



**HAL**  
open science

# Détection de changement sur des données géométriques tridimensionnelles

Daniel Girardeau-Montaut

► **To cite this version:**

Daniel Girardeau-Montaut. Détection de changement sur des données géométriques tridimensionnelles. domain\_other. Télécom ParisTech, 2006. English. NNT: . pastel-00001745

**HAL Id: pastel-00001745**

**<https://pastel.hal.science/pastel-00001745>**

Submitted on 22 May 2006

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



École Doctorale  
d'Informatique,  
Télécommunications  
et Électronique de Paris

# Thèse

présentée pour obtenir le grade de Docteur  
de l'École Nationale Supérieure des Télécommunications

**Spécialité : Traitement du Signal et Images**

**DANIEL GIRARDEAU-MONTAUT**

Détection de changement sur des données  
géométriques tridimensionnelles

Soutenue le 15 Mai 2006 devant le jury composé de

Jacques Droulez

Président

Michel Devy

Rapporteurs

Behzad Shariat

Nicolas Paparoditis

Examineurs

Francis Schmitt

Michel Roux

Directeur de thèse

Raphaël Marc

Co-directeur de thèse, invité



*« Il n'est pas bon que le pouvoir d'observer se développe  
plus vite que l'art d'interpréter. »*

Alain

*« Toute vertu est fondée sur la mesure. »*

Sénèque (!)





# Remerciements

Et bien tout d'abord, j'ouvrirai la danse avec les remerciements *professionnels* (bien que la frontière entre professionnel et personnel soit parfois floue).

Je tiens à remercier cordialement Raphaël Marc, d'EDF R&D, qui a toujours été très présent, impliqué, et qui m'a supporté - dans son bureau et ailleurs - depuis tant d'années (en temps que stagiaire, prestataire et enfin *thésard*!). Mille mercis. Ensuite viennent Michel Roux, son pendant à Telecom Paris, toujours à l'écoute et très sympathique, et Guillaume Thibaut, lui aussi d'EDF R&D, qui a beaucoup apporté à ma thèse et qui a su faire fonctionner - et amuser - au mieux mes derniers neurones.

Je remercie aussi Francis Schmitt, professeur à Telecom Paris, Nicolas Paparoditis, chercheur à l'IGN, et Xin Chen, directeur de Mensi, qui ont pris sur leur temps rare et précieux pour s'intéresser à mes travaux et m'ont ainsi permis d'effectuer une pré-soutenance fort enrichissante. J'associe à ces remarques bien sûr les autres membres de mon jury : Jacques Droulez, du Collège de France, qui m'a fait l'honneur de le présider, et mes rapporteurs, MM. Michel Devy et Behzad Shariat, qui se sont montrés très intéressés par mes travaux et ont parfaitement rempli leur rôle en me posant des questions si pertinentes. Je poursuis ces cordiaux remerciements, en les étendant aux autres piliers des réunions "2D-3D" du LPPA : Florent Duguet bien sûr, jeune docteur (reconverti?) avec qui j'ai eu de fructueux échanges et qui a su combler nombre de mes lacunes, ainsi que MM. Thomas Chaperon et Raouf Benjemaa, très sympathiques ingénieurs docteurs chez Mensi.

D'autres personnes très sympathiques et qui m'ont fait découvrir un métier nouveau (pour moi) et très intéressant sont les topographes du CNEPE de Tours. Cela me renvoie sur la planète EDF où j'ai pu rencontrer de très nombreuses personnes qu'il me serait impossible de toutes citer ici, même si beaucoup le mériteraient. Je veux au moins remercier tous les membres de l'équipe CAO du département Sinetics d'EDF R&D, qui ont toujours été accueillants, patients et ouverts (Pascal, Christophe, Yves, Alain, Stéphane, Michel, Sandrine, Gérard, Alice, Dominique, Bernard et les autres!). Je dois aussi remercier tout particulièrement Christian Derquenne, expert en statistiques à la R&D, qui a accompagné mes premiers balbutiements dans le domaine. Il en va de même avec les personnes du groupe TSI/TII de Telecom Paris, et en particulier mes nombreux compagnons de fortune, dont je citerai les prénoms dans un ordre chronologique approximatif de leur soutenance et en espérant ne pas en oublier trop. Tout d'abord ceux qui sont allés faire leur nid ailleurs : Carlos, Céline T., Alejandro, Ferdaous, Sveta, Tony, Dallila, Jasmine, Florent (bis), Saeid, Eve, Pau. Et ceux qui resteront un peu plus longtemps à Telecom :

---

Jamal, Céline H., Pénélope, Antonio (à qui je décerne au passage la palme de la "personne la plus sympathique" jamais rencontrée!), Gero, Jean-François, Julien, Thomas ... et j'espère de nombreux autres!

Je peux (enfin ;) passer aux remerciements *personnels* !

Tout d'abord, un immense merci à Myriam (et bien plus encore ;), qui m'a supporté alors qu'elle n'avait pas l'excuse de s'intéresser à ce que je faisais durant ma thèse ! Et qui j'espère me supportera encore "après" ;). Un grand merci aussi à son père, Jean-Pierre Delhomme, qui a su si bien éclairer mon chemin - pour l'avoir emprunté lui aussi il y a quelques années - et qui a su prendre sur son temps précieux pour relire et améliorer ce manuscrit.

Je remercie aussi bien sûr ma mère (qui a relu elle aussi ce manuscrit plusieurs fois avec bravoure et a toujours été fière de moi ;) et mon père, sans qui bien sûr je ne serai pas qui je suis. Et bien que j'entende des voix dans le fond qui me font remarquer qu'une telle assertion peut s'étendre à différents degrés à toute personne sur cette Terre, je me restreindrai à y inclure et à remercier - par pure flemme : le reste de ma famille (ma soeur et mes frères, mes neveux et nièces que j'embrasse, tous mes grands-parents - qu'ils puissent lire ces lignes ou non - et la très prolifique et variée famille Burnod) ainsi que tous mes amis : Julie et Julien, les éleveurs de lapins, Alex, Bruno, Jeremy et Jo, mes colocataires de longue date, tous les autres "golris" (Georges, Anne-So, Anthony, Tonio, Aubin, etc.), sans oublier Julie (la très courageuse expatriée), et de nombreux autres éparpillés un peu partout, en France et ailleurs (Jerôme & Isa, tous mes voeux de bonheur!).

Bon ça suffit, j'espère que ceux qui ne sont pas cités nominativement ici ne m'en voudront pas trop, mais je me rends compte aujourd'hui qu'écrire des remerciements est un exercice long et difficile, et qu'il vaut donc mieux remercier les gens de vive voix au cas par cas et autant que possible (ce que j'espère avoir toujours fait au mieux ;), plutôt que d'essayer de le faire d'un coup sur une feuille de papier (quelle drôle d'idée!).

Et encore merci pour le poisson.

---

# Résumé

Ce travail de thèse aborde le problème de la détection de changement à partir de données géométriques tridimensionnelles. Ces données peuvent représenter n'importe quel objet ou environnement mais le travail s'est focalisé sur des environnements complexes et de grande envergure (centrales nucléaires, plateformes pétrolières, etc.). L'idée principale est d'exploiter directement les nuages de points acquis par des scanner laser, sans passer par des reconstructions intermédiaires plus "haut niveau" (maillages triangulaires, etc.) mais qui sont souvent impossibles à calculer automatiquement et proprement sur de tels environnements. Les scanners laser sont des appareils compacts et portables capables de mesurer rapidement un très grand nombre de points 3D sur la surface des objets. Les nuages de points résultants sont des données précises et très détaillées, mais aussi très volumineuses et non structurées. Il est donc nécessaire de développer des traitements particuliers et performants.

Le manuscrit présente l'étude et la mise en place concrète d'un processus complet de détection de changements géométriques, se basant principalement sur des nuages de points 3D et des maillages surfaciques 3D. On propose en particulier une méthode robuste de calcul d'écarts directement entre nuages de points 3D, ainsi que deux algorithmes de segmentation géométrique de nuages associés à des valeurs d'écarts (dans le but d'isoler les zones de changement). Le premier algorithme met en IJuvre une méthode de classification des points du nuage par analyse statistique locale des valeurs d'écarts. Le second se base sur la propagation d'un contour, contrainte par la norme du gradient géométrique des valeurs d'écarts. On propose aussi un codage particulier d'une structure octree, qui, sous cette forme, peut être calculée très rapidement et sur mesure, ce qui permet d'accélérer fortement les calculs.

Plusieurs applications, principalement industrielles, sont finalement proposées. Une ouverture est faite vers des applications d'aide à la gestion de catastrophes naturelles ou industrielles, qui pourraient profiter de la rapidité et de la fiabilité des méthodes d'analyse et de suivi géométrique proposées.

---



# Abstract

This work deals with the detection of geometric changes between 3D point cloud data acquired by ground-based laser sensors (with a focus on complicated environments such as power plants or oil platforms). Ground-based laser sensors are compact and portable devices that are able to acquire rapidly and accurately a large amount of 3D points on the surface of objects. These point clouds are very dense and detailed, but also huge and unstructured. Therefore, we have developed fast and ad-hoc algorithms to deal with such datasets.

This manuscript presents the study and effective implementation of a complete change detection framework based on 3D point clouds and 3D mesh models. We mainly propose a method to compute distances directly between two 3D point clouds in a robust and accurate way, and also two different segmentation algorithms. The first segmentation algorithm is a "region based" approach that relies on a local statistical test. It classifies points of a cloud in two categories, depending on the distribution of distances computed for each point and its nearest neighbours. The second algorithm is an "edge based" approach that relies on a propagation scheme constrained by the gradient norms of the computed distances. We also propose a specific coding scheme of an octree structure. Such a structure can be computed very efficiently and in such a way that most of the following point based processings can be optimized and the global process is therefore greatly accelerated.

Eventually, we present several industrial applications of our framework. We also investigate the potential advantages that it could bring to a crisis management process, as it is fast and accurate.

---



# Liste des abréviations

Voici la signification des abréviations ou acronymes rencontrés dans ce manuscrit.

<b>AIEA</b>	Agence Internationale de l'Énergie Atomique
<b>ASN</b>	Autorité de Sûreté Nucléaire (française)
<b>BRGM</b>	Bureau de Recherches Géologiques et Minières
<b>BSP</b>	Binary Space Partition (structure)
<b>BTP</b>	Bâtiments et Travaux Publics
<b>CAO</b>	Conception Assistée par Ordinateur
<b>CCD</b>	Charge Coupled Device
<b>CMM</b>	Coordinate Measuring Machine
<b>CNES</b>	Centre National d'Etudes Spatiales
<b>CSG</b>	Constructive Solid Geometry (Géométrie Implicite)
<b>DLR</b>	Deutsches Zentrum für Luft und Raumfahrt
<b>EDF</b>	Électricité de France
<b>ESA</b>	European Space Agency
<b>ERS</b>	European Remote Sensing satellites
<b>FEMA</b>	Federal Emergency Management Agency
<b>GPS</b>	Global Positioning System
<b>ICP</b>	Iterative Closest Point (algorithme)
<b>IGN</b>	Institut Géographique National
<b>JPSD</b>	Joint Precision Strike Demonstration Project
<b>MNS</b>	Modèle Numérique de Surface
<b>MNT</b>	Modèle Numérique de Terrain
<b>MVA</b>	Mathématiques Vision Apprentissage (master)
<b>ONERA</b>	Office National d'Etudes et de Recherches Aérospatiales
<b>PDA</b>	Personal Digital Assistant
<b>R&amp;D</b>	Recherche et Développement
<b>ROBEA</b>	ROBotique et Entités Artificielles (programme de recherche)
<b>RTV</b>	Rapid Terrain Visualization (projet)
<b>SERTIT</b>	Service Régional de Traitement d'Image et de Télédétection
<b>SIG</b>	Système d'Information Géographique
<b>SLAM</b>	Simultaneous Localization And Mapping
<b>STL</b>	Standard Template Library
<b>TQC</b>	Tel Que Construit
<b>TQE</b>	Tel Qu'Existante (équivalent à TQC)

---





---

# Table des matières

Remerciements	i
Résumé	iii
Abstract	iv
Liste des abréviations	vi
Table des matières	vii
Table des figures	xii
Liste des tableaux	xvi
Introduction	1
<b>1 Détection de changement géométrique tridimensionnel</b>	<b>5</b>
1.1 Principe général de la détection de changement . . . . .	5
1.2 Taxonomie des changements tridimensionnels . . . . .	7
1.2.1 Disparition/Apparition . . . . .	8
1.2.2 Déplacement . . . . .	9
1.2.3 Déformation . . . . .	10
1.2.4 Classification non spatiale . . . . .	11
1.3 Motivations . . . . .	12
1.4 État de l'art en comparaison de données 3D . . . . .	13
1.4.1 Méthodes existantes . . . . .	13
1.4.2 Solutions logicielles . . . . .	16
1.5 Techniques d'acquisition de données 3D . . . . .	17
1.5.1 Laser . . . . .	17
1.5.2 Photogrammétrie . . . . .	24
1.5.3 Radar . . . . .	26
1.5.4 Mesure par contact . . . . .	29
1.6 Discussion sur les techniques d'acquisition . . . . .	30

---

---

<b>2</b>	<b>Calcul de distance sur un nuage de points</b>	<b>33</b>
2.1	Remarques préliminaires . . . . .	33
2.2	Distances d'un nuage de points à un modèle 3D . . . . .	35
2.2.1	Distance point-triangle . . . . .	35
2.2.2	Intersection entre une grille 3D et un ensemble de polygones . . . . .	36
2.3	Distances entre deux nuages de points . . . . .	38
2.3.1	Distances inter-nuages . . . . .	39
2.3.2	Améliorations de la mesure de distance . . . . .	44
2.3.3	Gestion des problèmes liés aux points de vue du capteur . . . . .	47
2.4	Résultats . . . . .	51
2.4.1	Comparaison Nuage/Maillage . . . . .	51
2.4.2	Comparaison Nuage/Nuage . . . . .	53
2.4.3	Résolution du problème d'échantillonnage . . . . .	55
2.4.4	Résolution du problème de visibilité . . . . .	56
<b>3</b>	<b>Exploitation de la mesure de distance</b>	<b>59</b>
3.1	Rapide état de l'art sur la segmentation . . . . .	60
3.1.1	Segmentation 3D de nuages de points . . . . .	60
3.1.2	Techniques issues du traitement d'image classique . . . . .	61
3.2	Segmentation par classification . . . . .	64
3.2.1	Classification globale et manuelle par seuillage . . . . .	64
3.2.2	Classification globale et automatique par les K-moyennes . . . . .	67
3.2.3	Classification par analyse statistique locale . . . . .	69
3.2.4	Extraction des composantes connexes en 3D . . . . .	80
3.3	Segmentation par propagation de contour . . . . .	82
3.3.1	L'algorithme de Fast-Marching . . . . .	83
3.3.2	Segmentation locale par propagation . . . . .	85
3.4	Résultats . . . . .	93
3.5	Prise en compte de l'asymétrie de comparaison . . . . .	94
3.5.1	Cas des disparitions/apparitions . . . . .	96
3.5.2	Cas des déplacements . . . . .	98
3.5.3	Cas des déformations . . . . .	100
3.5.4	Synthèse . . . . .	101
<b>4</b>	<b>Octree à codage numérique</b>	<b>103</b>
4.1	Introduction . . . . .	103
4.2	Définition . . . . .	104
4.3	Construction d'une octree par codage numérique . . . . .	105
4.4	Recherche des voisins . . . . .	107
4.5	Heuristique de détermination du niveau d'octree optimal pour la détermination de multiples voisinages . . . . .	110
<b>5</b>	<b>Applications</b>	<b>115</b>
5.1	Aide à la réalisation de documentation T.Q.C. . . . .	115
5.2	Suivi de chantier . . . . .	118

---

---

5.3	Contrôle géométrique d'ouvrages de génie civil . . . . .	120
5.3.1	Cadre . . . . .	120
5.3.2	Confrontation des méthodes de comparaison . . . . .	122
5.4	Suivi de glissement de terrain . . . . .	127
5.5	Cartographie d'urgence . . . . .	131
<b>Conclusions et perspectives</b>		<b>134</b>
<b>A Amélioration de la perception</b>		<b>139</b>
A.1	Problèmes de perception . . . . .	139
A.1.1	Perception des changements . . . . .	139
A.1.2	Perception des données 3D ponctuelles . . . . .	140
A.2	Calcul rapide de normales . . . . .	142
A.2.1	Méthode 1 : ajustement de plans par la méthode des moindres carrés	142
A.2.2	Méthode 2 : triangulation locale en 2D . . . . .	143
A.3	Calcul de la portion de ciel visible . . . . .	145
A.3.1	Principe . . . . .	146
A.3.2	Algorithme en 3D . . . . .	146
A.3.3	Applications à des données réelles . . . . .	150
A.3.4	Lien avec la détection de changement . . . . .	151
<b>B Taxonomie visuelle des changements</b>		<b>153</b>
<b>C Logiciel CloudCompare</b>		<b>159</b>
C.1	Présentation . . . . .	159
C.2	Fonctionnalités . . . . .	160
<b>D Données 3D</b>		<b>163</b>
D.1	Primitives élémentaires . . . . .	163
D.1.1	Point . . . . .	163
D.1.2	Segment et polyligne . . . . .	163
D.1.3	Polygone et polyèdre . . . . .	164
D.2	Représentation de surfaces . . . . .	165
D.2.1	Maillages . . . . .	165
D.2.2	TIN . . . . .	165
D.2.3	Grille $2D\frac{1}{2}$ . . . . .	165
D.2.4	Splines, Surface de Bézier et NURBS . . . . .	165
D.2.5	Surface de subdivision . . . . .	167
D.2.6	Géométrie Implicite (C.S.G.) . . . . .	167
<b>E Articles</b>		<b>169</b>
E.1	CHANGE DETECTION ON POINTS CLOUD DATA ACQUIRED WITH A GROUND LASER SCANNER . . . . .	169
E.2	RENDU EN PORTION DE CIEL VISIBLE DE GROS NUAGES DE POINTS 3D .	169
E.3	A POINT-BASED APPROACH FOR CAPTURE, DISPLAY AND ILLUSTRATION OF VERY COMPLEX ARCHEOLOGICAL ARTIFACTS . . . . .	169

---

**Bibliographie**

**176**

---

# Table des figures

1	Scanner laser terrestre (de marque <i>Riegl</i> , à gauche) et nuage de points 3D résultant (560 000 points, colorés en fonction de l'intensité du signal retour, à droite). . . . .	1
1.1	Classes de changement et équivalences. . . . .	6
1.2	Arbre principal de la taxonomie des changements géométriques tridimensionnels. . . . .	7
1.3	Apparition/disparition sans contact. . . . .	8
1.4	Apparition/disparition avec contact. . . . .	9
1.5	Déplacement sans recouvrement. . . . .	9
1.6	Déplacement avec recouvrement. . . . .	10
1.7	Déformation sans recouvrement. . . . .	10
1.8	Déformation avec recouvrement. . . . .	11
1.9	Exemple de pré-classification des changements en fonction de la granularité de la scène observée et du contexte d'observation. . . . .	12
1.10	Vue de l'interface du logiciel commercial <i>Geomagic Qualify</i> (Raindrop). . .	16
1.11	Dépendance entre la granularité de la scène observée et systèmes de mesure. .	18
1.12	Principe du scanner laser à <i>mesure du temps de vol</i> . . . . .	19
1.13	Principe du capteur laser à mesure par <i>triangulation plane</i> . . . . .	19
1.14	Scanner laser terrestre. . . . .	22
1.15	Principe du relevé laser aéroporté (à gauche) et vue d'un modèle de terrain obtenu avec cette technique (à droite). . . . .	23
1.16	Principe de la photogrammétrie. . . . .	24
1.17	Exemple de M.N.S. (en haut) et M.N.T. (en bas) obtenus par interférométrie radar aéroportée (sur une zone boisée au Canada). . . . .	27
1.18	Mesure de la subsidence (affaissement progressif, enfoncement) du quartier de la gare Saint-Lazare par interférométrie satellitaire entre 1997 et 1999, suite à des travaux souterrains. . . . .	29
1.19	Système de mesure par contact. . . . .	30
2.1	Distance d'un point (P) à un triangle (ABC). . . . .	36
2.2	Calcul de distances entre un nuage de points et un modèle 3D triangulé. .	37
2.3	Illustration de l'erreur $\varepsilon$ d'estimation de la distance d'un point P à un plan ( $\pi$ ) par le calcul de la distance de P à un nuage de points échantillonnés sur ce plan. . . . .	41

---

---

2.4	Distribution théorique des distances au plus proche voisin dans un processus de Poisson ( $\lambda = 2$ , voir equation 2.4). . . . .	42
2.5	Valeurs mesurées (bleu) et théoriques (rouges) de l'erreur quadratique moyenne en fonction du nombre de points échantillonnés aléatoirement sur un carré de 100 mm de côté (échelle des abscisses logarithmique). . . . .	43
2.6	Visualisation des histogrammes des distances des points ( $K = 10000$ ) calculées par rapport à différents semis de points aléatoires tirés sur un carré de 100 mm de côté. . . . .	43
2.7	Principe de l'amélioration de la mesure sur nuages peu denses par modélisation locale. . . . .	45
2.8	Illustration du problème des zones d'ombre dans les scans laser. . . . .	48
2.9	En haut : nuage de points (projection ortho-graphique, coloration en fonction de la hauteur). En bas : carte de distance correspondante (coloration en fonction de la distance des points au capteur laser). . . . .	49
2.10	Illustration d'un nuage composé de multiples points de vue accompagné des cartes de profondeur de chacun (échelle de vert à rouge en fonction de l'éloignement). . . . .	51
2.11	Comparaison d'un nuage de points laser (a) avec un modèle 3D (non représenté), et résultat du calcul de distance (b). . . . .	52
2.12	Visualisation des distances des points du nuage du 3ème jour par rapport au nuage du 2ème jour (non représenté) du chantier de Malakoff. . . . .	54
2.13	Influence de l'échantillonnage sur le calcul des distances <i>au plus proche voisin</i> : calculs des écarts et histogrammes correspondant sans (a) et avec (b) modélisation locale. . . . .	55
2.14	Calcul des distances <i>au plus proche voisin</i> filtrées en fonction de la visibilité, entre le 2 <sup>ème</sup> et le 3 <sup>ème</sup> jour du chantier de Malakoff. . . . .	57
2.15	Calcul des distances <i>au plus proche voisin</i> entre deux trémies (puits à section trapézoïdale) d'une centrale EDF. . . . .	58
3.1	Illustration de l'algorithme de segmentation géométrique d'un modèle 3D à base de watershed proposé par [Sun et al. 2002]. . . . .	64
3.2	Illustration du système d'interface graphique permettant la définition interactive de seuils de coupure (et de répartition des couleurs) sur un champ scalaire surfacique. . . . .	65
3.3	Illustration de la segmentation globale par seuillage manuel sur les valeurs de déplacement. . . . .	66
3.4	Illustration du comportement de l'algorithme des K-moyennes. . . . .	69
3.5	Illustration de la variation spatiale de l'erreur sur les écarts. . . . .	71
3.6	Illustration du calcul de la distance du $\chi^2$ entre l'histogramme $\{N_i\}$ correspondant à $n$ échantillons et une distribution $P(x)$ . . . . .	72
3.7	Illustration de l'influence de la taille du voisinage sur le test statistique local. . . . .	74
3.8	Distribution de <i>Weibull</i> ( $a, b$ ) pour différentes valeur de $a$ ( $b = 3.0$ ). . . . .	75
3.9	Comparaison de l'adaptation d'une loi normale (A) et d'une loi de Weibull (B) à un même échantillon de points. . . . .	76

---

---

3.10	Illustration de la cause de divergence du calcul de la distance du $\chi^2$ , lorsque les échantillons (et les classes de l'histogramme correspondant) sont loin de la moyenne de la distribution théorique. . . . .	78
3.11	Influence de la position du seuil de coupure sur la classification par filtrage statistique. . . . .	79
3.12	Composantes connexes en 8-connexité, détournées sur une image noir et blanc.	80
3.13	Processus complet de segmentation par classification. . . . .	82
3.14	Illustration du principe des <i>level sets</i> en 2D. . . . .	83
3.15	Illustration du déroulement de l'algorithme de <i>Fast Marching</i> sur une grille 2D. . . . .	84
3.16	Illustration de la limitation de la norme du gradient (inférieure à 1) dans le cas d'un champ scalaire correspondant à une distance euclidienne entre des points et une entité quelconque. . . . .	88
3.17	Illustration du calcul de gradient (B) sur un nuage valué (A) puis lissage par filtre gaussien sur la norme du gradient (C). . . . .	88
3.18	Relation entre écarts (à gauche) et normes du gradient (à droite). . . . .	89
3.19	Illustration de la propagation d'un front avec l'algorithme de <i>Fast Marching</i> .	91
3.20	Temps de trajet lors d'une propagation géodésique à vitesse constante par l'algorithme de <i>Fast Marching</i> , en fonction de l'ordre d'arrivée aux noeuds (sur le nuage représenté en figure 3.19). . . . .	92
3.21	Temps de trajet lors d'une propagation géodésique contrainte par la valeur du gradient (avec $\alpha = 50$ ) par l'algorithme de <i>Fast Marching</i> , en fonction de l'ordre d'arrivée aux noeuds (sur le nuage représenté en figure 3.19). . . . .	92
3.22	Comparaison des algorithmes de segmentation sur un même nuage de points.	94
3.23	Calculs d'écarts "symétriques" (C,D) entre deux nuages de points (A,B), avec prise en compte des points de vue du capteur. . . . .	96
3.24	Exemples de comparaison symétrique dans le cas d'apparitions. . . . .	97
3.25	Exemple de comparaison symétrique dans le cas d'un déplacement sans recouvrement. . . . .	98
3.26	Exemple de comparaison symétrique dans le cas d'un déplacement avec recouvrement. . . . .	99
3.27	Exemple de comparaison symétrique dans le cas d'une déformation avec recouvrement. . . . .	101
3.28	Schéma de principe du processus de décision permettant la caractérisation d'un changement à partir d'une comparaison "symétrique". . . . .	102
4.1	Principe de l'octree. . . . .	103
4.2	Principe du codage de la position d'une sous-cellule. . . . .	106
4.3	Principe de la détermination des plus proches voisins (représentation en 2D).	109
4.4	Temps de calcul réel et estimé pour l'extraction du plus proche voisin ( $k = 1$ ).	112
5.1	Illustration du principe de modélisation 3D d'une installation industrielle à partir de mesures ponctuelles. . . . .	116
5.2	Illustration du processus de contrôle de réalisation de modèle TQC. . . . .	118
5.3	Aéroréfrigérant. . . . .	121

---



---

5.4	Approximation d'une courbe par un polygone. . . . .	123
5.5	Résultat de la méthode par comparaison à une surface spline (gauche) et résultats fournis par les prestataires d'EDF (droite). . . . .	124
5.6	Position et couverture des 7 stations d'acquisition de la double campagne de relevés sur l'aéroréfrigérant de la centrale de Cruas. . . . .	125
5.7	Comparaison des nuages de points laser sur un aéroréfrigérant. . . . .	126
5.8	Comparaison des histogrammes des écarts entre 0 et 20 cm. . . . .	126
5.9	Illustration des différents types de glissements de terrain. . . . .	128
5.10	Calcul des écarts sur une zone de glissement de terrain. . . . .	130
5.11	Mise en évidence des contours d'un glissement de terrain. . . . .	130
5.12	Nuages de points acquis par un capteur laser ILRIS-3D (Optech) représentant la partie éventrée du Pentagone (gauche) et l'hôtel <i>Marriot</i> , bâtiment no. 3 du World Trade Center (droite) après les attentats du 11 septembre 2001. . . . .	133
5.13	Nuages de points acquis par un capteur aéroporté au-dessus du site de <i>Ground-Zero</i> à New-York avant (gauche) et après (droite) les attentats du 11 septembre 2001. . . . .	133
5.14	Processus complet de détection des changements en 3D et correspondance avec les différentes parties de ce manuscrit. . . . .	136
5.15	Interface de CloudCompare (ici, comparaison entre un nuage de points et un modèle 3D). . . . .	137
A.1	Techniques basiques d'amélioration de la perception d'un nuage de points. . . . .	141
A.2	Calcul des normales par ajustement d'un plan par la méthode des moindres carrés (limitée aux cellules d'octree). . . . .	143
A.3	Calcul des normales par triangulation locale (limitée aux cellules d'octree). . . . .	144
A.4	Principe de l'illumination en <i>Portion de Ciel Visible</i> . . . . .	146
A.5	Illustration d'un rayon lumineux tel que considéré dans l'algorithme de calcul de <i>Portion de Ciel Visible</i> 3D. . . . .	147
A.6	Résultat de l'éclairage en <i>Portion de Ciel Visible</i> sur le Dragon de Stanford. . . . .	148
A.7	Illustration du problème de crénelage en fonction du niveau d'octree. . . . .	149
A.8	Rendu en <i>Portion de Ciel Visible</i> sur un nuage en intérieur de centrale. . . . .	150
A.9	Rendu en <i>Portion de Ciel Visible</i> sur un nuage en intérieur de centrale. . . . .	151
A.10	Fusion entre rendu en <i>Portion de Ciel Visible</i> (niveau 9) et représentation en fausses couleurs du calcul de distance (entre deux nuages). . . . .	152
C.1	Aperçu de l'interface graphique de <i>CloudCompare</i> . . . . .	160
D.1	Point 3D . . . . .	163
D.2	Segment, polyligne ouverte et polyligne fermée . . . . .	164
D.3	Polygone (à droite) et polyèdre (à gauche) . . . . .	164
D.4	T.I.N. (à droite) et Grille $2D\frac{1}{2}$ (à gauche). . . . .	166
D.5	Courbe spline (en haut) et surface spline (en bas). . . . .	166
D.6	Exemple de surface de subdivision (règle <i>Quad-Triangle</i> de Stam & Loop). . . . .	167

---

# Liste des tableaux

1.1	Comparaison des techniques d'acquisition 3D. . . . .	31
2.1	Temps de calcul indicatifs (en secondes) pour différents niveaux d'octree, lors de la comparaison d'un nuage de points à un modèle 3D triangulé avec un algorithme du type <i>Metro</i> . . . . .	53
2.2	Temps de calcul indicatifs (en secondes) pour la comparaison de deux nuages de points du chantier de Malakoff par calcul de la distance <i>au plus proche voisin</i> . . . . .	54
2.3	Comparaison des temps (en secondes) pour le calcul de la distance <i>au plus proche voisin</i> entre deux nuages <i>sans</i> et <i>avec</i> modélisation locale. . . . .	56
3.1	Temps de calcul indicatifs (en secondes) pour la segmentation de nuages de points par l'algorithme de filtrage statistique local. . . . .	93
3.2	Temps de calcul indicatifs (en secondes) pour la segmentation de nuages de points par l'algorithme de propagation contrainte par la norme du gradient. . . . .	93
4.1	Codage numérique de la position d'une cellule de l'octree . . . . .	106
5.1	Comparaison des temps (en secondes) pour le calcul de la distance au plus proche voisin entre les nuages de l'aéroréfrigérant de Cruas acquis en janvier et mai 2005, <i>sans</i> et <i>avec</i> modélisation locale. . . . .	125

---



# Introduction



FIG. 1 – Scanner laser terrestre (de marque *Riegl*, à gauche) et nuage de points 3D résultant (560 000 points, colorés en fonction de l'intensité du signal retour, à droite).

Le besoin de caractériser les dimensions et la géométrie d'objets ou de portions de terrains s'est fait sentir bien avant notre ère : les premières mesures connues sont attribuées aux égyptiens, l'invention de la cartographie à Anaximandre (philosophe, mathématicien et astronome grec,  $V^{eme}$  siècle avant J.C.), et celles du niveau à eau et du cadastre aux romains. L'arpentage apparaît au moyen-âge, puis la topographie et la géographie connaissent un réel essor à partir du  $XVII^{eme}$  siècle, grâce aux avancées scientifiques et à la professionnalisation de la guerre. Enfin, la mesure tridimensionnelle *rapide* est rendue possible par l'invention du principe de la photogrammétrie<sup>1</sup> en 1849 par le Colonel Laussedat, technique qui sera concrétisée avec l'apparition de la photographie et qui est encore aujourd'hui prédominante dans le domaine de la mesure 3D.

Ces dix dernières années ont vu le développement et la démocratisation de nouveaux systèmes de mesure tridimensionnelle rapide et sans contact dénommés communément

---

<sup>1</sup>Science et art dont le sujet d'étude est la photographie dans l'intention de recueillir des données conduisant à des restitutions dimensionnelles et de déterminer la forme et la position d'un objet dans l'espace (d'après "Terminologie de Télédétection et Photogrammétrie", PUF, 1997).

---

*scanners laser* (voir figure 1). Il en existe déjà de nombreux modèles, ayant des portées allant du décimètre à plusieurs kilomètres. Leur précision est aussi très variable (entre quelques micromètres et quelques centaines de mètres) et dépend principalement de leur portée. Ils fonctionnent néanmoins tous selon le même principe : balayer l'espace avec un rayon laser qui leur permet de déterminer selon un grand nombre de directions la distance à laquelle se trouvent les objets (au niveau du point de contact entre le rayon et la surface de l'objet). Ils sont portables pour la plupart et sinon *véhiculables*. Ils sont enfin et surtout très rapides et fournissent en sortie du capteur une information 3D directement exploitable.

Prenant beaucoup d'importance par rapport à la photogrammétrie, les scanners laser sont en train de transformer en profondeur le domaine de la mesure 3D. Bien que souvent vus comme concurrents, les scanners laser ne devraient pas à terme remplacer totalement les systèmes photogrammétriques. Mais il est vrai que leurs cadences très élevées pour des précisions souvent excellentes par rapport à nombre de besoins font sortir de l'ombre certaines applications jusqu'ici trop fastidieuses, en accélèrent et en simplifient de nombreuses autres, et en font même apparaître de nouvelles. Les domaines qui sont aujourd'hui les plus touchés par ce véritable *phénomène*, outre l'industrie qui a très rapidement pu profiter de cette technologie de pointe (notamment grâce à des ressources financières importantes), sont la topographie, la conservation du patrimoine, l'archéologie, les métiers de la réalité virtuelle, et plus indirectement la géomatique<sup>2</sup>. C'est d'ailleurs au carrefour de ces différentes disciplines que se placent les travaux présentés dans ce manuscrit.

Parallèlement au développement des scanners laser, le vaste domaine de la détection de changement a connu lui aussi un fort regain d'intérêt, bien que cette notion soit un peu plus subjective étant donnée l'ampleur des techniques et disciplines englobées par ce terme. Néanmoins, si on se restreint au domaine de la télédétection des phénomènes naturels ou créés par l'homme, on a pu remarquer une très forte augmentation des articles scientifiques et applications concrètes de détection de changement au cours des dernières années. Ils concernent en premier lieu les moyens d'acquisition "grande échelle", avec les satellites optiques qui ont aujourd'hui des résolutions sub-métriques (*Quickbird*, *Ikonos*, et bientôt la constellation *Pléiades*) et les radars interférométriques ou les systèmes laser aéroportés qui peuvent couvrir rapidement des larges zones de terrain. Le milieu urbain d'une part et les entités géologiques de grande taille d'autre part (volcans, glaciers, glissements de terrain, etc.) focalisent une bonne partie des sujets de recherche.

Sans prétendre expliquer cet engouement, on peut tout de même mettre en avant le besoin vital, puis stratégique et désormais économique pour l'homme de suivre et prédire l'évolution de son environnement, afin de le contrôler ou à défaut, de réagir avant qu'il ne soit trop tard. C'est dans cette problématique que se place cette thèse, bien qu'elle s'applique à des échelles plus petites (de l'ordre de la taille d'un important site industriel, comme une raffinerie ou une centrale électrique). Elle utilise aussi des technologies et des données encore assez peu étudiées par le monde du traitement d'image classique, pourtant

---

<sup>2</sup>Ce sont les métiers de l'informatique appliqués à la géographie, en particulier à travers des S.I.G. (Systèmes d'Information Géographique).

---

très impliqué dans le domaine de la télédétection.

Dans ce document, on commence par évoquer des généralités sur la détection de changement géométrique au Chapitre 1. On y propose tout d'abord une taxonomie des changements, puis un état de l'art du domaine à travers un rappel des techniques existantes de comparaison de données tridimensionnelles, ainsi qu'une étude rapide des différentes techniques d'acquisition 3D qui nous a conduit au choix de la technologie laser comme base de nos travaux. Le Chapitre 2 présente en détail et d'un point de vue algorithmique deux méthodes de calcul de distance entre des données 3D typiquement utilisées pour de telles applications, à savoir des nuages de points et des maillages surfaciques. La première méthode, qui permet de comparer un nuage de points 3D à un maillage surfacique triangulaire, est pré-existante à la thèse mais est ici accélérée. La deuxième méthode, inédite, permet de comparer directement deux nuages de points. Le but est de gagner un temps précieux en évitant une phase coûteuse ou hasardeuse de modélisation d'un des deux nuages. Mais ce cas étant plus complexe et les données généralement moins propres, plusieurs moyens d'améliorer le résultat sont proposés. Le Chapitre 3 présente des techniques de segmentation du nuage à partir de l'information de distance, dont notamment une méthode par filtrage statistique local et une méthode par propagation contrainte par la norme du gradient du champ des écarts calculés en chaque point. On considère aussi à la fin de ce chapitre les apports potentiels d'un processus de comparaison *symétrique* à l'analyse des changements. Ce chapitre traite donc de la détection de changements géométriques à proprement parler. Le Chapitre 4 présente une implémentation spécifique d'une structure d'accélération des calculs sur un nuage de points (dénommée *octree*), qui profite à la plupart des algorithmes de traitement présentés dans ce manuscrit. Les différents algorithmes documentés aux chapitres 2, 3 et 4 ont été implémentés et testés au sein d'un logiciel développé spécialement pour cette thèse et dénommé *CloudCompare* (voir Annexe C). Enfin, toutes les méthodes présentées jusque-là étant plus liées à la nature des données qu'aux applications pour lesquelles elles ont été développées (à savoir d'une part le contrôle et l'aide à la réalisation de documentation T.Q.C.<sup>3</sup>, et d'autre part le suivi de chantier), nous avons voulu montrer qu'elles étaient généralisables dans une certaine mesure. Le Chapitre 5 illustre donc le champ d'application des méthodes développées au cours de cette thèse, en l'étendant notamment à des utilisations non initialement prévues et à caractère moins industriel. Enfin, un bilan de la méthode globale de détection des changements géométriques en 3D proposée dans ce manuscrit, est fait en conclusion. On y propose aussi plusieurs pistes de développements futurs et des ouvertures potentielles vers d'autres applications.

Daniel Girardeau-Montaut.

*Remarque : Cette thèse est financée par E.D.F. sous convention CIFRE<sup>4</sup>.*

---

<sup>3</sup>Tel Que Construit.

<sup>4</sup>Conventions Industrielles de Formation par la Recherche.

---



# Chapitre 1

## Détection de changement géométrique tridimensionnel

### 1.1 Principe général de la détection de changement

Le principe général de la *détection de changement géométrique en 3D* est celui de la localisation et si possible de la caractérisation des changements structurels survenus sur un objet<sup>1</sup> à partir de représentations numériques tridimensionnelles de celui-ci. Ces représentations peuvent être partielles ou schématiques. Les changements structurels considérés sont des disparitions ou apparitions, des déplacements et enfin des déformations. Leur nature et leur cause peuvent être quelconques. Ils peuvent intervenir sur un même objet entre deux époques (changements *temporels*) ou bien entre deux objets distincts ayant des structures théoriquement identiques ou proches (changements *atemporels*).

Une première remarque importante peut être faite sur ce dernier point, à savoir l'existence de changements temporels et atemporels. Il est vrai qu'on utilise plus communément le terme de *différences* plutôt que celui de *changements atemporels* lorsque l'on compare deux objets distincts. On peut pourtant toujours considérer avec un minimum d'imagination que deux entités indépendantes mais ayant la même configuration (comme deux voitures du même modèle par exemple) sont le même objet considéré à deux instants différents. Cet objet aura pu subir localement un certain nombre de déformations, déplacements et ajouts ou suppressions de composants. L'amalgame est d'autant plus facilement acceptable si l'on ne considère que les représentations numériques de ces objets. Il peut ainsi être parfois très difficile de déterminer si deux jeux de mesure ont été acquis sur un même objet à deux instants différents ou sur deux objets différents à la même époque (voire à des époques différentes). Le raisonnement inverse est bien sûr tout à fait possible. Il n'y a donc pas de différence fonctionnelle entre une comparaison *temporelle* et une com-

---

<sup>1</sup>Où un ensemble d'objets. L'amalgame entre ces deux notions sera fait systématiquement dans la suite de ce document si rien n'est précisé.

---



paraison *atemporelle*. Il n'y aura donc pas de différenciation des deux cas de figure dans la suite de ce manuscrit, et on parlera indifféremment de *différences* ou de *changements*.

Cette première remarque nous permet de faire une autre simplification des hypothèses de départ (relatives aux types de changements structuraux à considérer, et notamment concernant les déplacements). Puisque l'on peut toujours se ramener à un schéma de comparaison de type *temporel*, un déplacement<sup>2</sup> pourra être considéré comme la succession de deux événements : la disparition de l'objet de sa position initiale, suivie de sa réapparition à un autre endroit et/ou selon une autre orientation, ces deux événements ayant pu se dérouler à des époques quelconques comprises dans l'intervalle de temps séparant les deux mesures. La connaissance de la date précise de ces événements (et qui plus est de leur réalité) n'est pas requise. En effet, il n'est pas nécessaire de faire la différence entre un objet qui se déplace et un objet qui disparaît et réapparaît ailleurs (destruction puis reconstruction d'un bâtiment par exemple). Dans le cas général, la trajectoire réelle d'un objet se déplaçant entre deux points ne peut de toute façon pas être déduite à partir de la simple donnée de ses positions de départ et d'arrivée.

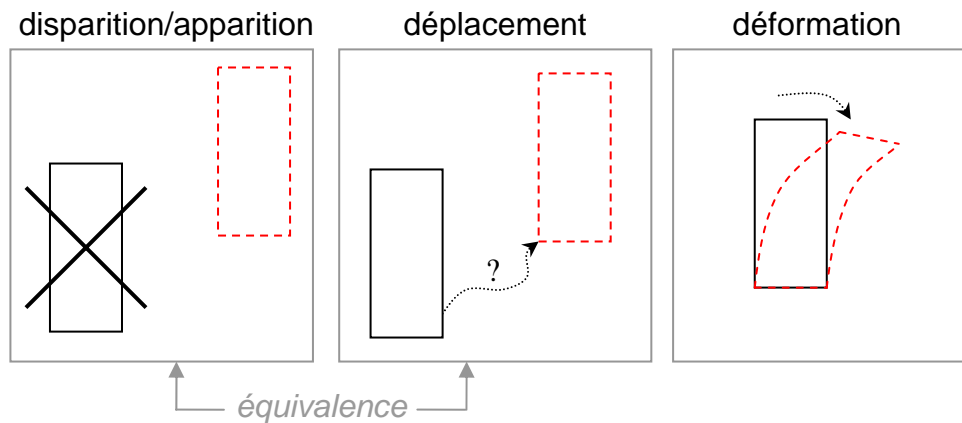


FIG. 1.1 – Classes de changement et équivalences.

Certes, dans certains cas très précis, l'existence d'un phénomène de modification de l'environnement caractéristique d'un déplacement de l'objet (comme des traces de roues), ou encore des contraintes de forme ou de fonctionnement de l'objet (comme une péniche sur un canal), pourraient permettre de déduire cette trajectoire ou au moins des étapes intermédiaires. Mais cette étude de trajectoire ne présente pas de réel intérêt pour les applications abordées dans le cadre de ce travail. L'information "tel objet s'est *déplacé* et se retrouve maintenant à tel endroit" est tout à fait suffisante et suffisamment complexe à obtenir. De plus, la différenciation entre un déplacement continu et un phénomène de disparition/apparition est généralement faite naturellement pour un observateur humain possédant un minimum de connaissances *a priori* de l'objet ou de la scène observée. Le

<sup>2</sup>Combinaison d'une translation et d'une rotation. Aussi appelé *déformation solide*.

développement d'algorithmes lourds et potentiellement approximatifs n'a donc pas été jugé très pertinent.

Une dernière remarque très importante est qu'un tel processus de comparaison n'est pas symétrique. On compare en effet la mesure d'un objet à un instant donné à une mesure antérieure considérée comme référence. Le résultat du processus de comparaison est donc toujours exprimé par rapport à cet état de référence. Ainsi, entre deux dates on considérera qu'un objet *disparaît* alors que si l'on inverse la flèche du temps, on considérera qu'il *apparaît*. Cette différence n'est cependant pas purement lexicale mais est au contraire étroitement liée au processus logique sous-jacent et pourra se retrouver en pratique dans les résultats des algorithmes de détection de changement. On veillera donc à étudier l'impact de l'asymétrie du processus de comparaison.

## 1.2 Taxonomie des changements tridimensionnels

Nous présentons dans cette section une taxonomie sommaire des changements géométriques que nous étudierons dans le cadre de cette thèse (l'annexe B présente des illustrations de chaque classe de cette taxonomie).

Cette taxonomie reprend la répartition des changements en trois classes principales : apparition/disparition, déplacement et déformation. Chacune de ces classes est séparée en deux sous-classes : "avec contact" et "sans contact" dans le cas des apparitions/disparitions et "avec recouvrement" et "sans recouvrement" dans les deux autres cas (Cf. figure 1.2).

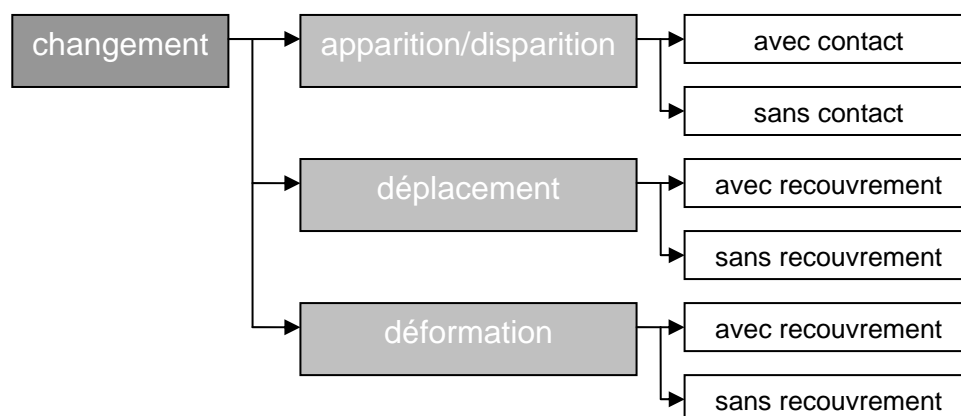


FIG. 1.2 – Arbre principal de la taxonomie des changements géométriques tridimensionnels.

On entend par "contact", dans le cas des apparitions/disparitions, le fait que la surface

---

de l'objet était initialement en contact ou bien qu'elle entre en contact avec une ou des surfaces immobiles de son environnement, et que ce contact est visible dans les données tridimensionnelles. De même, le terme de "recouvrement" dans les cas des déplacements et déformations, décrit le fait que la surface de l'objet après avoir subi le changement reste en intersection totale ou partielle avec la position qu'elle occupait avant. Mais nous verrons ceci plus en détail dans les sections suivantes.

Dans les sections qui suivent, on symbolise les différents types de changements par des schémas génériques simples : deux époques ou états différents nommés  $T_1$  et  $T_2$  sont mis en vis-à-vis. Les objets d'intérêt (ceux qui subissent le changement) sont en gris, leur position dans l'autre état peut-être rappelée par un trait pointillé et les surfaces immobiles sont représentées sous forme de traits noirs pleins.

### 1.2.1 Disparition/Apparition

#### Sans contact

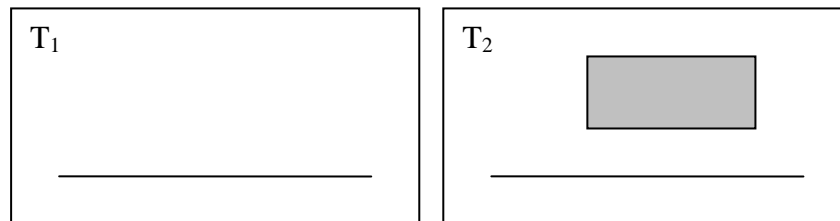


FIG. 1.3 – Apparition/disparition sans contact.

C'est sans doute le cas le plus simple : celui de l'apparition ou de la disparition entre les dates  $T_1$  et  $T_2$  d'un objet (décrit par un groupe de points 3D, un ensemble de facettes, etc.) qui *flotte*, typiquement porté ou suspendu par une structure invisible dans la représentation numérique (poutre, câble, etc.).

Le point important ici est donc l'absence de contact **dans la représentation numérique**, qui simplifiera grandement la détection du changement et la segmentation de l'objet.

#### Avec contact

L'objet qui apparaît ou disparaît repose ou s'appuie sur une surface préexistante qui n'a pas bougé mais dont tout ou partie est caché à  $T_2$  dans la représentation numérique.

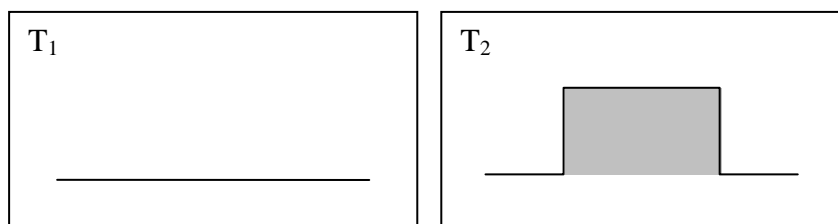


FIG. 1.4 – Apparition/disparition avec contact.

La seule différence avec le cas précédent est le contact entre l'objet et son environnement (on parle toujours de contact *visible* dans la représentation numérique). Celui-ci risque de compliquer l'étape de segmentation. Le cas du changement de type disparition/apparition reste cependant globalement le plus simple à détecter et à segmenter.

### 1.2.2 Déplacement

Le concept de déplacement désigne la combinaison d'une translation et d'une rotation (qu'on peut retrouver dans certains ouvrages sous le terme *déformation solide*).

Le terme de recouvrement désigne ici le fait que la surface de l'objet à  $T_2$  reste en intersection avec sa position à  $T_1$ .

#### Sans recouvrement

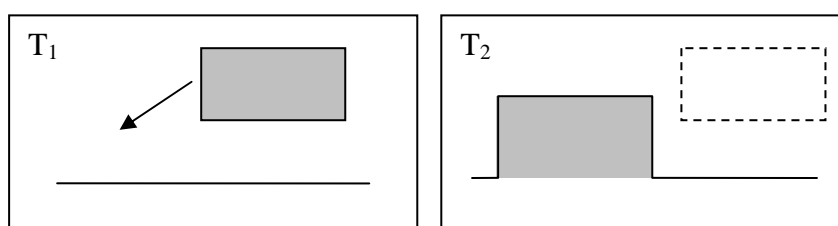


FIG. 1.5 – Déplacement sans recouvrement.

S'il n'y a pas de recouvrement, le déplacement est équivalent à une disparition suivie d'une apparition (avec ou sans contact) et cela en se référant à la remarque faite en section 1.1.

Dans l'exemple schématisé en figure 1.5, on a ainsi une disparition sans contact ( $T_1$ ) suivie d'une apparition avec contact ( $T_2$ ). Les remarques faites dans la section précédente s'appliquent donc de manière équivalente. La notion de *contact* est ici secondaire.

### Avec recouvrement

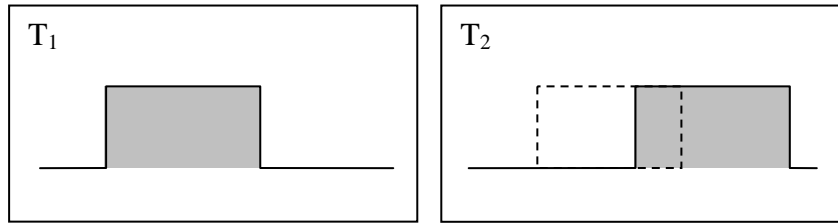


FIG. 1.6 – Déplacement avec recouvrement.

Dans cette configuration, l'objet repose ou non sur une surface préexistante (la notion de *contact* est ici encore secondaire), et se déplace à l'intérieur du volume qu'il délimitait à  $T_1$ . Il rentre en quelque sorte en intersection *avec lui-même*. C'est un cas qui pourra s'avérer beaucoup plus compliqué à détecter et surtout à segmenter proprement.

### 1.2.3 Déformation

La notion de déformation correspond aux phénomènes de déformation non solides et potentiellement chaotiques (torsion, effondrement, destruction, etc.) de la surface de l'objet d'intérêt. Cette classe de changement regroupe en pratique tout ce qui ne rentre pas dans les deux précédentes. La notion de recouvrement est la même que dans la section précédente.

#### Sans recouvrement

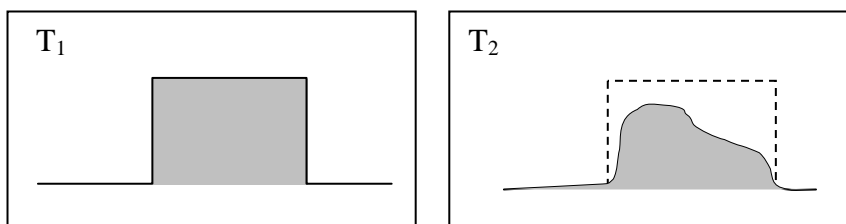


FIG. 1.7 – Déformation sans recouvrement.

L'objet subit une déformation non solide (torsion, effondrement, destruction, etc.) et sa surface à  $T_2$  n'est pas en intersection avec la surface occupée précédemment à  $T_1$ .

Ce cas, devrait rester relativement simple à détecter et à segmenter, mais son interprétation risque d'être très difficile (de par la nature chaotique du changement).

### Avec recouvrement

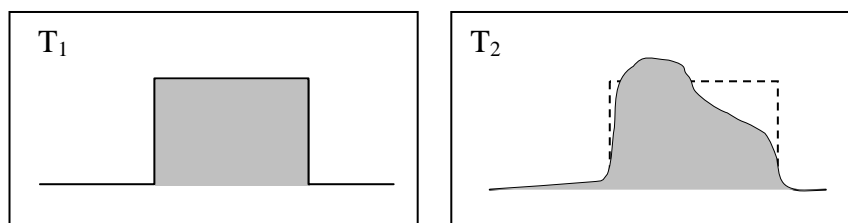


FIG. 1.8 – Déformation avec recouvrement.

L'objet subit une déformation non solide et sa surface à  $T_2$  est en intersection avec la surface occupée par celle-ci précédemment à  $T_1$ .

On est proche ici du cas du déplacement avec recouvrement (voir section précédente) mis à part qu'il n'y a plus forcément de rapport entre l'état de la surface avant changement et celui après changement. De plus, non seulement l'interprétation du changement risque d'être compliquée, mais sa simple segmentation risque d'être à elle seule très problématique. C'est sans conteste le cas le plus difficile.

#### 1.2.4 Classification non spatiale

Outre ce découpage qu'on peut qualifier de *spatial* (car réalisé en fonction des relations spatiales entre l'objet en mouvement et les objets immobiles, ainsi qu'entre les deux positions de l'objet en mouvement), on pourrait ajouter un découpage préalable en fonction du cadre d'application des techniques de détection ou encore de la taille de la scène analysée ou bien aussi de la taille des objets ayant subi des changements. Un tel découpage serait bien sûr un peu plus subjectif. Il ferait vraisemblablement varier les paramètres de segmentation et la sensibilité globale du processus. Il peut avoir comme avantage de supprimer certaines branches du découpage *spatial* en fonction de l'application ou du contexte de la détection de changement. Il est par contre virtuellement infini (ou du moins très vaste) et nous ne tenterons donc pas ici d'en donner une liste exhaustive.

On peut néanmoins proposer (voir figure 1.9) un exemple de classification préalable telle que l'on pourrait l'envisager dans le domaine industriel, qui sera abordé plus en détail dans la section suivante.

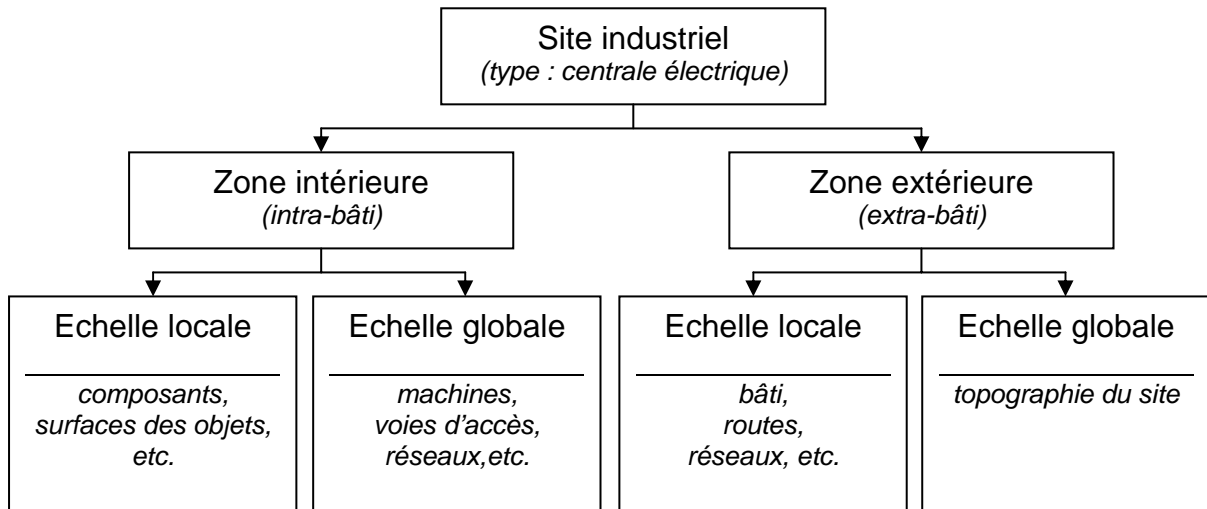


FIG. 1.9 – Exemple de pré-classification des changements en fonction de la granularité de la scène observée et du contexte d'observation.

*Dans les feuilles de l'arbre, la partie inférieure liste des exemples d'entités concernées par la procédure de détection de changement.*

### 1.3 Motivations

Les domaines où le besoin de détection de changement géométrique se fait ressentir sont de plus en plus nombreux et cela n'est pas étonnant dans un univers où la fréquence des changements s'intensifie et où les systèmes se compliquent<sup>3</sup>. Contrôler et suivre ces évolutions devient d'autant plus critique et stratégique que leur perception directe par l'homme est difficile voire physiquement impossible. EDF est particulièrement touchée par ces difficultés, de par ses activités. Son ex-statut de monopole public et la politique d'indépendance énergétique française développée depuis la fin de la seconde guerre mondiale en ont fait un des principaux gestionnaires de centrales nucléaires au monde, avec 58 réacteurs répartis sur 19 sites, auxquelles s'ajoute un nombre important de centrales hydroélectriques et thermiques. A l'instar des groupes pétrochimiques, EDF doit faire face quotidiennement à des problèmes liés à la taille et à la complexité de ses installations et doit aussi prouver sa capacité à faire face à de nouveaux problèmes ou de nouvelles contraintes, en particulier vis-à-vis de l'Autorité de Sûreté Nucléaire française (A.S.N.) mais aussi de l'opinion publique.

De plus, et paradoxalement, les opérations effectuées sur des systèmes toujours plus

<sup>3</sup>L'actualité récente nous en a donné un magnifique exemple avec la navette américaine Discovery qui embarquait lors de son vol au début du mois d'août 2005 un scanner laser (*Neptec*). Celui-ci était monté au bout du bras de la navette et a intégralement scanné la surface de l'appareil pour la comparer à son état "avant décollage" (pour détecter les différents impacts dus aux débris des propulseurs lors du décollage, qui ont causé la destruction de la précédente navette, Columbia, en 2003).

complexes doivent être toujours plus rapides, souvent pour des impératifs économiques ou de sécurité. Là encore, EDF est particulièrement sensible à ce phénomène, en particulier au niveau de son activité nucléaire. Les opérations effectuées dans les zones radioactives pouvant porter atteinte à la santé des opérateurs, elles doivent se dérouler le plus rapidement possible (et dans les meilleures conditions de sécurité). Mais un autre aspect de nature économique intervient également : les 58 réacteurs représentant près de 88% de la production totale d'énergie, l'arrêt d'une unité représente un pourcentage important de la production totale et a donc un coût direct très élevé en termes de perte de vente. Or, le vieillissement des moyens de production, l'augmentation de la consommation d'électricité, l'ouverture du marché européen, la privatisation progressive d'EDF, et enfin l'augmentation des températures moyennes provoquent une hausse significative des arrêts imprévus de réacteurs ainsi qu'une baisse des réserves d'eau nécessaires au fonctionnement des usines hydroélectriques en appoint. Ce sont autant de facteurs aggravants du coût, principalement économique mais aussi social, d'un arrêt de production. Force est donc de constater que le moindre arrêt d'un réacteur est extrêmement pénalisant à bien des points de vue. L'augmentation de la vitesse et de l'efficacité des opérations de maintenance, de réparation ou de construction devient alors critique.

Les travaux réalisés au cours de cette thèse ont été motivés principalement par deux applications ayant un caractère stratégique pour EDF. Elles nécessitent toutes les deux des outils permettant de faire de la détection de changement géométrique sous des contraintes de rapidité importantes. Ces deux applications sont d'une part *l'aide à la réalisation de documentation T.Q.C.* et d'autre part *le suivi de chantier* (elles sont détaillées dans le Chapitre 5). Néanmoins, ce manuscrit et les méthodes qui y sont exposées ont une vocation plus large et tenteront de répondre au problème de la détection de changement en limitant au maximum les suppositions et contraintes liées à ces applications, qui devraient plutôt être vues comme des fils conducteurs qui nous ont aidé à faire certains choix technologiques et à orienter nos travaux.

## 1.4 État de l'art en comparaison de données 3D

On s'intéresse ici aux méthodes et outils de comparaison de données 3D existants. Les techniques d'acquisition de ces données à partir de la réalité sont abordées dans la section suivante.

### 1.4.1 Méthodes existantes

On peut diviser les méthodes de mesure de différences géométriques 3D en deux catégories : des méthodes *surfaiques* (où l'on s'intéresse aux mouvements de la surface de l'objet) et des méthodes *volumiques* (où l'on veut quantifier les volumes déplacés).

---



Le calcul de différences *volumiques* ne fait pas partie des préoccupations de cette thèse. Pour mémoire, on peut dire qu'en général sont utilisées soit des approches algébriques *locales* d'intersection et d'union de modèles 3D (sur des maillages triangulaires et tétraédriques en particulier), soit des approches *globales* utilisant des descripteurs ou autres mesures de formes (moments invariants, squelettes, graphes de Reeb, harmoniques sphériques, etc). On peut se référer pour cette deuxième et vaste catégorie d'approches à [Tung 2005]. Les techniques de calcul de différences volumiques sont appliquées en général dans le premier cas à des modèles de terrains  $2D^{1/2}$  (pour le calcul de volumes extraits ou déplacés, dans le domaine de l'extraction minière ou pour du suivi de glissements de terrain typiquement) ou dans le deuxième cas à des modèles 3D complets d'objets uniques (pièces mécaniques, objets d'art, ou autres, pour de la recherche dans des bases de données 3D typiquement). On peut retenir que toutes ces techniques sont encore relativement lentes et parfois inopérantes avec des modèles véritablement 3D ayant une topologie complexe ou composés de plusieurs parties disjointes. Notons qu'il faut forcément disposer d'une notion de surface fermée sur les deux jeux de données pour pouvoir calculer un volume. Le calcul de différences volumiques n'est donc pas possible directement entre un nuage de points et un modèle ou entre deux nuages.

Les méthodes surfaciques reposent majoritairement aujourd'hui sur un même principe : la comparaison d'un nuage de points et d'un modèle (on va voir qu'on peut en effet toujours se ramener à une telle configuration). La quasi-unicité de cette approche s'explique par le fait qu'il est beaucoup plus simple et rapide de calculer la distance d'un point à une primitive 3D (triangle, plan, cylindre, etc.) que de calculer directement la distance - ou plutôt *le champ surfacique de distance* - entre deux primitives 3D. Dans les applications industrielles et académiques de comparaison géométrique, la référence prend toujours la forme d'un modèle 3D. Ce modèle peut avoir été conçu dans un cabinet d'étude ou par synthèse automatique, ou bien encore avoir été calculé à partir de mesures ponctuelles. Il y a par contre moins d'exclusivité sur le format de l'objet comparé qui peut prendre la forme aussi bien d'un modèle que d'un nuage. Lorsque l'on veut comparer deux modèles, on préfère en pratique échantillonner des points 3D sur la surface du modèle à comparer et revenir ainsi au schéma simple de comparaison d'un nuage à un modèle. Si l'échantillonnage est suffisamment dense, on peut retrouver une bonne approximation du champ des écarts par interpolation des valeurs calculées aux différents points. Inversement, lorsque l'on veut comparer deux nuages de points, on passe généralement par une étape préalable de modélisation d'un des deux nuages.

La communauté scientifique s'est semble-t-il peu intéressée au problème du calcul de distances entre un nuage de point et un maillage qui est, il est vrai, assez simple sous cette forme (aussi bien d'un point de vue mathématique qu'algorithmique). Les aspects statistiques entourant l'erreur faite sur les mesures ponctuelles ou sur la mesure de distance étaient beaucoup plus attirants. On peut tout de même citer quelques travaux isolés mais intéressants, ayant un lien avec le problème de la comparaison surfacique de données 3D :

- [Brick et al. 2004] se base sur un processus de comparaison 3D entre un nuage de points et un modèle C.A.O. théorique. Le calcul de distance est effectué par un logiciel commercial (Surfacer®). Le but est de trouver la meilleure homothétie qui
-

permet d'adapter un modèle C.A.O. de forme fixe à un nuage de points dont la forme (une dent) est globalement proche du modèle, mais légèrement affinée suite à un phénomène d'érosion et aussi beaucoup plus détaillée.

- [Bannoa 2004] présente une application de comparaison de surfaces 3D de précision micrométrique (surfaces de balles *imagées* après utilisation) avec un algorithme spécifique qui donne une évaluation binaire de la concordance locale des surfaces. Il n'y a pas de quantification précise de la déformation et la visualisation est binaire, en noir et blanc.
- Plus proche du centre d'intérêt de cette thèse, [Shih et al. 2004] expose un retour d'expérience de suivi d'un chantier par une série de relevés laser effectués à différentes dates. Là encore, la comparaison est binaire mais se fait directement entre deux nuages de points (l'algorithme n'est pas explicité). Le but est de mettre en évidence les parties ajoutées à la structure entre deux époques.
- [Breckon et Fischer 2004] propose une méthode de vérification d'un modèle 3D d'usine par comparaison avec des données acquises par un capteur laser (selon un unique point de vue). Le nuage de points est segmenté automatiquement en fonction de critères de courbure et de correspondance avec des primitives géométriques élémentaires (les auteurs se limitant à des *environnements industriels simples*, pour reprendre leurs termes). Le modèle théorique doit lui aussi être assez simple (voire même *très simple*, au vu de l'exemple unique donné dans l'article). La comparaison se fait alors entre les surfaces segmentées et les surfaces du modèle 3D (via des comparaisons d'extension, d'orientation et de courbure). Cette méthode n'est pas du tout applicable aux données dont EDF dispose et dépend énormément de la qualité de la segmentation préalable du nuage. Mais l'article présente, par contre, un double intérêt : d'une part sa bibliographie sur la reconnaissance d'objets en 3D, et d'autre part un système de confrontation de la profondeur des points laser et des surfaces du modèle correspondantes. Cette technique présente quelques similarités avec une méthode que nous avons mise au point pour filtrer le résultat d'une comparaison directe de deux nuages de points issus de scanner laser (Cf. section 2.3.3).
- [Gonçalves et al. 2004] se rapproche des travaux présentés dans ce manuscrit, en décrivant une application voisine de la documentation T.Q.C. Il traite de la comparaison d'un nuage laser (acquis dans une centrale nucléaire) avec un modèle 3D. Le but est de vérifier la concordance entre la réalité et les plans fournis par l'opérateur de la centrale à un organisme de contrôle (l'AIEA<sup>4</sup> dans ce cas). Le processus de comparaison et d'analyse est par contre principalement manuel.
- Enfin, il semble que la comparaison directe de deux nuages de points n'ait été abordée que par Memoli et Sapiro [Memoli et Sapiro 2004]. La méthode présentée dans cet article, très mathématique, n'est pas applicable sur des données réelles telles que celles qui peuvent être mesurées en milieu industriel. Le but est la comparaison de formes pour un problème de classification d'objets 3D. Les formes sont donc certes décrites par des nuages de points, mais leur échantillonnage doit être uniforme et dense sur toute la surface. Elles doivent de plus être topologiquement *simples* (manifold). Enfin le résultat de la comparaison est global et non local. Néanmoins,

---

<sup>4</sup>Agence Internationale de l'Énergie Atomique.

---

d'un point de vue général les travaux de Mémoli et Sapiro sur les nuages de points sont très intéressants et seront cités à plusieurs reprises dans ce manuscrit.

### 1.4.2 Solutions logicielles

Les solutions logicielles sont généralement adaptées à une application particulière, non pas pour des questions algorithmiques mais plutôt à cause des formats des données qui dépendent beaucoup de la technique utilisée pour les acquérir ainsi que des habitudes de chaque profession. On peut dire que chaque type d'utilisateur (topographe, ingénieur dans l'industrie, chercheur en informatique 3D, etc.) dispose de ses propres outils orientés *métier*.

Dans le domaine industriel (industrie manufacturière en particulier), la problématique est souvent de comparer une pièce à un modèle. Les logiciels vont donc se baser sur des données C.A.O. comme référence, et des données ponctuelles 3D issues de capteurs optiques ou par contact (et exprimées dans des formats *propriétaires* en général). Ils sont axés sur la production de rapports de comparaison synthétiques et détaillés, mettant en avant les défauts et différences de la pièce par rapport au modèle. L'interface est en général très travaillée, et prime largement sur l'algorithmique (mises à part certaines fonctions avancées qui permettent la génération plus ou moins automatique d'un modèle 3D à partir des données ponctuelles, à condition que celles-ci décrivent de manière uniforme des formes topologiquement simples). Exemple de logiciels (commerciaux) : *Imageware Inspect* de Metrix, *EvalViewer* de Alias, *CADCompare* de Metris, *Polyworks* d'InnovMetric ou encore *Geomagic Qualify* de Raindrop et *XO Verifier* de Rapidform.

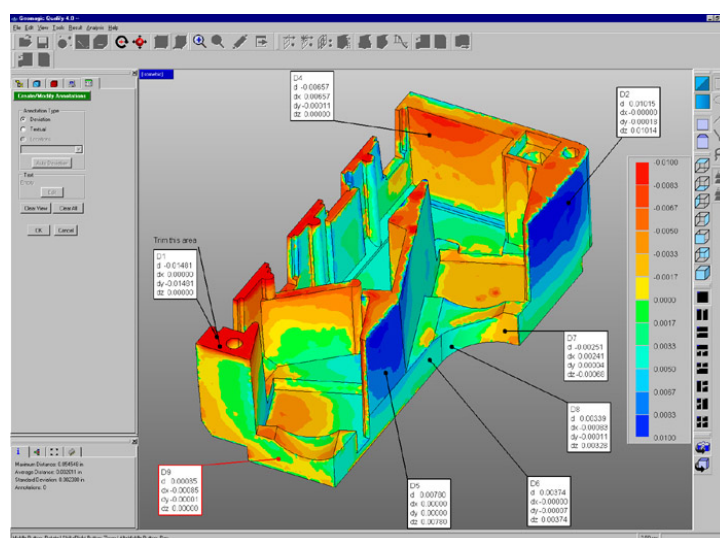


FIG. 1.10 – Vue de l'interface du logiciel commercial *Geomagic Qualify* (Raindrop).

Dans le domaine de la géophysique et des activités BTP, les données sont principalement des modèles numériques de terrain (voir section D.2.2) ou des modèles 3D simples (tunnels, routes, etc.). Ces logiciels ne sont pas forcément axés sur le calcul de différences et peuvent être très techniques (ils s'adressent généralement à des topographes et autres spécialistes de la mesure). Exemple de logiciels (commerciaux) : *I-Site Studio* de I-Site, *Realworks* de Trimble/Mensi, et les extensions *3D Analyst* et *Survey Analyst* du S.I.G. *ArcGIS* d'ESRI

Enfin dans le domaine académique, deux logiciels abordent en partie le problème de la comparaison géométrique en 3D. Ils ont été conçus à la base pour évaluer la qualité d'algorithmes de simplification de maillages. Il est probable que leur principe soit d'ailleurs repris dans nombre de logiciels commerciaux. Ces logiciels sont dénommés *Metro*[Cignoni et al. 1998] et *Mesh*[Aspert et al. 2002]. Leur principe est assez simple : des points sont échantillonnés sur un des deux modèles (pour se ramener ainsi au cas *standard*), puis le nuage est comparé à l'autre modèle (par calcul de la distance de chaque point au modèle). Enfin l'information de distance est rétro-propagée sur le premier modèle par interpolation des distances ponctuelles sur chaque facette. L'algorithme est détaillé dans la section 2.2.

On peut remarquer que tous ces logiciels ne font pas à proprement parler de *détection de changement*, mais plutôt de la *mesure d'écart*, qui est néanmoins l'étape préliminaire et obligatoire à tout processus de détection de changement.

## 1.5 Techniques d'acquisition de données 3D

Dans cette section sont présentées différentes techniques d'acquisition de données tridimensionnelles réelles. Elles sont classées en fonction de la technologie utilisée. Ce classement est quelque peu arbitraire car toutes ces techniques produisent des données finalement plus ou moins équivalentes (nuage de points, carte de profondeur, etc.). On peut par contre les classer selon la taille de la scène observée (voir figure 1.11, en reprenant l'exemple de classification non spatiale présenté en section 1.2.4). Pour ce qui nous préoccupe, à savoir le traitement des données, ce seront plutôt en pratique des caractéristiques annexes qui vont les différencier (densité des mesures, précision, erreur, etc.).

### 1.5.1 Laser

On ne reviendra pas ici sur le principe même du laser. On rappelle juste que le laser émet un faisceau de lumière cohérente par émission stimulée [Siegman 1971]. Il offre donc de très bonnes caractéristiques pour la mesure, car la diffraction du faisceau est extrêmement faible et sa longueur d'onde est unique.

---

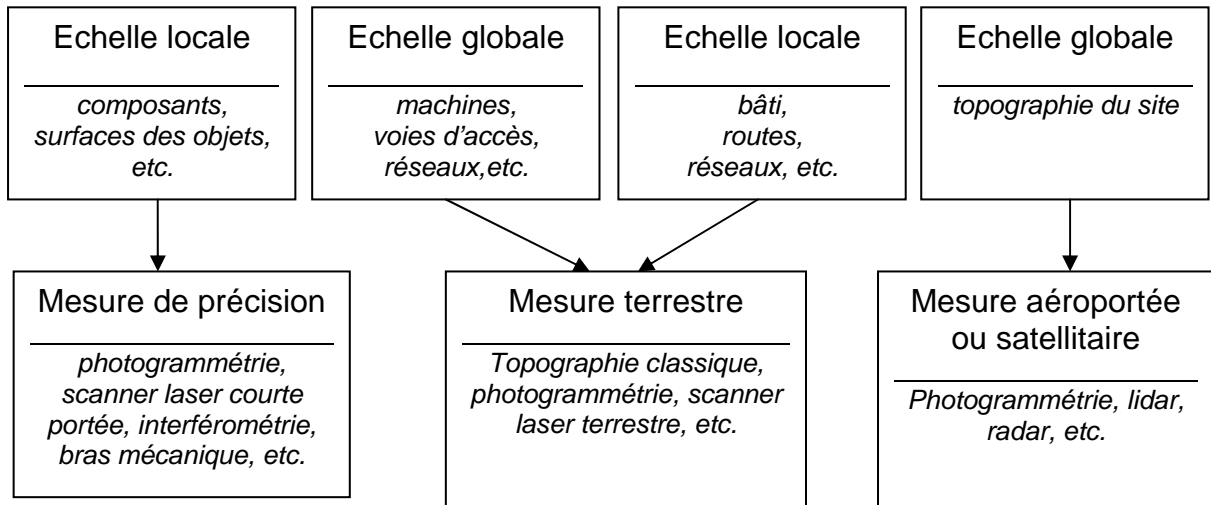


FIG. 1.11 – Dépendance entre la granularité de la scène observée et systèmes de mesure.  
(voir figure 1.9)

### Principe du scanner laser

Le nom générique des capteurs laser est LIDAR (pour *LIght Détection And Ranging*). Il en existe trois sortes :

- Range Finder (mesure de distance)
- Differential Absorption Lidar (mesure de concentrations chimiques)
- Doppler Lidar (mesure de vitesse)

Nous ne nous intéressons ici qu'aux *range finders*, et parmi ceux-ci aux capteurs qui permettent l'acquisition automatique de plusieurs mesures ponctuelles 3D (contrairement aux simples distance-mètres). On parle de *scanners* pour ces capteurs laser permettant de balayer une cible. Ces scanners sont principalement utilisés dans le milieu de la topographie (construction, maintenance), de l'industrie et de la culture (archéologie, conservation du patrimoine, etc.). Il y a deux grands principes de fonctionnement : la mesure du temps de vol et la triangulation plane.

### Mesure du temps de vol

Le premier principe est celui de la *mesure du temps de vol* (voir figure 1.12). Le scanner émet une impulsion lumineuse très courte (un tir) grâce à un laser, simultanément au déclenchement d'une horloge. Cette impulsion lumineuse est renvoyée partiellement par le premier obstacle qu'elle rencontre (par rétrodiffusion). Le signal de réception arrête l'horloge (ou du moins déclenche une mesure du temps d'arrivée dans le cas des scanners dits à *échos multiples*). La distance entre le capteur et la *cible* est alors directement proportionnelle au temps mis par l'impulsion lumineuse pour faire l'aller-retour.

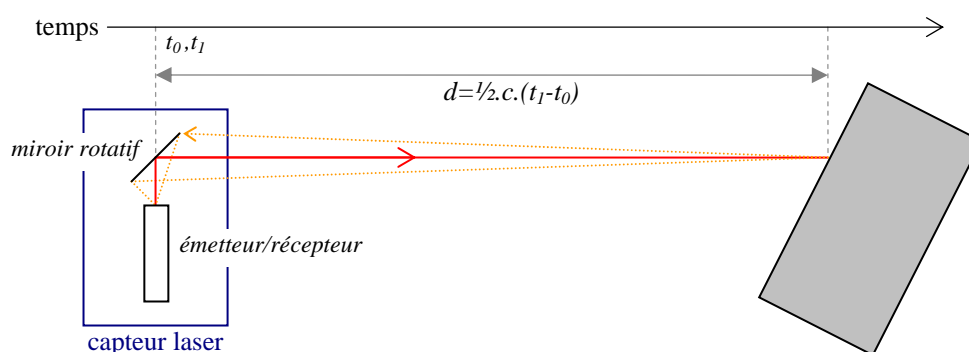


FIG. 1.12 – Principe du scanner laser à *mesure du temps de vol*.  
 $d$  est la distance à mesurer,  $t_0$  et  $t_1$  sont respectivement les instants d'émission de l'impulsion laser et de sa réception par le capteur.

Le faisceau lumineux est généralement dévié à la source par un miroir rotatif qui lui permet de balayer rapidement une portion de l'espace plus ou moins large et suivant une ou deux dimensions. Une exception notoire est le scanner aéroporté développé par la société allemande *TopoSys* qui utilise un ensemble de fibres optiques co-planaires (plus d'une centaine) plutôt qu'un balayage par miroir.

### Triangulation plane

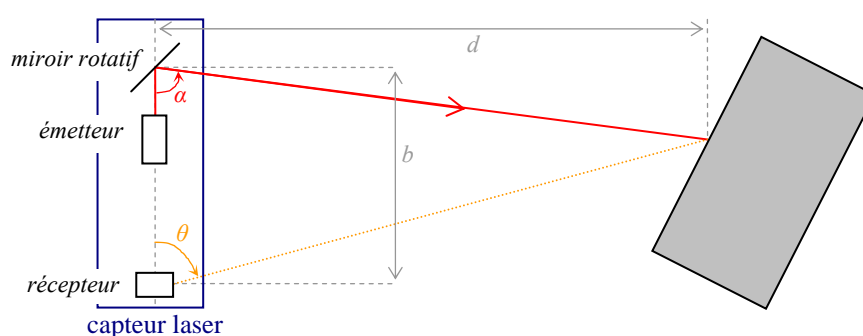


FIG. 1.13 – Principe du capteur laser à mesure par *triangulation plane*.

Le second type de scanner laser se base sur le principe de la *triangulation plane* (voir figure 1.13). Le scanner émet un rayon laser selon une direction  $\alpha$  précisément déterminée. Un récepteur CCD<sup>5</sup>, sensible à la longueur d'onde du laser et décalé par rapport à l'émetteur d'une distance connue ( $b$ ), voit la tache laser correspondante et donne une estimation de son orientation  $\theta$  selon cet autre point de vue. La distance entre l'émetteur laser et le

<sup>5</sup>Charge Coupled Device

récepteur CCD étant connue, la position de la tache laser (relativement au capteur) est alors déduite par une relation géométrique simple :

$$d = b * \left( \frac{1}{\tan(\alpha)} + \frac{1}{\tan(\theta)} \right)^{-1} \quad (1.1)$$

L'opération est répétée selon différentes directions, là encore par rotation d'un miroir en général. Ces scanners ont des portées beaucoup plus courtes mais une meilleure précision que les scanners à *mesure du temps de vol*.

### Intérêts des scanners laser

Les intérêts des scanners laser sont principalement :

- la rapidité du laser, qui permet la mesure de millions de points par minute,
- la possibilité de faire des mesures de nuit (grâce à l'éclairage *actif* du rayon laser) et aussi par relatif mauvais temps,
- la production en sortie du scanner d'un nuage de points 3D directement exploitable (cela est surtout vrai pour les scanners terrestres, les données aéroportées ayant besoin d'être corrigées après l'acquisition en fonction des paramètres d'attitude<sup>6</sup> du vecteur qui sont enregistrés lors du vol).

Il existe aussi des projets de recherche qui tentent d'exploiter des caractéristiques secondaires de l'écho laser, comme son amplitude (qui donne une quantification du coefficient de réflectivité de la surface de la cible touchée), ou encore sa forme (qui pourrait permettre de caractériser la micro-géométrie du matériau touché) : on peut citer [Wagner et al. 2004], mais surtout [Jutzi et Stilla 2004].

### Remarques sur la précision des scanners laser

Si la précision atteinte par les scanners laser est, en pratique, excellente, cette précision est très difficile à évaluer quantitativement à cause des très nombreux paramètres qui entrent en ligne de compte [Boehler et al. 2003].

Dans le cas des scanners laser à *mesure du temps de vol*, on peut au moins affirmer dans une première approche simpliste, qu'elle est limitée par la résolution en distance. Soit  $d$  la distance parcourue par le faisceau et  $c$  la vitesse de la lumière. On a de manière très simple :

$$d = c \frac{t}{2} \quad (1.2)$$

Donc dans un intervalle de temps  $\Delta t$ , on a :

$$\Delta d = c \frac{\Delta t}{2} \quad (1.3)$$

---

<sup>6</sup>Roulis, tangage et lacet.

Or, il existe un intervalle de temps minimum mesurable par le système, soit  $\Delta t_{min}$  ce temps. La distance minimum mesurable par le système, qu'on appellera résolution en distance, est alors égale à :

$$R_d = \Delta d_{min} = c \frac{\Delta t_{min}}{2} \quad (1.4)$$

(avec  $\Delta t_{min} = 1$  ns par exemple, on aurait en théorie  $R_d = 1.5$  cm)

Dans le cas des scanners à *triangulation laser plane*, les limitations sont principalement dues aux erreurs de détermination de l'orientation du faisceau laser lors de l'émission et de celle de la tache laser vue par le récepteur CCD (la précision du scanner laser est donc grandement liée à la résolution de ce dernier).

De plus, dans tous les cas, puisque le rayon subit toujours une légère diffraction, la tache laser sur la cible est loin d'être ponctuelle. On pourra donc difficilement mesurer des détails plus petits que cette tache ; sa taille et donc la précision du scanner dépendent ainsi directement de la distance de la cible au capteur.

Ces limitations sont des limitations du système de mesure lui-même. S'y ajoutent ensuite l'imprécision de la détermination du positionnement et de l'orientation du scanner, l'influence des conditions atmosphériques, le comportement thermique du laser, et un grand nombre d'autres facteurs qui ont tous des contributions individuelles infimes à l'erreur globale mais dont la somme peut ne plus être négligeable [Baltsavias-1 1999].

## LIDAR terrestre

Les scanners laser terrestres sont raisonnablement petits, posés sur un trépied ou même sur un véhicule [Zhao et Shibasaki 2002]. Ils balayent l'espace devant eux par rotation autour d'un ou deux axes grâce à un balayage motorisé ou des miroirs rotatifs. Ils ont des portées assez courtes par rapport aux capteurs aéroportés. Aujourd'hui, certains modèles atteignent tout de même plusieurs centaines de mètres (Mensi, Optech, Riegl, Z&F, etc.). Ils affichent des précisions de l'ordre du centimètre, voire inférieures, et des vitesses d'acquisition très grandes [Ullrich et al. 2002].

Effectués par des géomètres ou topographes sur le terrain, ces relevés sont beaucoup plus simples et rapides à réaliser que des relevés classiques (avec un théodolite typiquement). Leur précision, bien qu'inférieure, reste très bonne malgré une certaine variabilité des résultats en fonction de la surface balayée, des conditions d'illumination, d'humidité, etc. Étant données les hautes fréquences de balayage des scanners, on obtient très rapidement des nuages constitués de plusieurs millions de points. Ceci est à la fois un avantage puisque on obtient une représentation détaillée des objets scannés, mais aussi un problème potentiel pour le traitement et le stockage des données.

---



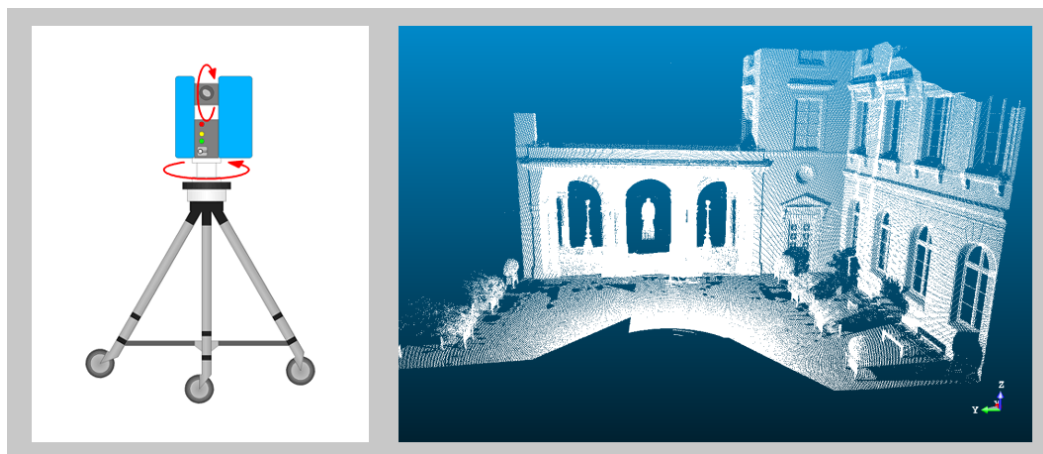


FIG. 1.14 – Scanner laser terrestre.

*Gauche : schéma de principe du balayage d'un scanner laser terrestre 3D (ici un modèle Zoller+Fröhlich). Droite : nuage de points acquis avec un scanner Trimble GS100 dans une cour intérieure du Collège de France (Paris).*

Leurs principaux défauts proviennent du fait que le point de vue du scanner terrestre est fixe lors d'une acquisition. Les zones d'ombres sont nombreuses, et l'acquisition d'une scène complète demande plusieurs relevés depuis différents points de vue et tout le travail de consolidation qui s'en suit. Cette opération, qui consiste à recalibrer toutes les données dans un même repère, est nécessaire puisque les coordonnées des points d'un nuage laser sont exprimées par rapport au capteur qui a été déplacé entre chaque prise de vue.

La précision des systèmes GPS<sup>7</sup> est insuffisante pour géo-localiser correctement les capteurs et donc les données acquises (surtout en comparaison avec la précision des scanners laser). Si l'on veut utiliser ces données dans un contexte géographique, il faut donc préalablement placer dans le champ du scanner des cibles de référence dont la position est déterminée précisément par des moyens classiques.

## LIDAR aéroporté

Les capteurs laser aéroportés sont fixés dans un avion (ou tout autre vecteur aérien équivalent comme un hélicoptère, un drone, etc.) de manière à pointer verticalement vers le sol. Ils sont associés à des systèmes GPS (embarqués dans l'avion mais aussi potentiellement situés au sol<sup>8</sup>) auxquels on adjoint des accéléromètres et des centrales inertielles pour mesurer l'attitude<sup>9</sup> de l'avion et améliorer la précision initiale du GPS, à

<sup>7</sup>Global Positioning System

<sup>8</sup>Via le dGPS pour *differential* GPS (relais au sol qui améliorent significativement la précision du réseau satellitaire).

<sup>9</sup>Roulis, tangage, etc.

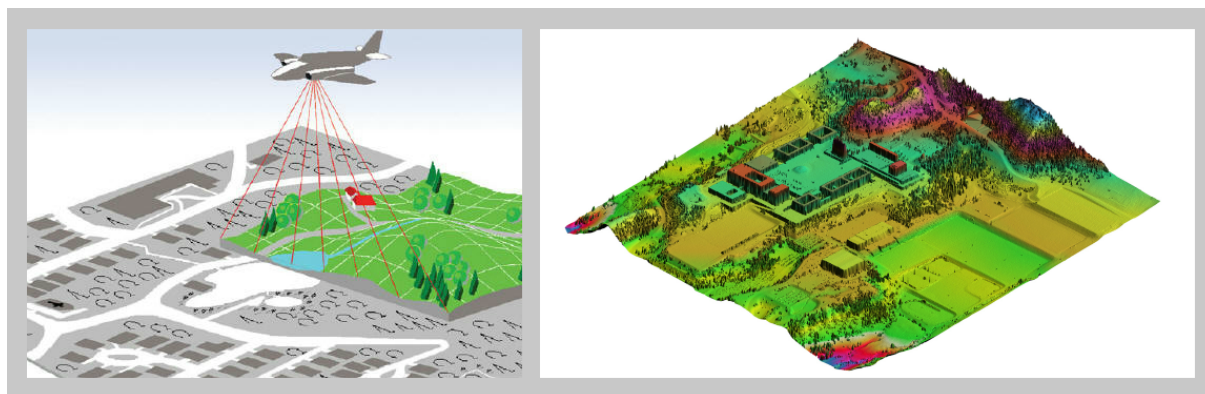


FIG. 1.15 – Principe du relevé laser aéroporté (à gauche) et vue d'un modèle de terrain obtenu avec cette technique (à droite).

(source des illustrations : <http://www.merrick.com>)

la fois spatiale (une dizaine de centimètres) et temporelle (la fréquence de rafraîchissement est au mieux d'une dizaine de hertz).

Le système laser balaye continûment avec une fréquence très élevée une plage d'angle fixe de quelques dizaines de degrés de part et d'autre de la verticale. Ce balayage se fait généralement dans un plan orthogonal à la direction de déplacement. Certains systèmes moins courants décrivent des traces au sol circulaires, voire plus complexes. La fréquence d'acquisition est très rapide. On obtient des bandeaux<sup>10</sup> de points 3D beaucoup plus longs que larges. La largeur dépend en effet de l'angle de balayage et de l'altitude de vol qui est en général comprise entre 1 et 2 km (au plus 6 km, mais la précision évolue de manière inversement proportionnelle à l'altitude) alors que la longueur est égale à celle du trajet du vecteur pendant la phase d'acquisition. Pour acquérir une zone aussi longue que large, le vecteur doit donc décrire des trajectoires de va-et-vient établies à l'avance (trajectoires dites *en hippodrome*). Le but est d'assurer un recouvrement optimal entre chaque bande pour ne pas manquer d'acquérir des zones de terrain tout en faisant le minimum d'allers-retours (Cf. [Baltsavias-1 1999] pour plus d'informations sur les relevés laser aéroportés).

La précision de ces systèmes est assez variable, et il est difficile de la connaître réellement sans faire des tests avec les matériels de chacun des fournisseurs [Baltsavias-2 1999], [Schenck 2001]. En général, même si cela dépend beaucoup de la hauteur de vol, on obtient par avion des données avec 1 à 4 points par m<sup>2</sup> et des précisions altimétriques de l'ordre de quelques décimètres.

Enfin, certains systèmes dits *multi-échos* peuvent, comme leur nom l'indique, enregistrer plusieurs échos successifs provenant d'une même émission. Cela permet d'obtenir directement les altitudes de différentes couches du *sursol* (bâtiments, végétation, etc.),

---

<sup>10</sup>Ces bandeaux (*strips* en anglais) sont beaucoup plus denses selon la direction orthogonale au sens de déplacement du vecteur (donc selon leur largeur).

dont en particulier les bords des toits des bâtiments et le feuillage des arbres (le laser a en effet la capacité de passer à travers car celui-ci n'est jamais parfaitement "étanche". On s'en rend bien compte lorsque l'on marche en forêt : on voit le soleil à travers de nombreux interstices dans le feuillage des arbres). Le spot laser a de plus une certaine surface lorsqu'il touche les objets au sol, et ce particulièrement en mode *aéroporté*. Il arrive donc que la tache du faisceau éclaire partiellement un bord de toit (ou des feuilles) et que le reste de l'énergie lumineuse continue son chemin jusqu'au sol. Si les deux taches (au sol et sur le premier obstacle) sont suffisamment grandes, alors les deux peuvent renvoyer des échos distincts qui seront tous les deux mesurés par le capteur.

### 1.5.2 Photogrammétrie

La photogrammétrie est une technique qui permet d'exécuter des mesures spatiales à partir de couples de photos prises selon des points de vue légèrement décalés. Elle se base sur le principe de la vision *stéréoscopique*, à l'instar de la vision humaine. Cette technique étudie également la création même de l'image et sa correction géométrique (notamment pour le calcul d'orthophotos<sup>11</sup>). Elle permet entre autre d'obtenir directement des Modèles Numériques de Surface (M.N.S. - Cf. annexe D.2.2) très fiables et sur mesure, mais dont la définition est tout de même limitée par la résolution des images.

#### Principe de la photogrammétrie

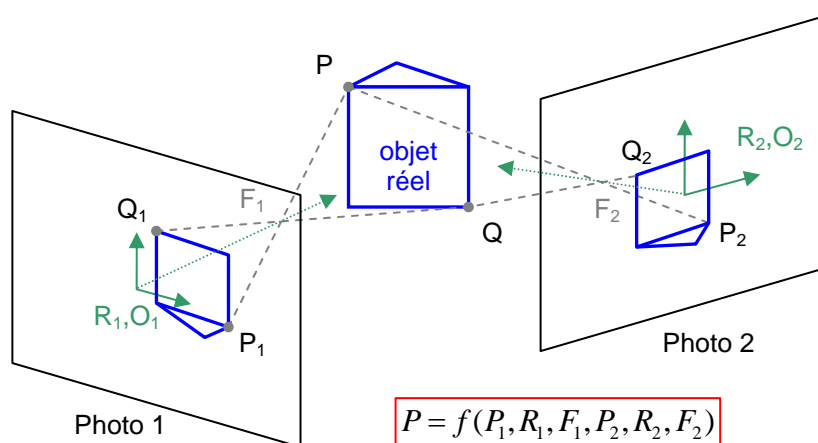


FIG. 1.16 – Principe de la photogrammétrie.

La stéréoscopie (ou perception de la troisième dimension) repose sur l'observation binoculaire. La vue d'un même objet sous deux angles différents, comme le font les deux

<sup>11</sup>Une orthophoto est une image photographique dont tous les pixels ont été recalculés de manière à être vus verticalement.

yeux, permet d'apprécier la profondeur (voir figure 1.16). La parallaxe que l'on perçoit entre les points homologues ( $(P_1, P_2)$  ou  $(Q_1, Q_2)$ ) d'un couple stéréoscopique (*Photo 1 & 2*) permet d'estimer des dénivelés et donc de restituer le relief des objets ; c'est le but de la photogrammétrie qui est utilisée pour l'établissement de cartes topographiques, en microscopie optique et électronique, pour des études de morphologie par photo-interprétation, ou encore pour de la reconstruction 3D d'objets ou de bâtiments. La connaissance précise des paramètres de prise de vue (position et orientation :  $(O_1, R_1)$  et  $(O_2, R_2)$ ) ainsi que des caractéristiques de l'appareil optique (qui est généralement calibré :  $F_1$  et  $F_2$  ainsi qu'un modèle de déformation optique) permet de retrouver les coordonnées 3D des points *imaginés*.

Note : un paramètre majeur pour la prise de vue est le rapport  $B/H$ . Si B est la *base* (l'écartement spatial entre les deux prises de vue), et H la distance de prise de vue, alors le rapport  $B/H$  donne une idée de l'écartement relatif entre les deux prises de vue, et donc de l'incidence visuelle des photos : plus les photos sont écartées, plus il sera facile d'évaluer et mesurer précisément la différence de position d'un même point entre les deux photos (et donc d'en déduire sa hauteur par des calculs géométriques simples, pour ce qui est de la théorie). Par contre, plus les photos sont écartées, et plus les parties cachées sont importantes. Il faut donc trouver un bon compromis entre ces deux aspects du problème.

## Photogrammétrie manuelle

La résolution de ce problème géométrique est mathématique mais nécessite un appariement préalable des couples de points. Cette opération, appelée *stéréo-restitution*, est généralement manuelle si l'on veut obtenir des résultats précis, comme pour la génération des cartes de l'IGN<sup>12</sup> à partir de clichés aériens. Dans ce cas, on utilise des machines sophistiquées appelées *stéréorestituteurs* (qui peuvent être analogiques, analytiques ou numériques). Il est possible d'utiliser l'outil informatique standard (avec des logiciels plus ou moins évolués qui couvrent une très large gamme de prix) mais les résultats sont moins précis et le processus encore moins rapide. Or la rapidité est de loin le plus gros problème de la photogrammétrie, puisqu'il faut plusieurs heures de travail pour traiter un couple de photos. Le résultat est par contre un des plus précis et des plus évolués possible, puisque l'opérateur peut créer le modèle 3D en même temps qu'il mesure les points (à condition que la scène modélisée ne soit pas trop complexe). Ce processus nécessite donc l'intervention d'un professionnel doté d'un important savoir-faire. Ceci est d'ailleurs vrai à tous les niveaux de la chaîne de production, que ce soit lors de la préparation de la campagne de prises de vue ou lors de la génération du plan (ou de la carte dans le cas de l'IGN). Le processus global est donc très précis mais aussi très lent et nécessite d'importants moyens humains et financiers.

---

<sup>12</sup>Institut Géographique National

---

## Photogrammétrie automatique

La communauté scientifique du domaine tente depuis de nombreuses années de remplacer le processus humain de mise en correspondance de points entre les deux prises de vue par un processus automatique. Les différentes stratégies reposent plus au moins sur le même principe : rechercher dans des zones théoriquement équivalentes les points les plus semblables. Cela est facilité si l'on connaît les paramètres de prise de vue de chaque photo. On peut ainsi déduire des lignes communes aux deux photos appelées *lignes épipolaires* (en relation avec la géométrie *épipolaire*<sup>13</sup> du couple de photos). La méthode d'appariement la plus courante est la corrélation (en considérant la teinte, le voisinage des points, etc.) mais d'autres méthodes moins classiques sont aussi utilisées (*Tensor Voting* [Chi-Keung et al. 2001], etc.). On obtient ainsi rapidement de nombreux points 3D uniformément répartis sur la scène. Par contre leur précision est directement liée à la résolution des prises de vue et les données ont généralement un aspect bruité très caractéristique. Ceci est acceptable pour des M.N.S. de faible résolution (à partir de couples d'images satellites par exemple, pour générer rapidement et sur de très grandes étendues des modèles de terrain approximatifs - avec des images *Spot 5*<sup>14</sup> par exemple, on peut typiquement obtenir des M.N.S. de 10 m de résolution avec des précisions de l'ordre de 3 m). En milieu urbain ou pour des objets isolés, la précision et la stabilité des processus automatiques est par contre encore faible. Elle profite cependant d'une forte dynamique de recherche (Cf. [Roux et Maître 2001] par exemple).

### 1.5.3 Radar

Le système RADAR (Radio Detection And Ranging) est un excellent outil de télédétection. Bien que longtemps considéré comme complexe et confidentiel (applications militaires obligent), il est aujourd'hui plus accessible et est utilisé dans de nombreux domaines (de l'aviation à la cartographie en passant par la géologie, la prospection maritime, etc.).

#### Principe de la mesure radar

Là encore nous n'entrerons pas dans les détails de la théorie. Son principe, tel qu'énoncé par l'américain Hugo Gernsback en 1891 est : « *une onde électromagnétique émise par une source se réfléchit sur des cibles, et l'analyse du signal reçu permet de détecter et de localiser ces cibles en supposant que la vitesse de propagation des ondes demeure à peu près constante* ». En pratique, une antenne radar utilise donc le même principe de

---

<sup>13</sup>En traitement d'images, géométrie particulière caractérisant les liens entre deux images stéréoscopiques. L'homologue d'un point d'une des deux images se trouve sur une droite connue dans l'autre image.

<sup>14</sup>Satellite optique (résolution des images : 2,5 m par pixel en monochrome).

---

*mesure du temps de vol* que les capteurs lasers (Cf. section 1.5.1), mais elle utilise une onde électro-magnétique très haute fréquence et non un rayon laser. Cela lui confère des propriétés très intéressantes (indépendance par rapport aux conditions climatiques et lumineuses, pouvoir de pénétration dans certains matériaux en fonction de la longueur d'onde, dépolarisation du signal donnant des informations sur le milieu traversé, etc.).

Une image radar représente l'amplitude et la phase du signal retour en fonction de l'angle d'incidence et de la position du capteur le long de sa trajectoire (orbite dans le cas d'un satellite). Son interprétation est donc difficile et les propriétés géométriques standard n'y sont pas du tout respectées. Il est tout de même possible de tirer une information 3D de ces données. On peut soit utiliser la *radargrammétrie* (une technique équivalente à la photogrammétrie mais appliquée à des images radar d'amplitude), ou bien utiliser le principe de l'interférométrie (appliquée à des images radar de phase), ou encore, moins couramment, la *radarclinométrie* [Paquerault 1998], qui consiste à déduire la hauteur des bâtiments grâce aux ombres qu'ils projettent (selon le principe plus général de "*Shape From Shading*").

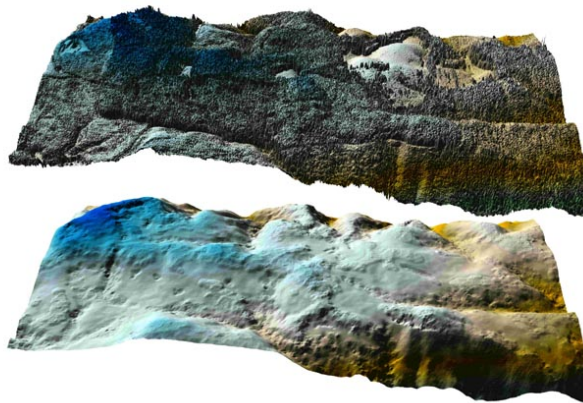


FIG. 1.17 – Exemple de M.N.S. (en haut) et M.N.T. (en bas) obtenus par interférométrie radar aéroportée (sur une zone boisée au Canada).

### Interférométrie radar

La technique d'interférométrie radar (appelée InSAR ou IfSAR) consiste à effectuer la différence entre deux images radar de phase [Rosen et Hensley 2000]. L'hypothèse fondamentale est que la contribution interne liée aux propriétés physiques, géométriques et diélectriques des cibles demeure stable entre les deux prises de vue. Si cette hypothèse de *cohérence* est vérifiée, alors l'interférogramme (la différence entre les deux images de phase) révèle des systèmes de franges liés à la topographie ou à la déformation de la surface. La phase du signal équivaut en effet à deux fois la distance du trajet entre le capteur et la sur-

face imagée, modulo  $2\pi$ . Cela permet donc de mesurer la différence de distance-temps avec une précision d'une fraction de la longueur d'onde (précision de quelques millimètres en pratique). La différence de distance-temps peut être reliée géométriquement à la hauteur des terrains observés.

Il est donc possible de mesurer la topographie d'un terrain par interférométrie radar [Crosetto 2002] : deux images sont acquises depuis deux points de vue différents au même instant. On utilise typiquement deux antennes placées sur la même plate-forme avec un décalage connu et qui font une acquisition simultanée de la même région. La différence de phase relative permet alors de construire un M.N.S. (la figure 1.17 représente le modèle d'une zone boisée acquis par cette technique).

Mais on peut aussi mesurer des déplacements par la technique d'interférométrie dite *différentielle*. Il existe deux méthodes :

- en utilisant trois images radar, on peut créer deux interférogrammes. La différence de ces deux jeux de données correspond alors à un interférogramme *différentiel*. Celui-ci cartographie les modifications verticales de la zone observée, là encore modulo  $2\pi$  (voir figure 1.18). On peut remplacer un des deux interférogrammes par un M.N.T., ce qui permet de corriger les effets géométriques du relief sur l'autre interférogramme.
- il est aussi possible d'utiliser directement deux images radar acquises par le même capteur depuis le même point de vue à deux instants différents [Preiss et al. 2003]. Le résultat est là encore une différence relative de la topographie entre les deux dates mais selon la direction d'acquisition. Lors des deux passages, la géométrie d'acquisition doit être strictement contrôlée afin d'éviter toute influence de la topographie sur la mesure et de permettre le calcul de la différence de phase. Cette situation exige que le capteur répète exactement la même trajectoire que lors du premier passage.

Il existe des systèmes InSar aéroportés : *STAR-3i* d'Intermap, système nord-américain issu d'un transfert technologique de l'armée américaine vers le privé, premier système radar aéroporté *commercial*, ainsi qu'une dizaine de prototypes de recherche dans le monde, dont en particulier *RAMSES* de l'ONERA<sup>15</sup> et *E-SAR* du DLR<sup>16</sup> qui sont respectivement français et allemand. Ces systèmes affichent aujourd'hui des précisions proches de celle du lidar aéroporté (entre 30 cm et 3 m.) et sont plus rapides car ils acquièrent des bandes de terrain plus larges. Il semble par contre qu'il y ait de sérieux problèmes d'incohérence spatiale au niveau des reliefs importants et des zones urbaines, auxquels il faut ajouter un problème de réflexions multiples de l'écho radar sur les bâtiments trop rapprochés. De plus, la qualité des données (qui est nécessaire pour assurer la cohérence des images différenciées) est assez difficile à maintenir d'un vol à l'autre. La technique n'a pas encore profité du même élan que celui qu'a connu le laser au cours de ces dernières années. Faute d'un meilleur développement, elle est rarement utilisée en dehors du continent nord-américain. Néanmoins, en dehors des zones urbaines on obtient beaucoup plus rapidement

---

<sup>15</sup>Office National d'Etudes et de Recherches Aérospatiales.

<sup>16</sup>Deutsches Zentrum für Luft und Raumfahrt.

---

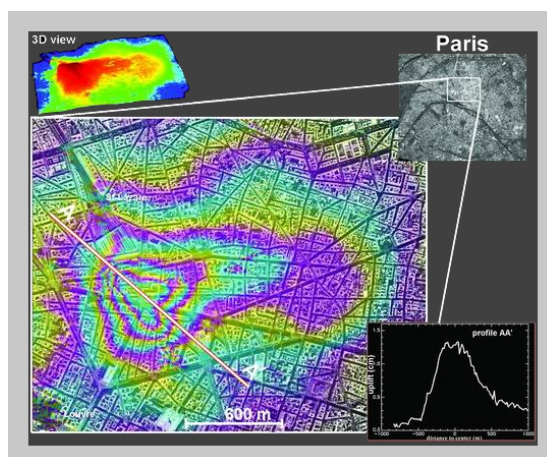


FIG. 1.18 – Mesure de la subsidence (affaissement progressif, enfoncement) du quartier de la gare Saint-Lazare par interférométrie satellitaire entre 1997 et 1999, suite à des travaux souterrains.

(données ERS/ESA, distribuées par Eurimage, et traités par le BRGM).

Les franges concentriques correspondent à des déplacements verticaux de 3 mm  
(déplacement maximal : 1,3 cm).

par cette technique des M.N.T. quasi-équivalents à ceux qu'on obtiendrait par relevé laser aéroporté [Gamba et Houshmand 2000].

Il est enfin important de noter que l'interférométrie radar temporelle est avant tout une technique de détection de changement géométrique à part entière, plutôt grande échelle. Dans un contexte satellitaire (le seul qui marche correctement actuellement : la condition de reproductibilité de la trajectoire est beaucoup plus simple à respecter pour un satellite que pour un avion) elle permet le contrôle des déplacements de grandes structures avec une précision millimétrique. Ses applications phares sont l'observation de volcans, failles, glaciers, larges portions de terrains, zones touchées par un séisme, etc.

#### 1.5.4 Mesure par contact

Bien que le terme soit plus général, nous désignons ici par *mesure par contact* (C.M.M., *Coordinate Measuring Machine* en anglais), les systèmes mécaniques ou optiques de mesure 3D qui nécessitent un pointage direct et physique du point mesuré.

Ce sont typiquement les bras de mesure mécaniques et assimilés. Ce sont des appareils reliés à une base fixe qui suivent et enregistrent les mouvements de translation et de rotation de leurs différents composants. Les composants sont assemblés, tous parfaitement calibrés, sous forme d'une chaîne. Chacun possède un certain nombre de degrés de liberté. La chaîne d'éléments est terminée par un pointeur qu'on place sur le point à mesurer



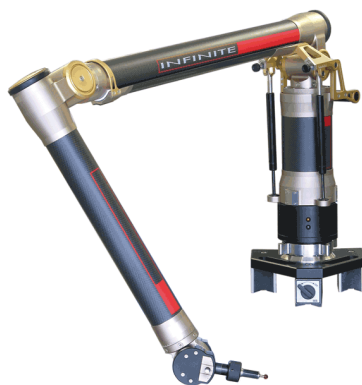


FIG. 1.19 – Système de mesure par contact.  
(bras articulé, modèle INFINITE Cimcore, d'*Hexagon Metrology*)

(manuellement ou automatiquement en fonction du modèle et de l'application). Le système intègre alors tous les déplacements pour déterminer la position exacte du point par rapport à sa base. On les retrouve surtout dans l'industrie, car leur coût, leur dimension et leur faible mobilité font qu'ils sont utilisés à poste, généralement pour une application spécifique (typiquement le contrôle structural de pièces manufacturées). Il existe des versions sans fil ni armature qui utilisent des champs magnétiques pour déterminer la position et l'orientation de l'outil de pointage [Frantz et al. 2003]. Mais quelle que soit la technologie, la portée de ces appareils est généralement courte et leur rendu est équivalent à celui d'un processus photogrammétrique manuel : des données ponctuelles 3D très précises mais peu denses et acquises lentement.

## 1.6 Discussion sur les techniques d'acquisition

Dans cette section, les techniques d'acquisition présentées précédemment sont rapidement confrontées les unes aux autres pour déterminer la plus apte à répondre aux principaux besoins énoncés en section 1.3. Bien que l'acquisition des données ne soit pas l'unique étape du processus de détection de changement, ni précisément l'objet de cette thèse, elle en est tout de même la base. Son choix est donc important pour la suite du processus.

Les critères de résolution et de précision vont dépendre des scènes observées (puisque'il faut alors des résolutions suffisantes pour distinguer les objets d'intérêt et une précision permettant de mesurer les déplacements que l'on souhaite détecter). On peut par contre établir une liste des autres critères importants qui vont dépendre plutôt de l'application envisagée. Nous avons regroupé ces différents critères sous la forme d'un tableau subjectif mais assez synthétique (1.1).

---

	laser		photo		radar	C.M.M.
	terrestre	aéroporté	terrestre	aéroporté	aéroporté	terrestre
(1)	rapide	lente	rapide	lente	lente	fixe
(2)	moyenne	grande	moyenne	grande	très grande	courte
(3)	int/ext	ext	int/ext	ext	ext	int
(4)	rapide	rapide	rapide	rapide	rapide	lente
(5)	cons	corr+cons	stereo	stereo	recalage	rien
(6)	rapide	moyen	lent	lent	moyen	très rapide
(7)	moyen	cher (+)	moyen	cher (-)	cher (-)	moyen

(*cons*=consolidation, *corr*=corrections, int/ext=intérieur/extérieur)

TAB. 1.1 – Comparaison des techniques d'acquisition 3D.

Les critères sont désignés par leur indice dans le tableau :

- (1) la vitesse de préparation du capteur,
- (2) la couverture et la portée du capteur,
- (3) son champ d'action (intérieur/extérieur en particulier),
- (4) le temps d'acquisition des données,
- (5) les opérations nécessaires avant de commencer à exploiter les données,
- (6) leur vitesse d'exécution,
- (7) et enfin le coût.

Pour les applications envisagées dans le domaine de la documentation T.Q.C., qui comme nous l'avons vu nécessitent d'opérer à l'intérieur et avec des portées moyennes, on voit clairement apparaître la prédominance du laser et de la photogrammétrie terrestre. Ce sont d'ailleurs les deux techniques couramment utilisées aujourd'hui. Les capteurs aéroportés sont trop imprécis et ils sont limités à des prises de vues extérieures avec un angle de vision fixe, ce qui est beaucoup trop limitatif pour ce genre d'application. De même, les bras de mesure (ou assimilés) étant généralement fixes et de courte portée, il n'était de toute façon pas envisageable de les utiliser dans ce cadre.

Pour les applications de suivi de chantier, le choix est un peu moins évident (mis à part les capteurs C.M.M. qui sont encore une fois totalement inutilisables pour ce type d'application). La contrainte de rapidité désigne sans conteste le laser terrestre, puisque celui-ci produit des données exploitables beaucoup plus rapidement que les autres méthodes (les données sont quasiment directement utilisables à la sortie du capteur). Sans cette contrainte, et surtout en fonction de la dimension et de l'accessibilité du chantier, les capteurs aéroportés présentent l'avantage majeur d'avoir des portées et des couvertures très importantes. Dans ce cas, le choix entre une campagne laser et une campagne photogrammétrique dépend de conditions plus spécifiques (a-t-on besoin d'un modèle 3D ou simplement d'un nuage de points dense ? Quelle est la précision nécessaire ? Les objets à mesurer sont-ils polygonaux et bien contrastés comme des bâtiments, ou relativement informes et irrégulièrement texturés comme de la terre ou des arbres ? Dispose-t-on de gros moyens financiers ? etc.).

Dans le cadre de cette étude, notre choix s'est porté vers les capteurs laser terrestres comme base de développement des méthodes et algorithmes de détection de changement géométriques en 3D (en essayant de limiter autant que possible la dépendance de ces méthodes vis à vis de la technologie laser). Le caractère innovant de cette technologie et la forte dynamique de recherche qui l'entoure ont pesé dans la balance puisque le développement rapide des capteurs laser allait vers des fréquences d'acquisition et des précisions toujours plus grandes. De plus, l'accès à un grand nombre de données, à plusieurs capteurs et à de nombreux retours d'expérience dans le domaine a bien sûr aussi contribué au choix de cette technologie. EDF avait en effet déjà une certaine expérience dans ce domaine après 10 ans d'utilisation intensive de capteurs laser dans ses centrales (justement pour des applications de T.Q.C.) et comptait accessoirement parmi ses filiales la société *Mensi* qui fabrique les capteurs *Soisic*, *GS100* et *GS200* (*Mensi* est aujourd'hui une filiale de la société *Trimble*). EDF est pionnière dans le domaine en France, notamment grâce à ses divisions de recherche et de topographie, et avait déjà conduit plusieurs thèses autour de cette technologie.

---

## Chapitre 2

# Calcul de distance sur un nuage de points

Nous présentons dans ce chapitre plusieurs algorithmes de calcul de distance : d'une part entre un nuage de points et un modèle 3D, et d'autre part entre deux nuages de points. L'idée est de calculer des écarts entre les différentes représentations tridimensionnelles (pour pouvoir par la suite détecter et si possible segmenter les zones de différence). Mais tout d'abord il convient de faire deux remarques importantes.

### 2.1 Remarques préliminaires

#### Le problème du recalage des données

Nous supposons que les données ont préalablement été recalées au mieux les unes par rapport aux autres. Si le protocole d'acquisition des données a prévu un positionnement et une détermination précise des différents points de vue du capteur (par levés topographiques de points fixes dans la scène par exemple), il devrait être possible d'exprimer simplement les données dans un repère unique (à la précision des instruments de mesure près). Si ce n'est pas le cas, on peut tenter d'utiliser des méthodes de recalage semi-automatiques voire automatiques (typiquement l'algorithme ICP<sup>1</sup>[Besl et McKay 1982] dans le cas des nuages de points). Mais il est important de noter que ce type d'algorithmes ne converge pas si les données sont trop différentes, les différences entre les deux jeux de données recalés étant ici assimilables à du bruit.

Toujours dans le cas des nuages de points, l'écart-type des écarts (réels) ne doit pas

---

<sup>1</sup>Iterative Closest Point

---

dépasser d'après Besl un seuil empirique de 10% de la taille de la scène pour que l'algorithme ICP converge. Nous avons donc modifié légèrement cet algorithme en ne prenant en compte à chaque itération que les points dont la distance à l'autre nuage est inférieure à un certain seuil. Ce seuil peut diminuer d'une itération à l'autre. Cela permet au processus itératif de converger d'une part, et surtout de ne pas être trop biaisé par la présence de points potentiellement très différents entre les deux nuages (qui sont des *outliers* pour l'optimisation par la méthode des moindres carrés). Les deux jeux de données doivent être par contre initialement au moins grossièrement recalés.

Il n'est malheureusement pas possible d'obtenir ainsi un recalage optimal dans le cas général, puisqu'il faudrait pour cela que le filtrage des points "différents" soit parfait. Or déterminer ces points est justement notre but et nous avons besoin pour cela de deux nuages bien recalés. C'est un problème imbriqué qui n'a donc pas de solution directe (une solution itérative serait envisageable mais certainement très lourde). Quelle que soit la méthode utilisée, l'erreur induite par le recalage doit être plusieurs fois inférieure à la plus petite déformation que l'on souhaite détecter, puisque cette erreur de positionnement se répercutera directement sur le résultat final.

## Distances globales et distances locales

Il existe de nombreuses manières de calculer une distance entre deux nuages de points. On peut les séparer en deux catégories : d'une part les distances *globales*, qui donnent une valeur unique entre deux ensembles, généralement utilisées pour des problèmes de classification selon des critères spécifiques (à l'instar des méthodes de mesures de formes appliquées à des modèles 3D et citées en section 1.4.1) et d'autre part les distances *locales*, qui donnent une information d'écart ou de déplacement par rapport à une autre forme et ce en chaque point du nuage (ou au moins pour différents sous-ensembles de points).

Une distance globale très courante lorsque l'on traite de nuages de points est la distance de Hausdorff<sup>2</sup>. Elle mesure de manière générique une distance entre deux ensembles. Dans le cas de nuages de points, on peut en donner une définition basée sur la distance euclidienne classique. Soient  $S$  et  $S'$  deux ensembles de points. On définit la distance entre un point  $p \in S$  et l'ensemble  $S'$  comme :

$$d(p, S') = \min_{p' \in S'} \| p - p' \|_2 \quad (2.1)$$

La distance de Hausdorff *directe* (non symétrique) est alors tout simplement :

$$D(S, S') = \max_{p \in S} d(p, S') \quad (2.2)$$

Il existe une distance de Hausdorff *symétrique* définie comme suit :

$$D_{\Sigma}(S, S') = \max(d(S, S'), d(S', S)) \quad (2.3)$$

---

<sup>2</sup>Felix Hausdorff (1868-1942), mathématicien allemand.

Il existe d'autres distances entre ensembles et d'autres méthodes de comparaison globale de données 3D comme le calcul de moments [Maas 1999] ou l'analyse en composantes principales. Mais toutes ces méthodes ne donnent aucune information sur les différences géométriques locales.

Nous nous intéresserons donc dans la suite de ce manuscrit, non pas aux distances globales entre deux ensembles mais plutôt à l'écart de chaque point du nuage (ou sous-ensemble de points) avec son homologue dans le nuage de référence. Cette information est par contre généralement plus longue à calculer puisqu'il faut prendre en compte la position relative des points.

Note : on exclut de cette dernière remarque le cas très particulier où il existe une relation d'équivalence connue entre des couples de points de chaque nuage, comme par exemple deux mesures successives de points 3D sur un même objet en des points particuliers ou préalablement marqués. Ce cas de figure est courant en photogrammétrie ou avec un processus manuel de pointage, mais il est très difficile, voire impossible à obtenir avec un capteur laser. Le calcul des écarts ne présente alors aucune difficulté particulière, puisqu'il suffit de calculer les distances séparant chaque couple de points.

## 2.2 Distances d'un nuage de points à un modèle 3D

Dans cette section on s'intéresse à un problème classique de comparaison de données 3D : le calcul de la distance d'un nuage de points à un modèle 3D. On va étudier en détail un algorithme performant et reconnu de comparaison d'un nuage de points à un maillage triangulaire, mis au point par P. Cignoni et al. et implémenté entre autres dans le logiciel *Metro* [Cignoni et al. 1998].

### 2.2.1 Distance point-triangle

La distance d'un point à un triangle en 3D est globalement celle du point au plan du triangle si sa projection sur celui-ci est à l'intérieur du triangle, et celle du point à l'arête la plus proche sinon.

Soit  $P$  un point 3D et  $(ABC)$  un triangle inclus dans un plan  $(\pi)$  quelconque. Soit  $H$  le projeté orthogonal de  $P$  sur  $(\pi)$ . Deux cas de figure :

- $H$  est à l'intérieur du triangle  $ABC$ , auquel cas  $d(P, ABC) = \|\overrightarrow{HP}\|$
- $H$  est à l'extérieur du triangle  $ABC$ , auquel cas on définit le point  $K$  comme étant le point appartenant à  $(ABC)$  le plus proche de  $H$ , et alors  $d(P, ABC) = \|\overrightarrow{KP}\|$ .

Si l'on dispose d'une information de normale associée au triangle  $(ABC)$ , il est alors

---

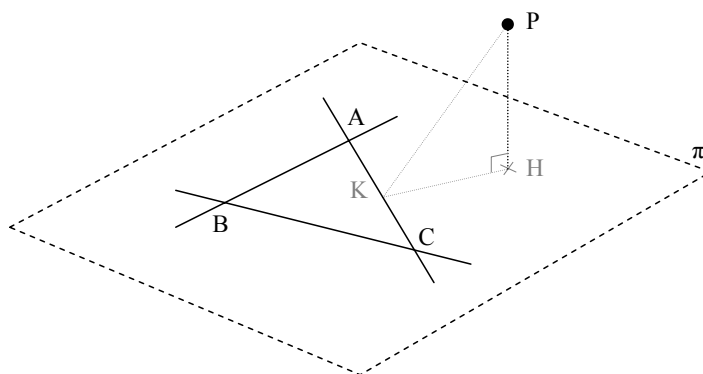


FIG. 2.1 – Distance d'un point (P) à un triangle (ABC).

possible d'affecter un signe à la distance calculée. La normale donne une information sur la notion d'intérieur et d'extérieur de la surface représentée localement par le triangle.

### 2.2.2 Intersection entre une grille 3D et un ensemble de polygones

Soit un nuage  $S$  composé de  $N_S$  points  $P_i$ ,  $0 \leq i < N_S$  et un maillage  $M$  composé de  $N_M$  triangles  $T_j$ ,  $0 \leq j < N_M$ . Soit enfin  $d(P_i, T_j)$  la distance d'un point  $P_i$  à un triangle  $T_j$ .

Un algorithme naïf pour calculer la distance de chaque point au maillage serait de calculer la distance de chaque point à tous les triangles du modèle et de garder la plus petite (Cf. algorithme 1). Au delà de quelques centaines d'éléments, un tel algorithme est totalement inefficace et serait un gouffre temporel.

---

**Algorithme 1** Distance des points d'un nuage  $S$  à un maillage triangulaire  $M$  (algorithme naïf)

---

```

pour  $0 \leq i < N_S$  faire
   $Dist_i \leftarrow d(P_i, T_0)$ 
  pour  $1 \leq j < N_M$  faire
     $Dist_i \leftarrow \min(Dist_i, d(P_i, T_j))$ 
  fin pour
fin pour

```

---

L'idée de Cignoni et al. est de minimiser le nombre de calculs de distances en ne comparant un point qu'aux triangles les plus proches. Pour cela, une grille 3D est utilisée pour évaluer grossièrement ces notions de voisinage et de proximité entre points et triangles. Une grille 3D est une subdivision d'une portion cubique de l'espace en zones elles-mêmes cubiques ayant toutes la même taille et juxtaposées selon les trois dimensions. Elle est

---

analogue à la structure *octree* (qui est une structure permettant la détermination des voisinages des points de manière efficace, détaillée dans le chapitre 4) considérée à un niveau de subdivision donné, mis à part que le nombre de cellules n'est pas forcément une puissance de 2.

Dans une première phase, cette grille 3D est *intersectée* avec les triangles : chaque triangle apporte une référence vers lui-même aux cellules qu'il intersecte (les cellules étant considérées comme des cubes dans l'espace). Cette opération peut être d'ailleurs grandement accélérée par une approche multi-échelles en utilisant justement une structure *octree* spatialement équivalente à la grille. A la fin du processus, les cellules possèdent un certain nombre de références vers des triangles dont la surface est au moins partiellement comprise dans la portion de l'espace qu'elles délimitent.

Dans une deuxième phase, on considère les points cellule par cellule. Si la cellule courante possède des références vers des triangles, on calcule les distances de chaque point inclus dans cette cellule avec l'ensemble des triangles référencés et on garde la distance minimale (en somme, on applique localement l'algorithme naïf). Si cette distance minimale est supérieure au rayon de la plus grande sphère centrée sur le point et incluse dans la cellule, ou si la cellule ne possède pas de référence, alors on recommence le même processus en incluant les cellules voisines de proche en proche, jusqu'à ce que la condition sur le rayon de la sphère englobée soit respectée (on retombera sur un schéma équivalent pour la recherche des plus proches voisins d'un point, en section 4.4).

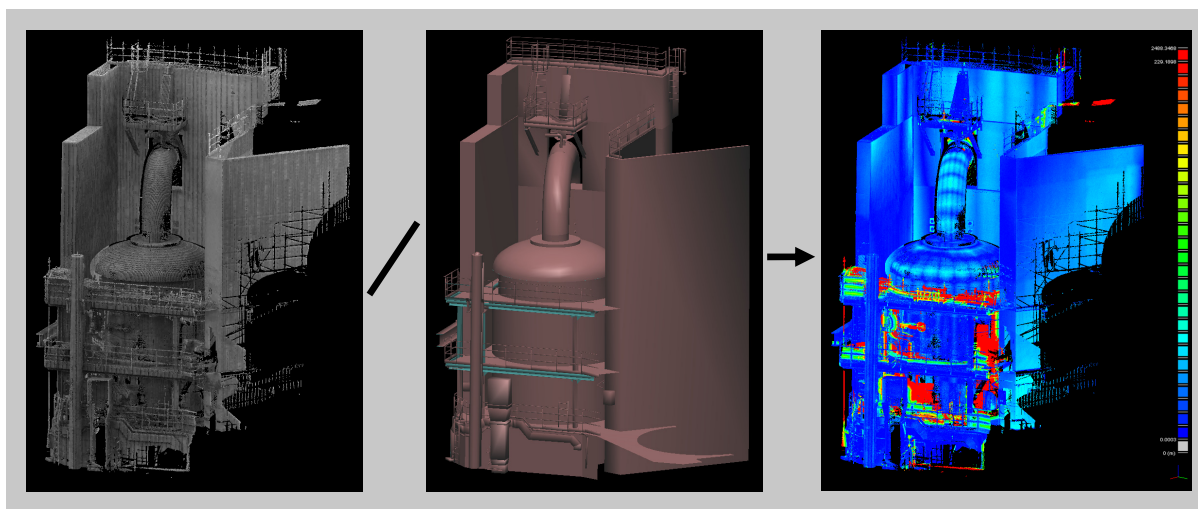


FIG. 2.2 – Calcul de distances entre un nuage de points et un modèle 3D triangulé. *A gauche : nuage avec éclairage synthétique de type "portion de ciel visible" (voir section A.3). Au milieu : modèle 3D. A droite : résultat du calcul de distance par un algorithme équivalent à "Metro". Temps de calcul indicatif : 4,9 s. avec 3 millions de points et 10 000 triangles. Les zones rouges correspondent aux distances les plus grandes.*



En pratique, cet algorithme est extrêmement rapide et permet, une fois implémenté proprement, de comparer des millions de points avec des centaines de milliers de triangles en quelques dizaines de secondes. La phase d'intersection de la grille 3D avec le maillage est en général beaucoup plus rapide que la seconde phase de recherche du plus proche triangle (qui est dépendante du rapport *pas de la grille/densité du nuage*, et aussi de la distance globale entre nuage et maillage).

Un exemple visuel de résultat de cet algorithme est représenté en figure 2.2. Cet algorithme est relativement simple à implémenter, et il a démontré d'excellentes performances une fois couplé avec la structure d'octree présentée au Chapitre 4 (Cf. section 2.4 pour des exemples de temps de calcul).

## 2.3 Distances entre deux nuages de points

L'idée principale de cette thèse est de comparer directement les nuages de points (laser en particulier). Le but est d'éviter de perdre du temps à modéliser le nuage de référence, sachant que cette opération est très coûteuse en temps et souvent hasardeuse lorsqu'elle est effectuée automatiquement sur des nuages de points volumineux et incomplets. Comparer deux nuages de points n'est par contre possible que si le nuage de référence est suffisamment dense et que les deux nuages ont des zones communes suffisamment grandes. C'est généralement le cas des données acquises par capteur laser, et c'est pourquoi nous nous focaliserons sur ces données par la suite (en particulier en section 2.3.2). Pour autant, certains algorithmes restent tout à fait généraux et peuvent être appliqués à tout type de nuage de points.

D'un point de vue théorique, comme nous l'avons vu au chapitre précédent (section 1.4.1), la comparaison directe de nuages de points 3D n'a été étudiée à notre connaissance que par Memoli et Sapiro [Memoli et Sapiro 2004]. Mais leur méthode ne s'applique qu'à des objets simples (une surface *manifold*), décrit par des nuages de points uniformément échantillonnés et complets. La comparaison est aussi et surtout globale. L'application est d'ailleurs la classification d'objets 3D et non la détection de changement. Shih et al. évoquent aussi dans [Shih et al. 2004] une comparaison binaire entre deux nuages de points (permettant d'évaluer les apparitions et disparitions de points) mais ils n'explicitent pas leur méthode de calcul.

Sinon, des techniques ont été développées récemment pour comparer directement des données laser aéroportées : [Murakami et al. 1999], [Vögtle et Steinle 2004] et [Vosselman et al. 2004]. Mais ces données étant en réalité en  $2D^{1/2}$  et non en 3D (à cause de la trajectoire du vecteur et de l'orientation du capteur), les algorithmes de comparaison se basent sur des cartes de distances (des grilles régulières 2D ayant une valeur d'élévation en chaque case, analogues à des images). Ces techniques de comparaison sont donc très proches du traitement d'image 2D classique. Elles ne conviennent pas en l'état à des données denses et

---

réellement 3D. Nous proposons donc une nouvelle approche pour le calcul des distances inter-nuages qui est applicable à des données telles que les relevés laser.

### 2.3.1 Distances inter-nuages

On a vu dans les remarques préliminaires de ce chapitre qu'il existait des distances inter-nuages globales et d'autres locales. Une comparaison globale ne permettant pas l'analyse fine des différences géométriques et encore moins leur caractérisation, nous pouvons donc écarter ce type de distances. Notre but est de quantifier le plus précisément possible, et en un temps raisonnable, la distance géométrique qui sépare chaque point du nuage comparé à la surface implicitement représentée par le nuage de référence qui lui correspond (ou du moins la plus proche si aucune ne lui correspond). En l'absence de notion d'objet, de surface et plus généralement d'une quelconque relation d'équivalence entre un point du nuage comparé et un point du nuage de référence, nous nous limiterons à évaluer la distance de chaque point du nuage comparé au point le plus proche dans le nuage de référence. Par la suite, on désignera ces deux points comme étant *homologues*.

#### Quelques distances locales simples entre ensembles de points

On peut définir de nombreuses distances qui se basent sur la notion de voisinage cubique induit par une grille 3D. Le principe est d'appliquer le même processus de subdivision aux deux nuages comparés en partant du même cube initial (typiquement le plus petit cube englobant les deux nuages). Ainsi les cellules des deux grilles sont spatialement équivalentes. On espère alors que les points de chaque nuage présents dans des cellules *homologues* représentent potentiellement la même surface (ce qui est partiellement vrai si les mouvements sont inférieurs à la taille d'une cellule). Cela permet d'appliquer des opérations de calcul de distance extrêmement rapidement puisque les calculs sont effectués cellule par cellule (par exemple : la moyenne ou la valeur minimale ou encore la valeur maximale des distances inter-points, l'angle entre les plans qui interpolent au sens des moindres carrés chaque ensemble de points, etc.). Plusieurs distances de ce type sont décrites dans [Girardeau-Montaut et al. 2005]. On évite aussi par ce biais les coûteuses opérations de recherche de voisinage. La précision est par contre toute relative, en particulier parce que l'on se base sur une notion de voisinage approximative et arbitraire. Les effets de bords peuvent être importants. Il peut de plus rester de nombreux points non comparés. On verra donc comment améliorer ces mesures de distances.

#### Distances au plus proche voisin

Nous appliquons ici un calcul similaire à celui nécessaire au calcul de la distance de Hausdorff *directe*, introduite en début de chapitre (équation 2.2), mis à part que nous ne

---

considérons pas la distance maximale mais bien la distance en chaque point. Cela revient à calculer pour tout point d'un nuage  $S$  sa distance au point le plus proche dans le nuage  $S'$ . On désignera donc par la suite l'ensemble des valeurs  $\{d_p = d(p, S'), p \in S\}$  comme les distances *au plus proche voisin* d'un nuage  $S$  par rapport à un nuage  $S'$ .

D'un point de vue algorithmique, le calcul des distances *au plus proche voisin* revient à déterminer pour chaque point de ce nuage l'ensemble de ses  $k$  plus proches voisins dans l'autre nuage avec  $k = 1$ . Une approche naïve (Cf. algorithme 2), à l'instar du calcul de la distance d'un nuage à un modèle triangulé, est quadratique et donc totalement inutilisable sur des nuages de plus de quelques milliers de points. Cela nous amènera au chapitre 4 à la notion de structuration du nuage de points 3D et à l'utilisation d'une structure octree pour améliorer la vitesse de détermination du voisinage d'un point.

---

**Algorithme 2** Calcul des distances *au plus proche voisin* d'un nuage  $S$  à un nuage  $S'$  (algorithme naïf)

---

```

pour  $\forall p \in S$  faire
   $d_p \leftarrow \text{inf}_+$ 
  pour  $\forall p' \in S'$  faire
     $d_p \leftarrow \min(d_p, \|p - p'\|_2)$ 
  fin pour
fin pour

```

---

Grâce à un octree (ou autre arbre de partition de l'espace), les calculs deviennent suffisamment rapides pour être appliqués à des nuages de plusieurs millions de points en quelques minutes voire quelques dizaines de secondes. Le temps de calcul dépend par contre beaucoup de la géométrie locale et de la proximité des deux nuages. On verra aussi que, dans le cas des octree tels que celui qu'on présente au Chapitre 4, le niveau de subdivision auquel la recherche de voisinage est effectuée est aussi un facteur important (Cf. section 4.5).

### Influence de l'échantillonnage sur la mesure

On étudie ici l'influence de l'échantillonnage du nuage de points de référence sur la précision de l'évaluation des distances obtenues par un calcul de distances *au plus proche voisin*.

Pour simplifier le problème, on considèrera des nuages constitués de points échantillonnés au hasard sur un plan (voir figure 2.3). Un tel semis de points aléatoires correspond à une distribution de Poisson<sup>3</sup> en deux dimensions.

---

<sup>3</sup>Un processus de Poisson est un processus ponctuel homogène isotrope pour lequel la disposition des points est complètement aléatoire à chaque réalisation. La probabilité de présence d'un point à une position donnée est en particulier indépendante de la position des autres points du processus.

---

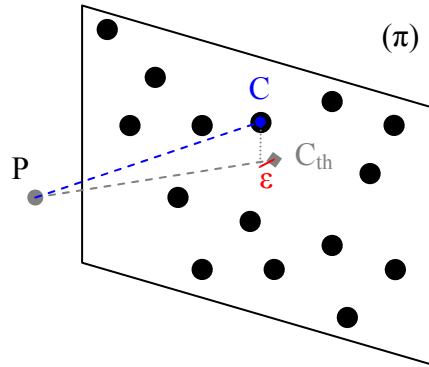


FIG. 2.3 – Illustration de l'erreur  $\varepsilon$  d'estimation de la distance d'un point P à un plan  $(\pi)$  par le calcul de la distance de P à un nuage de points échantillonnés sur ce plan.

Soient  $S_1$  et  $S_2$  deux nuages *de Poisson* (tels que définis ci-dessus) qui partagent le même plan  $(\pi)$ . En somme,  $S_1$  et  $S_2$  correspondent à deux acquisitions "aléatoires" d'une même surface plane, sans mouvement ni bruit de mesure. Par construction, la distance de  $S_2$  par rapport à la surface implicite représentée par  $S_1$  (le plan  $(\pi)$  ici) est nulle. Or, le calcul des distances *au plus proche voisin* entre  $S_2$  et  $S_1$  nous donnera des valeurs non nulles, égales, pour chacun des points de  $S_2$ , aux distances de leur plus proche voisin dans  $S_1$ . La théorie des distributions de Poisson nous indique que cette distance (c.à.d. l'écart entre un point quelconque du plan et son plus proche voisin aléatoire) suit une distribution de la forme :

$$P(d) = \lambda \cdot 2\pi \cdot d \cdot e^{-\lambda \cdot \pi \cdot d^2} \quad (2.4)$$

où  $d$  est la distance et  $\lambda$  la densité surfacique du semis de points aléatoires

Cette distribution a une moyenne  $\mu = \frac{1}{\sqrt{2\pi\lambda}}$  et une variance égale à  $\sigma^2 = \frac{1}{2\pi\lambda}$ . Toujours en théorie on devrait donc avoir une moyenne quadratique des distances égale à :

$$\sqrt{E[X^2]} = \sqrt{E[X - \mu]^2 + \mu^2} = \sqrt{\sigma^2 + \mu^2} = \frac{1}{\sqrt{\pi\lambda}} \quad (2.5)$$

On peut tester numériquement ce comportement. Soit un carré  $S_C$  de 100 mm de côté. On crée plusieurs ensembles de points tirés aléatoirement sur  $S_C$  et dénommés  $S_N$  où  $N$  est le nombre de points constituant chaque ensemble. On simule ainsi le processus de Poisson décrit ci-dessus, avec  $\lambda = \frac{N}{\text{aire}(S_C)}$ . On va alors tirer  $K$  points sur  $S_C$ , là encore aléatoirement, et calculer pour chacun de ces points sa distance au plus proche voisin par rapport à  $S_N$ . On pourra enfin calculer la moyenne quadratique de ces  $K$  distances (soit l'équivalent de la formule 2.5). L'opération est répétée pour les différentes valeurs de  $N$ , ce qui nous permettra de vérifier la loi de l'équation 2.4.

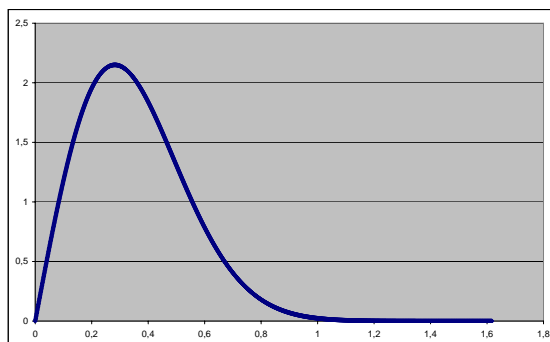


FIG. 2.4 – Distribution théorique des distances au plus proche voisin dans un processus de Poisson ( $\lambda = 2$ , voir equation 2.4).

Expérimentalement, on a choisi  $K = 10000$  et des valeurs de  $N$  comprises entre 5000 et 2 millions de points. On peut visualiser la courbe des valeurs de la moyenne quadratique ainsi mesurées (en bleu, figure 2.5) et celle des valeurs théoriques (en rouge, même figure), le tout en fonction de  $N$ . On peut voir que ces courbes sont très proches pour des valeurs de  $N$  supérieures à 10 000 ( $K$ , en fait). La théorie semble donc au moins confirmée pour des valeurs relativement grandes de  $N$ .

On peut aussi tracer pour différentes valeurs de  $N$  les histogrammes des  $K$  distances mesurées (donc le nombre cumulé d'occurrences pour différentes valeurs de distance). On peut voir 4 exemples de tels histogrammes pour  $N = 5000$ , 20000, 500000 et 5 millions de points (voir figure 2.6). Là encore les histogrammes sont proches de la courbe théorique correspondant à la formule 2.4 pour des valeurs de  $N$  supérieures à  $K$ . Par contre, sur l'histogramme correspondant à  $N = 5000$ , on voit apparaître une seconde distribution parasite.

Les écarts observés avec le modèle théorique, aussi bien au niveau du calcul de la moyenne quadratique que des histogrammes des distances, s'explique principalement par un effet de bord : les points tirés sur les bords du carré  $S_C$  ont en effet des voisinages tronqués, et leur plus proche voisin *théorique* se trouve parfois en dehors de  $S_C$ . La distance mesurée est donc dans ces cas plus grande. Mais cet effet est rapidement *dilué* pour des valeurs importantes de  $N$ , puisque les points concernés représentent un faible pourcentage du total et que la densité des points augmente.

Néanmoins, pour des valeurs supérieures à  $K$ , les mesures réelles respectent tout à fait le modèle théorique, et on observe alors que la moyenne quadratique des distances, qui est égale à l'erreur causée par la distance au plus proche voisin dans ce cas, diminue très rapidement en fonction de  $N$ . Cela laisse donc penser que si le nombre de points d'un nuage représentant une surface est suffisamment dense, il est possible d'évaluer la

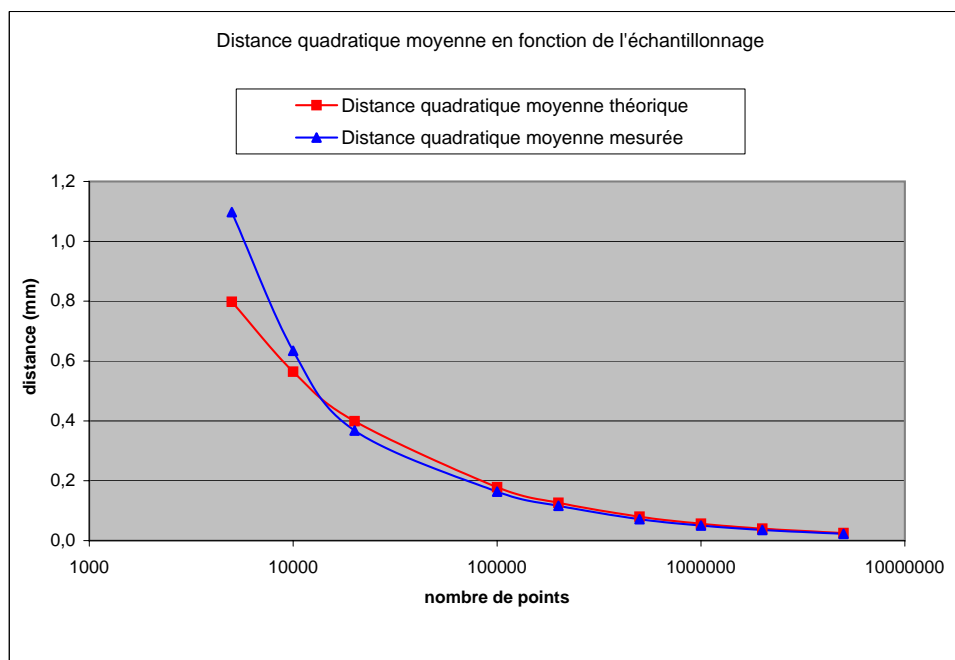


FIG. 2.5 – Valeurs mesurées (bleu) et théoriques (rouges) de l'erreur quadratique moyenne en fonction du nombre de points échantillonnés aléatoirement sur un carré de 100 mm de côté (échelle des abscisses logarithmique).

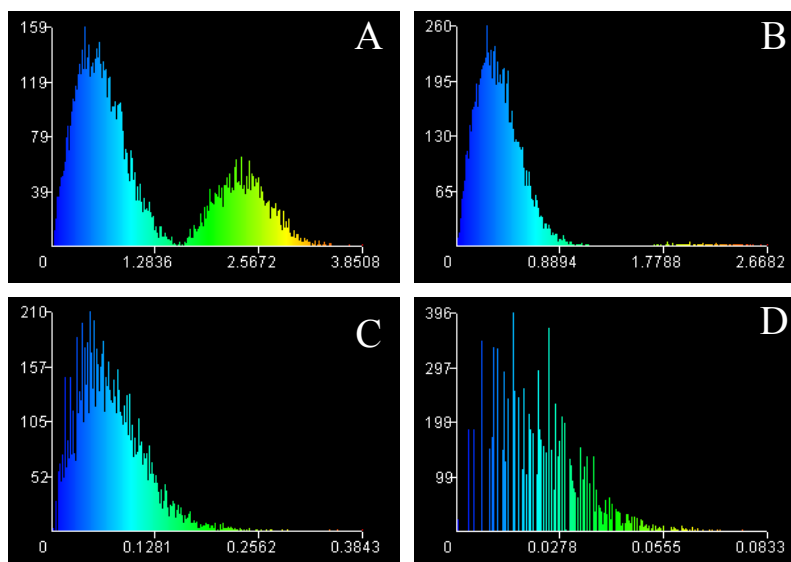


FIG. 2.6 – Visualisation des histogrammes des distances des points ( $K = 10000$ ) calculées par rapport à différents semis de points aléatoires tirés sur un carré de 100 mm de côté.

*A) 5000 points, B) 20 000 points, C) 500 000 points et D) 5 millions de points.*

*Les valeurs des abscisses sont en mm, et les ordonnées en "nombre de points".*

distance théorique de n'importe quel point de l'espace à cette surface avec une erreur limitée, en calculant simplement la distance de ce point au nuage. Il n'est alors pas forcément nécessaire de modéliser la surface de manière explicite si le nombre de points est suffisant. Le nuage contient en effet une description implicite de celle-ci. La précision du calcul dépendra alors principalement de la densité du nuage et des erreurs de mesure. En pratique, les capteurs laser permettent d'acquérir un nombre de points tellement grand qu'une bonne précision peut être atteinte dans le calcul de distance directement à partir du nuage. Il faut cependant mettre en regard l'erreur induite par l'échantillonnage et l'amplitude des déformations que l'on souhaite détecter. De toute façon la précision du calcul des distances sera en général limitée par celle des mesures (sans compter les erreurs numériques).

On peut remarquer que dans le cas *idéal* de nuages de points échantillonnés au hasard sur un plan (comme ceux envisagés ici), l'erreur quadratique moyenne est égale à la *distance moyenne inter-points* théorique qui peut être déterminée de la façon suivante : si  $N$  est le nombre de points échantillonnés et  $A$  l'aire de  $S_C$ , alors la surface moyenne par point est égale à  $\frac{A}{N}$ . Si on admet que cette surface est un disque parfait de rayon  $R$  ( $R$  représentant alors la *distance moyenne inter-points*), la surface de ce disque est  $\frac{A}{N} = \pi R^2$ . Donc on obtient au final :

$$R = \sqrt{\frac{A}{N\pi}} = \frac{1}{\sqrt{\lambda\pi}} \quad (2.6)$$

Le modèle *poissonien* envisagé ici pour les nuages est très simpliste. Les nuages de points laser sont en effet plutôt régulièrement échantillonnés selon des *lignes de scan* à cause de leur principe d'acquisition (par rotation d'axes ou de miroirs). Ils ne sont généralement pas non plus parfaitement plans, même en ne considérant que des voisinages locaux. Néanmoins, la formule 2.6 permet d'évaluer (et prédire) assez précisément l'erreur de mesure due à l'échantillonnage si les surfaces sont, au moins localement, suffisamment planes et relativement uniformément échantillonnées.

Note : il resterait à étudier l'influence d'un décalage relatif des deux nuages selon la troisième dimension (c.à.d. entre deux nuages *de Poisson* qui ne partagent pas le même plan) sur l'erreur due à l'échantillonnage. On pourrait aussi étudier l'influence de l'échantillonnage régulier des points sous forme de *lignes de scan*.

### 2.3.2 Améliorations de la mesure de distance

On a vu dans la partie précédente que l'utilisation d'un nuage de points à la place d'une surface modélisée est envisageable si ce nuage est suffisamment dense. Mais la densité de l'échantillonnage des points n'est pas la seule variable à prendre en compte. Les données lasers sont rarement échantillonnées de manière uniforme (cela dépend principalement de l'angle selon lequel le capteur "voit" la surface, du matériau de la surface et de la portée du capteur) et elles sont entachées de multiples erreurs (certaines sont propres au capteur

mais d'autres sont extrinsèques).

Le problème du bruit des mesures ne sera pas traité ici, au niveau du calcul des distances, car il est très complexe et le bruit est difficile à modéliser de manière réaliste sans un grand nombre d'informations sur le capteur et sur les conditions d'acquisition. Pour le cas particulier des capteurs laser terrestre, les articles [Boehler et al. 2003] et [Lichti et Gordon 2004] donnent un bon aperçu respectivement de la forte variabilité de l'erreur en fonction du modèle de capteur, et du grand nombre de sources d'erreurs à prendre en compte. On présentera en section 3.2.3 une méthode permettant la prise en compte du bruit a posteriori, lors de l'analyse des valeurs de distances, selon une approche plus pragmatique.

### Modélisation locale

Le problème de l'échantillonnage non uniforme ou peu dense du nuage de référence peut être traité efficacement dans une majorité de cas par une modélisation locale de la surface.

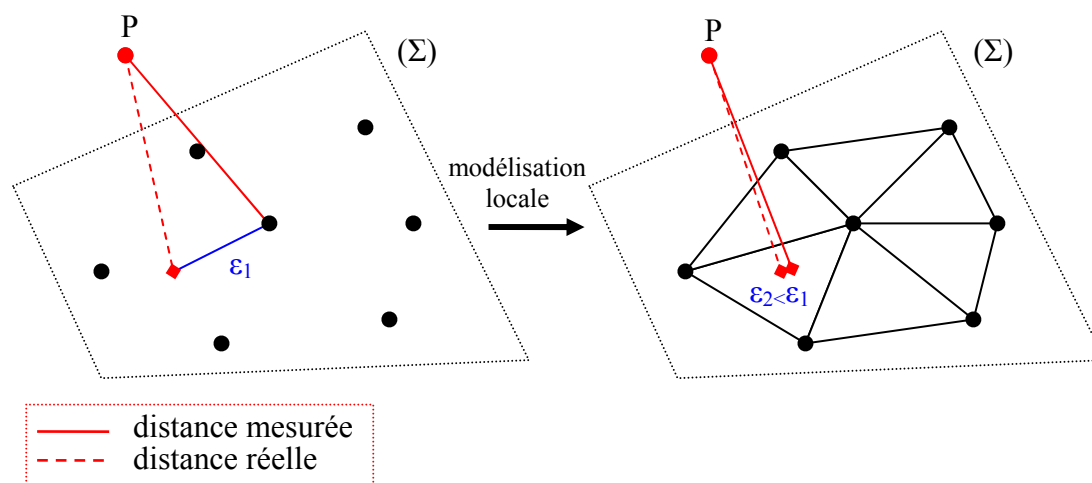


FIG. 2.7 – Principe de l'amélioration de la mesure sur nuages peu denses par modélisation locale.

Pour chaque point du nuage comparé, plutôt que de rechercher uniquement le point le plus proche, il est possible de déterminer ses  $k$  plus proches voisins dans le nuage de référence. Le nombre  $k$  sera typiquement fixé en fonction de la densité du nuage de référence, mais il est possible d'imaginer des conditions portant non pas sur le nombre de voisins mais plutôt sur l'extension spatiale du voisinage ou encore sur d'autres caractéristiques géométriques comme sa courbure. Ce voisinage doit, dans la majeure partie des cas, avoir une forme relativement plane et simple qui pourra alors être aisément modélisée.



On a utilisé lors de nos différents tests trois types de modélisations locales :

- interpolation des points par un plan construit par la méthode des moindres carrés (très rapide, mais peu précise sur des zones non planes),
- triangulation de Delaunay 2D des points projetés sur ce même plan "aux moindres carrés" (un peu moins rapide mais beaucoup plus robuste),
- et enfin, interpolation des points par une fonction de hauteur quadratique de la forme  $ax^2 + bx + cxy + dy + ey^2 + f = z$ , définie elle aussi par rapport au plan "aux moindres carrés" (les paramètres de l'équation sont déterminés par descente de gradient). Cette méthode est légèrement plus lente que la précédente, mais elle prend bien mieux en compte la courbure potentielle des surfaces implicitement représentées par le nuage de points.

D'autres modèles plus complexes pourraient être envisagés (comme une surface spline) mais le temps de calcul global ne doit pas être trop pénalisé par cette opération, car on risque de perdre l'un des intérêts majeurs du principe de la comparaison directe de deux nuages de points, à savoir sa rapidité.

Avec l'approche proposée, on se ramène localement à un problème de comparaison d'un point à un modèle (voir figure 2.7), tout en évitant d'avoir à calculer un modèle global et tous les problèmes qui accompagnent ce genre de processus (lenteur, consommation de mémoire, etc.). Étant donné le faible nombre de triangles en jeu, il n'est pas non plus nécessaire d'appliquer un algorithme de type *Metro*. L'algorithme 1 (naïf) est, dans ce cas précis, optimal.

On rencontrera toujours des cas limites où la modélisation locale sera quelque peu aberrante. Mais le résultat du calcul d'écart ne devrait alors pas être pire que celui obtenu avec une simple distance *au plus proche voisin* qui, dans ces cas particuliers, est généralement au moins aussi médiocre. On garde donc quelques erreurs par endroits, mais le résultat global est largement meilleur.

Cette approche n'est pas nécessaire pour des nuages très denses et il est important de noter que seul le nuage de référence est concerné par cette opération si celle-ci doit être appliquée. Elle reste donc optionnelle. Elle n'est pas adaptée non plus à des échantillonnages très peu denses (en particulier si le nuage est uniquement composé des sommets des objets polygonaux, comme les coins des bâtiments dans un nuage photogrammétrique par exemple). Si la grande majorité des voisinages locaux n'est absolument pas modélisable correctement, il vaudra alors mieux créer un modèle global par une technique ad-hoc pour pouvoir effectuer la comparaison de façon efficace.

---

### 2.3.3 Gestion des problèmes liés aux points de vue du capteur

#### Cas des parties cachées

Un autre problème important avec les nuages issus de capteurs laser découle de leur principe même d'acquisition (Cf. section 1.5.1). Étant donné que toutes les mesures sont effectuées depuis un point unique, les données laser présentent de nombreuses zones vides d'information du fait des *zones d'ombre* (ou *parties cachées*) projetées par les objets proches du capteur sur ceux qui sont plus loin (on peut faire l'analogie avec un oeil dont la vision serait obstruée par les objets au premier plan). De plus, le balayage du laser se fait généralement dans une plage angulaire fixe, ce qui provoque un découpage arbitraire des bords du nuage, indépendant de la géométrie des objets. Au final on obtient souvent des données qui, une fois visualisées en 3D, présentent de nombreuses zones vides d'information et donc des descriptions souvent partielles de la surface des objets. Cela est encore plus vrai pour les scanners laser à *triangulation plane* qui vont avoir des zones d'ombre par rapport à l'émetteur laser mais aussi par rapport au récepteur CCD.

Donc, mis à part le cas où l'on dispose de nuages acquis depuis la même position et selon la même orientation précisément, il est probable que ces zones d'ombre et que les portions de surface scannées d'un même objet ne seront pas les mêmes d'un relevé à l'autre. Une conséquence directe de ce phénomène est que la distance au plus proche voisin va donner des mesures potentiellement aberrantes pour les points qui sont vus dans un relevé et pas dans l'autre. En effet, par définition, chaque point du nuage comparé est associé au point le plus proche dans le nuage de référence. Ce point existe toujours mais il n'a pas forcément de rapport avec celui auquel il est associé : chacun peut appartenir à une surface différente et/ou être très éloigné de l'autre. Un problème du même type se pose lorsque des objets bougent entre les deux acquisitions, et ce, même si le capteur est immobile.

Considérons par exemple la figure 2.8 qui représente deux scans  $S_1$  et  $S_2$  acquis depuis le même point de vue, respectivement à des moments  $T_1$  et  $T_2$ . On peut voir qu'entre  $T_1$  et  $T_2$ , un objet (un mur) a disparu. A  $T_1$  ce mur occultait une bonne partie du champ de vision du scanner (toute la zone  $B$ ). A  $T_2$ , le scanner laser qui n'a pas été déplacé a alors un champ de vision plus important, et peut acquérir des mesures ponctuelles 3D dans la zone  $B$ . Or, lorsque l'on compare  $S_2$  (en bleu) à  $S_1$  (en rouge), il est impossible de déduire quoi que ce soit au sujet des points de la zone  $B$  car rien ne nous permet de déterminer directement à quoi ressemblait cette zone à  $T_1$  (puisque elle n'a pas été effectivement observée par le laser à cette époque). La distance au plus proche voisin associera forcément ces points à des points de  $S_1$  hors zone d'ombre (donc dans  $A$  ou  $C$ ), et donc potentiellement très distants. Il est bien sûr possible que, par hasard, les points de la zone  $B$  à  $T_2$  correspondent à ceux de la zone  $A$  à  $T_1$  (si le mur s'était effondré d'un bloc vers la droite, mais ce n'est pas le cas). Dans tous les cas, rien ne nous permet de l'affirmer simplement. Il faut donc laisser ces points de côté dans un premier temps pour assurer la pertinence de l'algorithme.

---

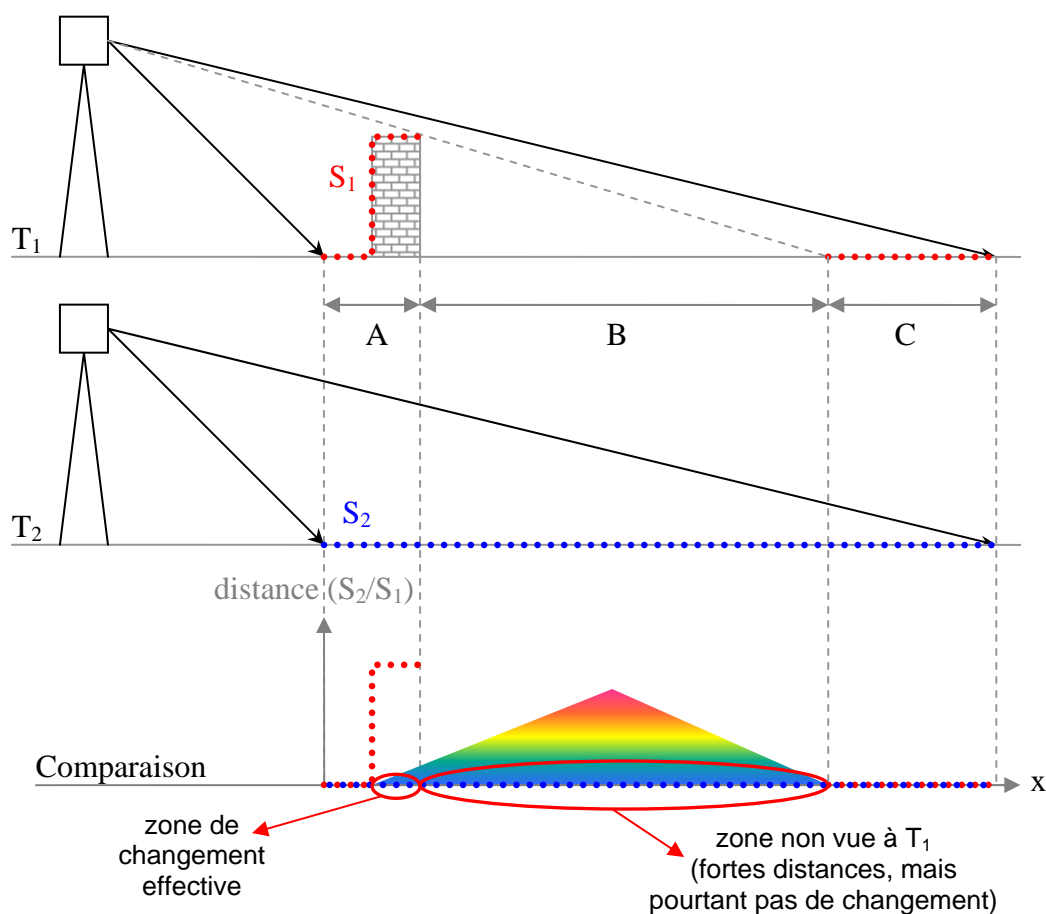


FIG. 2.8 – Illustration du problème des zones d'ombre dans les scans laser.

C'est sans doute le plus gros problème de cette distance lorsqu'elle est utilisée pour déterminer des différences géométriques. Pour faire face à ce problème, il faut donc filtrer les zones dissociées des nuages de point pour éviter de comparer "l'incomparable".

Pour ce faire, il est possible de se baser sur la cause même du problème, à savoir le principe d'acquisition des capteurs lasers. Ceux-ci balayent l'espace en lançant un rayon laser selon des angles longitudinaux et latitudinaux à peu près constants. Les rayons lasers ne se croisent jamais. Il est donc possible - et courant - de représenter un relevé effectué depuis un point de vue unique sous forme d'une grille  $2D^{1/2}$ , dont les axes horizontaux et verticaux correspondent aux angles de visée et dont la valeur en chaque "case" est égale à la distance parcourue par le rayon laser. On appelle cela une *carte* ou *image de distance* (*depth map* ou *depth image* en anglais). Ces grilles ne sont pourtant pas des images standard car certaines cases ne sont pas renseignées (absence d'écho laser à cause de la nature de la surface ou à cause de la limite de portée) et leur géométrie n'est pas euclidienne mais à projection sphérique (ce qui donne un léger effet *fish-eye*, du nom de la lentille qui imite la vision d'un oeil de poisson - voir figure 2.9). Certains capteurs utilisent d'ailleurs des pas angulaires de balayage non constants et il faut donc procéder

à un re-échantillonnage des données pour obtenir une structure régulière. Les petits trous de la grille peuvent être remplis par interpolation.

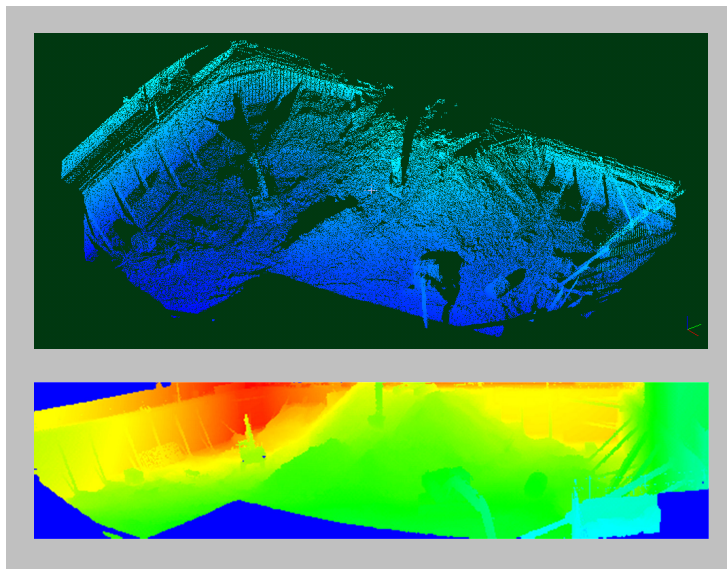


FIG. 2.9 – En haut : nuage de points (projection ortho-graphique, coloration en fonction de la hauteur). En bas : carte de distance correspondante (coloration en fonction de la distance des points au capteur laser).

Il est intéressant de faire l’analogie entre ces cartes de distance et la structure appelée *Z-Buffer*[Watkins 1970] dans le domaine du rendu graphique 3D. Cette structure a généralement la même taille que la mémoire d’affichage et contient la profondeur du point affiché en chaque pixel. Avant d’afficher un nouveau point correspondant à une entité 3D, on calcule sa profondeur par rapport au point de vue. Il est alors possible, grâce au *Z-Buffer*, de confronter cette profondeur à celle du pixel affiché : si le nouveau point est plus proche, il est affiché et le *Z-Buffer* est mis-à-jour. Sinon le point n’est pas affiché (il est *derrière* l’objet déjà affiché à l’écran et donc caché par celui-ci).

Les cartes de distances déduites d’un scan laser peuvent être utilisées de la même manière qu’un *Z-Buffer*, non pas pour un problème d’affichage mais justement pour déterminer les zones dissociées entre scans.

Voici la démarche à suivre pour déterminer simplement si un point  $P$  de  $S_2$  est *comparable* à  $S_1$ . On projette le point  $P$  dans le repère angulaire centré sur la position du capteur à  $T_1$  (en somme le repère de la carte de distance de  $S_1$  nommée  $C_1$ ). Soit  $P'$  le projeté de  $P$ . Si  $P'$  est en dehors des limites de  $C_1$ , alors  $P$  est en dehors du champ de vision du scanner et il est donc inutile de le comparer aux points de  $S_1$  par une technique du type *distance au plus proche voisin* (on peut le marquer comme étant *hors-champ*). Sinon, on effectue une comparaison commune avec un *Z-buffer* et il y a alors deux possibilités :

- Soit la distance entre  $P$  et le capteur (à  $T_1$ ) est inférieure ou égale à la valeur de la

- carte de distance  $C_1$  en  $P'$ . Dans ce cas,  $P$  est dans une zone de l'espace qui a été effectivement *vue* par le capteur à  $T_1$ . On peut donc le comparer aux points de  $S_1$ .
- Sinon, on marque ce point comme *invisible* et il faudra traiter son cas plus tard.

On peut aussi rajouter une dernière condition si l'on connaît la portée du capteur. Si la distance entre  $P$  et le capteur à  $T_1$  est supérieure à la portée du capteur, il peut être marqué comme *hors de portée*. Il ne devra pas être comparé en utilisant la distance au plus proche voisin, à l'instar des points *invisibles* et *hors-champ*. Des stratégies de récupération pourront être utilisées pour tous ces points "incomparables". Le processus de décision évoqué ci-dessus est synthétisé sous forme d'un algorithme (voir algorithme 2.3.3).

L'analogie entre une carte de distance et un Z-Buffer étant assez forte, la première hérite cependant des problèmes bien connus du second relatifs à la discrétisation de l'espace imposé par la grille et à l'imprécision de la détermination de la profondeur des points :

- Les zones de l'espace délimitées par une case de la grille peuvent être assez étendues, en particulier si la résolution de la grille est faible. Cela est d'autant plus vrai que la délimitation est angulaire dans le cas des cartes de profondeur d'un scan laser et la zone correspondante est donc pyramidale, avec une base qui s'agrandit rapidement avec la distance. Il peut donc y avoir de grandes discontinuités d'une case à l'autre (en particulier dans les zones du nuage où le laser a une incidence rasante). Cela fait apparaître dans le cas du Z-Buffer des sortes de marches (un effet crénelé très peu esthétique) et cela se traduit par des erreurs de filtrage pour les cartes de profondeur.
- Les distances des points au scanner souffrent des erreurs de mesures et de recalage, auxquelles s'ajoutent des imprécisions au niveau des calculs numériques. Il faut donc prévoir une marge d'erreur lors de la confrontation de la profondeur d'un point à la valeur de la grille. En pratique, on permet au point confronté d'avoir une profondeur inférieure ou égale à celle de la grille multipliée par un coefficient très légèrement supérieur à 1.

### Cas des points de vue multiples au sein d'un nuage

Il arrive couramment que les nuages de points soient composés de plusieurs relevés effectués depuis différentes positions (pour minimiser justement les zones d'ombre). Dans ce cas, il est nécessaire de connaître l'appartenance de chaque point du nuage aux différents scans qui le composent. La carte de distance de chacun des scans est calculée (Cf. figure 2.10), et l'ensemble forme une structure générique (qui englobe le cas où il n'y a qu'un seul point de vue). Le test de *visibilité* des points est alors fait en les comparant à chaque carte de distance de  $S_1$ , jusqu'à ce qu'ils soient *visibles* dans au moins une. Dans le cas contraire, on garde la caractéristique avérée la plus avantageuse (dans l'ordre subjectif : *invisible*, puis *hors de portée* et enfin *hors-champ*).

---

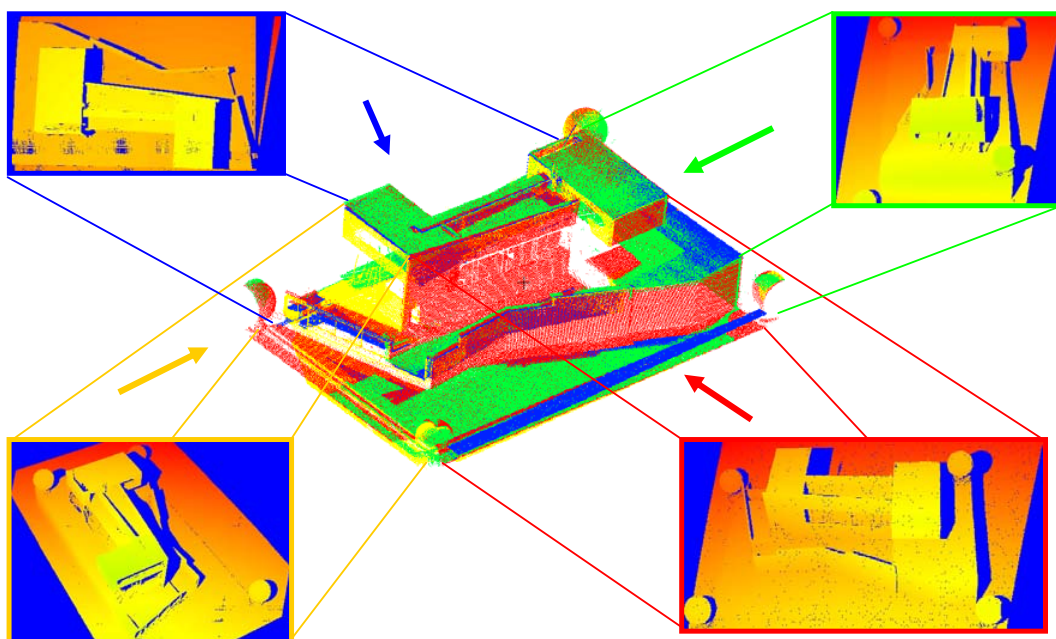


FIG. 2.10 – Illustration d'un nuage composé de multiples points de vue accompagné des cartes de profondeur de chacun (échelle de vert à rouge en fonction de l'éloignement).

*Le nuage représente une maquette (La maison de Rem Koolhaas, à Saint-Cloud) gracieusement prêtée par l'École d'Architecture de la Villette (Paris) et numérisée avec un scanner Soisic prêté par la société Trimble (ex-Mensi) lui aussi gracieusement.*

## 2.4 Résultats

Sont présentés dans cette section des illustrations et des temps indicatifs pour les différentes techniques de calcul de distance et leurs améliorations évoquées dans ce chapitre.

### 2.4.1 Comparaison Nuage/Maillage

Cette technique de comparaison pré-existant à cette thèse, nous nous concentrerons surtout sur les temps d'exécution de l'algorithme type *Metro* re-implémenté dans la plateforme de démonstration des algorithmes de cette thèse (dénommé *CloudCompare*, Cf. section C) pour pouvoir être associée à la structure d'octree proposée au chapitre 4. On prendra comme exemple illustratif une application de contrôle de réalisation de modèle 3D T.Q.C. (Cf. section 5.1) à l'intérieur d'une centrale électrique. Ce modèle est construit à partir d'un nuage de points laser. Le but était donc de contrôler la qualité du modèle, et en particulier l'oubli potentiel d'objets importants (Cf. figure 2.11).

Les problèmes de segmentation des distances et d'amélioration de la perception seront traités par la suite (Chapitre 3 pour la segmentation, et Annexe A pour la perception),

**Algorithme 3** Test de visibilité d'un point

**Entrées:** le point  $P$ ,  $\lambda$  la portée du capteur,  $N$  le nombre de points de vue et  $ZBuff$  les cartes de distance

**Sorties:**  $\beta$  la visibilité de  $P$

$\beta \leftarrow HORS\_CHAMP(= 3)$

**pour**  $1 \leq i \leq N$  **faire**

$Q \leftarrow$  projeter  $P$  selon le  $i^{eme}$  point de vue

**si**  $Q \in ZBuff[i]$  **alors**

**si** profondeur( $Q$ )  $< \lambda$  **alors**

**si** profondeur( $Q$ )  $< ZBuff[i](Q)$  **alors**

                renvoie  $VISIBLE(= 0)$

**sinon**

$\beta \leftarrow INVISIBLE(= 1)$

**fin si**

**sinon**

$\beta \leftarrow \min(\beta, HORS\_DE\_PORTEE(= 2))$

**fin si**

**fin si**

**fin pour**

renvoie  $\beta$

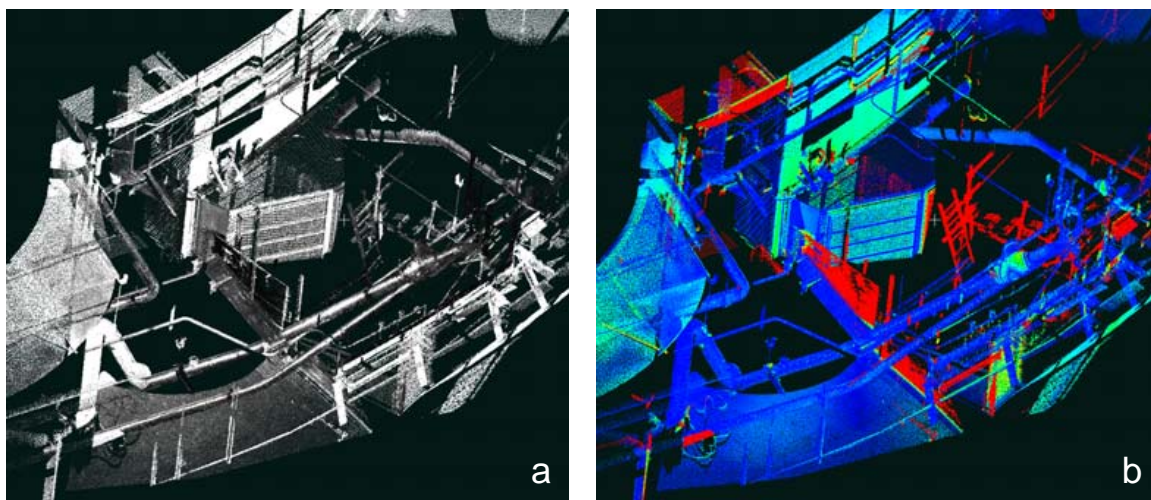


FIG. 2.11 – Comparaison d'un nuage de points laser (a) avec un modèle 3D (non représenté), et résultat du calcul de distance (b).

mais on peut déjà remarquer qu'afficher les distances selon une échelle fixe et globale pour tout le nuage peut poser problème étant donné que les *distances* d'un objet au modèle (ou à un autre nuage) sont relatives à sa taille et qu'un objet petit sera donc toujours moins pénalisé qu'un gros en cas d'apparition/disparition. Un algorithme de segmentation locale, capable de délimiter des objets ou portions d'objets apparues ou disparues, permettrait

de tous les afficher de la même manière, indépendamment de leur taille. Ceci rendrait la lecture des résultats plus claire ou du moins plus impartiale. En l'absence d'un tel outil de segmentation ou si celui-ci est inopérant ou trop lent, on a implémenté dans *CloudCompare* un système de potentiomètres permettant le réglage manuel interactif des différents paramètres d'affichage des distances (valeur minimale et maximale des distances des points affichés, saturation des couleurs, etc.). Cela permet, en jouant sur la dynamique d'affichage, de faire apparaître plus clairement les différences et leur géométrie. Il est ainsi possible d'adapter au mieux l'échelle de couleur à la zone d'intérêt (voir section 3.2.1).

Points	Triangles	Niv. 5	Niv. 6	Niv. 7	Niv. 8	Niv. 9
293587	38044	19,47	9,81	<b>7,6</b>	10,78	
2180182	24155		134,95	61,92	<b>46,94</b>	70,74
2725142	141494		429,13	128,85	63,4	<b>49,19</b>
2725142	53254		194,6	68,33	38,32	<b>38,03</b>
2975731	10089		37,64	<b>24,56</b>	25,44	
3412681	16553	105,19	<b>74,77</b>	91,13		

TAB. 2.1 – Temps de calcul indicatifs (en secondes) pour différents niveaux d'octree, lors de la comparaison d'un nuage de points à un modèle 3D triangulé avec un algorithme du type *Metro*

Le tableau 2.1 présente quelques temps de calcul indicatifs, mesurés sur un ordinateur portable doté d'un processeur Intel Centrino 1.7 GHz avec 1 Go de mémoire vive. Les résultats sont donnés pour différents niveaux d'octree, le meilleur apparaissant en gras. On peut remarquer que les temps ne sont pas linéaires en fonction du nombre de faces ou du nombre de points. Le niveau d'octree optimal auquel est fait le calcul ne semble pas obéir à une règle évidente (il semble par contre dépendre plutôt du nombre de triangles que du nombre de points).

## 2.4.2 Comparaison Nuage/Nuage

Les résultats ci-dessous sont principalement obtenus à partir d'un jeu de données issu d'une campagne de relevés laser effectués sur un chantier de déblaiement (en préparation d'un chantier de construction d'un immeuble, dans la ville de Malakoff dans les Hauts-de-Seine). Il consiste en 4 relevés effectués à un ou deux jours d'intervalle, depuis une position à peu près identique. Il y a un seul point de vue par scan. Étant donné la spécificité opératoire de ce relevé, on a aussi utilisé des nuages de points acquis en intérieur de centrale (certains ayant plusieurs points de vues) pour compléter le nombre de chronométrages présentés ci-dessous.

La comparaison par calcul de la distance *au plus proche voisin* sur deux nuages tels que ceux du chantier de Malakoff est très rapide comme l'atteste le tableau 2.2. Ce sont en effet des nuages de densité à peu près constante, avec une majorité de parties communes



et un nombre de points relativement faible (le nombre de points de chaque liste est rappelé entre parenthèses dans les colonnes "Référence" et "Comparé" du tableau). Seul un de ces nuages comporte une zone scannée volontairement beaucoup plus finement (le troisième jour). Ce nuage est quatre fois plus gros que les autres, néanmoins le temps de calcul n'est pas forcément plus élevé lorsqu'il sert de référence. En effet, l'augmentation du temps de calcul n'est pas proportionnelle au nombre de points du nuage de référence, mais évolue plutôt en fonction de celui du nuage comparé. Et comme on l'a déjà évoqué, la géométrie locale et les zones dissociées entre nuages influent aussi beaucoup sur le temps de calcul.

Référence	Comparé	Niv. 7 (en s.)
jour 2 (177171)	jour 1 (171380)	1,83
jour 4 (220931)	jour 1 (171380)	5,58
jour 3 (764665)	jour 2 (177171)	4,32
jour 4 (220931)	jour 2 (177171)	5,14
jour 2 (177171)	jour 3 (764665)	16,91

TAB. 2.2 – Temps de calcul indicatifs (en secondes) pour la comparaison de deux nuages de points du chantier de Malakoff par calcul de la distance *au plus proche voisin*.

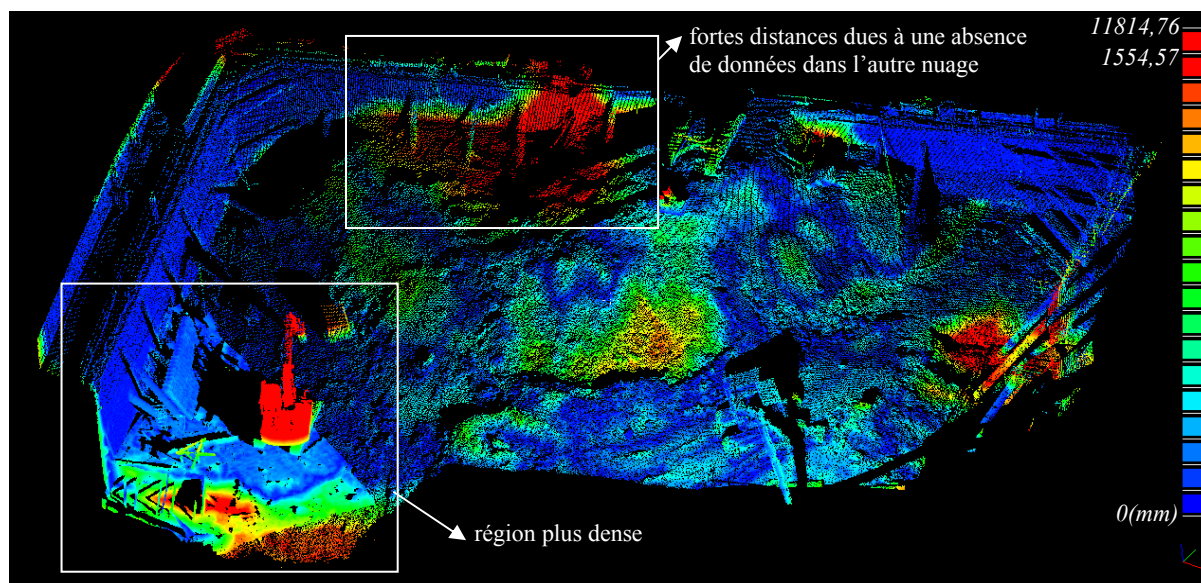


FIG. 2.12 – Visualisation des distances des points du nuage du 3ème jour par rapport au nuage du 2ème jour (non représenté) du chantier de Malakoff.

*Le nuage comporte une zone scannée plus finement (en bas à gauche). On peut remarquer l'arrivée d'une pelle mécanique et la présence d'une zone excavée à droite de l'image.*

La figure 2.12 présente le résultat d'un calcul de distance *au plus proche voisin* entre le deuxième jour et le troisième jour du chantier de Malakoff. La zone scannée plus finement

dans ce nuage n'a pas d'influence notable sur le résultat (mis à part le temps de calcul). On peut par contre observer d'autres problèmes caractéristiques de la distance *au plus proche voisin* : d'une part des zones planes et inchangées qui présentent des distances non homogènes (ce phénomène reste très faible, et n'est donc pas visible avec une échelle globale), et d'autre part des zones de distances très fortes qui ne correspondent pourtant pas à des changements (comme en haut de l'image). Le premier problème est celui de l'échantillonnage (Cf. section 2.3.2). L'autre est celui des points de vue et de la visibilité du capteur (Cf. section 2.3.3).

### 2.4.3 Résolution du problème d'échantillonnage

La figure 2.13 est une illustration du traitement par modélisation locale de l'erreur induite par l'échantillonnage des nuages de points sur le calcul de la distance *au plus proche voisin*. Elle correspond à une partie extraite du nuage représenté par la figure 2.12 (coin haut-gauche). On a modifié l'échelle de couleurs et filtré les points trop distants pour augmenter le contraste et ainsi faire mieux apparaître les variations de distances au niveau des zones planes.

L'utilisation d'une modélisation locale améliore visiblement le résultat sur de telles zones en rendant les mesures plus homogènes. La méthode utilisée pour la modélisation locale est ici la triangulation de Delaunay 2D appliquée aux points projetés sur leur plan obtenu par la technique des moindres carrés. Les temps de calcul sont à peine doublés. Le tableau 2.3 montre des mesures de temps effectuées sur différents nuages. Empiriquement, les temps de calcul ont été dans le pire des cas multipliés par 5.

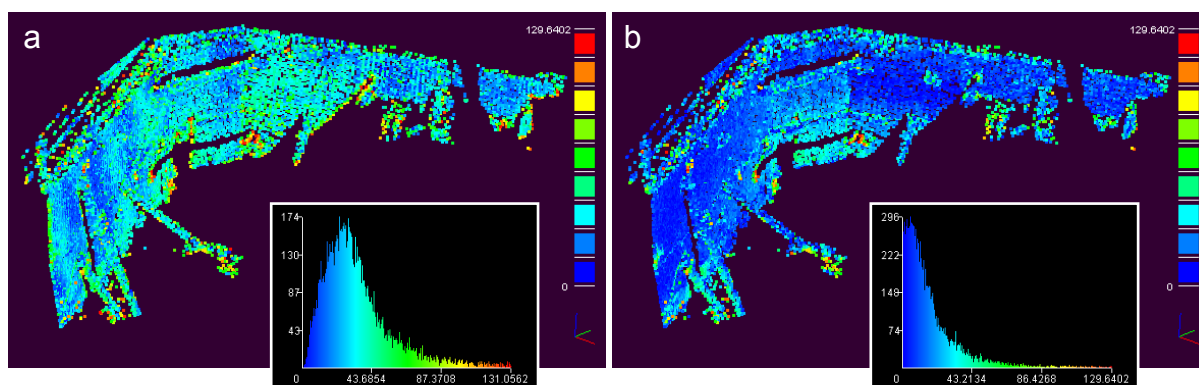


FIG. 2.13 – Influence de l'échantillonnage sur le calcul des distances *au plus proche voisin* : calculs des écarts et histogrammes correspondant sans (a) et avec (b) modélisation locale.

*Ici, la modélisation locale utilisée est la triangulation de Delaunay 2D. La figure représente un détail du nuage présenté en figure 2.12 (le coin en haut à gauche) avec une échelle de couleur différente pour faire ressortir les variations locales des distances.*

Référence	Comparé	Sans	Avec
jour 2 (177171)	jour 3 (764665)	4,32	9,57
trémie 1 (609804)	trémie 2 (837125)	18,42	29,47
couloir 1 (146352)	couloir 2 (292389)	4,88	10,46

TAB. 2.3 – Comparaison des temps (en secondes) pour le calcul de la distance *au plus proche voisin* entre deux nuages *sans* et *avec* modélisation locale.

"jour 2" et "jour 3" correspondent au chantier, et "trémie" et "couloir" à des nuages acquis en intérieur de centrale.

#### 2.4.4 Résolution du problème de visibilité

Pour déterminer la visibilité des points comparés selon le ou les points de vue du capteur ayant acquis le nuage de référence, on a vu qu'on pouvait calculer les cartes de profondeur des relevés correspondant à ces différents points de vue. On évite ainsi de prendre en compte des points qui ne sont pas *a priori comparables* au sens de la distance *au plus proche voisin*. Le calcul d'une carte de profondeur se fait par projection des points dans le repère du capteur. C'est une opération très rapide, même effectuée sur des millions de points et couplée à une opération de remplissage des trous. Le temps supplémentaire nécessaire pour la projection des points du nuage comparé - pour les confronter à la carte de profondeur - est équivalent, mais cette opération est faite en même temps que le calcul de la distance *au plus proche voisin*, et le processus global peut donc être légèrement ralenti. Mais il peut aussi bien être accéléré si le nombre de points considérés comme *invisibles* est important (leur distance n'étant pas calculée).

La figure 2.14 montre le résultat du calcul de la distance *au plus proche voisin* "filtrée", appliquée au même jeu de données que celui de la figure 2.12. On peut remarquer en particulier que la zone du haut présentant précédemment des distances très fortes est ici correctement filtrée. Par contre, d'autres zones qui étaient comparées *à raison* se retrouvent elles aussi filtrées. Algorithmiquement parlant elle ne sont pas différentiables. Il faudra donc les traiter dans un processus de rattrapage. Celui-ci peut être semi-automatique (si le nombre de zones filtrées n'est pas trop important) et simplement consister à demander à l'utilisateur de confirmer ou non le filtrage de chaque zone. Mais on peut aussi utiliser des heuristiques simples qui vont dépendre d'une connaissance *a priori* de l'utilisateur au sujet des données. Par exemple, sur des données de terrain comme celles du chantier de Malakoff, il est très efficace d'appliquer aux points filtrés une comparaison  $2D^{1/2}$ . C'est-à-dire qu'on compare toujours les points avec une distance *au plus proche voisin* mais selon une unique direction (la verticale par exemple, si la gravité a joué un rôle dans le processus de modification). Cf. [Girardeau-Montaut et al. 2005] (voir Annexe E.1) pour plus de détails ainsi qu'une discussion sur ce que peuvent être d'autres heuristiques.

La figure 2.15 montre un résultat équivalent sur deux nuages acquis en centrale et représentant des structures théoriquement équivalentes mais physiquement distinctes. Les

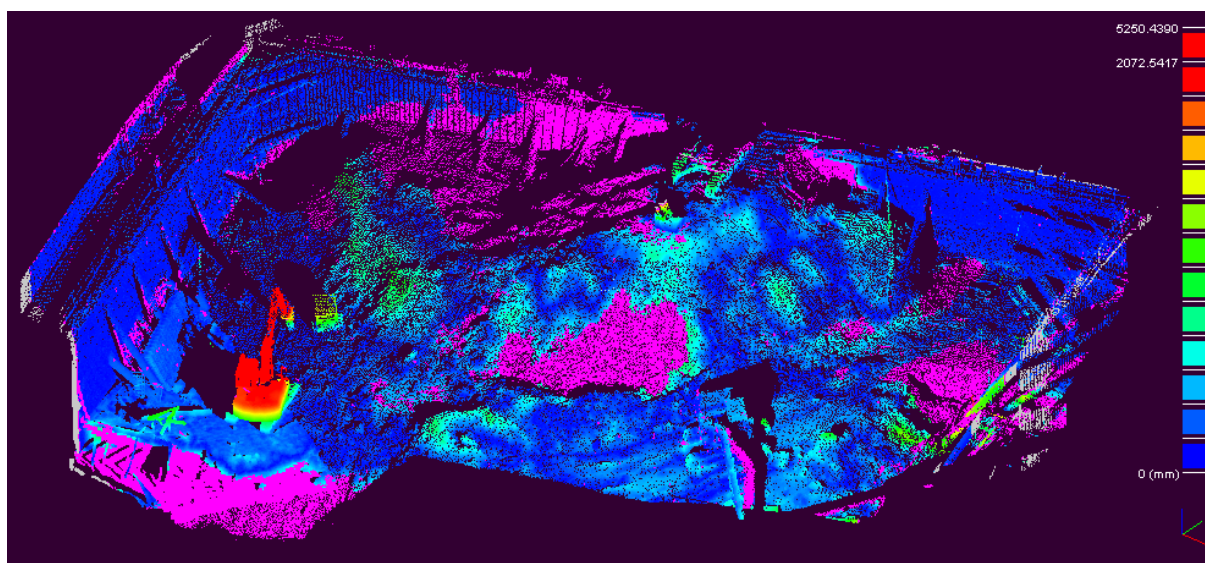


FIG. 2.14 – Calcul des distances *au plus proche voisin* filtrées en fonction de la visibilité, entre le 2<sup>ème</sup> et le 3<sup>ème</sup> jour du chantier de Malakoff.

*Les points "hors-champ" sont en gris et les points "invisibles" en rose. Ce résultat est à comparer avec la figure 2.12. Si la pelle mécanique a bien été conservée et l'artéfact du haut de l'image supprimé, l'excavation à droite de l'image a été par contre filtrée "à tort" (dans le sens où un utilisateur considérerait que c'est bien une zone de différence, et non une simple zone d'ombre).*

nuages sont beaucoup plus épars et sont composés respectivement de 6 et 10 points de vues. Le nombre de zones dissociées est très important et rend la lecture difficile sans filtrage (à gauche). Une fois filtrées, la visualisation des distances est beaucoup plus claire (à droite) et permet de se concentrer sur de véritables différences. Un processus de segmentation qui se basera sur la valeurs des écarts sera aussi plus robuste car il ne prendra ainsi en compte que les "vraies" différences.

Lorsque l'on travaille avec des nuages de points, des outils tels qu'une segmentation manuelle interactive du nuage ou encore la possibilité de *naviguer* en 3D autour de ce nuage sont de toute façon nécessaires pour une bonne perception des détails. Il est cependant possible, à partir du résultat du calcul de distances, d'envisager maintenant des approches de plus haut niveau permettant de segmenter et de reconnaître automatiquement ou semi-automatiquement les zones de changements. Le but étant là encore d'éviter à l'utilisateur un travail qui peut se montrer pénible ou difficile en attirant son attention sur les zones intéressantes et en tentant de pré-analyser la situation. C'est le thème du prochain chapitre.

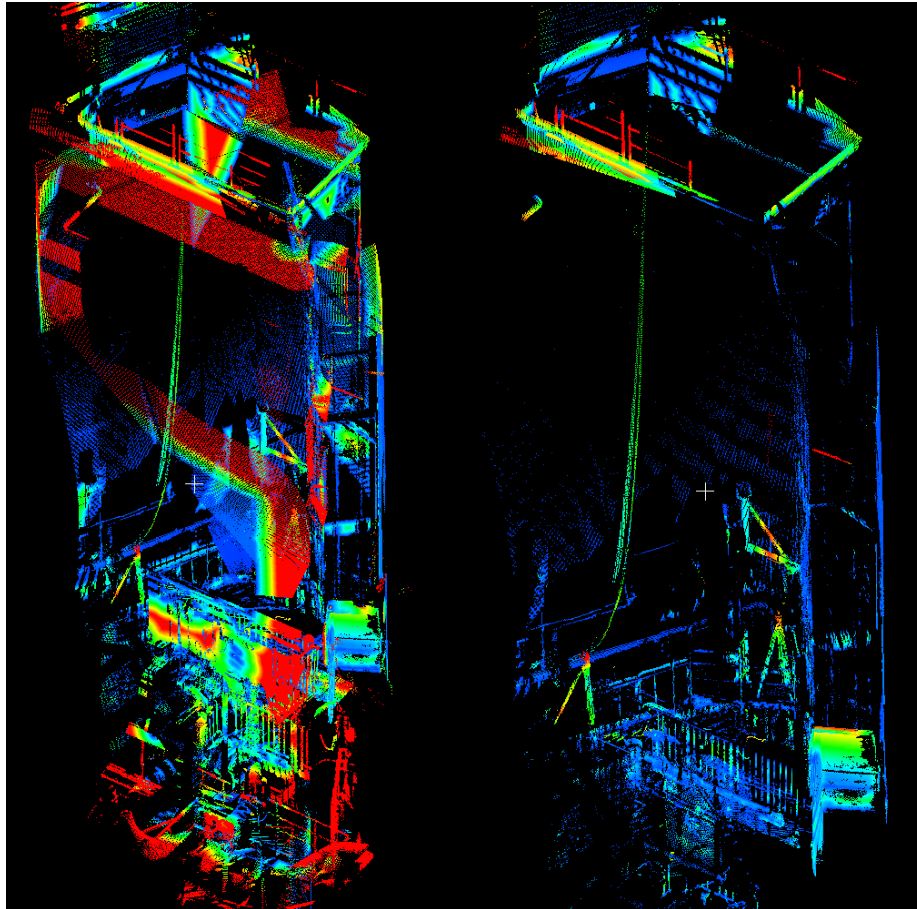


FIG. 2.15 – Calcul des distances *au plus proche voisin* entre deux trémies (puits à section trapézoïdale) d'une centrale EDF.

*A gauche : résultat brut ; à droite : filtrage en fonction de la visibilité.*

## Chapitre 3

# Exploitation de la mesure de distance

Le chapitre précédent traitait du calcul en chaque point d'un nuage 3D représentant un objet réel d'une distance par rapport à un autre état du même objet ou à un autre objet (nous avons vu qu'il était tolérable de faire l'amalgame entre ces deux notions au début du Chapitre 1). Pour calculer ces distances on a proposé deux méthodes qui permettent de s'adapter à la très grande majorité des cas qu'il est possible de rencontrer. L'une pré-existait à ces travaux et permet de comparer un nuage à un maillage (et par extension deux maillages entre eux), alors que l'autre est une méthode inédite que nous avons développée et qui permet la comparaison directe de deux nuages de points. Dans les deux cas, on obtient une valeur de distance en chaque point du nuage comparé. Cette valeur est une approximation plus ou moins précise du déplacement local de la surface implicitement représentée par le nuage, soit par rapport à un état antérieur, soit par rapport à une autre surface. On désignera dans la suite cette valeur par les termes "déplacement" ou "écart" plutôt que "distance" qui pourrait porter à confusion.

Dans le même esprit, et toujours selon la remarque faite en début du Chapitre 1, on fera l'amalgame par la suite entre les associations de termes "mobiles/immobiles" et "changés/inchangés" (le lecteur pourra les intervertir librement).

Pour se placer dans un cas plus général, on peut maintenant considérer qu'on dispose simplement comme donnée d'entrée d'un nuage avec une valeur scalaire en chaque point (le fait que cette valeur corresponde à un écart devient quelque peu secondaire). Un tel nuage peut être considéré comme un échantillonnage particulier d'un champ scalaire surfacique. Notre but va alors être de segmenter ce champ scalaire surfacique pour isoler les zones où celui-ci est non nul (et qui correspondent donc aux zones de changement dans notre cas). On cherchera si possible un résultat sous forme de sous-groupes de points cohérents, par exemple en fonction de leur proximité relative, espérant ainsi que ces sous-groupes correspondent à des morceaux de surfaces, voire à des objets entiers qui auraient subi un changement.

---

Le but d'une telle segmentation est bien entendu d'une part de simplifier et d'accélérer l'analyse en la limitant uniquement aux zones d'intérêt du nuage (celles ayant subi des changements) mais aussi de la porter à un niveau plus abstrait en ne traitant qu'un nombre limité de groupes de points qui devraient, idéalement, représenter des entités élémentaires au sens de l'application envisagée (un tel découpage impliquera par contre des traitements spécifiques faisant intervenir des connaissances à priori).

## 3.1 Rapide état de l'art sur la segmentation

Sans prétendre faire un état de l'art complet sur le domaine de la segmentation (puisque celui-ci est très vaste, et que ce n'est pas l'objet de cette thèse), nous dressons dans un premier temps un rapide inventaire des méthodes de segmentation existantes. Cet inventaire nous a permis de choisir la direction de nos essais dans le domaine, qui font l'objet de la deuxième partie ce chapitre (la troisième étant consacrée à l'étude de l'apport potentiel d'une prise en compte de la symétrie de comparaison pour améliorer la détection et l'analyse des changements).

Parmi les techniques de segmentation applicables aux nuages de points 3D, nous distinguerons les méthodes issues du traitement d'image classique 2D (avec plus ou moins d'adhérence au caractère bidimensionnel et régulier des données) des méthodes réellement 3D, ayant des spécificités propres à la troisième dimension, et ne nécessitant pas de réduction de la dimension du problème (comme l'utilisation des cartes de profondeurs 2D<sup>1/2</sup> produites par un scanner laser par exemple).

### 3.1.1 Segmentation 3D de nuages de points

Si l'on se réfère à la thèse de Thomas Chaperon [Chaperon 2002] sur la question, on observe que la plupart des méthodes de segmentation de nuages de points réellement 3D se basent sur la construction préalable d'un modèle (en général, un maillage). A partir de ce modèle, il est possible de calculer des orientations, des discontinuités ou des courbures qui, associées avec la notion de proximité et de surface (elles aussi fournies par le modèle), permettent de regrouper les points similaires proches les uns des autres. Ces méthodes donnent parfois de très bons résultats, mais sur des modèles simples et relativement réguliers. De toutes façons, les raisons qui nous ont poussé à éviter le calcul d'un modèle sur un nuage de points (évoquées en section 2.3) restant d'actualité, nous ne nous intéresserons pas plus avant à ces méthodes.

D'autres méthodes réellement 3D ne reposent pas sur la construction d'un modèle 3D, mais plutôt sur la projection de la position des points ainsi que de certaines caractéristiques géométriques dans un espace de paramètres plus ou moins compliqué. Nous pensons en particulier aux techniques de *Tensor Voting* (de Medioni et ses collègues de

---

*University of Southern California*) appliquées pour segmenter des nuages de points (acquis par capteur laser aéroporté par exemple [Schuster 2004]) ainsi qu'à des techniques plus simplistes mais un peu plus rapides comme celle proposée par Rabbani et van den Heuvel [Rabbani et van den Heuvel 2005] (mais initialement proposée par Vosselman dans [Vosselman et al. 2004-2]). Cette dernière méthode consiste à projeter l'orientation du vecteur normal de chaque point dans un espace de type Hough pour détecter les points alignés sur un même plan ou sur un cylindre. Ces deux méthodes présentent l'avantage d'avoir été développées spécifiquement pour traiter des nuages de points, sans recourir à des structures trop évoluées. Mais la première a un temps d'exécution qui augmente très rapidement avec le nombre de points (l'application *TensorVoting* est téléchargeable en ligne<sup>1</sup>), et la méthode de Vosselman et Rabbani ne semble marcher - là encore - que sur des scènes relativement simples et propres (elle a pourtant été développée pour traiter des nuages en intérieur d'usine, avec de nombreux tuyaux entremêlés).

### 3.1.2 Techniques issues du traitement d'image classique

Le traitement d'image classique nous fournit un nombre important de méthodes de segmentation plus éprouvées (et aussi plus étudiées) que les méthodes réellement 3D développées spécifiquement pour les nuages de points. Elles tirent parti bien sûr de la régularité spatiale des images et de la notion de voisinage simple fournie par la structure de grille (aussi bien en 2D qu'en 3D). On peut donc espérer en extraire des techniques robustes ou ayant un fort potentiel pour résoudre notre problème, au prix de certaines adaptations (soit des données, soit de la méthode). On sépare en général les techniques de segmentation en deux familles : les approches par régions (*region-based*) et les approches par contours (*edge-based*).

#### Approches par régions

Les approches par régions consistent à regrouper des points selon des critères de similarité et/ou de proximité. Ce regroupement peut se faire de manière assez simpliste par des méthodes proches de la classification en considérant l'histogramme de l'image (seuillage bayésien, seuillage de Neymann-Pearson, classification, régularisation markovienne, optimisation, etc.). Le problème avec ces méthodes est généralement l'absence de prise en compte de la localisation de l'information. On peut appliquer des méthodes un peu plus évoluées qui vont regrouper les pixels progressivement, de proche en proche, soit par "croissance/partage/réunion de régions" (méthodes dites par *transformation de régions*), soit par des raisonnements multi-échelles. Enfin, on peut évoquer l'existence de méthodes annexes à base de règles, qui sont un peu à cheval entre approches par régions et approches par contours (MDL - *Minimum Description Length* et l'approche de *Mumford et Shah* et ses variantes). Ces deux derniers types d'approches par régions semblent

---

<sup>1</sup><http://iris.usc.edu/tensorvt>



complexes à mettre en oeuvre et surtout peu adaptables au problème de la segmentation d'un nuage de points.

Nous présenterons en section 3.2.3 plusieurs méthodes de segmentation par classification, dont en particulier une se basant sur des considérations statistiques locales, qui pourront être rangées dans la première catégorie des approches par régions.

### Approches par contours

Les approches par contours consistent comme leur nom l'indique à détecter des contours (inflexions de la géométrie ou de la teinte) délimitant les zones à segmenter. Elles se basent principalement sur des filtres appliqués localement à l'image à travers des voisinages de pixels de taille limitée (gradient, gradient morphologique, laplacien, Kirsch, Canny, Shen, Deriche, etc.). L'application de ces filtres est alors suivie d'opérations morphologiques de fermeture des contours détectés, ou de seuillage par hystérésis, etc. Il existe des méthodes applicables au nuage de points laser qui s'inspirent directement de ces techniques en exploitant la structure  $2D^{1/2}$  des cartes de profondeurs associées à chaque point de vue d'un relevé laser (Cf. section 2.3.3). D'après [Chaperon 2002], elles sont généralement très rapides mais sont aussi réputées peu robustes. Il ne paraît donc pas très pertinent de tenter de les utiliser pour notre propre problème, d'autant que la donnée des cartes de profondeurs n'est pas toujours disponible (en particulier si on ne traite pas un nuage de points issu d'un scanner laser).

Il existe néanmoins une famille d'approches par contours très importante et qui ne se base pas sur l'application d'un filtre local : les *contours actifs*. Cette famille de méthodes a été introduite par Kass, Witkins et Terzopoulos [Kass et al. 1987]. Elle repose sur le principe d'adapter la forme et la position d'un contour pré-initialisé pour qu'il épouse au mieux la forme de la zone que l'on cherche à segmenter. L'évolution du contour est en général contrainte par différentes forces (des forces propres au contour, des forces appliquées par le milieu - ici l'image - et enfin des forces externes pouvant traduire des contraintes spécifiques au problème). Le contour peut-être exprimé de manière paramétrique (cas des *snakes*) ou de manière implicite en passant par la définition d'une fonctionnelle sur tout l'espace dont le contour serait une ligne de niveau (cas des *level sets*). On peut aussi calculer l'influence de l'image sur le contour de manière plus ou moins simple (à partir du simple gradient de l'image jusqu'au champ de déplacement appelé GVF pour *Gradient Vector Flow* [Xu et Prince 1997]). Nous ne rentrerons pas ici plus dans les détails de cette vaste famille de méthodes de segmentation/reconstruction. Nous pouvons par contre évoquer le fait que ce sont des méthodes qui profitent d'une forte dynamique de recherche, en particulier dans le domaine du traitement d'images médicales 3D. Elles présentent un très fort potentiel et semblent en pratique de bonnes candidates à une adaptation au cas des nuages de points.

Cette avis semble d'ailleurs être soutenu par l'utilisation d'un algorithme dénommé

---

*Fast Marching* dans un certain nombre d'articles présentant des traitements appliqués à des nuages de points. C'est une technique numérique de propagation géodésique d'un ou plusieurs fronts sur une surface qui se base sur le principe des *level sets*. Nous présenterons ainsi en section 3.3.2 un algorithme de segmentation d'un nuage de points valués basé sur l'algorithme du *Fast Marching*. Cet algorithme pourra être considéré comme une technique de segmentation à mi-chemin entre la croissance de régions et les contours actifs.

### Le cas particulier du *watershed*

Il reste un dernier type de méthode de segmentation qui ne rentre pas vraiment dans les cases "segmentation par régions" ou "segmentation par contours" (bien qu'il soit parfois classé dans la première catégorie). C'est l'algorithme phare de segmentation en morphologie mathématique : la ligne de partage des eaux (plus communément appelé *watershed* qui est le terme équivalent en anglais). Cet algorithme a été introduit par [Beucher et Lantuejoul 1979]. L'idée principale sous-jacente est de considérer une image en niveaux de gris (ou plus généralement la norme du gradient de l'image) comme un relief. Le processus de segmentation consiste alors à inonder ce relief progressivement en partant des minima. Lorsque deux *bassins versants* se rencontrent, une *ligne de partage des eaux* est créée, et l'ensemble de ces lignes définit alors les contours. Il existe différentes manières de réaliser cette inondation (montée/descente des eaux, immersion - Cf. [Vincent et Soille 1991] - etc.) et différentes manières de gérer l'apparition des bassins versants et des contours correspondants (principalement des méthodes hiérarchiques ou non).

Pour notre problème particulier, l'article [Page et al. 2003] (basé lui aussi sur l'algorithme de *Fast Marching*) à défaut de nous donner des solutions pour segmenter un nuage de points (il semble qu'aucun algorithme de *watershed* spécifique aux nuages de points n'ait été développé), nous a permis de nous rendre compte que l'approche par ligne de partage des eaux est finalement proche de celle que nous proposons en section 3.3.2. C'est principalement les conditions d'arrêt de la propagation qui sont différentes entre les deux méthodes.

En pratique, nos tests de l'algorithme de *watershed* adapté au cas des nuages de points valués ont fait apparaître une très forte tendance à sur-segmenter (même en minimisant au maximum le nombre de "bassins versants" en jouant sur l'étape de détection des minima locaux). Ce problème particulier des approches de type *watershed* est bien connu, et la meilleure solution pratique semble être de fusionner les bassins versants qui se mélangent les uns aux autres soit automatiquement (voir figure 3.1), soit en ajoutant une étape manuelle dite de "marquage" pour plus de robustesse (par exemple : un trait continu dessiné à la souris et qui recouvre grossièrement les zones que l'utilisateur sait communes). Une telle étape, même en supposant qu'elle fonctionne parfaitement, ne semble pas très adaptée dans notre cas, puisque l'utilisateur a justement beaucoup de mal à repérer les zones d'intérêt et à percevoir leur forme.

---

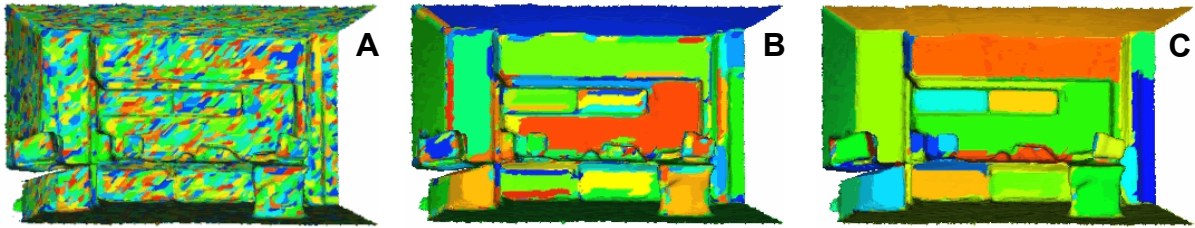


FIG. 3.1 – Illustration de l’algorithme de segmentation géométrique d’un modèle 3D à base de watershed proposé par [Sun et al. 2002].

*L’image A représente le résultat initial de l’algorithme de watershed, et les images B et C représentent différentes phases de regroupement des zones sur-segmentées.*

Nous avons donc préféré approfondir l’approche par propagation de contour. Néanmoins, il existe des liens indéniables entre les deux algorithmes, en particulier au niveau du principe de la propagation contrainte par la norme du gradient.

## 3.2 Segmentation par classification

On peut diviser le processus de segmentation tel qu’énoncé en début de chapitre en deux phases : une première phase de séparation des points en deux groupes (d’une part les points "immobiles/inchangés", et d’autre part les points ayant subi un changement) et une seconde phase de regroupement des points à partir de critères de proximité pour former des sous-groupes cohérents.

La première phase correspond à un processus de classification. Nous allons donc considérer dans cette section trois méthodes de classification : deux méthodes dites "globales" (nous reviendrons sur ce concept en temps voulu), très simples à mettre en oeuvre, dont une manuelle et une automatique, et une troisième méthode "locale" et automatique, certes plus évoluée mais beaucoup plus satisfaisante. Enfin nous proposerons une méthode de regroupement des points sous forme de *composantes connexes*.

### 3.2.1 Classification globale et manuelle par seuillage

Pour diviser un ensemble de valeurs scalaires, une solution très simple est de les ordonner puis de fixer un seuil séparant l’ensemble en deux parties. Dans notre cas, les points peuvent être ordonnés selon leur valeur d’écart associée (dans l’ordre croissant) et on peut fixer un seuil compris entre la valeur minimale et la valeur maximale, qui séparera ainsi celui-ci en deux parties : les points "inchangés" en dessous de cette valeur de seuil et les autres, au dessus.

En gardant toujours une approche très simpliste, et en profitant du fait que l'on se place dans un cadre semi-automatique, il est possible de fixer cette valeur manuellement. L'utilisateur, en jouant sur un élément d'interface graphique, peut faire varier ce seuil tout en ayant un retour visuel en temps réel lui permettant d'évaluer directement le résultat de la classification. Une fois qu'il a trouvé une valeur visuellement satisfaisante (c.à.d. que la frontière entre points "inchangés" et points "changés" semble bien positionnée), l'utilisateur peut valider cette séparation et poursuivre le processus.

Remarque : l'interface du logiciel de démonstration de cette thèse (*CloudCompare*) permet de définir manuellement deux seuils de coupure et donc potentiellement trois groupes (voir figure 3.2). Mais pour cette application particulière, le second seuil devrait être laissé égal à la valeur maximale des écarts du nuage. L'élément d'interface présenté ici permet aussi la définition de deux seuils relatifs à la coloration des points, ce qui peut aider l'utilisateur à positionner les seuils de coupure mais n'est pas non plus un pré-requis pour l'application présentée ici.

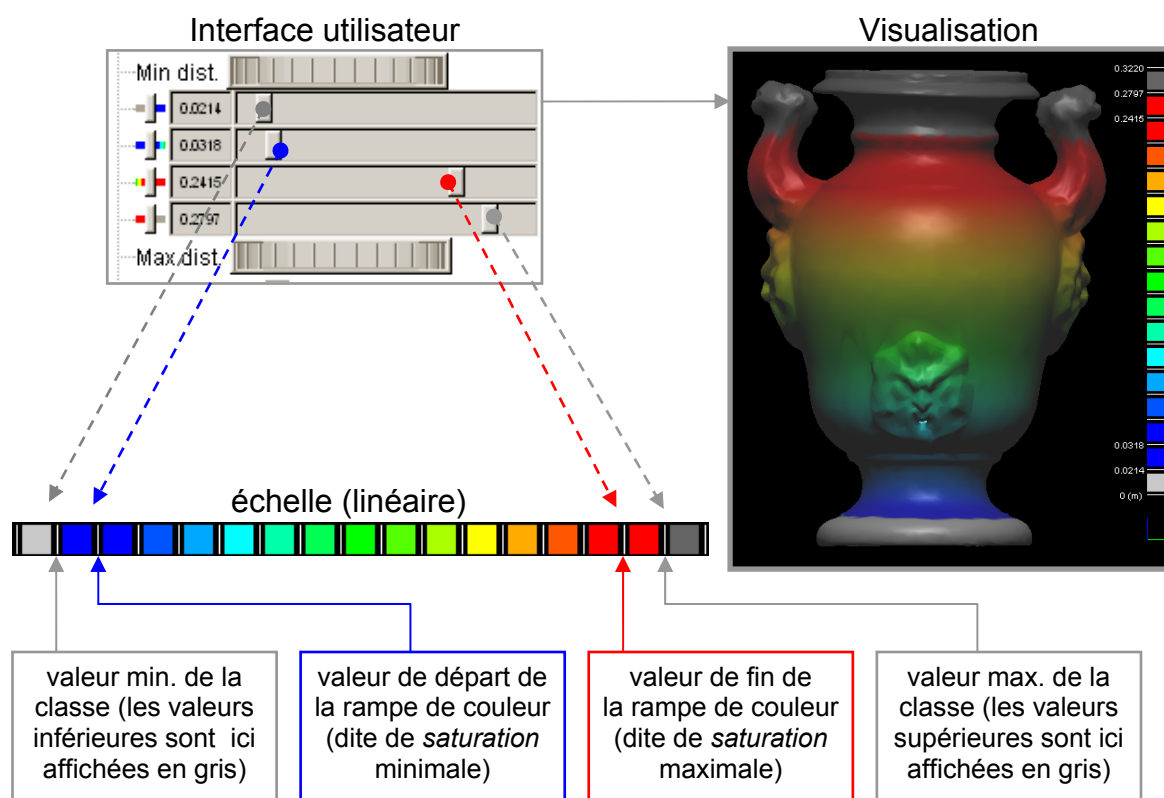


FIG. 3.2 – Illustration du système d'interface graphique permettant la définition interactive de seuils de coupure (et de répartition des couleurs) sur un champ scalaire surfacique. Le champ scalaire est ici représenté en fausses couleurs allant du bleu au rouge selon un ordre croissant pour les valeurs situées entre les deux seuils. Les autres points apparaissent en gris.

On qualifie ce processus de "global" car un seuil unique est fixé pour tout le nuage et la coupure ainsi opérée ne prend donc pas en compte les tailles relatives des différents objets représentés par le nuage. En fixant un seuil trop haut, on risque de "rater" des petits objets ayant subi un changement (et qui vont donc se retrouver dans le groupe des points "inchangés" - voir figure 3.3), et en fixant un seuil trop bas on va garder trop de points, rendant la classification et la suite du processus inopérante. Enfin, et de manière plus générale, le processus ne prend pas en compte les variations locales de précision du champ scalaire (le bruit de mesure par exemple) et il n'est pas robuste à la présence de valeurs aberrantes.

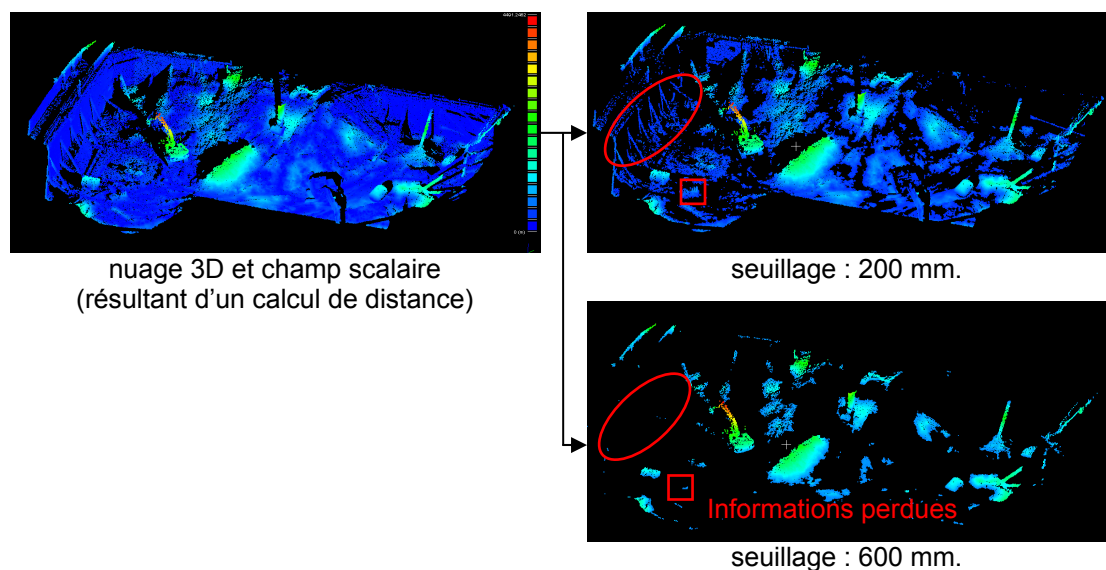


FIG. 3.3 – Illustration de la segmentation globale par seuillage manuel sur les valeurs de déplacement.

*Un seuil élevé donne un résultat plus clair, mais des informations potentiellement importantes sont supprimées (des poutres et un groupe électrogène, encadrés en rouge ici).*

Cette approche a donc trois défauts évidents : la nécessité de l'intervention humaine ralentit et alourdit le processus, la valeur fixée manuellement est généralement subjective et imprécise, enfin et surtout elle est globale. Elle possède tout de même deux points forts. Tout d'abord, elle s'est avérée très pratique et d'une grande facilité de mise en oeuvre. L'opération est très intuitive et assez rapide, ce qui permet à l'utilisateur de la répéter plusieurs fois avec des seuils différents pour pallier son caractère "global". Mais le véritable intérêt a été l'amélioration de la perception de la géométrie et des limites des zones déformées grâce au côté dynamique et interactif de cette méthode.

En effet, la perception de la troisième dimension sur un écran en deux dimensions est de façon générale un problème important. Mais ce problème se fait particulièrement

ressentir lorsque l'on travaille avec des nuages de points. Ceux-ci sont par nature échantillonnés de manière assez irrégulière une fois projetés dans l'espace écran. Ils sont souvent dépourvus de couleur ou toute autre information de teinte réaliste (éclairage, etc.) et ils ne représentent les surfaces des objets que de manière partielle.

On peut bien sûr grandement améliorer notre perception de la forme de ces nuages en colorant artificiellement les points (en fonction de l'altitude, ou de la valeur de distance qui leur est associée - comme cela est le cas pour une bonne partie des illustrations de ce manuscrit). Mais pour autant, les détails, les formes locales et surtout les limites des zones de déformation restent difficilement perceptibles<sup>2</sup>. Au même titre que changer le point de vue ou jouer sur la parallaxe améliore grandement la perception d'une forme tridimensionnelle en général, régler interactivement la coloration des points ou le seuil de coupure a un effet très bénéfique sur la perception des formes et de leurs limites. Ceci est d'autant plus vrai que dans notre cas les valeurs associées sont des déplacements qui sont plus ou moins liés à la forme des objets.

### 3.2.2 Classification globale et automatique par les K-moyennes

En gardant une approche de classification globale, il est possible d'automatiser la détermination du seuil de coupure dans une certaine mesure. Ceci peut être fait en appliquant au champ scalaire un algorithme de classification usuel dénommé "K-means", ou "K-moyennes" en français (qui est un cas particulier de l'algorithme plus général des "nuées dynamiques").

C'est un algorithme de classification non supervisé d'un ensemble  $E$  en  $k$  classes,  $k$  étant un entier naturel fixé (non nul) et  $E$  étant muni d'une notion de distance entre ses éléments. Cet algorithme est généralement appliqué à des cas où les éléments de  $E$  sont vectoriels (et ont donc plus plusieurs composantes) mais il peut également être utilisé pour le cas où, comme ici, les éléments de  $E$  sont scalaires.

L'algorithme consiste sommairement à regrouper les différents éléments de l'ensemble  $E$  autour de  $k$  centres, chaque élément de  $E$  étant associé au centre le plus proche, et chaque centre étant égal au centre de gravité (ici à la moyenne) des éléments qui lui sont associés. On regroupe ainsi les points en "amas" selon leur proximité relative. La position des centres d'amas est déterminée automatiquement par un processus itératif. C'est en somme un processus itératif de minimisation de l'inertie (la variance dans notre cas) intra-classe.

On donne une version de l'algorithme en pseudo-code (Cf. algorithme 3.2.2). On peut

---

<sup>2</sup>Notons que la perception des limites est de toute façon une notion subjective lorsque l'on travaille avec une représentation échantillonnée d'un ensemble d'objets connectés les uns aux autres et composés de nombreuses sous-parties. Tout dépend donc de la granularité des composants que l'on considère ainsi que de l'échelle à laquelle se place l'analyse.

---

remarquer que l'algorithme nécessite une initialisation des centres d'amas. Ceci permet tout simplement de pré-positionner ces centres si on dispose d'une méthode qui permet de déterminer approximativement leur position (pour ainsi accélérer la convergence de l'algorithme). Autrement, on peut tout simplement espacer les centres régulièrement sur l'intervalle des valeurs possibles par exemple.

---

**Algorithme 4** Algorithme des K-moyennes

---

**Entrées:**  $E : \{x_i, 0 < i \leq N\}$  un ensemble de  $N$  éléments  $x_i$  et  $P_k : \{p_j, 0 < j \leq k\}$ , les positions initiales des  $k$  centres d'amas (qui peuvent être quelconques mais ne doivent pas être égales deux à deux)

**Sorties:**  $C_k : \{c_j, 0 < j \leq k\}$ , les positions finales des centres d'amas

$C_k \leftarrow \{0\}$

**tant que**  $C_k \neq P_k$  **faire**

$C_k \leftarrow P_k$

**pour**  $0 < i \leq N$  **faire**

**pour**  $0 < j \leq k$  **faire**

calculer la distance entre  $x_i$  et  $p_j$

**fin pour**

associer  $x_i$  avec le centre  $p_j$  le plus proche

**fin pour**

**pour**  $0 < j \leq k$  **faire**

recalculer le centre  $p_j$  avec les nouvelles associations

**fin pour**

**fin tant que**

renvoie  $C_k$  (et les associations entre points et centres)

---

Lorsqu'on applique l'algorithme des K-moyennes avec deux classes sur le champ scalaire associé au nuage (la *distance* entre les éléments et les centres d'amas est alors simplement la valeur absolue), on obtient ainsi un processus de classification automatique simple et rapide, qui nous donnera par construction deux classes disjointes (dont l'une regroupera les valeurs les plus faibles, et l'autre les valeurs les plus fortes). Si le nuage est composé d'un nombre important (largement majoritaire) de points "inchangés", alors on peut deviner que la première classe sera composée de cette très grande majorité (avec un *centre* - c'est à dire une moyenne - proche de zéro) et l'autre classe regroupera les autres points qui ont une dispersion plus importante. Le *centre* de cette seconde classe correspondra à une moyenne des écarts supérieurs à zéro, pondérée par la taille des objets concernés (mais cela dépend bien sûr plus finement de la forme de ces objets et de la couverture et de la densité du nuage). En fait, la position précise des centres de classes n'est pas une information pertinente ici, seule la division automatique du nuage en des deux classes cohérentes nous intéresse.

En pratique, cet algorithme donne généralement des résultats satisfaisants (voir figure 3.4), car les nuages de points sont souvent composés d'un nombre de points "changés" minoritaire. Le seuil fixé est par contre là encore global (voir la section précédente

---

pour tout ce que cela implique) et il reste relativement imprécis car il dépend tout de même, bien que dans une faible mesure, de la répartition des valeurs d'écart sur tout le nuage (si l'on rajoute ou on enlève des points, cela va faire varier très légèrement la position des centres et donc la frontière entre les classes). C'est typiquement un très bon moyen de chercher une première approximation du seuil global de coupure (pour accélérer l'opération manuelle de l'utilisateur évoquée dans la section précédente, par exemple).

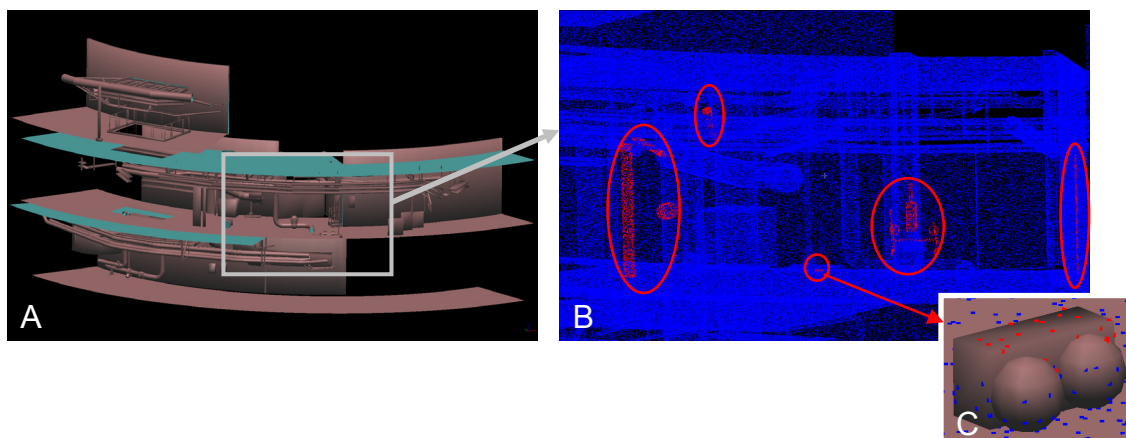


FIG. 3.4 – Illustration du comportement de l'algorithme des K-moyennes. *Comparaison d'un modèle 3D (A) à un nuage de points comportant quelques différences (des tuyaux manquant principalement). Résultat de la classification automatique (B) et détail (C) montrant que le seuil déterminé est suffisamment précis pour classer correctement des points correspondant à des objets très petits (à condition que le rapport points "changés" / points "inchangés" soit faible).*

### 3.2.3 Classification par analyse statistique locale

On a vu dans les sections précédentes qu'il était possible de fixer par des moyens simples un seuil unique, et donc global, pour séparer rapidement le nuage de points en deux classes. On s'est par contre aussi rendu compte que le fait que ce seuil soit global posait de nombreux problèmes, en particulier à cause des différences relatives de taille des objets déplacés ou déformés, mais aussi et finalement surtout parce que cela ne permet pas de prendre en compte une réalité essentielle des nuages de points obtenus par un quelconque procédé de numérisation tridimensionnelle : la précision des données et la variation spatiale de cette précision. Les points "inchangés" n'ont pas, en pratique, une valeur de déplacement parfaitement nulle, et l'erreur qui entache cette valeur n'est généralement pas constante sur tout le nuage (voir figure 3.5). Les raisons en sont multiples :

- Tout d'abord les valeurs d'écart associées au nuage proviennent d'un calcul numérique reposant sur un nombre important d'opérations sur des réels et le résultat est donc limité en précision par ces calculs ainsi que par l'expression numérique des



positions des points dans l'espace. Cette première source d'erreur est heureusement très faible (en général inférieure à  $10^{-6}$  en relatif) et elle est constante sur tout le nuage.

- On a aussi mis en avant en section 2.3.1 une erreur *quasi-poissonnienne* due à l'échantillonnage des nuages de points (dans le cas d'une comparaison directe sans modélisation locale entre deux nuages de points).
- Ensuite un nombre important d'erreurs proviennent des données elles-mêmes :
  - si on a affaire à un nuage de points, non seulement ceux-ci présentent des erreurs de positionnement qui dépendent du processus d'acquisition mais aussi du processus de consolidation des différentes parties du nuage (si celui-ci a été acquis depuis plusieurs points de vue). Les erreurs d'acquisition sont généralement anisotropes, et dépendent d'un nombre impressionnant de facteurs, dont une partie est intrinsèque (précision mécanique et électronique, température de l'appareil, aberrations des optiques, résolution du capteur C.C.D., etc.) et l'autre partie, potentiellement encore plus vaste, est extrinsèque (propriété de la surface, de l'éclairage, température extérieure, etc.). Plus généralement, la précision varie aussi directement avec la distance qui sépare la zone mesurée du capteur. Ce cumul d'erreurs fait que le bruit d'un capteur est loin d'être gaussien et spatialement homogène, et il variera même légèrement lors de chaque campagne de mesure.
  - si on a affaire à un modèle 3D, il peut provenir soit de plans théoriques, ce qui fait qu'il peut n'avoir aucun rapport avec la réalité (l'erreur est alors totalement imprévisible), soit d'un processus de reconstruction par rapport à des données réelles, auquel cas le modèle hérite des imprécisions qui entachent ces données auxquelles il faut rajouter des erreurs dues à la simplification de la réalité ainsi qu'au processus d'ajustement des primitives géométriques (généralement simples, comme des plans ou des cylindres) sur des morceaux du nuage de points, avec des méthodes reposant sur des optimisations aux moindres carrés.
- Enfin, dans le cas particulier des processus de calcul d'écarts entre données 3D tels qu'ils sont présentés dans le chapitre 2, les résultats ne sont pas signés. On observe donc en quelque sorte un repliement autour de zéro de l'histogramme des valeurs.

En pratique, vu le nombre de sources d'erreurs différentes qui s'accroissent selon des processus non forcément linéaires, et aussi vu le nombre de combinaisons possibles que l'on pourra rencontrer en comparant deux jeux de données (en terme de type et surtout de procédé de fabrication), il est impossible, dans le cas général, de pré-supposer la forme de la distribution statistique du bruit global qui entache la mesure des écarts.

On peut encore dire que, d'une part, le processus de classification ne devrait considérer comme "changés" que les points dont les valeurs d'écarts sont supérieures au bruit et, d'autre part, ses décisions devraient reposer sur des considérations locales. Notre idée a donc été de mettre au point un processus de test statistique local du champ scalaire associé à un nuage de points, dont le but est de filtrer les points dont la valeur d'écart est incluse dans le bruit. Ces points seront alors considérés par défaut comme "inchangés/immobiles".

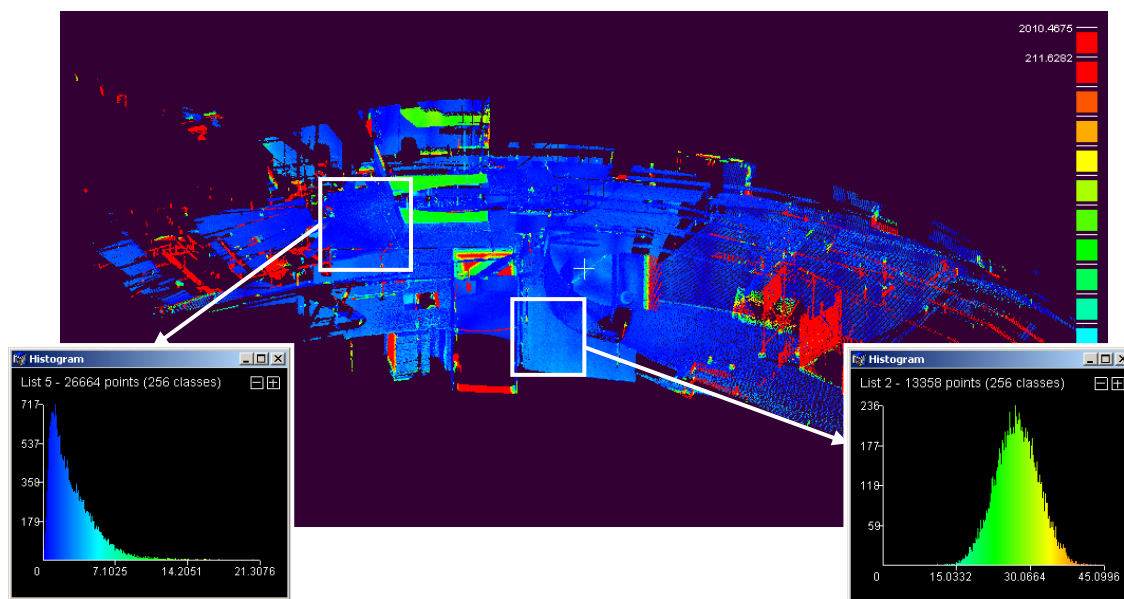


FIG. 3.5 – Illustration de la variation spatiale de l'erreur sur les écarts.  
*Nuage d'intérieur de centrale, comparé au modèle 3D T.Q.C. correspondant (non représenté ici), et histogrammes locaux des écarts sur des zones "inchangées".*

### Test statistique local du $\chi^2$

Cette technique de test statistique s'inspire très librement de l'article non publié de Hu et al. dénommé *Statistical 3D Segmentation With Greedy Connected Component Labelling Refinement* (téléchargeable en ligne sur la page web du laboratoire *Prism* de l'Arizona State University) et qui est lié à un autre article des mêmes auteurs [Hu et al. 2002] qui reprend certaines des idées proposées. L'article propose une méthode pour segmenter une image LSCM ("Multichannel Laser Scanning Confocal Microscopy" ou "Microscope Confocal à Balayage Laser" en français) qui est une image en 3 dimensions constituée de voxels<sup>3</sup>. La méthode consiste à regrouper les voxels voisins ayant des intensités proches pour former des sous-groupes dénommés "Connected Components" ("Composantes Connexes" en français). Nous utiliserons d'ailleurs nous-même ce concept, sous une autre forme, dans la section 3.2.4. Le principe de la méthode de Hu et al. est, dans une première passe, de calculer localement autour de chaque voxel les paramètres empiriques d'une distribution statistique paramétrique de paramètres  $(a, b)$  à partir d'un voisinage de taille  $n \times n \times n$  (avec  $n = 3$  ou  $5$  typiquement). Dans une seconde passe, les voxels voisins ayant des paramètres proches sont regroupés (cette étape n'est d'ailleurs pas très claire dans l'article, mais elle ne nous intéresse pas ici).

Notre méthode de test statistique s'inspire uniquement du concept de distribution sta-

---

<sup>3</sup>Un voxel, ou "volumetric pixel" est comme son nom l'indique un pixel en 3D, ce qui correspond plus explicitement à la cellule d'une grille 3D munie d'une valeur scalaire assimilable à une intensité.

tistique locale de l'intensité. L'article utilise comme distribution statistique paramétrique la loi de Weibull (du nom de l'ingénieur suédois qui l'a introduite, initialement pour des problèmes de calcul de fiabilité) qui s'avérera être très utile pour *représenter* le bruit global des écarts d'un nuage de points (Cf. section 3.2.3).

### Rappel : le test du $\chi^2$

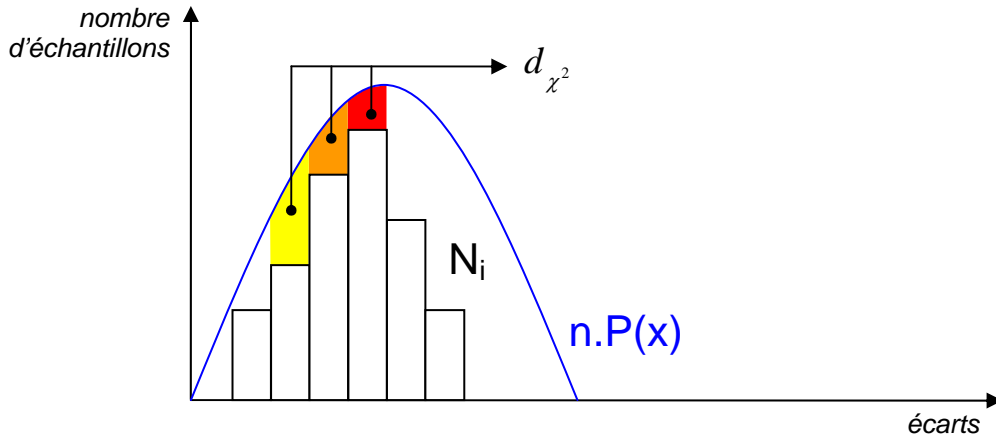


FIG. 3.6 – Illustration du calcul de la distance du  $\chi^2$  entre l'histogramme  $\{N_i\}$  correspondant à  $n$  échantillons et une distribution  $P(x)$ .

Soit  $n$  échantillons représentés par  $\{N_i, i \in [1, k]\}$  (un histogramme de  $k$  classes) et une distribution théorique  $P(x)$  (dans notre cas particulier,  $x \in [0, +\infty[$ ). Le test du  $\chi^2$  avec  $k - 1$  degrés de liberté, consiste à calculer la distance du  $\chi^2$  définie par :

$$d_k = \sum_{i=1}^k \frac{(N_i - nP_i)^2}{nP_i} \quad (3.1)$$

Remarque : ici, puisque l'on se place dans le cas continu,  $P_i$  est égal à la probabilité  $P$  cumulée sur tout l'intervalle correspondant à la  $i^{\text{ème}}$  classe de l'histogramme ( $P_i$  est donc égal à l'intégrale de la densité de probabilité  $P(x)$  sur cet intervalle).

La distance  $d_k$  est en quelque sorte la somme des carrés des différences d'aire par morceaux entre une distribution théorique - ici celle du bruit, donnée en entrée - et celle d'un échantillon - ici celui des écarts des points du voisinage - pondérées par l'aire de la distribution théorique sur chaque morceau (voir figure 3.6). Grossièrement, on peut dire que c'est une mesure de la déviation du réalisé par rapport à l'attendu.

Le test du  $\chi^2$  permet de confirmer ou d'infirmer l'hypothèse selon laquelle les données ne suivent pas la distribution théorique, et ce avec une certaine *marge d'erreur*. En

pratique, il consiste simplement à comparer la distance  $d_k$  à un seuil qui dépend à la fois du nombre de degrés de liberté du problème ( $k - 1$  ici) ainsi que d'une certaine marge d'erreur  $p$  ( $p < 1$ ).

Soit  $s_k^p$  ce seuil. Les valeurs de ce seuil sont couramment tabulées dans tout bon livre de statistiques ([Saporta 1990] par exemple) mais il est aussi possible de les déterminer numériquement de manière approchée à partir de la formule suivante :

$$p = 1 - [2^{k-1}\Gamma(\frac{d}{2})]^{-1} \int_{s_k^p}^{+\infty} t^{\frac{d}{2}-1} e^{-\frac{t}{2}} dt \quad (3.2)$$

avec  $\Gamma$ , la fonction gamma, définie par :

$$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt \quad (3.3)$$

Remarques :

- attention, le test du  $\chi^2$  ne nous dit pas que les données suivent la distribution théorique avec un seuil de confiance de  $1 - p$ . Elle permet uniquement d'affirmer (avec une marge d'erreur de  $p$ ) qu'elles ne suivent pas la distribution théorique. C'est une différence importante. Si la distance  $d_k$  calculée est supérieure à  $s_k^p$ , alors on peut seulement affirmer que les échantillons représentés par  $\{N_i\}$  ne suivent pas la distribution  $P$ , au risque de se tromper de  $100.p$  %.
- citant [Saporta 1990], une condition pour la bonne application du test du  $\chi^2$  est que lors du calcul de  $d_k$ ,  $nP_i$  soit supérieur à 5, quel que soit  $i$  (mais on peut tolérer 3 ou même 1 en queue de distribution d'après certains auteurs, ce qui est d'ailleurs nécessaire lorsque  $n$  est petit).

### Filtrage statistique local d'un nuage de points

Notre méthode nécessite comme paramètres d'entrée d'une part une distribution statistique quelconque dont les paramètres sont connus, et qui est censée représenter la distribution du bruit qui entache les données, et d'autre part la marge de confiance du test du  $\chi^2$ . Son principe général consiste alors simplement à effectuer localement un test du  $\chi^2$ , en considérant un voisinage autour de chaque point du nuage dont on va extraire un histogramme des écarts. D'après le rappel qui a été fait précédemment, le test du  $\chi^2$  nous donne alors une indication sur la non appartenance probable de la valeur en un point (ou plus exactement des valeurs dans son voisinage) au bruit. Ainsi, si la distance du  $\chi^2$  calculée pour le point est supérieure au seuil du  $\chi^2$  suivant la marge de confiance définie en entrée, alors le point est classé comme *changé* (et donc, par défaut, il sera *inchangé* dans le cas contraire). La marge de confiance permet, dans une certaine mesure, de prendre en compte les variations spatiales du bruit.

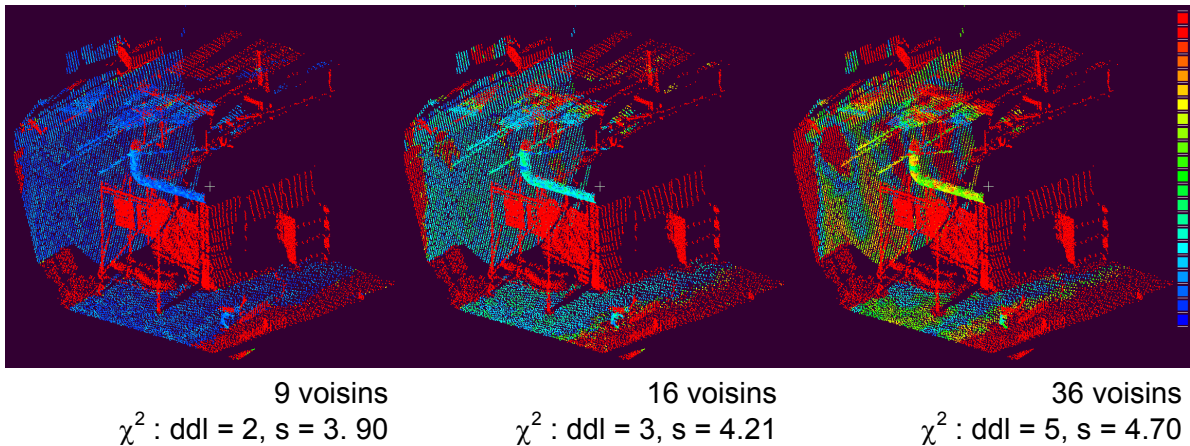


FIG. 3.7 – Illustration de l'influence de la taille du voisinage sur le test statistique local. Les points rouges sont les points "changés" (au dessus du seuil du  $\chi^2$ ). Les autres ont des couleurs proportionnelles à leur distance du  $\chi^2$ , allant du bleu au rouge. On peut voir que lorsque le voisinage augmente, la distance du  $\chi^2$  est plus pénalisante (relativement au seuil) et on observe donc un léger épaissement des bordures des zones rouges, ainsi qu'une augmentation du contraste dans les zones "inchangées".  
(ddl = degré de liberté)

Un dernier paramètre de l'algorithme, mais qui n'a pas besoin d'être défini par l'utilisateur, est la taille du voisinage. Celui-ci peut être défini comme une sphère de voisinage (pour éviter tout problème local de densité du nuage), le rayon de la sphère dépendant alors de la densité moyenne du nuage par exemple. Mais on peut aussi, par simplicité, ne considérer comme voisinage que les  $n$  plus proches voisins. On calcule l'histogramme avec  $k = \sqrt{n}$  classes (qui est le nombre maximum théorique de classes).

La sensibilité de l'algorithme vis-à-vis de la taille du voisinage n'est pas très forte (voir figure 3.7). Néanmoins, plus le voisinage est grand, et plus l'algorithme sera "sévère", dans le sens où il classera plus de points comme étant *changés* (en effet, plus il y a de points de distance forte dans le voisinage et plus la distance du  $\chi^2$  va diverger). Il sera aussi plus lent, puisque la taille du voisinage à déterminer pour chaque point est plus importante. Il sera par contre plus robuste, puisque l'histogramme est calculé avec plus de valeurs. Une bonne idée est donc d'appliquer un premier filtrage rapide, avec peu de voisins, à tout le nuage, puis de ré-appliquer aux points proches de la limite du  $\chi^2$ , et donc potentiellement mal classés, un test prenant en compte un voisinage plus important. Ainsi, on assure une meilleure définition de la frontière et on limite efficacement le temps de calcul supplémentaire nécessaire.

En pratique la méthode est assez rapide et son temps d'exécution dépend principalement de la capacité du programme à extraire des voisinages autour de chaque point rapidement (le test du  $\chi^2$  est totalement négligeable, en terme de complexité, par rapport à la détermination des voisinages). Ses résultats sont aussi excellents, à condition d'avoir

réussi à définir correctement la distribution statistique moyenne du bruit. Or nous avons vu que sa connaissance a priori dans le cas de nuages de points réels est peu probable. Nous présentons donc maintenant une distribution statistique paramétrique dite "de Weibull", qui présente de bonnes propriétés et s'adapte bien aux bruits non gaussiens. Il existe aussi une méthode permettant de calculer ses paramètres à partir d'un échantillon de données, permettant ainsi d'initialiser correctement le test statistique, via la sélection préalable d'une zone caractéristique (des points "inchangés", dans notre cas).

### La distribution de Weibull

La distribution de Weibull<sup>4</sup> est une loi de probabilité continue à deux paramètres ( $a, b$ ) plus un paramètre de translation  $x_0$  (que nous considérerons nul dans la suite) et dont une des formes est :

$$P(x) = \frac{a}{b} \left(\frac{x - x_0}{b}\right)^{a-1} \exp\left[-\left(\frac{x - x_0}{b}\right)^a\right] \quad (3.4)$$

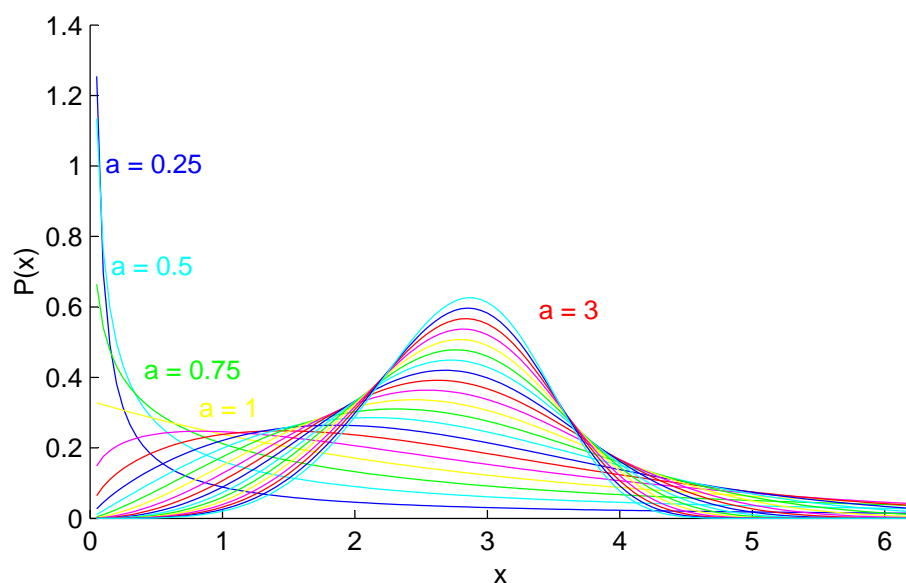


FIG. 3.8 – Distribution de  $Weibull(a, b)$  pour différentes valeurs de  $a$  ( $b = 3.0$ ).

La figure 3.8 montre des tracés de cette distribution pour différentes valeurs du paramètre  $a$ . Cette distribution présente comme intérêt particulier le fait qu'elle se confond avec des lois plus classiques pour certaines valeurs du paramètre  $a$  : pour  $a = 1$  elle est équivalente à une distribution de Poisson, pour  $a = 2$ , à une distribution de Rayleigh, et enfin pour  $a = 3$ , à une distribution Normale (Gaussienne). Ainsi, comme l'illustre la figure 3.9, elle s'adapte mieux à des distributions de bruit "réalistes" et potentiellement plus complexes qu'une loi simple (Normale, Poisson, etc.) qui sera trop rigide.

<sup>4</sup>Ingénieur et mathématicien suédois (1887-1979)

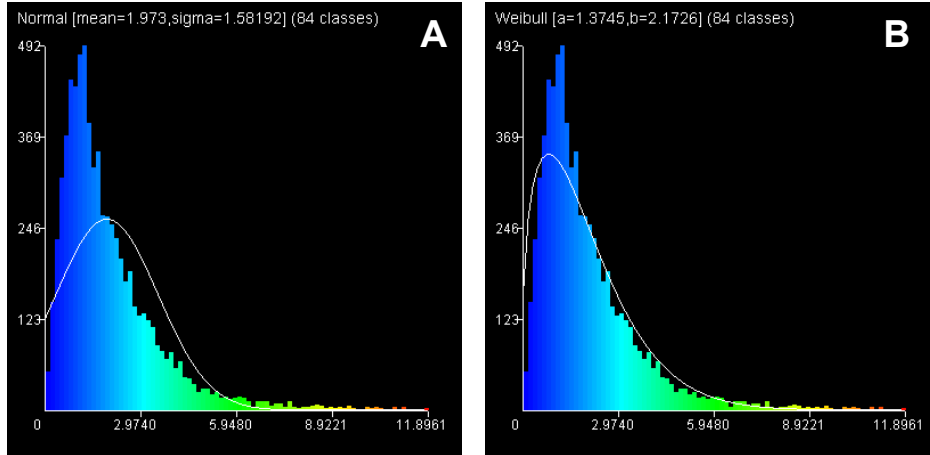


FIG. 3.9 – Comparaison de l'adaptation d'une loi normale (A) et d'une loi de Weibull (B) à un même échantillon de points.

*Un test du  $\chi^2$  entre chaque distribution et l'échantillon peut nous aider à quantifier l'adéquation dans chaque cas :  $d_{\chi^2}(A) = 4285.29$  contre  $d_{\chi^2}(B) = 900.63$ .*

De plus, [Hirose 1996] propose une méthode de détermination numérique de ses paramètres à partir d'un ensemble d'échantillons. Lorsque l'on veut appliquer le filtrage statistique local à un nuage dont on ne connaît pas à priori la distribution du bruit, on peut donc calculer les paramètres d'une loi de Weibull (ou d'une autre distribution paramétrique, à condition de disposer d'une méthode numérique d'estimation des paramètres) à partir d'un échantillon de points caractéristique d'une zone repérée comme "inchangée" (ou "immobile") et dont les écarts ne seront donc constitués que par du bruit.

En pratique, cette opération peut-être très aisément réalisée par l'opérateur. Celui-ci, avant de lancer l'opération de filtrage, découpe une zone du nuage dont les écarts sont quasiment nuls et présentent un bruit caractéristique du nuage (la forme des surfaces représentées par les points peut-être quelconque - seule la distribution des valeurs des écarts compte). Le programme peut alors extraire automatiquement les paramètres de la distribution qui s'adapte au mieux à cet échantillon, puis appliquer le filtrage avec ces paramètres.

Enfin, il est possible de calculer une moyenne et une variance pour une loi de Weibull à partir de ses paramètres  $a$  et  $b$ . Ceci peut permettre à l'utilisateur de mieux comprendre les résultats de la mesure des paramètres à partir d'un échantillon. Soit  $\mu$  la moyenne,  $\sigma^2$  la variance, et  $\Gamma$  la fonction gamma (équation 3.3). On a :

$$\mu = b\Gamma(1 + a^{-1}) \quad (3.5)$$

$$\sigma^2 = b^2[\Gamma(1 + 2a^{-1}) - \Gamma^2(1 + a^{-1})] \quad (3.6)$$

Remarque : il apparaît en pratique que l'étape de détermination des paramètres peut être évitée si l'on compare des données qui proviennent d'un même appareil de mesure utilisé

dans des conditions toujours à peu près similaires. C'est le cas par exemple d'un scanner laser utilisé en intérieur de centrale par les topographes d'EDF : on utilise toujours le même appareil, dans des environnements semblables (même éclairage, mêmes types de surfaces, mêmes couleurs, etc.) et il est de plus opéré par les mêmes personnes. Dans ce cas, la variation de l'erreur apparaît être très faible et une fois de bons paramètres trouvés, ceux-ci peuvent être ré-utilisés lors de chaque processus de comparaison.

### Discussion sur le seuil du $\chi^2$

En pratique, même avec une loi de Weibull qui s'adapte mieux aux données que des lois plus rigides (comme on l'a vu à la section précédente), l'adéquation entre la distribution théorique du bruit et la distribution réelle n'est pas parfaite. Sur certaines zones d'un nuage que l'utilisateur considérerait comme "inchangées", les distances du  $\chi^2$  seront donc parfois supérieures au seuil théorique du  $\chi^2$  (même pour une marge de confiance très faible). On peut par exemple remarquer que dans l'exemple de la figure 3.9, le résultat du test du  $\chi^2$  pour la distribution de Weibull, bien que meilleur que celui de la distribution Normale, reste très supérieur aux valeurs tabulées (qui sont de l'ordre de 50 à 80 dans ce cas là, en fonction du degré de confiance). On a de plus dit précédemment que la marge d'erreur du test permet de prendre en compte partiellement les variations locales de l'erreur. Mais cette prise en compte n'est que partielle et le test s'avère donc parfois trop "sévère" (en particulier en cas de fortes variations spatiales du bruit). On entre bien sûr dans un domaine subjectif (la *sévérité* de l'algorithme, ce que l'utilisateur *considérerait*, etc.), alors que le test du  $\chi^2$ , bien que statistique, ne l'est pas du tout. Il est tout de même possible de rendre la méthode plus souple, en jouant sur la manière de calculer la distance du  $\chi^2$ .

Lors du calcul de la distance du  $\chi^2$ , on est censé s'assurer que les  $nP_i$  sont tous au moins supérieurs à 1 (voire 3, et même 5 en général). Cette condition théorique se justifie par le fait que s'il existe un  $i$  tel que  $nP_i$  est très petit, alors le rapport  $\frac{(N_i - nP_i)^2}{nP_i}$  (qui est la contribution de la  $i^{me}$  classe à  $d_k$ ) est très grand et la distance du  $\chi^2$  va rapidement diverger. Pour respecter cette condition il faut en pratique soit fusionner certaines classes voisines (sachant que le nombre minimal de classes est 2) et donc baisser au passage le nombre de degrés de liberté, soit positionner directement les limites des  $k$  classes en fonction de la distribution théorique. On devine par contre que cette condition implique que tous les voisinages dont les écarts sont largement supérieurs au bruit auront des distances du  $\chi^2$  identiques (puisque le nombre de voisin est fixe, et que leurs valeurs s'accumuleront toutes dans la même classe de l'histogramme, celle qui est située à l'extrême droite).

Tout ceci est bien sûr cohérent du point de vue du test du  $\chi^2$  dont le but est simplement d'infirmer ou confirmer une hypothèse, et non de quantifier précisément l'éloignement des échantillons par rapport à la distribution théorique. Une conséquence est néanmoins que tous les points ayant des écarts supérieurs au bruit auront tous la même distance du  $\chi^2$ . Il sera alors difficile pour l'utilisateur de raffiner la segmentation en jouant sur le seuil de



coupure.

On pourra donc préférer travailler avec des distances du  $\chi^2$  divergentes, en ne respectant volontairement pas la condition sur la valeur minimale des  $nP_i$  (ou plutôt en abaissant très fortement cette valeur minimale, pour s'assurer simplement que les calculs numériques ne divergeront pas dans les cas extrêmes - voir la figure 3.10 qui illustre un cas où  $nP_i \simeq 0, \forall i \in [1, k]$ ). Cela ne modifie pas le résultat du test du  $\chi^2$ , et permet d'avoir des valeurs de  $d_k$  différentes sur un plus grand intervalle, offrant à l'utilisateur un marge plus importante pour corriger si besoin la position du seuil de coupure.

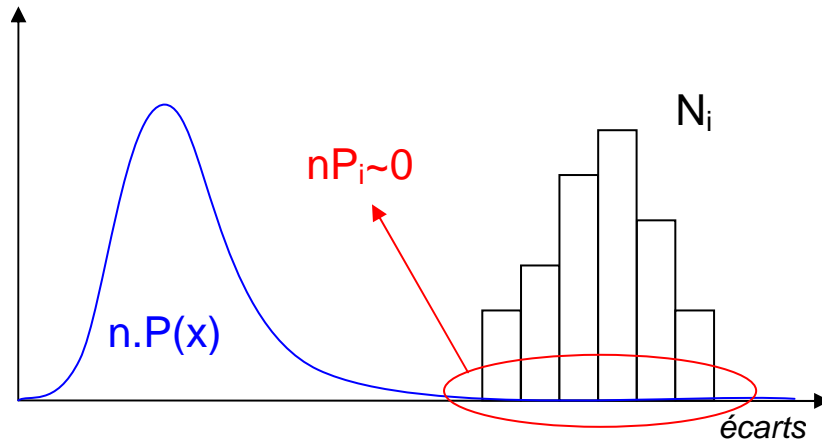


FIG. 3.10 – Illustration de la cause de divergence du calcul de la distance du  $\chi^2$ , lorsque les échantillons (et les classes de l’histogramme correspondant) sont loin de la moyenne de la distribution théorique.

En pratique, même en repoussant fortement la limite de saturation du calcul de la distance du  $\chi^2$ , des écarts de quelques centaines voire quelques milliers d’unités sur le seuil de coupure n’ont quasiment aucun impact sur la classification et concerneront un nombre de points très faible qui seront situés sur les bordures des zones de changement (car la distance ainsi calculée diverge très vite). Ainsi l’utilisateur dispose d’une plus grande marge de manoeuvre pour définir le seuil de coupure, sans pour autant risquer de dénaturer totalement le résultat de la classification. Ceci permet aussi d’utiliser comme seuil de coupure la valeur empirique trouvée lors de la détermination des paramètres (la distance du  $\chi^2$  entre le modèle déterminé et l’échantillon) plutôt que la valeur tabulée.

L’interface de *CloudCompare* (logiciel de démonstration des algorithmes de cette thèse), propose par défaut le seuil du  $\chi^2$  tabulé comme seuil de coupure. Mais l’utilisateur peut le modifier manuellement et voir le résultat de la modification s’afficher directement à l’écran, comme dans le cas du seuillage manuel (voir section 3.2.1), mis à part que les valeurs segmentées ne sont plus les écarts du nuage mais les distances du  $\chi^2$  calculées lors du test statistique local. En pratique, la modification du seuil influe donc très peu sur les résultats de la classification, et concerne surtout des zones mal scannées ou relativement

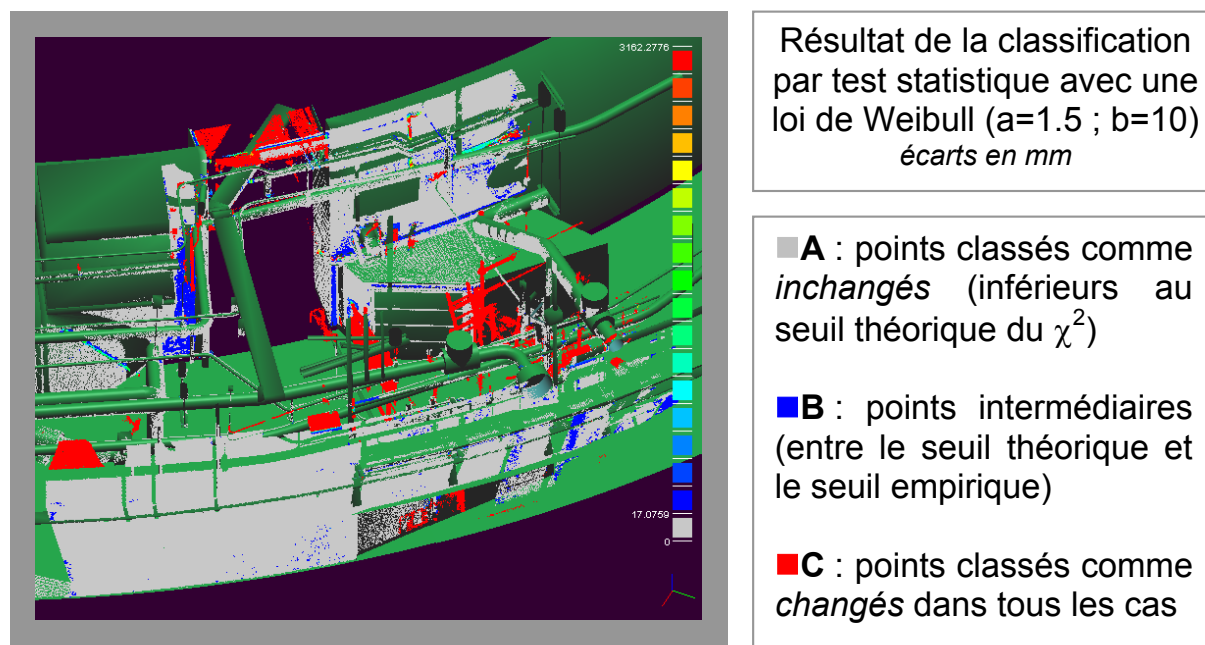


FIG. 3.11 – Influence de la position du seuil de coupure sur la classification par filtrage statistique.

*A (gris) : les points dont la distance du  $\chi^2$  est inférieure au seuil théorique du  $\chi^2$  ;  
 B (bleu) : ceux, assez rares, dont cette distance est comprise entre le seuil théorique et le seuil empirique ; C (rouge) : les autres points.*

mal modélisées (un cylindre modélisé par un nombre insuffisant de facettes planes typiquement). La figure 3.11 présente le résultat d'un filtrage statistique avec la loi de Weibull sur un cas réel. On a représenté ici, en plus des deux classes recherchées (en gris et rouge), les points compris entre le seuil théorique du  $\chi^2$  et le seuil mesuré empiriquement lors de la détermination des paramètres de la loi de Weibull sur un échantillon (en bleu). On voit que le nombre de points concernés est très faible, et que ces points ne correspondent pas à des zones d'intérêt mais plutôt à des zones courbées ou à des bordures, qui sont souvent moins bien modélisées.

Remarque : dans le cas où une distribution théorique du bruit n'est pas disponible, et où l'extraction des paramètres d'une loi paramétrique à partir d'un échantillon de données caractéristique ne serait pas satisfaisante, on pourrait utiliser directement la distribution empirique d'un tel échantillon pour représenter le bruit. La comparaison entre cette distribution empirique et les échantillons extraits autour des points du nuage pourrait alors se faire par un test de type Kolmogorov-Smirnoff (qui remplacerait donc le test du  $\chi^2$  dans la méthode).

### 3.2.4 Extraction des composantes connexes en 3D

Pour finaliser le processus de segmentation du nuage de points à partir de la classification (manuelle ou automatique), il faut regrouper les points du nuage considérés comme "changés" en fonction de leur proximité relative. Ceci peut s'effectuer par une étape d'extraction des "composantes connexes". C'est une technique courante en traitement d'image qui consiste à déterminer des groupes de pixels de valeur égale connectés les uns aux autres (voir figure 3.12). Dans le cas des images, il existe deux manières de considérer que des pixels se touchent : s'ils partagent un bord (auquel cas 4 voisins sont éligibles, on parle de *4-connexité*) ou alors si les pixels se touchent même ponctuellement au niveau de leurs coins (ainsi les 8 voisins du pixel sont éligibles et on parle donc de *8-connexité*).



FIG. 3.12 – Composantes connexes en 8-connexité, détournées sur une image noir et blanc. En 4-connexité, les 4 barres de la forme "0" seraient des composantes connexes indépendantes. Le "point" du point d'interrogation est une composante connexe à lui tout seul, et ce quelle que soit la connexité.

Dans le cas des nuages de points, il semble plus difficile de considérer que des points se touchent. On peut par contre projeter un nuage dans une grille 3D binaire, en considérant que chaque case de la grille aura la valeur "1" si elle contient au moins un point, et "0" sinon. Cette grille a une structure régulière équivalente à celle des images 2D et elle correspond d'une certaine manière à une approximation de la surface représentée par le nuage (avec une précision toute relative, certes).

On a adapté un algorithme classique d'extraction de composantes connexes 2D au cas des nuages en 3D (en fait, au cas des grilles en 3D si on ignore l'étape de projection des points dans la grille). On considère la grille 3D comme un empilement de tranches 2D. On applique à chaque tranche un algorithme d'extraction des composantes connexes classique, puis on regroupe les différentes composantes extraites selon la dimension d'empilement. Au passage, on étend donc la notion de connexité à la troisième dimension. On peut désormais en considérer 3 types : la 6-connexité (les centres des faces de la cellule), la 18-connexité (les centres des faces et des arêtes) ou enfin la 26-connexité (on rajoute les sommets).

Le marquage des composantes connexes en 2D peut être effectué selon deux méthodes. Soit en deux passes [Rosenfeld et Pfaltz 1966] : dans une première on marque les points en fonction de leur connexité directe puis dans une seconde on regroupe les ensembles de points qui se touchent grâce à une table d'équivalence. L'autre méthode [Lumia et al. 1983] se fait en une seule passe en utilisant une table d'équivalence plus complexe et locale (mais en consommant plus de mémoire, bien que ceci ne soit plus un problème aujourd'hui). En théorie n'importe quel algorithme de calcul d'extraction de composantes connexes en 2D peut être utilisé.

Voici en détail le principe de l'algorithme en 3D (qui est équivalent à la généralisation de l'algorithme 2D à la 3D proposée dans [Lumia et al. 1983]) :

- on projette le nuage de points dans une grille 3D régulière et binaire. On pourra aussi utiliser directement la structure octree (voir Chapitre 4) si on en dispose, en la considérant à un niveau donné de subdivision  $N$ .
- cette grille peut alors être elle-même considérée comme un empilement de *tranches 2D* carrées et régulières (de taille  $2^N$  dans le cas de l'octree).
- chacune de ces tranches peut être considérée comme une image binaire (de  $2^N \times 2^N$  pixels dans le cas de l'octree). La valeur d'un pixel correspond simplement à la présence ou non de points dans les cellules correspondantes.
- on applique l'algorithme d'extraction des composantes connexes en 2D à chacune des tranches.
- on fusionne ensuite les composantes connexes de deux tranches successives en fonction de leur connexité mutuelle suivant la direction orthogonale au plan des tranches. Ceci est fait très simplement là encore par le biais d'une table d'équivalence.
- ce principe de fusion est propagé à travers toute la grille, en commençant par les deux premières tranches puis en avançant d'une tranche à chaque fois.
- on regroupe les points présents dans les cellules connexes en sous-nuages.

L'algorithme nécessite un paramètre qui est le pas de la grille 3D, et qui va définir la finesse de la segmentation. Plus le pas est grossier, plus les cellules seront agglomérées, et moins l'algorithme appliquera de "découpages" du nuage initial. Inversement, un pas trop petit peut provoquer une sur-segmentation. Idéalement ce paramètre devrait dépendre de la densité locale du nuage (et l'implémentation de la méthode devrait donc supporter l'utilisation d'un pas adaptatif).

Cette technique d'extraction des composantes connexes est illustrée par la figure 3.13. Elle est rapide et efficace, à condition de bien définir le pas de la grille 3D. Le résultat est un ensemble de sous-nuages de points, qui correspondent à des objets (ou à des portions d'objets, voire encore à des agglomérats d'objets) ayant subi des changements. Le but initial de cette phase de segmentation, évoqué en début de chapitre, est ainsi atteint puisque l'on a efficacement réduit le nombre d'entités géométriques à considérer. On pourra alors appliquer à chaque sous-nuage des traitements spécifiques (comme des opérations de reconnaissance de forme par exemple).

Remarque : cette technique de segmentation profite beaucoup du fait que dans une scène

---

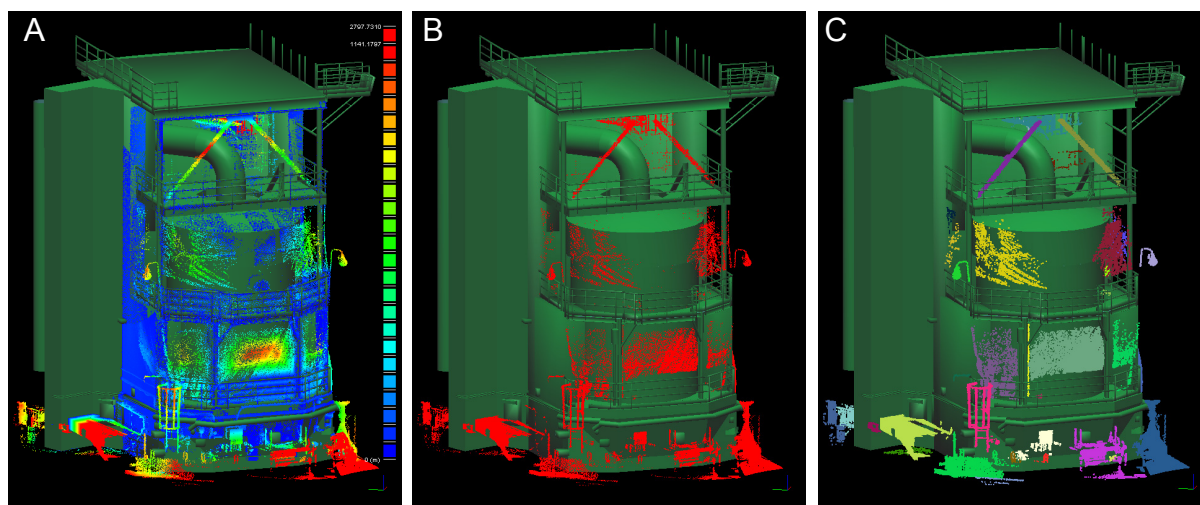


FIG. 3.13 – Processus complet de segmentation par classification.

*A* : calcul des écarts entre un nuage de points et un modèle; *B* : classification (points "changés" en rouge); *C* : extraction des composantes connexes.

3D, les objets en déplacement ou les surfaces déformées sont par définition soit connectées à des objets ou des surfaces qui ne bougent pas, soit ne sont connectées à rien (du moins dans le nuage de points). Le filtrage des points "inchangés" isole dans le premier cas l'objet ou la surface "changeante", et l'extraction des composantes connexes peut ainsi dans les deux cas regrouper les points isolés.

### 3.3 Segmentation par propagation de contour

Comme nous l'avons évoqué dans la section 3.1, on peut également aborder le problème de la segmentation d'un nuage de points valués à partir d'une approche par propagation d'un contour. La propagation du contour peut être réalisée à l'aide d'un algorithme de propagation géodésique d'un front sur une surface dénommé *Fast Marching*. On retrouve cette technique de propagation, entre autre, dans des applications de ré-échantillonnage de nuages de points [Moenning et Dodgson 2003], de calcul de distances géodésiques [Memoli et Sapiro 2003], de reconstruction de surface [Peyré et Cohen 2005] et même de segmentation [Ho et al. 2005] (l'article traitant du cadre général de la segmentation à partir de considérations géométriques, sans proposer de méthode particulière).

Avant de détailler le fonctionnement de l'algorithme de segmentation à proprement parler, nous traiterons tout d'abord de points théoriques et pratiques au sujet de l'algorithme de *Fast Marching*.

### 3.3.1 L'algorithme de Fast-Marching

#### Cas général

L'algorithme de *Fast Marching* a été proposé initialement par Sethian [Sethian 1996], bien que son principe général ait été introduit dans [Tsitsiklis 1995]. Dans sa forme initiale, c'est un algorithme de propagation d'un front (ou d'un contour) sur une grille euclidienne 2D. Des méthodes équivalentes ont été développées parallèlement mais il semble que ce soit cet algorithme qui se soit imposé, en particulier grâce à sa rapidité. Il a été la source de nombreux travaux, en particulier pour l'étendre aux grilles 3D puis plus récemment au cas des maillages triangulaires, et aussi pour ce qui nous intéresse, au calcul rapide de distances géodésiques sur des surfaces implicites [Memoli et Sapiro 2001] ou des nuages de points [Memoli et Sapiro 2003] (le tout en  $n$  dimensions). En pratique, l'algorithme du *Fast Marching* est souvent utilisé pour des applications de segmentation 2D ou 3D mais aussi plus généralement pour toute application qui nécessite un parcours géodésique d'une surface (reconstruction de surface, ré-échantillonnage spatial, etc.).

La théorie mathématique sur laquelle repose l'algorithme et ses différentes déclinaisons est très dense, et nous nous bornerons donc au cas le plus simple pour pouvoir nous focaliser sur la partie algorithmique qui est beaucoup plus parlante.

La méthode générale repose, comme nous l'avons déjà évoqué dans la section précédente, sur le principe des *level sets*, et consiste donc, grossièrement, à considérer le front ou le contour à propager comme une ligne de niveau d'une hyper-surface (voir figure 3.14).

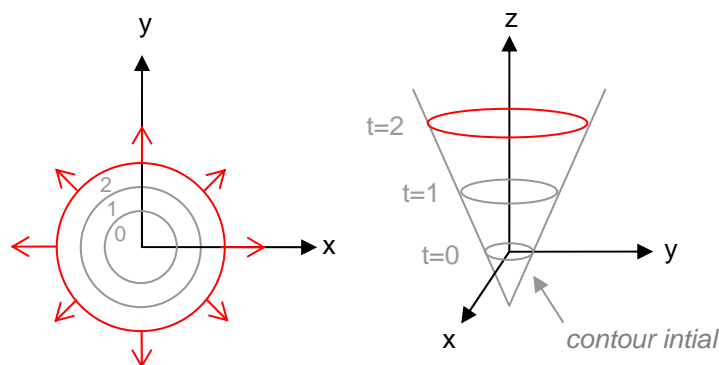


FIG. 3.14 – Illustration du principe des *level sets* en 2D.

*A gauche : évolution du contour en 2D. A droite : hyper-surface définie par la position du front aux différents instants  $t$ .*

Soit  $T$  le temps de trajet du front, et  $F > 0$  la *vitesse* de propagation permise par le milieu (on peut voir aussi  $1/F$  comme la *résistance* du milieu). En l'absence d'autres forces (forces propres au contour ou contraintes externes), l'évolution de la courbe peut

être décrite par l'équation *Eikonal* (ou *équation de Hamilton-Jacobi statique*) :

$$|\nabla T|F = 1 \quad (3.7)$$

Sous cette forme, l'équation 3.7 traduit simplement le fait que la vitesse d'évolution du temps de trajet est inversement proportionnelle à la résistance du milieu. On peut aussi remarquer qu'il y a une relation directe entre la position du front et son temps d'arrivée si la vitesse de propagation est constante.

L'algorithme de Sethian consiste à résoudre cette équation en chaque noeud d'une grille régulière, de proche en proche (voir figure 3.15). La propagation se fait en partant d'un ou plusieurs noeuds de la grille, prédéfinis lors de l'initialisation comme étant des *germes* de la propagation (ils correspondent à l'état initial du front, à  $t = 0$ ). Les germes ne sont pas forcément tous voisins (il peut y avoir plusieurs foyers de propagation différents). On estime ensuite le temps d'arrivée du front au niveau des voisins de chaque *germe* qui ne sont pas eux-mêmes des *germes*. On rajoute celui qui a le plus petit temps d'arrivée au groupe des *germes*, et on rajoute ses propres voisins *non germes* au groupe des voisins. L'opération est répétée par itération jusqu'à ce qu'il n'y ait plus de noeud non traité dans la grille. On traite ainsi les noeuds selon l'arrivée chronologique du front et chaque noeud n'est considéré qu'une fois, d'où une quasi-linéarité de l'algorithme.

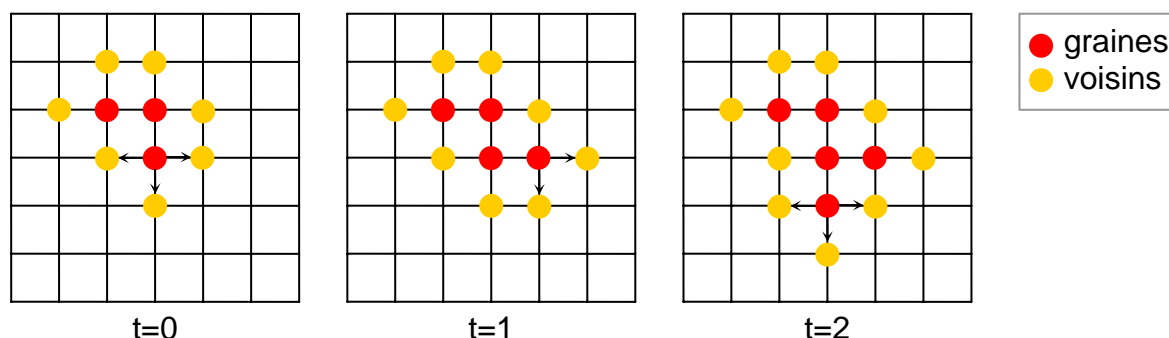


FIG. 3.15 – Illustration du déroulement de l'algorithme de *Fast Marching* sur une grille 2D.

D'un point de vue mathématique, la seule difficulté peut résider dans le calcul du temps d'arrivée du front (et donc la résolution de l'équation 3.7) en un noeud dont plusieurs voisins sont des germes. Si la résistance du milieu (ou la vitesse de propagation) est constante, il suffira de résoudre une simple équation quadratique en considérant selon chaque dimension le noeud voisin dont le temps d'arrivée est le plus petit (on passe "simplement" l'équation 3.7 au carré). Si la résistance du milieu varie localement, alors on doit considérer les ralentissements imposés par le milieu selon chaque direction.

D'un point de vue algorithmique, la principale difficulté peut résider dans la gestion du groupe des noeuds voisins. On doit en effet marquer comme nouveau "germe" toujours

celui dont le temps d'arrivée est le plus petit. Il faut le retirer de la pile et y insérer ses voisins aux bonnes positions. Ceci étant fait à chaque itération, l'opération peut devenir relativement lente lorsque la pile contient beaucoup d'éléments. Yatziv, Bartesaghi et Sapiro [Yatziv et al. 2005] ont proposé une méthode efficace pour améliorer significativement cette étape. Leur méthode consiste globalement à discrétiser les temps d'arrivée et à ne gérer qu'une pile constituée d'un nombre limité de listes de noeuds. Ces listes contiennent des références de noeuds non triés et dont les temps d'arrivée discrétisés sont équivalents. Donc l'amélioration des performances de l'algorithme se fait au prix d'une légère approximation (on prend en effet le premier élément de la première liste de la pile, qui n'est pas forcément celui qui a exactement le temps d'arrivée le plus petit).

### Cas des nuages de points

L'algorithme de *Fast Marching* est conçu initialement pour s'appliquer à des grilles 2D pleines, et par extension directe aux grilles 3D pleines (sans aucun changement réel de l'algorithme). Memoli et Sapiro ont montré qu'il était possible d'utiliser une version très légèrement modifiée de l'algorithme de *Fast Marching* pour calculer une distance géodésique sur un nuage de points [Memoli et Sapiro 2003]. Leur idée est d'utiliser une grille 3D englobant le nuage comme support de l'algorithme standard. D'une part, ils ont démontré qu'il était alors possible de ne calculer le temps d'arrivée qu'au niveau des noeuds proches du nuage. Les autres noeuds peuvent être retirés de l'ensemble des noeuds considérés par l'algorithme, ou bien on peut leur assigner une valeur de *résistance* infinie. D'autre part, les temps d'arrivée au niveau des points peuvent être calculés par interpolation à partir des valeurs des noeuds les plus proches. L'approximation ainsi faite est bornée et elle dépend uniquement du pas de la grille 3D. Il est donc possible d'utiliser l'algorithme de Fast-Marching sur un nuage de points au prix d'une approximation toute relative. Dans notre cas, on ne va pas s'intéresser à la valeur précise du temps d'arrivée en chaque point, mais juste à la propagation du contour, sans être à cheval sur l'ordre d'arrivée précis des points. On peut donc utiliser l'algorithme de *Fast Marching* adapté au cas des nuages de points, mais aussi la structure accélératrice proposée par Yatziv et al.

#### 3.3.2 Segmentation locale par propagation

On veut segmenter les zones correspondant aux changements dans un nuage dont chaque point est muni d'une valeur d'écart censée quantifier sa nature "changeante" et par extension celle de la surface qu'il représente. Suite à ce qui a été dit précédemment, on peut donc par exemple lancer la propagation d'un front/contour à partir d'un point dont la valeur d'écart est forte ; faire en sorte que cette propagation prenne à la fois en compte la topographie locale de la surface sur laquelle il se propage mais aussi la variation des valeurs d'écart ; et enfin idéalement faire en sorte que la propagation s'arrête au niveau

---



des bordures des zones de changement. Tous les points touchés par ce front au cours de la propagation seraient alors regroupés sous la forme d'un sous-nuage qui devrait correspondre, toujours idéalement, à un objet ou à une portion de surface ayant subi un changement.

Notre idée est, en pratique, d'appliquer une telle méthode en la faisant toujours démarrer du point du nuage dont la valeur d'écart est la plus forte (et aussi supérieure à un seuil minimal permettant d'être sûr que ce point est bien "mobile"), puis de retirer du nuage initial les points ainsi segmentés et de recommencer l'opération jusqu'à ce qu'il n'y ait plus de point dont la valeur d'écart soit suffisamment grande. Les avantages théoriques de la méthode sont principalement :

- les points segmentés sont directement regroupés sous forme de sous-nuages de points. Il n'est pas nécessaire d'appliquer des traitements secondaires de regroupement comme l'extraction des composantes connexes (Cf. section 3.2.4).
- a priori, la méthode est moins dépendante du bruit sur les écarts puisqu'on part des valeurs les plus fortes qui sont généralement très supérieures au bruit.
- enfin, lors de la propagation, on pourrait rajouter des contraintes sur la forme du sous-nuage segmenté, en prenant typiquement en compte la courbure, et n'extraire ainsi que des portions de plans, de cylindre, etc..

Reste à définir correctement la correspondance entre valeur d'écart et résistance du milieu d'une part, et les critères d'arrêt de la propagation d'autre part. On peut là encore s'inspirer du traitement d'image classique : le gradient d'intensité dans une image est généralement un bon indicateur des contours mais aussi des évolutions locales d'intensité. Nous proposons dans la section suivante une technique pour calculer le gradient d'un champ scalaire (ici les écarts) porté par un nuage de points.

### Calcul du gradient d'un champ scalaire porté par un nuage de points

Soit  $f$  une fonction de  $\mathbb{R}^2$  dans  $\mathbb{R}$  dérivable. Le gradient de  $f$  est défini comme suit :

$$\overrightarrow{\text{gradient}}(f) = \overrightarrow{\nabla}(f) = \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \quad (3.8)$$

Si  $f$  représente un champ scalaire surfacique, alors on peut interpréter le gradient calculé en un point quelconque de la surface comme un vecteur dirigé vers la direction où l'augmentation du champ scalaire est localement maximale. La norme du gradient est proportionnelle à la "vitesse" d'évolution du champ scalaire au point où il est exprimé. On aura donc un gradient de norme nulle dans les zones où le champ est à peu près constant (mais pas forcément nul), et un gradient constant dans les zones où le champ scalaire évolue linéairement. Dans le cas qui nous intéresse, un gradient constant du champ des écarts est par exemple typique d'un objet se déplaçant en bloc perpendiculairement à une surface fixe, comme un mur qui "sortirait" du sol. Dans le même ordre d'idée, les

lignes *de crête* de la norme du gradient délimitent généralement les objets qui bougent ou apparaissent.

Soit  $P$  un point d'un nuage muni d'un champ scalaire surfacique  $f(P)$ . On veut calculer le gradient du champ scalaire au niveau du point  $P$ . Soit  $V_k = \{P_i, i \in [1, k]\}$  un voisinage de  $k$  points inclus dans une sphère centrée sur  $P$  et de rayon  $R$  ( $R$  étant constant). Une approximation du gradient est alors :

$$\vec{\nabla} f_P = \frac{1}{k} \sum_{i=1}^k \frac{\overrightarrow{PP_i}}{\|\overrightarrow{PP_i}\|^2} (f(P_i) - f(P)) \quad (3.9)$$

Remarques :

- on ne peut pas prendre ici les  $k$  plus proches voisins, car le résultat du calcul dépendrait alors de la densité locale du nuage.
- si le champ scalaire représente une distance euclidienne entre les points du nuage et une entité quelconque (typiquement un modèle - Cf. section 2.2 - ou un point - Cf. section 2.3), alors il peut être intéressant de noter que la norme du gradient en chaque point d'un tel champ scalaire est forcément inférieure ou égale à 1 (voir figure 3.16 pour une illustration simpliste de ceci). La démonstration est simple et repose sur l'inégalité  $a^2 + b^2 \leq (a + b)^2$  (si les points sont distants entre eux d'une distance  $d = \|\overrightarrow{P_i P_j}\|$ , alors leurs "écarts" par rapport à une autre entité - ici,  $f(P_i)$  et  $f(P_j)$  - ne peuvent pas être différents l'un de l'autre de plus de  $d$ ). Ceci permet de filtrer très efficacement les valeurs aberrantes, qui peuvent apparaître à la fois à cause du caractère approximatif du calcul du gradient proposé ici, mais aussi parce que les valeurs scalaires du champ ne sont pas non plus forcément exactes ni cohérentes.

En pratique, cette méthode est très rapide et elle donne des résultats tout à fait satisfaisants. Néanmoins, les résultats peuvent varier en fonction du bruit du champ scalaire (qui n'est pas forcément constant sur tout le nuage) et dans une moindre mesure de la densité du nuage, de la présence de micro-relief, etc. Il peut donc être nécessaire après un calcul du gradient de lisser le résultat. Toujours par analogie avec le traitement d'image, ceci peut se faire par un processus de "moyennage" des valeurs, avec par exemple un filtre gaussien. Soit un point  $P$  et un paramètre  $\sigma$  correspondant à l'écart-type d'une fonction gaussienne définie sur tout l'espace :

$$G_P(Q) = \frac{1}{2\pi\sigma^2} \exp -\frac{\|\overrightarrow{PQ}\|^2}{2\sigma^2}, Q \in \mathbb{R}^3 \quad (3.10)$$

Cette fonction  $G_P$  nous permet de calculer une moyenne des valeurs de gradient des points voisins de  $P$  pondérée en fonction décroissante de leur distance à  $P$ . En pratique, il n'est pas nécessaire de calculer la moyenne avec tous les points du nuage, car au delà d'une distance égale à 2 ou 3 $\sigma$ , la contribution des voisins est quasiment nulle et donc négligeable.

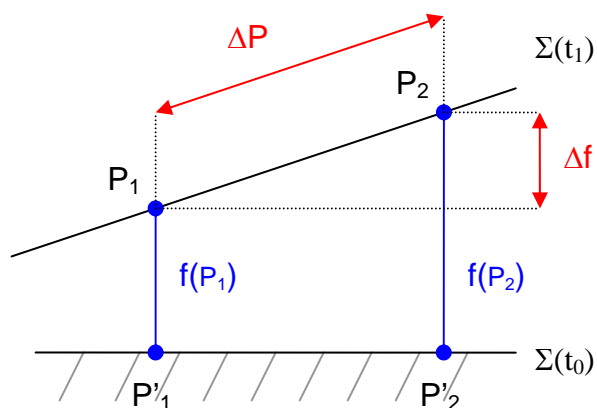


FIG. 3.16 – Illustration de la limitation de la norme du gradient (inférieure à 1) dans le cas d'un champ scalaire correspondant à une distance euclidienne entre des points et une entité quelconque.

$\Sigma$  est la surface à deux époques différentes  $t_0$  et  $t_1$ .  $P_1$  et  $P_2$  sont deux points de la surface à  $t_1$  dont on a calculé les écarts  $f(P_1)$  et  $f(P_2)$  par rapport à la surface à  $t_0$  (distance aux points les plus proches  $P'_1$  et  $P'_2$ ). Dans le cas très simple schématisé ici, on voit que la différence des écarts correspond à un côté d'un triangle rectangle, et sera donc toujours inférieure à la distance entre  $P_1$  et  $P_2$  qui correspond à l'hypoténuse de ce triangle. Le cas général est bien plus compliqué mais peut se démontrer par l'absurde.

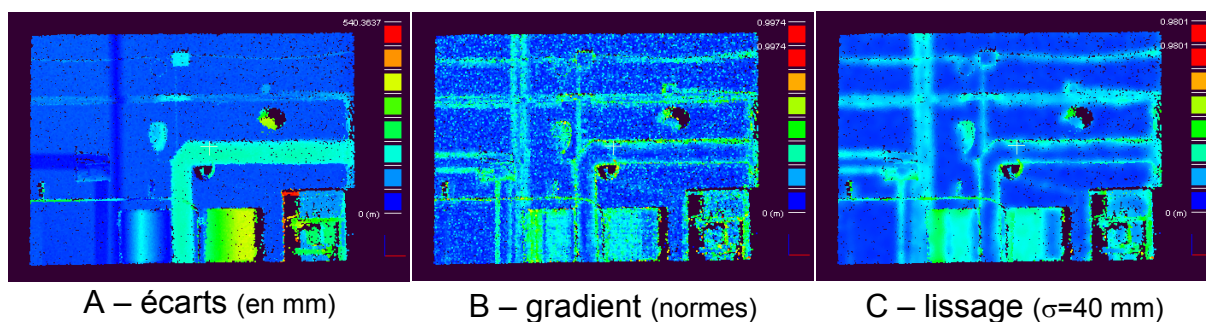


FIG. 3.17 – Illustration du calcul de gradient (B) sur un nuage valué (A) puis lissage par filtre gaussien sur la norme du gradient (C).

Les figures 3.17 et 3.18 permettent de visualiser sur un exemple concret le résultat du calcul de gradient sur un nuage de points (ainsi que l'effet d'un filtre gaussien sur ce résultat, dans la première figure).

### Propagation d'un front dirigée par le gradient

Le principe de la deuxième étape du processus de segmentation est de propager un front sur le nuage de points en partant d'un point dont on sait qu'il appartient à une zone de changement, et de contraindre cette propagation pour qu'elle s'arrête au niveau des bordures de la zone de changement. On a choisi de contraindre la propagation en fonction de la norme du gradient du champ des écarts calculés en chaque point.

La norme du gradient des écarts nous donne une indication sur l'évolution du champ des écarts :

- si le gradient est fort, alors les écarts des points évoluent localement rapidement. On est donc sans doute sur une zone qui s'éloigne de plus en plus de la référence.
- si au contraire le gradient est très faible, alors les écarts évoluent peu. On est donc probablement soit dans une zone de changement mais à distance constante de la référence (la face supérieure d'un parallélépipède qui reposerait sur le sol et serait apparu par exemple), soit plus fréquemment dans une zone inchangée.

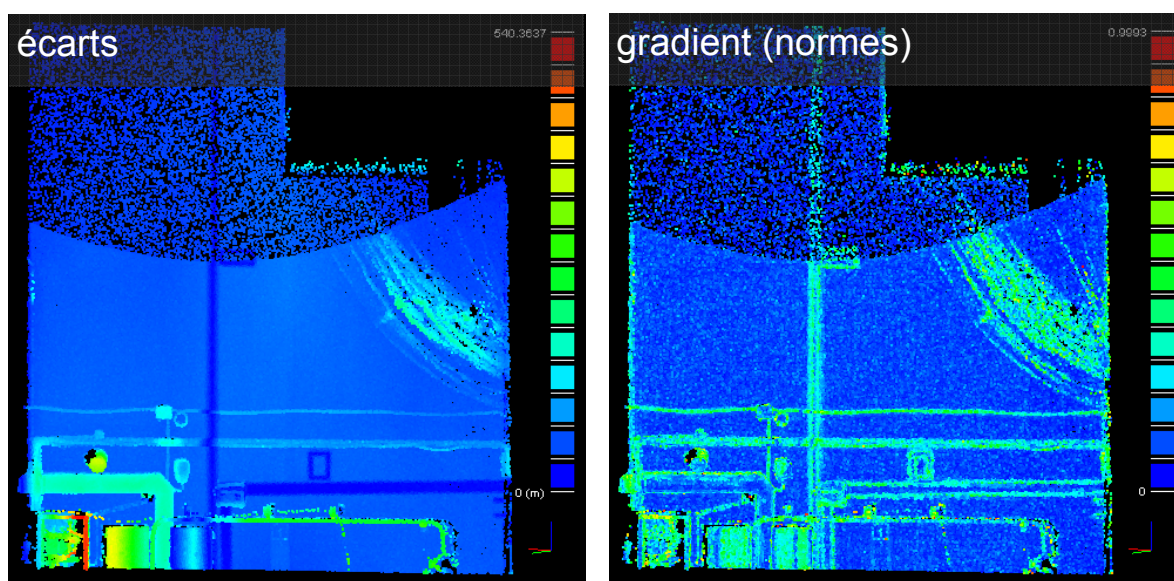


FIG. 3.18 – Relation entre écarts (à gauche) et normes du gradient (à droite).  
*Le nuage représente un mur parcouru par des câbles et autres tuyaux. Il est comparé au plan théorique du mur uniquement (non représenté). On peut remarquer que les zones d'écart maximal ne sont pas forcément celles où la norme du gradient est maximale. Le gradient atteint sa norme maximale dans les zones où les écarts évoluent orthogonalement aux surfaces statiques.*

La propagation, telle qu'on l'a prévue, part du point dont l'écart est maximum mais qui a donc potentiellement un gradient nul (si le maximum est dans une zone plate, comme dans le cas de la figure 3.18). On doit donc favoriser la propagation vers des

normes de gradient supérieures, et la rendre plus difficile dans l'autre sens. On a donc modifié légèrement l'algorithme du *Fast Marching* pour pouvoir accélérer ou ralentir la propagation localement (en faisant varier localement la valeur de  $F$  dans l'équation 3.7). Le temps de trajet entre deux noeuds  $P_i$  et  $P_j$  (dans le sens  $P_i \rightarrow P_j$ ) doit être une fonction décroissante de la différence signée de leur normes de gradient (soit  $\|\overrightarrow{\nabla P_j}\| - \|\overrightarrow{\nabla P_i}\|$ ). Il est par contre nécessaire que ce facteur d'accélération soit toujours positif pour la bonne marche de l'algorithme. Pour déduire ce facteur en fonction de la différence de gradient, on peut utiliser par exemple une fonction exponentielle pour passer de l'intervalle  $[-1; 1]$  vers un intervalle  $[0; +\infty[$ , une accélération inférieure à 1 étant un ralentissement. On peut au passage définir un paramètre  $\alpha$  qui permettra d'exagérer ce facteur d'accélération et ainsi mieux détecter les sauts de temps de trajet (qui seront notre critère de coupure). La fonction de calcul de l'accélération locale  $F_\alpha$  à partir de la différence des normes du gradient sera alors de la forme :

$$F_\alpha(P_i \rightarrow P_j) = \exp[\alpha(\|\overrightarrow{\nabla P_j}\| - \|\overrightarrow{\nabla P_i}\|)] \quad (3.11)$$

$F_\alpha$  remplace alors  $F$  dans l'équation Eikonal (3.7)

Comme on l'a déjà à moitié évoqué, le critère d'arrêt de la propagation sera simplement un seuil sur le saut temporel effectué par le front pour passer d'un noeud au suivant. En effet, les bordures des zones de changement sont caractérisées par le passage d'une zone de gradient non nul et normalement assez fort (puisque les points s'éloignent de la surface de référence) à une zone de gradient nul ou quasiment nul (qui correspond aux points qui n'ont pas bougé). Au niveau des bordures, la propagation se fait donc d'un gradient fort vers un gradient faible, ce qui est fortement pénalisé par la fonction d'accélération  $F_\alpha$  (d'autant plus que  $\alpha$  est grand). On peut donc définir un seuil sur la valeur de  $F_\alpha$  par exemple pour arrêter la propagation.

La figure 3.19 permet de visualiser les temps de trajet directement sur la géométrie d'un objet (qu'on voit à gauche coloré en fonction des écarts) dans le cas d'une propagation standard (au centre) et contrainte par la norme du gradient (à droite). La figure 3.20 représente l'évolution du temps d'arrivée aux noeuds lors de la propagation standard par *Fast Marching* (à vitesse constante). On voit que le temps évolue ici de manière à peu près stable. On peut enfin comparer cette courbe avec la figure 3.21 qui représente la même chose dans le cas d'une propagation accélérée localement en fonction de la valeur du gradient. On voit que le temps de trajet saute d'un coup (au niveau de la frontière entre le galet et le plan, qui correspond à la zone de passage du gradient de valeurs très fortes vers des valeurs nulles). Le gradient semble donc être un critère efficace pour asservir correctement la propagation et ainsi déterminer les bordures des zones de différences.

Nous insistons par contre sur la nécessité d'être assez sévère sur ce critère d'arrêt de la propagation (et donc mettre un seuil bas sur l'accélération) car bien qu'efficace, il n'est pas très robuste. Il vaut donc mieux sur-segmenter plutôt que de prendre le risque que la propagation passe par dessus la frontière de la zone de différence et continue alors à se

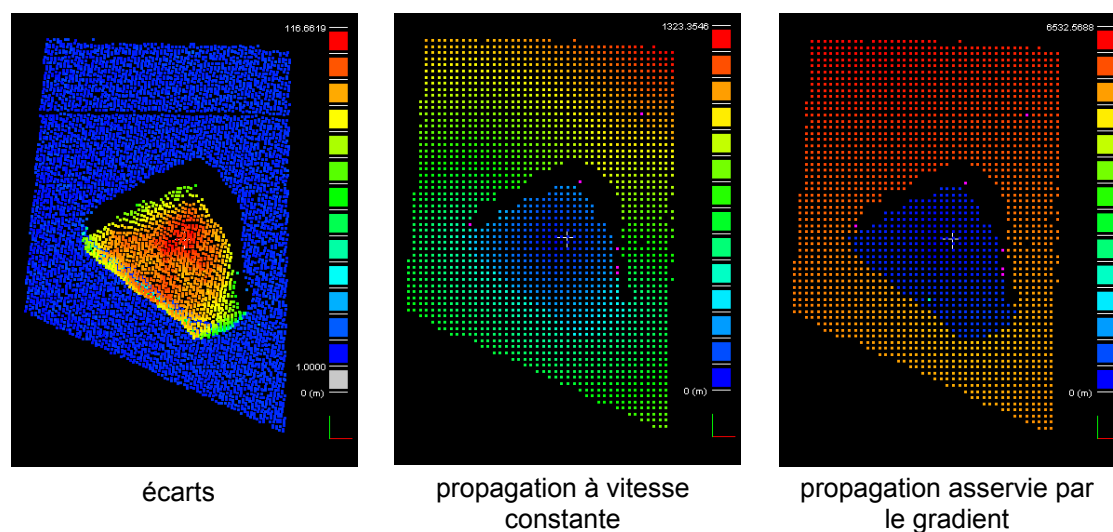


FIG. 3.19 – Illustration de la propagation d'un front avec l'algorithme de *Fast Marching*. A gauche : le nuage de points correspondant à un galet, coloré en fonction des écarts. Au milieu : propagation géodésique à vitesse constante. A droite : propagation géodésique accélérée localement en fonction de l'évolution du gradient. Les temps d'arrivée sont représentés sur les deux images de droite au niveau des noeuds de la grille, avec une coloration du bleu au rouge pour les temps croissants.

propager dans les zones "immobiles". Les transitions douces, typiques d'un changement de type *déformation* sont les plus sensibles de ce point de vue, car le saut de gradient à la frontière peut être en pratique assez faible.

Remarque : avec un simple seuil sur l'accélération, qui elle-même dépend d'un saut négatif de la norme du gradient, on risque parfois de stopper la propagation trop tôt, en particulier si l'objet ayant subi un changement présente des "paliers" d'écarts (des zones qui encerclent totalement le point d'où débute la propagation et où les écarts sont constants, tout en faisant partie de la zone de changement). Dans ce cas on sur-segmentera l'objet, en le coupant en morceaux au niveau de chaque palier. Ceci n'est pas forcément un problème puisqu'il y a de fortes chances que les paliers délimitent différents composants de l'objet. Nous sommes néanmoins conscients du caractère encore assez élémentaire du critère d'arrêt de la propagation. Celui-ci devrait faire l'objet d'une étude plus approfondie (que nous n'avons pu mener faute de temps), ainsi qu'une meilleure exploitation du potentiel de l'approche par propagation d'un contour, en particulier pour ce qui est de la prise en compte des propriétés géométriques des ensembles de points segmentés.

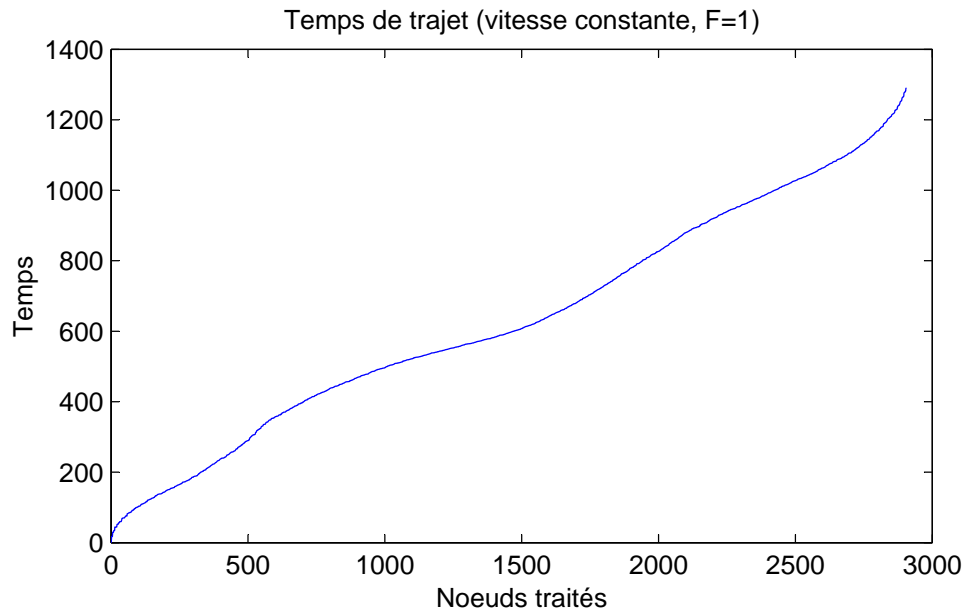


FIG. 3.20 – Temps de trajet lors d’une propagation géodésique à vitesse constante par l’algorithme de *Fast Marching*, en fonction de l’ordre d’arrivée aux noeuds (sur le nuage représenté en figure 3.19).

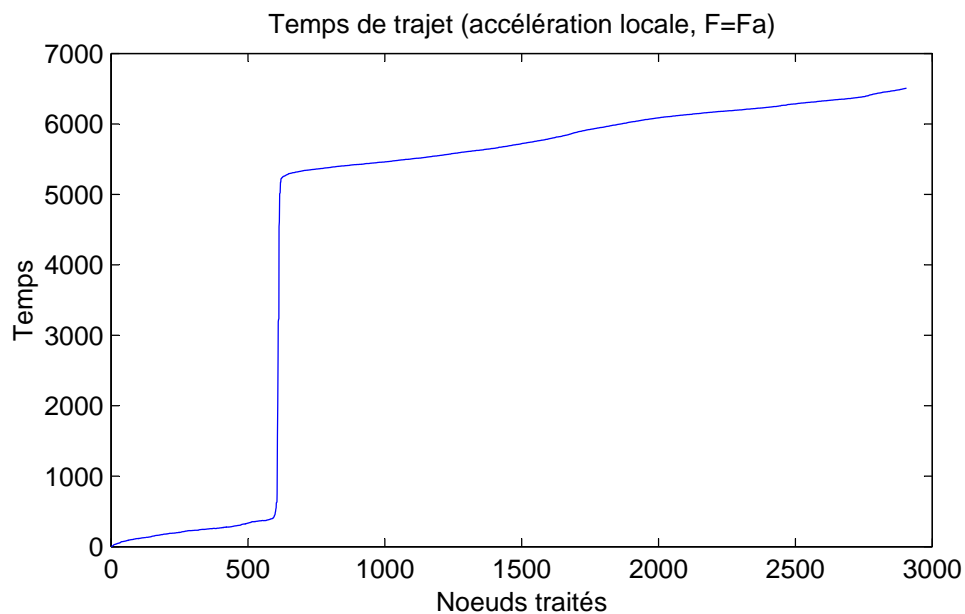


FIG. 3.21 – Temps de trajet lors d’une propagation géodésique contrainte par la valeur du gradient (avec  $\alpha = 50$ ) par l’algorithme de *Fast Marching*, en fonction de l’ordre d’arrivée aux noeuds (sur le nuage représenté en figure 3.19).

*Le saut autour du 600<sup>me</sup> noeud correspond à l’arrivée du front aux bords de l’objet.*

## 3.4 Résultats

Nous présentons tout d'abord des temps d'exécution des deux algorithmes de segmentation "automatiques" présentés en section 3.2.3 et 3.3. Les calculs sont effectués sur un ordinateur portable avec 1 Go de mémoire vive et un processeur Intel Centrino 1.7 Ghz. Le tableau 3.1 présente des temps de calculs indicatifs pour l'algorithme de segmentation par filtrage statistique local sur différents nuages de points. Les temps relatifs des différentes étapes sont détaillés : le calcul de l'octree (qui est une structure permettant la détermination des voisinages des points de manière efficace, détaillée dans le chapitre 4), le filtrage statistique en temps que tel et enfin l'extraction des composantes connexes. Parallèlement, le tableau 3.2 présente des temps de calculs indicatifs pour l'algorithme de segmentation par propagation contrainte par la norme du gradient, et ce sur les mêmes nuages. Les différentes étapes sont dans ce second cas : le calcul de l'octree, le calcul du gradient en chaque point, l'application d'un filtrage gaussien sur les normes du gradient (processus optionnel), et enfin les propagations à partir des points dont la valeur d'écart est maximale. On peut voir que cette méthode est globalement plus lente, et qu'elle génère aussi plus de sous-nuages (on a déjà évoqué la tendance de cet algorithme à sur-segmenter les nuages). Les paramètres sont aussi plus difficiles à définir et elle risque de sembler moins stable du point de vue d'un utilisateur. Sa lenteur s'explique principalement par le fait que l'opération la plus coûteuse est la détermination des plus proches voisins (même lorsque celle-ci est améliorée par l'utilisation d'une structure *octree*). Le premier algorithme n'y a recours qu'une fois (lors du test du  $\chi^2$  local) alors que le second y a recours deux fois (une première fois lors du calcul du gradient, et une seconde lors du filtrage gaussien). Au total, le deuxième algorithme est à peu près 4 fois plus lent. Il garde néanmoins un fort potentiel et devrait faire l'objet d'une étude approfondie (ainsi que d'une optimisation du code qui n'a pas pu être réalisée, faute de temps).

Nuage (points)	Octree (s)	Filtrage (s)	C.C. (s)	Total (s)	Nb. C.C.
34931	0,12	1,16	0,06	1,34	49
932136	1,32	44,33	0,81	46,46	197
2180182	3,03	104,93	0,94	108,9	261

*C.C. = Composantes Connexes*

TAB. 3.1 – Temps de calcul indicatifs (en secondes) pour la segmentation de nuages de points par l'algorithme de filtrage statistique local.

Nuage (points)	Octree (s)	Gradient (s)	Flou (s)	Propag. (s)	Total (s)	Nb. C.C.
34931	0,11	0,63	0,84	0,84	2,42	125
932136	1,37	37,47	114	18,19	171,03	679
2180182	3,03	100,37	230,15	57,14	390,69	491

TAB. 3.2 – Temps de calcul indicatifs (en secondes) pour la segmentation de nuages de points par l'algorithme de propagation contrainte par la norme du gradient.



Nous pouvons enfin comparer en figure 3.22 les résultats de ces deux algorithmes appliqués à des données réelles. On peut clairement visualiser la tendance de l'algorithme de segmentation par propagation à sur-segmenter le nuage.

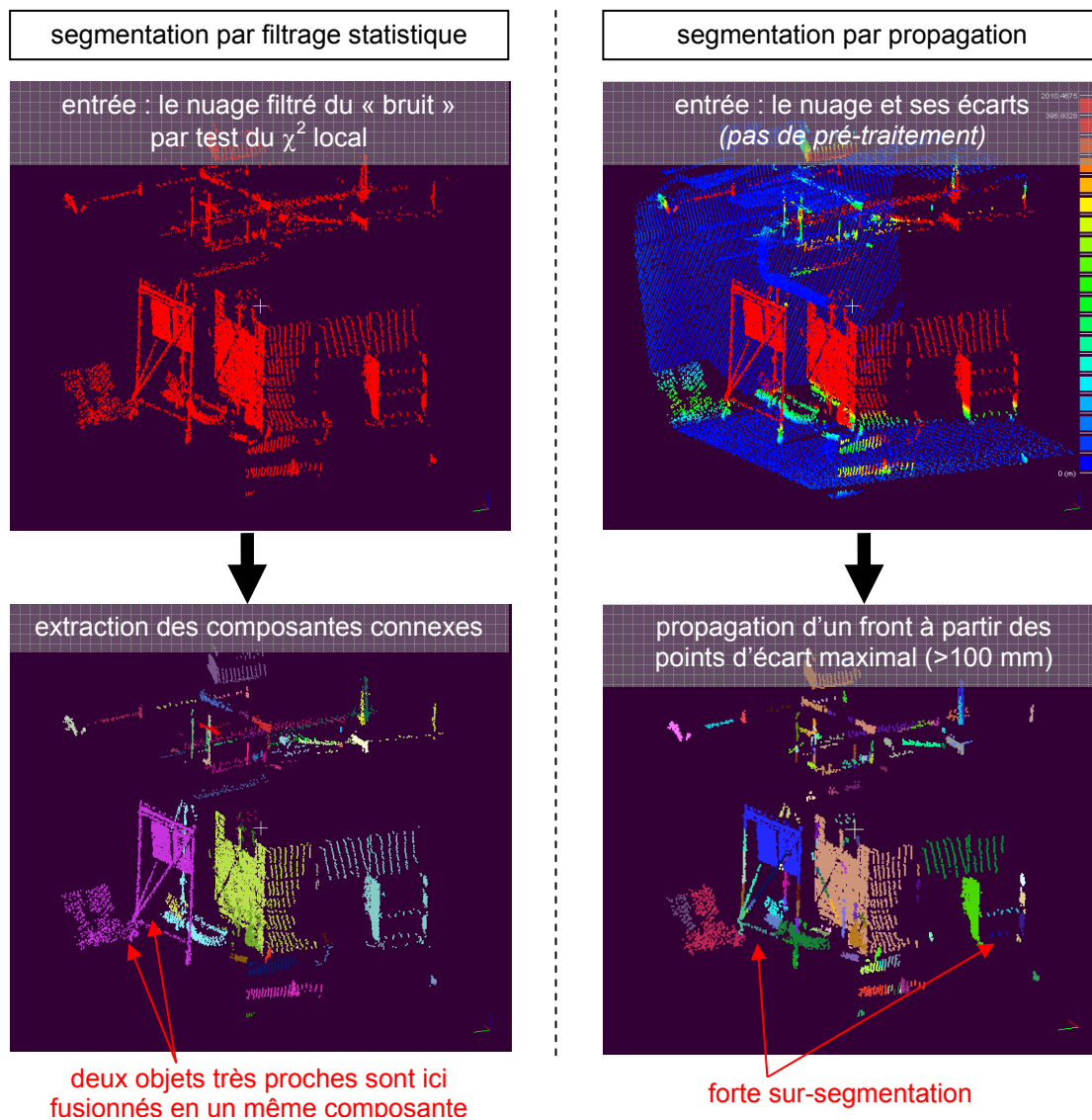


FIG. 3.22 – Comparaison des algorithmes de segmentation sur un même nuage de points.

### 3.5 Prise en compte de l'asymétrie de comparaison

Dans les deux précédentes sections de ce chapitre, nous avons proposé plusieurs méthodes de segmentation d'un nuage de points 3D valué permettant d'isoler les zones ayant subi des modifications. On détecte ainsi des changements, ou plutôt des zones touchées par

des changements, qu'on peut faire ressortir graphiquement. Ceci améliore considérablement la perception par un utilisateur des différences entre deux jeux de données, simplifie grandement son travail en limitant les zones d'intérêt et accélère donc le processus global.

Néanmoins, il faudrait dans l'idéal être capable d'une part de caractériser les changements et d'autre part de reconnaître les objets touchés, le tout automatiquement. Reconnaître les objets touchés par des changements implique d'ailleurs d'être capable de remonter aux objets entiers alors qu'on ne segmente parfois qu'un morceau de leur surface (celui qui s'est effectivement déplacé ou déformé). De telles opérations avancées de diagnostic pourraient représenter une avancée majeure dans le domaine de la détection de changement, à condition par contre soit d'être très fiables, soit de fournir une valeur de confiance pour chacun de leurs résultats (car un diagnostic nécessitant le contrôle par l'opérateur de tout ou une majorité des zones de changement apporterait finalement peu à ce que l'on a déjà réalisé jusque ici). Cela permettrait par exemple d'établir automatiquement que tel objet s'est déplacé en partant de telle position pour arriver à telle autre position, que tel objet a disparu, ou encore que tel objet s'est déformé. "Tel objet" pourrait d'abord être désigné graphiquement, sous forme d'un sous-nuage de points (tous les points étant colorés d'une couleur unique), mais dans un dernier degré de sophistication, on pourrait enfin imaginer que le système reconnaisse l'objet à partir d'une base de données (en confrontant les points à des modèles 3D par exemple) et soit ainsi capable de le nommer.

Ces considérations sont encore de l'ordre de la spéculation et nous ne prétendons pas apporter une solution ici. Elles impliqueraient une intelligence artificielle évoluée et des capacités de reconnaissance de forme rapides et très robustes. Nous nous contenterons dans cette section de faire des observations sur l'apport potentiel de l'asymétrie de comparaison au problème de la segmentation et surtout de la caractérisation des déplacements. Nous avons en effet rapidement évoqué le caractère asymétrique du processus de comparaison de données 3D en section 1.1, sans donner plus de précisions. Le processus de comparaison n'est pas symétrique car d'une part, les données qui vont supporter les calculs d'écarts, selon qu'on les effectue dans un sens ou dans l'autre, ne sont généralement pas les mêmes (ni dans leur nature, ni dans leur échantillonnage ou leur couverture spatiale) et d'autre part, dans le cas des nuages laser, le point de vue n'est potentiellement pas le même, ce qui va entre autre modifier les critères de visibilité des points (Cf. section 2.3.3). L'analyse que l'on va mener ci-après, sans donner de solutions concrètes, va au moins dans le sens d'un diagnostic évolué tel qu'on l'a envisagé ci-dessus, et pourrait typiquement être à la base de travaux futurs.

L'idée principale est de comparer les données dans les deux sens pour tenter de caractériser par analyse croisée les types de changement qui ont pu intervenir sur les zones segmentées (plutôt que de se limiter au sens "logique"). Nous commençons par étudier le comportement d'une telle comparaison "symétrique" pour les différentes classes de changements (le lecteur peut se référer à l'annexe B qui présente des exemples de comparaisons détaillées ainsi que le résultat "idéal" - en terme de segmentation, voire de diagnostic avancé - que l'on pourrait espérer dans chaque cas, et ce pour les différentes classes de

---

changements).

### 3.5.1 Cas des disparitions/apparitions

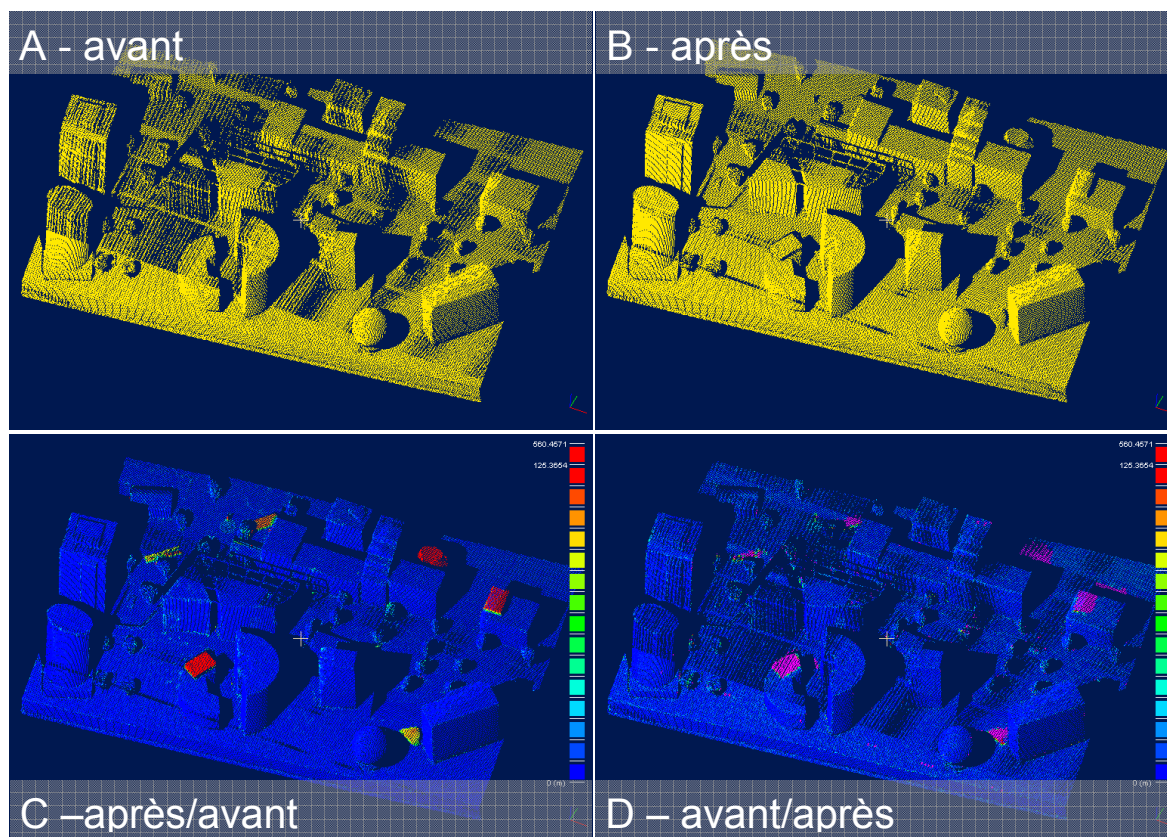


FIG. 3.23 – Calculs d'écarts "symétriques" (C,D) entre deux nuages de points (A,B), avec prise en compte des points de vue du capteur.

*Le nuage représente une maquette de ville imaginaire réalisée par un étudiant de l'École d'Architecture de la Villette (Paris) et numérisée avec un scanner Soisic prêté par la société Trimble (à l'instar du nuage présenté en figure 2.10). La couleur rose correspond aux zones déterminées comme "cachées" lors de l'analyse de la visibilité du capteur (voir section 2.3.3).*

On considère deux nuages de points laser acquis à deux instants différents et dont on connaît les différents points de vue. On traite ici uniquement le cas de l'apparition pour simplifier le raisonnement, mais le cas de la disparition est totalement équivalent si on inverse le rôle de chaque nuage.

La comparaison logique entre les deux nuages ("après" par rapport à "avant") fait apparaître plusieurs zones de changements (voir figure 3.23-C). Ces zones sont isolées,

clairement détachées du reste et donc facilement segmentables. Il est par contre impossible de déterminer *algorithmiquement* avec ces seules données quel type de changement a touché ces zones (sans parler de ce que représentent ces zones).

Nous pouvons tenter de calculer les écarts dans l'autre sens ("avant" par rapport à "après"). On voit alors apparaître en rose (figure 3.23-D) des zones *invisibles* (notion introduite en section 2.3.3). Le contour de ces zones rappelle les silhouettes des zones segmentées en "rouge" lors de la comparaison dans le sens "logique" (voir figure 3.24, *détail 1*), bien que leurs positions ne soient pas forcément très proches l'une de l'autre (voir le *détail 2* de la même figure).

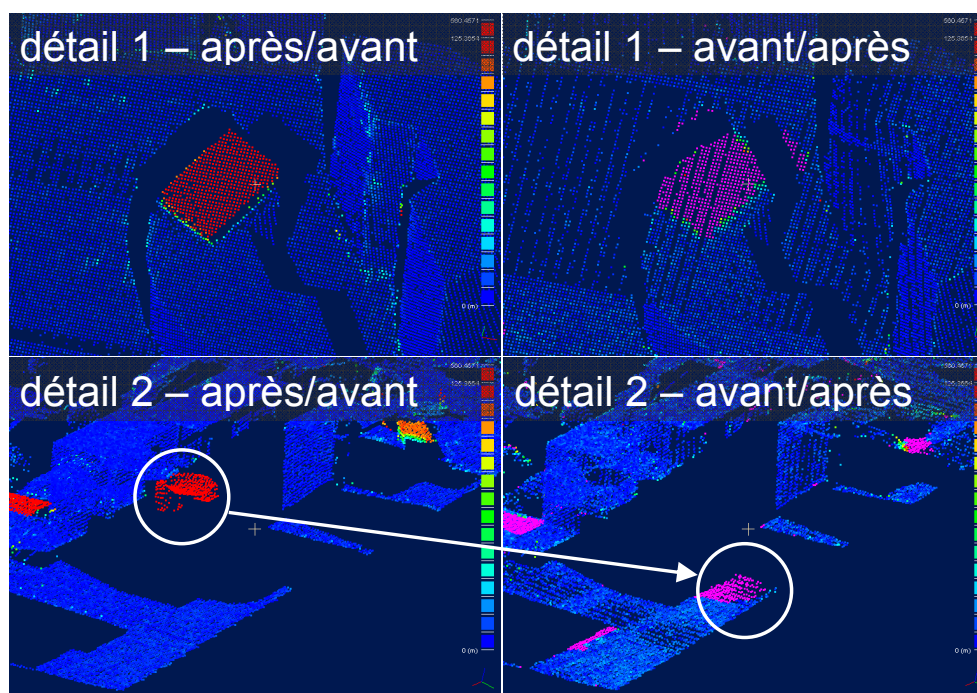


FIG. 3.24 – Exemples de comparaison symétrique dans le cas d'apparitions.

Ceci s'explique simplement par le fait que ces zones "roses" sont les ombres portées du point de vue du capteur des zones segmentées, qui s'avèrent ici être des *apparitions* et n'étaient donc pas présentes dans le premier nuage. Il est donc logique que le contour des zones "roses" soit directement lié à la silhouette des zones segmentées. De plus, lors du calcul de visibilité, on pourrait associer directement les points "roses" (dans "D") avec les points "rouges" (dans "C") qui les occultent, et ainsi déterminer directement une correspondance entre chaque zone segmentée et chaque zone d'ombre. La forme du contour ne nous apporte donc pas d'information que nous n'ayons pas déjà.

La forme de la surface où l'ombre se projette peut, par contre, nous donner des indications sur la nature du changement (en particulier au niveau de sa frontière). Si les changements avaient été des déplacements avec recouvrement (cas que nous traitons dans

la section suivante) on verrait apparaître des différences effectives (en rouge) à côté des zones roses lors de la comparaison inverse. Et dans le cas de déformations, on verrait apparaître, soit là encore, des zones rouges (cas de la déformation avec recouvrement) soit alors des points roses dont la forme n'aurait probablement pas du tout la forme de la surface environnante, mais plus probablement celle de l'objet segmenté (voire ni l'une ni l'autre, dans le cas d'une destruction particulièrement chaotique).

On peut donc établir deux règles permettant, dans une certaine mesure, de caractériser un changement de type apparition/disparition, à partir des deux caractéristiques des zones "roses" que l'on vient d'évoquer :

- 1) les zones d'ombre ("roses") sont isolées, au sens où il n'y a pas de points distants ("rouges") adjacents/proches autour de leur frontière.
- 2) elles sont généralement projetées sur des surfaces quelconques, dont la forme n'a pas de rapport avec celle de l'objet qui est à l'origine de l'ombre. En pratique, il n'y a pas en général de rupture de pente forte ou chaotique au niveau de la frontière (sauf dans le cas où l'objet qui est apparu cache entièrement et exactement un autre objet).

### 3.5.2 Cas des déplacements

#### Sans recouvrement

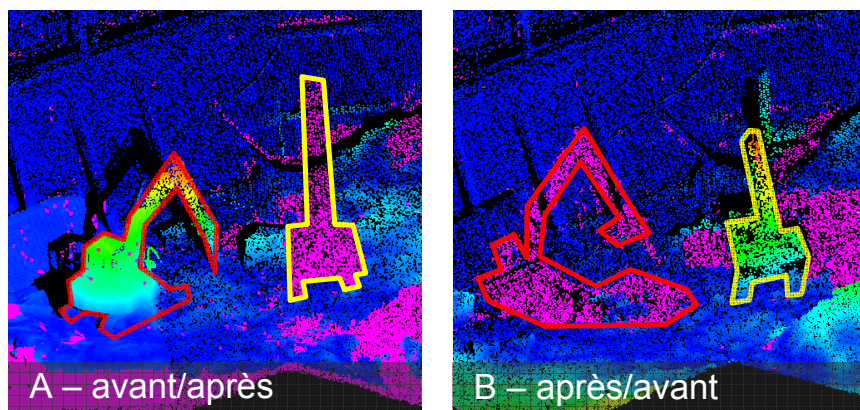


FIG. 3.25 – Exemple de comparaison symétrique dans le cas d'un déplacement sans recouvrement.

On a déjà vu en section 1.2, que le cas des déplacements sans recouvrement était équivalent dans le principe au cas des disparitions/apparitions. Les remarques faites ci-avant au sujet de la comparaison symétrique dans le cas d'un changement de type disparition/apparition sont donc encore valables.

On peut visualiser un exemple d'un tel changement en figure 3.25 (extrait des nuages du chantier de Malakoff). On voit qu'un moyen de différencier le cas particulier des dé-



placements sans recouvrement de celui des apparitions/disparitions serait de reconnaître la similitude entre deux nuages segmentés à partir de chaque carte d'écart (c.à.d. pour chaque sens de comparaison). Dans cet exemple, on pourrait extraire et rapprocher deux états de la pelleuse (en "rouge" dans A et "jaune" dans B). Le rapprochement pourrait se faire soit à partir des points segmentés (par recalage automatique de nuages de points, grâce à une technique du type ICP par exemple), ou peut-être, pour aller plus vite, à partir des contours (bien que cela semble moins évident à priori).

### Avec recouvrement

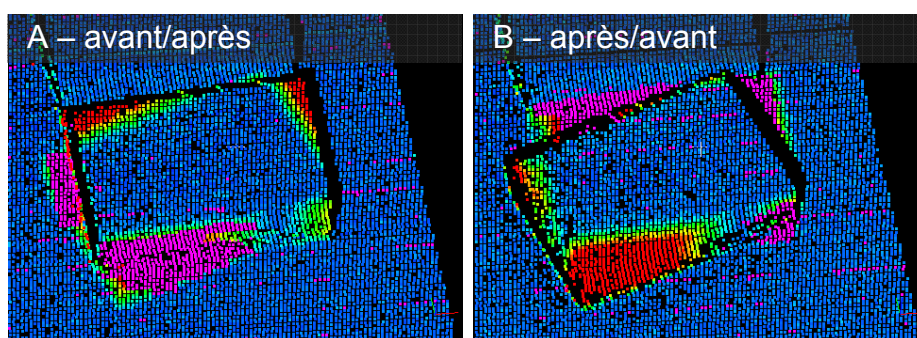


FIG. 3.26 – Exemple de comparaison symétrique dans le cas d'un déplacement avec recouvrement.

Un déplacement *avec recouvrement* peut impliquer qu'une partie de l'objet n'a pas bougé, ou plutôt qu'une partie de l'objet apparaîtra comme "immobile" lors du calcul des écarts. De plus, quel que soit le sens de comparaison, outre la zone "immobile" (qui ne correspondra pas à la même partie de l'objet dans chaque cas), on devrait aussi observer une partie *visible* dotée d'écarts non nuls, et enfin une dernière partie *invisible*.

La figure 3.26 donne un exemple d'un tel déplacement (cas d'un parallépipède qui pivote sur lui même). On observe pour chaque comparaison les trois types de zones qu'on vient d'évoquer. Comme on l'a dit précédemment, les zones "roses" sont généralement proches dans ce cas de zones "rouges" du même nuage (bien qu'elles ne soient pas toujours forcément adjacentes au sens strict). Si le parallépipède s'était translaté selon la direction de sa longueur, les zones "roses" et "rouges" seraient disposées de part et d'autre de la zone "bleue", et ce dans les deux sens de comparaison. C'est certes un cas limite, mais celui-ci pourrait être analysé à tort comme étant la succession d'une disparition d'un bord du parallépipède suivie de l'apparition de l'autre côté d'un morceau équivalent. Pour envisager une méthode robuste de caractérisation du déplacement, on rejoint donc un besoin évoqué précédemment pour les opérations de diagnostic avancé : être capable de remonter à l'objet complet à partir des morceaux de surfaces segmentés. On pourrait reconnaître alors la juxtaposition des 3 zones et on serait ainsi capable de caractériser avec plus d'assurance tous les cas de déplacement avec recouvrement.

De plus, bien que dans le cas général on observe souvent une quasi-adjacence entre zones roses et zones rouges, on ne pourrait pas pour autant désigner l'objet concerné par le déplacement dans sa globalité. Il paraît donc de toute façon très intéressant d'être capable de segmenter géométriquement ces objets dans le nuage pour pouvoir effectuer un diagnostic précis.

Ce cas est par contre différentiable du suivant (celui des déformations) grâce au caractère régulier des surfaces au niveau des frontières des zones "roses". Dans le cas des déplacements, la géométrie au niveau des frontières devrait être régulière mais avec des changements d'orientation forts (typiquement des plans orthogonaux). Au contraire, dans le cas des déformations, la géométrie au niveau des frontières sera généralement courbe et les changements de pentes beaucoup plus lisses.

### 3.5.3 Cas des déformations

Le dernier cas de changement à envisager est celui des déformations. Il est beaucoup plus complexe étant donné le nombre potentiellement infinis de cas qu'il englobe. On peut par contre raisonner par élimination.

#### Sans recouvrement

La première condition émise dans le cas des apparitions/disparitions était l'isolement des zones "roses". Cela est envisageable pour certains cas de déformation sans recouvrement. Dans ces cas là, la deuxième condition sera discriminante : en général la zone "rose" correspondra à l'objet dans son état initial, et la transition avec les surfaces avoisinantes sera probablement irrégulière et fortement discontinue. On obtient d'ailleurs au passage, en segmentant les points roses, le nuage correspondant à l'objet avant déformation. Sachant qu'on possède déjà le nuage de points correspondant à son état après déformation, cela pourrait par exemple permettre une analyse plus poussée et spécifique de la déformation.

#### Avec recouvrement

Dans le cas des déformations avec recouvrement, la première condition n'est jamais vérifiée. On se retrouve par contre dans une configuration très proche de celle des déplacements avec recouvrement (voir figure 3.27). Là encore c'est la deuxième condition qui devrait nous permettre de différencier les deux cas.

---

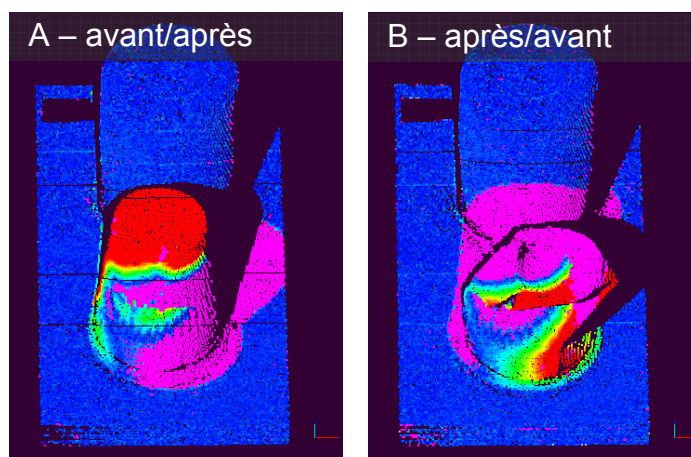


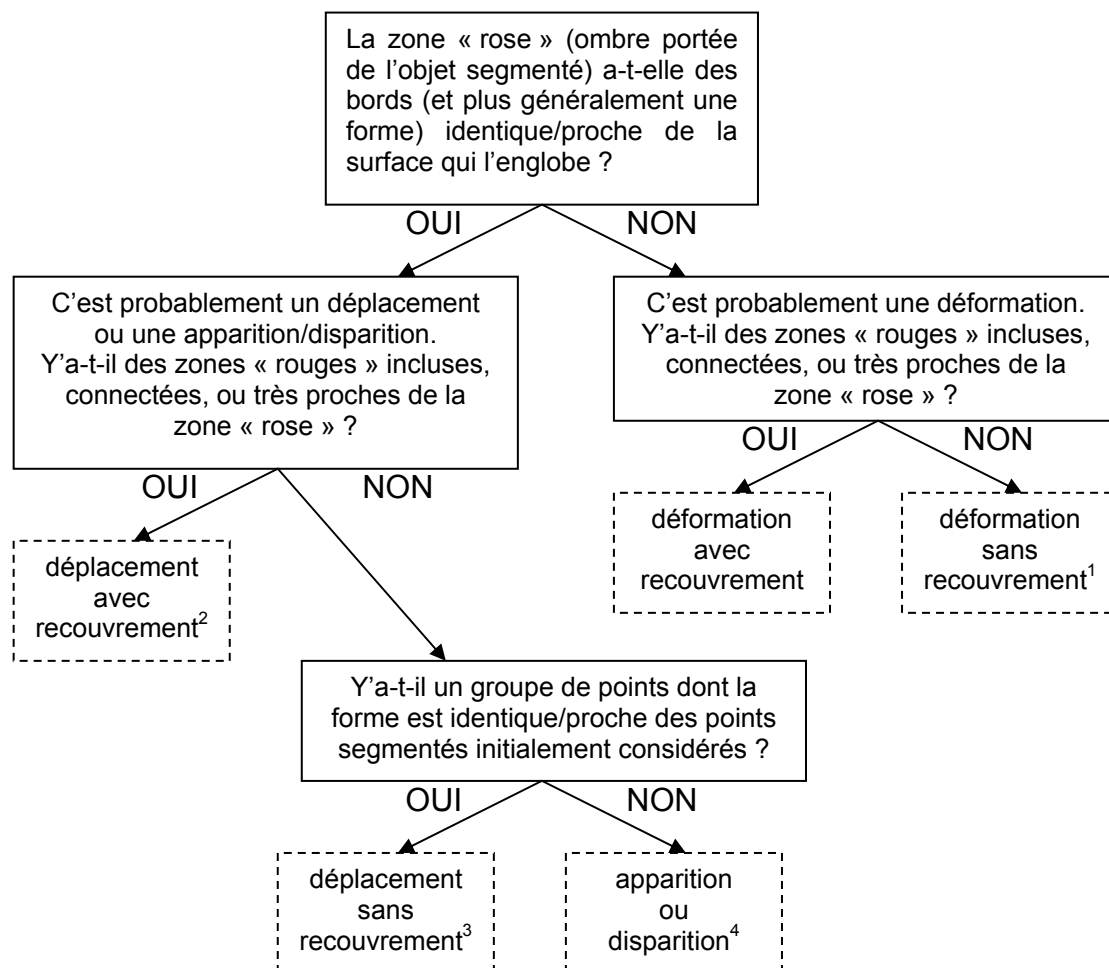
FIG. 3.27 – Exemple de comparaison symétrique dans le cas d'une déformation avec recouvrement.

### 3.5.4 Synthèse

Après l'observation du comportement de la comparaison "symétrique" pour les trois classes de changements, on voit se dessiner un arbre de décision assez simple permettant la caractérisation du type de changement qui est intervenu sur une zone donnée et préalablement segmentée (voir figure 3.28). Celui-ci se base simplement sur le respect ou non des deux conditions énoncées dans le cas des disparitions/apparitions.

Ce n'est bien entendu qu'un schéma de principe, puisqu'il reste assez peu précis et repose en particulier sur des moyens de décision qui ne sont pas très clairs ou qui n'ont pas de solution technique à l'heure actuelle (du moins pas sous une forme fiable). De plus, il n'est valable que pour des changements "localement uniques", c'est à dire qu'il ne prend pas en compte la possibilité que plusieurs changements se superposent, comme par exemple un véhicule qui se déplacerait sur un terrain mouvant (cas de la pelleteuse du chantier de Malakoff par exemple - illustrée entre autre en figure 3.25).





<sup>1</sup> dans ce cas, les points « roses » correspondent à l'objet « avant déformation ». Une analyse est donc possible.

<sup>2</sup> il faudrait en théorie segmenter géométriquement l'objet *en entier* pour prendre une décision plus robuste.

<sup>3</sup> on connaît alors les deux positions de l'objet et on peut donc *représenter* le déplacement (symboliquement).

<sup>4</sup> on peut assigner une couleur particulière pour signifier un nouvel objet (ou inversement).

FIG. 3.28 – Schéma de principe du processus de décision permettant la caractérisation d'un changement à partir d'une comparaison "symétrique".

---

## Chapitre 4

# Octree à codage numérique

Nous introduisons dans ce chapitre le concept d'octree, et en particulier un codage particulier et inédit de celui-ci, qui permet d'effectuer des calculs rapides sur des gros nuages de points non structurés.

### 4.1 Introduction

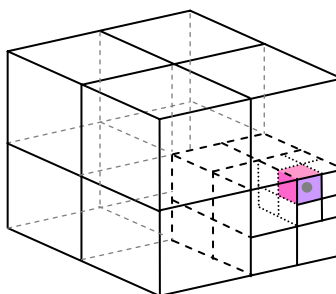


FIG. 4.1 – Principe de l'octree.

Les nuages de points 3D sont des structures très simples qui se résument généralement à une suite de triplets constitués des trois coordonnées de chaque point (associées parfois à certains attributs comme une couleur, une intensité de réflectance ou un vecteur normal, mais cela ne rentre pas en ligne de compte ici). Un nuage de points étant par défaut non structuré, il n'y a rien qui permette d'obtenir rapidement des informations sur le voisinage d'un point et toutes les questions qui en découlent : quel est le nombre de voisins dans une sphère de rayon  $R$  centrée sur le point en question ? Quels sont les  $k$  points les plus proches ? Quelle est la densité du nuage dans une zone précise ? etc.

Si l'on veut pouvoir effectuer des opérations géométriques avancées sur un nuage de

---

points, il faut donc le structurer. Il existe différentes structures dont les plus connues sont sans doute les triangulations (2D) et tétraedrisations (3D) qui sont des maillages respectivement de surface et de volume. Mais les algorithmes permettant leur création sont relativement lents et hasardeux (ceci est particulièrement vrai pour la triangulation surfacique de nuage de points 3D incomplets). Ces structures prennent aussi beaucoup de place en mémoire pour stocker les définitions des arêtes ou autres primitives structurantes.

L'octree [Meagher 1980] est une structure plus simple et moins consommatrice en temps et en mémoire. Elle forme avec le *BSP<sup>1</sup> Tree* et le *KD-Tree*, la famille des arbres de partition de l'espace.

## 4.2 Définition

Un *octree*, ou *arbre octaïdre*, correspond à la division régulière et récursive de l'espace en 8 parties égales (voir figure 4.1). L'équivalent en 2D est un *quadtrees*. En pratique on part du plus petit cube englobant totalement le nuage de points 3D : celui-ci est divisé en 8 sous-cubes qui sont eux-même divisés en 8 et ainsi de suite. Ce processus récursif de subdivision s'arrête quand un niveau maximal prédéfini  $N$  est atteint ou quand il n'y a plus de points à l'intérieur du sous-cube (plus communément appelé *cellule*). Pour un niveau de subdivision  $n \in [1, N]$ , il y a donc potentiellement  $2^{3n}$  cellules mais seules les cellules comportant des points sont représentées en mémoire.

Une telle structure permet de déterminer rapidement les points présents dans une zone précise de l'espace 3D ou les plus proches voisins d'un point donné (qu'on appellera *point-requête* par la suite). Pour faire cela avec un nuage non structuré, il faudrait calculer la distance du point-requête à tous les points du nuage, puis les trier en fonction de cette distance (ce qui est absolument sous-optimal pour des nuages de plus d'une centaine de points). Avec un octree, on considère uniquement la cellule englobant le point-requête, puis ses cellules voisines de manière itérative jusqu'à atteindre le nombre minimal de points voisins requis. Mais il peut y avoir plusieurs points dans une même cellule et les cellules étant cubiques, on doit considérer légèrement plus de points que ceux présents effectivement dans la sphère de proximité du point-requête (nous aborderons ce problème plus en détail par la suite, en section 4.4). Cela fait qu'un octree n'est pas aussi optimal qu'un maillage de ce point de vue. De plus, la vitesse de cette opération dépend beaucoup du niveau d'octree auquel elle est effectuée. En effet, si le niveau de subdivision est faible, les cellules pourront contenir beaucoup de points et le calcul de distance (suivi du tri) va prendre plus de temps. Au contraire, si on travaille à des niveaux trop *profonds* de l'octree, le nombre de cellules voisines à parcourir pour trouver suffisamment de voisins pourra être grand (beaucoup d'entre elles étant vides). Là encore le processus peut en être ralenti. Il faut donc toujours trouver le niveau optimal pour une opération donnée. Ce niveau dépend principalement du nombre total de points mais aussi de la géométrie

---

<sup>1</sup>Binary Space Partition.

locale du nuage (une détermination automatique et exacte du niveau optimal ne semble malheureusement pas possible, mais nous proposerons une heuristique pour tenter de le deviner en section 4.5).

Les octrees sont aussi communément utilisés pour des problèmes de *lancer de rayon* [Havran 1999], de compression [Botsch et al. 2002], [Duguet et Drettakis 2004] et d'affichage des données [Neuez 2002]. On peut en effet localement remplacer les points contenus dans une cellule de l'octree par le centre de la cellule ou le centre de gravité des points (ou une primitive graphique quelconque) sans générer trop d'erreurs dans l'absolu. Cette erreur est limitée par la taille de la cellule et donc par le niveau de subdivision auquel on effectue cette simplification. Visuellement, dans le cadre d'optimisation de l'affichage ou de transmission des données par un réseau, on peut choisir dynamiquement un niveau de subdivision tel que le nombre de points correspondants soit toujours inférieur à une limite (en fonction des capacités du réseau ou de la carte graphique), assurant ainsi la visualisation la plus précise possible sans diminuer la fréquence d'affichage.

Les octrees sont en général implémentés sous forme d'arbres, avec 8 branches à chaque noeud, chaque branche correspondant à un des 8 sous-cubes. Les feuilles de l'arbre correspondent alors aux plus petites cellules contenant des points. Il est aussi possible d'utiliser directement un codage numérique de la position de ces cellules, ce qui évite d'avoir à supporter la structure et la construction de l'arbre. Une structure de ce type est présentée dans la section suivante.

### 4.3 Construction d'un octree par codage numérique

Soit  $S$  un nuage de  $N_S$  points  $P_{i,i=1\dots N_S}$ . Soit  $C$  le plus petit cube englobant  $S$  et  $N$  le niveau maximal de subdivision de l'octree construit à partir de  $C$ .

L'appartenance d'un point à une cellule de l'octree peut être codée numériquement, à l'instar de la technique du *Bucketing* [Edahiro et al. 1984] (appelée aussi *Boxing*) mis à part que l'on désire garder une notion de hiérarchie spatiale, comme dans un octree codé sous la forme d'un arbre. Le *Bucketing* consiste à projeter le nuage de points dans une grille 3D régulière, chaque point étant associé à un indice numérique unique représentant la position de la cellule où il tombe. On retrouve le même principe de codage numérique de la position des cellules d'un octree dans [Friskén et Perry 2002] (avec un codage différent, et moins performant à notre avis).

Chaque point  $P_i$  est associé à un code composé de  $N$  groupes de 3 bits  $b_x b_y b_z$ . Un groupe de 3 bits correspond pour chaque niveau de subdivision à la position du sous-cube englobant le point, relativement au niveau supérieur (Cf. figure 4.2). Il y a 8 sous-cubes, donc 3 bits suffisent à coder leur position. Le choix de la désignation d'un sous-cube par un chiffre donné est fait de telle manière que chaque bit corresponde à une

---

des trois dimensions : il encode la position du point par rapport au plan orthogonal à cette dimension et qui coupe le cube en deux parties égales. Ces groupes de 3 bits (qui sont en quelque sorte des coordonnées locales) sont concaténés pour tous les niveaux de subdivision de 1 à  $N$ , et forment ainsi la coordonnée globale de la cellule. Les groupes des premiers niveaux sont assignés aux bits de poids le plus fort (voir tableau 4.1).

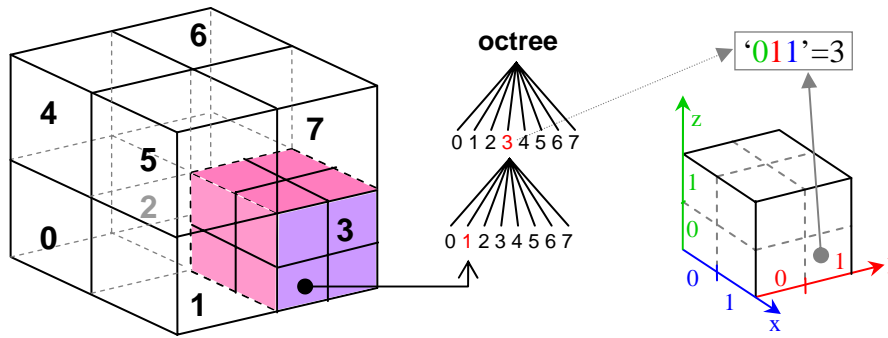


FIG. 4.2 – Principe du codage de la position d’une sous-cellule.

Niveau	1	2	3	...	$N$
Code	$b_x b_y b_z$	$b_x b_y b_z$	$b_x b_y b_z$	...	$b_x b_y b_z$
Poids	fort	...	...	...	faible

TAB. 4.1 – Codage numérique de la position d’une cellule de l’octree

On obtient ainsi pour chaque point un code numérique désignant la cellule à laquelle il appartient de manière unique et ce pour tous les niveaux de subdivision. Le calcul du code est très simple et rapide puisqu’il ne correspond qu’à  $3N$  tests et au pire  $3N$  additions/soustractions par point. La partie codage de l’algorithme est donc linéaire ( $\sim O(N_S)$ ).

La liste des codes est ensuite triée parallèlement à la liste des points, pour garder une cohérence entre codes et points sans avoir à maintenir une table d’équivalence des indices en mémoire qui prendrait trop de place. Le tri permet une recherche efficace d’un code donné et donc d’une cellule par dichotomie. L’autre avantage du tri est de permettre la détermination de tous les points présents dans une même cellule à un niveau  $n \in [1, N]$ . Pour cela, il suffit de ne considérer que la partie de leur code formée par les  $3n$  bits de poids le plus fort. En effet, les sous-cellules de cette cellule partagent toute cette partie du code par construction, et se retrouvent de plus les unes à la suite des autres dans le tableau grâce au tri. La phase de tri peut être faite en  $\sim O(N_S \cdot \log(N_S))$  (voir la remarque qui suit à ce sujet).

Remarque : au sujet du tri, nous avons modifié la version de l’algorithme *sort* de la STL<sup>2</sup>

<sup>2</sup>Standard Template Library (bibliothèque d’algorithmes génériques en C++ garantissant des performances optimales).

pour pouvoir trier un tableau et en modifier parallèlement un autre. L'algorithme de tri utilisé dans la STL est aujourd'hui *Introsort*[Musser 1997] qui est une version améliorée de *Quicksort*[Hoare 1962] qui évite les cas dégénérés de ce dernier (*Quicksort* est en effet sous-optimal - quadratique - lorsqu'il est appliqué à un tableau déjà trié). *Introsort* est au pire logarithmique et est toujours au moins aussi rapide que *Quicksort* dans le cas général.

Un codage numérique de l'octree présente comme avantages majeurs :

- une très grande simplicité des calculs qui rend sa création très rapide (et très facile à implémenter), même sur des nuages de points imposants.
- L'insertion ou la suppression d'un point ne nécessite pas le re-calcul de toute la structure puisqu'il suffit d'insérer ou supprimer un code dans la liste qui est déjà triée (à condition, dans le cas de l'insertion, que la boîte englobante du nuage ne change pas).
- La mémoire occupée par la structure ne dépend que du niveau maximal de subdivision et du nombre de points (pour un octree de niveau 10, on a besoin de 30 bits par code qu'on peut donc stocker sous forme d'entiers longs de 32 bits. Les 2 bits restants peuvent servir de *flags* (pour l'affichage du point par exemple).

## 4.4 Recherche des voisins

La recherche de voisins avec un octree à codage numérique est un peu plus complexe que sa construction.

### Voisinage de cellules

Localiser une cellule donnée via son code est très rapide, grâce à une simple dichotomie. Par contre, ses cellules voisines ne sont pas voisines dans la liste des codes (ce sont les sous-cellules qui sont voisines dans cette liste). Il reste cependant possible de déterminer leur code à partir de celui de la cellule en cours par simple calcul. Il faut ensuite refaire une dichotomie pour trouver cette nouvelle cellule dans la liste (ou ne pas la trouver, auquel cas elle est vide). Deux méthodes permettant de calculer le code d'une cellule voisine sont proposées ci-dessous.

Soit  $n \in [1, N]$  le niveau d'octree auquel l'opération est effectuée. En ne considérant que les cellules voisines directes (c.a.d. à plus ou moins une case de la cellule en cours), il est possible d'appliquer des additions et soustractions binaires sur la partie du code constituée des  $3n$  bits de poids lourd. Soit  $(p, q, r)$  les coordonnées de la cellule en cours dans la grille, et  $(p + i, q + j, r + k)$  avec  $\{i, j, k\} \in [-1, 1]^3$  celles d'une cellule voisine (on a donc aussi la condition  $|i| + |j| + |k| > 0$ ). Selon chaque dimension  $dim = \{x \mapsto 0; y \mapsto 1; z \mapsto 2\}$

---

on extrait les  $n$  bits de poids les plus forts qui lui correspondent, soient  $\{b_{3l+dim}, l \in [0, n-1]\}$  ( $b_l$  étant le  $l$ -ème bit du code en partant de la gauche). Sur ces groupes de  $n$  bits, qui correspondent à la position de la cellule selon une dimension, on applique une décrémentation si  $i, j$  ou  $k$  est strictement négatif, une incrémentation si  $i, j$  ou  $k$  est strictement positif et on ne fait rien si ils sont nuls. Les décrétements et incrétements correspondent simplement en pratique à des décalage d'une case (vers la gauche ou vers la droite) par rapport à la position courante et selon la dimension considérée. Après avoir traité chaque dimension, on recompose le code de la cellule voisine en *entrelaçant* les trois groupes de  $n$  bits. Bien que compliqué sur le papier, l'implémentation de cette procédure est plutôt rapide. En effet, il n'est pas nécessaire de décomposer réellement le code en 3 parties pour faire les calculs.

L'autre solution, moins rapide mais plus simple, a surtout l'avantage de permettre de déterminer des voisins non-directs (donc avec  $|i|, |j|, |k| > 1$ ). En reprenant les notations ci-dessus, on a grâce au principe de construction de l'octree,  $p, q, r = \sum_{l=0}^{n-1} b_{3l+dim} * 2^{n-1-l}$ . Donc on peut en déduire les coordonnées du voisin avec 3 additions ( $p+i, q+j, r+k$ ) et faire le processus inverse pour retrouver le code de la cellule voisine.

Dans les deux cas le code calculé est tronqué, étant donné qu'on obtient seulement ses  $3n$  derniers bits (l'autre partie du code n'est de toute façon pas utile et ne correspond à rien puisque l'on travaille au niveau  $n$ ). La seule réelle différence entre les deux méthodes est au niveau des calculs qui sont d'une part des opérations sur des bits (donc rapides mais limitées) et d'autre part des calculs sur des entiers (donc un peu plus lents, mais plus simples à manipuler).

## Voisinage de points

Une autre opération, un peu plus complexe, consiste à rechercher les  $k$ -plus proches voisins d'un point-requête. Tout le problème vient du fait que l'on considère des voisinages de cellules cubiques alors que l'on travaille avec la distance euclidienne qui induit une notion de voisinage sphérique pour les points. De plus le point-requête n'est pas forcément au centre d'une cellule (voir figure 4.3). Il est donc possible qu'un point situé dans le voisinage de cellules à l'ordre  $j+1$  soit plus proche du point-requête que ceux qui sont inclus dans le voisinage à l'ordre  $j$ . Ainsi, il arrive que la distance d'un point au point-requête soit supérieure au rayon de la plus grande sphère centrée sur le point-requête et totalement inscrite dans le voisinage de cellules considéré (les cercles inscrits dans la figure 4.3). Dans ce cas, on ne peut pas affirmer que ce point fait partie des  $k$ -plus proches voisins sans avoir au préalable calculé la distance des points inclus dans le voisinages d'ordre supérieur.

On commence donc par déterminer quels points se trouvent dans la même cellule que le point-requête. On calcule la distance de tous ces points au point-requête, on les trie en fonction de cette distance et on garde les  $k$  plus proches, pour autant qu'il y ait au moins

$k$  points dans la sphère inscrite sus-mentionnée. Si ce n'est pas le cas, il faut augmenter la taille du voisinage de cellule et répéter les opérations de calcul de distance et de tri jusqu'à ce que la distance du  $k$ -ième voisin soit inférieure au rayon de la sphère inscrite dans le plus grand voisinage considéré.

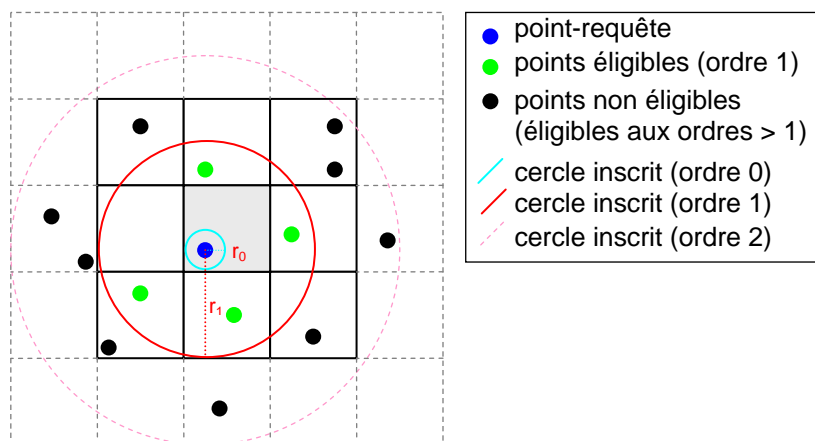


FIG. 4.3 – Principe de la détermination des plus proches voisins (représentation en 2D).

Voici la formule qui permet de calculer le rayon  $r_j$  de la sphère inscrite pour un ordre de voisinage  $j$  ( $j \geq 0$ ). Soit  $P$  le point-requête,  $C$  le centre de la cellule incluant  $P$  et  $l$  la dimension d'une cellule de l'octree au niveau de subdivision courant.  $r_j$  est tout simplement égale à la distance entre  $P$  et la face du cube délimitant le voisinage d'ordre  $j$  la plus proche :

$$r_j = \min_{dim \in \{x,y,z\}} \left[ \left( j + \frac{1}{2} \right) * l - |P_{dim} - C_{dim}| \right] \quad (4.1)$$

Il est possible d'optimiser le processus lorsque celui-ci est appliqué successivement à plusieurs points d'une même cellule d'octree. En effet, déterminer la bonne taille de voisinage par itération est une opération pénalisante. Or, il y a beaucoup de chances que la taille de voisinage nécessaire à la détermination des plus proches voisins d'un point d'une cellule soit égale ou au moins très proche (à plus ou moins une cellule près) de celles des autres points de la même cellule. Une fois qu'on a déterminé la valeur pour un point, on peut donc appliquer directement l'algorithme de recherche des plus proches voisins aux points de la même cellule avec cette même valeur. On peut aussi au passage se souvenir des points inclus dans ce voisinage, économisant là encore un temps précieux. Le coût supplémentaire nécessaire pour stocker ces valeurs (principalement du temps de copie des valeurs en mémoire) est minime à côté de celui gagné par ce biais. Cette approche est donc valable dès qu'on recherche les voisins de plus de deux points d'une même cellule. Seuls le calcul des distances puis le tri (là encore effectué avec *Introsort* - Cf. remarque en fin de section 4.3) doivent être effectués pour chaque point-requête.

Remarque : on effectue la recherche de voisins à travers l'octree à niveau de subdivision



constant, plutôt que d'utiliser une approche multi-échelle, plus courante lorsque l'on travaille avec un octree codé sous forme d'un arbre. Nous justifions ce choix d'une part à cause de notre méthode de codage numérique (qui *met à plat* la structure de l'octree), qui pénalise fortement l'opération qui consiste à déterminer pour une cellule donnée à un niveau donné la cellule qui l'englobe au niveau supérieur. Cela empêche donc d'utiliser efficacement des approches *bottom-up* (les approches *top-down*, sont par contre relativement efficaces, mais ne sont pas utiles pour la détermination de voisinages). L'autre raison est que cette recherche à niveau constant est plus rapide lorsque les deux nuages comparés sont majoritairement proches (on compare globalement moins de points). En contrepartie, la méthode est plus lente lorsque elle traite des points très éloignés de l'autre nuage (ce qui est rarement le cas, puisqu'une analyse géométrique se limite normalement aux zones communes). Elle nécessite aussi la détermination préalable du niveau de subdivision de l'octree auquel va être exécutée la recherche et qui offre les meilleures performances. Il est important de noter que le niveau de subdivision de l'octree auquel est effectué le calcul ne joue pas sur la qualité du résultat - qui est toujours le même - mais uniquement sur la vitesse du calcul. Nous proposons dans la section suivante une heuristique permettant de déterminer de manière simple ce niveau optimal dans une majorité des cas.

## 4.5 Heuristique de détermination du niveau d'octree optimal pour la détermination de multiples voisinages

Pour déterminer précisément le niveau de subdivision optimal pour un calcul de détermination des plus proches voisins, il faudrait faire une analyse spatiale, au moins approximative, de la position relative des points des deux nuages. Or une telle opération nécessiterait un temps de traitement supplémentaire très important par rapport au temps qu'elle permettrait de gagner. Néanmoins, si on applique la recherche des  $k$ -plus proches voisins pour un nombre relativement important de points-requêtes (ce qui sera vrai en général par la suite), il est possible de modéliser le comportement temporel du processus global en fonction du niveau de subdivision et ainsi déterminer le niveau le plus "rapide".

Pour cela, considérons l'ensemble des  $N$  points d'un nuage  $S_1$  auxquels nous voulons appliquer l'extraction de  $k$  voisins dans un nuage  $S_2$  ( $S_1$  et  $S_2$  peuvent désigner le même nuage). On calcule alors les octrees de  $S_1$  et  $S_2$  à partir du même cube de départ qui est le plus petit cube englobant les deux nuages (c'est automatiquement vrai si  $S_1$  et  $S_2$  sont un unique et même nuage, sinon cette contrainte ne posera pas de problème pour les algorithmes présentés au chapitre 2). Puisque les octrees des deux nuages sont calculés à partir du même cube de départ, leurs cellules sont spatialement équivalentes. Ainsi, à un niveau  $n$  de subdivision, une cellule de l'octree du nuage  $S_1$  qui aura le même code qu'une cellule de l'octree du nuage  $S_2$  (ou plutôt les mêmes  $3n$  premiers bits) aura les mêmes limites dans l'espace.

On détermine aussi lors du calcul des octrees leurs nombres exacts de cellules pour tous les niveaux de subdivision. On peut alors calculer les densités moyennes des cellules de chaque octree :  $\overline{N}_{i,j}$  où  $i$  désigne le nuage et  $j$  le niveau de subdivision.

Voici maintenant comment nous avons construit notre modèle de comportement temporel. L'algorithme de recherche des  $k$  plus proches voisins, tel que décrit dans la section précédente, se décompose en deux phases distinctes : d'une part la détermination des voisinages de cellules et des points contenus dans ces voisinages et d'autre part le calcul des distances suivi de leur tri pour déterminer les  $k$ -plus proches voisins.

La première phase se fait par recherche dichotomique dans la liste des codes pour chaque cellule formant le voisinage de cellule. Elle n'est faite que si la cellule de l'octree de  $S_1$  qui englobe le point-requête n'a pas d'homologue dans l'octree de  $S_2$ , ou bien si le nombre de points "éligibles" dans cette cellule homologue est inférieur à  $k$ .

La seconde phase consiste principalement à calculer les distances entre le point-requête et les points potentiellement voisins (déterminés lors de la phase précédente), puis à les trier. Elle a une complexité proportionnelle à ce nombre de voisins potentiels (multiplié par une constante proche de 6 à cause des 3 multiplications et des 3 soustractions nécessaires au calcul de la distance entre deux points en 3D) plus ce même nombre multiplié par son logarithme (et multiplié par une constante proche de 2 à cause des tests nécessaires au tri). Soit en somme une complexité qui évolue en  $\sim \eta * (3 + \log(\eta))$ , si  $\eta$  est le nombre de voisins potentiels.

Pour ce qui concerne les  $N$  points-requêtes, deux cas se présentent. Soit ils font partie d'une cellule qui a une homologue dans l'octree de  $S_2$ , et  $\overline{N}_{2,n} \geq k$ , auquel cas la première phase est réduite à la détermination des points présents dans la cellule homologue (ce qui est négligeable). Le calcul des distances puis le tri se font alors avec  $\overline{N}_{2,n}$  voisins potentiels en moyenne (cas "1"). Soit leur cellule englobante n'a pas d'homologue, ou alors  $\overline{N}_{2,n} < k$ , auquel cas la première phase va prendre beaucoup plus d'importance, selon une tendance qu'il est par contre difficile d'estimer mais qui va principalement dépendre des voisinages de cellules, et donc basiquement du nombre de cellules concernées (cas "2").

Finalement, on considère l'ensemble des cellules de l'octree de  $S_1$  qui englobent les  $N$  points-requêtes. On les sépare en deux sous-groupes, à savoir celles qui ont des homologues dans l'octree de  $S_2$  et dont le nombre de points inclus est supérieur à  $k$  (soit  $N_h$  leur nombre) et les autres (soit  $N_{\overline{h}}$  leur nombre). Ceci se fait très rapidement par comparaison des deux listes des codes des cellules. En somme, on veut estimer à partir de la proportion de cellules non homologues quelle importance va prendre la phase de détermination des voisinages de cellules par rapport à celle du calcul des distances suivivi du tri.

Toutes les cellules vont suivre au minimum le comportement type du cas "1", mais les  $N_{\overline{h}}$  cellules qui n'ont pas d'homologue (ou pas assez de points inclus) vont nécessiter un traitement supplémentaire qui représente un surcoût important pour l'algorithme. On peut donc établir une loi du comportement temporel de l'algorithme, pour un niveau de

---

subdivision  $n$ , telle que celle-ci :

$$t_n = t_{dist} * [N * \bar{N}_{2,n} * (3 + \log(\bar{N}_{2,n}))] + t_{vois} * f(N_{\bar{h}}) \quad (4.2)$$

$t_{dist}$  et  $t_{vois}$  sont des constantes liées aux temps de calcul effectif des différentes opérations nécessaires à chaque phase (et indépendantes du nombre de points) et  $f$  représente la tendance d'évolution du temps de calcul de la phase de recherche des voisinages en fonction de  $N_{\bar{h}}$ . Empiriquement, en mesurant sur différents cas bien distincts les proportions de temps passées par l'algorithme d'une part à calculer les distances et trier les points, et d'autre part à chercher des cellules voisines, on a pu se rendre compte que  $f$  est soit linéaire, soit quadratique. Ceci s'explique par le fait que les décalages globaux entre les cellules des deux octrees peuvent venir soit d'une translation entre les deux listes (auquel cas la taille du voisinage de cellules à extraire est globalement toujours la même), soit d'un éloignement progressif d'une partie de  $S_1$  par rapport au nuage  $S_2$ , (auquel cas la taille des voisinages de cellules à extraire augmente progressivement en fonction de cet éloignement et le nombre de cellules concernées, ainsi que le nombre de points potentiellement voisins, augmente au moins quadratiquement). Il subsiste donc une inconnue mais néanmoins, si le nombre maximal de niveaux de subdivision reste faible (autour de 10 typiquement), la dérive due à l'approximation de  $f$  par une fonction linéaire au reste en général faible et l'erreur ainsi causée par l'heuristique est au pire d'un ou deux niveaux de subdivision.

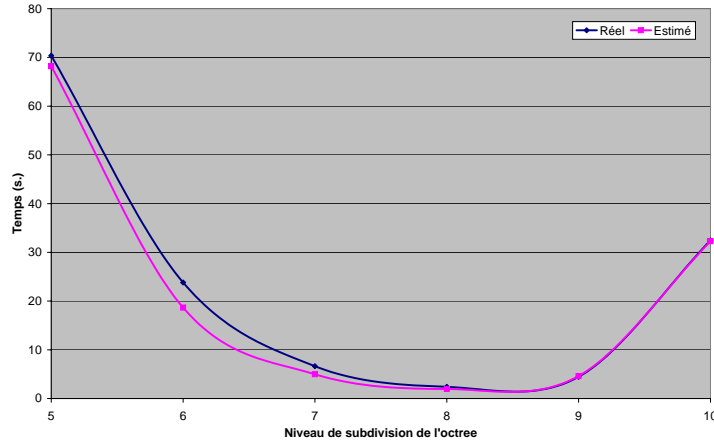


FIG. 4.4 – Temps de calcul réel et estimé pour l'extraction du plus proche voisin ( $k = 1$ ).  
*Le nuage comparé est composé de 625931 points, et le nuage de référence (de forme proche) de 627339 points.*

L'heuristique consiste donc simplement à appliquer la formule 4.2 pour chaque niveau de subdivision, et à choisir celui qui donne le meilleur "temps". Une fois les octrees

calculés, le temps passé à déterminer les différents paramètres nécessaires à l'estimation des "durées" pour chaque niveau de subdivision est minime en rapport au temps de calcul des distances.

A défaut d'une solution exacte, cette heuristique fournit une première approximation du niveau optimal, généralement bonne (voir figure 4.4), ce qui évite ainsi à l'utilisateur d'avoir à connaître précisément le fonctionnement de l'algorithme, ou même simplement d'avoir à tester différentes valeurs du niveau de subdivision pour trouver la plus rapide pour une configuration donnée. Il reste cependant à améliorer son fonctionnement (grâce à une meilleure analyse des mécanismes de l'algorithme ou en utilisant une mesure de forme globale - et rapide - pour mieux estimer le type de décalage global qui peut exister entre les deux nuages).

Remarque : avec une variation  $f$  linéaire, et pour l'implémentation que nous avons faite des différents algorithmes et structures intervenant dans la recherche des voisins, le rapport  $\frac{t_{vois}}{t_{dist}}$  s'est avéré très stable, autour de  $10^3$ . Ce nombre dépend principalement de l'implémentation mais aussi, sans doute, de la génération du processeur utilisé. On utilise ce rapport dans la formule 4.2 plutôt que les valeurs mesurées de  $t_{vois}$  et  $t_{dist}$ .

---



# Chapitre 5

## Applications

Bien que les méthodes présentées dans les chapitres précédents aient été développées en pensant principalement aux applications d'aide à la réalisation de documentation TQC et de suivi de chantier, il est possible de les utiliser pour d'autres applications de détection de changement. Nous présentons donc dans ce chapitre différents exemples d'application de ces méthodes, en commençant par les plus "pragmatiques" (dans le sens où elles répondent à un besoin actuel et sont effectivement utilisées dans ce sens) jusqu'à des méthodes plus en *avance de phase*, voire encore totalement spéculatives.

### 5.1 Aide à la réalisation de documentation T.Q.C.

L'optimisation des opérations de maintenance est stratégique pour une entreprise comme EDF. Cette optimisation nécessite une connaissance très précise de la géométrie des installations. Le domaine d'activité correspondant à l'acquisition et à la mise à jour de cette connaissance est désigné par l'acronyme T.Q.C. qui correspond aux mots *Tel Que Construit* (on peut trouver aussi l'équivalent T.Q.E. pour *Tel Qu'Existant*). Ce domaine regroupe en pratique toutes les activités consistant à établir ou à actualiser des modèles géométriques d'une installation à partir de mesures faites sur le terrain. Ce type de modèles se substitue aux modèles ou aux plans théoriques utilisés lors de la construction et qui ont été établis par un architecte ou un cabinet d'étude. Cela revient en somme à corriger ou à refaire les plans pour qu'ils correspondent à la réalité de l'usine *telle qu'elle est construite*, contrairement aux plans initiaux qui ne sont jamais parfaitement respectés ou aux plans trop anciens qui ne prennent pas en compte les nombreuses modifications apportées à l'ouvrage depuis sa création.

Cette discipline est proche de la rétro-ingénierie des pièces manufacturées. C'est l'opération qui consiste à établir un modèle d'une pièce produite pour le comparer au modèle théorique. Cela permet d'appliquer des procédures de contrôle de qualité ou de sécurité,

---

ou encore de diagnostiquer des problèmes dans la chaîne de production.

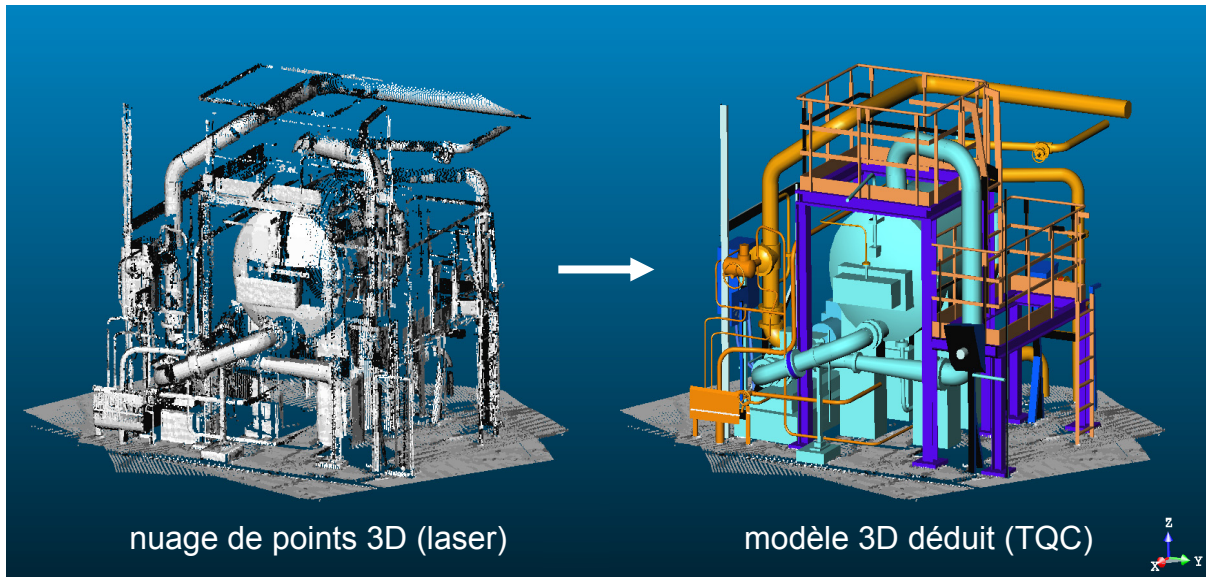


FIG. 5.1 – Illustration du principe de modélisation 3D d'une installation industrielle à partir de mesures ponctuelles.

*Ici les points sont obtenus à l'aide d'un scanner laser Soisic de la société Mensi (exemple issu de leur site internet).*

La documentation T.Q.C. est utilisée pour la planification d'opérations de maintenance, de modification (*revamping*) ou de démantèlement d'installations. Cela est courant dans les complexes industriels de grande envergure subissant de nombreuses évolutions ou possédant un grand nombre de composants imbriqués (raffineries, plate-formes pétrolières, centrales énergétiques, chaînes de production, etc.). Tant qu'elle reposait sur des mesures traditionnelles, la réalisation d'une telle documentation était très fastidieuse et donc rare. Elle a pris beaucoup d'essor avec l'apparition des techniques de photogrammétrie terrestre puis plus récemment de relevé laser (ces techniques seront détaillées dans la section 1.5). Ces techniques permettent l'acquisition d'un grand nombre de mesures tridimensionnelles à distance et avec un temps de présence sur le terrain restreint, permettant ainsi d'atteindre des zones dangereuses ou difficilement accessibles pour un homme.

En règle générale, la simple mesure ne suffit pas. Elle n'est que la base d'une seconde opération qui consiste à construire le modèle par assemblage de primitives géométriques élémentaires calées au plus près des points mesurés (figure 5.1). Cette opération se fait à l'aide d'outils de C.A.O. (*Conception Assistée par Ordinateur*) spécifiques qui simplifient le travail de l'opérateur à l'aide d'approches semi-automatiques faisant intervenir principalement des algorithmes de recalage de primitives 3D sur des nuages de points ou encore de segmentation automatique. Le travail reste tout de même assez long mais peut être effectué dans des délais raisonnables par une équipe d'opérateurs entraînés. Ces outils connaissent un développement parallèle à celui des techniques de mesure citées pré-

cédemment et profitent de nombreux travaux de recherche. On peut citer en particulier la thèse de Thomas Chaperon intitulée « *Segmentation de nuage de points 3D pour la modélisation automatique d'environnements industriels numérisés* » (École des Mines de Paris) soutenue en 2002 [Chaperon 2002].

Cette opération de modélisation ou de mise à jour de modèles 3D est délicate. Que ce soit sur le terrain ou en observant un modèle tridimensionnel, plusieurs facteurs limitent en effet la perception et surtout la quantification des différences par un humain :

- la faiblesse du déplacement par rapport à la taille de l'objet,
- le nombre très important d'objets identiques ou ayant des formes proches,
- et enfin des couleurs ou des textures qui peuvent perturber la perception des formes et des distances.

Avec le développement des applications de T.Q.C. pour des environnements industriels denses et complexes sont donc apparus de nouveaux besoins de détection de changement géométrique :

- le contrôle de la phase de modélisation, par comparaison du modèle produit avec les données ponctuelles tridimensionnelles qui ont permis sa création (cas des données laser en particulier). Le but est de contrôler l'oubli d'objets importants, les erreurs de recalage de primitives, etc.
- la vérification rapide de la concordance du modèle avec la réalité avant la planification d'une intervention exceptionnelle. Les mesures ne sont alors faites que dans la zone précise de l'intervention.
- la minimisation et l'optimisation du travail nécessaire à la réalisation d'une nouvelle modélisation :
  - si celle-ci est effectuée sur la même usine, on veut savoir quelles zones de l'usine ont subi des changements et impliquent donc une correction du modèle (on comparera alors le nouveau nuage de points à l'ancien modèle),
  - si celle-ci est effectuée sur une autre usine ayant théoriquement la même configuration, on veut pouvoir récupérer un maximum d'informations du modèle de la première usine (auquel cas on comparera celui-ci au nuage de points de la deuxième usine, cette comparaison s'accompagnant si possible d'une comparaison directe des deux nuages pour une analyse plus détaillée).

La figure 5.2 présente un processus typique de contrôle de réalisation d'un modèle TQC à partir d'un nuage laser acquis en intérieur de centrale. L'étape de comparaison est négligeable en terme de temps de travail mais est pour autant cruciale et critique vis-à-vis du processus global. *CloudCompare*, le logiciel de démonstration de cette thèse, est d'ores et déjà utilisé dans le cadre de ce processus à EDF, actuellement en phase de test, et devrait être à court terme *mis en production*.

---



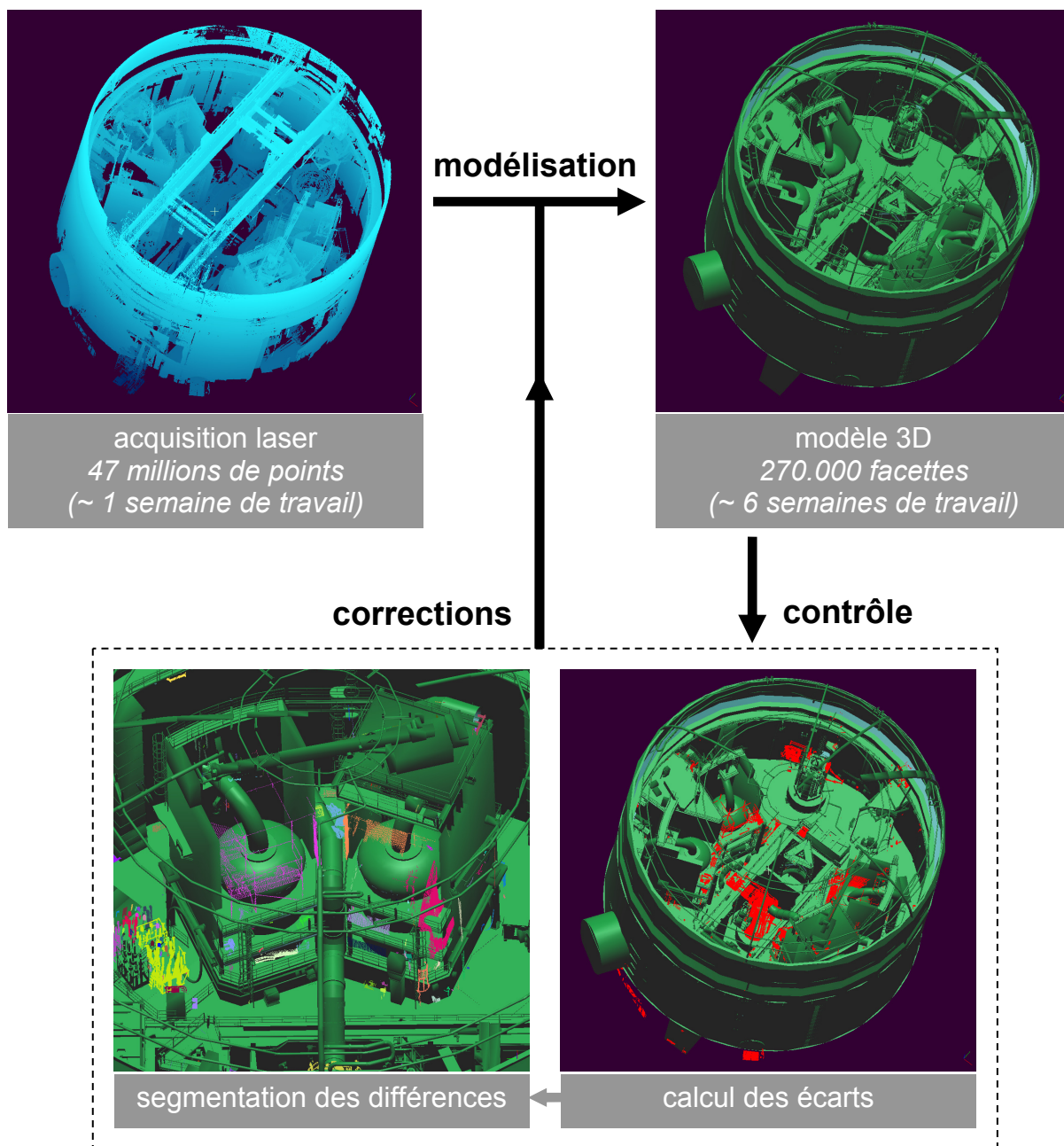


FIG. 5.2 – Illustration du processus de contrôle de réalisation de modèle TQC.

## 5.2 Suivi de chantier

Toujours dans le but d'optimiser mais surtout de contrôler la qualité des travaux réalisés, une autre application où la détection de changement présente un fort intérêt est celle du suivi de chantier. On désigne par chantier toute opération de construction, modification ou démantèlement d'installations. L'idée est donc ici de mesurer régulièrement

et rapidement l'état d'un tel chantier, d'une part pour évaluer l'avancement des opérations et d'autre part pour contrôler le respect des plans lors des phases de construction ou de modification. Un contrôle régulier permet de détecter rapidement des dérives par rapport aux spécifications et permet donc de les corriger plus rapidement. On assure une meilleure qualité globale de la structure et on valide plus rapidement la conformité des différentes étapes clés du chantier. La contrainte de rapidité prend ici toute sa dimension car l'opération est répétée souvent et doit pouvoir être réalisée *à la volée*, pour diagnostiquer rapidement une zone précise du chantier (avant d'effectuer une opération imprévue par exemple).

Remarque : le *suivi de chantier* est un besoin fortement ressenti depuis plusieurs années, mais qui n'a pas encore trouvé de réponse concrète. Le terme "suivi de chantier" est d'ailleurs plutôt utilisé dans un sens plus général, et ne se borne pas simplement à la mesure de l'avancement et des écarts entre attendu et réalisé. Il englobe aussi les activités de gestion et de localisation des différents appareils impliqués dans les travaux, des ressources, etc. Cela correspondrait donc plutôt à une solution logicielle regroupant différentes applications permettant la gestion globale d'un chantier.

La partie "contrôle géométrique" du suivi de chantier à laquelle nous nous sommes intéressés, se rapproche des techniques utilisées dans le domaine de l'exploitation de carrières ou des chantiers de déblaiement par exemple (dont le but est généralement de contrôler les quantités de matière extraite et assurer ainsi une facturation plus précise). Ces techniques se basent sur deux mesures effectuées respectivement au début et à la fin du chantier. Des modèles tridimensionnels sont créés à partir de ces deux jeux de données, et sont ensuite soustraits l'un à l'autre pour calculer précisément les volumes extraits. Les mesures sont généralement effectuées par photogrammétrie (aérienne ou terrestre, en fonction des dimensions du chantier ou de la carrière) car il n'y a pas de contraintes de rapidité en général et les mesures ne sont effectuées que deux fois (et la comparaison une seule fois).

Néanmoins, les capteurs lasers sont de plus en plus utilisés dans le domaine du génie civil (ponts, autoroutes, etc.) et ils permettent d'envisager de nouvelles méthodes de travail (à l'instar du GPS embarqué sur les engins de chantier) ou de contrôle des réalisations. On peut ainsi envisager des *schémas* de comparaison de données 3D :

- D'une part, pour évaluer la concordance entre le travail effectué et les plans de construction, on doit comparer la mesure de l'état final à ces plans (généralement des modèles C.A.O.). On doit donc comparer un nuage de points à un modèle théorique.
  - D'autre part, dans le cas de l'évaluation de l'avancement du chantier, deux cas se présentent :
    - on peut tout d'abord comparer chaque état intermédiaire à l'état initial (qui prendra en général lui aussi la forme d'un modèle 3D). On se ramène donc au cas évoqué ci-dessus.
    - ou on peut avoir à comparer deux états intermédiaires entre eux, auquel cas on devra comparer directement deux nuages de points.
-

Dans ce domaine par contre, nous n'en sommes encore qu'à l'état de propositions. L'utilisation de telles techniques en conditions réelles n'est pas encore effective et nécessiterait la confrontation de besoins réels et pratiques avec des solutions techniques existantes, associées par exemple aux travaux présentés dans ce manuscrit.

## 5.3 Contrôle géométrique d'ouvrages de génie civil

### 5.3.1 Cadre

Une des principales applications parmi les opérations de comparaison géométrique tridimensionnelle (et qui est d'ailleurs peut-être la plus *évidente*) est celle du contrôle dimensionnel et structurel. Le but est de contrôler la conformité de la forme ou de la surface d'un objet par rapport à un modèle théorique ou à une forme antérieure, et ce généralement pour des soucis de qualité ou de sécurité.

L'industrie manufacturière (automobile, plastique, métallurgique, etc.) a été pionnière dans ce domaine et a contribué très largement à son développement. De nombreux systèmes qui reposent sur différentes technologies (mesure par contact, photogrammétrie, scanning laser, etc.) sont aujourd'hui disponibles. Le but général est de tester la forme d'une pièce usinée ou moulée par rapport à un modèle théorique. La décision finale concerne généralement le rejet de la pièce (ou d'un lot de pièces) mais la détection et si possible le diagnostic des défauts potentiels provenant des machines impliquées dans le processus de fabrication peut être aussi importante. La décision repose sur des seuils de tolérance définis à partir de critères de qualité ou de sécurité. Le modèle théorique peut correspondre à des plans CAO réalisés par un bureau d'étude ou à une forme moyenne obtenue à partir d'un grand nombre d'échantillons et censé représenter le comportement *normal* de la machine. La comparaison implique donc, dans la majorité des cas, des pièces uniques dont la mesure et la représentation 3D qui en est déduite sont très précises et complètes. Les méthodes de comparaison correspondantes sont alors simples, efficaces et robustes (Cf. section 1.4.1). Les logiciels sont aboutis. Les capacités de progrès de ces techniques reposent donc plutôt sur l'amélioration de la précision et de la rapidité des systèmes de mesure.

L'autre secteur moteur du domaine est celui des industries de génie civil et de géotechnique qui s'intéressent à des objets ou à des groupes d'objets beaucoup plus grands avec des précisions généralement moindres (moindres dans l'absolu - elle peuvent être équivalentes à celles rencontrées dans l'industrie si on les rapporte à la taille des objets). Les applications courantes sont le suivi de chantier (excavation, construction, extraction et transport de matières premières, etc.) et le contrôle géométrique d'ouvrages de génie civil (ponts, barrages, tunnels, routes, etc.). C'est le domaine des géomètres et topographes, qui réalisent des mesures ponctuelles sur le terrain (à l'aide d'instruments très précis comme des tachéomètres, télé-mètres lasers, etc.) ou des mesures plus globales (par des

---

techniques comme la photogrammétrie, la bathymétrie, etc.). Dans tous les cas, le résultat est un ensemble d'informations ponctuelles 3D précises mais généralement peu denses, dont la mesure nécessite des temps d'acquisition et de traitement importants. Le travail de comparaison est ensuite logiquement poursuivi par les mêmes personnes, puisque l'on reste dans le domaine de la géométrie et de la mesure de précision. Les méthodes de comparaison restent fortement basées sur des opérations manuelles et sur l'expérience de l'opérateur (le tout avec des logiciels encore peu adaptés, qui sont souvent des extensions de logiciels standards de CAO, comme Autocad d'*Autodesk* ou Microstation de *Bentley*). Tous ces facteurs ralentissent l'obtention de résultats et rendent très difficile le contrôle et la compréhension, par un non-professionnel, du travail réalisé. L'apparition des scanners laser depuis une dizaine d'années commence à transformer la profession et ouvre la voie à de nouvelles habitudes de travail ainsi qu'à de nouvelles applications.

C'est dans ce contexte et selon cette optique que la division R&D et la division topographique d'EDF s'intéressent à une application de contrôle géométrique appliquée au suivi d'un aéroréfrigérant. Cet imposant ouvrage de génie civil, d'une hauteur avoisinant 200 mètres, est entièrement constitué de béton et a approximativement la forme d'un hyperboloïde de révolution (voir figure 5.3). Il permet de refroidir la température du circuit d'eau chaude secondaire d'une centrale. Ce type d'ouvrage n'est pas spécifique aux centrales nucléaires et est fréquemment utilisé dans la pétrochimie (avec des tailles certes inférieures). Sa surface peut être considérée comme une coquille d'oeuf à l'échelle de l'ouvrage puisque son épaisseur varie entre 60 cm à la base et 30 cm au sommet. Elle subit des déformations élastiques rapides dues à l'action du vent et de la chaleur ambiante, et des déformations irréversibles plus lentes dues au vieillissement. Ces ouvrages sont constamment suivis et inspectés pour réparer les défauts structurels (fissures, etc.) et aussi tenter de prédire et surtout prévenir un effondrement potentiel<sup>1</sup>.



FIG. 5.3 – Aéroréfrigérant.

En pratique, des mesures photogrammétriques de l'ouvrage sont effectuées réguliè-

---

<sup>1</sup> bien que cela ne présente aucun risque vis-à-vis de l'activité *nucléaire* de la centrale, mais plutôt parce qu'il est très long et difficile de reconstruire un tel ouvrage (la centrale étant bien sûr paralysée durant les travaux), sans compter l'impact psychologique très négatif auprès du public.

---

rement et une entreprise spécialisée réalise une carte des évolutions. De leur côté les chercheurs et les topographes d'EDF ont testé les méthodes classiques de comparaison des données photogrammétriques pour valider et comprendre le travail des prestataires. Cette étude a fait l'objet d'un stage (nous évoquerons rapidement ses résultats un peu plus loin). Les topographes d'EDF ont également effectué des relevés laser d'un aéroréfrigérant à deux dates différentes (avec un intervalle de 5 mois), pour étudier la possibilité d'effectuer l'opération de comparaison directement sur ces données, par une méthode restant à définir. Il a donc paru naturel d'appliquer à ces données les méthodes de comparaison développées dans le cadre de cette thèse.

### 5.3.2 Confrontation des méthodes de comparaison

#### Comparaison à partir de données photogrammétriques

Nous présentons ici une technique de comparaison de deux nuages de points acquis par photogrammétrie. Elle a été étudiée dans le cadre d'un stage au sein de l'équipe CAO d'EDF R&D, effectué par Salma Bougacha (alors élève de l'École Centrale Paris et du master M.V.A.<sup>2</sup> de L'École Normale Supérieure de Cachan). Le but du stage était l'étude des techniques d'analyse des défauts et déformations sur un aéroréfrigérant.

Les nuages de points photogrammétriques d'aéroréfrigérants dont EDF dispose sont très peu denses (quelques milliers de points répartis régulièrement sur un ouvrage dont la surface est supérieure à  $50\,000\text{ m}^2$ ). L'application de la formule 2.6 (section 2.3.1) avec  $A = 50000$  et  $N = 8000$ , nous donne une erreur quadratique moyenne due à l'échantillonnage de 1,4 m ! A priori, la comparaison directe des deux nuages était donc impensable. En pratique, les points photogrammétriques sont les mêmes d'une campagne à l'autre et il est donc possible de les apparier deux à deux dans ce cas précis (c'est ce que font les prestataires d'EDF). Ce n'est par contre pas le cas général et il a paru intéressant de pousser l'étude vers des solutions plus *génériques*. Il est donc rapidement apparu nécessaire pour cela de se ramener au schéma classique de comparaison "nuage/modèle". Il était nécessaire pour cela de modéliser le nuage de référence préalablement. Or, un maillage triangulaire - ou plus généralement polygonal - introduit beaucoup d'erreurs dans la comparaison. Étant donné le faible nombre de points, ce sont des modèles très anguleux qui représentent très mal le côté lisse et pseudo-cylindrique de la surface de révolution. C'est d'ailleurs un problème général des maillages polygonaux censés représenter des surfaces courbes avec peu de sommets (voir figure 5.4). La technique de comparaison directe des nuages par modélisation locale (section 2.3.2) n'est donc pas non plus adaptée.

L'idée du stage a donc été d'utiliser des maillages paramétriques comme une surface spline (Cf. section D.2.4) qui a la propriété de permettre un bien meilleur contrôle de la déformation de la surface et assure une certaine continuité aux points de raccord. Il existe

---

<sup>2</sup>M.V.A. : Mathématiques / Vision / Apprentissage

---

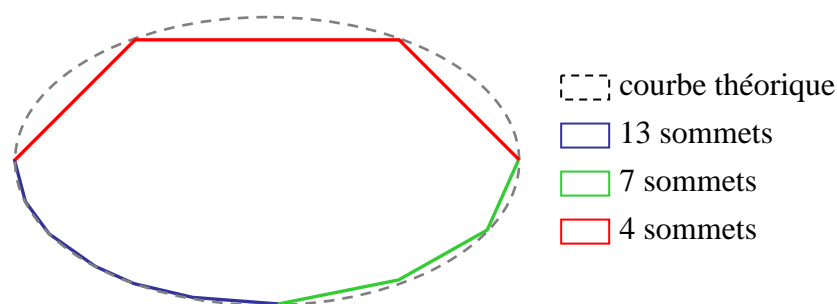


FIG. 5.4 – Approximation d'une courbe par un polygone.

de plus des algorithmes permettant de créer rapidement de tels modèles à partir d'un modèle polygonal ([Ueshiba et Roth 1999]). Cela revient en quelque sorte à lisser le maillage polygonal. De plus, une particularité des aéroréfrigérants est leur forme qui permet de les *déplier* selon leur axe de révolution vertical pour se ramener à une surface 2D. Bien que le but soit toujours d'évaluer des distances en 3D, cela permet de grandement simplifier certaines étapes du processus, en particulier la phase de calcul du modèle paramétrique : celui-ci devient très simple à calculer puisque on peut le déduire d'une simple triangulation de Delaunay 2D suivie de quelques opérations supplémentaires pour raccorder les bords et ainsi *fermer* le maillage rétro-projeté en 3D. Un autre avantage de la méthode de Ueshiba et al. est, en reprenant les termes de Salma Bougacha, que « *elle a l'avantage de construire des patches [de Bézier] de degré 3 car toutes les mailles sont triangulaires alors que les méthodes générales de construction de splines requièrent des patches de degré 4. Elle est pour cette raison très rapide [...]* ».

On peut alors calculer la distance de chaque point du nuage comparé au modèle paramétrique. On utilise une distance radiale et signée dans ce cas. Le nuage comparé est ensuite déplié à son tour en 2D pour pouvoir calculer des lignes de niveaux, appelées *iso-déformées* et qui sont la forme standard d'un rapport d'étude de déformation sur ce type d'ouvrage. Il est apparu que la méthode mise au point lors du stage donnait des résultats similaires à ceux obtenus par les prestataires d'EDF. Cela validait ainsi la méthode de calcul (voir figure 5.5).

### Comparaison à partir de données laser

On a utilisé les techniques développées dans le cadre de cette thèse sur deux relevés laser effectués en janvier et mai 2005 sur un aéroréfrigérant de la centrale nucléaire de Cruas (Drôme). Il n'a malheureusement pas été possible d'obtenir des données laser et photogrammétriques sur le même aéroréfrigérant et aux mêmes dates, ce qui aurait permis une comparaison directe des deux techniques.

Les deux nuages laser ont été acquis avec le même scanner laser (modèle : Riegl LMSZ-

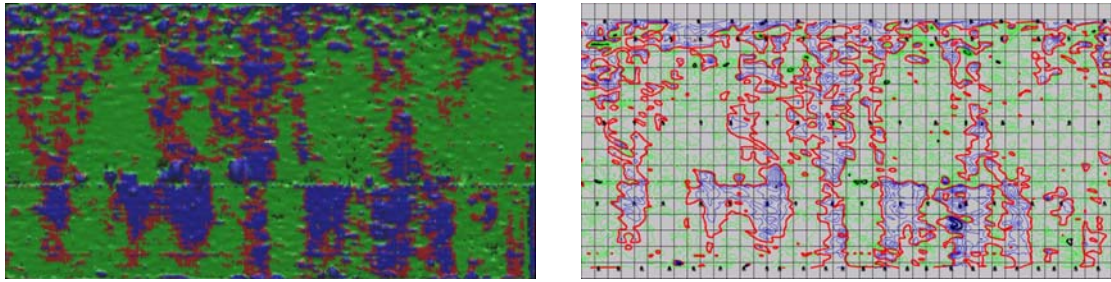


FIG. 5.5 – Résultat de la méthode par comparaison à une surface spline (gauche) et résultats fournis par les prestataires d'EDF (droite).

*La légende des couleurs est le rouge pour une déformation nulle, le noir pour les déformations de  $\pm 5$  cm, le vert pour déformations comprises entre 0 et -5 cm et le bleu pour des déformations comprises entre 0 et +5 cm.*

420i). Mais le scanner était dérégulé pendant la campagne de janvier, sans doute après un choc, et malgré une correction partielle des données suite au diagnostic du problème par le constructeur, le nuage de janvier reste fortement bruité. Les deux nuages sont chacun composés de sept points de vue répartis à peu près régulièrement sur tout le pourtour de l'ouvrage. Les points de vues, appelés aussi *stations*, sont positionnés précisément grâce à des points d'appui et sont communs aux deux séries d'acquisitions (voir figure 5.6). L'acquisition totale a été réalisée chaque fois en l'espace d'une seule journée. Les différents nuages correspondant à chaque station sont consolidés directement grâce à la connaissance précise de la position du scanner et des paramètres de prise de vue (aucun algorithme de consolidation - qui permet une optimisation globale ou locale de la concordance entre les différentes zones de recouvrement des nuages - n'est appliqué). Enfin, le nuage global est segmenté pour ne représenter que l'aéroréfrigérant, car le scanner acquiert en effet tous les bâtiments et autres objets situés aux alentours et qui se trouvent dans son champ de vision. Au final, chaque nuage consolidé et segmenté est composé approximativement de 13 millions de points.

L'utilisation d'un repère topographique précis et unique tout au long du processus permet d'avoir directement un repère commun pour comparer les deux nuages. En effet, toute autre technique, comme la consolidation automatique des points de vue, aurait décalé la position et l'orientation de l'axe de révolution de chaque représentation de l'aéroréfrigérant, rendant la comparaison imprécise et ajoutant ainsi une erreur totalement incontrôlable. En se basant uniquement sur les points d'appui fixes et proprement géoréférencés par des techniques topographiques classiques très précises, on peut admettre que les principales sources d'erreur de positionnement entre les deux nuages (qui se superposent aux différences) restent la distorsion globale et les imprécisions locales du scanner.

La méthode de comparaison directe de nuages de points présentée dans le chapitre 2 peut être appliquée à ces deux nuages. L'erreur quadratique moyenne théorique due à l'échantillonnage se situe en effet autour de 3,5 cm. Mais les déformations attendues se

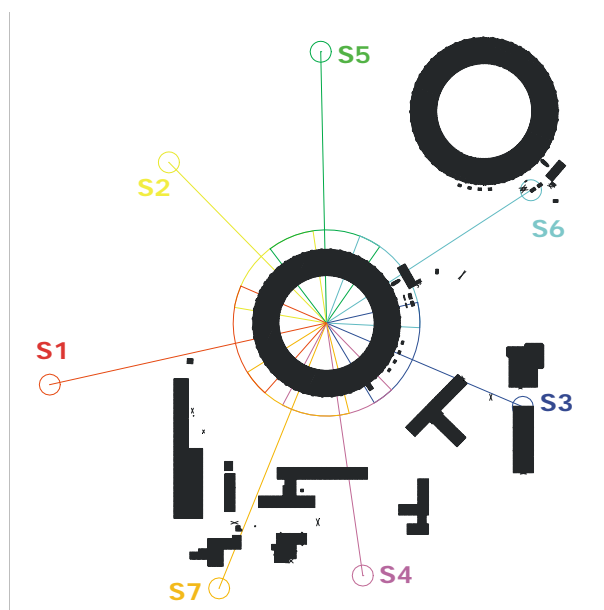


FIG. 5.6 – Position et couverture des 7 stations d'acquisition de la double campagne de relevés sur l'aéroréfrigérant de la centrale de Cruas.

situant entre 0 et 10 cm, il est préférable d'utiliser l'optimisation par modélisation locale (section 2.3.2) pour améliorer légèrement les résultats du calcul de distance. Enfin, les nuages étant tous les deux complets et proprement segmentés, il n'est pas nécessaire d'utiliser le filtrage par point de vue (section 2.3.3). On présente au tableau 5.1 et en figure 5.7 les résultats du calcul de distance entre les deux nuages, avec et sans modélisation locale. On peut aussi observer les histogrammes correspondants, figure 5.8, où l'on voit apparaître comme prévu un biais de 3 à 4 cm dû à l'échantillonnage lorsque celui-ci n'est pas corrigé par modélisation locale.

Niveau d'octree	8	9
sans modélisation locale	254,5 s.	77,27 s.
avec modélisation locale	650,9 s.	382,38 s.

TAB. 5.1 – Comparaison des temps (en secondes) pour le calcul de la distance au plus proche voisin entre les nuages de l'aéroréfrigérant de Cruas acquis en janvier et mai 2005, *sans* et *avec* modélisation locale.

L'analyse ne peut malheureusement pas être poussée plus loin puisqu'il n'existe pas encore de *vérité terrain* sur cet ouvrage qui nous permettrait de confronter ces résultats. On peut tout de même faire deux remarques :

- On voit apparaître sur les nuages coloriés en fonction de la distance des zones de contraste très nettes, avec des bords courbes et réguliers (au pied de l'ouvrage en particulier). Ces zones correspondent en fait à des erreurs de recalage entre les



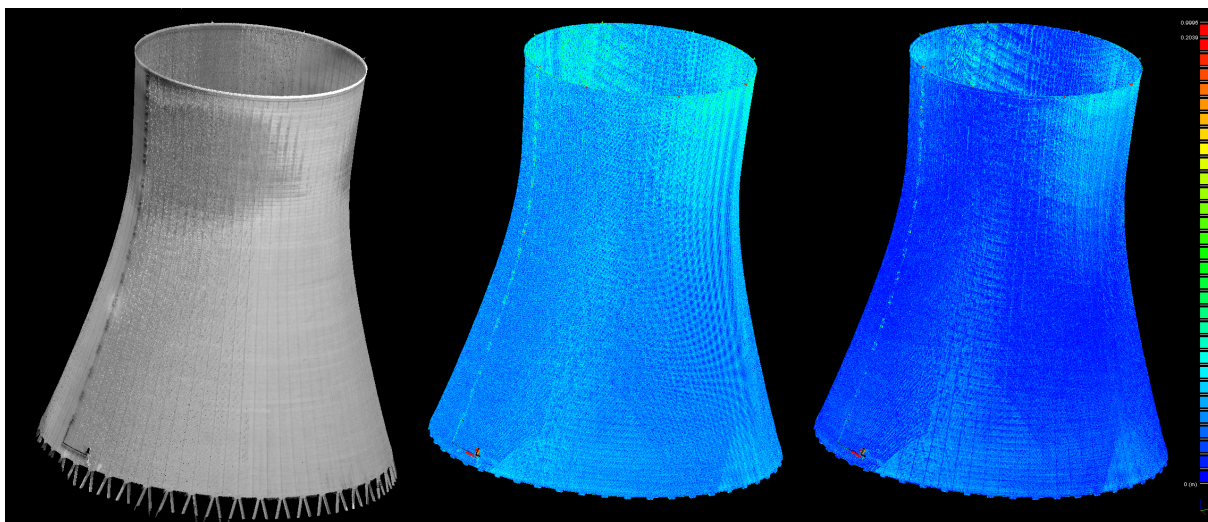


FIG. 5.7 – Comparaison des nuages de points laser sur un aéroréfrigérant.  
*A gauche : nuage coloré par couplage du scanner avec un appareil photographique. Au centre : distance au plus proche voisin simple. A droite : distance au plus proche voisin avec modélisation locale. Les franges que l'on voit apparaître sur le nuage du milieu sont dues à l'échantillonnage particulier des zones de recouvrement entre les sous-nuages correspondant aux différentes stations.*

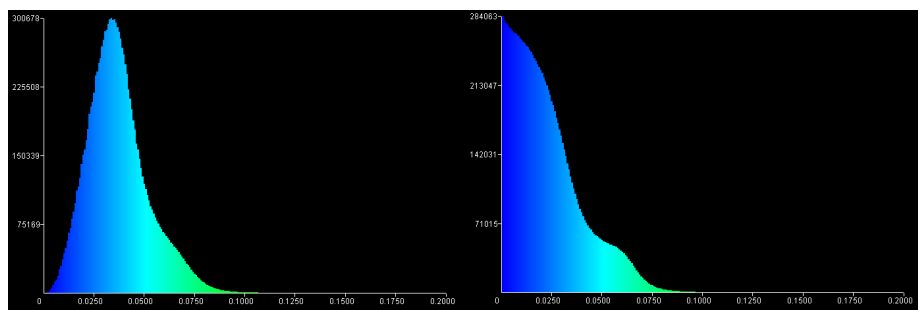


FIG. 5.8 – Comparaison des histogrammes des écarts entre 0 et 20 cm.  
*Sans (à gauche) et avec (à droite) modélisation locale. On peut voir clairement l'influence de l'échantillonnage des points laser sur l'histogramme de gauche (qui introduit un biais de 3 à 4 cm.). La modélisation locale supprime efficacement ce biais. Dans tous les cas, la très grande majorité des distances est inférieure à 8 cm.*

différents points de vue. Les distances dans ces zones sont globalement décalées par rapport à celles des zones avoisinantes. Bien que ce décalage soit faible, la taille importante de ces zones le rend particulièrement visible. Heureusement cette erreur reste faible dans l'absolu, et la qualité de la consolidation devrait de toute façon être grandement améliorée par l'utilisation d'un capteur correctement réglé.

- Une fois le problème d'échantillonnage résolu par l'utilisation de modèles locaux, l'histogramme à droite de la figure 5.8 nous montre que 99,9% des points se trouvent

à des distances inférieures à 10 cm, et près de 90% en dessous de 5 cm. Or, étant données les conditions d'acquisition ainsi que la distance du scanner à l'ouvrage (entre 100 et 200 m. en fonction de la hauteur des points), on obtient des distances qui sont de l'ordre de la portée du scanner. On pourrait donc fortement douter de la validité de ces données. Pourtant, celles-ci ne semblent pas totalement aberrantes, comme en témoignent ces zones de forte déformation relative qu'on peut observer en haut de l'ouvrage et qui semblent bien correspondre à un problème caractéristique des aéroréfrigérants (leur crête a fortement tendance à s'élargir). Bien sûr cela reste à vérifier de manière rigoureuse. Cela va en tout cas dans le sens de Gordon et al. ([Gordon et al. 2004]) qui ont mis en avant la capacité des scanners laser à remplacer la photogrammétrie dans les opérations de contrôle de déformation, car la très forte densité des données laser rattrape leur imprécision.

## 5.4 Suivi de glissement de terrain

Un glissement de terrain est défini comme le déplacement d'une masse de terrains meubles ou rocheux au long d'une surface de rupture par cisaillement, qui correspond souvent à une discontinuité préexistante. Le mouvement est engendré par l'action de la gravité, de forces extérieures (hydrauliques ou sismiques) ou d'une modification des conditions aux limites.

C'est un phénomène très courant, localisé principalement dans les zones montagneuses ou sur les rives des fleuves et les côtes maritimes (voir figure 5.9). L'extension d'un glissement de terrain est très variable mais peut atteindre plusieurs kilomètres dans les cas extrêmes. Le mouvement est plus ou moins rapide et son évolution n'est pas forcément continue. Les effets peuvent être dévastateurs, car les masses impliquées sont généralement énormes et emportent tout sur leur passage. On en répertorie de nombreux en France, la plupart dans des zones montagneuses (Alpes, Pyrénées, Massif Central, etc.). Ils sont donc suivis constamment par le Ministère de l'Écologie et du Développement Durable (MEDD) en partenariat avec l'Office National des Forêts (ONF), les autorités locales (DRIRE) et des centres de recherches (BRGM, LCPC). Une base de donnée recensant tous les glissements de terrain est consultable sur internet (<http://www.bdmvt.net>).

La classification des mouvements de terrain peut se faire en fonction de la vitesse avec laquelle ils se produisent. La première catégorie regroupe les mouvements lents et continus, tels que les *affaissements*, les *tassements* et les *glissements*. En s'accéléralant, ces derniers peuvent alors être rattachés, tout comme les *effondrements*, à la seconde catégorie : les mouvements rapides et brusques.

Certains glissements de terrain menacent directement des routes, des villages et aussi des ouvrages hydro-électriques gérés par EDF. Dans ce dernier cas, les effets d'un glissement de terrain peuvent s'avérer particulièrement dangereux, non pas à cause des vies

---

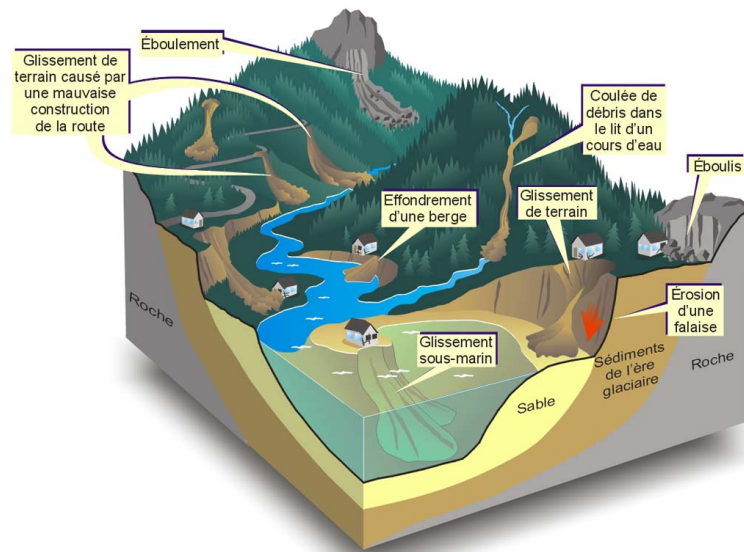


FIG. 5.9 – Illustration des différents types de glissements de terrain.  
 (source : site internet du Ministère des Ressources Naturelles du Canada)

humaines directement menacées (généralement aucune) mais parce que la masse de terrain qui tomberait dans la retenue d'eau en amont du barrage pourrait provoquer un déplacement d'eau soudain et important, qui lui-même pourrait passer par dessus la crête du barrage, voire même endommager cette crête et provoquer la rupture du barrage. Dans les deux cas, cela résulterait en une très forte montée des eaux en aval, avec probablement un front d'eau important emportant tout sur son passage ainsi qu'une inondation de la vallée. Les populations touchées seraient alors beaucoup plus nombreuses, et la catastrophe d'autant plus meurtrière et paralysante.

C'est pour cette raison qu'EDF suit de près les zones de glissements de terrains autour de certains de ses barrages. Aujourd'hui le suivi est généralement réalisé par des campagnes de mesure manuelles (théodolites, photogrammétrie terrestre, systèmes dGPS, etc.) espacées selon des intervalles de plusieurs mois ou plusieurs années en fonction de la vitesse du glissement. Ces campagnes sont lentes et présentent un danger potentiel pour les opérateurs qui font les mesures sur la zone de glissement même. EDF envisage donc la possibilité de recourir à de nouvelles technologies "sûres" et plus rapides pour suivre les glissements. Les principales méthodes envisagées et testées sont la photogrammétrie hélicoptérée, la mesure par radar interférométrique satellitaire (section 1.5.3) et enfin le relevé laser terrestre (section 1.5.1).

Dans le cas d'une application de suivi d'un glissement de terrain, il peut être intéressant d'utiliser des techniques de calcul d'écart comme celles proposées dans ce manuscrit. En effet, contrairement aux mesures manuelles ou photogrammétriques qui donnent des séries de points parfaitement localisés et qu'on peut associer deux à deux d'un relevé à l'autre

(permettant ainsi le calcul direct d'un champ de déplacements, avec un vecteur défini en chaque point), les capteurs lasers fournissent un très grand nombre de mesures mais dont la position est localement aléatoire, et qu'il n'est donc pas possible d'associer deux à deux. On est en pratique dans un cas tout à fait similaire au calcul d'écart entre deux nuages de points qui a été traité en section 2.3.1.

Le cas précis de l'utilisation de scanners laser terrestres pour suivre des glissements de terrain a déjà été envisagé et relaté dans des articles de recherche. On peut ainsi citer [Hsiao et al. 2004] et [Bitelli et al. 2004] qui comparent deux nuages de points laser acquis sur le site d'un glissement de terrain en modélisant chacun des nuages. Le but est d'obtenir un modèle numérique de surface (MNS - voir Annexe D) en profitant du fait que les zones de glissement ont en général une topographie " $2D\frac{1}{2}$ ", et qu'il est donc possible d'appliquer une triangulation simple et topologiquement correcte en projetant les points du nuage sur un plan. Ceci leur permet alors d'extraire des profils et des lignes de niveaux, de reprojeter les limites du glissement sur des photos ou des rendus de synthèse en 3D, et enfin et surtout d'extraire à partir du MNS, un modèle numérique de terrain (MNT) qui correspond au MNS débarrassé des éléments du sur-sol (arbres, blocs rocheux, etc.). La comparaison des MNT leur permet ainsi d'évaluer le mouvement du terrain uniquement (ils s'inspirent notamment de travaux antérieurs de filtrage du sur-sol à partir de données laser aériennes, qui sont elles aussi  $2D\frac{1}{2}$ ). On peut enfin citer des travaux de [Steinmann et Bonnard 2002] du laboratoire LMS de l'EPFL, ainsi que le système *SiteMonitor* développé par la société "3DLM", qui utilisent des scanners laser *à poste* pour observer un glissement de terrain (ils profitent alors du fait que le point de vue du scanner reste unique et qu'il est donc encore possible d'appliquer des traitements en  $2D\frac{1}{2}$ ). Globalement les intérêts des capteurs laser pour ce type d'opération sont leur vitesse et leur portée relativement grande qui permet un suivi continu ou en urgence de zones difficiles d'accès ou dangereuses.

Pour tester nos algorithmes sur de tels cas, le constructeur de scanners laser Mensi (filiale de Trimble) nous a gracieusement fourni, via son antenne japonaise, des nuages laser acquis sur une falaise au Japon, avant et après un éboulement. La figure 5.10 nous donne un aperçu de la carte de distance qui peut être calculée entre le nuage "après éboulement" et le nuage "avant éboulement" (on peut remarquer que le nuage "après éboulement" est beaucoup plus petit que l'autre, et qu'il se borne aux strictes limites de la zone d'éboulement - ce qui est assez étonnant, et potentiellement problématique).

La carte des écarts qui peut être calculée entre les deux nuages grâce à notre méthode est assez précise étant données la simplicité du problème et la bonne couverture du nuage "avant", qui permet une modélisation locale robuste lors de la comparaison. De plus, nous ne sommes pas limités par la topographie de la zone observée, qui pourrait être réellement 3D (avec des renforcements, des creux, ou autres concavités). Ensuite, la possibilité de jouer sur le seuillage des distances affichées permet dans un second temps la détermination simple et rapide de la frontière de la zone de glissement (on pourrait aussi utiliser les méthodes de segmentation automatiques présentées au chapitre 3 si la zone de glissement était plus complexe et la frontière moins évidente).

---

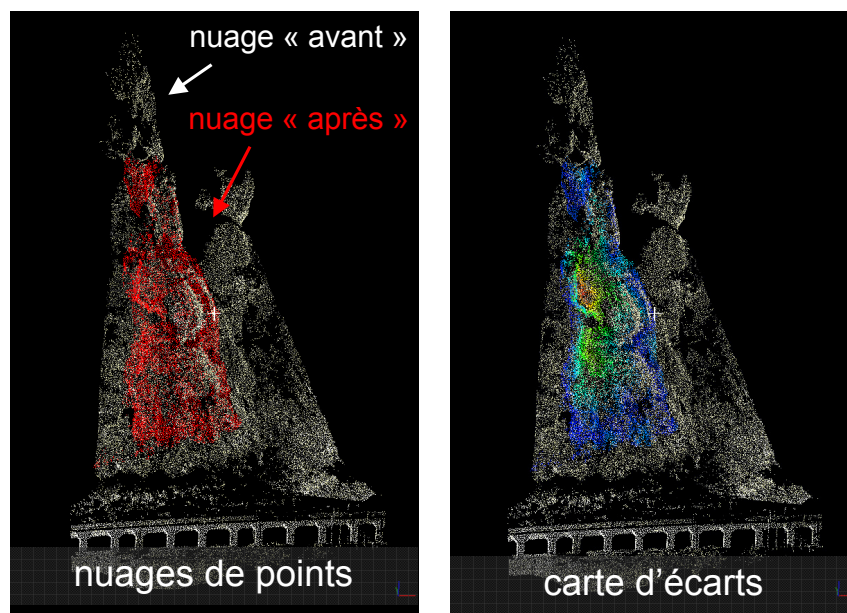


FIG. 5.10 – Calcul des écarts sur une zone de glissement de terrain.  
*A gauche : nuages de points fournis par Mensi Japan sur une zone d'effondrement d'une falaise. A droite : carte d'écart "après/avant" mise en situation.*

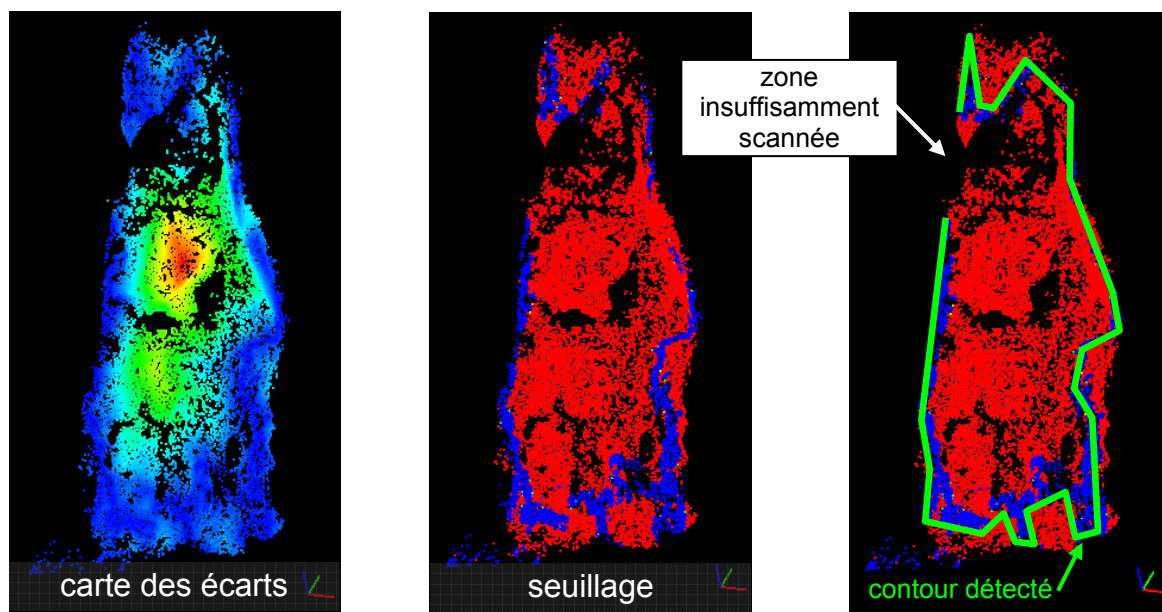


FIG. 5.11 – Mise en évidence des contours d'un glissement de terrain.  
*A gauche : carte des écarts ; au centre : seuillage faisant apparaître la frontière de la zone de glissement ; à droite : contour "théorique" faisant ressortir une zone du glissement non observée.*

On peut voir en figure 5.11, que dans l'exemple fourni par Mensi, cette frontière n'est d'ailleurs pas intégralement représentée par le nuage (en haut à gauche). Il s'avère que le nuage acquis après l'effondrement a une couverture insuffisante. Cela nous laisse penser que si les opérateurs qui ont réalisé ces mesures avaient disposé d'un outil permettant cette comparaison simple, très rapide et précise, ils auraient pu se rendre compte de leur erreur et scanner la zone manquante. Ceci peut sembler être un détail, mais il s'avère que, d'un point de vue plus général que la simple observation des glissement de terrains, la capacité de pouvoir appliquer des analyses et comparaisons rapides sur les nuages de points directement à la sortie du scanner pourrait améliorer nettement la qualité et l'efficacité des campagnes de relevé laser.

## 5.5 Cartographie d'urgence

Les événements tragiques qui se sont déroulés le 11 septembre 2001 à New-York, outre le fait qu'ils ont eu de nombreuses conséquences assez évidentes sur le plan social et politique aux États-Unis et plus généralement sur la géopolitique mondiale, ont aussi démontré et concrétisé le formidable potentiel qu'offrent les nouveaux systèmes de télémétrie laser et radar pour la gestion de crises. Alors que les catastrophes civiles et industrielles prenaient des proportions *inimaginables* et quelque peu inquiétantes<sup>3</sup>, les méthodes de gestion de crise et les outils de réponse d'urgence se montraient à la hauteur, jouant parfaitement leur rôle au moment crucial. Bien que modernes, les systèmes (de relevé tridimensionnels rapides en particulier) étaient déjà opérationnels et avaient fait leurs preuves dans de nombreuses applications, et même dans des situations de crise comme des inondations. Les militaires américains avaient déjà pressenti l'intérêt et le potentiel de telles technologies, et l'existence du projet R.T.V.<sup>4</sup> n'était pas une simple coïncidence. Le Canada était aussi pionnier dans ce domaine, inventant le terme de *cartographie d'urgence* pour les catastrophes naturelles qui sont particulièrement étendues là-bas (à l'image du pays et des conditions climatiques extrêmes qui peuvent y régner).

Dans le domaine civil, l'émergence dans les années 90 de nombreuses sociétés commerciales proposant des relevés topographiques par laser, ainsi qu'une abondante littérature scientifique sur le sujet, sont une preuve supplémentaire du grand intérêt que les gens portaient déjà à ces techniques. Mais le fait est qu'après le 11 Septembre, les instances dirigeantes et les professionnels du risque et de la gestion de crise se sont trouvés devant une situation inédite, aux proportions exceptionnelles. Il est évident que sans ces systèmes de cartographie d'urgence, les répercussions auraient été encore plus graves. Aujourd'hui ces techniques continuent de se développer et de s'améliorer. Leur utilisation pour la gestion de crise, en particulier à grande échelle et dans des conditions où les hommes auront du

---

<sup>3</sup>en particulier en France, avec l'explosion à Toulouse de l'usine A.Z.F., dix jours plus tard

<sup>4</sup>Le programme *Rapid Terrain Visualization*, du J.P.S.D. (*Joint Precision Strike Demonstration Project* de l'armée américaine). Cf. "21st Century Terrain - Entering the Urban World" par Jeffrey T. Turner et Christian P. Moscoso

---

mal à être physiquement présents sur place, est en plein essor. Une dynamique mondiale de recherche et de réflexion commence à apparaître. C'est dans ce contexte qu'EDF R&D, organe de recherche et de prospection d'EDF, s'intéresse à la cartographie d'urgence. Que ce soit pour des causes naturelles, des agressions extérieures, ou des accidents, il n'est pas inutile et même plutôt légitime pour une entreprise comme EDF de s'intéresser de près à ces techniques émergentes.

Dans ce cadre, et en admettant que l'on dispose de modèles 3D des zones à risque (usines SEVESO, centrales nucléaires, villes, etc.), considérons le scénario suivant : juste après, voire pendant, la catastrophe, un engin volant ou roulant, drone ou téléguidé, équipé d'un scanner laser et/ou d'un radar pourrait survoler ou parcourir la zone et acquérir très rapidement des données 3D denses et précises. On peut citer comme exemples de telles solutions le projet *Stanford Autonomous Helicopter Project* [Thrun et al. 2003] (qui aborde d'ailleurs depuis peu le problème de la détection de changement) et plus généralement la littérature dans le domaine du *SLAM*<sup>5</sup>, le programme ROBEA<sup>6</sup> en France, etc. Ces données pourraient alors être comparées et traitées pour générer des cartes précises et exploitables quasiment en temps réel. De plus, plusieurs applications de détection de changement géométrique pourraient aussi en tirer partie :

- La détection automatique ou semi-automatique des zones endommagées, et plus particulièrement des voies d'accès pour permettre une planification des secours rapide et efficace.
- Une mise à jour continue des cartes, par répétition de l'opération (l'obsolescence de l'information étant très rapide en cas de crise).
- La détection des zones dangereuses et des possibles effondrements de bâtiments. Cette opération peut aussi être effectuée de manière ciblée par une équipe au sol transportant un capteur laser (voir figure 5.12). La rapidité des algorithmes utilisés pour effectuer ces opérations, comme ceux développés dans le cadre de cette thèse, peut donc être cruciale.
- Enfin, la constitution de modèles 3D qui pourront permettre à plus long terme d'organiser les travaux de déblaiement et de mesurer leur avancement.

Tout ceci peut paraître très théorique et quelque peu visionnaire, mais ce n'est pourtant ni plus ni moins que ce qui a été effectué par la FEMA<sup>7</sup> associée à la mairie de New-York après les attentats du 11 septembre 2001 [Huyck et Adams 2002], [Kern 2001] et [FEMA 2002].

Bien que ces arguments et ce qui s'est passé ces dernières années dans le monde (attentats, tremblements de terre, raz de marée<sup>8</sup>, etc.) n'aient apparemment pas suffi à

<sup>5</sup>Simultaneous Localization And Mapping

<sup>6</sup>"ROBotique et Entités Artificielles" : programme interdisciplinaire CNRS, INRIA et l'ACI Cognitive du ministère de la Recherche.

<sup>7</sup>Federal Emergency Management Agency, organisme fédéral américain de gestion de crise.

<sup>8</sup>On peut par contre citer le service SERTIT (Service Régional de Traitement d'Image et de Télédétection), situé à Strasbourg, dont la vocation est "d'extraire et de mettre en forme de l'information à partir des données produites par les systèmes d'observation de la Terre". Ils produisent en pratique des cartes réalisées à partir d'images satellites très rapidement en cas de crise et se sont montrés particulièrement



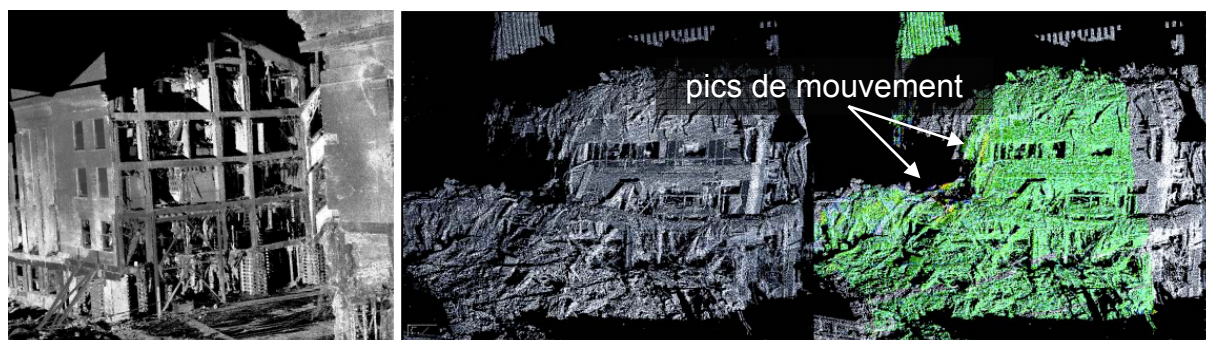


FIG. 5.12 – Nuages de points acquis par un capteur laser ILRIS-3D (Optech) représentant la partie éventrée du Pentagone (gauche) et l'hôtel *Marriot*, bâtiment no. 3 du World Trade Center (droite) après les attentats du 11 septembre 2001.

*L'image de l'hôtel met en évidence la subsidence entre deux acquisitions faites le 2 octobre à 18h42 puis 20h09 (les couleurs jaune/orange indiquent un mouvement de 76 cm à peu près).*

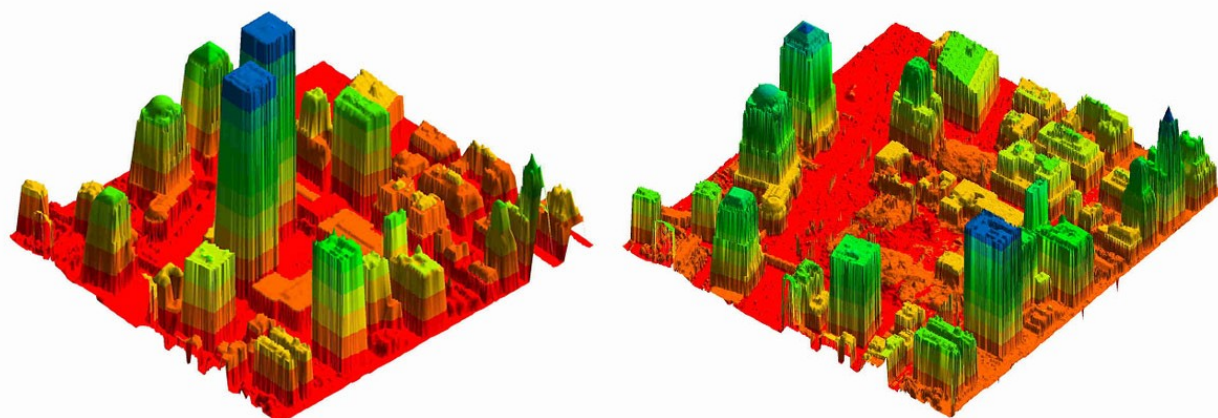


FIG. 5.13 – Nuages de points acquis par un capteur aéroporté au-dessus du site de *Ground-Zero* à New-York avant (gauche) et après (droite) les attentats du 11 septembre 2001.

déclencher un vaste programme d'acquisition de données 3D sur les sites sensibles et stratégiques du territoire français ou même simplement une systématisation du procédé lors de la construction de nouveaux sites de ce type, ils peuvent au moins être considérés comme des raisons supplémentaires d'utiliser des techniques d'acquisition 3D dans le cadre de l'exploitation normale de ces sites. En effet, outre l'intérêt aujourd'hui avéré de l'acquisition laser pour conduire des opérations de maintenance et garder à jour la documentation TQC du site, les données ainsi acquises gardent une valeur à plus long terme puisqu'elles pourront permettre une amélioration très forte de l'efficacité de potentielles opérations de gestion de crise à venir (sans nécessiter aucun traitement supplémentaire). L'optimisation

---

performants lors du Tsunami qui a touché l'Asie du sud-est en décembre 2004.

---



des opérations de sauvetage et surtout de déblaiement, la reprise plus rapide de l'activité, ainsi qu'un rattrapage partiel de la mauvaise presse causée par l'événement (grâce à la démonstration de sa maîtrise), devraient finir de convaincre les plus *sceptiques*.

Enfin, ce type d'application n'est pas incompatible avec les approches satellitaires de cartographie d'urgence (et de détection de changement en général) qui suscitent aussi un grand intérêt, en particulier pour des catastrophes de très grande envergure (comme le Tsunami du 26 décembre 2004 qui a frappé le sud-est de l'Asie ou les nombreux tremblements de terre, feux de forêts, etc.). Les échelles traitées ne sont pas les mêmes, ce qui rend ces techniques complémentaires : l'une permet d'avoir une vision d'ensemble et l'autre un diagnostic plus détaillé des zones critiques. Leurs sensibilités ne sont pas non plus les mêmes : la disponibilité des satellites n'est pas toujours évidente et leurs mesures dépendent des conditions atmosphériques et de la luminosité, alors que les capteurs aéroportés ou terrestres (ou leurs vecteurs) sont très sensibles aux perturbations locales (vent, fumée, poussière, décharges électromagnétiques, etc.).

---

# Conclusions et perspectives

## Conclusions

On a tenté de décrire, à travers les différents chapitres de ce manuscrit, un processus complet de détection de changement géométrique en 3D applicable à des données ponctuelles denses issues typiquement d'un scanner laser. Face à l'objectif de rapidité que certaines applications imposaient, nous avons tout d'abord étudié et testé une technique notoirement rapide de calcul de distance entre un nuage de points et un modèle 3D. Mais bien qu'effectivement très efficace, l'application de cette technique lorsque l'on ne dispose que de deux nuages de points nécessite la construction d'un modèle géométrique, opération longue et aussi hasardeuse lorsqu'elle est automatique. Nous avons donc proposé une méthode inédite permettant de comparer directement deux nuages de points. Ceci permet d'accélérer considérablement le processus au prix de légères imprécisions (dont il est possible d'évaluer l'influence sur le résultat final préalablement au calcul grâce à une formule très simple, ou alors qu'on peut réduire grâce à certains traitements supplémentaires). Mais le calcul de distance n'est que l'étape préliminaire du processus de détection de changement. Nous avons proposé dans le chapitre suivant plusieurs techniques de segmentation (manuelle ou automatique) qui permettent l'extraction des différentes zones de changements. Dans le chapitre suivant, nous avons présenté une structure d'accélération des calculs sur des données 3D (un octree) et une implémentation particulière de celle-ci qui présente divers avantages et qui profite à tous les algorithmes proposés jusque là. Enfin, nous avons montré qu'il est possible d'utiliser tout ou une partie de cette chaîne d'opérations pour d'autres applications que celles envisagées initialement.

Chaque étape n'est pas constituée d'une unique opération, mais plutôt de plusieurs briques qu'il est possible de choisir en fonction de contraintes à la fois au niveau de la durée admissible du traitement et de la qualité attendue du résultat produit. Le processus peut donc être adapté au cas par cas (voir figure 5.14), permettant ainsi aussi bien des opérations extrêmement rapides de comparaison (pour des applications de diagnostic sur le terrain typiquement) que des analyses plus coûteuses en temps mais plus fines (si l'on dispose de plus de temps mais aussi de plus de données).

Ce processus a été mis en place concrètement via le logiciel de démonstration des algorithmes de cette thèse : *CloudCompare* (Cf. section C et figure 5.15). Sans être parfait-

---

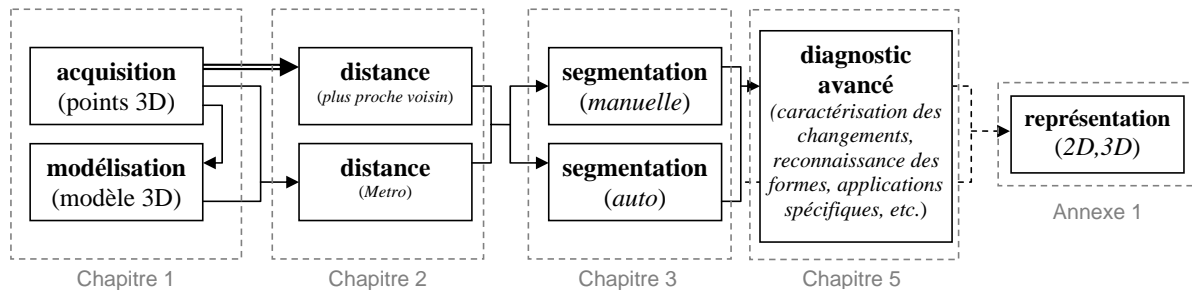


FIG. 5.14 – Processus complet de détection des changements en 3D et correspondance avec les différentes parties de ce manuscrit.

tement fonctionnel, car non pensé pour une mise en production, ce logiciel a néanmoins déjà permis à plusieurs personnes d’effectuer différentes tâches concrètes de comparaison géométrique de données 3D, accélérant ainsi des opérations jusque-là manuelles et fastidieuses.

On peut enfin évoquer un dernier argument en faveur de l’intervention d’un utilisateur tout au long du processus : lorsque des opérations telles que celles qui sont présentées dans ce manuscrit sont effectuées par un processus automatique plutôt que par un être humain (et en particulier sur des données très complexes), on note systématiquement un gain de confiance surestimé dans le résultat. Pourtant, aucun traitement informatisé de ce type n’est parfaitement infaillible. Il est donc plutôt sain que l’utilisateur participe activement au processus et remette en cause certains résultats. Le côté très professionnel que peut rapidement prendre une interface logicielle, ainsi que la présentation de résultats qui semblent tellement précis ne doit pas faire oublier que toute la logique du processus repose sur des heuristiques, seuils et autres constantes qui n’assurent qu’une performance globalement optimale, que l’on pourrait difficilement prétendre systématiquement optimale. Il n’en reste pas moins que la synergie entre calculs algorithmiques puissants et interactivité graphique permet seule d’atteindre des résultats satisfaisants sur des grands jeux de données et dans des temps records.

## Perspectives

Étant donné le nombre de composants imbriqués et interchangeable qui prennent part au processus global de détection de changement présenté dans ce manuscrit, les optimisations et améliorations possibles sont nombreuses. Voici celles qui nous semblent prioritaires :

- le calcul des distances entre deux nuages de points pourrait être amélioré en s’inspirant de travaux sur la génération de surface à partir de nuages. Le but n’est pas d’intégrer directement ces algorithmes au processus de calcul de distance (auquel cas on ne pourrait plus vraiment parler de comparaison *directe* de nuages de points)

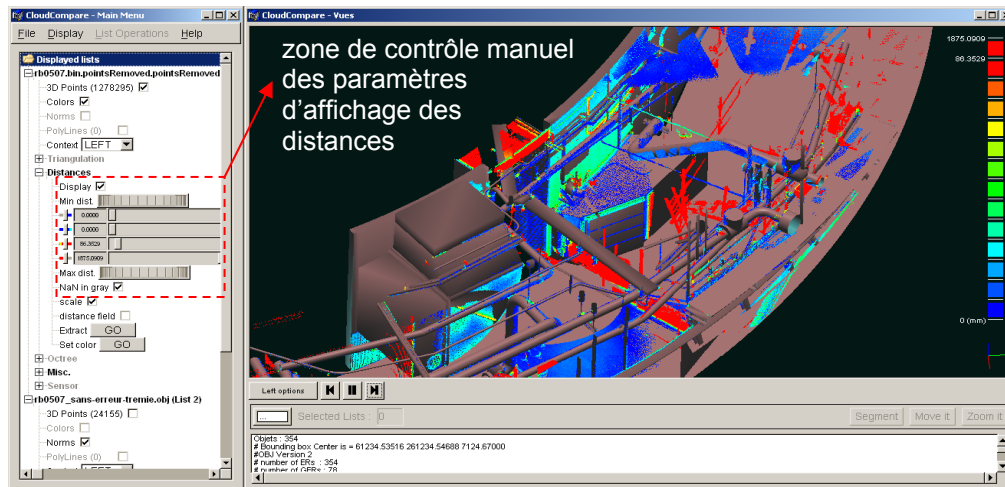


FIG. 5.15 – Interface de CloudCompare (ici, comparaison entre un nuage de points et un modèle 3D).

mais plutôt de se positionner à mi-chemin, avec des reconstructions locales robustes et rapides (à l'instar de la modélisation locale proposée en section 2.3.2 mais selon un principe plus évolué). On pourrait par exemple s'inspirer des travaux de [Morse 2001] ou [Sapiro et Memoli 2002] et [Memoli et Sapiro 2003] sur l'utilisation de surfaces implicites et de nuages 3D ainsi que d'approches statistiques.

- Les méthodes de segmentation présentées dans la section 3 peuvent toujours être améliorées (en particulier la méthode par propagation d'un front contrainte par la norme du gradient, qui malgré son fort potentiel est encore peu robuste). Néanmoins, les réels enjeux du domaine de la détection de changement reposent plutôt dans la capacité à analyser les changements en tant que tels, en les caractérisant de la manière la plus précise possible mais aussi en reconnaissant les objets touchés (une application d'"intelligence artificielle" effectuant la reconnaissance des formes à partir des zones de changement segmentées pourrait en particulier profiter des développements récents dans ce domaine). Le développement de méthodes de "diagnostic automatique" est la suite logique de nos travaux.
- Enfin, les applications potentielles proposées au Chapitre 5 sont toutes développables indépendamment :
  - L'aide à la réalisation de documentation T.Q.C. est sans doute l'application la mieux développée et la plus aboutie. Elle profiterait néanmoins directement de toute avancée dans le domaine du diagnostic automatique cité ci-dessus.
  - Il n'existe pas encore d'application globale de suivi de chantier satisfaisante, et la partie géométrique devrait être étudiée plus spécifiquement, en relation avec une véritable expression de besoins.
  - Le potentiel des capteurs laser pour les opérations de contrôle géométrique sur des ouvrages de génie civil ou le suivi de glissement de terrain nous semble évident, et la comparaison directe des nuages de points tout à fait adaptée. Des travaux spécifiques, en particulier au niveau du traitement de l'information de distance,

pourraient donc être réalisés dans ce domaine.

- La cartographie d’urgence est un sujet en plein essor, qui peut directement profiter des travaux présentés dans ce manuscrit, mais aussi des techniques d’acquisition de données 3D (ou autres, comme la température ou la radioactivité par exemple) en conditions extrêmes et sans intervention humaine, avec des drones ou des appareils téléguidés. Ceci nous renvoie aux travaux de SLAM<sup>9</sup> dans le domaine de la robotique (*Stanford Autonomous Helicopter Project*[Thrun et al. 2003], travaux du LAAS, etc.) mais aussi aux problèmes de fusion de données.

Étant donné que le processus repose encore sur une importante interactivité avec un opérateur humain, on peut rajouter à cette liste les travaux qui pourraient être menés sur l’ergonomie de l’interface homme-machine nécessaire à la mise en production d’un tel processus. De plus, et dans le même esprit que l’amélioration de la perception, des travaux pourraient être menés sur l’échange et le partage des résultats (travail collaboratif à distance, diffusion de données volumineuses sur un réseau, etc.) ainsi que sur les moyens de l’utiliser sur le terrain (diffusion sur des interfaces portables de type PDA<sup>10</sup>, etc.). L’utilisation de telles techniques pour améliorer la vitesse et la fiabilité du travail d’acquisition de données laser pourrait d’ailleurs être une application potentielle à part entière. Concrètement, des routines de détection de changement ou simplement de calcul de distance pourraient être embarquées sur les PC de pilotage des scanners laser.

A long terme, la détection de changements géométriques tridimensionnels, qui est au carrefour de nombreux domaines scientifiques, devrait se développer largement et permettre une systématisation des procédures de contrôle et de suivi géométrique, en particulier dans les domaines de l’industrie, du BTP mais aussi pour une gestion plus efficace des catastrophes de grande envergure.

---

<sup>9</sup>Localisation et Cartographie Simultanée, en français.

<sup>10</sup>Personal Digital Assistant.

---

## Annexe A

# Amélioration de la perception

Cette partie traite d'un problème annexe mais néanmoins essentiel : la représentation des résultats de détection de changement, et plus généralement des nuages de points 3D. Dans un contexte d'urgence ou lorsque les utilisateurs ne sont pas entraînés, que ce soit sur le terrain ou dans un bureau, une mauvaise représentation peut être un véritable frein, voire une source d'erreur, dans la compréhension des résultats. Le processus de détection de changement ne serait donc pas complet sans un moyen de représenter ses résultats de manière claire et si possible rapide. On présente ici en particulier un algorithme inédit de calcul de Portion de Ciel Visible (PCV) sur un gros nuage de points.

## A.1 Problèmes de perception

La perception des formes et des changements à partir de données tridimensionnelles en général, et de données ponctuelles en particulier, est un réel problème. Un des cas les moins évidents est justement celui de la visualisation des données qui se réduisent à des coordonnées de points. On s'intéresse ici en détail à ces problèmes et surtout à quelques moyens simples d'améliorer la perception des formes à partir d'un nuage de points 3D.

### A.1.1 Perception des changements

Pour ce qui est du problème spécifique de la perception de changements sur des modèles 3D, on peut se rapporter à des travaux de psychologie relatifs à la perception. D. J. Simons de l'université d'Harvard, s'est intéressé à travers de nombreux articles au problème de la détection des changements par l'être humain, et en particulier à partir de représentations 3D numériques (des modèles 3D assez simples mais texturés en l'occurrence). Avec P. Williams, ils ont conduit plusieurs expériences sur ce sujet précis (relatées

---

dans [Williams et Simons 2000]) dans le but d'établir un lien entre la taille et le nombre des changements et les capacités des sujets à les percevoir. Parmi les nombreuses observations faites lors de ces expériences, on peut en retenir plusieurs :

- premièrement, et de manière assez évidente, plus les changements sont grands et nombreux et prennent donc de l'importance par rapport à la taille du modèle 3D, plus la détection du changement est aisée. Le cas le plus difficile est donc un changement petit et isolé (au sein d'un modèle pourtant très simple par rapport aux modèles d'intérieur de centrale),
- deuxièmement, la détection des changements est plus aisée lorsque le sujet est familier avec les objets et leur environnement,
- enfin, un point très intéressant qui va dans une certaine mesure dans le sens de ce qu'on a pu observer au sujet de la segmentation manuelle (Cf. section 3.2.1) est que la détection est aussi plus facile lorsque le changement intervient à l'écran sur un objet en mouvement.

Néanmoins, même dans ce cas idéal et avec des objets 3D très simples, la détection des changements par l'être humain est loin d'être évidente.

En un sens, l'objectif de cette thèse étant la détection automatique (ou au moins semi-automatique) des changements, par le biais de capteurs spécifiques et de traitements informatiques mettant en évidence les zones touchées, elle tend donc à contourner le problème de la perception humaine. Il n'en reste pas moins que la barrière de la perception persiste même après la détection des changements, puisque les résultats doivent être synthétisés et représentés efficacement de façon à transmettre l'information à un utilisateur. Nous nous intéresserons donc dans la suite de ce chapitre à des techniques d'amélioration de la perception des formes sur des nuages de points.

### A.1.2 Perception des données 3D ponctuelles

Les données ponctuelles sont, dans notre cas, généralement dépourvues de couleurs ou de toute autre information analogue tant qu'un calcul d'attributs des points (comme la valeur des déplacements) n'a pas été effectué. Or, il est très difficile de percevoir la forme d'un objet représenté par affichage direct du nuage selon un point de vue unique, et sans coloration (figure A.1, à gauche), en particulier si ce point de vue est quelconque et n'est pas choisi dans ce but.

Il existe tout de même quelques moyens simples d'améliorer significativement la perception de sa forme globale : d'une part en jouant sur la coloration de points, et d'autre part en jouant sur le mouvement de l'objet et donc la parallaxe induite. Ce sont des techniques intuitives, car elles ne sont pas liées à la nature numérique ou ponctuelle des données considérées ici.

Un objet réel est toujours mieux perçu lorsque sa teinte ainsi que l'éclairage, et donc au

---

final ses couleurs, sont bien contrastées. De même, la vue de l'objet en mouvement permet une meilleure perception de sa forme globale ainsi que des détails dans une certaine mesure. Les facteurs de cette amélioration sont principalement : le jeu de superposition des formes locales, l'évolution du contour de l'objet et l'évolution des ombres si la source lumineuse est fixe. Numériquement, il en va de même pour un nuage de points. Un mouvement même léger de translation (en vision projective) ou de rotation donne une vision plus nette de la troisième dimension grâce à la parallaxe qui fait qu'un point plus éloigné bougera moins qu'un point proche, et ce proportionnellement à sa distance. C'est une notion à laquelle l'être humain est habitué (et même conditionné) et qui participe activement à la perception de la profondeur.

La coloration aide aussi beaucoup à percevoir la forme globale et même locale des objets. On peut très rapidement colorer artificiellement le nuage, en assignant une couleur à chaque point en fonction de son éloignement par rapport à un centre donné ou en fonction de la valeur d'une de ses coordonnées par exemple (figure A.1, au milieu). Cela apporte quasi instantanément une information supplémentaire sur la distance relative des points les uns par rapport aux autres et fait ainsi mieux ressortir la forme globale ainsi que certaines formes locales très saillantes. Mais le contraste reste en général faible et dépend beaucoup de la taille de l'objet. Enfin, le résultat est totalement non photo-réaliste.

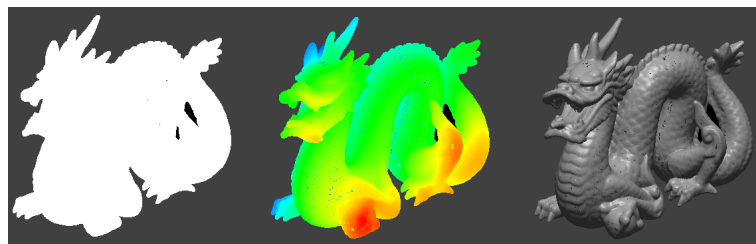


FIG. A.1 – Techniques basiques d'amélioration de la perception d'un nuage de points.  
*A gauche : nuage de point brut ; au centre : par coloration ;  
à droite : par affichage des normales.*

*Modèle "Dragon" (Stanford University Computer Graphics Laboratory, 437645 points).*

On améliore beaucoup plus la perception en modifiant la teinte des points en fonction d'une source d'éclairage directionnelle. Cela donne une information riche sur la direction locale et donc la forme de la surface. Pour cela, il faut par contre disposer d'une information de normale en chaque point (qui est la normale à la surface de l'objet évaluée au niveau du point). Celle-ci permet de faire varier la teinte du point en fonction de la direction d'éclairage par un simple produit scalaire entre cette normale et le vecteur d'incidence de la lumière (figure A.1, à droite). L'affichage est donc très rapide.

Cette information peut être obtenue lors de l'acquisition, ce qui est rapide et robuste et donne des résultats généralement très satisfaisants. Si ce n'est pas le cas, ou si cette donnée a été perdue, il faut donc la retrouver par calcul, ce qui est moins rapide et beaucoup plus problématique. Deux techniques de calcul des normales sur un nuage de points sont



étudiées dans la section suivante. On propose une amélioration notable des performances de l'une d'entre elles (la meilleure d'un point de vue qualitatif), au prix d'une très légère approximation qui n'a visiblement que très peu d'influence sur le résultat final.

Mais bien que rapide et détaillé, ce type de représentation n'est toujours pas photo-réaliste et peut donc perturber légèrement la perception, en particulier pour des personnes non habituées. Le problème se situe à très petite échelle, au niveau de la micro-géométrie. En effet, lors d'un affichage en fonction des normales, la teinte d'un point ne va dépendre que de la direction locale de la surface. Un petit morceau de surface situé au fond d'un creux pourra donc apparaître beaucoup plus lumineux qu'une partie plus vaste située dans une zone largement ouverte mais dont l'orientation serait plus *rasante*. Ce défaut peut-être corrigé par des calculs de synthèse d'éclairage dits de *Portion de Ciel Visible*, beaucoup plus réalistes et satisfaisants d'un point de vue perceptif, mais aussi beaucoup plus lourds. Cette technique de synthèse d'éclairage est étudiée en détail dans la dernière section de ce chapitre. On présente là encore un algorithme inédit développé dans le cadre de cette thèse, qui permet la synthèse d'éclairage de type *Portion de Ciel Visible* directement sur un nuage de points.

## A.2 Calcul rapide de normales

### A.2.1 Méthode 1 : ajustement de plans par la méthode des moindres carrés

Une méthode standard de calcul des normales sur un nuage de points repose sur le principe d'ajustement local des points par un plan calculé par la méthode des moindres carrés. Elle a été largement étudiée (Cf. [Mitra et Nguyen 2003] et en particulier la bibliographie de cet article) et est couramment utilisée. Elle a l'avantage d'être simple et robuste au bruit.

Voici son principe général :

- soit un point quelconque du nuage :  $P_i$ ,
- on extrait un voisinage  $N_k$  de  $k$  points autour de  $P_i$ ,
- on calcule le plan interpolant  $N_k$  par la méthode des moindres carrés,
- la normale au plan donne une approximation de la normale (non orientée) en  $P_i$
- le processus est répété pour chaque point du nuage.

Dans *CloudCompare*, logiciel de démonstration des algorithmes de cette thèse, cette méthode est simplifiée pour gagner du temps. En effet, l'extraction d'un voisinage coûtant assez cher en temps malgré l'octree, on applique ce principe en limitant la recherche des  $k$  plus proches voisins à l'intérieur de la cellule de l'octree à laquelle le point appartient. Si la cellule est suffisamment petite, cela améliore considérablement la rapidité des calculs.

---

En théorie il existe un effet de bord important dû au fait que les points des bords de la cellule auront des voisinages fortement déséquilibrés (tous les points du voisinage sont du même côté et n'encerclent donc pas le point traité). Il semble pourtant que l'effet en soit tout à fait minime puisque l'évaluation visuelle ne permet pas de le détecter (voir figure A.2).

Si nécessaire, on pourrait améliorer le résultat en faisant dépendre le nombre de points du voisinage à extraire de la courbure locale et de la densité du nuage. Mais dans ce cas, soit les calculs seraient alourdis, soit l'on aurait besoin de connaissances à priori comme la densité des points et le bruit du capteur (et il faudrait donc augmenter le nombre de paramètres de l'algorithme).

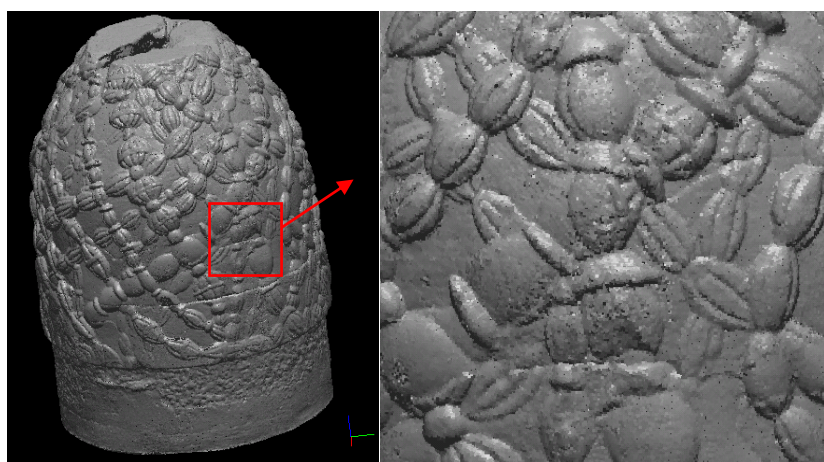


FIG. A.2 – Calcul des normales par ajustement d'un plan par la méthode des moindres carrés (limitée aux cellules d'octree).

*A gauche : illustration du calcul ; à droite : zoom sur une partie bruitée.*

*Modèle : Omphalos de Delphes (École Française d'Athènes / Fondation EDF, 832700 points).*

*Temps de calcul indicatif : 30.3 secondes (niveau 6 de l'octree).*

Bien que cette méthode soit tout à fait satisfaisante du point de vue du résultat, le temps de calcul reste relativement important. L'extraction des voisinages, même limitée à l'intérieur des cellules d'octree, ainsi que la détermination des plans aux moindres carrés, ont en effet un coût non négligeable lorsqu'ils sont appliqués à chaque point.

## A.2.2 Méthode 2 : triangulation locale en 2D

On présente ici une méthode plus approximative et moins robuste au bruit mais qui a comme principal avantage d'être extrêmement rapide parce qu'on n'ajuste plus un plan pour chaque point du nuage, mais seulement par groupe de points. Il faut l'envisager

comme une aide à la visualisation, le calcul étant appliqué temporairement pour une *découverte* du nuage (par exemple pendant que l'algorithme "propre" présenté précédemment serait appliqué en arrière-plan).

Voici son principe :

- via l'octree, on extrait les points cellule par cellule, ce qui est très rapide et correspond à des groupes de points assez importants si le niveau de subdivision de l'octree n'est pas trop grand.
- s'il n'y a pas assez de points dans la cellule (typiquement moins de six), on extrait alors un voisinage de points plus important autour du centre de cette cellule.
- pour chaque groupe de points, on calcule le plan aux moindres carrés,
- on projette les points sur ce plan,
- on triangule les points projetés (en 2D, donc, typiquement par une triangulation de Delaunay),
- on calcule les normales aux sommets des triangles (simple produit vectoriel de deux arêtes),
- on déduit alors une approximation de la normale des points en accumulant la normale des différents triangles en chacun des sommets. Il existe plusieurs méthodes d'accumulation des normales, mais on retiendra en particulier la moyenne équitable (chaque triangle a le même poids) ou la moyenne pondérée par la surface des triangles.

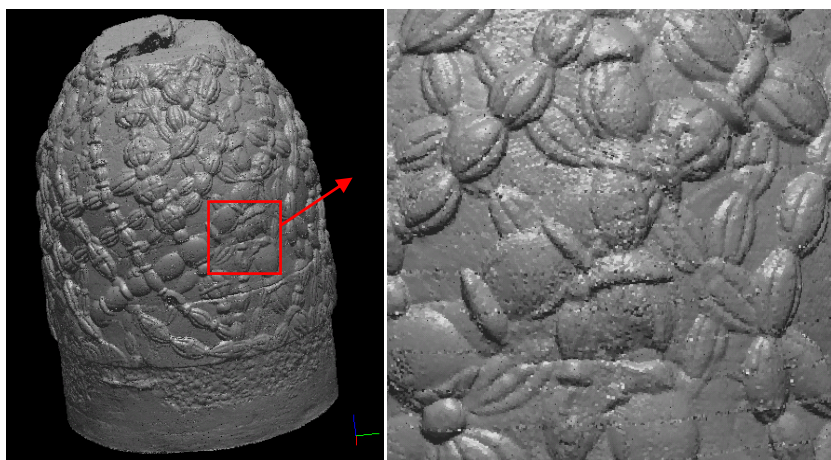


FIG. A.3 – Calcul des normales par triangulation locale (limitée aux cellules d'octree).

*A gauche : illustration du calcul ; à droite : zoom sur une partie bruitée.*

*Modèle : Omphalos de Delphes (École Française d'Athènes / Fondation EDF).*

*Temps de calcul indicatif : 4.04 secondes (niveau 5 de l'octree).*

L'aspect global est satisfaisant (voir figure A.3) et l'algorithme est effectivement très rapide (jusqu'à 10 fois plus rapide que l'algorithme par ajustement d'un plan aux moindres carrés). On peut par contre remarquer un bruit important à l'interface des cellules (i.e. sur la bordure extérieure de la triangulation 2D des points d'une cellule) qui fait apparaître des lignes régulières de teinte légèrement différente. Ceci est dû au fait que la triangulation

repose sur l'enveloppe convexe des points et crée ainsi sur les bords certains triangles qui ne correspondent pas à la surface de l'objet et dont les inclinaisons sont beaucoup plus fortes que la surface réelle (en particulier si celle-ci est courbée). Ces triangles viennent perturber le calcul des normales par accumulation. Le résultat peut être ainsi assez dégradé, en particulier si le niveau de subdivision de l'octree est grand. Cet effet est d'autant plus visible qu'il est régulier et systématique.

Pour corriger cet effet, nous avons eu l'idée de retirer de la triangulation 2D les triangles qui sont sur l'enveloppe convexe (en pratique, tous ceux qui ont au moins un point de l'enveloppe convexe parmi leurs 3 sommets). Les points qui se retrouvent alors sans estimation de normale ou avec une estimation trop incertaine (moins de trois contributions typiquement) seraient traités spécifiquement par la méthode de calcul des normales par recalage d'un plan. Les développements sont passablement plus lourds. Après implémentation et tests, les résultats ne se sont révélés ni meilleurs ni plus rapides, dépassant parfois même en temps de calcul la méthode globale aux moindres carrés. Ceci est dû principalement au temps supplémentaire nécessaire pour filtrer les triangles présents sur l'enveloppe convexe. Cette méthode n'a donc pas permis d'obtenir un résultat intermédiaire, en terme de qualité d'estimation des normales et de temps de calcul, entre les deux méthodes présentées dans cette section et a été abandonnée. Le choix pour l'utilisateur dans *CloudCompare* se fait donc entre un algorithme précis et lent (Méthode 1) ou un algorithme très rapide mais régulièrement bruité (Méthode 2).

Enfin, nous n'avons pas abordé le problème de la détermination du sens des normales. Celle-ci est très simpliste dans *CloudCompare* : elle se fait simplement par rapport au centre de gravité du nuage. Celui-ci peut être à l'intérieur ou à l'extérieur (de la matière). L'utilisateur le précise avant de lancer le calcul. Toutes les normales sont alors orientées de manière à pointer vers l'extérieur. Cela pose bien sûr des problèmes importants pour des objets présentant des concavités. Il existe des méthodes plus robustes par parcours d'arbre (type *breadth first search*, [Hoppe et al. 1992]) mais elles semblent assez gourmandes en temps de calcul et nécessitent des développements conséquents. On pourrait aussi utiliser l'algorithme de *Fast Marching* présenté en section 3.3.1. Cette problématique étant déjà en marge du sujet de cette thèse, il n'a pas paru nécessaire de l'approfondir.

## A.3 Calcul de la portion de ciel visible

Toujours dans le but d'améliorer la représentation d'un nuage de points, et en prenant en compte les observations faites dans la section A.1.2, nous avons mis au point une méthode d'illumination globale directe d'un nuage de points 3D. Elle repose sur un principe qu'on a déjà rencontré plusieurs fois dans ce manuscrit, à savoir que la surface de l'objet peut être approximée par l'octree (considéré à un niveau de subdivision donné). On applique alors un éclairage de type *Portion de Ciel Visible*.

---

### A.3.1 Principe

Il consiste, comme son nom l'indique, à colorier chaque point en fonction de la portion de ciel libre vue par ce point. On prend ainsi en compte la géométrie locale de l'objet et on joue sur les ombres portées pour améliorer la perception des formes. Cela correspond dans la réalité à la partie directe de l'éclairage ambiant qui participe pour une grande part au *réalisme* de la scène. On est par contre encore loin du photo-réalisme dans l'absolu, car il manque la partie secondaire due aux multiples réflexions de la lumière sur les objets environnants, ainsi que la prise en compte des propriétés physiques de la surface de l'objet, sa texture, etc. Ce type d'éclairage est proche de deux techniques de synthèse d'image : *Global Illumination* et *Sky Dome Rendering*.

Plus formellement, l'éclairage en *Portion de Ciel Visible* consiste à évaluer en chaque point la quantité d'énergie lumineuse reçue (voir figure A.4). Cette énergie est émise de manière isotrope, parallèle et infinie par l'hémisphère supérieur (défini par un axe des  $Z$  positif, qui peut être choisi arbitrairement). On assigne à chaque point un niveau de gris, proportionnellement à l'angle solide  $\phi$  de la portion de ciel vue depuis ce point rapporté à l'angle solide de l'hémisphère ( $2\pi$ ).

Les calculs peuvent être normalisés à ce que le pixel le plus lumineux possible ait directement une valeur qui puisse être codée sur 8 bits.

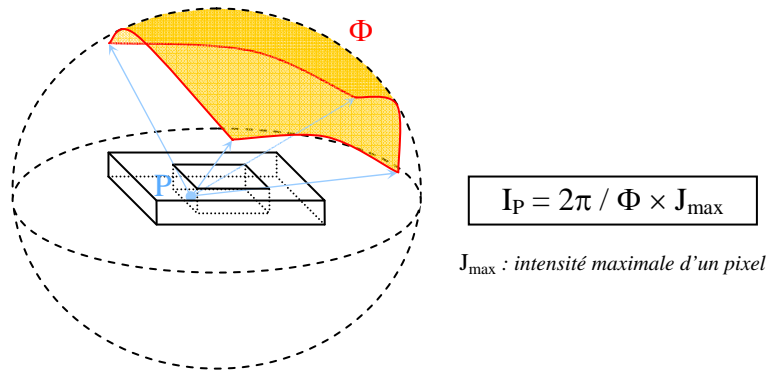


FIG. A.4 – Principe de l'illumination en *Portion de Ciel Visible*.

### A.3.2 Algorithme en 3D

L'algorithme est détaillé dans l'article [Duguet et Girardeau-Montaut 2004], aux côtés d'un algorithme équivalent développé spécifiquement pour des données ponctuelles  $2D^{1/2}$  (et mis au point par Florent Duguet, alors doctorant dans l'équipe *REVES* de l'INRIA en partenariat avec l'équipe *TSI* de Télécom Paris).

Le principe général de l'algorithme en 3D repose sur le fait que l'énergie reçue par un point est proportionnelle au nombre de rayons lumineux qui touchent ce point. Ainsi, plutôt que d'analyser la forme du voisinage du point et d'étendre cette analyse progressivement à tout le nuage pour déterminer les occlusions de la ligne d'horizon du point (ce qui est très complexe en 3D), on peut lancer un grand nombre de rayons sur le nuage, accumuler leur énergie pour chaque point qu'ils touchent et obtenir ainsi au final l'énergie totale reçue par chaque point. Comme on l'a dit précédemment, on utilise l'octree comme approximation de la surface du nuage. Celui-ci va permettre en particulier de déterminer la propagation et les occlusions partielles ou complètes de chaque rayon en fonction de la configuration des points dans les cellules qu'il traverse.

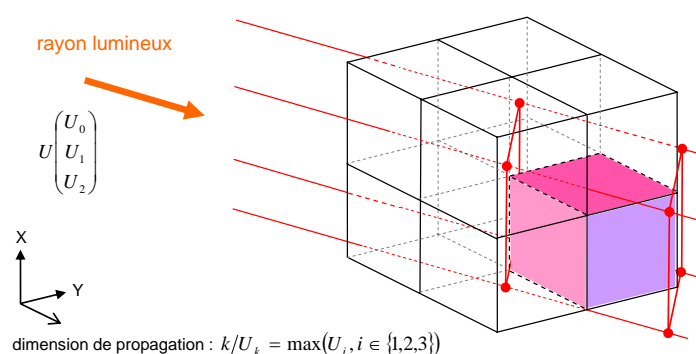


FIG. A.5 – Illustration d'un rayon lumineux tel que considéré dans l'algorithme de calcul de *Portion de Ciel Visible* 3D.

Voici rapidement le principe de l'algorithme :

- On considère l'octree à un niveau de subdivision donné.
- On échantillonne de manière isotrope des directions sur l'hémisphère supérieur [Saff et Kuijlaars 1997]. Soit  $N$  le nombre de directions.
- Pour chacune de ces directions, on propage un front de rayons lumineux parallèles sur tout l'octree. Pour cela l'octree est considéré comme une série de tranches 2D empilées selon la dimension principale de marche de la lumière (c.à.d. celle pour laquelle la coordonnée du vecteur directeur est la plus grande en valeur absolue). Les rayons sont considérés comme des parallélépipèdes, de dimension infinie selon cette direction principale, et dont la section englobe totalement une cellule (voir figure A.5).
- On initialise un masque de la taille d'une tranche avec la valeur de l'énergie nominale des rayons lumineux pour toutes les cellules.
- On confronte ce masque à la première tranche :
  - pour les cellules où il y a des points, on accumule la valeur d'énergie du masque à tous les points puis on met à jour cette valeur en fonction du nombre de points. Ces deux opérations peuvent être faites de manière assez précise en prenant en compte l'occultation des cellules voisines et surtout la disposition des points dans la cellule courante. Dans l'article, on utilise en effet un schéma particulier de subdivision du

rayon lumineux en 9 parties flottantes pour une gestion plus précise des transferts d'énergie lumineuse (voir l'article pour plus de précisions).

- Une fois tout le masque mis à jour et la tranche d'octree traitée, on passe à la tranche suivante en recommençant le même processus (c'est la phase de propagation).
- Enfin, on colorie les points en fonction de l'énergie accumulée, sachant que l'énergie maximale que peut recevoir un point est égale à  $N$  fois l'énergie nominale d'un rayon (fixée à  $\frac{255}{N}$  suite à la remarque faite précédemment ou à l'unité pour plus de simplicité).

On peut voir en figure A.4 un exemple de résultat sur le *Dragon de Stanford* (qui a déjà servi d'exemple en figure A.1). Ce résultat est très propre car le nombre de points est suffisamment grand et le niveau de subdivision de l'octree profond (relativement à la taille de l'objet).

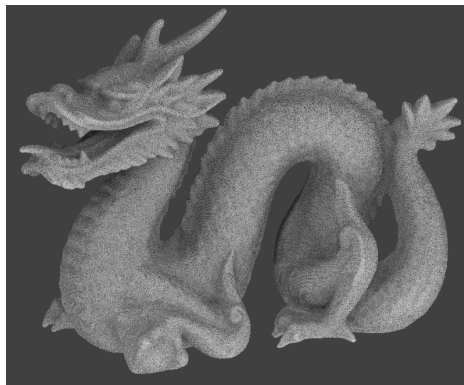


FIG. A.6 – Résultat de l'éclairage en *Portion de Ciel Visible* sur le Dragon de Stanford. Calcul au niveau 10 de l'octree, avec 32 rayons et 4 millions de points, échantillonnés au hasard sur la surface du modèle.

Le temps de calcul dépend principalement du niveau d'octree auquel est effectué la propagation des rayons. En effet, l'algorithme parcourt pour chaque direction d'éclairage toutes les cellules de la grille 3D (qu'on peut tout de même limiter aux extensions exactes du nuage selon les trois dimensions, et non au cube englobant théorique). Si le niveau d'octree est  $L$  et le nombre de rayons  $N$ , il parcourt donc  $2^{3L}N$  cellules, ce qui représente déjà  $134217728 * N$  cellules au niveau  $L = 9$ , qui est pourtant le niveau empirique minimal à partir duquel on commence à obtenir un résultat qu'on peut qualifier de visuellement agréable. Heureusement, une implémentation optimisée et une machine disposant de suffisamment de mémoire permettent de faire un tel calcul en quelques minutes. Mais il est vrai que pour obtenir des résultats très fins, il faut descendre vers des niveaux encore plus profonds (typiquement autour de 11 ou 12). Or, puisque le temps de calcul évolue en  $2^3L$ , chaque niveau supplémentaire multiplie le temps de calcul par 8 (un peu moins en réalité, car le nombre de points joue aussi, et un certain nombre d'opérations sont indépendantes du niveau d'octree). Il faut donc quelques heures de calcul pour obtenir un résultat très précis sur un nuage conséquent (entre 4 et 5 heures pour le nuage *des danseuses* - composé

de 27 millions de points - qui est présenté en fin de l'article). Mais il faut rappeler ici qu'il n'y a pas besoin de calculer de surface, seule la donnée des points 3D suffit à faire ce calcul (ainsi qu'à le représenter). Ainsi, on peut appliquer cette technique à de très gros nuages de points, avec pour seule limite théorique la mémoire disponible. Il existe des approches qui se basent sur des maillages triangulaires (Shadevis de Borgo et. al [Borgo et al. 2001] en particulier) et qui sont beaucoup plus rapides. L'utilisation d'un maillage permet aussi d'éviter les problèmes de "trous" que l'on rencontre avec l'octree. Mais il faut prendre en compte le temps et la difficulté de création du maillage (sans trous) qui exploiterait la totalité de l'information contenue dans les dizaines de millions de points du nuage.

Mais outre le temps de calcul, deux autres problèmes subsistent. Premièrement, l'octree ayant un place prépondérante dans l'algorithme, un aspect crénelé se retrouve dans les résultats, en particulier pour des niveaux de subdivision assez faible. Malgré un effort de lissage des résultats lors de la propagation, la structure cubique des cellules de l'octree se fait fortement ressentir. Puisque la surface du nuage est approximée par l'octree, il est assez logique que la surface qui *apparaît* dans le résultat présente des sortes de créneaux cubiques. Ceci explique la remarque faite dans le paragraphe précédent sur la nécessité de faire des calculs à des niveaux très profonds de l'octree pour obtenir des résultats fins (voir figure A.7). Pour éliminer à peu près totalement le crénelage, il faudrait en théorie, et au minimum, que la taille d'une cellule d'octree soit plus petite que la taille d'un pixel lors de l'affichage. Ceci peut d'ailleurs permettre de déterminer directement le niveau d'octree minimum nécessaire pour obtenir un résultat optimal pour une résolution donnée.

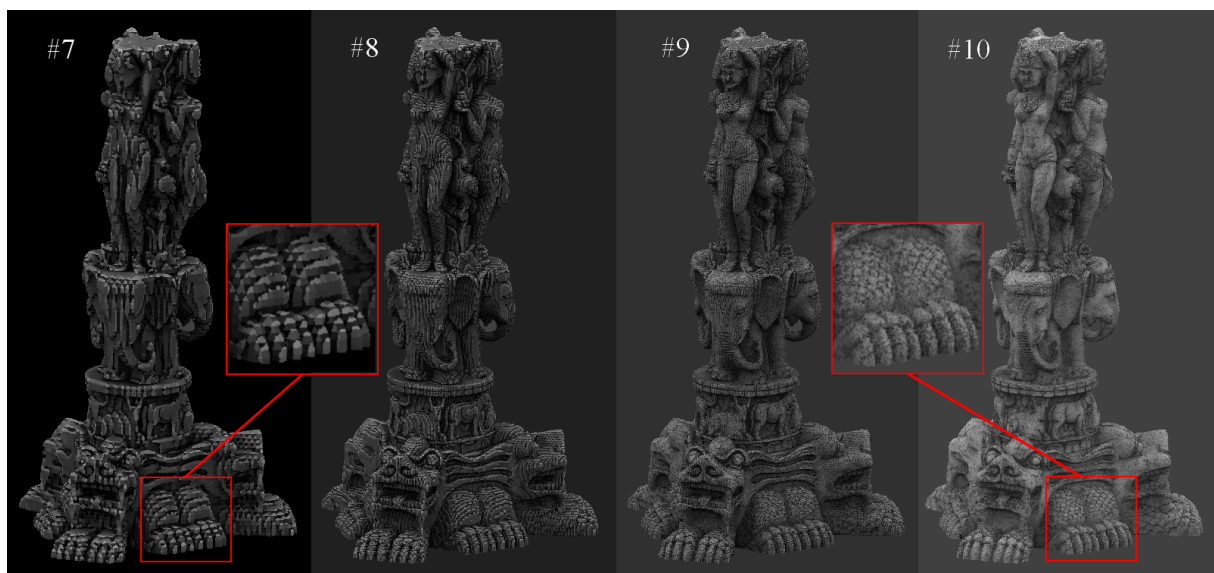


FIG. A.7 – Illustration du problème de crénelage en fonction du niveau d'octree.

(Le niveau d'octree est ici compris entre 7 et 10).

Agrandissement d'un détail de la statue pour les niveaux 7 et 10.

Modèle : "Thai Statuette" (XYZ RGB / Stanford University Computer Graphics Laboratory, 5 millions de points).



Le deuxième problème, qui est fortement lié au premier, vient de la densité du nuage. En effet si celui-ci n'est pas assez dense ou est incomplet, des trous apparaissent dans la surface induite par l'octree lorsque l'on se place à des niveaux trop profonds (la taille d'une cellule d'octree pouvant rapidement devenir plus petite que l'espace moyen entre deux points). Ces trous, surtout lorsqu'ils sont grands, ont une influence non négligeable sur le résultat final puisqu'ils laissent passer une certaine quantité d'énergie qui peut alors s'accumuler sur des points situés derrière et qui sont en théorie cachés (selon la direction d'éclairage courante). On peut donc voir apparaître des taches plus ou moins grosses en fonction de la taille des trous. Et si le nuage est très fortement troué, alors bien sûr le résultat n'aura plus rien en commun avec l'éclairage *naturel* de l'objet complet.

### A.3.3 Applications à des données réelles

#### Industrielles

En intérieur de centrale ou pour du suivi de chantier, les relevés laser sont rarement effectués de manière exhaustive sur toute la surface de l'objet, avec de nombreux points de vue et un souci systématique du détail. Les nuages sont donc très incomplets. Un éclairage global est alors peu réaliste comme on a pu le faire remarquer en fin de section précédente. Mais il est tout de même possible d'appliquer ce type d'éclairage à des zones restreintes ou des objets ciblés. On peut voir des exemples de rendu sur de tels objets figures A.8 et A.9.

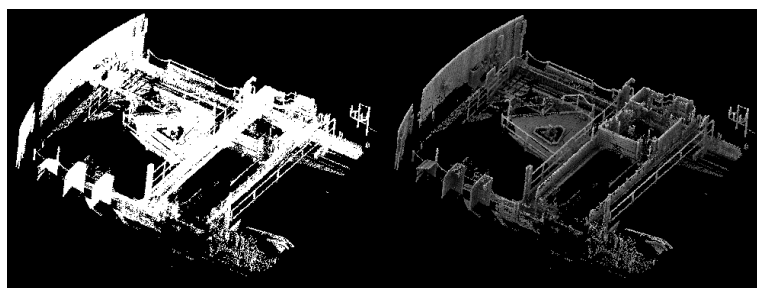


FIG. A.8 – Rendu en Portion de Ciel Visible sur un nuage en intérieur de centrale.  
(432 000 points, 10 secondes au niveau 8 de l'octree).

#### Archéologiques

Cette technique d'illumination (à partir de nuages de points  $2D^{1/2}$  ou 3D, ou encore sur des maillages) s'est avérée aussi très utile pour la représentation de relevés laser effectués lors de travaux archéologiques. Ces travaux concernaient la *Colonne des Danseuses*

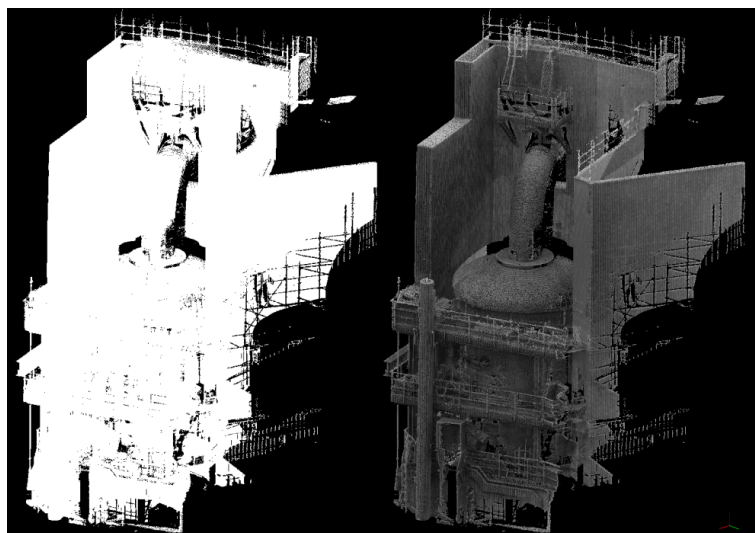


FIG. A.9 – Rendu en Portion de Ciel Visible sur un nuage en intérieur de centrale.  
(3 millions de points, 220 secondes au niveau 9 de l'octree).

à Delphes, imposant ouvrage d'une quinzaine de mètres (totalement écroulé) qui en particulier est couronné par l'*Omphalos*, sculpture en forme d'ogive qui symbolisait le *nombril du monde* dans l'antiquité. Ces travaux ont été sponsorisés par la fondation EDF. Un article a été publié à leur sujet (Cf. [Duguet et al. 2004]) et montre, entre autres, quelques résultats des deux algorithmes de calcul de *Portion de Ciel Visible* présentés dans [Duguet et Girardeau-Montaut 2004] et appliqués à des nuages de points représentant des morceaux de la colonne. Ce traitement présente en effet l'intérêt de ne pas modifier la position des données acquises par le scanner. Le résultat conserve ainsi un maximum d'*authenticité*, au sens où un traitement humain ou informatique n'a pas rajouté de déformation ou de biais à ces données déjà entachées d'erreurs. Cet aspect semble beaucoup toucher les archéologues qui s'appliquent à mettre le moins possible d'intermédiaires entre l'objet et sa représentation, pour que celle-ci soit la plus objective possible. Le style très minéral du rendu en *Portion de Ciel Visible* s'accorde aussi très bien avec le fait que les statues et autres éléments scannés étaient en pierre. Contrairement à une représentation avec des normales qui a un aspect très "métallique", le rendu en *Portion de Ciel Visible* est plus mat et plus proche des dessins d'archéologie classiques du *XIX<sup>ème</sup>* siècle. Un livre présentant, entre autres, de tels résultats est en cours de rédaction par Jean-Luc Martinez, conservateur en chef du Département des Antiquités grecques étrusques et romaines du Musée du Louvre, et ancien membre de l'École Française d'Athènes.

### A.3.4 Lien avec la détection de changement

Nous présentons maintenant un exemple de ce qu'il est possible de réaliser en termes de cartographie des changements, grâce à l'éclairage en portion de ciel visible. Le principe

est tout simplement de coupler l'information de distance codée en fausse couleur avec celle de la teinte donnée par l'éclairage en portion de ciel visible. Ceci permet de visualiser les déformations en même temps que la géométrie de l'objet. On peut voir un exemple de rendu de ce type en figure A.10. L'apport perceptif du rendu en portion de ciel visible est indéniable (surtout lorsqu'on compare l'image du bas et celle au milieu à gauche qui représente uniquement les distances en fausses couleurs).

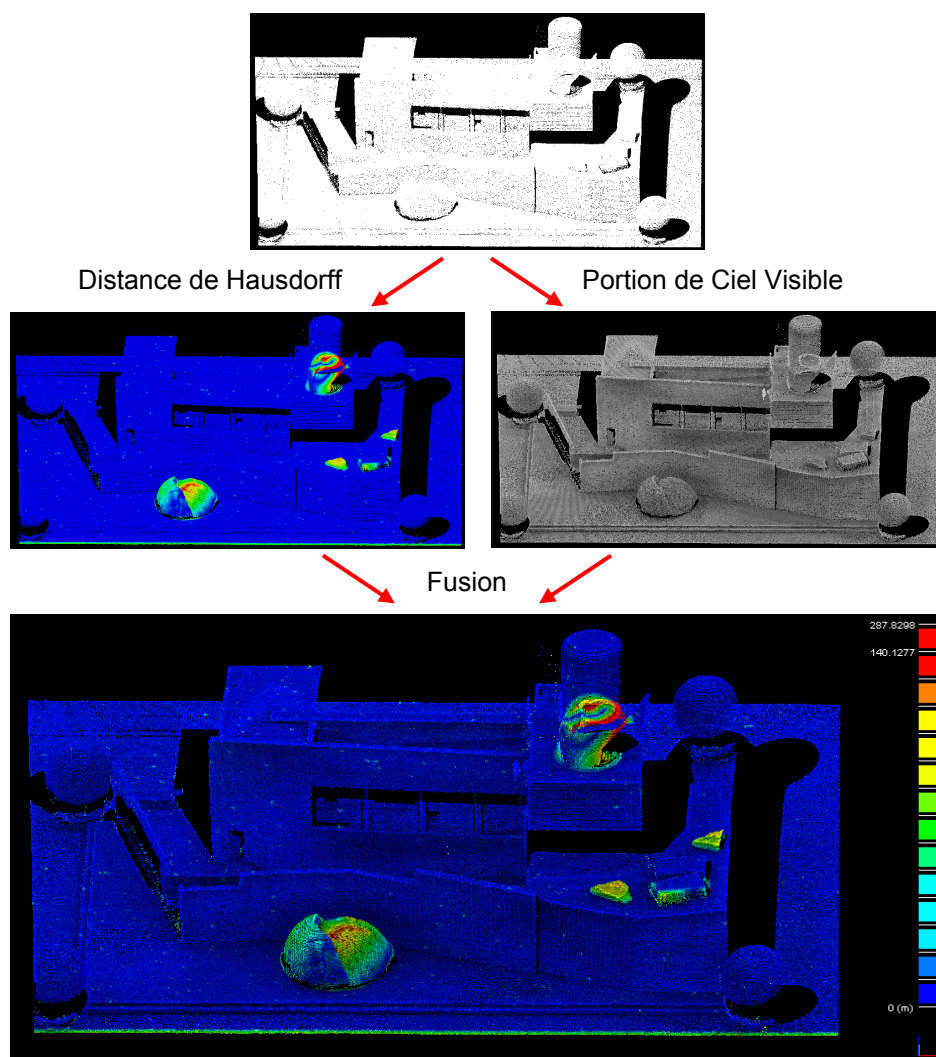


FIG. A.10 – Fusion entre rendu en Portion de Ciel Visible (niveau 9) et représentation en fausses couleurs du calcul de distance (entre deux nuages).

Une fois la portion de ciel visible calculée sur le nuage, il est d'ailleurs possible de réaliser la fusion des informations de distance et d'intensité en temps réel, à la volée. Ceci permet donc à l'utilisateur de continuer à faire varier l'échelle des valeurs (dans le cas d'une segmentation manuelle) pour produire une carte des distances optimale qui pourra être alors transmise sous la forme d'une image (ou même d'un nuage de points 3D coloré) à d'autres intervenants.

## Annexe B

# Taxonomie visuelle des changements

Cette annexe présente différents exemples de comparaisons "idéales" de données 3D (modèles ou nuages de points). Sont typiquement représentées : les données d'entrée (nuages/modèle "avant" et "après"), la carte des écarts dans chacun des sens de comparaison ("avant/après" et "après/avant"), et enfin ce qu'on pourrait idéalement attendre d'un processus de segmentation des changements, voire même d'un diagnostic avancé. Chaque type de changement évoqué dans la taxonomie établie en section 1.2 est illustrée.

**Type 1** : apparition/disparition

**Type 2-1** : déplacement sans recouvrement

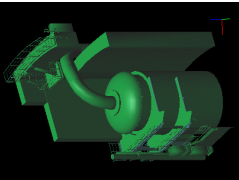
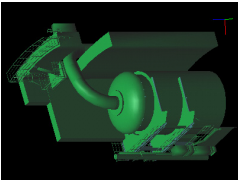
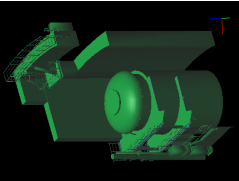
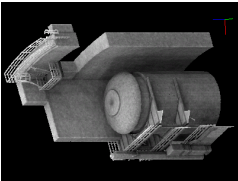
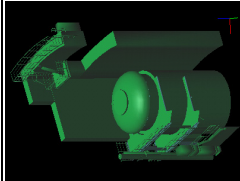
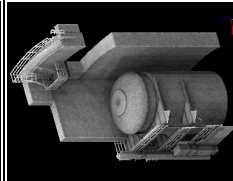
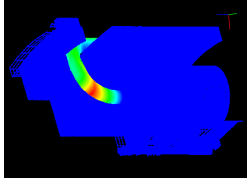
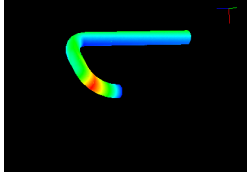

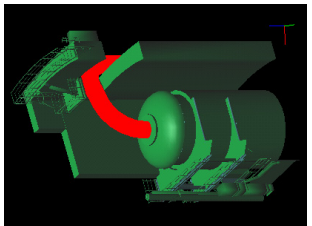
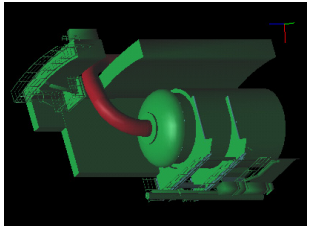
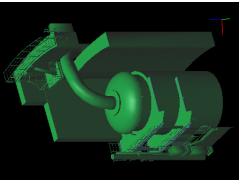
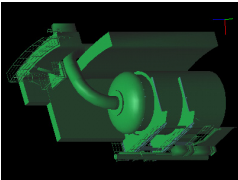
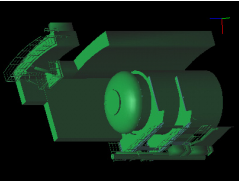
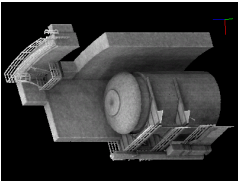
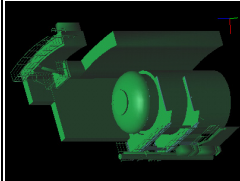
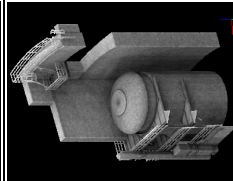
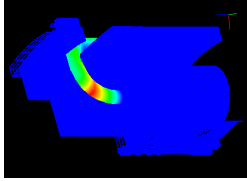
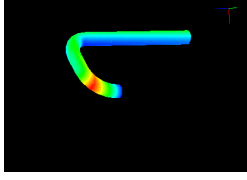

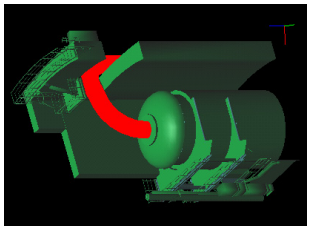
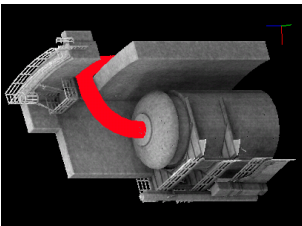
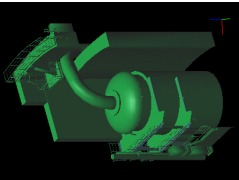
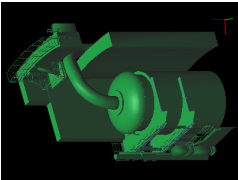
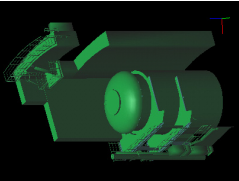
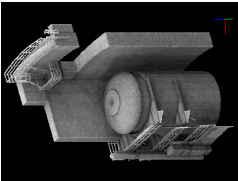
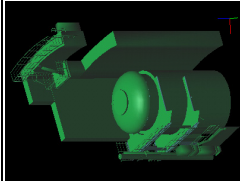
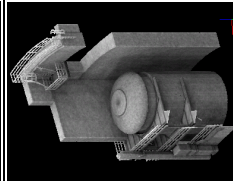
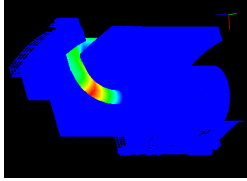
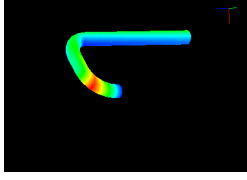

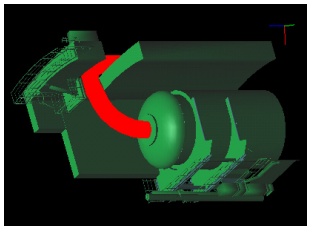
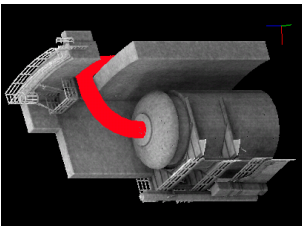
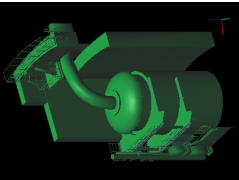
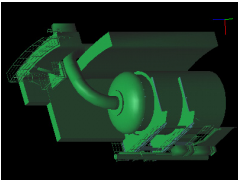
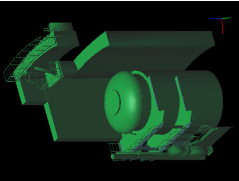
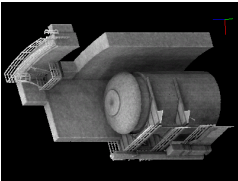
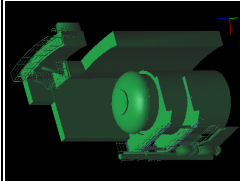
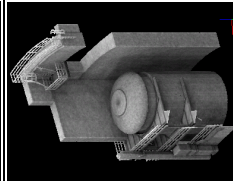
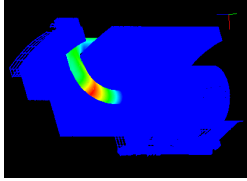
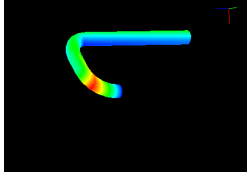

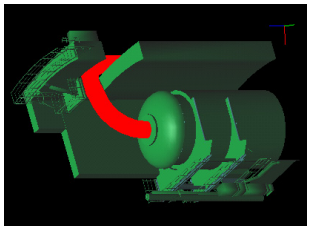
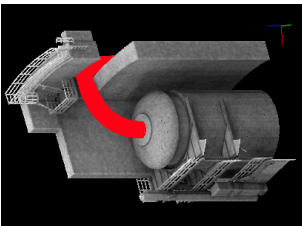
**Type 2-2** : déplacement avec recouvrement

**Type 3-1** : déformation sans recouvrement

**Type 3-2** : déformation avec recouvrement


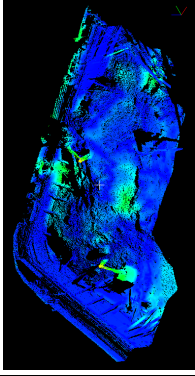
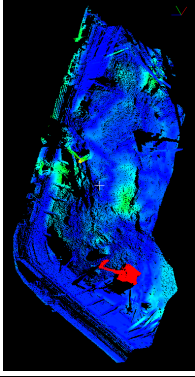
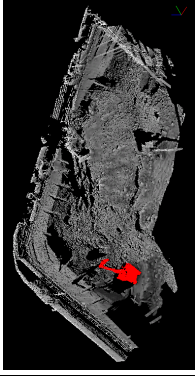
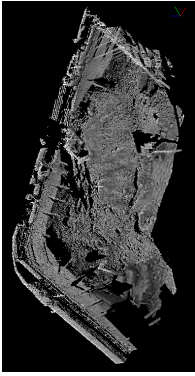
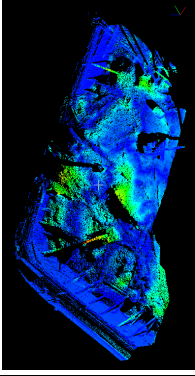
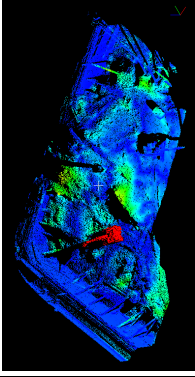
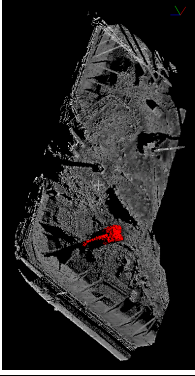
---

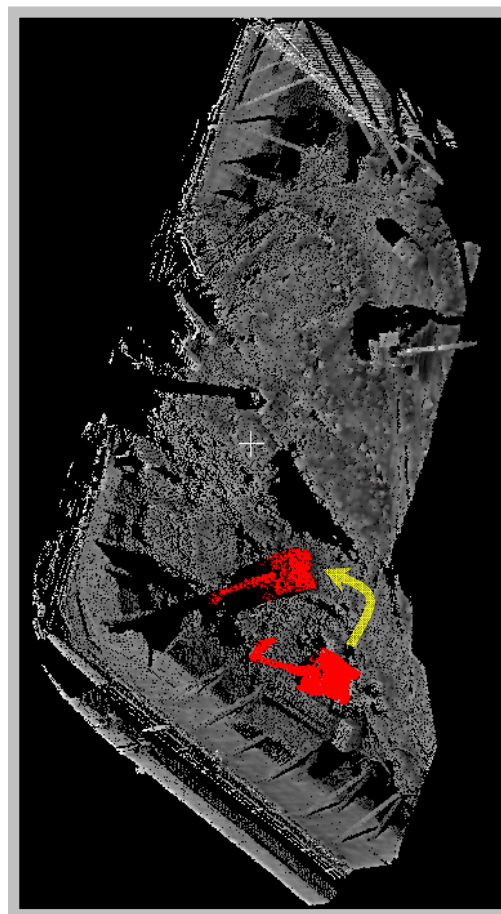
Type 1 – apparition/disparition

	référence		comparé				carte de distance → seuillage → segmentation			groupe des points	lien au modèle
modèle / modèle											
modèle / nuage											
nuage / modèle											
nuage / nuage											

*pas de modèle*

Type 2-1 – déplacement, sans recouvrement

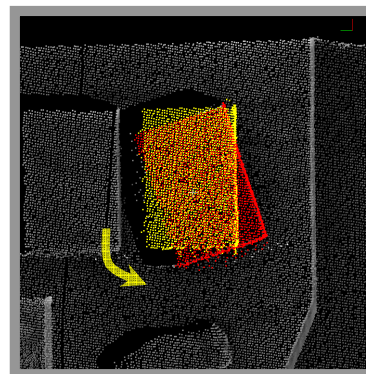
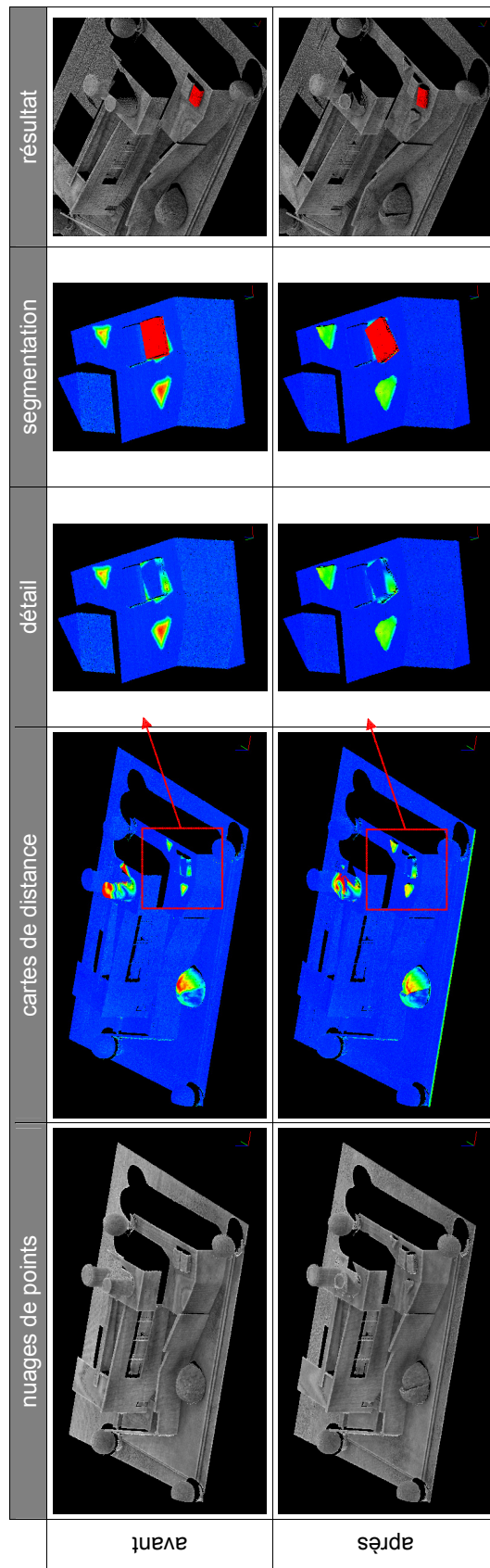
	nuages de points	cartes de distance	segmentation	résultat
Jour 1 (/jour 2)				
Jour 2 (/jour 1)				



Exemple de représentation idéale du mouvement  
(après appariement des groupes de points segmentés)

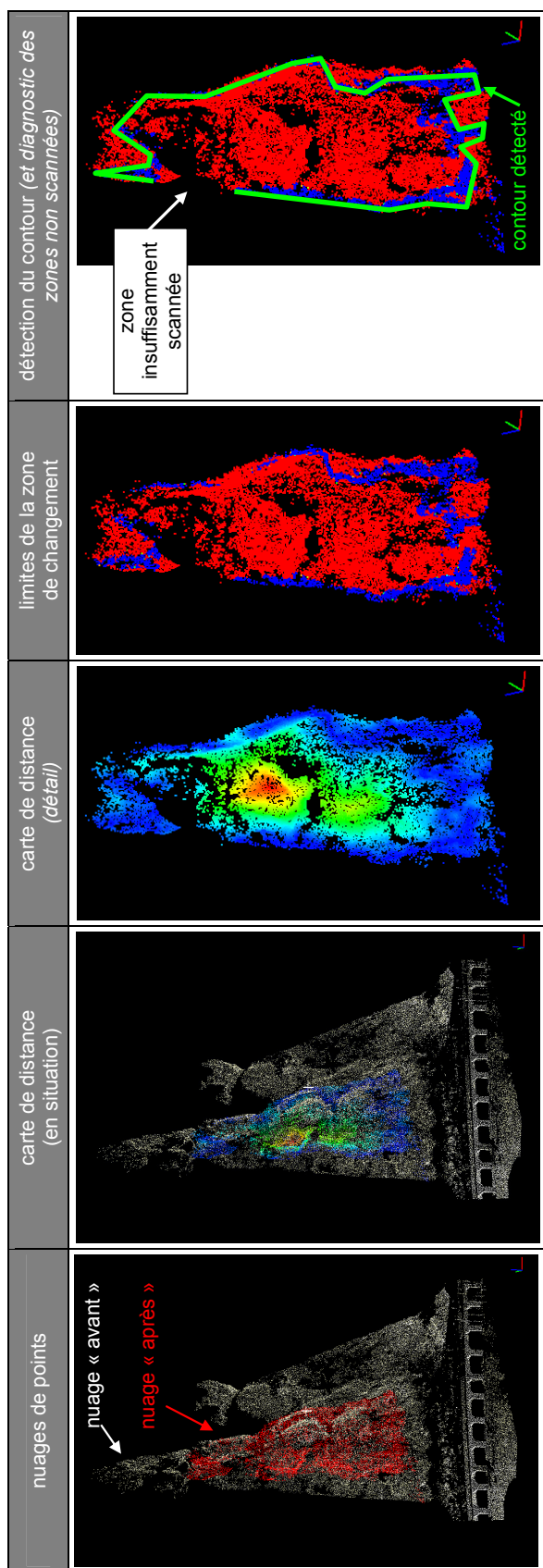


Type 2-2 – déplacement, avec recouvrement

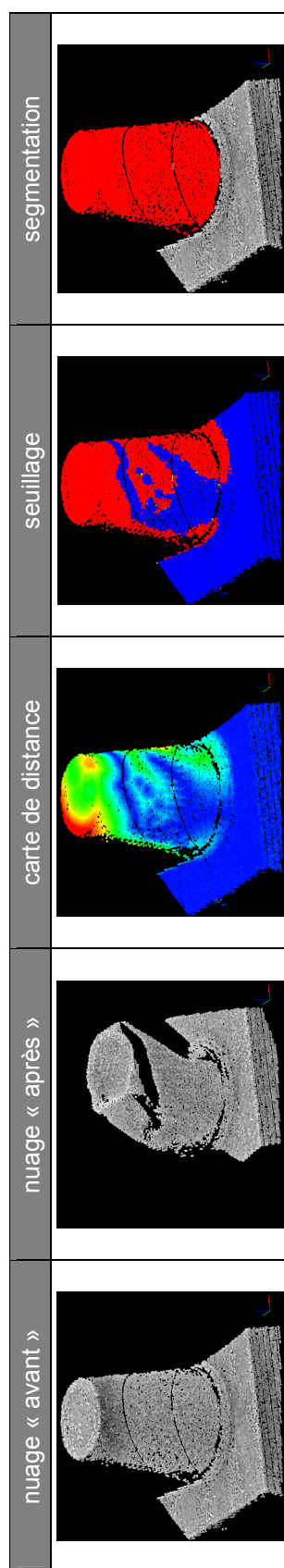


Exemple de représentation idéale du mouvement  
(après appariement des groupes de points segmentés)

Type 3-1 – déformation, sans recouvrement



Type 3-2 – déformation, avec recouvrement







## Annexe C

# Logiciel CloudCompare

### C.1 Présentation

*CloudCompare* est la plateforme logicielle de démonstration des algorithmes et techniques de comparaison mis au point au cours de cette thèse. Il a été développé en C++, sous Microsoft Visual Studio 6.0 (sous Windows donc) mais devrait être en théorie portable tel quel sous Linux. En effet, l'interface graphique repose sur la librairie Open Source FLTK (pour *Fast Light Toolkit*) qui est portable. Outre les éléments d'interface graphique, FLTK comporte aussi un environnement *OpenGL* pour l'affichage 3D.

Mis à part FLTK, le projet utilise plusieurs bibliothèques externes :

**FLU** (J. Bryan, *Ohio Supercomputer Center*, Ohio State University), qui est une collection d'éléments d'interface évolués pour FLTK, dont en particulier un arbre de navigation (visible dans la fenêtre gauche de la figure C.1).

**RPLY** (Diego Nehab, Princeton University), légèrement corrigée et modifiée pour marcher au mieux sous Windows, qui permet l'ouverture et l'écriture de fichiers au format PLY (format développé par l'université de Stanford).

**Triangle** (Jonathan Richard Shewchuk, Berkeley University), qui permet la création rapide et totalement contrôlée de triangulation 2D de nuages de points.

Le projet est conséquent. Il est composé d'une quarantaine de classes et d'approximativement 18 000 lignes de code dans sa version standard. A cela peuvent se rajouter plusieurs modules : l'un qui permet l'ouverture de fichiers Autocad DXF via une librairie (*dxflib*, A. Mustun & R. Campbell Jr.), ou de fichiers VRML/X3D via une autre librairie (*CyberX3D*, Satoshi Konno) et enfin un autre qui a été développé par S. Bougacha (stagiaire à EDF R&D, Département SINETICS, équipe CAO, été 2003) et qui permet la comparaison de données 3D acquises sur des aéroréfrigérants via une modélisation *Spline* du nuage de points photogrammétrique de référence.

---

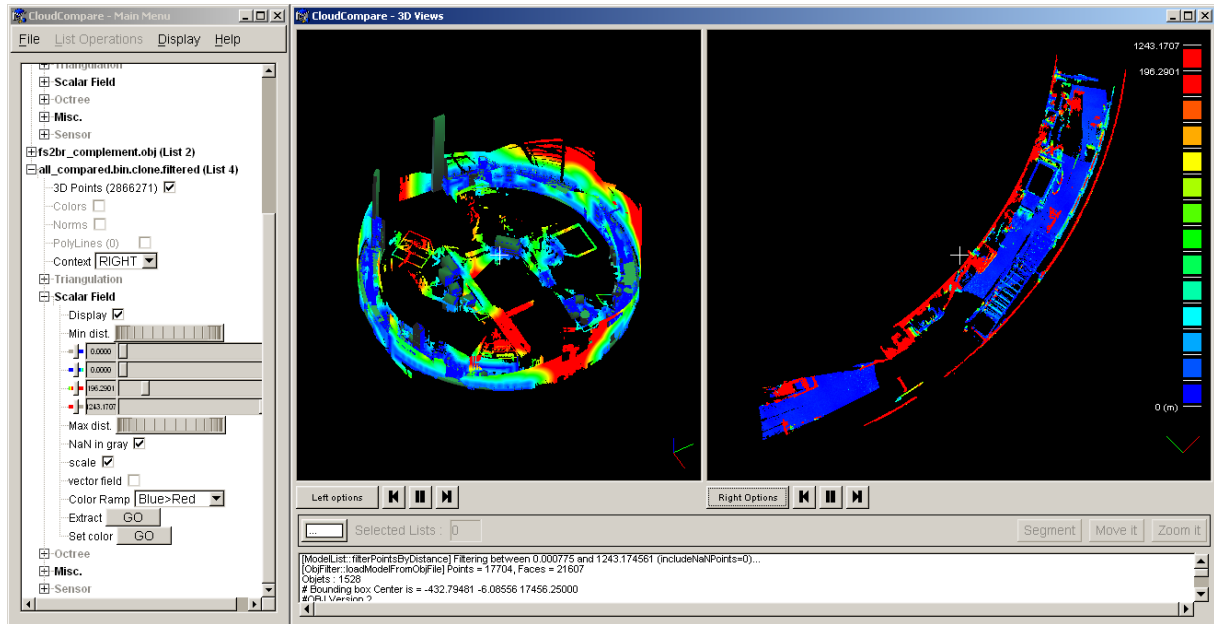


FIG. C.1 – Aperçu de l'interface graphique de *CloudCompare*.

## C.2 Fonctionnalités

**Couleurs** Couleur unique, Teinte unique, Dégradé selon une dimension, Calcul de *Portion de Ciel Visible*.

**Normales** Calcul par triangulation locale, Calcul par ajustement d'un plan aux moindres carrés, Calcul par ajustement d'une fonction quadratique, Inversion globale du sens.

**Octree** Calcul de l'octree, Décimation des points, Ré-échantillonnage, Extraction des composantes connexes.

**Triangulation (2D)** Calcul direct (verticale : axe des Z), Calcul par projection des points du nuage sur un plan ajusté par la méthode des moindres carrés (avec ou sans l'enveloppe convexe), Calcul de surface, Échantillonnage de points sur la surface.

**Capteur** Projection des points selon le point de vue du capteur (création d'une *Carte de Distances*).

**Distances** Distance entre deux nuages (au plus proche voisin, avec options d'optimisation par modélisation locale et/ou par prise en compte du point de vue), Distance entre un nuage et un modèle (type *Metro*), Extraction de l'ensemble des points les plus proches, Calcul de la transformée de distance de type Chamfrein <3,4,5>, Calcul de la transformée de distance par *Fast Marching*, Calcul d'une distance géodésique sur la surface.

**Champ Scalaire** Filtrage des points du nuage par seuillage sur les valeurs scalaires, Calcul de l'histogramme, Calcul du Gradient, Calcul d'un filtre Gaussien, Multiplication de deux champs scalaires.

**Statistiques** Calcul des paramètres d'une loi de Gauss ou de Weibull à partir de l'histogramme des distances, Test du  $\chi^2$  global, Test du  $\chi^2$  local.

---

**Segmentation** Algorithme des nuées dynamiques (K-Means), algorithme de segmentation par propagation d'un front contraint par la norme du gradient.

**Comparaison** Outil d'enchaînement d'algorithmes de comparaison, Recalage automatique de deux nuages de points (algorithme ICP de *Besl*[Besl et McKay 1982]).

**Transformations** Translation, Homothétie, Fusion.

**Développement** Développement cylindrique selon un axe (déterminé automatiquement ou spécifié manuellement), Développement conique.

**Divers** Calcul de densité locale, Calcul de courbure, Calcul de matrices de covariance (simple et croisée), Calcul de moments.

**Interface** Segmentation par définition de frontières polygonales à l'écran, Segmentation interactive en fonction de la valeur de distance, Affichage de l'échelle, Affichage d'une grille de mesure, etc.

**Fichiers** Chargement et Sauvegarde(\*) : ASCII(\*) (détection semi-automatique du format), SOI (Capteur *Soisic*), Binaire *CloudCompare*(\*), Binaire *TexMesh3D* (Carlos Hernandez, TSI), Binaire PN(\*) (Points+Normales), Binaire PCV (Points+Teintes), PLY(\*) (*Stanford*), OBJ(\*) (*Wavefront*), PWN(\*) (Points+Poids+Normales, pour MPU de Y. Ohtake[Ohtake et al. 2003]).

---



# Annexe D

## Données 3D

### D.1 Primitives élémentaires

#### D.1.1 Point

Le point est la primitive 3D la plus élémentaire. Elle consiste simplement en 3 coordonnées (une selon chaque dimension). Exprimées dans un repère muni d'une origine, ces coordonnées définissent la position unique du point. Un ensemble de points est appelé *nuage de points*. Si le nombre de points est suffisant, on verra que ces *nuages de points* peuvent représenter de manière fidèle des objets complexes en définissant leur surface de manière implicite.

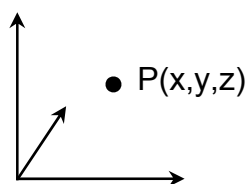


FIG. D.1 – Point 3D

#### D.1.2 Segment et polyligne

Un segment est défini par deux points (appelés aussi sommets) et correspond à la portion de droite qui passe entre ces deux points. Une polyligne est définie par  $n$  sommets

---

ordonnés et correspond à la succession des segments passant par chaque sommet (mis à part le dernier) et son successeur. Une polyligne peut-être *ouverte* ou *fermée*, auquel cas le premier et le dernier sommet définissent aussi un segment (le contour est ainsi *fermé*). Les points d'une polyligne 3D ne sont pas nécessairement coplanaires.

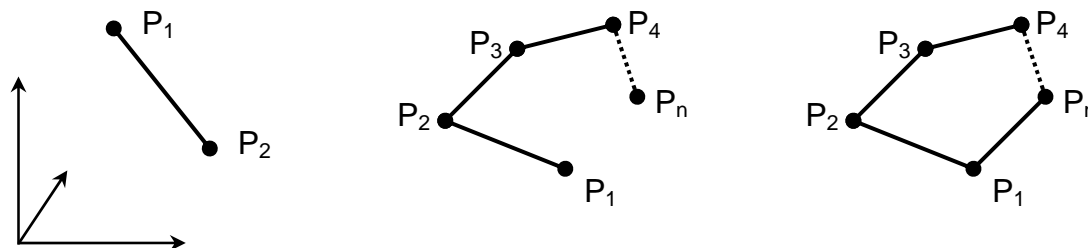


FIG. D.2 – Segment, polyligne ouverte et polyligne fermée

### D.1.3 Polygone et polyèdre

Un polygone est la portion de plan délimitée par une polyligne fermée (les sommets de la polyligne sont dans ce cas coplanaires). En général on considérera plutôt des polygones simples (avec 3 ou 4 sommets, c.à.d. des triangles ou des parallélépipèdes) mais il n'y a aucune limitation. Chaque segment de la polyligne est appelée *arête* du polygone.

Un polyèdre est un volume délimité par un ensemble de polygones. Chaque polygone est appelé *face* du polyèdre.

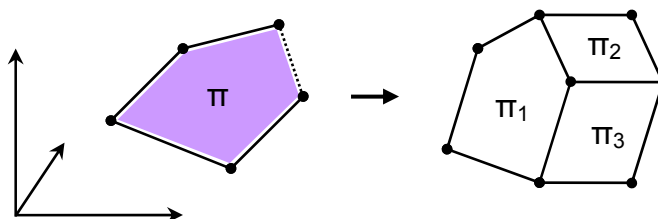


FIG. D.3 – Polygone (à droite) et polyèdre (à gauche)

## D.2 Représentation de surfaces

### D.2.1 Maillages

Un maillage est l'approximation d'une surface lisse par un ensemble de polygones partageant un ensemble fini de sommets et d'arêtes (qui forment alors un *graphe* ou un *réseau*). Les polygones sont généralement adjacents et on évite autant que possible la présence de trous dans le maillage (sauf si bien sûr la surface approximée comporte effectivement des orifices).

### D.2.2 TIN

Un TIN (Triangulated Irregular Network - Réseau irrégulier triangulé en français) est un maillage particulier formé uniquement de triangles. C'est typiquement le résultat d'algorithmes de triangulation comme celui de *Delaunay*. C'est aussi couramment, avec la grille  $2D\frac{1}{2}$  (voir ci-dessous), le format typique des modèles 3D représentant des portions de terrain et qui sont appelés M.N.T. et M.N.S.<sup>1</sup>. Les systèmes de mesure donnent plus ou moins directement une représentation du terrain avec le sur-sol (M.N.S.). Il est cependant possible de déduire un M.N.T. (uniquement le sol) à partir d'un M.N.S. par filtrage manuel ou automatique (avec plus au moins de succès dans ce deuxième cas) [Vosselman et al. 2002].

### D.2.3 Grille $2D\frac{1}{2}$

Une grille 2D est un autre maillage particulier dont les sommets et les arêtes sont tous disposés sur une grille régulière selon deux dimensions. La 3ème dimension est par contre libre pour chaque sommet.

### D.2.4 Splines, Surface de Bézier et NURBS

Nous évoquons ici différents modèles dits *paramétriques* couramment utilisés dans les applications de CAO.

Nous ne rentrerons pas dans les détails de la théorie des splines[DeBoor 1978] mais en voici un résumé : une courbe spline est une suite de segments de courbe polynomiale

---

<sup>1</sup>Modèles Numérique de Terrain, qui ne représentent que le sol, sans le *sur-sol*, contrairement aux Modèles Numériques de Surface

---



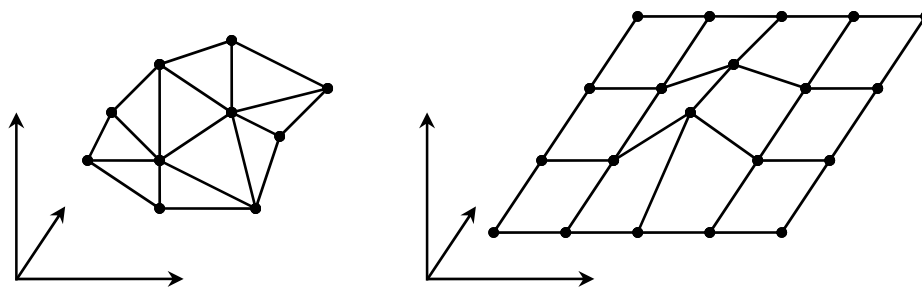


FIG. D.4 – T.I.N. (à droite) et Grille  $2D\frac{1}{2}$  (à gauche).

reliés par des points de raccord où la continuité de la courbe et parfois de ses dérivées est contrainte (classe  $C^1$  ou  $C^2$  en général). Ces points de raccord sont appelés *points de contrôle*. Une surface spline est le produit de courbes splines selon deux directions. Les points de contrôle sont alors disposés selon une sorte de grille régulière (voir figure D.5). Ce type de surface permet un contrôle total des déformations.

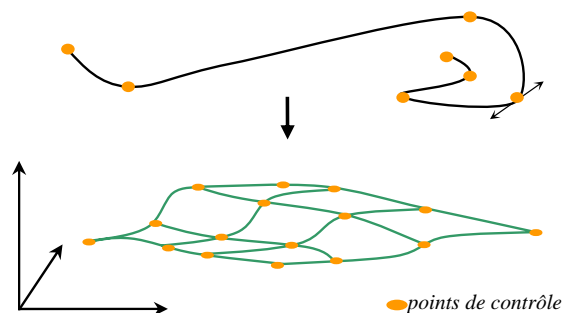


FIG. D.5 – Courbe spline (en haut) et surface spline (en bas).

Une surface de Bézier<sup>2</sup> est un cas particulier des surfaces splines (c'est une surface bicubique, soit le produit de deux splines de degré 3). On appelle *carreau de Bézier* un sous-ensemble de la grille de  $4*4$  points de contrôle. Les *carreaux de Bézier* sont très utiles car ils peuvent être utilisés sous forme de mosaïques pour créer des surfaces complexes tout en permettant un contrôle local de la déformation.

Enfin, les NURBS[Piegl 2001] (Non Uniform Rational Beta-Splines) sont des surfaces de Bézier particulières où l'influence de chaque point de contrôle est pondérée via un espace de paramètres.

<sup>2</sup>Pierre Bézier (1910 - 1999)[Pouget et Demengel 1998] : ingénieur à la régie Renault

### D.2.5 Surface de subdivision

Une surface de subdivision est définie comme la surface limite générée par une infinité d'opérations de subdivision appliquées sur un polyèdre grossier. Elle peut donc modéliser une surface lisse par morceaux de topologie arbitraires (contrairement à un modèle NURBS qui est défini sur un rectangle paramétrique) tout en gardant une représentation simple (un maillage polygonal) et une quantité de données faible. De plus une surface de subdivision est un modèle intrinsèquement multi-résolution (ce qui permet de régler la précision de sa représentation lors de l'affichage par exemple).

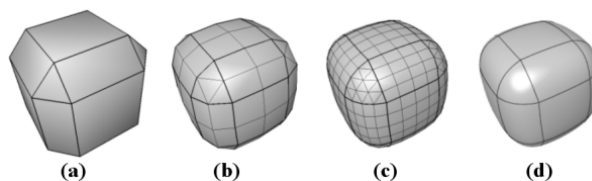


FIG. D.6 – Exemple de surface de subdivision (règle *Quad-Triangle* de Stam & Loop).  
(a) Polyèdre de contrôle, (b,c) 1 et 2 niveaux de subdivision, (d) surface limite

### D.2.6 Géométrie Implicite (C.S.G.)

Une dernière grande classe de modèles 3D est celle de la géométrie implicite (C.S.G.<sup>3</sup> en anglais). Elle repose sur un principe de construction des objets 3D par application de règles tirées de l'algèbre de Boole (union, différence, intersection, etc.) entre primitives 3D. On part en général de primitives simples (cube, sphère, cylindre, cône, tore et prisme). La forme finale de l'objet est obtenue par opérations successives entre des primitives simples ou des primitives résultant d'associations précédentes. La description de l'objet peut donc être codée sous la forme d'un arbre binaire et offre ainsi la possibilité d'obtenir des descriptions extrêmement compactes et parfaitement exactes d'objets en 3D. La création de tels modèles peut être par contre très fastidieuse, en particulier si l'on veut un résultat détaillé, et leur exploitation est complexe. En pratique, on utilise le principe de combinaison booléenne comme méthode de création d'un objet, puis le résultat est maillé de manière plus *traditionnelle* pour une future exploitation.

---

<sup>3</sup>Constructive Solid Geometry

---



## Annexe E

### Articles

**E.1 CHANGE DETECTION ON POINTS CLOUD DATA ACQUIRED WITH A GROUND LASER SCANNER**

**E.2 RENDU EN PORTION DE CIEL VISIBLE DE GROS NUAGES DE POINTS 3D**

Non présent dans cette version du manuscrit.

**E.3 A POINT-BASED APPROACH FOR CAPTURE, DISPLAY AND ILLUSTRATION OF VERY COMPLEX ARCHEOLOGICAL ARTIFACTS**

Non présent dans cette version du manuscrit.

---

# CHANGE DETECTION ON POINTS CLOUD DATA ACQUIRED WITH A GROUND LASER SCANNER

D. Girardeau-Montaut<sup>a,b</sup>, Michel Roux<sup>a</sup>, Raphaël Marc<sup>b</sup>, Guillaume Thibault<sup>b</sup>

<sup>a</sup> Telecom Paris (Ecole Nationale Supérieure des Télécommunications, Paris) (daniel.girardeau-montaut, michel.roux)@enst.fr

<sup>b</sup> EDF R&D (Electricité de France – Direction des Etudes et Recherches, Clamart) (raphael.marc, guillaume.thibault)@edf.fr

## Commission III WG III/3

**KEY WORDS:** Ground LiDAR, laser scanning, points cloud comparison, change detection, monitoring.

### ABSTRACT:

Ground based laser scanning is now well settled in the fields of cultural heritage, as-built modelling and monitoring applications. We intend to use laser scanning to detect changes on building sites or inside facilities, mainly for monitoring purposes but also to be able to provide synthetic information in case of an emergency. In both cases, the main constraint is time. A minimum precision on the measure of movements is also required, depending on the type of application. Laser scanner seems to be the perfect tool for such applications as it quickly acquires a large amount of accurate 3D data. But the comparison of so huge datasets implies the use of appropriate structures and ad-hoc algorithms. A specific octree structure is described, and then several simple cloud-to-cloud comparison techniques are presented. The best one, based on the Hausdorff distance computation is improved on various points. Also, as a full automatic process seems still unachievable, a software framework has been developed. It intends to minimize human intervention and therefore prevents from wasting the LiDAR speed in time-consuming post-processing operations.

## 1. INTRODUCTION

Change detection on 3D data is a rather new technique for non-professional users, considering that it has been done exclusively by professional surveyors by now. The main reason is that before laser scanning, the techniques to acquire 3D points on real scenes were topography and photogrammetry, which are very precise but time consuming (especially if one is interested in tiny details). It can hardly be done by non-professionals or in an automatic way. Moreover it is actually very inconvenient to apply such techniques in dense industrial context, with pipes and cables everywhere. Today, aerial or ground based laser scanners acquire much more information in a global but unstructured way.

A common technique to perform change detection on a point cloud is to compute its distance to a 3D reference model. It can be done either directly or by creating an intermediary model on top of the points. The reference model can either be theoretical or also created from real data. Point-to-mesh and mesh-to-mesh distances have been very well studied and demonstrated by academic software such as Metro (P. Cignoni et al. 1998) or Mesh (N. Aspert et al. 2002). Both need time and advanced treatments.

In our process, we could use automatic 3D shape reconstruction on a point cloud. It is a challenging problem that is widely and intensively studied. However, this technique still faces big issues that prevent us from using it: important computation time, especially on true 3D clouds with millions of points, and a very bad behaviour on complex or dense scenes with incomplete coverage.

In the particular case of aerial laser scanning, the sensor displacement implies that the acquired dataset is  $2D\frac{1}{2}$ . Therefore it can be rather easily modelled with polygons or transformed into a depth image. This has led to the development of specific comparison techniques, mainly for urban areas

(Murakami et al. 1998, Vögtle et al. 2003, Vosselman et al. 2004). The main application is database updating.

Finally, we can refer to the work of Mémoli and Sapiro (Mémoli et al. 2004) who have deeply analyzed the direct point clouds comparison problem in a very formal way. However, the point clouds have to be once again uniformly sampled with a good coverage, they must represent unique objects (manifolds) and the comparison result is global.

It seems that no specific method has been developed for direct cloud-to-cloud comparison of ground-based LiDAR data. Approaching techniques in the field of as-built verification or building site monitoring consist either in binary comparison of point clouds for material appearance/disappearance mapping (Shih et al. 2004) or the comparison of a laser scanned point cloud with an existing as-built 3D model in a dense industrial environment (Gonçalves et al. 2003).

In this paper we present a method for the direct comparison of two or more point clouds acquired with a ground-based laser scanner in order to detect local changes under important time constraints. Section 2 deals with data acquisition and registration issues. In section 3 we present naive comparison methods which only need a light structure computation (octree). In section 4 we discuss their results and carry out several improvements, providing some assumptions made in section 2 are respected. Conclusions are presented in section 5.

## 2. DATA ACQUISITION AND REFERENCING

The first test consists of a building site monitoring application. The dataset has been acquired on a common downtown building site, during the excavation phase (at the very beginning of the building process). We used a TRIMBLE GS200 ground-based laser scanner. Point clouds were acquired every day, from almost the same position and orientation each time (mainly for

stability reasons). There are 4 scans, about 200.000 points each (See Figure 1).

The second test consists of a change detection application inside a power plant (a highly obstructed environment). After ten years of intense scanning of its facilities, EDF (Electricity of France) owns a huge database of laser scanned point clouds. Some are acquired in the same area at different epochs; others correspond to different areas but to equivalent material. Most of them contain millions of points.

For both tests, knowledge of the laser scanner position and orientation is assumed. If point clouds are composed of multiple registered scans, the scan membership information for every point is also required for the improvement step that is proposed in section 4. There is no limitation on the number of scans or the total amount of points.

The registration process is done by standard techniques (physical targets, point matching, etc.). Of course, registration error directly interferes with the change detection process and may generate false detections. We note that automatic registration procedures and especially those relying on the ICP algorithm (Besl et al. 1992) do not converge with too different point clouds: the standard deviation of the points corresponding to a change (which are equivalent to noise here) should be less than 10% of the object size. If it is not the case, one should keep only the points that have not – or nearly not – moved. As a first approximation, assuming both clouds are roughly registered, it is possible to set a threshold value and consider during the ICP process only points whose distance to their nearest neighbour in the other cloud is less than this value. If there are enough of them and if they are sufficiently well dispatched all over the scene, the iterative registration process will converge and give fairly good results.

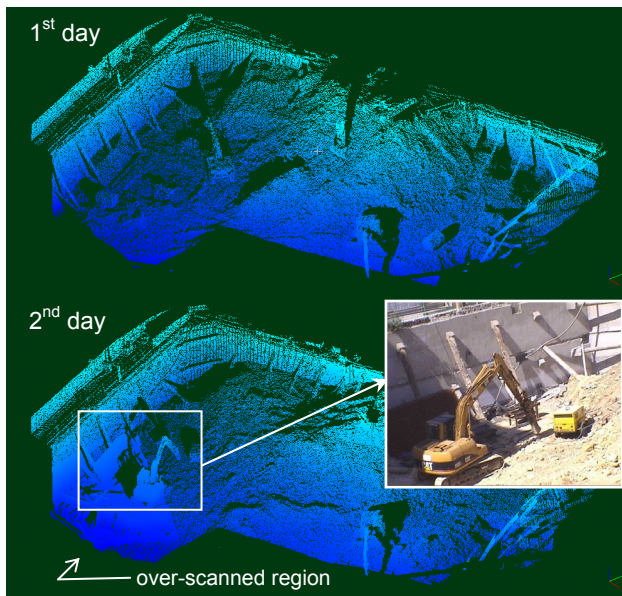


Figure 1 - Building site scans examples (two first days) coloured by height

### 3. DIRECT COMPARISON WITH OCTREE

Although we deal with large clouds (up to 30 millions of points), we wish to have an estimation of what has changed between two scans in a short or at least reasonable time. As

indicated in the introduction, meshing algorithms applied to so huge and complex data are slow and hazardous. Moreover, the memory usage issue is critical, and meshes are not the best structure to face it. Thus we use a specific octree structure to ensure a good ratio between memory usage and points neighbourhood extraction speed.

### 3.1 The octree structure

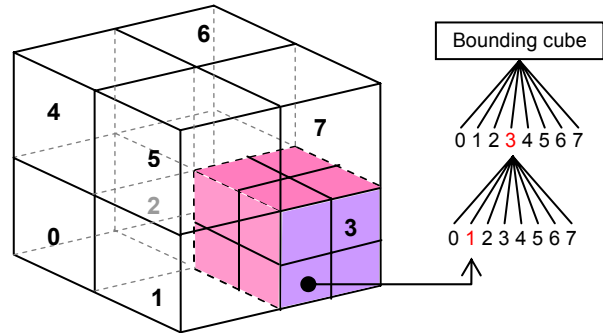


Figure 2 - Octree subdivision principle

An octree consists in a recursive and regular subdivision of the 3D space. In practical, the cubical bounding box of the set of points is divided into 8 equivalent cubes and this division process is recursively repeated for each cube. It stops when no more point lies in a cube or when a maximum preset level is reached. Such a structure gives the ability to rapidly determine which points lay in a specific cube and in its neighbouring cubes. Nearest neighbours extraction and equivalent processes become thus very fast.

An octree structure is generally (and logically) implemented with a tree structure but it is also possible to use a numerical coding scheme: for every point, a  $3*n$  bits long code is computed, where  $n$  is the maximum subdivision level of the octree. For a given level, 3 bits are used to code a number from 0 to 7. This number corresponds to the sub-cube (also called a cell) where the point lies. Each bit corresponds simply to the cell position relatively to its parent cube along one dimension. These numbers are concatenated for all successive levels to build the cell code (see Table 3). This coding step is very fast as the coding process is linear (it has a complexity of  $k.O(N)$  where  $N$  is the number of points and  $k \approx n$ ).

Level	1	2	3	...	n
Bits	$b_z b_y b_x$	$b_z b_y b_x$	$b_z b_y b_x$	...	$b_z b_y b_x$
	<i>most significant bit</i>				<i>least significant bit</i>

Table 3 - numerical coding of an octree cell position

Eventually, codes are sorted to make this octree structure more efficient. It makes possible to find rather quickly a given code with a simple dichotomic search, or all the points that lay in a cell at a given level  $l$  by considering only the first  $3*l$  bits of codes. The sorting step has a complexity of  $O(n.log(n))$ . This octree structure is simple to implement and offers a good flexibility (it is easy to add or suppress points).

We will close this discussion about implementation by referring to more sophisticated octree structures that would perfectly match our expectations - but for a highest implementation cost - such as Botsch et al. 2002 or Duguet and Drettakis 2004.

### 3.2 Octree-based comparison process

Assuming that clouds are expressed in the same reference frame, a mere idea is to apply the same octree subdivision process to all of them. Each octree structure is computed starting from the same initial cube which is the smallest cube that bounds all clouds. By this way, similar cells in all octrees are spatially equivalent. Therefore, one can expect that the subsets of points lying in these homologous cells are indeed “comparable” or at least, are part of the same object.

Based on this principle, a lot of “homologous cells comparison strategies” can be defined. They are generally as slow as they are accurate. Below are examples of such strategies. Let  $S$  and  $S'$  be two subsets of points contained in two homologous cells at a given subdivision level  $l$ . Let  $N$  and  $N'$  be respectively the sizes of  $S$  and  $S'$ . By default,  $S$  will be the compared cloud and  $S'$  the reference cloud (the comparison process is not symmetrical).

#### Strategy 1 – “Average distance”

All points of  $S$  are labelled with the same value which is the average of the distances between each point of  $S$  and its nearest neighbour in  $S'$  (it could also be the minimum of maximum distance). This can be quickly computed if  $l$  is big enough or equivalently if  $N$  and  $N'$  are rather small (see Table 8). It is robust to noise and density variations. But it is very inaccurate and should only be used in a preview step (see Figure 4).

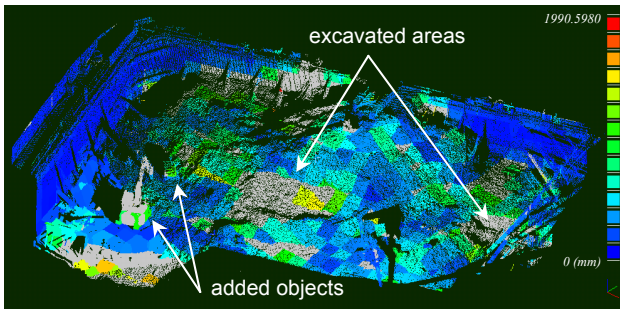


Figure 4 - “average distance” between the two first scans of the building site, computed at octree level 5 (points that have not been compared are displayed in grey).

As distance approximations are not exactly the same for different values of  $l$ , better results are achieved by applying the process recursively, keeping for each point the minimum value from a level to another (see Figure 5). However, some points are left not compared, as they are too far from the reference cloud (they lay in octree cells that have no homolog).

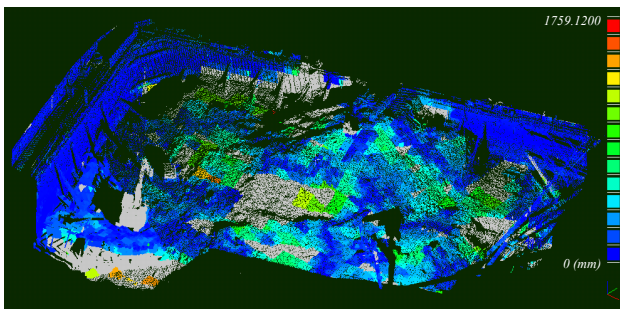


Figure 5 - “average distance” between the two first scans of the building site, recursively computed from octree level 5 to 10.

#### Strategy 2 – “Best fitting plane orientation”

The least square plane is computed for both subsets. All points of  $S$  are labelled with the same value which is the angle between these planes. This is a simple tilt change approximation (and therefore a kind of shape modification information) but shifts are not detected. Once again, this is only a *quick and rough* evaluation of what has changed. We observe similar results compared to the first strategy (see Figure 6, to be compared with Figure 5). Once again, a recursive approach gives better results.

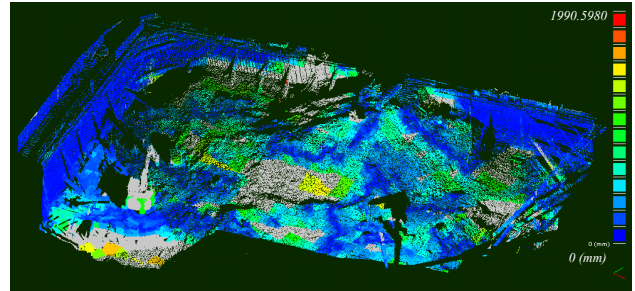


Figure 6 - “best fitting plane orientation” recursively computed from octree level 5 to 10

#### Strategy 3 – “Hausdorff distance”

The Hausdorff distance between two sets of points is a common distance that consists in computing for each point  $p$  of a cloud  $S$  the distance to its nearest point in the other cloud  $S'$ :

$$d(p, S') = \min_{p' \in S'} \|p - p'\|_2 \quad (1)$$

This is much more precise than the two previous methods but slightly slower. It is also very dependent on the point density variations between scans as it does not consider any implicit or explicit surface, but only points. The Hausdorff distance computation relies also on the octree structure but the nearest neighbour of a point does not lie necessarily in the homolog cell. For each point  $p$ , in order to assure that the truly nearest point has been founded, one must compute the distance to all points included in the neighbouring cells until the minimum distance is lower than the radius of the biggest sphere centred on  $p$  and fully included in this neighbourhood. There are no un-compared points by definition (see Figure 7). Note that the subdivision level at which the calculation is done does not change the result but only the computation timing (see Table).

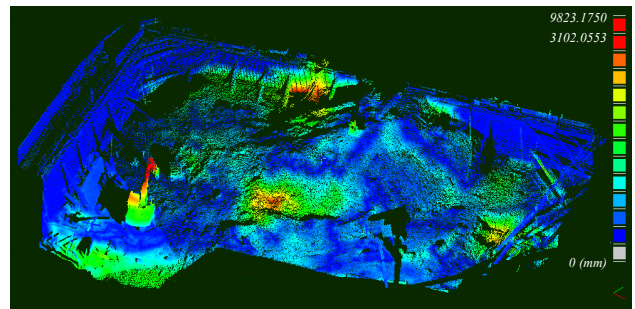


Figure 7 – “Hausdorff distance”

### 3.3 Computation costs and early results discussion

Here are presented indicative computation timings for the strategies presented above (see Table 8). These tests have been



made on a notebook with a 1.7 GHz Centrino™ processor and 1 Go of memory. The first tested dataset is composed of the two first scans of the building site (170.000 and 765.000 points). The second dataset is composed of two clouds made of multiple consolidated scans and acquired inside a power plant (850.000 and 612.000 points). A third dataset, composed of two clouds of 12 million points each, is used for scalability test.

We observe that strategies relying only on homologous cells subsets comparison are of course faster than the true Hausdorff distance. However, their output is so approximate that they should only be used as preview processes. The Hausdorff distance is surely the best solution to our problem. It has however some important issues:

- As it has already been said, it is very sensitive to point sampling variations between scans.
- Its computation time depends greatly on the octree level (see Table 9) and the points repartition in space.
- A point will always have a nearest point in the other scan but it does not necessarily mean that their distance corresponds to any real change.

Process	dataset 1*	dataset 2*
octrees computation	01.25 s	01.76 s
strategy 1 <i>average</i>	02.21 s	14.97 s (6)
strategy 2 <i>one pass</i>	00.11 s (5)	00.19 s (6)
strategy 2 <i>recursive</i>	00.47 s (5-10)	01.33 s (6-10)
strategy 3 <i>Hausdorff</i>	07.07 s (7)	21.38 s (8)

Table 8 - computation costs comparison

(\*) numbers in braces are the octree levels used for computation that approximately correspond to the same number of points per cell

octree level	dataset 1	dataset 2	dataset 3
5	36.11 s	n.a.	n.a.
6	12.88 s	129.47 s	n.a.
7	07.07 s	042.78 s	n.a.
8	24.03 s	021.38 s	563.59 s
9	n.a.	116.32 s	657.64 s

Table 9 - octree level influence on Hausdorff distance computation time (best in yellow)

To address the first issue, we suggest to model locally the surface to avoid density variation issues. For each point of the first cloud, once the nearest point in the second cloud has been determined, a set of its nearest neighbours is extracted. With this set, a local surface model is computed (the least square plane, a Delaunay triangulation, a spline surface or a quadratic height function for example). Finally, the distance from the point of the first cloud to this surface patch is computed. By this way, any point sampling influence is avoided (see Figure 10). It can still remain error, depending on the chosen model, especially on highly curved or angular surfaces but it is much smaller. It should also be an optional treatment as it is not always useful while rather time consuming (with an appropriate algorithm and Delaunay triangulation, a multiplication of the process time by 2 to 4 has been observed).

To fix the second issue, we compute the average number of points per cell for every level (for the cost of one table scan – linear process – thanks to the octree structure described in section 3.1). The smallest level of subdivision that gives an average number of points per cell below a certain limit (which depends greatly on the computer capacities but which should not be greater than a few hundreds of points) is then chosen for computation.

Finally, the last issue is surely the most challenging. It does not result from the Hausdorff distance itself but more likely because of the laser scanning principle. Imagine a laser sensor at a fixed position, scanning a changing scene at different epochs. Because of the displacements of occluding materials, the laser beam will not go through the same portions of the 3D space (even if the field of view remains the same for each scan).

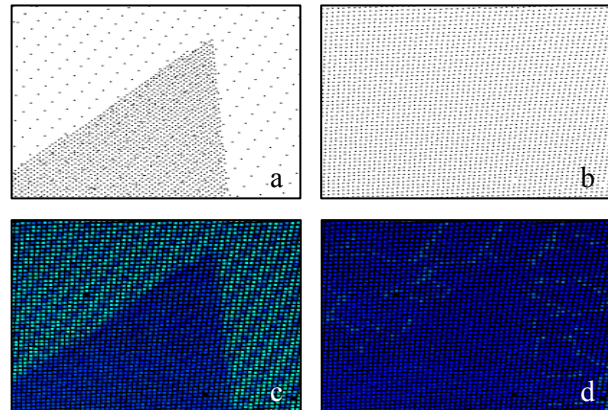


Figure 10 - Comparison of a set of non-uniformly sampled points (a) with uniformly sampled points (b). Points are sampled on the same plane. Results of the Hausdorff distance computed without (c) and with (d) local modelling (Delaunay triangulation).

Consider Figure 11: in case of disappearing material (top figure), the laser beam at epoch 2 goes through a portion of space that has never been scanned during the first epoch (the visible pink zone below the orange area). The fact is that it is impossible to have a clue about what may have changed in this shadow area. It is only possible to conclude that something has disappeared or moved in the first scan. However, a human - who have some *a priori* information on the scanned scene - would say that the two subsets of points on the left are indeed “comparable” while the points at the right have no homologous points in the first scan (bottom figure). A solution to this problem is proposed in the next section.

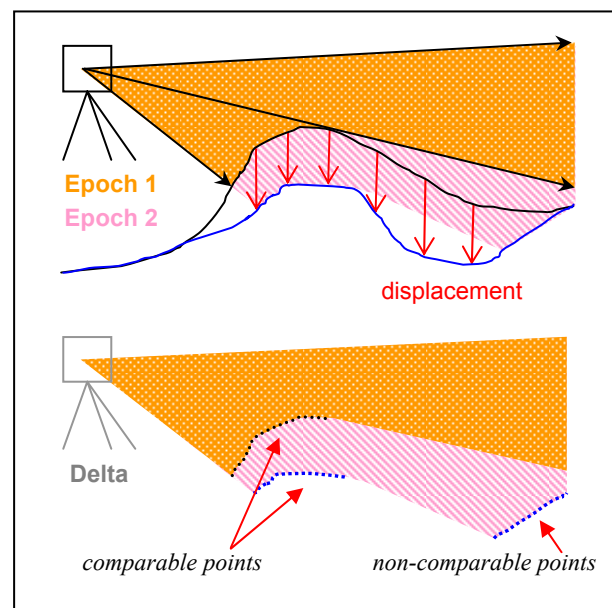


Figure 11 – scanner shadow areas



#### 4. IMPROVEMENT WITH DEPTH MAPS

In the previous section, we have pointed out a consequence of the laser scanning principle that prevents us from using the Hausdorff distance as a “change detector” in some specific areas. To solve this problem, points that lie in those areas are filtered out in a first step and then specific treatments (chosen by a human operator, thanks to *a priori* knowledge of the scanned scene) are applied to these points.

##### 4.1.1 Visibility maps

The filtering phase consists in sorting the points into two categories: each point of the compared scan lies either in an area of space that has been effectively observed by the sensor during the scanning process of the reference scene (i.e. the laser beam went through this area at both epochs), or in a shadow area of the first sensor.

The process relies on a Z-buffer (Watkins 1970) which is a common structure in computer graphics applications. As most of ground laser scanners use rotating mirrors and/or axes that move with constant angle steps, their 3D measures can actually be expressed in 2D½: 3D points are projected into a 2D grid in the scanner angular coordinate system (see Figure 12). This grid is called a “depth image” or “depth map” (see Figure 13) and is equivalent to a Z-buffer. Every pixel in this image corresponds to a given direction and its associated value is the distance from the scanner to the nearest obstacle in this direction.

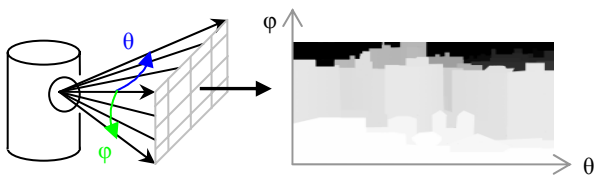


Figure 12 – scanning principle and depth image projection

Some holes may exist in the depth image as some laser pulses have not been reflected back. To fix this, a simple “hole-filling” process is applied to the image (holes are replaced by the mean value of its neighbours). Finally, if one knows also the maximum sensor range, it can be used to initialize the buffer (otherwise we initialise it with zero values).

The depth image is associated with the sensor position and orientation. With such a structure, the visibility of any 3D point can be determined: a queried 3D point is projected into the 2D image space; if the projected point is outside of the image, it means that the 3D point is outside the scanner field of view. Otherwise, the distance between the point and the scanner centre is compared to the buffer value at the same position. If the point is closer, it is labelled as *visible* (e.g. the corresponding space area has been actually visited by the laser beam) else it is *invisible*. This visibility test is made during the comparison process, before each nearest neighbour request. If it is positive, the Hausdorff distance is calculated, otherwise the invisibility type is stored (*out of field of view*, *out of range* or *hidden*). Invisible points will be processed in a specific way (see next section).

The visibility map is only computed for the reference cloud. If this cloud is composed of multiple scans, membership to the different point of views (*p.o.v.*) should be known for every point (as requested in Section 2). A visibility map is then

computed for each *p.o.v.*, and the visibility query is made for every *p.o.v.* until one test is positive (e.g. the point is *visible* from at least one *p.o.v.*). Otherwise the point is *invisible*.

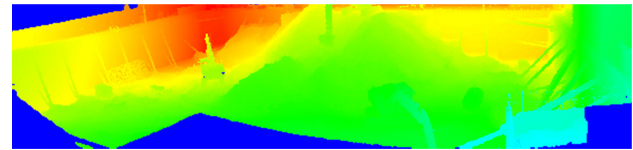


Figure 13 - depth image of the first scan of the building site (equivalent to the 1st day in Figure 1)

Note: as for graphical Z-buffers, an uncertainty value is used for the distance test. It is a positive epsilon variable added to the buffer range value during the test. It is a simple way to take errors into account (mainly sensor and registration errors), but also to arbitrarily consider small distances as differences and not as *hidden* configurations (depending on the objects sizes). For outdoor scenes with high ranges, we set epsilon between 10 cm and 1 meter. For indoor scanning, with lower ranges and a much higher precision, we use values between 3 cm and 10 cm.

##### 4.1.2 Invisible points handling

It appears that *invisible* points can be classified into three groups:

- Points that are *out of field of view* or *out of range*. We can forget them, as there is no way to compare them.
- Points that are labelled as *hidden* because there has been a shape modification or a shift of the object surface, farer from the scanner (a true change).
- Points that are labelled as *hidden* because something has disappeared or shifted, revealing what was behind it, in its shadow (see Figure 14).

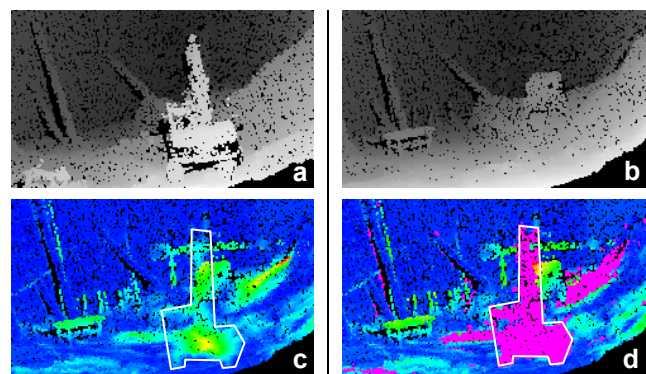


Figure 14 - Comparison of day 3 (b) with day 1 (a). With a simple Hausdorff distance (c), points in the shovel shadow are labelled as *different* (non-blue points inside the white polygon). After a visibility check (d), those points are labelled as *hidden* (in pink).

In order to make the distinction between the two *hidden* cases, a solution is to use a third scan. Given three scans acquired at three different epochs ( $t-1$ ,  $t$  and  $t+1$ ), let's consider a set of points that are labelled as *hidden* at epoch  $t$ . If those points have not moved between epoch  $t+1$  and  $t-1$ , then there is a great probability that they have not moved at epoch  $t$ . If they have moved at epoch  $t+1$ , the outside border of the shadow zone at epoch  $t$  can give a hint on what happened inside.

If there is no third scan or some ambiguities remain, different strategies can be proposed as a choice to the user:

1. To compare the remaining points in a 2D½ way (the user must specify a direction). It is most indicated for outdoor datasets or more generally for nearly 2D½ datasets. For example, the user may guess that gravity is the main reason for the displacement of some hidden points, and will therefore want to compare them to points vertically above or below them.
2. To compare the *hidden* points with their occluding points in the reference cloud (which can be identified via the Z-Buffer). If the two sets have at least partially the same shape, there is a good probability that they correspond to the same shifted surface.
3. To manually select and tag subsets of points with specific values.

Finally, whatever processes the user has chosen, he obtains one set of points labelled with distance values, and one set of points that could not be compared but which are labelled with visibility tags. This allows to produce accurate maps of 3D changes without false detection (see Figure 15-a).

To extract higher level information, a 3D connected components extraction process can be applied on this dataset. The user sets (graphically or numerically) a distance threshold to filter out points considered as unmodified (see Figure 15-b). Doing this, only points corresponding to “changes” remain under the form of isolated subsets of points (as they were connected to unchanged surfaces that have been filtered out). Then, the octree structure – considered as successive 2D slices of cells – is used to perform on each slice a simple binary connected components extraction process (cells can have two values: *empty* or *occupied*). The results are merged in 3D by regrouping components in the orthogonal direction (see Figure 15-c).

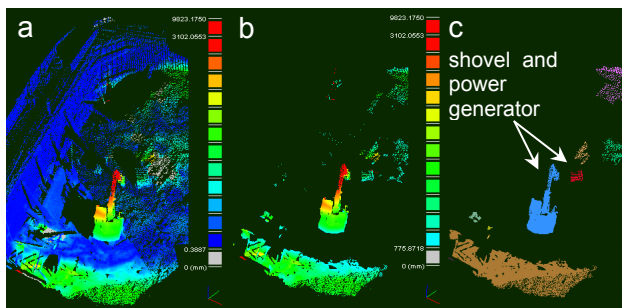


Figure 15 – Objects extraction principle: distance calculation (a), distance thresholding (b) and 3D connected components extraction (c).

## 5. CONCLUSIONS

We have presented a direct point-to-point comparison framework. Given two datasets acquired on the same area, one can use either preview processes (for on-site verification) or a more precise comparison process relying on Hausdorff distance. This last process can be refined if the sensor pose and scan membership information for every points are available. Speed constraint is assured thanks to the use of a light octree structure such as the one presented in section 3.1. Results can be used for basic change mapping or human driven verification, but also for higher level treatments, such as 3D connected components extraction and objects recognition. No time consuming process is needed, allowing the whole distance calculation and change detection process to be completed in a short time.

In case of an emergency or more generally if the user is not familiar with 3D point clouds, knowledge of the sensor parameters or manually setting threshold values can become critical issues. Therefore, we are now working on automatic distance thresholding to achieve unsupervised changing objects extraction (allowing automatic recognition of 3D objects for vehicle tracking or displacements monitoring). We are also working on automatic sensor pose estimation (directly from the 3D points repartition in space, without any information other than the sensor type). Another field of research is the classification and recognition of the different types of change that can occur on geometry, in order to produce cleaner and more informative results.

## REFERENCES

- N. Aspert, D. Santa-Cruz and T. Ebrahimi, 2002. MESH: Measuring Error between Surfaces using the Hausdorff distance. In: *proceedings of the IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 705-708.
- P. Besl and N. McKay, 1992. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14 (2), pp. 239-256.
- M. Botsch, A. Wiratanaya, and L. Kobbelt, 2002. Efficient high quality rendering of point sampled geometry. In: *proceedings of the Eurographics Workshop on Rendering 02*.
- P. Cignoni, C. Rocchini, and R. Scopigno, 1998. Metro: measuring error on simplified surfaces. *Computer Graphics Forum*, vol. 17(2), pp. 167-174.
- F. Duguet and G. Drettakis, 2004. Flexible Point-Based Rendering on Mobile Devices. *IEEE Computer Graphics and Applications*, n°4 vol. 24.
- J. G.M. Gonçalves, V. Sequeira, B. Chesnay, C. Creusot, S. Johnson and J. Whichello, 2003. 3D Laser Range Scanner for Design Verification. In: *proceedings of the 44th Meeting of the Institute for Nuclear Materials Management, Phoenix, Arizona*.
- F. Mémoli and G. Sapiro, 2004. Comparing Point Clouds. *Eurographics Symposium on Geometry Processing*.
- H. Murakami, K. Nakagawa, H. Hasegawa, T. Shibata and E. Iwanami, 1999. *Change detection of buildings using an airborne laser scanner*, ISPRS Journal of Photogrammetry & Remote Sensing, Vol. 54, pp. 148-152.
- N-J. Shih, M-C. Wu and J. Kunz, 2004. The Inspections of As-built Construction Records by 3D Point Clouds. *CIFE Working Paper #090*, Stanford University.
- T. Vögtle and E. Steinle, 2004, Detection And Recognition of Changes in Building Geometry Derived From Multitemporal Laserscanning Data, In: *proceedings of the XXth ISPRS Congress (Istanbul)*, Vol. 35, Part B2, p. 428-433.
- G. Vosselman, B.G.H. Gorte and G. Sithol, 2004, Change Detection for Updating Medium Scale Maps Using Laser Altimetry, In: *proceedings of the XXth ISPRS Congress (Istanbul)*, Vol. 35, Part B3.
- G. S. Watkins, 1970. A Real-time Visible Surface Algorithm. *Technical Report UTEC-CSc-70-101*, Dep. Comp. Sci., U. of Utah, Salt Lake City.



## Bibliographie

- [Aspert et al. 2002] N. Aspert, D. Santa-Cruz, T. Ebrahimi, "MESH : Measuring Error between Surfaces using the Hausdorff distance", in *proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, vol. I, p. 705-708, 2002.
- [Baltsavias-1 1999] E. P. Baltsavias, "Airborne laser scanning : basic relations and formulas", in *ISPRS Journal of Photogrammetry & Remote Sensing*, Vol. 54, p. 199-214, 1999.
- [Baltsavias-2 1999] E. P. Baltsavias, "Airborne Laser Scanning : existing systems and firms and other resources", in *ISPRS Journal of Photogrammetry & Remote Sensing*, Vol. 54, p. 164-198, 1999.
- [Bannoa 2004] A. Bannoa, T. Masudaa, K. Ikeuchia, "Three dimensional visualization and comparison of impressions on fired bullets", in *Forensic Science International*, Vol. 140, p. 233-240, 2004.
- [Besl et McKay 1982] P. Besl, N. McKay, "A method for registration of 3-D shapes", *IEEE Trans. Pattern Anal. Mach. Intell*, Vol. 14(2), p. 239 - 256, 1982.
- [Beucher et Lantuejoul 1979] F. Meyer and S. Beucher, "Use of Watersheds in Contour Detection", in *Proc. Int. Workshop on Image Processing, CCETT/IRISA, Rennes*, p. 17-21, September 1979.
- [Bitelli et al. 2004] G. Bitelli, M. Dubbini, A. Zanutta, "Terrestrial Laser Scanning and Digital Photogrammetry Techniques To Monitor Landslide Bodies", in *proceedings of the XXth ISPRS Congress (Istanbul)*, Commission V, WG V/2, p. 246, 2004.
- [Breckon et Fischer 2004] T. Breckon, R. Fischer, "Environment Authentication through 3D Structural Analysis", in *proceedings of ICIAR 2004, Porto, Portugal*, Part IIA, Vol. 3212, 2004.
- [Boehler et al. 2003] W. Boehler, M. Bordas Vicent, A. Marbs, "Investigating Laser Scanner Accuracy", *CIPA Symposium*, Turkey, Oct 2003.
- [Borgo et al. 2001] R. Borgo, P. Cignoni, R. Scopigno, "An easy way to use visualization system for huge cultural heritage meshes", *VAST01 Conference*, Greece, 2001.
- [Botsch et al. 2002] M. Botsch, A. Wiratanaya, and L. Kobbelt, "Efficient high quality rendering of point sampled geometry", in *proceedings of the Eurographics Workshop on Rendering 02*, 2002.
-

- 
- [Brick et al. 2004] E-M. Brick, H. Rudolph, J. Arnold, R. G. Luthardt, “Analysis of three-dimensional sinter shrinkage of copings made from alumina in an innovative direct shaping process”, *Computerized Medical Imaging and Graphics*, Vol. 28, p. 159-165, 2004.
- [Chaperon 2002] T. Chaperon, “Segmentation de nuage de points 3D pour la modélisation automatique d’environnements industriels numérisés”, *thèse de l’Ecole des Mines de Paris*, soutenue en 2002.
- [Chi-Keung et al. 2001] T. Chi-Keung, G. Medioni, L. Mi-Suen, “N-dimensional tensor voting and application to epipolar geometry estimation”, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 23, No 8, p. 829-844, Jan 2001.
- [Cignoni et al. 1998] P. Cignoni, C. Rocchini, R. Scopigno, “Metro : measuring error on simplified surfaces”, *Computer Graphics Forum*, vol. 17(2), p. 167-174, 1998.
- [Crosetto 2002] M. Crosetto, “Calibration and validation of SAR interferometry for DEM generation”, *Journal of Photogrammetry & Remote Sensing*, Vol. 57, p. 213-227, 2002.
- [DeBoor 1978] C. De Boor, “A Practical Guide to Splines”, Springer-Verlag (New York), 1978.
- [Duguet et al. 2004] F. Duguet, G. Drettakis, D. Girardeau-Montaut, J-L. Martinez, F. Schmitt, “A Point-Based Approach for Capture, Display and Illustration of Very Complex Archeological Artefacts”, *VAST04 Conference*, Belgium, p. 1-10, 2004.
- [Duguet et Drettakis 2004] F. Duguet and G. Drettakis, “Flexible Point-Based Rendering on Mobile Devices”, *TR-4833*, 2003.
- [Duguet et Girardeau-Montaut 2004] F. Duguet, D. Girardeau-Montaut, “Rendu en portion de ciel visible de gros nuages de points 3D”, in *actes des 17èmes journées de LÁFIG*, Poitiers, 2004.
- [Edahiro et al. 1984] M. Edahiro, I. Kokubo and T. Asano, “A new point-location algorithm and its practical efficiency – comparison with existing algorithms”, *ACM Trans. on Graphics*, vol. 3, p. 86-109, April 1984.
- [FEMA 2002] Federal Emergency Management Agency (FEMA), “A Nation Remembers, A Nation Recovers : Responding to September 11, 2001, One Year Later”, *FEMA Report*, 2002.
- [Frantz et al. 2003] D. Frantz, A. Wiles, S. Leis, S. Kirsch, “Accuracy assessment protocols for electromagnetic tracking systems”, in *Physics in Medicine and Biology*, Vol. 48, No. 14, Jul 2003.
- [Frissen et Perry 2002] S. F. Frissen, R. N. Perry, “Simple and Efficient Traversal Methods for Quadrees and Octrees”, *Technical Report 2002-41*, Mitsubishi Electric Research Laboratories, 2002.
- [Gamba et Houshmand 2000] P. Gamba, B. Houshmand, “Digital Surface Models and Building Extraction : A Comparison of IFSAR and LIDAR Data”, *IEEE Transaction on Geoscience and Remote Sensing*, Vol. 34, No. 4, 2000.
- [Girardeau-Montaut et al. 2005] D. Girardeau-Montaut, M. Roux, R. Marc, G. Thibault, “Change Detection on Points Cloud Data acquired with a Ground Laser Scanner”,
-

- Laser Scanning 2005*, ISPRS Workshop, Enschede, the Netherlands, September 12-14, 2005.
- [Gonçalves et al. 2004] J. G.M. Gonçalves, V. Sequeira, B. Chesnay, C. Creusot, S. Johnson, J. Whichello, “3D Laser Range Scanner for Design Verification”, in *proceedings of the 44th Meeting of the Institute for Nuclear Materials Management*, Phoenix, Arizona, 2003.
- [Gordon et al. 2004] S. Gordon, D. Lichti, M. Stewart, J. Franke, “Modelling Point Clouds for Precise Structural Deformation measurement”, in *Proceedings of the XXth Congress of the International Society for Photogrammetry and Remote Sensing, Istanbul*, Vol. 35, Part B5, 2004.
- [Havran 1999] V. Havran, “A Summary of Octree Ray Traversal Algorithms”, *Ray Tracing News*, Vol 12(2), p. 11-23, 1999.
- [Hirose 1996] H. Hirose “Maximum Likelihood Estimation in the 3-parameter Weibull distribution : a Look Through the Generalized Extreme-value Distribution”, *IEEE Transaction on Dielectrics and Electrical Insulation*, Vol 3(1), p. 43-55, 1996.
- [Ho et al. 2005] H. P. Ho, Y. Chen, H. Liu, P. Shi, “Level set active contours on unstructured point cloud”, *Computer Vision and Pattern Recognition (CVPR) 2005*, Vol 2, p. 655-662, 20-25 June 2005.
- [Hoare 1962] C. A. R. Hoare, “Quicksort”, *Computer Journal*, Vol 5(1), p. 10-15, 1962.
- [Hoppe et al. 1992] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, “Surface reconstruction from unorganized points”, *Computer Graphics*, Vol. 26(2), p. 71-78, 1992.
- [Hsiao et al. 2004] K.H. Hsiao, J.K. Liu, M.F. Yu, Y.H. Tseng, “Change Detection of Landslide Terrains Using Ground-based Lidar Data”, in *proceedings of the XXth ISPRS Congress (Istanbul)*, Commission VII, WG VII/5, p. 617, 2004.
- [Hu et al. 2002] J. Hu, A. Razdan, G. Nielson, G. Farin, D. Page Baluch, D. G. Capco “Volume Segmentation Using Weibull E-SD Fields”, *Transaction on Visualization and Computer Graphics*, Vol. 9, no. 3, p. 320-328, 2002.
- [Huyck et Adams 2002] C.K. Huyck and B.J. Adams, “Emergency Response in the Wake of the World Trade Center Attack : The Remote Sensing Perspective”, *Engineering and Organizational Issues Related to the World Trade Center Terrorist Attack*, Vol. 3, MCEER Special Report Series, Jun 2002.
- [Jutzi et Stilla 2004] B. Jutzi, U. Stilla, “Extraction of Features from objects in Urban Areas Using Space-Time Analysis of Recorded Laser Pulses”, in *Proceedings of the XXth Congress of the International Society for Photogrammetry and Remote Sensing, Istanbul*, Vol. 35, Part B2, p. 1-6, 2004.
- [Kass et al. 1987] M. Kass, A. Witkin, et D. Terzopoulos “Snakes : Active contour models”, in *Proceedings of IEEE International Conference on Computer Vision*, p. 259-2681, 1987.
- [Kern 2001] J. Kern (Optech), “Mapping Ground Zero”, *Professional Surveyor*, Vol. 21, No. 10, Nov 2001.
-

- 
- [Krüger 2005] J. Krüger, J. Schneider, R. Westermann, “DuoDecim - A Structure for Point Scan Compression and Rendering”, in *proceedings of the Symposium on Point-Based Graphics 2005*, 2005.
- [Lichti et Gordon 2004] D. Lichti and Stuart J. Gordon, “Error Propagation in Directly Georeferenced Terrestrial Laser Scanner Point Clouds for Cultural Heritage Recording”, in *proceedings of the FIG Working Week 2004, Athens, Greece*, 2004.
- [Lumia et al. 1983] R. Lumia, L. Shapiro and O. Zuniga, “A new connected components algorithm for virtual memory computers”, *Computer Vision, Graphics and Image Processing*, Vol. 22, p. 328-339, 1983.
- [Maas 1999] H.-G. Maas, “Closed solutions for the determination of parametric building models from invariant moments of airborne laserscanner data”, *ISPRS Conference 'Automatic Extraction of GIS Objects from Digital Imagery'*, München/Germany, 1999.
- [Meagher 1980] D. Meagher, “Octree encoding : A new technique for the representation, manipulation and display of arbitrary three dimensional objects by computer”, *Tech. Rep. IPL, TR-80-111*, Rensselaer Polytechnic institute, Troy, N.Y., 1980.
- [Memoli et Sapiro 2001] F. Memoli and G. Sapiro, “Fast Computation of Weighted Distance Functions and Geodesics on Implicit Hyper-Surfaces”, *Journal of Computational Physics*, Vol. 173(1), p. 764-795, 2001.
- [Memoli et Sapiro 2003] F. Memoli and G. Sapiro, “Distance Functions and Geodesics on Point Clouds”, *Technical Report 1902, IMA, University of Minnesota, USA*, Dec. 2002/April 2003.
- [Memoli et Sapiro 2004] F. Mémoli and G. Sapiro, “Comparing Point Clouds”, *Eurographics Symposium on Geometry Processing*, 2004.
- [Mitra et Nguyen 2003] N. J. Mitra, A. Nguyen, “Estimating Surface Normals in Noisy Point Cloud Data”, in *proceedings of Symposium on Computational Geometry*, Vol. 22, p. 322-328, 2003.
- [Moening et Dodgson 2003] C. Moening et N. A. Dodgson “SFast Marching Farthest Point Sampling”, in *Proc. EUROGRAPHICS*, 2003.
- [Morse 2001] B. Morse, T.S. Yoo, P. Rheingans, D.T. Chen, K.R. Subramanian, “Interpolating Implicit Surfaces From Scattered Surface Data Using Compactly Supported Radial Basis Functions”, in *proceedings of the International Conference on Shape Modeling and Applications (SMI2001)*, p. 89-98, 2001.
- [Murakami et al. 1999] H. Murakami, K. Nakagawa, H. Hasegawa, T. Shibata and E. Iwanami, “Change detection of buildings using an airborne laser scanner”, *ISPRS Journal of Photogrammetry & Remote Sensing*, Vol. 54, p. 148-152, 1999.
- [Musser 1997] D. R. Musser, “Introspective Sorting and Selection Algorithms”, *Software Practice and Experience*, Vol. 27(8), 1997.
- [Neuez 2002] G. Neuez, “Range Camera Imaging : from Human Body to Very Large Points Scenes Visualization”, *PhD Thesis*, Chalmers University of Technology, Sweden, techn. report No 425, Jun 2002.
-

- [Ohtake et al. 2003] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, H-P. Seidel, “Multi-level Partition of Unity Implicits”, *SIGGRAPH*, 2003.
- [Page et al. 2003] D.L. Page, A. Koschan, M.A. Abidi “Perception-based 3D Triangle Mesh Segmentation using Fast Marching Watersheds”, in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, Vol 2, p. 27-32, 2003.
- [Paquerault 1998] S. Paquerault “Restitution du relief à partir d’images radar par radar-clinométrie”, *thèse de l’Ecole Nationale Supérieure des Télécommunications*, soutenue en 1998.
- [Peyré et Cohen 2005] G. Peyré et L. Cohen “Geodesic Computations for Fast and Accurate Surface Remeshing and Parameterization”, *Progress in Nonlinear Differential Equations and Their Applications*, Vol. 63, p. 157-171, 2005.
- [Piegl 2001] L. Piegl, “On NURBS : A Survey”, in *IEEE Computer Graphics and Applications*, Vol. 11, No. 1, p. 55 - 71, Jan 1991.
- [Pouget et Demengel 1998] J-P. Pouget, G. Demengel, “Modèle de Bézier, des B.Splines et des NURBS, mathématiques des courbes et des surfaces. Mathématiques des courbes et des surfaces. Outils pour l’ingénieur. Bases pour la CAO”, Ellipses (Paris), 1998.
- [Preiss et al. 2003] M. Preiss, D. Gray, N. Stacy, “A Change Detection Statistic for Repeat Pass Interferometric SAR”, in *Proceedings of ICASSP 2003 - International Conference on Acoustics, Speech, and Signal Processing, Hong Kong*, p. 241-244, Apr 2003.
- [Preiss et al. 2003] M. Preiss, D. Gray, N. Stacy, “A Change Detection Statistic for Repeat Pass Interferometric SAR”, in *Proceedings of ICASSP 2003 - International Conference on Acoustics, Speech, and Signal Processing, Hong Kong*, p. 241-244, Apr 2003.
- [Rabbani et van den Heuvel 2005] T. Rabbani, F. van den Heuvel “Efficient Hough transform for automatic detection of cylinders in point clouds”, *Laser Scanning 2005*, ISPRS Workshop, Enschede, the Netherlands, September 12-14, 2005.
- [Rosen et Hensley 2000] P. Rosen, S. Hensley, I. Joughin, F. Li, S. Madsen, E. Rodriguez, and R. Goldstein, “Synthetic Aperture Radar Interferometry”, in *Proceedings of the IEEE*, Vol. 88, no. 3, Mar 2000.
- [Rosenfeld et Pfaltz 1966] A. Rosenfeld, J. Pfaltz, “Sequential operations in digital picture processing”, *Journal of the ACM*, Vol. 13, p. 471-494, 1966.
- [Roux et Maître 2001] M. Roux, H. Maître, “Some more steps towards 3D reconstruction of urban areas from multiple views”, in *Automatic Extraction of Man-Made Objects from Aerial and Space Images (III)*, Ascona, Suisse, p. 135-147, Jun 2001.
- [Rusinkiewicz et Levoy 2003] S. Rusinkiewicz, M. Levoy, “QSplat : A Multiresolution Point Rendering System for Large Meshes”, *SIGGRAPH*, 2000.
- [Saff et Kuijlaars 1997] E. B. Saff, A. B.J. Kuijlaars, “Distributing Many Points on a Sphere”, *The Mathematical Intelligencer*, Vol. 19, No. 1, Springer-Verlag (New-York), 1997.
- [Sapiro et Memoli 2002] G. Sapiro, F. Memoli, “Fast computation of distance functions and geodesics on implicit surfaces”, *non publié*, 2002.
-



- 
- [Saporta 1990] G. Saporta, "Probabilités, analyse de données et statistique", éditions Technip, ISBN 2-7108-0565-0, 1990.
- [Schenk 2001] T. Schenk, "Systematic Errors in Airborne Laser Scanners", in *Technical Notes in Photogrammetry*, No 19, Jan 2001.
- [Schuster 2004] H.F. Schuster, "Segmentation of Lidar Data Using The Tensor Voting Framework", in *proceedings of the XXth ISPRS Congress (Istanbul)*, Commission III, 2004.
- [Sethian 1996] J.A. Sethian, "Fast marching level set methods for three-dimensional photolithography development", in *Proceedings of the SPIE 1996 International Symposium on Microlithography, Santa Clara, CA*, 1996.
- [Shih et al. 2004] N-J. Shih, M-C. Wu, J. Kunz, "The Inspections of As-built Construction Records by 3D Point Clouds", *CCIFE Working Paper #090*, Stanford University, 2004.
- [Siegman 1971] A.E. Siegman, "An Introduction to lasers and masers", Mc Graw-Hill, 1971.
- [Steinmann et Bonnard 2002] G. Steinmann, Ch. Bonnard "Mesures en continu effectuées par le LMS sur des instabilités de terrains", *Etude LMS no S5919 - Projet de recherche FRAN2-CTI*, EPFL, juin 2002.
- [Sun et al. 2002] Y. Sun, D. L. Page, J. K. Paik, A. Koschan, and M. A. Abidi, "Triangle meshes-based edge detection and its application to surface segmentation and adaptive surface smoothing", *IEEE Int'l Conf. on Image Processing*, 2002.
- [Thrun et al. 2003] S. Thrun, M. Diel, D. Hähnel, "Scan Alignment and 3-D Surface Modeling with a Helicopter Platform", *The 4th International Conference on Field and Service Robotics*, July 14-16, 2003.
- [Tsitsiklis 1995] J. N. Tsitsiklis, "Efficient algorithms for globally optimal trajectories", *IEEE Trans. Autom. Control*, Vol. 40, p. 15-28, 1995.
- [Tung 2005] T. Tung, "Indexation 3D de base de données d'objets par graphes de Reeb améliorés", *thèse de l'Ecole Nationale Supérieure des Télécommunications*, soutenue en 2005.
- [Ueshiba et Roth 1999] T. Ueshiba, G. Roth, "Generating Smooth Surfaces with Bicubic Splines over Triangular Meshes : Toward Automatic Model Building From Unorganized 3D Points", in *Proceedings of the Second International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM'99)*, Ottawa, Ontario, Canada, p. 302-311, 1999.
- [Ullrich et al. 2002] A. Ullrich, N. Studnicka, J. Riegl, "Long-range high-performance time-of-flight-based 3D imaging sensors", in *Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission*, 2002.
- [Vincent et Soille 1991] L. Vincent and P. Soille, "Watersheds in digital spaces : an efficient algorithm based on immersion simulations", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 13, No 6, p. 583-598, June 1991.
- [Vögtle et Steinle 2004] T. Vögtle, E. Steinle, "Detection And Recognition of Changes in Building Geometry Derived From Multitemporal Laserscanning Data", in *proceedings of the XXth ISPRS Congress (Istanbul)*, Vol. 35, Part B2, p. 428-433, 2004.
-

- 
- [Vosselman et al. 2004] G. Vosselman, B.G.H. Gorte and G. Sithole, "Change Detection for Updating Medium Scale Maps Using Laser Altimetry", in *proceedings of the XXth ISPRS Congress (Istanbul)*, Vol. 35, Part B3, 2004.
- [Vosselman et al. 2004-2] G. Vosselman, B.G.H. Gorte, G. Sithole and T. Rabbani "Recognising structure in laser scanner point clouds", *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 46, part 8/W2, Freiburg, Germany, October 4-6, p. 33-38, 2004.
- [Vosselman et al. 2002] G. Vosselman, H.G. Maas, "Adjustment and Filtering of Raw Laser Altimetry Data", in *proceedings of the OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Detailed Digital Elevation Models*, OEEPE Publication No. 40, 2002.
- [Wagner et al. 2004] W. Wagner, A. Ullrich, T. Melzer, C. Briese, K. Kraus, "From Single-pulse to Full-waveform Airborne Laser Scanners : Potential and Practical Challenges", in *Proceedings of the XXth Congress of the International Society for Photogrammetry and Remote Sensing, Istanbul*, Vol. 35, Part B3, p. 201-206, 2004.
- [Watkins 1970] G. S. Watkins, "A Real-time Visible Surface Algorithm", *Technical Report UTEC-CSc-70-101*, Dep. Comptr. Sci., U. of Utah, Salt Lake City, 1970.
- [Williams et Simons 2000] P. Williams, D. J. Simons, "Detecting Changes in Novel, Complex Three-dimensional Objects", *Visual Cognition*, p. 297-322, 2000.
- [Xu et Prince 1997] C. Xu et J.L. Prince, K. Kraus, "Gradient Vector Flow : A New External Force for Snakes", in *Proc. IEEE Conf. on Comp. Vis. Patt. Recog. (CVPR)*, Los Alamitos, Comp. Soc. Press, p. 66-71, June 1997.
- [Yatziv et al. 2005] L. Yatziv, A. Bartesaghi, and G. Sapiro, "O(n) Implementation of the Fast Marching Algorithm", *IMA Preprint Series No. 2021*, February 2005.
- [Zhao et Shibasaki 2002] H. Zhao, R. Shibasaki, "Surface Modeling of Urban 3D Objects from Vehicle-Borne Laser Range Data", in *Proceedings of Photogrammetric Computer Vision, Graz*, Vol. 34, Part A, p. 412-417, Sep 2002. " Long-range
- [Zwicker et al. 2002] M. Zwicker, M. Pauly, O. Knoll, M. Gross, "Pointshop 3D : an interactive system for point-based surface editing", in *proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002.
-

