.

4.4.3.1.4 Démonstration libre

Les expérimentations que nous avons conduites au près du personnel militaire ont invariablement mené à d'intéressantes questions et demandes d'approfondissements. Nous avons choisi d'illustrer nos réponses à l'aide du site de l'Ile saint Louis, à Paris, qui présente l'avantage d'être le plus complet tout en offrant une précision de donnée plus qu'honorable.

4.4.3.2 Fonctionnalités démontrées

Le déroulement des scénarios a permis de mettre en évidence quelques-unes des fonctionnalités de notre prototype en matière de navigation et d'interaction 3D, de requêtes d'intervisibilité, de calculs de trajectoire, de requêtes sémantiques ou encore de requêtes géométriques.

Navigation et Interaction 3D :

- Déplacement à travers une scène 3D, survol de la scène, déplacement contraint au sol à travers les rues de Paris, mode « track ball »
- Zoom, DeZoom
- Saisie de points 3D
- Texturage du MNT
- Modification de la géométrie d'un objet
- Export VRML
- Interrogation sur la sémantique d'une donnée à partir d'une saisie interactive dans la fenêtre 2D ou 3D

Intervisibilité :

- Carte de visibilité
- Test de visibilité totale
- Test de visibilité partielle
- Test de visibilité en milieu urbain
- Test de visibilité en milieu montagneux
- Positionnement de sentinelles
- Positionnement d'antennes relais

Calculs de trajectoires :

- Trajectoire piéton
- Trajectoire camion, avantage de la vraie 3D
- Recherche d'itinéraires non vus de sentinelles
- Trajectoire au sol tenant compte du MNT
- Trajectoire aériennes (hélicoptère, avion)
- Préférence de déplacement en fonction de la nature du sol (les voitures roulent préférentiellement sur les routes, même en trajectoire libre)
- Recherche d'itinéraire sur un graphe (réseau routier)

Requêtes sémantiques :

- Lecture des attributs sémantiques d'une donnée
- Mise à jour d'une donnée
- Recherche dans la base de données

Requêtes géométriques :

- Calculs de distance
- Création de zone tampon autour d'un point
- Création d'une zone tampon autour d'une trajectoire

4.4.4 Synthèse et conclusion sur les expérimentations

De manière générale, les fonctionnalités proposées dans TriGO ont reçu un bon accueil de la part des expérimentateurs qui ont souligné l'aspect innovant de notre prototype. Ils ont notamment apprécié les deux principales fonctionnalités du démonstrateur :

- les requêtes d'intervisibilité 3D ;
- la recherche de trajectoire prenant en compte les objets de la scène, la nature du terrain, la nature du mobile, le dénivelé et la visibilité.

Aucune fonction proposée par le démonstrateur n'a été unanimement rejetée. Les critiques portent essentiellement sur un manque de finitions et d'ergonomies (positionner une cible à partir de ses coordonnées géographiques, pouvoir se placer directement à la position d'un observateur, paramétrage plus simple, amélioration des symboles). Ce manque est lié à la nature même de notre application qui n'est qu'un prototype et non un produit fini.

Toutefois ces expérimentations ont révélé de nouvelles attentes, communes ou spécifiques, au sein de notre communauté expérimentatrice :

- Une des remarques les plus récurrentes est assez paradoxale, mais bien compréhensible : la majorité des expérimentateurs reconnaît que l'acquisition de données 3D est très délicate, qu'il faut souvent se contenter de données très succinctes et pourtant elle souhaite pouvoir naviguer au sein de scènes 3D réalistes et détaillées. Nous avions volontairement écarté le plaquage de textures au niveau de l'objet pour soulager les processus d'affichage, cette campagne de démonstration nous aura clairement fait comprendre que les textures sont aujourd'hui incontournables, quelle que soit l'application 3D envisagée. Réaliste ne veut pas dire exacte, et une solution efficace pour pallier la collecte souvent difficile de photos de bâtiments réside dans l'utilisation de textures génériques. Cette approche conduit cependant quelques organismes à la méfiance, ces derniers considérant qu'une information erronée peut être parfois plus préjudiciable qu'un manque d'information.

- La modélisation de l'intérieur des bâtiments est une autre problématique abordée par plusieurs organismes. D'un point de vue théorique et technique, de telles modélisations doivent pouvoir se faire sous TriGO sans contraintes majeures. La plupart des algorithmes de requêtes ne rencontreront aucune difficulté si on les applique à ce contexte d'intérieur. Une fois de plus, le facteur limitant risque d'être la disponibilité des données plutôt que leur traitement.

- Le géo-positionnement en temps réel sera sans doute dans les années à venir un élément incontournable de tout système d'information géographique. Le récent lancement officiel du programme européen Galileo de positionnement par satellite, concurrent direct du système GPS américain, amène à penser que le nombre déjà important d'applications utilisatrices de tels systèmes devrait croître assez significativement dans un futur proche. L'objectif de notre prototype était de mettre en valeur les avantages à tirer d'une modélisation véritablement 3D dans un SIG, la cause du positionnement en temps réel étant depuis bien longtemps acquise dans les esprits des utilisateurs; aussi n'avons nous pas songé à intégrer un module de type GPS dans notre application. Cependant cette intégration est aujourd'hui assez répandue et ne présente aucune difficulté technique, nous pourrions doter TriGO d'un tel module dans des évolutions ultérieures si la demande en était faite.

D'autres suggestions plus directement liées aux spécificités même des métiers ont vu le jour. Elles s'écartent quelque peu de l'axe initial que nous nous étions fixé, c'est-à-dire offrir un outil commun de manipulation de données géographiques 3D à des utilisateurs aux domaines d'applications hétérogènes. En d'autres termes, nous souhaitions privilégier les fonctions utiles à tous les corps de métiers concernés, plutôt que de développer une application tentant de répondre à un éventail de

besoins spécifiques. Ces expérimentations ont toutefois permis d'évoquer certaines pistes intéressantes pour le développement d'outils 3D à plus long terme.

- Les interventions militaires sont souvent fortement dépendantes de la météorologie ou conditionnées par celle-ci. A l'image de ce qui existe déjà dans certaines simulations (bien souvent à caractère ludique), il nous a été rapporté que modéliser les conditions climatiques serait d'un grand intérêt pour certaines missions (notamment aériennes).

- L'évaluation des dégâts causés par une intervention est une étape importante de la préparation de mission. Selon le résultat attendu, les moyens à mettre en œuvre varient très fortement. De simples textures peuvent suffire à rendre l'aspect visuel de bâtiments détruits ou d'arbres incendiés. En revanche, une modélisation plus précise des effets de souffle provoqués par une ou plusieurs explosions demande l'élaboration d'un outil à part entière et représente un travail très éloigné de la trivialité.

5 CONCLUSION

Synthèse de l'étude

L'objectif premier de notre travail de thèse était de concevoir, puis d'implémenter les caractéristiques architecturales et fonctionnelles d'un système d'information géographique tridimensionnel. Très rapidement, nous avons choisi de ne pas nous contenter de l'approche 2,5D classiquement rencontrée chez la plupart des grands éditeurs de SIG. Nous avons jugé celle-ci trop restrictive et ne permettant pas d'exploiter pleinement l'information potentiellement véhiculée par une base de données géographiques 3D.

Le modèle de donnée que nous proposons n'a pas la prétention d'être universel, mais il a le mérite de fournir à notre prototype une structure de base relativement simple sur laquelle des algorithmes de traitement et d'analyse ont pu être appliqués avec efficacité.

La forme 3D des objets géographiques est décrite suivant une modélisation géométrique de type B-Rep. Ce modèle de représentation surfacique permet de modéliser simplement les formes 3D les plus complexes. Toutefois, une simple modélisation géométrique n'offre aucune garantie de cohérence sur les données décrites; aussi avons nous opté pour l'ajout d'une topologie « structurelle » au dessus du niveau de modélisation géométrique. Notre modèle de topologie « structurelle » est fortement inspiré des courants classiques de modélisation topologique 3D ([MOLENAAR 1990], [TROTT 1999]), à ceci près que nous privilégions les relations entre les primitives nœud et face plutôt que de respecter le schéma traditionnel qui lie le nœud à un arc puis l'arc à une face (voir 2.3.2.2).

Les fonctionnalités de gestion des réseaux (électricité, eau, gaz, égouts, routes, voie ferrée, etc.) sont une composante essentielle d'un système d'information géographique, qu'il soit 2D ou 3D. Or, la relative complexité de notre modèle de topologie « structurelle » est mal adaptée à la modélisation de ces réseaux. Aussi avons nous choisi de décrire ces derniers par un modèle topologique plus simple dit modèle de topologie « réseau ». Ce modèle, comparable à ceux couramment utilisés dans les SIG 2D, s'appuie sur une représentation du réseau en un graphe planaire.

Le terrain est quant à lui modélisé par un maillage de triangles irréguliers (TIN). Contrairement au format RSG, le format TIN décrit le MNT par un ensemble de facettes (triangles). Sous cette forme, il est possible d'inclure le terrain dans les processus d'analyses et de traitements, au même titre que n'importe quel objet du sur-sol.

L'implémentation de ce modèle de données nous a permis de mettre en place des algorithmes d'analyse et de traitement exploitant pleinement le potentiel tridimensionnel des objets géographiques modélisés. Parmi ces algorithmes, ce sont ceux s'appliquant aux calculs de visibilité et aux recherches de trajectoires qui profitent au mieux des avantages d'une modélisation véritablement 3D.

En fonction des objectifs que les utilisateurs souhaitent atteindre, nos calculs d'intervisibilité statuent sur l'état de visibilité partielle ou sur l'état de visibilité totale d'un objet par rapport à un observateur. Pour y parvenir, nous utilisons deux types d'algorithmes : les algorithmes appliqués aux tests de visibilité partielle sont des algorithmes de lancer de rayons alors que les tests de visibilité totale font souvent appel à des recherches d'intersections entre polyèdres complexes.

Parmi les applications liées à la recherche de chemin optimal, nous distinguons celles qui s'effectuent sur des graphes prédéterminés (de type calcul d'itinéraire sur réseau routier) de celles où le graphe est à reconstruire (recherche de trajectoires dites « libres »). La reconstruction d'un graphe se fait par la discrétisation, dans une grille 2D ou 3D, des objets de la scène géographique qui ont des rôles de contraintes dans le déplacement du mobile. Les trajectoires recherchées dépendent donc des objets du sur-sol, mais aussi de la nature du terrain traversé, du relief, des caractéristiques de l'objet en mouvement ou encore de critères de visibilité.

La plupart des algorithmes d'analyse et de traitement des données spatiales, calculs d'intervisibilité et de trajectoire compris, procèdent souvent à des recherches d'objets sur critères spatiaux. Afin d'optimiser ces requêtes, nous avons implémenté un index spatial de type R-Tree 3D qui offre un accès plus direct et donc plus rapide aux données.

Vers un système opérationnel

Le prototype, issu des réflexions que nous avons portées sur la modélisation, l'architecture, ou encore l'algorithmie utile à un système d'information géographique tridimensionnel, a été expérimenté et validé auprès d'une communauté d'utilisateurs intéressés par la mise en œuvre d'un tel SIG3D. De manière générale, notre prototype a reçu un bon accueil de la part des expérimentateurs, qui ont particulièrement apprécié ses aspects novateurs. Les fonctionnalités présentées ont été jugées pertinentes et semblent correspondre aux attentes d'organismes manipulant au quotidien des bases de données géographiques.

Cependant, notre logiciel n'en est encore qu'au stade du prototype. L'évolution vers une application parfaitement opérationnelle, désirée et demandée par la majorité des personnes ayant assisté aux expérimentations, passe par la prise en compte de quelques améliorations. Certaines seront assez simples d'intégration, comme celles touchant aux aspects ergonomiques ou à l'ajout de fonctionnalités élémentaires. D'autres demanderont sans doute une phase de réflexion plus importante avant de pouvoir être mises en place. Nous pensons notamment au chargement des données : dans notre prototype, l'intégralité de la scène 3D, c'est-à-dire les formes 3D des objets et le modèle numérique de terrain, est chargée en mémoire au lancement de l'application. Un tel chargement risque de trouver rapidement ses limites avec la croissance des scènes géographiques à gérer. Un investissement dans des composants matériels hauts de gamme et performants peut permettre de repousser provisoirement ces limites. Néanmoins, il ne nous semble pas envisageable de concevoir un SIG 3D opérationnel ne traitant pas ces problèmes de chargement des données en profondeur. Une solution éventuelle peut tenir dans l'implémentation d'un système de chargement dynamique. Ce système ne chargerait en mémoire que les objets « utiles » à un instant t, c'est-à-dire ceux immédiatement impliqués dans les processus d'affichages, de traitements ou d'analyses.

Ouverture sur d'autres applications

Les travaux exposés tout au long de cette thèse ont été initiés par une étude commanditée par la cellule d'étude en géographie numérique de la Délégation Générale pour l'Armement. Dans ce contexte, les recherches menées n'ont pu être qu'influencées par les besoins des organismes à l'origine de ce projet et ce sont leurs attentes que nous avons avant tout cherché à satisfaire. Pourtant, nous ne pensons pas que le développement d'un système d'information géographique tridimensionnel doit être cantonné à des applications liées à la défense nationale. Moyennant quelques adaptations plus ou moins importantes, l'outil que nous avons mis au point peut rendre de nombreux services à un éventail assez large de professionnels aux domaines de compétences variés. Nous pouvons différencier quatre catégories de nouveaux utilisateurs :

- La première catégorie est constituée de ceux qui auront une utilisation d'un SIG 3D en tout point comparable à celle pour laquelle le prototype a été développé. Il est évident que des entités chargées de protection ou sécurité civile doivent avoir des besoins identiques à ceux émis par certaines unités militaires. La police nationale a, par exemple, des exigences très proches de celles de la gendarmerie nationale en matière d'équipement logiciel.

- La deuxième catégorie regroupe ceux qui tirent parti des fonctionnalités développées au cours de notre thèse, mais employées à des fins différentes de celles pour lesquelles elles ont été pensées. Les bureaux en aménagement urbain, les sociétés chargées de la mise en place de réseaux télécoms ou les sociétés chargées du positionnement des panneaux publicitaires sont autant d'acteurs de la vie économique intéressés par les possibilités d'un SIG 3D en matière d'intervisibilité. Les offices du tourisme seront eux plus attirés par les possibilités de visualisation et d'interaction avec les objets géographiques. Les recherches de trajectoires sauront probablement plaire aux sociétés de transports publics ou aux syndicats d'initiatives cherchant à mettre en place des itinéraires valorisant les patrimoines locaux.

- La troisième catégorie rassemble ceux qui ont besoin de fonctionnalités supplémentaires pour disposer d'un SIG 3D adapté à leur métier. Il serait, par exemple, utile qu'un tel outil puisse appréhender des phénomènes liés aux catastrophes naturelles telles que les inondations, les pollutions ou toute propagation « libre » d'un fluide dans un espace géographique. La gestion des risques naturels fait aujourd'hui partie des priorités de bon nombre de municipalités.

- Enfin, la dernière catégorie recense les utilisateurs pour lesquels des modifications plus profondes sont à entreprendre dans l'architecture même du SIG 3D pour que celui-ci puisse répondre à leurs problématiques. Les sociétés minières, les laboratoires géologiques, les instituts océanographiques ont, par exemple, un fort besoin en modélisation volumique 3D. Or notre prototype se contente aujourd'hui de modéliser le contour surfacique des objets, ce qui reste insuffisant pour couvrir le besoin de ces organismes.

Quoiqu'il en soit, devant la multitude des applications envisageables, les systèmes d'information géographique tridimensionnels semblent aujourd'hui promis à un bel avenir.

BIBLIOGRAPHIE

[BECKMANN 1990]
BECKMANN, N., KRIEGEL, H., SCHEIDER, R. et SEEGER, B. (1990). The R*Tree: An Efficient and Robust Access Method for Points and Rectangles. *ACM SIGMOD Conference on Management of Data*, 1990.

[BELLMAN 1958]
BELLMAN, R. (1958). On a Routing Problem. *Quart. Appl. Math,* 16, 87-90.

[BENTLEY 1975]
BENTLEY, J. L. (1975). Multi-Dimensionnal Binary Search Trees Used For Associated Searching. *Communications of the ACM,* 18, 509-517.

[BENTLEY 1979]
BENTLEY, J. L. et FRIEDMAN, J. H. (1979). Data Structure for Range Searching. *ACM Computing Survey,* 11(4), 397-409.

[BERNARD 2000]
BERNARD, G., RAMOS, F. et LEBARD, A. (2000). *Rapport de synthèse sur l'analyse des différents modèles 3D.* EADS S&DE, Geomatics, S3D/RPT/008, juin 2000.

[BLELLOCH 1990]
BLELLOCH, G. E. (1990). *Vector Models for Data-Parallel Computing*. Editeur: MIT Press, Cambridge (USA).

[BOISSENAT 1992]
BOISSENAT, J. D. et DOBRINDT, K. (1992). On-Line Randomized Construction of the Upper Envelope of Triangles and Surface Patches in R3. *4th Canadian Conference on Computational Geometry*, 1992.

[BRESENHAM 1965]
BRESENHAM, J. E. (1965). Algortihm for Computer Control of a Digital Plotter. *IDM Systems Journal,* 4(1), 25-30.

[BREUNING 1994]
BREUNING, M., BODE, T. et CREMERS, A. B. (1994). Implementation of Elementary Geometric Database for a 3D-GIS. *6th International Symposium on Spatial Data Handling*, Edinburgh, Scotland, 5-10 septembre 1994.

[BRUZZONE 1995]
BRUZZONE, E., DE FLORIANI, L. et MAGILLO, P. (1995). Udating Visibility Information on Multi-Resolution Terrain Models. *Conference on Spatial Information Theory. Lecture Notes in Computer Science 988*, 1995.

[BURTON 1993]
BURTON, D. (1993). *On the Inverse Shortest Path Problem.* Université de Notre-Dame de la Paix de Namur, soutenue en 1993, 136 p.

[CARVALHO 1995]
CARVALHO, J. (1995). *Extraction Automatique d'Informations Géomorphométriques à partir de Modèle Numérique de Terrain.* Université Paris 7, soutenue le 7 avril 1995, 172 p.

[CIGNONI 1995]
CIGNONI, P., PUPPO, E. et SCOPIGNO, R. (1995). Representation and Visualization of Terrain Surface at Variable Resolution. *The Visual Computer,* 13(5), 199-217.

[DAVID 1991]
DAVID, B. (1991). *Modélisation, représentaiton et gestion de l'information géographique, une approche en relationnel étendu.* Université de Paris 6, Soutenue le 8 juillet 1991, p.

[DE CAMBRAY 1994]
DE CAMBRAY, B. (1994). *Etude de la modélisation, de la manipulation et de la représentation de l'information spatiale 3D dans les bases de données géographiques.* Paris 6, 197 p.

[DE FLORIANI 1988]
DE FLORIANI, L. et FALCIDIENO, B. (1988). A Hierarchical Boundary Model for Solid Object Representation. *ACM Transaction on Graphics,* 7(1), 42-60.

[DE FLORIANI 1989]
DE FLORIANI, L. (1989). A Pyramidal Data Structure for Triangle-Based Surface Description. *IEEE Computer Graphics and Applications,* 9, 67-78.

[DE FLORIANI 1991]
DE FLORIANI, L., FALCIDIENO, B., NAGY, G. et PIENOVI, C. (1991). On Sorting Triangle in Delaunay Tesselation. *Algorithmica,* 6, 522-556.

[DE FLORIANI 1993]
DE FLORIANI, L. et MAGILLO, P. (1993). Computing Visibility Maps on a Digital Terrain Model. *Lecture Notes in Computer Science,* 716, 248-269.

[DE FLORIANI 1994]
DE FLORIANI, L., MONTANI, C. et SCOPIRO, R. (1994). Parallelizing Visibility Computations on Triangulated Terrains. *International Journal of Geographical Information Systems,* 8, 515-532.

[DE FLORIANI 1998]
DE FLORIANI, L. et MAGILLO, P. (1998). Intervisibility on Terrains. *Geographical Information Systems*, Addison-Wesley**,** 543-556.

[DE LA LOSA 1999]
DE LA LOSA, A. et CERVELLE, B. (1999). 3D Topological Modeling and Visualisation for 3D GIS. *Computers & Graphics,* 23, 469-478.

[DE LA LOSA 2000]
DE LA LOSA, A. (2000). *Modélisation de la troisième dimension dans les bases de données géographiques.* Université de Marne-La-Vallée, soutenue le 28 janvier 2000, 174 p.

[DELPY 1993]
DELPY, T. et ZEITOUNI, K. (1993). A Graph Model to Describe Topological Relationships in 3D World. *Third International Conference on Computer in urban Planing and Urban Management*, Atlanta, Georgia, 23-25 juillet 1993.

[DIJKSTRA 1959]
DIJKSTRA, E. W. (1959). A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik,* 1, 269-271.

[DONNAY 1992]
DONNAY, J. (1992). Détermination en mode maillé du champ d'intervisibilité dans un modèle numérique de terrain. *Cartographica,* 29(3-4), 75-82.

[DROESBEKE 1987]
DROESBEKE, F., HALLIN, M. et LEFEVRE, C. (1987). *Les Graphes par l'Exemple*. Editeur: Ellipse, Paris.

[EDELSBRUNNER 1989]
EDELSBRUNNER, H., GUIBAS, L. J. et SHARIR, M. (1989). The Upper Envelope of Piecewise Linear Functions: Algorithms and Applications. *Discrete and Computational Geometry,* 4, 311-336.

[EGENHOFER 1990]
EGENHOFER, M. J. et R., H. J. (1990). A mathematical framework for definition of topological relationships. *4th international symposium on Spatial Data Handling*, Zurich, Switzerland, 1990.

[FISHER 1993]
FISHER, P. F. (1993). Algorithm and Implementation Uncertainty in Viewshed Analysis. *International Journal of Geographical Information Systems,* 7, 331-347.

[FISHER 1994]
FISHER, P. F. (1994). Stretching the viewshed. *The Sixth International Symposium on Spatial Data Handling*, 1994.

[FLICK 1996]
FLICK, S. (1996). How to Support Spatial Objects in a 3D-GIS. *First International Conference of GeoComputation*, Leeds, UK, 1996.

[FOLEY 1994]
FOLEY, J., VAN DAM, A., FEINER, S., HUGHES, J. et PHILLIPS, R. (1994). *Introduction to computer Graphics*. Editeur: Addison-Wesley,

[FORD 1956]
FORD, L. R. (1956). *Network Flow Theory.* The Rand Corporation, p-923, 1956.

[FORD 1962]
FORD, L. R. et FULKERSON, D. R. (1962). *Flows in Networks*. Editeur: Princeton University Press, Princeton.

[FRANKLIN 2000]
FRANKLIN, W. R. (2000). Application of Analytical Cartography. *Cartography and Geography Information Systems,* 27(3), 225-237.

[GARDARIN 1991]
GARDARIN, G. (1991). *Base de Données - Objets et Relationnel*. Editeur: Eyrolles,

[GERBE 2002]
GERBE, P. (2002). Outils, données et applications de l'information géographique en 3D. *GeoEvenement*, Paris, Avril 2002.

[GURSOZ 1988]
GURSOZ, E., CHOI, Y. et PRINZ, F. B. (1988). Vertex Based Representation of Non-Manifold Boundaries. *NSF working conference on Geometric Modeling*, Rensselearville, USA, 18-22 septembre 1988.

[GUTTMAN 1984]
GUTTMAN, A. (1984). A Dynamic Index Structure for Spatial Searching. *ACM SIGMOD Conference on Management of Data*, 1984.

[ISO/TC211 2001]
ISO/TC211 (2001). *Geographic Information - Spatial Schema.* Internationnal Standard Organization, ISO/DIS 19107, 2001.

[JONES 1989]
JONES, C. B. (1989). Data Structure for Three-Dimensional Spatial Information Systems in Geology. *International Journal of Geographical Information Systems,* 3(1), 15-31.

[JORGENSEN 2002]
JORGENSEN, T. E. (2002). *Survey of Terrain Visualization Software.* US Army Topographic Engineering Center, Topography, Imagery and Geospatial Reserch Division, 12 juin 2002,

[KATZ 1992]
KATZ, M. J., OVERMARS, M. H. et SHARIR, M. (1992). Efficient Hidden Surface Removal for Objects with Small Union Size. *Computational Geometry Theory Application,* 2, 223-234.

[KIRKBY 1996]
KIRKBY, S. D., POLLIITT, S. et EKLUND, P. W. (1996). Implementing a Shortest Path Algorithm in a 3D GIS Environment. *International Symposium in Spatial Data Handling*, Delft, August 12-16.

[KOFLER 1998]
KOFLER, M. (1998). *R-Trees for Visualizing and Organizing Large 3D GIS Database.* Université de Gratz, soutenue le 27 octobre 1998, 103 p.

[LADNER 1998]
LADNER, R. (1998). *Non-Manifold Winged-Edge Topology for 3D Modelling.* MSc Thesis, University of New Orleans, Mai 1998, 112 p.

[LADNER 1999]
LADNER, R., SHAW, K. et ABDELGUERFI, M. (1999). 3D Synthetic Environment Representation Using the "Non-Manifold 3D Winged-Edge" Data Structure. *Interoperating Geographic Information Systems, Second International Conference*, Zurich, Switzerland, mars 1999.

[LANGLOIS 1994]
LANGLOIS, P. (1994). Formalisation des concepts topologiques en géomatique. *Revue internationale de géomatique,* 4(2/1994), 181-205.

[LEBARD 2000]
LEBARD, A., RAMOS, F. et BERNARD, G. (2000). *Etude d'une Topologie Associé aux Données 3D.* EADS, Sys&DE, S3D/RPT/009, septembre 2000.

[LEE 1991]
LEE, J. (1991). Analyses of Visibility Sites on Topographic Surfaces. *International Journal of Geographical Information Systems,* 5, 413-429.

[LEUTENEGGER 1997]
LEUTENEGGER, S., EDGINGTON, J. et LOPEZ, M. A. (1997). STR: A Simple and Efficient Algorithm for a R-Tree Packing. *the 1997 International Conference on Data Engineering (ICDE 1997)*, 1997.

[LIENHART 1991]
LIENHART, P. (1991). Topological Model for Boundary Representation : a Comparison with n-dimensional Generalized Maps. *Computer -Aided Design,* 23(1), 59-82.

[LINDSTROM 1997]
LINDSTROM, P., KOLLER, D., RIBARSKY, W., HODGES, L., OP DEN BOSH, A. et FAUST, N. (1997). *An Integrated Global GIS and Visual Simulation System.* Graphics, Visualization, & Usability Center Georgia Institute of Technology, GVU Technical Report 97-07, 1997.

[MILLS 1992]
MILLS, K., FOX, G. et HEINBACH, R. (1992). Implementing an Intervisibility Analysis Model on a Parallel Computing System. *Computer and Geosciences,* 18, 1047-1054.

[MOLENAAR 1990]
MOLENAAR, M. (1990). A Formal Data Structure for Three Dimensional Vector Maps. *1st European Conference on Geographical Information Systems*, Amsterdam, The Netherlands, 1990.

[MOLLER 1997]
MOLLER, T. (1997). A Fast Triangle-Triangle Intersection Test. *Journal of Graphics Tools,* 2(2).

[MOORE 1959]
MOORE, E. F. (1959). The Shortest Path Through a Maze. *International Symposium on the Theory of Switching*, Harvard University, Cambridge MA, 1959.

[NIEVERGELT 1984]
NIEVERGELT, J., HINTERBERGER, H. et SEVCIK, K. C. (1984). The Grid File : An Adaptable, Symetric Multikey Structure. *ACM Trenseaction and Database Systems,* 9, 38-71.

[Open-Cascade 1999]
Open-Cascade (1999). www.opencascade.org.

[OpenGIS 1999]
OpenGIS (1999). *OpenGIS Simple Features Specification for SQL Révision 1.1.* OpenGIS, 99-049, mai 99.

[OVERMARS 1992]
OVERMARS, M. H. et SHARIR, M. (1992). A Simple Output-Sensitive Algorithm for Hidden Surface Removal. *ACM Transaction on Graphics,* 11, 1-11.

[PAJAROLA 1998]
PAJAROLA, B. P. (1998). *Access to Large Scale Terrain and Image Database in Geoinformation Systems".* Swiss Federal Institute of Technology, 12729, 1998.

[PGSC 1998]
PGSC (1998). *VPF 3D Topology Extentions, 3D Topology Model Technical Report.* PAR Government systems Corporation, NMA202-97-c-1044, octobre 1998.

[PIGOT 1991]
PIGOT, S. (1991). Topological Models for 3D Spatial Information Systems. *Auto-Carto'10*, 1991.

[PITTWAY 1967]
PITTWAY, M. L. V. (1967). Algorithm for drawing Ellipses or Hyperbolae with a Digital Plotter. *Computer Journal,* 10(3), 282-289.

[POLLITT 1995]
POLLITT, S. (1995). *A 3-D Spatial Information System for Emergency Routing in Okayama City.* BSc Thesis, University of Adelaide, soutenue en 1995, 81 p.

[PREPARATA 1992]
PREPARATA, F. P. et VITTER, J. S. (1992). A Simplified Technique for Hidden-Line Elimination in Terrains. *Lecture Notes in Computer Science,* 577, 135-144.

[PUPPO 1996]
PUPPO, E. (1996). Variable Resolution Terrain Surfaces. *Canadian Conference on Computational Geometry*, 1996.

[PUPPO 1997]
PUPPO, E. et MARZANO, P. (1997). Discrete Visibility Problems and Graph Algorthm. *International Journal of Geographical Information Science,* 11(2), 139-162.

[RAMOS 2000]
RAMOS, F. et BERNARD, G. (2000). *Rapport de Synthèse sur la Visualisation.* S3D/RPT/013, septembre 2000.

[RAMOS 2001]
RAMOS, F., LEBARD, A. et BERNARD, G. (2001). *Rapport d'analyse et de synthèse sur les requêtes.* S3D/RPT/012, janvier 2001.

[RAMOS 2002]
RAMOS, F. (2002). A Multi-Level Approach for 3D Modeling in Geographical Information Systems. *Internationnal Society for Photogrametry and Remote Sensing, commission IV, Symposium 2002*, Ottawa, Canada, juillet 2002.

[RAPER 1991]
RAPER, J. F. et KELK, B. (1991). Three-Dimensional GIS. *Geographical Information Systems: Principles and Applications.* D. J. MAGUIRE, M. F. GOODCHILD and D. W. RHIND, 299-316.

[REDDY 2000]
REDDY, M., IVERSON, L. et LECLERC, Y. G. (2000). Under the hood of GeoVRML 1.0. *Proceedings of the fifth symposium on Virtual reality modeling language (Web3D-VRML)*, Monterey, California, US, ACM Press, 2000.

[REIF 1988]
REIF, J. H. et SEN, S. (1988). An Efficient Output-Sensitive Hidden-Surface Removal Algorithm and its Parallelization. *4th Annual ACM Symposium on Computational Geometry*, 1988.

[RHIN 1997]
RHIN, C. (1997). *Modélisation et Gestion de Données Géographiques Multi-Sources.* Université Versailles-Saint-Quentin-en-Yvelines, soutenue le 3 juillet 1997, 189 p.

[RIKKER 1994]
RIKKER, R., MOLENAAR, M. et STUIVER, J. (1994). A Query Oriented Implementation of a Topologic Data Structure for 3-dimensional Vector Map. *International Journal of Geographical Information Systems,* 8(3), 243-260.

[SAMET 1990]
SAMET, H. (1990). *Application of Spatial Data Structures*. Editeur: Addison-Wesley,

[SELLIS 1987]
SELLIS, T., ROUSSOPOULOS, N. et FALOUSTOS, C. (1987). The R+Tree: A Dynamic Index for Multi-Dimensional Objects. *International Conference on Very Large Data Bases*, 1987.

[SHAPIRA 1990]
SHAPIRA, M. (1990). *Visibility on Terrain Labeling.* Rensselaer Polytechnic Institute, mai 1990, mai 1990.

[SHAW 1998]
SHAW, K., ABDELGUERFI, M. et LADNER, R. (1998). VPF+: AVPF Extension Providing 3D Modeling Topology. *Image 98 Conference*, Scottsdale, Arizona, EU, aout 1998.

[SOYEZ 1994]
SOYEZ, F. (1994). *PERMI : Planification Evoluée par la Robotique Mobile Intelligente.* Université de Picardie Jules Verne, soutenue le 18 février 1994, p.

[STEFANKIS 1995]
STEFANKIS, E. et KAVOURAS, M. (1995). On the Determination of the Optimum Path in Space. *COSIT'95*, 1995.

[TAMMINEN 1983]
TAMMINEN, M. (1983). Performance Analysis of Cell Based Geometric File Organization. *International Journal of Computer Vision, Graphics and Image Processing,* 24, 160-181.

[TAMMINEN 1984]
TAMMINEN, M. (1984). Comments on Quad and Octrees. *Communications of the ACM,* 27, 248-249.

[TargetJr 1997]
TargetJr (1997). http://www.esat.kuleuven.ac.be/~targetjr/GE_mirror/.

[TROTT 1999]
TROTT, K. et GREASLEY, I. (1999). A 3D Spatial Data Model for Terrain Reasoning. *Geocomputation 99*, 1999.

[VAN AKEN 1984]
VAN AKEN, J. R. (1984). An Efficient Ellipse-Drawing Algorithm. *CG & A,* 4(9), 24-35.

[VAN OOSTEROM 1990]
VAN OOSTEROM, P. (1990). A Modified Binary Space Portioning Tree for Geographical Information Systems. *International Journal of Geographical Information Systems,* 4, 133-146.

[VAN OOSTEROM 1991]
VAN OOSTEROM, P. (1991). The Reactive Tree : A Storage Structure for a Seamless, Scaleless Geographic Database. *American Congress on Surveying and Mapping*, 1991.

[VAN OOSTEROM 1994]
VAN OOSTEROM, P., VERTEGAAL, W. et VAN HEKKEN, M. (1994). Integrated 3D modelling within a GIS. *Advanced Geographic Data Modelling*, 80-95.

[WORBOYS 1998]
WORBOYS, M. F. (1998). Relational Database and Beyond. *Geographical Information Systems*. Addison-Weysley**,** 373-384.

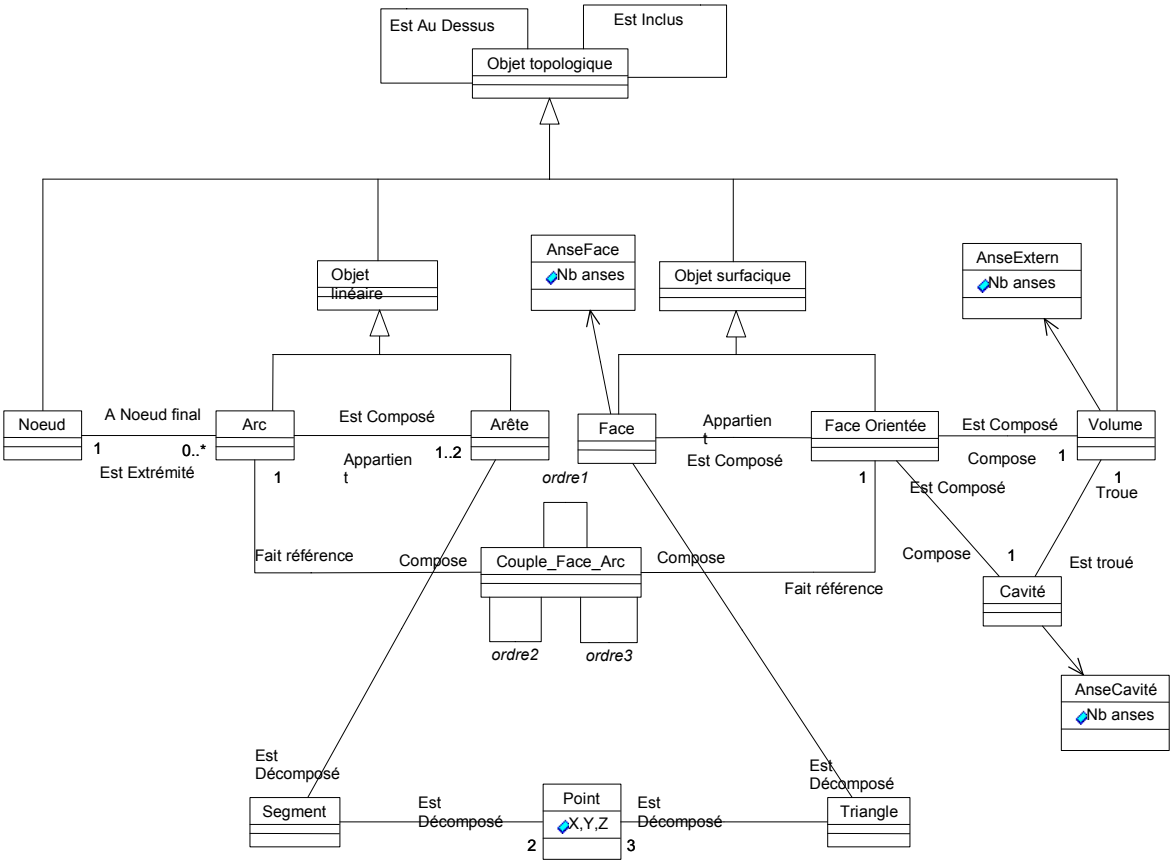
[ZEITOUNI 1995]
ZEITOUNI, K. (1995). Topological Modelling for 3D GIS. *4th International Conference on Computers in Urban Planning and Urban Management*, Melbourne, Juillet 1995.

[ZLATANOVA 1996]
ZLATANOVA, S., PILOUK, M. et TEMPFLI, K. (1996). Building Reconstruction From Aerial Images and Creation of 3D Topologic Data Structure. *IAPR/TC-7 Workshop*, Graz, Austria, 2-3 septembre 1996.

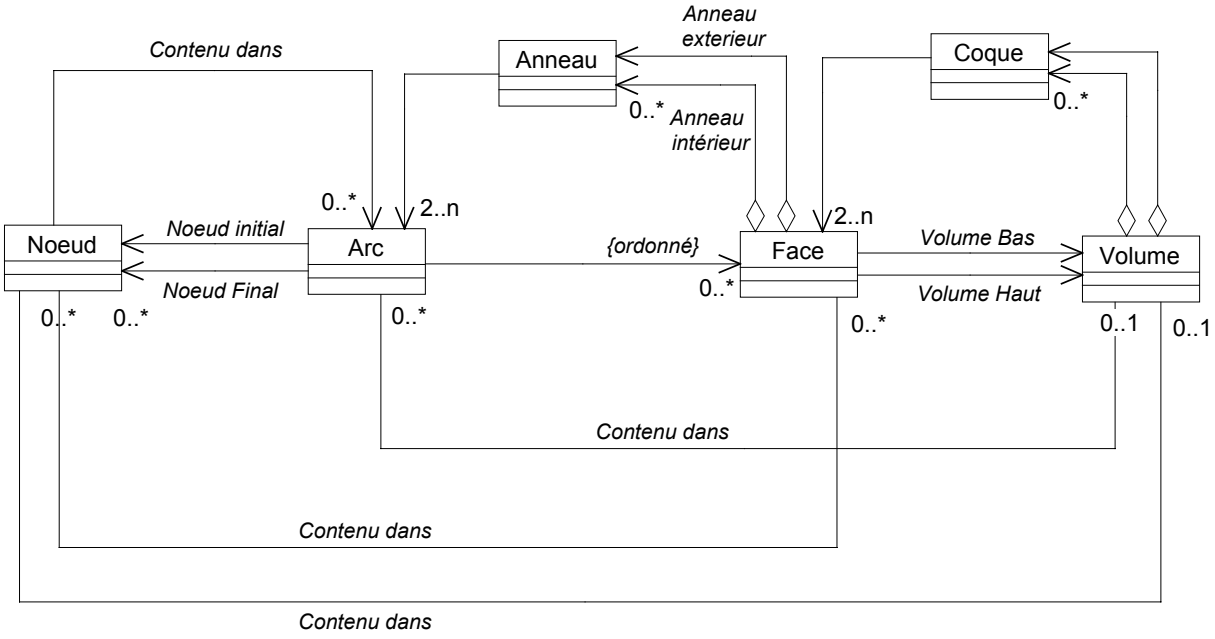
[ZLATANOVA 1999]
ZLATANOVA, S. (1999). An Alternative for a 3D GIS. *International Symposium on Modern Information and GPS Technology - Aspects ans Implications of Their Aplication*, Sofia, 11-12 novembre 1999.

ANNEXE : Quelques modèles topologiques

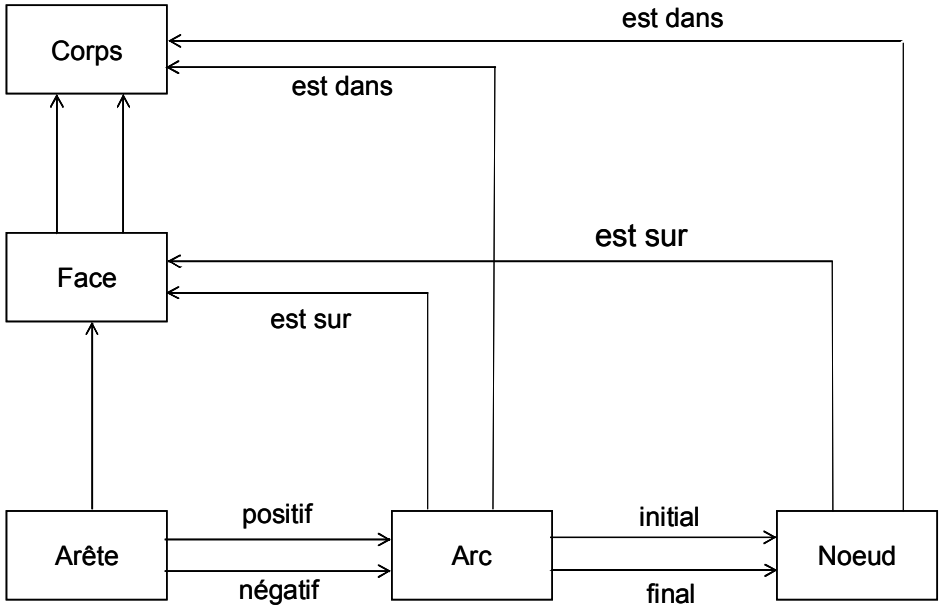
- **Modèle Geo3D [DE LA LOSA 2000]**



- **Modèle VPF+ [TROTT 1999]**



- **Modèle FDS [MOLENAAR 1990]**



- **Modèle ISO [ISO/TC211 2001]**