THESE

Présentée à :

L'Institut National des Sciences Appliquées de Rouen

En vue de l'obtention du grade de :

Docteur en « Mathématiques appliquées»

Par

Denis BRAZEY

Reconnaissance de formes et suivi de mouvements en 4D temps-réel
~
Restauration de cartes de profondeur

Soutenue le 09 décembre 2014

Devant le jury composé de :

Directeurs de thèse	
M. Christian GOUT	Professeur des Universités (INSA Rouen)
M. Bruno PORTIER	Professeur des Universités (INSA Rouen)
Rapporteurs	
M. Gilles CELEUX	Directeur de recherche (INRIA Saclay)
M. Olivier GIBARU	Professeur des Universités (Arts et Métiers ParisTech, Lille)
Examinateurs	
Mme. Marie-Laurence MAZURE	Professeur des Universités (Université Joseph Fourier, Grenoble)
M. Frédéric RICHARD	Professeur des Universités (Université Aix-Marseille)
Mme. Natalie FORTIER	Maître de conférences (INSA Rouen)
M. Jérôme BROSSAIS	Responsable développements logiciels (Prynɛl, Corpeau)

PRYNƐL

Le vin est le breuvage le plus sain et le plus hygiénique qui soit.

Louis PASTEUR (1822-1895)

Résumé / Abstract

Résumé

Durant la dernière décennie, le nombre de systèmes de vidéo-surveillance a fortement augmenté. Cette forte croissance s'accompagne d'un besoin d'automatisation de plus en plus important. Dans le même temps, les systèmes d'acquisition 3D se sont démocratisés, offrant ainsi des perspectives d'applications intéressantes pour les problématiques liées à la vidéo-surveillance.

Dans le cadre de cette thèse, nous nous intéressons à plusieurs problématiques liées au traitement de données 3D. La première concerne la détection et le suivi de personnes dans des séquences d'images de profondeur. Nous proposons une amélioration d'une méthode existante basée sur une étape de segmentation, puis de suivi des personnes. Notre apport concerne principalement la détection de personnes partiellement masquées par les bords de l'image. La deuxième problématique abordée est la détection et la modélisation de têtes dans un nuage de points 3D. Pour cela, nous adoptons une approche probabiliste basée sur un nouveau modèle de mélange sphérique. La forme des objets est considérée de part la distribution statistique des points dans l'espace. Le modèle est basé sur une nouvelle densité de probabilité décrivant des points répartis autour d'une surface sphérique. Le nuage de points 3D est donc modélisé par un ensemble de sphères dont le nombre est variable. Cette partie constitue la contribution majeure de la thèse. Les têtes sont ensuite détectées par un ensemble de règles heuristiques simples. La méthode a été adaptée pour fonctionner en quasi temps-réel. Une généralisation au modèle de mélange ellipsoïdal a été proposée. La dernière application traitée est liée à la restauration d'images de profondeur présentant des données manquantes. Nous proposons pour cela d'utiliser une méthode d'approximation de surface par D^m-splines d'interpolation pour approcher et restaurer les données. Des changements d'échelles sont appliqués pour atténuer les oscillations parasites dues aux fortes variations des données. L'avantage de cette approche est qu'elle permet de rééchantilloner le nuage de points 3D sur une grille de précision choisie.

Les résultats présentés illustrent l'efficacité des algorithmes développés. Le travail réalisé a été en partie industrialisé dans le logiciel propriétaire de la société partenaire. Ces travaux offrent de nombreuses perspectives de recherche intéressantes.

Mots-clés : capteurs 3D, détection de têtes, modèle de mélange sphérique, distribution elliptique, algorithme EM, restauration, D^m-spline, changements d'échelles.

Abstract

In the last decade, the number of CCTV systems strongly increased. This significant overgrowth comes with a more and more important need of automation. In the same time, 3D sensors offer interesting perspectives in the domain of CCTV image processing.

In this dissertation, we are interested in several issues related to 3D data processing. The first one concerns people detection and tracking in depth map sequences. We propose an improvement of an existing method based on a segmentation stage followed by a tracking module. Our contribution is the detection of persons partially hidden by the image borders. The second issue is head detection and modelling in 3D point clouds. In order to do this, we adopt a probabilistic approach based on a new spherical mixture model. The shape of objects is considered through the statistical distribution of points in 3D space. The mixture model is based on a new probability density function modelling points spread around a spherical surface. The point cloud is therefore modelled by a variable number of spheres. This part constitutes the main contribution of this work. Heads are then detected thanks to simple heuristic rules. The method has been adapted to reach a near real-time performance. A generalization to ellipsoïdal surfaces has been explored. The last considered application deals with the restoration of deteriorated depth maps containing missing data. To solve this problem, we propose to use a surface approximation method based on interpolation D^m-splines to approximate and restore the image. Scale transforms are applied to avoid undesirable oscillations due to the large variations in the data. The advantage of this approach is that the 3D point cloud can be resampled on a finer grid.

Presented results illustrate the efficency of the developped algorithms. Furthermore, this work has been partially included in the proprietary software of the partner company. This work offers some interesting research perspectives.

Keywords : 3D sensor, head detection, spherical mixture model, elliptical distribution, EM algorithm, inpainting, D^m-spline, scale transformation.

Remerciements

Tout d'abord, je souhaiterais remercier Christian Gout et Bruno Portier, mes deux directeurs de thèse, pour avoir dirigé mes recherches pendant ces trois années. L'aboutissement d'une thèse repose autant sur le travail du doctorant que sur celui des directeurs de thèse. Je remercie Christian Gout pour m'avoir si bien accueilli dans son laboratoire, pour ses sages conseils, sa disponibilité, sa patience et sa détermination au moins aussi grande que sa gentillesse. Je tiens à remercier Bruno Portier pour son soutien sans failles, sa grande compétence, sa rigueur, son enthousiasme, sa gentillesse et sa curiosité scientifique. J'espère lui avoir donné l'envie de retenter l'aventure de l'encadrement d'une thèse.

Je remercie également Gilles Celeux et Olivier Gibaru de m'avoir fait l'honneur d'évaluer ce travail ainsi que Marie-Laurence Mazure et Frédéric Richard pour avoir accepté de participer au jury. Leurs expertises ont débouché sur des remarques très constructives.

Le sujet de thèse n'aurait pu exister sans les sociétés Prynɛl et TEB. Je tiens donc à remercier Stéphane Bidault pour m'avoir permis d'effectuer ce travail au sein de sa société. Je souhaite remercier tout particulièrement l'équipe Prynɛl informatique pour tous ces bons moments passés : Jérôme Brossais, pour m'avoir donné sa confiance depuis plusieurs années, Thomas Chalumeau, expert reconnu et sportif aguerri, pour son aide, Laurent, Antony, les deux Olivier, Jessica, Emilien et Michaël. Je remercie aussi l'équipe "électronique" : Vincent, Clément, David et Éric, pour leur soutien quotidien, et enfin l'artiste Ludovic pour son inspiration sans limites.

Merci à tous les membres du LMI pour leur accueil chaleureux et pour leur gentillesse. Je pense notamment à Carole, Natalie, Anastasia, Vladimir, Nicolas, Omar, Brigitte, Florentina, Zacharie, Solène, Théo et Wilfredo.

Tout ceci n'aurait pas été possible sans le soutien de mes parents et de ma famille. Une pensée également pour Alvin dont la thèse se terminera bientôt. Enfin, un grand merci à Thomas L. pour tous ses encouragements.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Les systèmes de vidéo-surveillance sont de plus de plus présents dans les lieux publics, les locaux d'entreprises et les habitations privées. Ce développement très rapide s'explique par un besoin croissant de protection des biens, des lieux et des personnes. On estime qu'à ce jour, plus d'une dizaine de millions de caméras de vidéo-surveillance sont installées dans le monde. Ce secteur d'activité est vaste et couvre plusieurs marchés de taille importante.

Les premiers systèmes de vidéo-surveillance étaient simplement constitués de moniteurs de visualisation reliés à des caméras. Les systèmes actuels sont beaucoup plus sophistiqués et proposent des fonctionnalités avancées. L'exploitation de la grande quantité d'informations disponibles est difficile et nécessite une intervention humaine. Le traitement de cet ensemble considérable de données par des opérateurs est délicat puisqu'il nécessiterait beaucoup de ressources. Par conséquent, l'automatisation de certaines tâches est devenue essentielle et indispensable. "L'intelligence" du système est un enjeu central puisqu'elle constitue une plus value importante.

La société TEB®, installée à Corpeau en Bourgogne, est spécialisée depuis 35 ans dans la conception et la fabrication de systèmes de vidéo-surveillance. Elle apporte régulièrement des innovations importantes aussi bien matérielles que logicielles. La société Prynɛl, bureau d'études de TEB®, conçoit toute une gamme de produits destinés à être distribués. Elle développe entre autres des algorithmes de traitement d'images intégrés à son produit phare, l'enregistreur numérique multifonctions Digipryn®. Le but de ces algorithmes est d'exploiter et d'analyser les flux vidéos pour en tirer des informations utiles.

Les travaux présentés dans ce manuscrit s'intéressent aux possibilités d'utilisation d'une nouvelle génération de caméras, les capteurs 3D, pour détecter des personnes dans une pièce. En effet, soucieuse de rester en pointe, la société TEB® souhaite tirer profit de ces nouveaux capteurs. Les capteurs 3D se sont démocratisés avec l'apparition de la console de jeux Xbox de Microsoft contrôlées directement par les mouvements de l'utilisateur. Ils fournissent une information sur la forme de la surface des objets en plus de l'information classique de couleur. Ces caméras donnent une meilleure connaissance de l'environnement observé. Elles offrent donc des possibilités de conception d'algorithmes plus efficaces par la prise en compte de l'information de distance disponible avec ces nouveaux capteurs. Cependant, les techniques de mesure existantes ne permettent pas toujours la bonne estimation de la distance. Ce constat nous incite donc à nous intéresser au problème de restauration de cartes de profondeur. Étant donnée une image comportant des zones de données manquantes, le but est de restaurer les pixels ne contenant aucune information. Le succès de ce pré-traitement des images assure le bon fonctionnement des algorithmes appliqués. Le problème de la détection de personnes se pose par exemple dans les applications de comptage de personnes, de détection d'intrusions, d'accès sécurisé à une zone, d'évacuation d'urgence de bâtiments et d'analyse marketing. Il constitue un sujet de recherche toujours actif dans la communauté de la vision par ordinateur.

Les travaux présentés dans ce manuscrit sont le résultat d'une collaboration entre la société Prynɛl et le laboratoire LMI (Laboratoire de Mathématiques de l'INSA de Rouen) dans le cadre d'une thèse CIFRE. L'objectif de la thèse est de proposer des algorithmes de détection de personnes dans des images provenant de capteurs 3D et d'explorer le problème de restauration d'images de profondeur.

Ce manuscrit est organisé en trois parties. La **première** partie traite du problème de détection et de suivi de personnes dans des séquences d'images de profondeur. Elle débute par une présentation des principales technologies d'acquisition 3D et de quelques algorithmes utilisés dans la thèse (Chapitre 1). Nous proposons ensuite une amélioration d'une méthode existante pour détecter des personnes partiellement masquées par les bords de l'image ou par leur trop grande proximité avec le capteur (Chapitre 2). La **deuxième** partie concerne la mise au point d'un nouvel algorithme de détection de têtes dans un nuage de points 3D représentant des personnes. Nous adoptons pour cela une approche probabiliste et statistique nous conduisant à proposer un nouveau modèle de mélange sphérique et une méthode d'estimation de ses paramètres (Chapitre 3). Des règles de détections heuristiques simples sont ensuite appliquées pour détecter les têtes (Chapitre 4). Diverses optimisations seront également mises en œuvre pour réduire le temps de calcul. Nous explorerons enfin la généralisation du modèle proposé au cas ellipsoïdal (Chapitre 5). La **troisième** et dernière partie se concentre sur la restauration d'images de profondeur présentant des données manquantes. Après avoir effectué un certain nombre de rappels (Chapitre 6), nous proposons d'appliquer une méthode d'approximation de surface par D^m-splines d'interpolation faisant intervenir des changements d'échelle (Chapitre 7). Ce manuscrit sera conclu par une **synthèse** des travaux effectués durant la thèse suivie de **perspectives** de recherches futures.

Première partie

Détection et suivi de personnes à l'aide de capteurs 3D

Les algorithmes de traitement d'images intégrés aux logiciels de vidéo-surveillance constituent une aide à la décision importante. Le système stocke et analyse les images représentant la scène observée. Les nouvelles générations de capteurs 3D, capables de capturer une représentation tridimensionnelle en temps-réel, offrent des possibilités nouvelles. La compréhension du fonctionnement des différentes technologies d'acquisition 3D est fondamentale pour mettre en place des algorithmes efficaces. La détection et le suivi des personnes dans les images est un problème central dans le cadre de la vidéo-surveillance. Le but de cette première partie est donc double : d'une part comprendre le fonctionnement les capteurs 3D et introduire un certain nombre d'algorithmes classiques et d'autre part de proposer une méthode de détection et de suivi de personnes dans une séquence d'images de profondeur.

Le problème de la détection et du suivi de personnes à partir de capteurs 3D a été beaucoup étudiée. Dans le système considéré, le capteur est positionné verticalement à une hauteur faible, ce qui implique que les têtes des personnes ne sont pas toujours visibles. De plus, on souhaite détecter tous les individus, y compris ceux partiellement masqués par les bords de l'image. Par conséquent, nous choisissons de nous baser sur des travaux existants ([86]) dont nous proposons plusieurs améliorations pour l'adapter à nos contraintes particulières.

Cette partie est composée de deux chapitres. Le premier est consacré à la présentation des capteurs 3D et à l'introduction d'algorithmes classiques (Chapitre 1). Le second détaille l'algorithme proposé et expose les résultats obtenus sur des séquences d'images de profondeur (Chapitre 2).

Chapitre 1

Utilisation de capteurs 3D en vidéo-surveillance et traitement de données 2.5D et 3D

Sommaire

1.1 Introduction

Ce premier chapitre est consacré à la présentation des principales technologies d'acquisition 3D et des algorithmes de traitement d'images souvent utilisés en vidéo-surveillance durant les premières étapes de la chaine de traitement. La plupart des algorithmes décrits dans ce chapitre seront appliqués en situation réelle lors d'expérimentations dans d'autres chapitres de la thèse.

Nous commencerons par présenter différentes technologies permettant l'acquisition de données 3D. Nous nous intéresserons aux capteurs 3D récents, plus spécifiquement à ceux dont nous nous servirons dans cette thèse. La connaissance des caractéristiques techniques et du comportement des capteurs 3D dans des situations spécifiques permettra d'améliorer les traitements. Les capteurs 3D sont en pleine évolution et de nouveaux modèles sont commercialisés chaque année sur le marché. Il sont principalement utilisés (en ce qui concerne le grand public) dans le domaine du jeux vidéo et de l'interaction homme-machine. Des évolutions sont régulièrement apportées à ces produits, comme par exemple des résolutions de plus en plus importantes ou des fréquences d'acquisition plus rapides.

Les systèmes de vidéo-surveillance sont présents dans des lieux très divers comme des parkings, des routes, des bureaux, des magasins ou encore des véhicules. Le choix des caméras installées dépend de leurs caractéristiques techniques (angle de vue, qualité d'image, transmission des images), de leurs emplacements et de l'application. Les capteurs 3D constituent un nouveau type de caméra à partir desquelles des traitements d'images sophistiqués peuvent être réalisés. Dans cette thèse, nous utilisons ces nouveaux capteurs pour répondre à des problématiques spécifiques au domaine de la vidéo-surveillance.

Les algorithmes de traitement d'images ont longtemps été appliqués à des images couleur. Chaque pixel contient une information colorimétrique quantifiant la couleur des objets. La compréhension du contenu de l'image était alors basée sur la notion d'apparence. L'analyse de la forme des objets n'était possible que par le biais de leur projection 2D sur le plan de l'image. Les technologies d'acquisition 3D ont apporté une information supplémentaire de distance pour chacun des pixels. Selon la précision de la mesure, cette nouvelle information peut être utilisée pour dissocier des objets proches dans l'image mais éloignés dans l'espace ou pour étudier plus finement la forme des objets. Les capteurs actuels permettent, sous certaines conditions d'utilisation, d'obtenir la forme tridimensionnelle de la surface des objets. L'analyse de cette information 3D est justement l'objet des travaux de cette thèse.

Une chaine de traitement d'image comporte différents niveaux successifs. Les premiers algorithmes appliqués ont généralement pour rôle d'améliorer la qualité des images avant de réaliser des opérations plus complexes. Nous nous intéressons dans ce chapitre à quelques méthodes couramment employées en vidéo-surveillance et adaptées aux données 3D. Ces méthodes ont pour but principal de filtrer les données, de les segmenter et d'en effectuer une première modélisation. Une partie des algorithmes exposés dans ce chapitre seront mis en œuvre dans les différentes applications pratiques de la thèse.

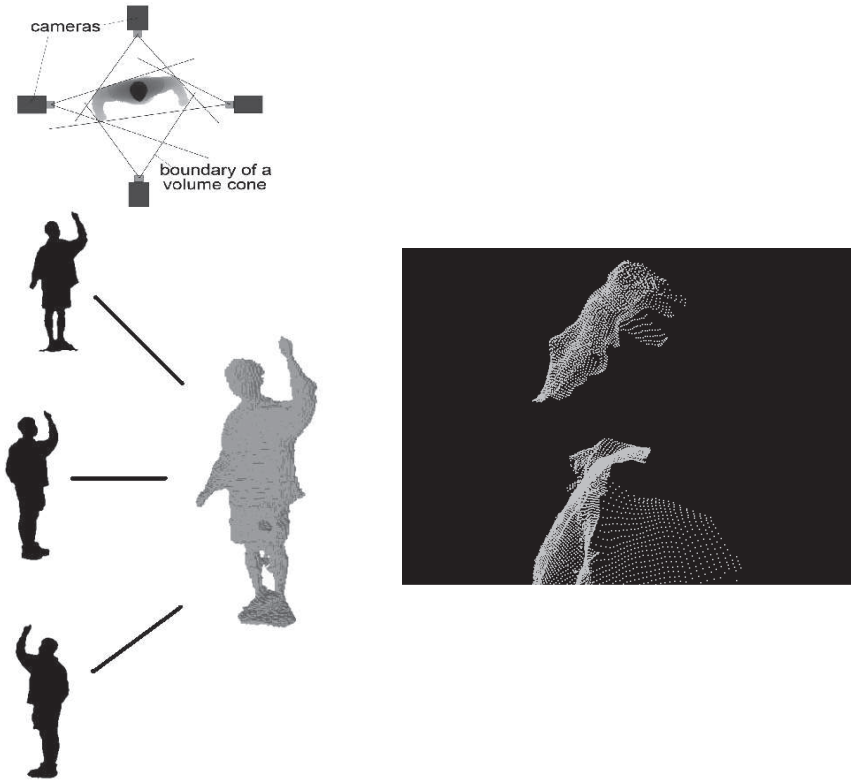
Ce chapitre est organisé de la manière suivante. Nous commencerons par lister et comparer les différents systèmes d'acquisition 3D récents en section 1.2. Nous présenterons ensuite des méthodes de filtrage, de segmentation et de modélisation applicables aux données provenant des capteurs 3D en section 1.3. La conclusion du chapitre sera établie en section 1.4.

1.2 Acquisition de données tridimensionnelles

Les données 3D peuvent se présenter sous différentes formes, chacune apportant une information de nature différente. Le type de donnée manipulé guide la construction et l'implémentation des algorithmes de traitement.

Une première catégorie de systèmes permet de capturer une information **volumétrique**. L'espace est divisé en volumes élémentaires, généralement cubiques, appelés voxels. La représentation volumétrique consiste à connaître l'appartenance ou non de chaque voxel à l'intérieur d'un objet. L'ensemble des volumes élémentaires est alors divisé en deux classes : les éléments *vides* et les éléments appartenant à un *objet*. Généralement, l'information 3D est mesurée par un système multi-caméras, pour observer la scène sous plusieurs angles (voir par exemple [118, 52, 201, 29, 117]). Un exemple de représentation volumétrique est donné Figure 1.2.1(a). La taille des voxels définit la précision de la représentation et par conséquent la quantité de mémoire nécessaire à son stockage.

Dans ces travaux, nous nous intéressons à un autre type de données tridimensionnelles. Les données que nous manipulerons dans cette thèse se présentent sous la forme d'un **nuage de points** échantillonnés sur la surface des objets. La plupart du temps, les points sont calculés à partir d'une **image de profondeur** (données 2.5D) contenant, pour chaque pixel, la distance de l'objet au capteur. L'information volumétrique n'est donc pas accessible directement comme cela était le cas avec la première représentation. Un exemple de nuage de points 3D est représenté Figure 1.2.1(b).



(a) (b)

FIGURE 1.2.1 – En (a), une reconstruction volumétrique multi-vues et en (b), un nuage de points 3D.

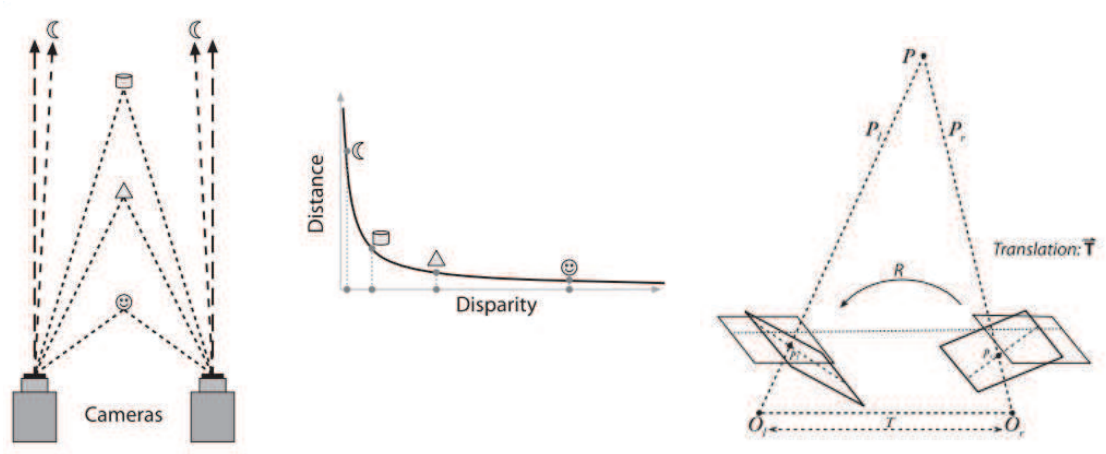
Il existe plusieurs types de systèmes permettant l'acquisition de cartes de profondeur ou de nuages de points 3D. Dans cette section, nous nous concentrons sur trois technologies : les systèmes stéréoscopiques en section 1.2.1, les capteurs à temps de vol en section 1.2.2 et les capteurs à lumière infrarouge structurée en section 1.2.3. Chaque système d'acquisition mesure la distance par un procédé qui lui est propre et présente des avantages et des inconvénients par rapport aux autres. L'ensemble des capteurs présentés ont été utilisés dans de nombreux cadres d'application, ce qui justifie l'intérêt que nous leur portons.

1.2.1 Systèmes stéréoscopiques

Les systèmes stéréoscopiques sont, comme le système visuel humain, composés de deux capteurs fournissant une information colorimétrique. L'information de distance est estimée à partir de ces deux images par un algorithme spécifique. Il existe différentes méthodes d'estimation de cartes de profondeur à partir de données stéréoscopiques (voir [186, 36, 119]) mais nous ne décrirons ici que le principe de fonctionnement général. Les systèmes basés sur la stéréovision sont dits passifs puisque la distance est estimée sans influer sur l'environnement.

L'estimation de la profondeur à partir de deux images couleurs (gauche et droite) notées I_g et I_d est basée sur le décalage en pixels entre les projections du même objet sur les deux images. Ce décalage, appelé disparité, est inversement proportionnel à la distance de l'objet au système. Les projections (pixels) d'un point P de l'espace sur les images gauches et droites sont respectivement notés $p_g = (i_g, j_g)$ et $p_d = (i_d, j_d)$. Un objet éloigné du système de mesure aura tendance à avoir des projections p_g et p_d proches dans les deux images tandis qu'un objet proche aura des projections éloignées (distance euclidienne en pixels).

Si les caractéristiques du système sont connues (distances focales, centre optique et positions relatives des caméras), alors la distance réelle peut être calculée à partir de la disparité mesurée entre les images. Ce lien entre la disparité et la distance réelle est illustré par le schéma de la Figure 1.2.2-(a).



(a) (b)

FIGURE 1.2.2 – En (a), relation entre distance et disparité et en (b), processus de rectification.

L'estimation de la disparité nécessite donc de connaître les correspondances entre les pixels communs aux deux images. Cette information est obtenue par un algorithme de mise en correspondance des pixels. Étant donné un pixel (i_g, j_g) de l'image I_g, ces algorithmes ont pour but de déterminer (s'il existe) le pixel (i_d, j_d) de la seconde image I_d représentant le même point P de la scène.

Pour faciliter cette recherche, les images manipulées sont préalablement rectifiées (transformées) afin d'être correctement alignées l'une par rapport à l'autre. Après rectification, les lignes des images se correspondent et la recherche des correspondances entre les pixels peut être réduite à une recherche par lignes. La rectification des images simplifie et accélère le problème de mise en correspondance. Le schéma de la Figure 1.2.2-(b) illustre le processus de rectification.

Les paramètres intrinsèques (liés à la projection de l'espace de la scène sur l'image du capteur) et extrinsèques (position relative des deux capteurs) sont estimés lors d'une phase de calibration du système. Une mire de calibration constituée de motifs prédéfinis est placée devant les caméras. La détection de points particuliers sur les deux images permet de créer des correspondances précises utilisées pour estimer les différents paramètres. Enfin, une correction de la distorsion est effectuée pour préserver les caractéristiques des motifs avant de réaliser l'estimation des paramètres. La mire de calibration utilisée par la librairie OpenCV ([145]) est représentée Figure 1.2.3. Les coins séparant les cases du damier permettent de générer des correspondances précises.

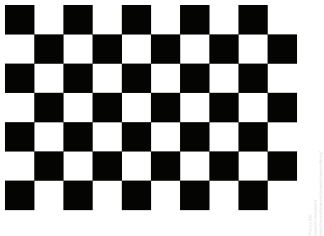


FIGURE 1.2.3 – Mire de calibration utilisée par la librairie OpenCV.

La distance de fonctionnement d'un système stéréoscopique varie selon l'écartement des deux capteurs. Ces systèmes sont très sensibles aux variations de luminosité et les cartes de profondeur produites peuvent contenir, en pratique, des artefacts. Le temps de calcul de la carte de profondeur est plus élevé que pour les autres types de capteurs. Une calibration manuelle via la mire est nécessaire dès que la position et l'orientation relative des caméras sont modifiés. Des implémentations des algorithmes d'estimation de la carte de profondeur sont disponibles par exemple dans la librairie OpenCV. Une image de profondeur obtenue par un système stéréoscopique à partir d'images de test classiques est représentée Figure 1.2.4.

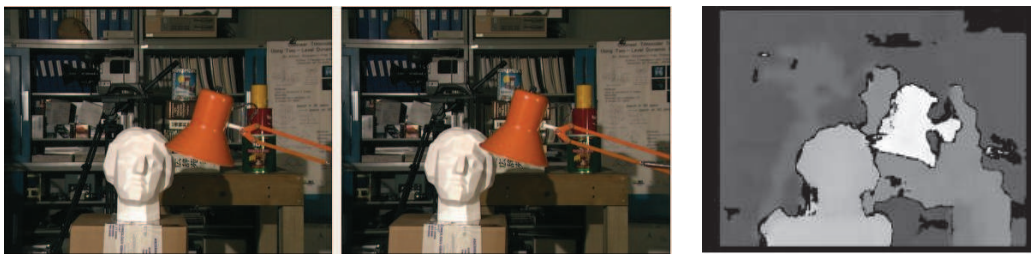


FIGURE 1.2.4 – Images couleurs et carte de profondeur obtenue avec les outils proposés par OpenCV.

Notons enfin que ce procédé est adaptable dans le cas de N capteurs (voir [175] pour une comparaison de plusieurs algorithmes). Dans ce cas, l'intégralité de la scène peut être reconstituée. On parle alors de systèmes stéréoscopiques multi-vues.

1.2.2 Capteurs à temps de vol

Les capteurs à temps de vol appartiennent à la famille des capteurs dits actifs. La mesure de la profondeur est basée sur le temps mis par une impulsion lumineuse à parcourir le trajet du capteur vers la surface de l'objet. Cette catégorie de capteur effectue la mesure de la distance simultanément sur tous les pixels de l'image, contrairement à d'autres capteurs (lasers) procédant par balayage. Le procédé de mesure ne fonctionne pas en extérieur à cause de la lumière naturelle du soleil et sur certaines surfaces comme par exemple le verre. Les capteurs à temps de vol sont composés au minimum d'un emmetteur et d'un récepteur lumineux. Certains sont également équipés d'une caméra couleur classique.

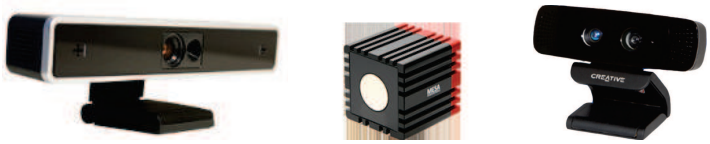
La mesure de la distance est basée sur un principe simple. Une source lumineuse emmet des pulses de lumière sur la scène à intervalles de temps constants. La fréquence d'envoi des impulsions est très rapide. Généralement, le signal appartient au domaine du proche infrarouge et est donc invisible à l'œil nu. La distance à l'objet est directement proportionnelle au temps nécessaire au signal pour effectuer un aller retour. Connaissant la vitesse de la lumière c et le temps Δt mis par le signal lumineux pour effectuer le trajet aller-retour de la source vers l'objet, la distance d séparant le capteur de la surface est donnée par

$$d = \frac{c\,\Delta t}{2}. \tag{1.2.1}$$

La vitesse de la lumière c étant proche de 300000 km/s, l'intervalle de temps entre l'émission et la réception du signal est très court. La complexité du système est matérielle et non pas calculatoire.

La distance de fonctionnement varie selon le signal émis. Certains capteurs peuvent voir de près (de quelques centimètres à un mètre) et d'autres beaucoup plus loin (de quelques mètres à plusieurs dizaines de mètres). L'avantage de ces capteurs est que l'estimation de la distance ne nécessite pas d'algorithmes de calcul lourds comme cela est le cas pour la stéréovision. Les images sont acquises à une fréquence élevée (jusqu'à 50 images par secondes) et en une seule fois. En revanche, la mesure est sensible aux autres sources lumineuses comme le soleil. La résolution des capteurs à temps de vol est généralement plus faible que celle des autres capteurs à cause de la complexité matérielle.

Parmi les systèmes proposés dans le commerce, on trouve notamment les SwissRanger développés par Mesa Imaging ([106]) et le DepthSense DS311 développé par SoftKinetic ([181]). Ces capteurs sont représentés Figure 1.2.5. Le Creative Interactive Gesture Camera Toolkit ([59]) est composé d'un capteur à temps de vol et du SDK Intel Perceptual Computing développés par Creative et Intel. Il sont spécialement conçus pour une vision proche (d'une dizaine de centimètres à un mètre) dans le but de reconnaître les positions de la main. Un exemple d'image acquise est donnée Figure 1.2.6.



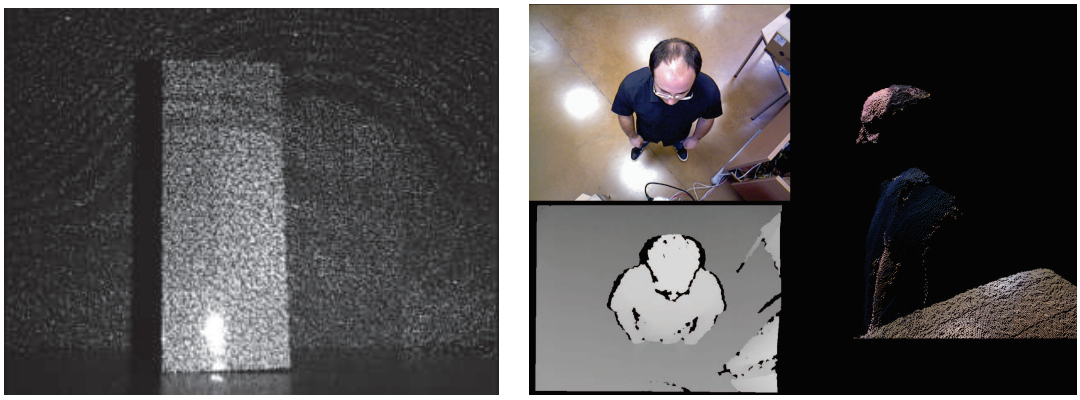
(a) (b) (c)

FIGURE 1.2.5 – Capteurs DepthSense DS311 (a), SwissRanger 4000 (b) et Creative (c).

FIGURE 1.2.6 – Image de profondeur de la documentation Intel Perceptual Computing SDK.

1.2.3 Capteurs à lumière structurée

La dernière technologie que nous présentons est celle que nous avons majoritairement utilisé lors de nos expérimentations. Les capteurs à lumière structurée, appartenant à la famille des capteurs actifs, mesurent la distance en analysant la déformation d'un motif projeté sur la scène. Ce motif peut être visible à l'œil nu ou appartenir au domaine du proche infrarouge. Les capteurs 3D récents à lumière structurée sont composés d'un projecteur infrarouge, d'un récepteur infrarouge et éventuellement d'un capteur couleur classique. Le projecteur projette un motif sur la scène et le récepteur fournit une image contenant le motif observé depuis sa position. Selon les déformations constatées, la distance peut être estimée rapidement. Dans le cas où un capteur couleur est également disponible, une calibration (similaire à celle des systèmes stéréoscopiques) permet de connaitre la correspondance entre les pixels des images couleur et de profondeur. Des exemples de motifs infrarouges et d'images acquises par un capteur à lumière structurée sont donnés Figure 1.2.7.



(a) (b)

FIGURE 1.2.7 – En (a) le motif infrarouge projeté par le capteur Kinect ([112]) et en (b) un exemple de données (image couleur, image de profondeur et nuage de points 3D) obtenues par un capteur à lumière structurée.

L'avantage de ce type de capteur est que la distance peut être estimée rapidement. Les résolutions sont généralement plus élevées que celles des capteurs à temps de vol. Le coût de ces capteurs est faible et leur développement en pleine expansion puisque de nouveaux modèles sont fréquemment proposés. En revanche, cette technologie fonctionne mal en extérieur et sur certaines surfaces (sur les vitres ou sur des surfaces spéculaires par exemple).

Les deux capteurs à lumière structurée les plus connus sont la Kinect de Microsoft ([60]) et le XTion d'Asus ([58]) représentés Figure 1.2.8-(a)(b). Les caractéristiques de ces capteurs sont similaires : une résolution maximale de 640 $\times$ 480 pixels à une fréquence de 30 images par secondes. La distance de fonctionnement est comprise entre 50 centimètres et 5 mètres. La Kinect, développée par PrimSense depuis fin 2010, est intégrée à la plateforme de jeu XBox ([60]) pour permettre d'interagir avec la console avec les mouvements du corps. Le capteur a été beaucoup utilisé dans la communauté de la vision par ordinateur. La qualité des données 3D a été analysée dans [112, 75, 122]. Le SDK OpenNI propose des fonctionnalités d'analyse d'images pour détecter les personnes et modéliser leurs postures. Les résultats obtenus avec ce SDK sont représentés Figure 1.2.8-(c). Les algorithmes proposés dans cette librairie (voir [177, 178]) sont efficaces et reposent sur un algorithme d'apprentissage intensif. Toutefois, ils ne fonctionnent qu'en vue de face.

(a) (b) (c)

FIGURE 1.2.8 – Capteurs Kinect en (a) et Asus XTion en (b). En (c), les deux utilisateurs ont été détectés et modélisés dans l'image de profondeur.

Enfin, le capteur 3D ENSENSO N20 de chez IDS ([103]) représenté Figure 1.2.9 combine les techniques de stéréovision et de lumière structurée pour profiter des avantages des deux technologies et ainsi améliorer la précision des données. La profondeur de champ s'en trouve par conséquent réduite (un mètre maximum selon les modèles). Le capteur est interfacable en USB 3.0 ou en GigaEthernet.

FIGURE 1.2.9 – Capteur ENSENSO N20 de chez IDS.

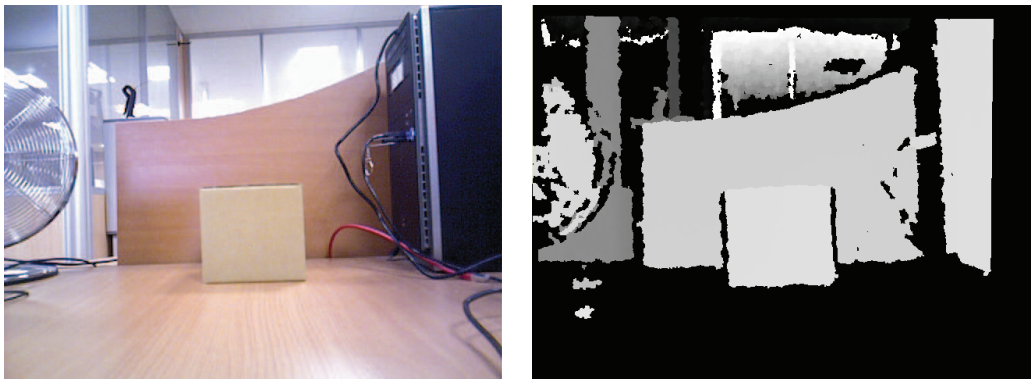
Pixels indéterminés.

Les systèmes d'acquisition présentés ne sont pas toujours capables de mesurer la distance pour chacun des pixels de l'image. En pratique, une partie des pixels ne contiennent aucune information de distance. Les pixels concernés sont dits indéterminés et sont, par convention, affectés d'une valeur nulle. Pour comprendre ce phénomène, il faut connaître le processus de mesure de la distance par le capteur.

Les systèmes stéréoscopiques déterminent la distance à partir de plusieurs images couleur. L'efficacité de la méthode de mise en correspondance des images dépend beaucoup de la texture des objets, de leur éloignement des capteurs et de l'environnement d'acquisition.

Les capteurs actifs (à temps de vol et à lumière structurée) émettent un signal dans la scène dont le retour est ensuite analysé pour calculer la carte de profondeur. Certaines surfaces transparentes ou absorbantes ne sont donc pas mesurables puisqu'elles ne restituent pas correctement le signal. La lumière naturelle, de même nature que le signal émis, perturbe la mesure. On remarque également que les bords de l'image de profondeur sont parfois indéterminés. Ceci est dû au recalage (transformation) entre les images couleur et de profondeur pour que les pixels des deux images se correspondent. Une surface quasi parallèle avec la ligne de visée du capteur constitue un cas limite pouvant poser des problèmes de mesure. Enfin, un objet situé en dehors de l'intervalle de distances mesurables sera indéterminé (distances minimales et maximales de fonctionnement). Une image de profondeur contenant des pixels indéterminés est représentée Figure 1.2.10.

La présence des pixels indéterminés est à prendre en compte dans les algorithmes. Ils doivent être traités avec une attention particulière. Des techniques de restauration adaptées ont été développées pour pour estimer les valeurs manquantes. Ce point particulier sera détaillé dans la suite de la thèse.



(a) (b)

FIGURE 1.2.10 – En (a) une image couleur, et en (b) l'image de profondeur associée contenant des pixels indéterminés (en noir).

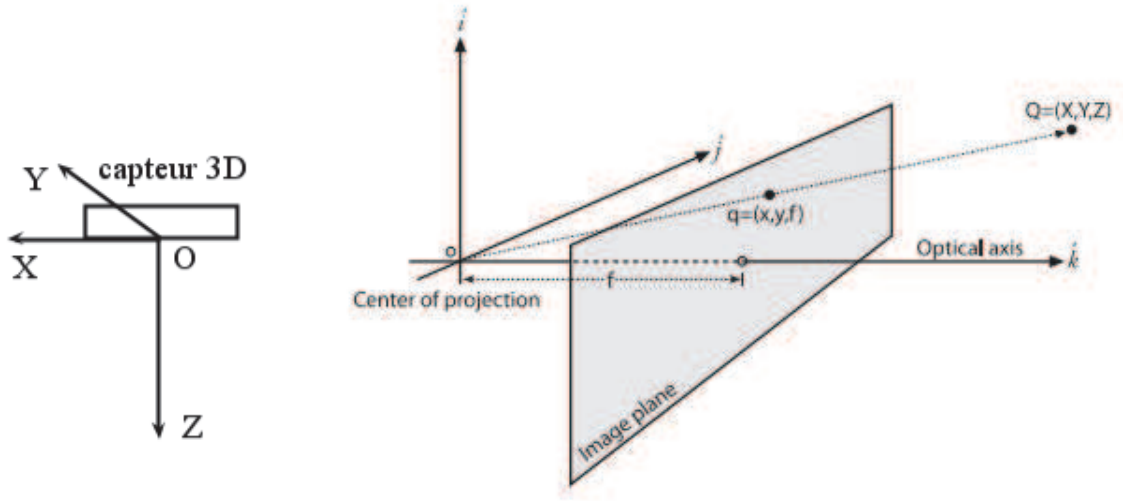
1.3 Traitement de données tridimensionnelles

Les données acquises par le capteur 3D se présentent sous la forme d'une image de profondeur ou d'un nuage de points 3D. Dans tous les cas les données contiennent un bruit de mesure dépendant du capteur utilisé et des propriétés des objets de la scène. Nous commençons par donner en section 1.3.1 les relations permettant de convertir une image de profondeur en un nuage de points 3D. Nous aborderons ensuite en section 1.3.2 quelques techniques de filtrage ayant pour but d'atténuer le bruit de mesure. Nous exposerons enfin des méthodes de segmentation en section 1.3.3, de croissance de régions en section 1.3.4, d'extraction de primitives en section 1.3.5 et de changement de repère en section 1.3.6.

1.3.1 Calcul du nuage de points 3D

Nous allons détailler les relations permettant de convertir une image de profondeur en un nuage de points 3D. Généralement, les outils fournis avec les capteurs disposent de fonctions retournant les coordonnées des points dans le repère du capteur. Lorsque le calcul des points est effectué par la caméra, le nombre d'images par seconde diminue. Nous préférons par conséquent effectuer par nous même ce calcul.

Une image I de dimensions $L \times H$ est un tableau à deux dimensions indicé par deux entiers (i, j) affectés respectivement aux lignes et aux colonnes de l'image. La valeur contenue au pixel (i, j) est notée $I(i, j)$. Dans notre cas, chaque pixel contient la distance du capteur au point $p \in \mathbb{R}^3$ dont la projection appartient au pixel (i, j). Les coordonnées du point p sont exprimées dans le repère R_c lié au capteur et représenté Figure 1.3.1-(a). L'axe (Oz) est orthogonal à l'image et les axes (Ox) et (Oy) sont respectivement alignés avec les lignes et les colonnes de l'image.



(a) (b)

FIGURE 1.3.1 – En (a) le repère R_c associé au capteur et en (b) le modèle de projection d'une caméra pinhole.

Le modèle de projection couramment utilisé est celui de la caméra pinhole représenté Figure 1.3.1-(b). Les paramètres α_x et α_y désignent respectivement les angles de vue horizontaux et verticaux du capteur liés à la distance focale par les relations

$$\alpha_x = 2 \arctan\left(\frac{L}{2f_x}\right) \tag{1.3.1}$$

$$\alpha_y = 2 \arctan\left(\frac{H}{2f_y}\right) \tag{1.3.2}$$

où L est la largeur de l'image, H sa hauteur et où f_x, f_y sont les distances focales horizontales et verticales du capteur. En supposant les paramètres intrinsèques (voir section 1.2.1) du capteur connus, les relations permettant de calculer les coordonnées 3D du point $p = (p_x, p_y, p_z) \in \mathbb{R}^3$ à partir du pixel $(i, j, I(i, j))$ de l'image de profondeur I sont données par

$$\begin{cases} p_x = \dfrac{L/2 - j}{L/2} I(i,j) \tan\left(\dfrac{\alpha_x}{2}\right), \\ p_y = \dfrac{H/2 - i}{H/2} I(i,j) \tan\left(\dfrac{\alpha_y}{2}\right), \\ p_z = I(i,j). \end{cases} . \tag{1.3.3}$$

Dans ces relations, i et j désignent respectivement les indices de la ligne et de la colonne du pixel considéré, en choisissant comme origine le coin supérieur gauche de l'image. D'un point de vue pratique, les quantités $\frac{L}{2}$, $\frac{H}{2}$, $\tan\left(\frac{\alpha_x}{2}\right)$ et $\tan\left(\frac{\alpha_y}{2}\right)$ ne sont calculées qu'une seule fois à l'initialisation. Les capteurs 3D peuvent dans certains cas ne pas réussir à estimer pour un pixel donné la valeur de la distance, par exemple sur une surface réfléchissante. Un pixel ne contenant aucune information de distance est dit indéterminé et est affecté d'une valeur nulle. Toutes les composantes du point obtenu par les relations données ci-dessus valent alors zéro. Cette convention sera conservée dans toute la suite de la thèse.

1.3.2 Filtrage des données

Les opérations de filtrage présentées dans cette section ont pour but d'atténuer le bruit de mesure et de supprimer les points abbérants contenus dans les données. Les données traitées peuvent se présenter sous la forme d'un nuage de points 3D (section 1.3.2.1) ou d'une image de profondeur (section 1.3.2.2). Les méthodes décrites dans cette partie sont implémentées dans la librairie de traitement de nuages de points 3D PCL (Point Cloud Library) ([165]).

1.3.2.1 Suppression des points aberrants

Le filtrage présenté dans ce paragraphe a pour objectif de détecter et de supprimer les points dits aberrants (*outliers*) correspondant à des mesures erronées. Sa mise en œuvre permet d'améliorer la qualité des traitements ultérieurs et parfois de les accélérer puisque la taille du nuage de points est réduite. Par exemple, l'estimation des normales aux surfaces et de la courbure sont sensibles au bruit et aux points aberrants (et c'est aussi le cas d'autres descripteurs, notamment ceux basés sur ces estimations).

Soit P un nuage de points 3D. Le k-voisinage d'un point $p \in P$ est noté $N_k(p)$ et contient les k éléments de P les plus proches de p. Une première méthode simple de suppression consiste à définir pour chaque point p un support et à comptabiliser le nombre de points de $N_k(p)$ y appartenant. Si ce nombre de voisins est trop faible le point est considéré comme isolé et supprimé. Le support généralement choisi est une sphère centrée en p et de rayon r. Le rayon r et le nombre minimal de points doivent être choisis conjointement et dépendent notamment de la densité du nuage P. Ce procédé est illustré Figure 1.3.2.

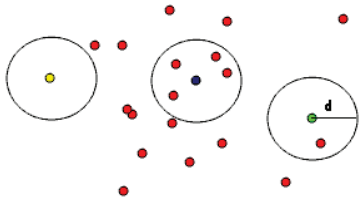


FIGURE 1.3.2 – Les points dont le support ne comportent pas assez de voisins sont supprimés.

Dans [166], les points aberrants sont déterminés par une analyse statistique du voisinage du point considéré. En chaque point $p \in P$, la distance moyenne $\overline{d}_p$ à ses k voisins $q_i \in N_k(p)$ est calculée. La moyenne μ_k et l'écart type σ_k de la distribution ainsi obtenue sont estimés. La méthode consiste alors à conserver les points dont la distance moyenne à ses k voisins est similaire aux autres distances calculées pour les autres points du nuage. Autrement dit, le nuage de points résultant est

$$P^{\star} = \left\{ p \in P, \mid \overline{d}_p - \mu_k \mid \leq \alpha \, \sigma_k \right\} \tag{1.3.4}$$

où μ_k est la distance moyenne observée dans le nuage, σ_k son écart-type et $\alpha > 0$ un paramètre. Ceci revient à supposer que la distribution des distances est gaussienne et à ne conserver que les données présentant une probabilité d'apparition suffisamment élevée (liée à la valeur du paramètre α choisie). Un exemple de résultat est présenté Figure 1.3.3.

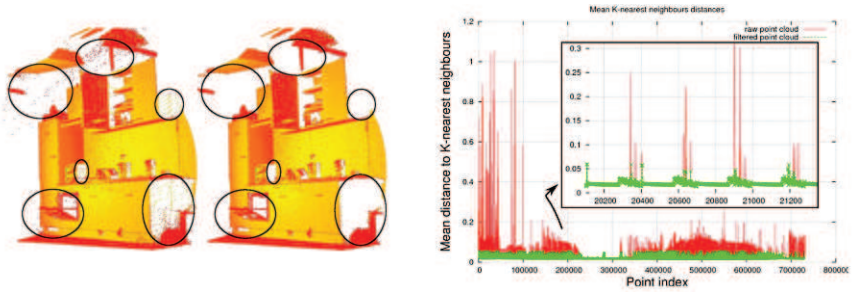


FIGURE 1.3.3 – Les points dont la distance au voisinage est différente de celle des autres points sont supprimés.

Une approche alternative ([163]) consiste à utiliser deux nuages de points P_i et P_{i+1} successifs issus du même point de vue. Pour chaque point $p \in P_i$, le point le plus proche $q \in P_{i+1}$ est recherché. Si ces points sont trop éloignés l'un de l'autre, alors p peut être supprimé. Les paires de points restants peuvent ensuite être moyennés. On suppose dans ce cas que les points abbérants ne sont pas constamment présents. Une autre technique plus complexe ([135]) utilisant des points de vue différents a également été développée.

1.3.2.2 Filtrage bilatéral

Le filtre gaussien est un filtre classique très utilisé en traitement d'image. Il peut être appliqué pour lisser une image et atténuer le bruit. Cette opération permet d'homogénéifier localement les valeurs des pixels. Le poids de chacun des pixels est déterminé par une gaussienne centrée sur le pixel en cours de traitement. Ce filtre est basé sur la proximité spatiale des pixels dans l'image. Dans ce cas, toutes les zones sont considérées de la même manière et les contours des objets ne sont pas conservés. Le filtre bilatéral ([191]) a justement été mis au point pour pallier ce problème. Il a été initialement appliqué à des images en couleur et en niveaux de gris.

Le principe du filtre bilatéral est de prendre en compte à la fois la proximité spatiale et la proximité en terme d'apparence des pixels. Le filtre bilatéral est alors une convolution de noyau

$$\exp\left(-\frac{1}{2}\left(\frac{d(x,x_k)}{\sigma_d}\right)^2\right)\ \exp\left(-\frac{1}{2}\left(\frac{\delta(I(x),I(x_k))}{\sigma_r}\right)^2\right) \tag{1.3.5}$$

où d est une mesure de proximité spatiale entre le pixel x et l'un de ses voisins x_k et δ une mesure de similarité d'apparence. Les paramètres σ_d et σ_r sont des facteurs d'échelles pour chaque mesure.

Il est clair que ce noyau permet à la fois de lisser les régions homogènes tout en conservant les contours. Dans le cas d'une région homogène, le terme d'apparence sera proche de 1 et le filtre se comportera comme un filtre gaussien classique. En revanche, dans le cas d'un pixel situé dans une zone séparant deux objets distincts, les pixels appartenant à l'autre objet auront un poids faible grâce au terme d'apparence. Les contours des objets sont ainsi préservés tout en lissant les régions plus homogènes.

L'application de ce type de filtre aux images de profondeur est très utile. L'apparence désigne ici les valeurs de distance contenues dans l'image. La conservation des contours est primordiale puisque les intensités des pixels représentent des distances. La représentation de la surface d'un objet peut donc être lissée sans pour autant être contaminée par d'autres objets dont la projection sur l'image est proche. Un filtre gaussien ferait apparaître des points ne correspondant à aucun objet physique entre les bords des objets. Ce filtre est connu pour sa simplicité d'implémentation et son efficacité.

1.3.3 Segmentation de données 2.5D

Nous présentons dans cette section deux méthodes de segmentation couramment utilisées sur des images de profondeur : la soustraction de fond en section 1.3.3.1 et l'appartenance à un volume 3D en section 1.3.3.2. Nous nous concentrons pour la première sur les méthodes de type statistique et nous proposons pour la seconde une méthode de calcul rapide.

1.3.3.1 Soustraction de fond

La soustraction de fond est une méthode qui est très utilisée en traitement d'images. Il s'agit d'un traitement préliminaire courant dans le domaine de l'analyse vidéo. Ces méthodes peuvent être mises en œuvre lorsque le capteur et la scène sont fixes l'un par rapport à l'autre. Le capteur doit être statique et toujours observer la même scène. Ces méthodes font la plupart du temps partie des premières séries de traitements appliqués à l'image après acquisition.

Les méthodes de soustraction de fond permettent de segmenter les images en deux classes : l'*avant plan* et l'*arrière plan*. L'avant plan est constitué des objets nouveaux initialement non présents dans le champ de vision de la caméra. Les objets d'intérêts peuvent ainsi être extraits des images. Les traitements ultérieurs pourront donc se concentrer sur ces zones précises de l'image.

Les méthodes de soustraction de fond sont basées sur une modélisation plus ou moins complexe des valeurs des pixels représentant le fond de la scène. Cette modélisation est effectuée durant une période d'apprentissage. Le traitement consiste ensuite à comparer chaque nouvelle image avec le modèle et à décider si le pixel courant appartient ou non au fond. Les zones d'intérêt de l'image sont les objets nouveaux suffisamment différents du fond de la scène qui a été appris. Une mise à jour du modèle au cours temps peut être envisagée pour l'adapter aux évolutions (cas où le fond est modifié) et gagner en robustesse. De manière générale, les algorithmes de soustraction de fond sont composés des trois étapes suivantes

- **Construction** du modèle représentant l'arrière plan à partir d'une séquence de n images. Cette étape n'est réalisée qu'une seule fois à l'initialisation.
- **Comparaison** de l'image courante avec le modèle et **classification** des pixels (avant ou arrière plan).
- **Mise à jour** du modèle avec la dernière image observée pour prendre en compte l'évolution du fond.

Ces méthodes ont été initialement conçues pour opérer sur des images couleur ou en niveaux de gris. Les difficultés rencontrées lors de la mise en œuvre de ces algorithmes sont multiples. Une variation soudaine de luminosité peut modifier significativement tous les pixels de l'image et la quantité de mouvement détectée peut s'avérer très importante. Les ombres des objets sont généralement détectées puisqu'elles modifient également l'apparence de l'image par rapport à l'arrière plan. Un objet dont l'apparence est similaire au fond ne pourra être correctement détecté. Par exemple, un objet blanc sur un fond blanc sera difficilement identifiable.

Dans ces approches, le fond est supposé constant et le capteur immobile. Néanmoins, dans certaines situations, l'arrière plan peut être dynamique et présenter une multimodalité (cas d'un arbre déplacé par le vent). Une modification du fond dans sa composition (un objet est déplacé) perturbe également le bon fonctionnement du traitement. Enfin, la configuration du système peut aboutir à des vibrations déplaçant le capteur, dans le cas par exemple d'une caméra fixée à un mur non rigide et près d'une porte. Ces problèmes ont été constatés et des techniques particulières ont été mises au point pour y remédier.

Les méthodes décrites ci-dessous concernent principalement les images en niveaux de gris (un seul canal de couleur). Dans le cas d'images couleurs, les opérations peuvent être effectuées sur chacune des composantes des pixels pour améliorer les résultats.

La modélisation la plus simple consiste à modéliser chaque pixel par sa valeur moyenne (ou médiane) sur les n premières images de la séquence. Chaque nouvelle image I_t est ensuite comparée à l'image de fond B et l'image du masque de mouvement F_t est alors obtenue suivant la règle de décision

$$F_t(i,j) = \begin{cases} 1 & \text{si } \mid I_t(i,j) - B(i,j) \mid \geq \delta \\ 0 & \text{sinon} \end{cases} \tag{1.3.6}$$

où δ est un seuil de tolérance sur la différence d'apparence. Cette modélisation peut être améliorée en prenant en compte la dispersion des valeurs observées lors de la phase d'apprentissage. La valeur de

chaque pixel (i, j) est modélisée par une gaussienne de centre B_{ij} et de variance σ_{ij}^2. On suppose que la distribution des valeurs prises par un pixel durant la phase d'apprentissage suit une loi Normale. La règle de décision devient alors

$$F_t(i,j) = \begin{cases} 1 & \text{si } \mid I_t(i,j) - B(i,j) \mid \geq k\,\sigma_{ij} \\ 0 & \text{sinon} \end{cases} \tag{1.3.7}$$

où k est un coefficient réel positif contrôlant la distance maximale à la moyenne tolérée. La prise en compte de la variance des observations permet à la méthode de s'adapter aux différentes zones de l'image.

L'approche proposée dans [76] généralise cette modélisation statistique de l'arrière plan. Dans ces travaux, chaque pixel est modélisé par un mélange de K gaussiennes dont la densité de probabilité est de la forme

$$f(x;\Theta) = \sum_{k=1}^{K} \pi_k\, g(x;\theta_k) \tag{1.3.8}$$

où g est la densité de la gaussienne, $\theta_k = (\mu_k, \sigma_k)$ les paramètres de la k-ème composante du mélange et les π_k les proportions du mélange vérifiant $\pi_k \geq 0\ \forall k \in [1, K]$ et $\sum_{k=1}^{K} \pi_k = 1$. Le vecteur $\Theta = (\pi_1, \ldots, \pi_K, \theta_1 \ldots, \theta_K)$ contient tous les paramètres inconnus du modèle.

La phase d'apprentissage consiste à estimer les paramètres du mélange à partir des n premières observations de la séquence d'images. La règle de classification est basée sur la probabilité d'appartenance de l'observation courante au modèle (ou à chacune de ses composantes). Cette méthode permet de remédier aux cas où le fond est dynamique et présente une multimodalité (cas de l'arbre en mouvement). Chaque mode correspond à l'une des composantes du mélange et est pris en compte lors de la phase de décision. Le nombre K de composantes est généralement choisi identique pour tous les pixels. Une technique de choix adaptatif de ce paramètre a été proposée dans [213].

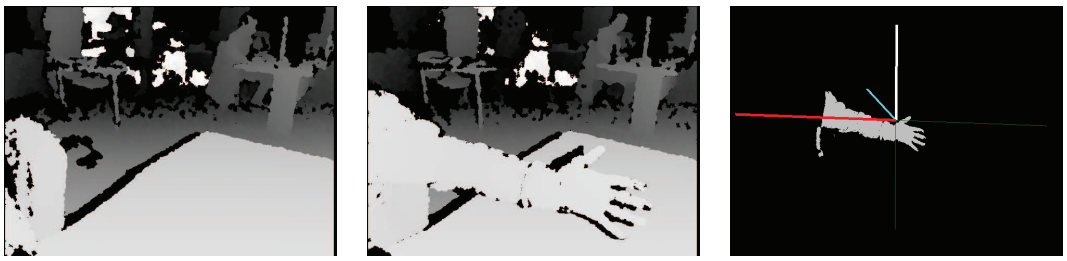
De nombreuses autres méthodes de soustraction de fond ont été proposées. Nous citerons en particulier les Codebooks ([109]), l'estimation non paramétrique de la densité des pixels du fond par un estimateur à noyaux ([77]) et les EigenBackgrounds ([143]) reposant sur une réduction de la dimension par ACP. Des présentations et comparaisons des méthodes classiques ont été réalisées dans [53, 33, 20, 32].

Les algorithmes de soustraction de fond ont été adaptés aux images de profondeur. Il est la plupart du temps possible, moyennant quelques adaptations, d'appliquer les méthodes décrites précédemment. Les problèmes des ombres, des variations de luminosité et des objets d'apparence identique au fond ne se posent plus puisque les images de profondeur fournies par le capteur ne sont pas sensibles à ces facteurs. En revanche, l'information de distance n'est pas toujours disponible sur la totalité de l'image et peut être non fiable dans certains cas selon le type de capteur utilisé. De la même manière qu'un objet blanc sur un fond blanc ne pouvait être être détecté dans des images couleur, un objet spatialement proche du fond pourra ne pas être correctement extrait (par exemple un pied sur le sol). Enfin, l'erreur de mesure s'amplifiant avec la distance au capteur, des objets très proches peuvent être confondus s'ils sont trop éloignés du capteur.

Dans [79], l'arrière plan est modélisé par un modèle à deux composantes dont l'une est gaussienne. Lorsque la correspondance entre les images couleur et de profondeur est connue, ces deux informations peuvent être combinées afin d'améliorer les résultats. Les travaux de [90] utilisent un mélange de gaussiennes pour modéliser la distribution des pixels du fond dans l'espace (R, G, B, D), où les

composantes RGB représentent la couleur et D la profondeur. Une seule composante du mélange est conservée pour modéliser le fond, les autres étant supposées appartenir à des objets en mouvement présents lors de la phase d'apprentissage. La couleur et la profondeur sont traitées séparément puis recombinées pour la classification. Dans [95, 96], toutes les composantes du modèle de mélange sont mises à contribution dans la modélisation de l'arrière plan. Les données sont traitées cette fois ci dans l'espace (Y, U, V, D) dans lequel les couleurs ont été normalisées.

Pour nos expérimentations, nous choisissons d'appliquer la méthode de modélisation des pixels par une valeur moyenne. Cette méthode simple et efficace est suffisante dans notre cadre d'application. Un exemple de résultat obtenu par cette méthode est donné Figure 1.3.4. Le bras de l'individu a été correctement classé en avant plan et des algorithmes plus complexes peuvent être appliqués seulement sur cet objet. Afin de prendre en compte les zones de l'image où la mesure de la distance est difficile (présence de pixels indéterminés), un pixel de l'image de fond est conservé seulement si au moins 15 % des n pixels ayant servi à sa modélisation n'étaient pas indéterminés. Dans le cas contraire le fond n'est pas modélisé et tous les pixels non nuls à cette position seront classés en avant plan. Ceci permet de conserver entièrement les objets passant devant une zone du fond toujours indéterminée (le bord de la table dans l'exemple). Bien entendu les pixels indéterminés ne sont pas utilisés pour calculer la valeur moyenne. La valeur absolue dans (1.3.7) peut également être supprimée. Le seul paramètre à choisir est le seuil $\delta > 0$ représentant la distance minimale séparant un objet du fond.



(a) (b) (c)

FIGURE 1.3.4 – En (a) l'image de profondeur moyenne modélisant le fond, en (b) l'image de profondeur courante et en (c) le nuage des points en mouvement dans le repère associé au capteur 3D.

1.3.3.2 Volume d'intérêt 3D

Dans certaines situations, il peut être utile de ne considérer pour le traitement que les données appartenant à une certaine zone de l'espace. Cette zone peut être définie par rapport au capteur lui même ou par rapport à un élément de la scène. L'utilisation d'un volume d'intérêt 3D permet de sélectionner un sous ensemble de points parmi le nuage complet. Une partie des données est alors directement écartée de la suite des traitements. Ce procédé peut s'avérer très efficace lorsque le cadre d'application le permet. La technique décrite dans cette partie a été mise au point pendant la thèse et n'a, à notre connaissance, pas fait l'objet de publications.

Le cas le plus simple est celui où la zone d'intérêt est définie par un seuillage de l'image de profondeur. Il est très facile d'isoler les points dont la distance au capteur appartient à une plage de valeurs $[z_{min}, z_{max}]$. Il suffit pour cela de parcourir l'image de profondeur et de tester tous les pixels. Le seuillage d'une image est une opération rapide.

Des volumes plus complexes peuvent être considérés. Nous nous plaçons dans le cas ou une équation cartésienne du plan (P) représentant par exemple le sol ou un support de travail est connue. Le volume d'intérêt, noté V, est défini par une base polygonale Q appartenant au support (P) et par une hauteur h. La base Q est définie par ses N sommets appartenant à (P). Un exemple de volume d'intérêt est représenté Figure 1.3.5.

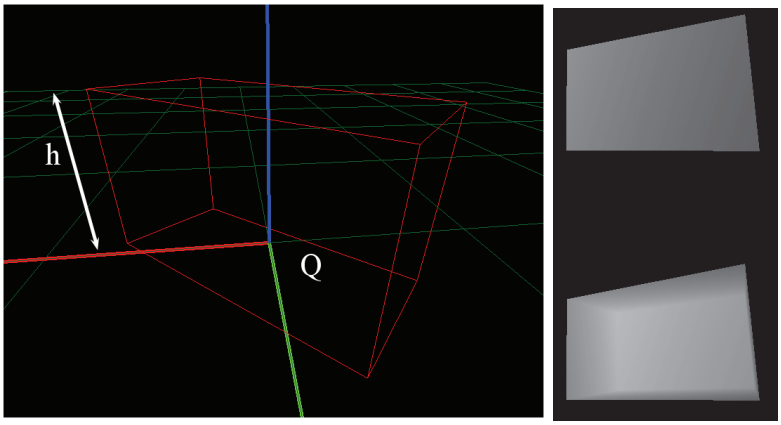


FIGURE 1.3.5 – Le volume d'intérêt V (en rouge) est défini à partir d'une base polygonale Q placée sur le sol (grille verte) et d'une hauteur h. Les axes sont ceux du repère associé au support $R_{support}$. Les masques I_{invis} (haut) et I_{vis} (bas) déduits du volume V sont représentés à droite.

Le but est de tester l'appartenance de chaque point $p = (p_x, p_y, p_z) \in \mathbb{R}^3$ à notre volume d'intérêt V. Une manière naïve de procéder consiste à utiliser les coordonnées $p_{support} = (ps_x, ps_y, ps_z)$ du point dans le repère $R_{support}$ associé au support (P). Dans notre cas, le repère $R_{support}$ est défini de telle manière que l'axe (O_z) soit orthogonal à (P). Les conditions d'appartenance au volume V sont dans ce cas

$$\begin{array}{l} ps_z \in [0, h] \\ (ps_x, ps_y) \in Q \end{array} . \tag{1.3.9}$$

La composante z du point doit être inférieure à la hauteur h de V et le projeté du point sur le support doit appartenir à la base polygonale Q. Le second test est le plus coûteux en temps de calcul. Pour cette raison, nous préférons proposer une technique plus rapide consistant à calculer le rendu des faces visibles d'une part et invisibles d'autre part du volume d'intérêt V par rapport au capteur. A partir de maintenant, nous supposons que la base polygonale Q est convexe.

Tout d'abord, les équations des droites (d_{ij}) passant par le centre optique O du capteur et le pixel (i, j) sont calculées. Ensuite, les points d'intersection entre les droites (d_{ij}) et les différentes faces du volume d'intérêt peuvent être obtenus. Enfin, il suffit de stocker pour chaque pixel la distance minimale et maximale du centre O aux points d'intersection. Les images de profondeur synthétiques résultantes sont les rendus des faces visibles (I_{vis}) et invisibles (I_{invis}) du volume V. Ces images ne sont calculées qu'une seule fois à l'initialisation lorsque le volume a été défini. Notons que le calcul des équations des droites (d_{ij}) nécessitent de connaître les caractéristiques internes du capteur utilisé. Ce procédé est illustré sur le schéma de la Figure 1.3.6 et des images de rendu sont représentées Figure 1.3.5.

Le test d'appartenance du point p issu du pixel (i, j) (dans le repère du capteur) au volume d'intérêt V devient alors simplement

$$p_z \in [I_{vis}(i, j), I_{invis}(i, j)] \,. \tag{1.3.10}$$

Si la composante de profondeur du point p est comprise entre les valeurs minimales et maximales stockées au pixel correspondant, alors le point est bien situé à l'intérieur du volume d'intérêt. Dans le cas contraire, il est situé en dehors. Les résultats obtenus sont illustrés Figure 1.3.7.

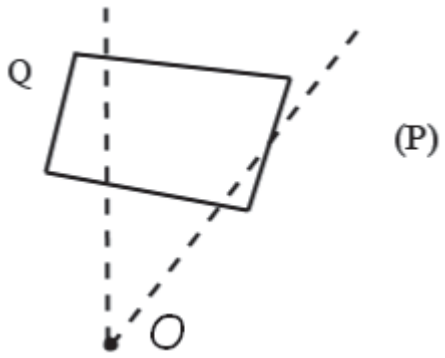
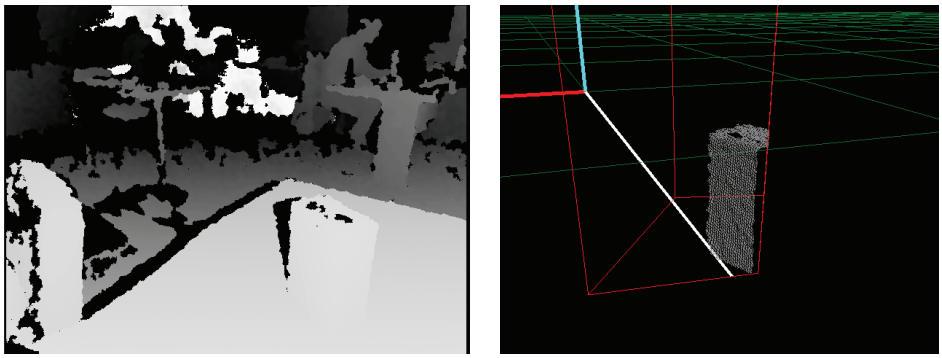


FIGURE 1.3.6 – Pour chaque pixel, les points d'intersections entre la droite (d_{ij}) en pointillés et les faces du volume sont calculés. A chaque pixel est associé une droite. Il est clair qu'un point appartenant au volume est compris entre les deux points d'intersection extrémaux. Le schéma est en vue de dessus.

L'hypothèse de convexité de la base Q est très importante puisqu'elle nous assure que les distances comprises entre les valeurs minimales et maximales calculées pour chaque pixel correspondent bien à des points appartenant au volume. Ceci n'est pas toujours vrai dans le cas d'une base non convexe.



(a) (b)

FIGURE 1.3.7 – En (a) l'image de profondeur et en (b) la portion du nuage de points contenu dans la zone d'intérêt. Seule une partie de la boîte appartient au volume V.

L'avantage de cette méthode est que le temps de calcul du test d'appartenance est inférieur à celui de la méthode naïve. Sa complexité est similaire à celle du seuillage simple de l'image de profondeur. En effet, le test d'appartenance du projeté à la base Q, qui était le plus coûteux, n'est plus nécessaire. De plus, la méthode proposée est directement applicable sur l'image de profondeur initiale sans aucun traitement préalable. La méthode naïve nécessite de calculer les coordonnées du point p à partir de l'image de profondeur et d'effectuer le changement de repère de $R_{capteur}$ vers $R_{support}$. Ceci n'est plus nécessaire avec la méthode proposée, ce qui constitue un gain de temps profitable. Néanmoins, les images synthétiques doivent être recalculées lorsque le volume d'intérêt V est modifié.

1.3.4 Méthodes de croissance de régions

Les méthodes de type croissance de régions (*region growing*) sont assez intuitives et consistent à former des groupes de points selon des critères de proximité. La mesure de proximité la plus naturelle est la distance entre deux points. La distance de Minkowski d'ordre p entre deux points q et q' de dimension n est définie par

$$d_p\left(q, q'\right) = \left(\sum_{i=1}^{n} \mid q_i - q'_i \mid^p\right)^{1/p}. \tag{1.3.11}$$

Les propriétés des points sont examinées afin de déterminer l'appartenance à un groupe. Le nombre de points dans chaque région augmente au fur et à mesure, d'où le nom de cette catégorie de méthodes. L'ajout d'un point q à un cluster C est autorisé si la condition $\min\{d_p(q, q_j), q_j \in C\} < D$ est satisfaite, où D est un seuil sur la distance. Dans ce cas, deux groupes $C_1 = \{p_i\}_{i=1}^{\#C_1}$ et $C_2 = \{q_j\}_{j=1}^{\#c_2}$ distincts vérifieront $\min_{i,j}\{d_p(p_i, q_j), p_i \in C_1, q_j \in C_2\} \geq D$, c'est à dire que deux points séparés par une distance supérieure à D n'appartiendront pas à la même classe. Si l'on choisit $p = 2$, on parle de méthode d'*euclidian clustering*.

D'autres propriétés peuvent être prises en compte lors de la segmentation. L'angle formé par les normales à la surface en chacun des points est une bonne indication de leur appartenance à la même région / surface (voir [154]). Des points proches ayant des normales de même direction appartiennent très probablement au même objet. Le critère associé à cette règle est $\arccos\left(\langle n_i, n_j\rangle\right) \leq \alpha_{th}$, où α_{th} est une tolérance sur l'angle entre les normales n_i et n_j. Ce critère permet d'extraire des régions lisses et de différencier des surfaces différentes séparées par une arête vive (les deux faces d'un toit par exemple) puisque l'orientation des normales change brusquement. Dans le cas où l'information de couleur est disponible, une distance colorimétrique entre les points peut être introduite (voir par exemple [167, 211]).

L'implémentation pratique de ces méthodes repose généralement sur un point de départ appelé *graine* dont les voisins sont testés et éventuellement ajoutés à la région courante. Ce procédé est ensuite répété itérativement tant que des nouveaux points peuvent être ajoutés.

Ces méthodes ont été utilisées pour extraire des objets d'un nuage de points. Par exemple, des objets posés sur une surface plane peuvent être segmentés. Les travaux présentés dans [164] utilisent un algorithme de segmentation basé sur les normales pour extraire des objets d'intérêt. La différence avec ce qui a été présenté ci-dessus est que les points ajoutés à la région ne sont réutilisés comme graines que si leur courbure c est inférieure à un seuil $\widehat{c}$. Ceci permet une exploration dans la direction de la courbure la plus faible. Ces méthodes ont l'avantage d'être rapides. Notons enfin qu'elles peuvent être accélérées en stockant les points dans des structures de données efficaces de type arbres ou grilles de voxels (voir par exemple [211]).

1.3.5 Extraction de primitives

La détection de formes simples dans un nuage de points n'est pas un problème simple. La base de nombreux algorithmes consiste à détecter des sphères, des cylindres, des plans, ou d'autres primitives géométriques. L'extraction de ces formes est une étape du processus de segmentation et de compréhension de la composition de la scène observée. Les méthodes existantes peuvent être classées en deux catégories.

La première consiste à effectuer une première segmentation des données (avec une méthode décrite dans cette partie par exemple). Les points sont préalablement groupés en classes avant qu'un modèle leur soit ajusté. L'erreur d'ajustement commise est une quantification de la bonne ou mauvaise modélisation des données. On se demande si les points d'une classe peuvent être modélisés par une primitive donnée.

La seconde approche consiste à rechercher une forme directement dans le nuage de points complet. Les formes sont recherchées dans le nuage de points tout entier sans segmentation préalable. Pour cela, deux méthodes majeures sont employées. La transformée de Hough, opérant dans l'espace des paramètres, permet d'estimer les vecteurs de paramètres des formes présentes. Le paradigme RANSAC est un processus itératif ne permettant d'assurer la bonne détection qu'avec une certaine probabilité puisqu'elle fait intervenir un tirage aléatoire. RANSAC est très utilisé pour sa robustesse aux points abbérants (dits *outliers*).

Les formes considérées dans cette partie sont des formes géométriques simples, même si certaines méthodes ont été adaptées à des objets quelconques. Les deux types d'approches ont chacune leurs forces et leurs faiblesses. Nous présenterons les méthodes de type moindres carrés dans un premier paragraphe tandis que les deux suivants décriront les méthodes de type transformée de Hough et RANSAC.

Ajustement par moindres carrés Étant donné un ensemble de n points $\{p_i\}_{i=1}^n$ de $\mathbb{R}^d$, $d \geq 2$, le but est d'ajuster un modèle décrivant une forme géométrique définie par

$$\forall x \in \mathbb{R}^d, \quad f(x, \theta) = 0 \tag{1.3.12}$$

où θ est un vecteur de paramètres et f une fonction à valeurs réelles. Pour déterminer le vecteur de paramètres $\theta^\star$ caractérisant la forme géométrique, on cherche à minimiser le critère de moindres carrés

$$E(\theta) = \sum_{i=1}^{n} f(p_i, \theta)^2. \tag{1.3.13}$$

Le critère (ou l'erreur) E à minimiser est la somme des carrés des résidus (généralement notés $e_i(\theta)$), quantifiant la proximité entre le point p_i et la surface de la primitive géométrique. La valeur $E(\theta)$ est donc l'erreur globale commise en approchant les données par le modèle de paramètre θ. La minimisation de l'erreur E permet de déterminer le paramètre optimal $\theta^\star$ du modèle expliquant au mieux (au sens de l'erreur choisie) les observations. La minimisation peut dans certains cas se faire algébriquement. On peut parfois se ramener à la résolution d'un système linéaire $A\,\theta = b$. Dans le cas contraire, un algorithme de moindres carrés non-linéaires peut être appliqué. Par exemple dans [176], la modélisation des données par un cylindre nécessite d'utiliser un algorithme de Levenberg-Marquardt. Une illustration est représentée Figure 1.3.8 dans le cas du cylindre.

FIGURE 1.3.8 – Les données sont ajustées par un modèle de cylindre.

Transformée de Hough La transformée de Hough est une technique initialement créée pour détecter des droites dans les images ([101, 105]). Elle est basée sur un système de votes dans l'espace des paramètres de la forme recherchée. De multiples instances d'un modèle peuvent être détectées dans un ensemble de données sans en connaître par avance le nombre. La segmentation préalable des points appartenant à la forme n'est pas nécessaire et l'algorithme travaille sur le nuage de points complet pouvant contenir de nombreux autres objets.

La transformée de Hough opère dans l'espace des paramètres du modèle. Sa dimension dépend donc de la paramétrisation adoptée. Dans le cas de la recherche de droites d'équation $y = ax + b$ dans un ensemble de n points du plan $E = (x_i, y_i)_{i=1}^{n}$, l'espace de recherche est de dimension 2 : une pour a et une pour b. Chaque point (x_i, y_i) va voter pour un ensemble de paramètres définit par $b = y - ax$. Une droite dans l'espace (a, b) est alors associée à chaque point de E. Les points (a, b) de cette droite représentent toutes les équations possibles des droites du plan passant par le point (x_i, y_i) considéré. En pratique, la paramétrisation $r = x\cos(\theta) + y\sin(\theta)$ est préférée puisqu'elle permet de représenter les droites verticales et les valeurs des paramètres sont généralement finis (ce que ne permet pas la paramétrisation précédente). La technique est illustrée dans le cas de la recherche de droites par un schéma Figure 1.3.9.

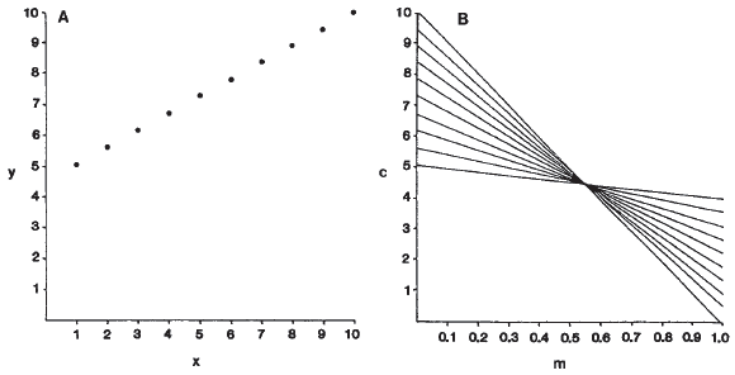


FIGURE 1.3.9 – Les points du plan (à gauche) votent chacun pour une droite dans l'espace des paramètres (à droite). Le point d'intersection représente les paramètres (a, b) de la droite recherchée.

L'implémentation de la transformée de Hough passe par la construction d'un accumulateur qui est une discrétisation de l'espace de recherche. Les votes sont accumulés dans ce dernier et les détections correspondent aux entrées ayant recueillies suffisamment de votes. Les paramètres de cet algorithme sont les pas de discrétisation pour chacun des paramètres et le nombre minimal de votes pour valider une détection.

Cette technique a été utilisée pour détecter des droites, des cercles et des ellipses dans le plan et s'est avérée efficace. Néanmoins, la détection d'objets 3D demande une quantité de calcul importante puisque le nombre de paramètres est élevé. Il s'agit de l'inconvénient majeur de cette méthode, la quantité de mémoire pour stocker l'accumulateur augmente fortement avec la dimension du problème. Il est donc clair que le choix de la paramétrisation (et donc le nombre de paramètres) va beaucoup influencer l'efficacité de l'algorithme. La méthode a été généralisée dans [17] afin de détecter des formes arbitraires.

La méthode a été appliquée à la détection de formes 3D comme les sphères ([3]), les plans ([188, 31, 30]) ou encore les cylindres ([153]). Des auteurs ont proposé d'appliquer plusieurs transformées successives en dimension faible plutôt qu'une seule en dimension élevée, comme par exemple dans [153] pour la détection de cylindres. Un exemple de résultat obtenu par cette méthode est présenté Figure 1.3.10. Cette idée de vote dans un espace de paramètres (*Hough voting scheme*) a été mise en œuvre dans [192] pour détecter des objets contenus dans une base de données. De manière générale, la complexité de la méthode pour la détection de formes 3D reste importante à cause du grand nombre de paramètres.

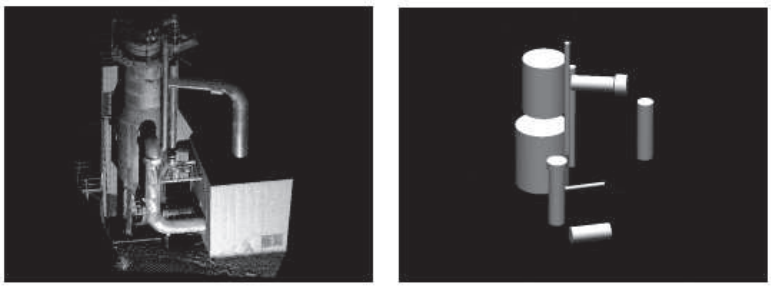


FIGURE 1.3.10 – De multiples instances de cylindres ont été extraites (droite) de la scène (gauche).

Paradigme RANSAC Le paradigme RANSAC (RANdom SAmple Consensus) est un outil permettant d'extraire des formes géométriques d'un nuage de points. Cette méthode a été introduite dans [84]. Son succès réside dans sa robustesse aux données aberrantes. RANSAC n'est pas un simple algorithme, mais une stratégie générale (*RANSAC-like strategy*) ([70, 214, 168, 170, 193]).

Dans cette partie, on appellera *inlier* un point appartenant à la forme recherchée et *outlier* un point n'y appartenant pas. L'algorithme RANSAC a été appliqué à l'extraction de diverses primitives, comme par exemple des plans ([188, 174, 208]), des sphères ([169]) et plus généralement toute forme paramétrée. Il s'agit d'un processus itératif non déterministe. Une itération de l'algorithme peut se décomposer en deux étapes :

– sélection aléatoire d'un ensemble minimal de N points et estimation des paramètres θ du modèle M
– recherche des points consistants au modèle et réajustement éventuel

Les méthodes de moindres carrés présentées précédemment utilisent l'intégralité des données qui leur sont fournies. Le modèle est estimé à partir d'un ensemble de points sans se demander s'ils appartiennent ou non à la forme recherchée. Au lieu d'utiliser l'intégralité des n points disponibles, l'algorithme RANSAC va sélectionner N points aléatoirement, N étant le nombre minimal d'observations nécessaires pour estimer le modèle. Par exemple, si le modèle recherché est un plan, N vaut 3. Un premier vecteur de paramètres θ est alors estimé.

L'étape suivante consiste à rechercher les points "proches" du modèle parmi toutes les données. Ces points sont appelés *points consistants* au modèle candidat. Leur nombre donne une première quantification de la qualité de l'hypothèse testée. Un point p est dit consistant au modèle $M(\theta)$ si la distance $d(p, M(\theta)) \leq t$, où t est un seuil de consistance et d une distance adéquate. Enfin, si le nombre de points consistants est suffisamment élevé, le modèle est réajusté.

Ce processus est reproduit itérativement k fois. A chaque itération, un nouvel ensemble minimal de points est sélectionné. Il est clair que cet algorithme ne permet de trouver correctement le bon modèle qu'à une certaine probabilité dépendant du nombre d'itérations effectuées. Un exemple fréquemment utilisé pour présenter cette méthode est donné Figure 1.3.11.

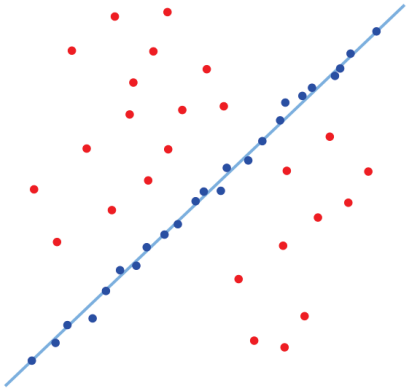


FIGURE 1.3.11 – La droite obtenue par la méthode RANSAC est correcte et n'a pas été influencée par la présence des outliers (en rouge). Les points bleus sont consistants au modèle. Une méthode de moindres carrés classique aurait pris en compte tous les points et aurait fourni un mauvais résultat.

La probabilité de succès de l'algorithme dépend du nombre d'itérations effectuées. Soit p la probabilité de succès souhaitée et ω la probabilité de choisir un point inlier parmi toutes les données. La probabilité ω peut s'écrire

$$\omega = \frac{n_{inliers}}{n} \tag{1.3.14}$$

où $n_{inliers}$ est le nombre de points appartenant à la forme recherchée. Cette quantité est généralement inconnue (dans certaines applications, la proportion de points appartenant à la forme recherchée peut être approximativement connue).

En supposant les tirages indépendants entre eux, la probabilité de choisir N points inliers est ω^N et donc la probabilité de choisir au moins un outlier, c'est à dire de travailler sur un mauvais modèle est $1-\omega^N$. Enfin, la probabilité de ne jamais tirer un ensemble minimal de N points ne contenant que des inliers en k itérations est de $(1-\omega^N)^k$.

On souhaite que

$$(1-\omega^N)^k = 1-p, \tag{1.3.15}$$

ce qui donne

$$k = \frac{\log(1-p)}{\log(1-\omega^N)}. \tag{1.3.16}$$

Cette dernière relation permet de choisir le nombre k d'itérations à effectuer afin de trouver une forme représentant une proportion ω des données et avec une probabilité de réussite p. En pratique, il est préférable de fixer un nombre maximal d'itérations k_{max} pour éviter d'effectuer un nombre trop important d'itérations. La recherche d'un objet dans un nuage de points prendra d'autant plus de temps que l'objet sera de petite taille et que la probabilité de succès sera élevée.

De nombreuses variantes de cette méthode ont été proposées. Une première amélioration simple consiste à guider la sélection aléatoire des N points. En effet, lorsqu'un premier point a été choisi, il peut être judicieux de poursuivre la sélection dans une zone plus ou moins proche de ce premier tirage. Par exemple, si l'on recherche une sphère de rayon inférieur à r cm, il est préférable de choisir les autres points à moins de r cm du premier. D'autres règles peuvent être imaginées selon les informations a priori disponibles. Ce procédé augmente la probabilité de sélectionner des inliers. Des structures de type octree, kd-tree ou une grille de voxels peuvent être utilisées afin d'accéder plus rapidement aux points proches d'un élément donné.

Les variantes proposées incluent WaldSAC ([127]), PROSAC (PROgressive SAmple Consensus) ([54]), LoRANSAC (LOcally Optimized RANSAC) ([55]), et de nombreuses autres. Des variantes de la méthode sont résumées dans [155]. La méthode RANSAC a été étendue dans [148] à la recherche d'objets plus complexes (position et orientation) dans une scène.

1.3.6 Changement de repère

Le changement de repère est une opération simple de passage d'un système de coordonnées à un autre. Dans notre cadre d'étude, nous aurons besoin de calculer la transformation géométrique permettant de passer d'un repère à un autre et d'appliquer cette transformation à un ensemble de points 3D. Nous avons déjà introduit le repère dans lequel le nuage de points calculé est exprimé. Il est représenté sur la Figure 1.3.1 de la section 1.3.1.

On peut vouloir associer à un objet particulier un repère qui lui est propre dans lequel il sera plus facile d'effectuer certaines opérations. Par exemple, si l'on suppose que le plan 3D représentant un support de travail (par exemple le sol d'une pièce ou une table), un repère local dont l'axe de la composante z est aligné avec la normale à ce plan peut être défini. La transformation liant les deux repères décrit la position et l'orientation relative des deux systèmes de coordonnées et donc entre les deux objets considérés.

Soient $p = (p_x, p_y, p_z) \in \mathbb{R}^3$ un point de l'espace dont les coordonnées sont exprimées dans un repère orthonormé R_1 et R_2 un second repère orthonormé. La transformation rigide décrivant la relation géométrique entre les deux repères (de R_1 vers R_2) s'écrit

$$\phi = (R, T) \tag{1.3.17}$$

où $R = (r_{ij})_{i,j=1}^{3}$ est une matrice de rotation de $\mathbb{R}^3$ et $T = (t_x, t_y, t_z)^T$ un vecteur de translation dans l'espace. La transformation ϕ est une isométrie conservant les distances et ne déforme pas le nuage de points. Les cordonnées $p^{'}$ du point p exprimées dans le repère R_2 sont

$$p^{'} = \phi(p) = Rp + T. \tag{1.3.18}$$

Cette notation peut être simplifiée en considérant les coordonnées homogènes du point p,

$$p = (p_x, p_y, p_z, 1)^T. \tag{1.3.19}$$

Avec cette notation, la transformation rigide peut s'exprimer sous la forme d'une matrice carrée de taille 4

$$M = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

et les coordonnées du point p dans R_2 sont alors données par le produit matriciel

$$p^{'} = M\, p. \tag{1.3.20}$$

Lorsque les deux repères sont connus, on cherche à calculer la transformation rigide ϕ de paramètres (R, T). Les axes du repère R_1 sont notés $\left(O_x^1, O_y^1, O_z^1\right)$ et ceux du repère R_2 exprimés dans R_1 $\left(O_x^2, O_y^2, O_z^2\right)$. Les paramètres de la transformation sont données par

$$R = \left[O_x^2, O_y^2, O_z^2\right] \text{ et } T = \overrightarrow{O^1 O^2}. \tag{1.3.21}$$

La transformation rigide liant les deux repères est représentée Figure 1.3.12. Le plan 3D modélisant le support peut être obtenu par une des méthodes présentée en section 1.3.5.

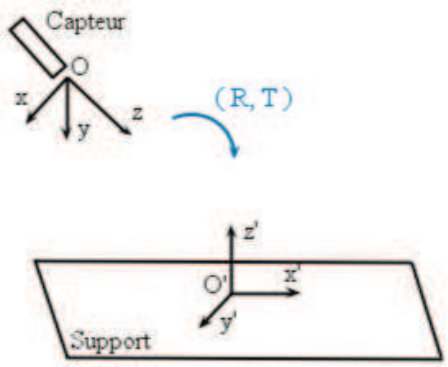


FIGURE 1.3.12 – Transformation rigide liant les repères associés au capteur et à un support de travail.

1.4 Conclusion

Nous avons dans ce chapitre présenté succinctement les différentes technologies d'acquisition de données 3D. Nous avons insisté sur les systèmes stéréoscopiques, à temps de vol et à lumière structurée. Les capteurs fonctionnant à l'aide d'une lumière structurée seront utilisés lors de nos expérimentations. Les cartes de profondeur produites sont suffisamment précises et contiennent assez peu de régions indéterminées, pour une utilisation en intérieur. Les nuages de points 3D seront calculés avec les relations données dans cette section. La maîtrise de ce calcul nous permet d'une part de maîtriser entièrement la chaine de traitement et d'autre part de ne pas diminuer la fréquence d'acquisition. Nous n'avons pas traité dans ce chapitre la question de la précision de la mesure de la distance. Ce point a été développé dans [112]. Dans notre cadre d'application, la précision des capteurs à lumière structurée se révèle suffisante (de l'ordre de quelques millimètres).

Nous avons ensuite dans un second temps présenté quelques algorithmes de traitements des données 3D. Les opérations de bas niveaux ont la plupart du temps pour but d'améliorer la qualité des données comme c'est le cas du filtrage bilatéral et de la suppression des points isolés. Après ce filtrage, une deuxième série de traitements cherche à segmenter et à modéliser les données 3D. Des techniques de soustraction de fond, de sélection de volumes, de croissance de régions et de recherche de primitives géométriques simples ont été présentées en détail. Une partie des techniques décrites seront appliquées lors des expérimentations effectuées dans les chapitres suivants de la thèse.

Ce chapitre offre un premier aperçu des avantages et inconvénients des capteurs 3D par rapport aux caméras traditionnelles dans le cadre d'un système de vidéo-surveillance.

Chapitre 2

Un algorithme automatique de détection et de suivi de personnes à partir d'une séquence d'images de profondeur

Sommaire

2.1 Introduction

Nous nous intéressons dans ce chapitre au problème de la détection et du suivi de personnes dans des images de profondeur. Ce problème classique en traitement d'image est une composante importante de nombreux systèmes de vision par ordinateur. Il a fait l'objet de beaucoup de travaux de recherche et reste une tâche difficile encore étudiée actuellement. Dans le contexte de la vidéo-surveillance, les applications de comptage, de sécurisation de périmètres, d'évacuation d'urgence de bâtiments et d'analyse marketing sont basées sur la connaissance de la position des personnes dans les images.

Les solutions algorithmiques de détection de personnes dépendent fortement du point de vue depuis lequel la scène est observée. En effet, selon le cas de figure considéré, l'apparence et la forme des personnes varient, ce qui conduit à adopter des stratégies différentes. Nous classons donc les méthodes de la littérature en deux catégories : celles concernant des personnes vues de face ou de profil et celles concernant des personnes vues de dessus. La Figure 2.1.1 illustre cette différence. Le travail présenté dans ce chapitre traitera de personnes vues de dessus.

En plus de la configuration du système, le type d'information retourné par le capteur guide lui aussi la conception des algorithmes. Nous nous focalisons ici sur les travaux utilisant des caméras couleur classiques ou des capteurs de profondeur. Les premiers travaux utilisaient des caméras classiques fournissant des images couleur contenant une information sur l'apparence et la forme 2D des objets. L'apparition de capteurs capables de mesurer une distance pour chaque pixel a conduit la communauté à développer de nouveaux algorithmes. Ces systèmes fournissent généralement pour chaque pixel une information colorimétrique et une information de distance. La forme 3D de la surface des objets peut alors être prise en compte dans des traitements. De tels capteurs ont été présentés en détail dans le premier chapitre. Dans ce chapitre, nous considèrerons des cartes de profondeur acquises avec un capteur à lumière structurée. Des images couleur et de profondeur sont présentées Figure 2.1.1.

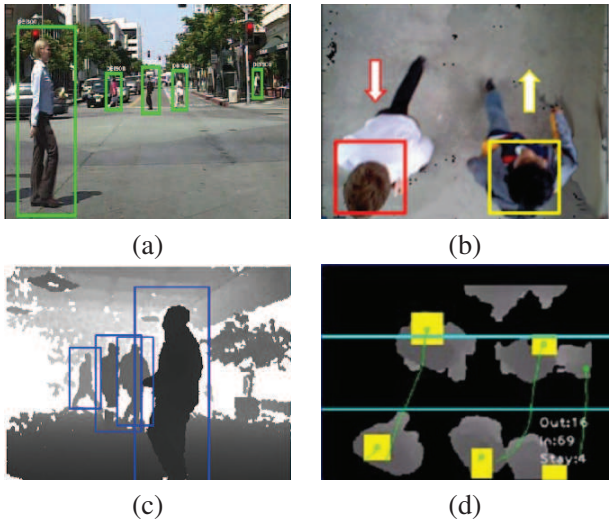


FIGURE 2.1.1 – En (a) et (c), les personnes sont vues de face ou de profil alors qu'en (b) et (d), elles sont vues de dessus. En (a) et (b) les images donnent une information colorimétrique alors qu'en (c) et (d) elles donnent une information de distance.

Commençons par mentionner quelques algorithmes traitant le cas de la vue de côté. Les histogrammes de gradients orientés (HOG) (voir par exemple [63, 78, 72]) sont des descripteurs donnant lieu aux méthodes de détection les plus efficaces dans les images couleur. Des histogrammes 1D représentant l'orientation des gradients de l'image sont utilisés pour entraîner des classifieurs et détecter des personnes. Dans [182], les gradients sont extraits des images couleur et de profondeur (données RGB-D). La méthode de détection proposée utilise un histogramme décrivant localement la direction de la variation de la profondeur (HOD) et un histogramme de gradients orientés (HOG). Les résultats obtenus sur chacune des images sont combinés pour profiter des avantages des deux détecteurs. Dans [210], les auteurs construisent une carte de hauteur dans laquelle chaque pixel contient non plus la distance au capteur mais la distance au sol. Les individus sont ensuite extraits par un algorithme de croissance de régions. Cette notion de carte de hauteur est également exploitée dans [207]. L'image est segmentée en plusieurs classes correspondant à des intervalles de hauteur. Des opérations morphologiques, plus particulièrement une ouverture d'élément structurant circulaire, permettent de déterminer les positions des têtes dans l'image. Enfin, un suivi est réalisé via un filtre de Kalman pour prédire les positions futures des têtes. Des modèles de forme 2D et 3D ont été définis dans [205] pour modéliser les têtes. Les personnes complètes sont par la suite extraites de l'image de profondeur par un algorithme de croissance de régions. Un apprentissage intensif sur des images de profondeur a été proposé dans [177, 178] pour modéliser les personnes par un modèle squelettique. Cet algorithme est rapide et les résultats obtenus sont très bons. La méthode présentée dans [104] constitue un autre exemple de travaux faisant intervenir un algorithme d'apprentissage. Le descripteur utilisé (RDSF) décrit la différence entre des histogrammes de profondeur locaux.

Plusieurs difficultés apparaissent lors de la recherche de personnes vues de face. La première d'entre elles réside dans le nombre très important de postures possibles. En effet, selon l'activité effectuée par la personne, sa posture peut varier de manière importante. En plus de cela, dans cette configuration en vue de face, une personne peut en cacher une autre. Ce phénomène est connu sous le nom d'occultation. Enfin, deux personnes proches seront difficiles à dissocier. Il faut ajouter à ces principales difficultés les problèmes liés à l'environnement et à l'acquisition comme par exemple les changements de luminosité (capteurs couleur) et les pixels indéterminés (capteurs 3D).

Intéressons nous plus précisément aux travaux antérieurs dont la problématique est proche de la nôtre, i.e. cherchant à détecter des individus vus de dessus. Une première série de travaux utilise pour cela l'information de couleur. Le système proposé dans [113] consiste en une caméra verticale observant la scène depuis le dessus. L'algorithme de détection est basé sur une soustraction de fond, des opérations morphologiques et un suivi temporel des objets. Le mouvement des objets est stocké pour ensuite prédire la position des objets lors de la recherche de correspondances. La zone de comptage est délimitée par des lignes virtuelles. Dans [10], les personnes sont extraites de l'image de mouvement par un algorithme de classification K-means. Le nombre de personnes est estimé par le nombre maximal de classes pour lequel la distance inter-classes minimale est acceptable. Le suivi est effectué par un algorithme de type glouton dans lequel les correspondances sont choisies les unes après les autres. Le comptage de personnes effectué dans [18] analyse les objets en mouvement le long de plusieurs lignes virtuelles de comptage. Les pixels des images de mouvement sont additionnés sur chaque ligne pour détecter les objets les franchissant. Le sens de passage est déterminé par le calcul du flot optique entre des régions d'intérêt. Les résultats des différentes lignes de comptage sont alors combinés pour rendre la méthode plus robuste. Dans [136], les personnes sont détectées par une recherche de cercles dans l'image des contours du mouvement via une transformée de Hough. Le suivi est réalisé par une méthode de tracking par flot optique. Les trajectoires obtenues sont ensuite validées pour supprimer

les fausses détections.

Des méthodes ont bien évidemment été développées dans le cas de cartes de profondeur. Dans [99], les personnes sont détectées dans l'image de mouvement par un algorithme de croissance de régions. Les détections sont ensuite filtrées grâce à un filtre de Kalman étendu. Le système proposé inclut un mécanisme d'identification des individus sur des caméras différentes. Un motif sphérique composé de zones positives et négatives a été utilisé dans [187] pour trouver des têtes. Cette technique est basée sur un a priori fort puisqu'elle repose sur un modèle de répartition des points représentant une tête. En vue de dessus, les têtes des individus correspondent à des minima locaux dans l'image de profondeur. Un algorithme de WaterFilling [204] exploite cette idée pour détecter et suivre des individus dans un flux de personnes. Les trajectoires des individus trackés sont analysées pour être validées. Une autre approche [86] consiste à extraire des parties de l'image avec une méthode de croissance de régions et à identifier parmi elles les têtes par le biais d'un score. La méthode de suivi proposée, basée sur l'algorithme Hongrois, permet à la méthode de résoudre les cas de non détections. Enfin, des méthodes se servant de bases de données contenant des images dont la vérité terrain est connue ont été développées. Un SVM (Support Vector Machine) a été entraîné dans [158] pour discriminer la tête des épaules. En effet, la tête et les épaules ont des formes proches (arrondies). Une erreur classique est de détecter une épaule comme étant la tête d'une personne. Le descripteur choisi est basé sur des histogrammes décrivant la répartition des différences de profondeurs moyennes entre des blocs de position prédéfinies dans l'image. Dans [87], plusieurs images de profondeur sont sélectionnées dans une base de données par comparaison de vecteurs descripteurs. La position de la tête est ensuite estimée par interpolation (pondérée) des résultats contenus dans la base.

Dans les méthodes présentées ci-dessus, certains problèmes sont récurrents et difficiles. Plusieurs personnes proches les unes des autres sont plus difficiles à détecter et à suivre que des personnes isolées. Notons toutefois que dans le cas d'un capteur vertical, le problème d'occlusion est limité puisqu'une personne ne peut pas être entièrement cachée par une autre. Les méthodes de suivi employées sont souvent sensibles à des mouvements brusques. Enfin, il arrive parfois que les personnes portent des objets dont la forme et l'apparence sont imprévisibles, ce qui peut engendrer des erreurs. Pour terminer, notons que certaines des méthodes présentées ci-dessus ont été conçues pour suivre des personnes en mouvement selon une direction connue (par exemple de haut en bas dans l'image).

Dans ce chapitre, nous choisissons de nous baser sur les travaux [86]. Nous utilisons un capteur 3D à lumière structurée positionné verticalement. Le système que nous considérons, présenté dans la section suivante, présente quelques différences avec celui de [86]. Par conséquent, nous apportons un certain nombre de modifications à l'algorithme initial pour prendre en compte les spécificités de notre installation. En particulier, la contrainte principale est que notre capteur 3D est positionné à une hauteur faible. Cette contrainte est liée à l'infrastructure des locaux. Elle implique que les personnes à détecter peuvent être partiellement invisibles à cause de leur trop grande proximité avec le capteur 3D. De plus, les têtes des personnes peuvent être tronquées par les bords de l'image. Tous ces points seront abordés plus précisément en début de chapitre lors de la présentation du système. L'efficacité de la méthode sera quantifiée sur des séquences d'images de profondeur.

Le chapitre est organisé de la manière suivante. Le système considéré et le pré-traitement des images de profondeur est décrit section 2.2. L'algorithme de segmentation et de suivi des têtes est détaillé section 2.3. Des expérimentations seront menées section 2.4. Nous conclurons enfin le chapitre section 2.5.

2.2 Pré-traitement de l'image de profondeur

Dans cette section nous décrivons brièvement le système de suivi et les algorithmes de pré-traitement appliqués aux images de profondeur. Le système considéré dans ce chapitre, schématisé Figure 2.2.1, est composé d'un capteur 3D positionné à une hauteur d'environ 2.50 mètres en position quasi verticale et dirigé en direction du sol. Les individus sont situés en dessous de la caméra. La différence principale avec le système considéré dans [86] est que le capteur est placé beaucoup plus proche des têtes des personnes que l'on souhaite détecter.

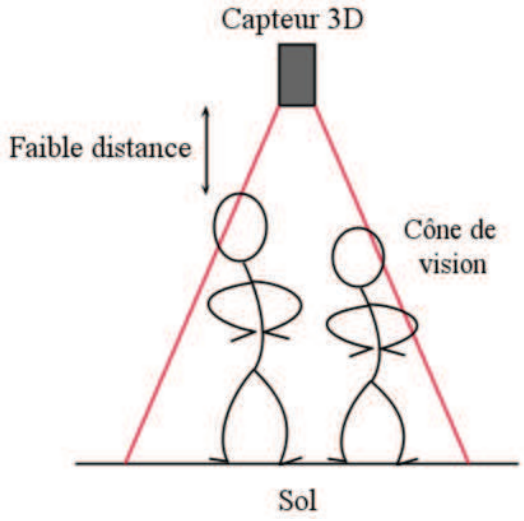


FIGURE 2.2.1 – Schéma représentant le système considéré.

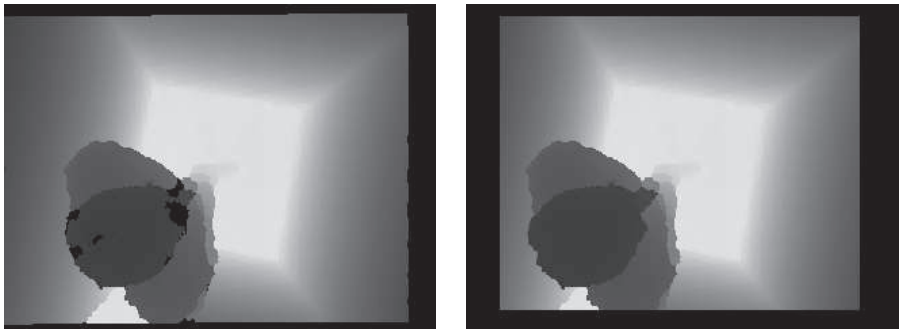
Les images de profondeur acquises par les capteurs 3D contiennent des pixels indéterminés ne contenant aucune information de profondeur. L'orientation des surfaces des objets, leur composition et la lumière naturelle du soleil peuvent gêner la mesure de la distance. Nous avons vu au Chapitre 1 que tous les types de capteurs 3D sont concernés par ce problème.

L'algorithme sur lequel ce chapitre est basé analyse l'image de profondeur pour trouver les têtes des individus. Les zones indéterminées peuvent donc nuire considérablement à son bon fonctionnement. La première étape de la méthode consiste donc à donner une valeur de distance aux pixels indéterminés.

On distingue deux catégories de pixels indéterminés. La première concerne les pixels dont le contenu n'a pas pu être calculé pour les raisons décrites ci-dessus. La seconde inclus les pixels représentant des objets situés trop proches du capteur. Comme mentionné au Chapitre 1, les différents capteurs 3D ont des distances minimales et maximales de fonctionnement. Notre système utilise un capteur à lumière structurée de type Kinect ([60]) dont la distance minimale d'observation est d'environ $d_0 = 500$ mm. Par conséquent tous les objets situés à une distance inférieure à d_0 ne seront pas visibles dans les données et on observera une tache noire sur l'image de profondeur. Une image provenant du système considéré contenant des pixels des deux catégories est donnée Figure 2.2.2. Sur cet exemple, les pixels représentant le bord du bras appartiennent à la première catégorie tandis que ceux de la tête font partie de la seconde. La cause de l'indétermination étant de nature différente, les deux classes de pixels seront traitées par des méthodes différentes.

FIGURE 2.2.2 – Image couleur et de profondeur provenant du système de suivi. Les pixels du bras appartiennent à la première catégorie de pixels indéterminés et ceux de la tête font partie de la seconde.

Dans [86], les auteurs appliquent un algorithme simple pour donner une valeur aux pixels indéterminés appartenant à la première catégorie. Les zones indéterminées sont reconstruites de l'extérieur vers l'intérieur, du bord de la zone en direction du centre et par bandes. Les pixels indéterminés dont au moins l'un des quatre voisins contient une distance peuvent être restaurés. La valeur de distance retenue est la profondeur moyenne des voisins. Un parcours de l'image complète aura pour effet de remplir une partie des pixels indéterminés. Plusieurs parcours successifs sont effectués pour finalement tous les supprimer. Cette approche est simple et son coût calculatoire est faible. Un exemple de résultat est donné Figure 2.2.3.



(a) (b)

FIGURE 2.2.3 – En (a) l'image originale et en (b) l'image de résultat avec l'algorithme proposé dans [86]. Les zones indéterminées sont correctement reconstruites.

Du fait de la position basse du capteur, nous sommes confrontés à la seconde catégorie de pixels indéterminés, ce qui n'est pas le cas dans [86]. La tête d'une personne assez grande sera entièrement indéterminée et ne pourra par conséquent pas être détectée correctement. La méthode décrite dans le paragraphe précédent n'est alors plus suffisante. Le résultat obtenu sur une image où la tête est entièrement indéterminée est représenté Figure 2.2.4. On constate que les pixels appartenant à la tête n'ont pas été correctement ré-estimés. Les valeurs affectées à ces pixels sont trop élevées et correspondent aux profondeurs des objets adjacents. Les pixels valides situés autour de la tête sont dans cette configuration les épaules. Cette méthode n'est donc pas satisfaisante pour ce cas particulier et un autre algorithme doit être développé.

(a) (b)

FIGURE 2.2.4 – En (a) l'image originale et en (b) l'image de résultat avec l'algorithme proposé dans [86]. La tête n'est pas correctement reconstruite.

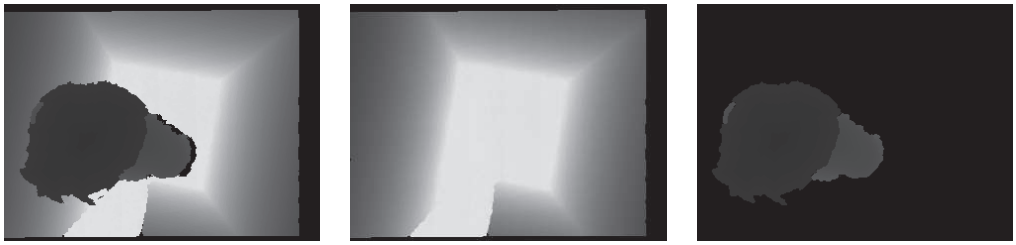
Le problème consiste à affecter des valeurs de distance cohérentes à des objets entièrement indéterminés dans l'image. Ces objets étant assez proches du capteur, leurs tailles en pixels sur l'image sont importantes. Par conséquent, nous déterminons les zones représentant des objets trop proches par un algorithme de croissance de régions sur les pixels indéterminés. Une composante connexe de dimensions supérieure à $l_{min} \times h_{min}$ et composée d'au moins n_{min} pixels indéterminés est considérée comme proche et sera traitée par l'algorithme suivant. Les pixels de la région sont remplis du bord vers le centre comme dans l'algorithme précédent. La première bande de pixels est affectée de la profondeur d_0 qui correspond à la distance limite en dessous de laquelle la mesure ne peut plus être effectuée. Nous choisissons cette distance particulière car l'objet est supposé être proche du capteur à une distance inférieure à cette limite. L'image est ensuite reparcourue et les pixels appartenant à la bande suivante sont affectée de la valeur $d_0 - i\,\epsilon$, où i est l'indice de la bande courante et ϵ un pas en mm. Le paramètre ϵ, déterminé expérimentalement, contrôle la forme des objets reconstruits.

Cette méthode de restauration suppose que les objets à reconstruire sont de forme arrondie, ce qui est souvent le cas puisque nous nous intéressons à des individus. Dans le cas d'un bras levé par exemple, les pixels ajoutés donneront une forme allongée ne ressemblant pas à une tête. Un exemple de résultats est donné Figure 2.2.5. La tête a été reconstruite de manière acceptable puisque sa forme permettra sa détection lors des étapes suivantes de l'algorithme. La complexité de la méthode est faible.

(a) (b) (c)

FIGURE 2.2.5 – En (a) l'image initiale, en (b) l'image de résultat avec la méthode proposée et en (c) le nuage de points 3D de la tête reconstruite.

Enfin, un algorithme de soustraction de fond similaire à celui présentée au Chapitre 1 section 1.3.3.1 est appliqué pour extraire les personnes de l'image. La Figure 2.2.6 contient un exemple de résultat obtenu. Les pixels ajoutés sur la tête à l'étape précédente font bien partie du masque de mouvement puisque leurs valeurs sont très différentes de celles modélisant le fond. L'image finalement obtenue peut ensuite être manipulée par les modules suivants de la chaîne de traitement.



(a) (b) (c)

FIGURE 2.2.6 – En (a) l'image initiale, en (b) le fond moyen de la scène et en (c) l'image de mouvement.

2.3 Détection et suivi de personnes dans des images de profondeur

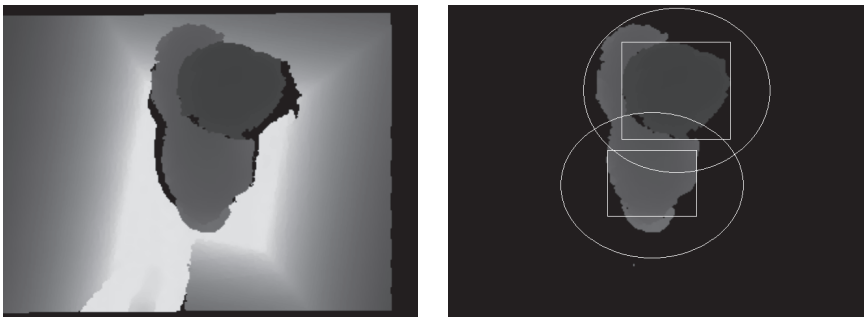
Cette section concerne les étapes de détection et de suivi des têtes dans la séquence d'images. Nous verrons que le suivi des personnes d'une image à l'autre apporte une information importante permettant d'améliorer les résultats de détection et d'éviter des non détections. L'algorithme de segmentation est détaillé section 2.3.1 et celui de suivi section 2.3.2.

2.3.1 Segmentation des têtes

La première étape de l'algorithme consiste à segmenter les têtes des individus dans l'image. La méthode présentée dans [86] est modifiée pour prendre en compte les cas où la tête est située au bord de l'image et non entièrement visible. L'algorithme décrit dans la suite est appliqué à l'image de sortie du module décrit section 2.2.

Tout d'abord, des régions appelées *têtes candidates* sont recherchées dans l'image. Pour cela, le pixel de profondeur minimale p_{min} est sélectionné comme point de départ d'un algorithme de croissance de régions avec pour paramètre $P = 100$ mm. Un nouveau pixel p_{ij} voisin du pixel courant est ajouté à la région s'il satisfait la condition $p_{ij} < p_{min}+P$. La valeur du paramètre P est légèrement inférieure à la hauteur attendue d'une tête. La composante connexe résultante est conservée pour la suite si sa taille est acceptable.

Une première modification est apportée à la méthode initiale. Un masque prédéfini est appliqué sur le centre de la région pour ne conserver que les pixels y appartenant. Nous choisissons d'utiliser un masque circulaire de rayon r pixels. La plupart des pixels appartenant au corps de la personne sont donc écartés. Le choix du paramètre r dépend du système et est fixé empiriquement. L'application du masque circulaire diminue le nombre de régions candidates qui seront analysées dans la suite. Si le rayon est bien choisi, le risque de fausse détection peut être réduit. Le processus est répété itérativement tant que des régions candidates sont trouvées. Une région candidate et son masque sont représentés Figure 2.3.1.



(a) (b)

FIGURE 2.3.1 – En (a) l'image initiale et en (b) les régions candidates et leurs masques.

Chaque tête candidate est ensuite évaluée par les trois descripteurs présentés dans [86] :

- l'*aire projetée* s_1,
- le *coefficient de sphéricité* s_2,
- le *ratio de test* s_3.

Ces trois quantités sont calculées pour vérifier si une région donnée est ou non une tête. L'*aire projetée* est une mesure de l'aire de la projection de la tête sur le sol. Pour effectuer ce calcul, les pixels de la région sont convertis en points 3D. La projection est calculée en supposant que l'axe optique est orthogonal au sol. Le *coefficient de sphéricité* quantifie le caractère circulaire de la région et est défini comme étant le pourcentage de pixels dont la distance au centre de la région est inférieure à $\sqrt{A\,/\,\pi}$, où A est le nombre de pixels composant la région. Enfin, le *ratio de test* permet de vérifier si la région constitue un maxima local en comparaison des zones adjacentes dans l'image. Ces scores sont ramenés dans l'intervalle $[0, 1]$ grâce à une fonction logistique l de paramètres $c \in \mathbb{R}$ et $T \in \mathbb{R}$ de la forme

$$l(x) = \frac{1}{1 + e^{-c(x-T)}} \tag{2.3.1}$$

et combinés pour obtenir le score final s de la région calculé comme suit

$$s = l(s_1)\, l(s_2)\, l(s_3). \tag{2.3.2}$$

Les paramètres T et c sont respectivement des paramètres de position et de forme de la fonction logistique, choisis expérimentalement. Une tête candidate est finalement détectée comme une tête si son score est suffisamment élevé, c'est à dire

$$s \geq s_{min} \tag{2.3.3}$$

où s_{min} est le score minimal qu'une tête doit posséder. Comme recommandé dans [86], nous choisissons $s_{min} = 0.7$.

Toutefois, comme mentionné dans [86], une limite de la méthode est que les têtes doivent être entièrement visibles sur l'image pour être détectées. En effet, une tête incomplète conduira à un score faible. Le capteur étant situé assez bas dans notre système, les têtes des personnes ne sont pas toujours entièrement visibles. Pour résoudre ce problème, nous proposons de choisir de manière adaptative des paramètres c' et T' comme suit

$$\begin{cases} c' = c \\ T' = T \times K \end{cases} \tag{2.3.4}$$

pour le calcul de s_1 uniquement et lorsque la région est située au bord de l'image. Afin de prendre en compte le fait que la tête est tronquée, on s'intéresse au ratio entre l'aire de la partie visible de la tête et l'aire de la tête complète. Pour cela, on suppose que la tête est un cercle de diamètre a. Comme cela est illustré Figure 2.3.2, d et a désignent respectivement les longueurs minimales et maximales de la boîte englobante de la tête tronquée dans l'image. La partie visible de la tête est approchée par une demi-ellipse de demi-axes d et l. La longueur l vaut $\sqrt{d(a-d)}$ et par conséquent l'aire de la partie visible vaut $\pi\, d\sqrt{d(a-d)}/2$. Le ratio recherché a donc finalement pour expression

$$K = \frac{2\,d\,\sqrt{d(a-d)}}{a^2} \tag{2.3.5}$$

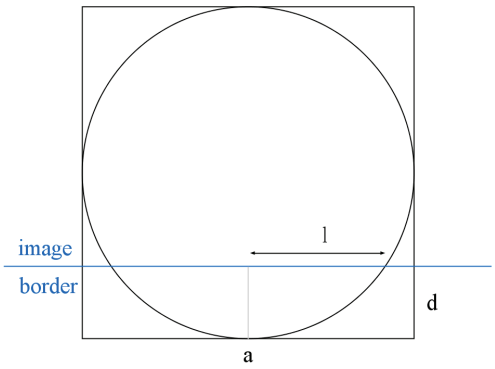
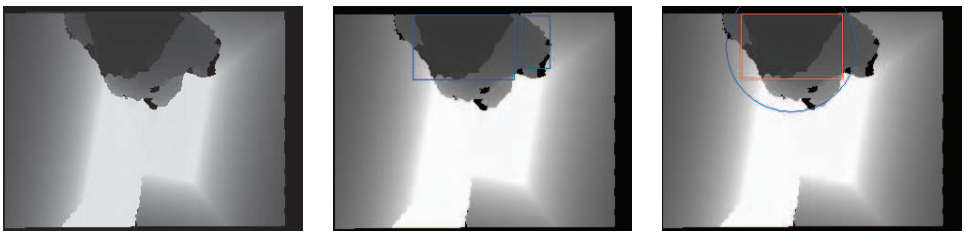


FIGURE 2.3.2 – Les têtes sont supposées avoir une forme circulaire en vue de dessus. Lorsqu'une tête est tronquée par un bord de l'image, la partie visible est proche d'une demi-ellipse. K est donc choisi comme étant le ratio entre les aires de la partie visible (demi-ellipse) et de la forme complète (cercle)

Le choix des paramètres c' et T' produira des scores élevés même si la tête est tronquée par un bord de l'image. La Figure 2.3.3 illustre la différence entre les scores calculés par l'ancienne et la nouvelle méthode. Dans cet exemple, la tête n'est pas détectée avec les coefficients de base alors qu'elle l'est avec les coefficients modifiés. Les rectangles bleus sont les têtes candidates dont les scores sont trop faibles pour être détectées comme des têtes. Les rectangles rouges représentent les têtes détectées. Les cercles bleus sont les masques circulaires. Les scores de la tête sont $s_1 = 0.69$, $s_2 = 0.96$, $s_3 = 0.99$ et $s = 0.66$ pour la non détection et $s_1 = 0.97$, $s_2 = 0.96$, $s_3 = 0.99$ et $s = 0.92$ pour la détection. Les nouveaux paramètres donnent dans ce cas un score suffisamment élevé pour détecter la tête tronquée.



(a) (b) (c)

FIGURE 2.3.3 – En (a) l'image traitée, en (b) le résultat de la détection avec les coefficients de base et en (c) avec les coefficients modifiés.

2.3.2 Suivi des personnes

La seconde étape de l'algorithme est constituée d'un algorithme de suivi des têtes. La position et les descripteurs des têtes détectées dans la séquence sont connus. Le module de suivi permettra, nous le verrons, d'améliorer les résultats de la détection.

Soit $F_i = \{H_1, H_2, \ldots\}$ l'ensemble des têtes détectées dans l'image numéro i. Chaque objet est accompagné de son score $s(H_j)$ calculé lors de la phase de détection. Le but du suivi est de mettre en correspondance les éléments des ensembles F_i et F_{i+1} pour connaître le déplacement de chaque objet. À cause des non et fausses détections, il peut arriver que les deux ensembles ne contiennent pas le même nombre d'éléments. La connaissance du chemin parcouru par les objets détectés apporte une information pouvant aider à corriger les résultats de la détection. Le principe est illustré Figure 2.3.4 par un schéma tiré de [86]. Sur cet exemple, la non détection ponctuelle de l'image $i + 1$ peut être corrigée si l'on connait le déplacement antérieur des personnes.

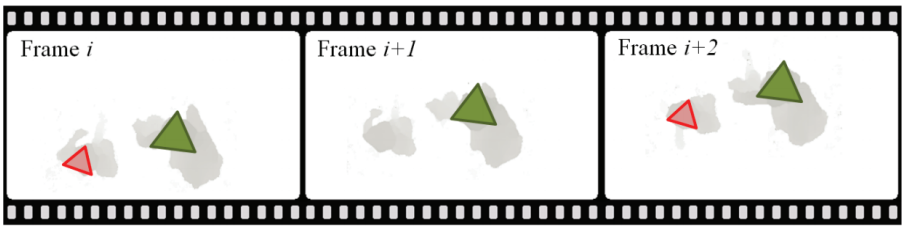


FIGURE 2.3.4 – Les deux objets ont été détectés dans les images i et $i + 2$, mais pas $i + 1$. L'analyse du déplacement des objets pourrait permettre de savoir qu'un objet est manquant dans l'image i.

L'algorithme de suivi proposé dans [86] est basé sur l'introduction d'*objets virtuels*. Pour chaque objet détecté dans l'image précédente i, un objet virtuel est ajouté dans l'image $i + 1$. Sa position est prédite par un modèle à vitesse constante à partir du déplacement constaté dans les dernières images. Son score est atténué par un facteur p_f afin qu'il soit inférieur à celui de l'objet initial. Les objets virtuels sont ajoutés à la position où devraient normalement se trouver les objets si leurs déplacements sont constants. Chaque élément de F_i pourra donc être mis en correspondance avec un élément de F_{i+1}. Un objet virtuel ne peut être associé qu'avec l'objet l'ayant généré. Dans le cas d'une non détection à l'image $i + 1$, l'objet virtuel remplacera l'objet qui n'a pas été détecté. Dans le cas où la détection s'est bien déroulée, l'objet virtuel ne sera pas conservé par l'algorithme de mise en correspondance puisque son score est inférieur à celui de l'objet original. Le schéma de la Figure 2.3.5 illustre le principe décrit ci-dessus.

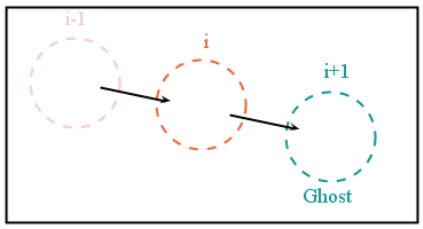


FIGURE 2.3.5 – Le déplacement de l'objet entre les images $i - 1$ et i est utilisé pour positionner le nouvel élément dans l'image $i + 1$.

Le problème de mise en correspondance des objets détectés dans les deux images (et des objets virtuels) est modélisé par un graphe biparti. Les sommets représentent les objets à associer provenant des deux images et les arêtes les possibles associations. Une arête est ajoutée entre deux sommets du graphe si la distance Euclidienne (en pixels) séparant les deux objets est inférieure à la limite d_{maxi}. Le poids w_{kl} de l'arête reliant les objets $H_k \in F_i$ et $H_l \in F_{i+1}$ vaut

$$w_{kl} = d_2(H_k, H_l) + (1 - s(H_k)\, s(H_l)\, sim(H_k, H_l))\, pen \tag{2.3.6}$$

où le premier terme représente la distance Euclidienne en pixels entre le centre de l'objet H_l et la position prédite de l'objet H_k, s les scores calculés lors de la phase de détection, pen un coefficient de pénalité et sim une fonction de similarité. La similarité entre deux objets est quantifiée par la distance Euclidienne entre les vecteurs descripteurs des objets, formés de l'aire en pixels et la profondeur moyenne des régions formant les objets. Le paramètre pen permet de pondérer les termes composant le poids de l'arête. Le graphe biparti modélisant le problème de mise en correspondance est représenté Figure 2.3.6. Le problème consiste à choisir les arêtes de telle sorte que tous les éléments de F_i soient associés à un élément de F_{i+1} et que la somme de leurs poids soit minimale. Un algorithme Hongrois ([116]) est appliqué pour résoudre ce problème. Les objets seront donc associés de telle sorte que la similarité globale soit la plus élevée possible. Le graphe étant de taille faible, la coût calculatoire l'est également. L'algorithme est résumé par la Figure 2.3.7 tirée de [86].

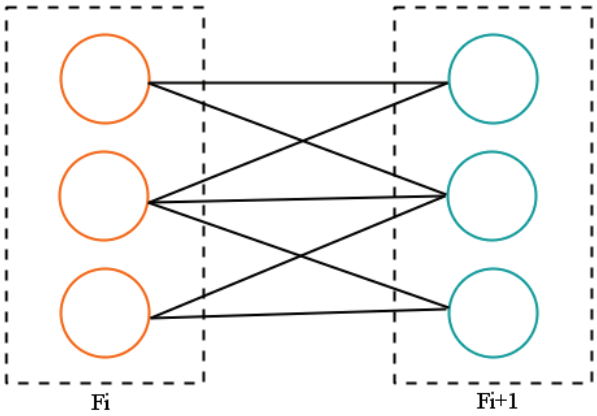


FIGURE 2.3.6 – Les objets détectés dans les images i et $i + 1$ forment deux ensembles de sommets. Les associations possibles sont représentées par des arêtes allant d'un ensemble à l'autre. Le graphe est biparti puisqu'aucune arête ne relie les sommets d'un même ensemble.

Algorithm 2: Tracking

Input: set of objects in frame F_i and frame F_{i+1}
Output: matching between objects in F_i and F_{i+1}

/* create ghosts */
foreach $H \in F_i$ **do**
  $(x_H, y_H) \leftarrow$ predicted position of H in F_{i+1};
  $H_G \leftarrow$ copy of H at position (x_H, y_H);
  $p(H_G) \leftarrow p_f \cdot p(H)$;
  if $p(H_G) \geq p_{min}$ **then** add H_G to F_{i+1};
  ;
end

create a weighted bipartite graph $G = (F_i \cup F_{i+1}, E)$;
foreach $H \in F_i$, $H' \in F_{i+1}$ **do**
  if $dist(H, H') > max_{dist}$ **then** continue;
  ;
  if H' *is a ghost* $\wedge$ H' *is not a ghost of* H **then** continue;
  ;
  create edge $e = (H, H')$;
  $(x_H, y_H) \leftarrow$ predicted position of H in F_{i+1};
  $w(e) \leftarrow dist((x_H, y_H), H') + (1 - p(H) \cdot p(H') \cdot similarity(H, H')) \cdot Penalty$;
end

compute minimum-weighted bipartite matching in G;
remove from F_{i+1} all unmatched ghosts;

FIGURE 2.3.7 – Algorithme de suivi tiré de [86].

2.4 Expérimentations

Dans cette partie, l'algorithme de détection et de suivi est évalué sur des séquences d'images de profondeur acquises avec un capteur 3D à lumière structurée et représentant une ou deux personnes. Le but est de montrer que la méthode donne de bons résultats dans le cas de personnes situées au bord de l'image et que la stratégie de reconstruction adoptée ne perturbe pas la détection.

La méthode est tout d'abord évaluée sur le même jeu de données que dans [204, 86]. Il est composé de deux sous-ensembles d'images distincts. Le premier est composé de 2834 images avec 4541 têtes et le second de 1500 images avec 1553 têtes. Ces premières séquences d'images nous permettent de valider la méthode de détection et de suivi de têtes. Cependant, bien que les images de test soient les mêmes, les résultats attendus ne sont pas identiques à ceux de [204, 86] puisque nous cherchons à détecter les têtes complètes et tronquées. Les images de ce premier jeu de données ne contiennent pas de têtes entièrement masquées à cause de leur trop grande proximité avec le capteur 3D. L'algorithme de restauration présenté en début de chapitre ne sera donc pas utilisé.

Les performances de l'algorithme sont évaluées à travers deux mesures classiques : le taux de recall (proportion de personnes bien détectées) et la précision (proportion de détections qui sont des personnes). Les résultats expérimentaux sont présentés Table 2.4.1. Des images d'illustration sont données Figure 2.4.1. L'évaluation montre que la méthode fonctionne bien. Les modifications apportées à l'algorithme de base améliorent les résultats dans le cas de têtes tronquées. Le cas le plus problématique concerne les personnes proches de tailles assez différentes, comme par exemple un adulte accompagné d'un enfant. Dans ce cas particulier, il est difficile de dissocier la tête de l'enfant du corps de l'adulte.

TABLE 2.4.1 – Comparaison des résultats expérimentaux

Données	Ensemble A1		Ensemble A2	
Méthode	Recall	Précision	Recall	Précision
Waterfilling ([204])	98,82%	98,83%	99,47%	99,57%
Croissance de régions ([86])	99,78%	99,89%	99,23%	99,62%
Algorithme proposé	99,82%	99,89%	99,46%	99,65%

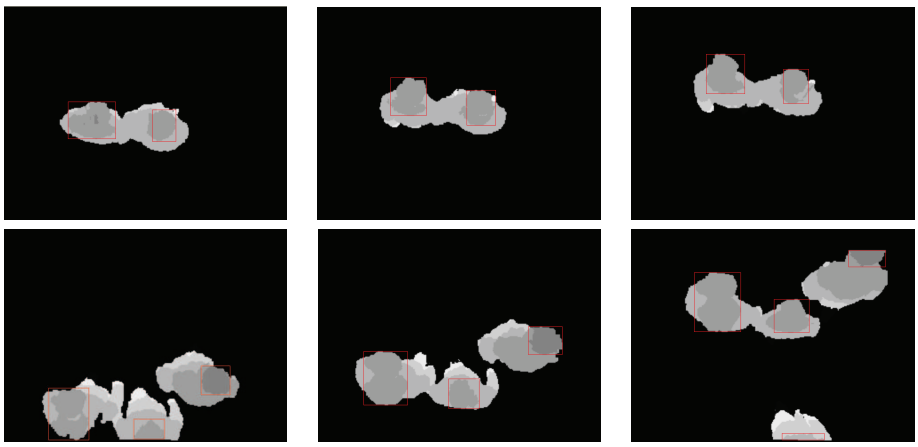


FIGURE 2.4.1 – Résultats obtenus sur deux séquences d'images.

L'algorithme est ensuite expérimenté sur quatre séquences d'images acquises par le système présenté en section 2.2. Pour observer l'influence de la méthode de reconstruction spécifique proposée, des personnes de petite et moyenne taille ont été sollicitées. L'ensemble des séquences de test totalise 960 images de profondeur. Nous avons choisi de travailler sur des images de dimensions 320×240. Nous rappelons que l'algorithme sur lequel nous nous sommes basés était initialement conçu pour analyser des flux de personnes depuis un point de vue beaucoup plus élevé que celui de notre système.

Les paramètres de l'algorithme de reconstruction des régions indéterminées ont été fixés après quelques tests à $d_0 = 500$ mm et $\epsilon = 1$ mm. Les composantes connexes de pixels indéterminés de dimensions supérieures à 50×50 pixels sont considérées comme étant un objet proche du capteur nécessitant l'application de l'algorithme spécifique de restauration. Les dimensions minimales et maximales d'une région pouvant s'apparenter à une tête ont été fixées à 30 et 120 pixels. Les valeurs minimales sont assez faibles pour conserver les têtes tronquées par le bord de l'image. Les valeurs maximales sont choisies assez grandes puisque le positionnement bas du capteur implique que les têtes sont de taille importante sur l'image. Le paramètre de la méthode de croissance de régions a été fixé à $P = 100$ mm. Le score minimal pour qu'un objet virtuel soit conservé est de 0.5 et le coefficient de pénalité est de 0.5. Deux têtes détectées dans deux images consécutives sont reliées par une arête si la distance les séparant (dans l'image) est inférieure à $d_{maxi} = 75$ pixels. Enfin, les paramètres c et T ont été respectivement fixés à 300 et 0.00225 pour s_1, 0.4 et 75 pour s_2 et 0.66 et 75 pour s_3. Tous ces paramètres ont été choisis empiriquement après avoir effectué un certain nombre de tests. Ils sont en partie dépendant de la configuration du système (position et orientation du capteur). La méthode est assez sensible au choix des paramètres (surtout c et T).

Les résultats obtenus sont présentés Table 2.4.2. Les taux de détection révèlent que la méthode est efficace sur nos données. On conclut donc que la technique de restauration ne perturbe pas la détection. Le bon fonctionnement de la méthode est illustré Figures 2.4.2 et 2.4.3 où des personnes tronquées ont été correctement détectées.

TABLE 2.4.2 – Résultats expérimentaux

Méthode	Recall	Précision
Méthode proposée	98,96%	99,06%

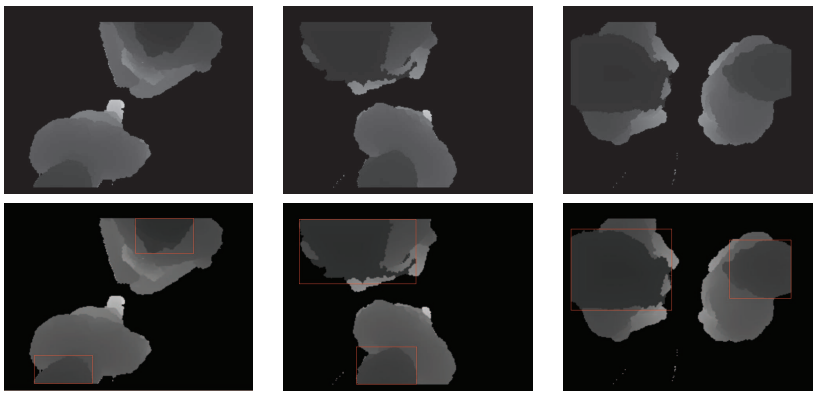


FIGURE 2.4.2 – Résultats expérimentaux. Les têtes détectées sont encadrées en rouge.

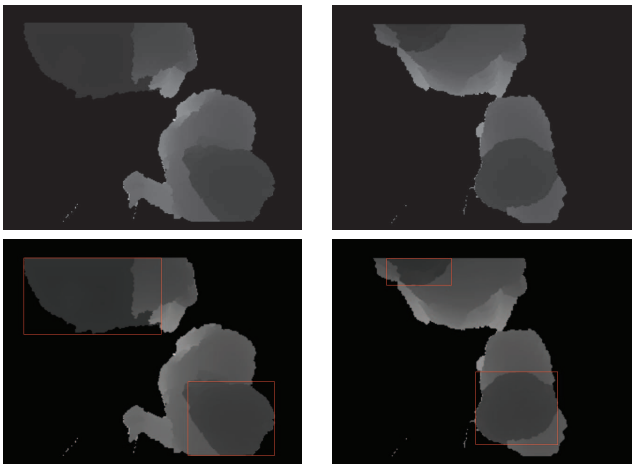
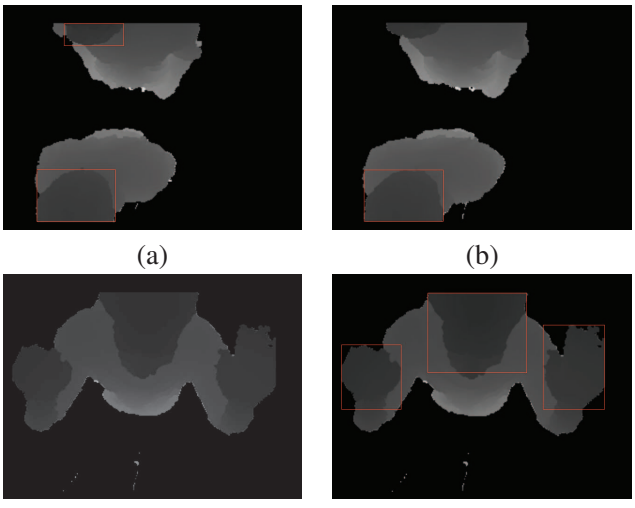


FIGURE 2.4.3 – Résultats expérimentaux. Les têtes détectées sont encadrées en rouge.

L'avantage principal de cette méthode est qu'elle gère le cas de têtes tronquées par le bord de l'image. Toutefois, comme cela est illustré Figure 2.4.4-(a, b), une tête tronquée ne sera pas détectée si sa partie visible sur l'image est trop petite. Sur l'exemple de la figure, la tête incomplète a été détectée sur la première image ($s = 0.81$) mais pas sur la seconde ($s = 0.78$). Les expérimentations ont également montré que la méthode a tendance à détecter les bras et les mains levés, mais cette situation arrive rarement en pratique. Ce dernier point est illustré Figure 2.4.4-(c, d).



(a) (b)

(c) (d)

FIGURE 2.4.4 – Cas de non fonctionnement de l'algorithme. En (a) et (b) les têtes détectées sur deux images consécutives. En (c) et en (d), une image de profondeur et les têtes détectées.

Les algorithmes ont été expérimentés sur une machine équipée d'un processeur Intel core i7 avec 8 Go de RAM. La méthode, programmée en C++, peut traiter un flux vidéo à environ 20 images par secondes. La contrainte de traitement en temps-réel est donc pleinement satisfaite.

2.5 Conclusion

Dans ce chapitre, un algorithme de détection et de suivi de têtes a été modifié et adapté pour fonctionner dans notre configuration particulière. Une stratégie de restauration d'images de profondeur a été développée. Cette méthode spécifique à notre système est très rapide et permet de reconstruire des objets partiellement masqués. La méthode de détection a été adaptée pour fonctionner dans le cas de personnes tronquées par les bords de l'image sans pour autant augmenter la complexité de l'algorithme.

Les expérimentations ont prouvé l'efficacité de la méthode. Les taux de détection sont élevés et l'algorithme fonctionne en temps-réel. Une idée d'amélioration de la méthode pourrait consister à fusionner les informations provenant de plusieurs capteurs afin de couvrir des zones plus larges. Ceci permettrait de réduire le nombre de cas où les têtes sont tronquées par les bords de l'image.

Deuxième partie

Segmentation et modélisation d'un nuage de points 3D à l'aide de nouveaux modèles de mélange pour la détection de têtes

La détection de personnes dans les images est un problème récurrent en vidéo-surveillance. Cette problématique intervient dans les applications liées au comptage, à la détection d'intrusions ou encore à l'accès sécurisé à une zone. Du point de vue de la sécurité, en plus de la détection des individus, toute information supplémentaire caractérisant les personnes est intéressante, comme par exemple la taille. Les capteurs 3D sont bien adaptés à cette problématique puisqu'ils fournissent des informations sur la forme et donc sur les dimensions des éléments composant la scène.

Le but de cette partie est donc de proposer une nouvelle méthode de modélisation et de détection de têtes dans un nuage de points 3D représentant un groupe de personnes. Nous choisissons pour cela de travailler directement sur le nuage de points 3D représentant la scène. On souhaite modéliser chacune des têtes par une forme géométrique simple la caractérisant. Le système considéré est composé d'un seul capteur 3D positionné en position verticale. L'algorithme doit fonctionner en quasi temps-réel, de manière autonome et sans intervention extérieure. La notion de quasi temps-réel signifie que quasiment toutes les images retournées par le capteur peuvent être traitées.

Pour atteindre ces objectifs, nous adoptons une approche probabiliste et statistique. Le problème soulevé nous conduit à introduire un nouveau modèle de mélange sphérique Chapitre 3 dont nous proposerons une méthode d'estimation des différents paramètres. Le nuage de points est modélisé par un ensemble de sphères. La méthode proposée est ensuite appliquée au problème de la détection et de la modélisation de têtes à l'aide d'un seul capteur 3D au Chapitre 4. Des règles heuristiques simples sont définies pour détecter les têtes. Des stratégies de choix du nombre de composantes et de réduction du temps de calcul seront proposées et expérimentées. Enfin, pour montrer que le travail réalisé offre des perspectives de recherche intéressantes, nous explorons dans le Chapitre 5 la généralisation du modèle sphérique au cas ellipsoïdal.

Chapitre 3

Segmentation et modélisation d'un nuage de points 3D par un modèle de mélange sphérique

Sommaire

3.1 Introduction

Dans ce chapitre, nous nous intéressons au problème de segmentation et de modélisation d'un nuage de points 3D. Les données manipulées proviennent d'un capteur 3D similaire à ceux présentés au Chapitre 1. Notre but est de séparer les différents objets de la scène et de les modéliser par un ensemble de primitives géométriques simples. Nous chercherons ensuite dans les chapitres suivants de la thèse à appliquer la méthode proposée à des données représentant un groupe de personnes afin de détecter leurs têtes.

Des algorithmes de segmentation et de classification appliqués à des nuages de points 3D ont été étudiés au Chapitre 1 section 1.3. Nous en avons conclu que les méthodes existantes de recherche de formes géométriques dans un ensemble de points (transformée de Hough, RANSAC) sont beaucoup trop coûteuses. Notre application pratique requiert une complexité satisfaisant au maximum la contrainte de temps-réel. De plus, en pratique la taille des jeux de données manipulés est de l'ordre de quelques dizaines de milliers de points selon la complexité de la scène. Pour ces raisons, nous travaillons à la mise au point d'une autre approche de résolution.

Ce constat nous amène à proposer un nouveau modèle probabiliste basé sur les modèles de mélange (voir par exemple [128]). La forme des objets est considérée de par la distribution statistique des points dans l'espace. La répartition des points est analysée et détermine une classification souple des données. Les modèles de mélange sont des modèles probabilistes comportant plusieurs composantes chacune modélisée par une densité de probabilité. Nous allons utiliser ces modèles pour à la fois segmenter et modéliser nos données. Chaque classe est modélisée par une forme géométrique caractérisée par un ensemble de paramètres. L'idée est donc de définir une densité de probabilité capable de décrire de manière probabiliste une forme géométrique.

Une caractéristique importante du type de modélisation retenu est que les points sont affectés aux différentes classes avec une certaine probabilité. On parle alors d'affectation *soft* contrairement à une affectation *hard* dans laquelle un point est associé à une seule et unique classe. Une autre caractéristique intéressante des modèles de mélange est leur capacité à modéliser des distributions proches l'une de l'autre ou se chevauchant. Cet aspect sera très utile dans notre cas de figure puisqu'on peut alors espérer segmenter et modéliser correctement deux objets physiquement proches ou en contact.

Les modèles de mélange, gaussiens en particulier, ont été beaucoup utilisés en statistiques notamment pour des problèmes de classification ([128]). Ils ont été adaptés dans le domaine de la vision par ordinateur par exemple pour des problèmes de soustraction de fond ([76]), de segmentation d'images ([203, 38]), de recalage de nuages de points 3D ([108]) et de détection de nouveaux éléments dans des environnements 3D ([147]). Ces modèles ont déjà permis par le passé d'apporter des avancées importantes dans le domaine du traitement d'images. Nous pensons notamment aux travaux présentés dans [76] traitant le problème de la soustraction de fond par un mélange de distributions gaussiennes.

Dans [147], les nuages de points 3D sont directement modélisés par un mélange de densités gaussiennes et des primitives 3D sont ensuite extraites des classes obtenues. Malheureusement, l'utilisation de distributions gaussiennes n'est pas adapté à la modélisation de points situés sur des surfaces. Pour répondre à cette inadéquation, nous introduisons une nouvelle densité de probabilité décrivant des points répartis autour d'une surface avec des fluctuations dans la direction normale. La nouvelle densité est conçue dans l'optique de décrire correctement les données provenant du capteur 3D. Nous avons en effet souligné Chapitre 1 que ce type de capteur fournissait un ensemble de points échantillonnés sur la partie visible de la surface des objets. Dans ce chapitre, nous nous focalisons sur le

cas de la sphère, qui nous le verrons, convient à notre application pratique cherchant à modéliser des têtes.

Nous prouverons l'appartenance de la nouvelle distribution à la famille des distributions elliptiques et nous en déduirons des propriétés utiles pour réaliser nos calculs. La densité sphérique sera ensuite intégrée dans un modèle de mélange dont nous proposerons une méthode d'estimation des paramètres inconnus. Nous étudierons ensuite les propriétés des estimateurs. Le comportement de la méthode d'estimation sera expérimentée sur des données simulées puis sur des jeux de données réels. Enfin, nous observerons avec une attention particulière les résultats obtenus lorsque les points sont non-uniformément répartis sur la surface des objets et que la quantité de données manquantes est élevée. L'application pratique visée n'utilise qu'un seul capteur 3D et seule une partie des objets sera visible sur les images. Ce chapitre est focalisé sur l'étude du nouveau modèle et l'estimation de ses paramètres. Les problématiques liées à la mise en œuvre de l'algorithme en conditions réelles seront abordées ultérieurement.

Ce chapitre est organisé de la manière suivante. Nous commencerons par présenter de manière générale la famille des distributions elliptiques en section 3.2. Nous introduirons ensuite le nouveau modèle de mélange sphérique en section 3.3. Le problème d'estimation des paramètres inconnus sera abordé en section 3.4. Les performances de la méthode proposée seront évaluées sur des données simulées en section 3.5 et sur des données réelles en section 3.6. Nous conclurons le chapitre en section 3.7 et détaillerons une partie des calculs en Annexes A, B, C, D et E.

3.2 Distributions elliptiques

Les distributions normales multidimensionnelles ont été beaucoup utilisées pour modéliser statistiquement des données issues de nombreux phénomènes. Dans cette partie, nous nous intéressons à la famille des distributions dites elliptiques, incluant notamment les distributions Gaussiennes, de Cauchy, de Laplace, Exponentielles, et de nombreuses autres. Elles ont été introduite en 1970 dans [111] puis notamment étudiées dans [39, 82, 81, 94, 89]. Une vue d'ensemble de leurs principales caractérisations et propriétés a été effectuée dans [88]. Il s'agit d'une vaste famille de distributions à partir desquelles il est possible de construire des modèles statistiques. Leurs densités de probabilité ont la particularité d'être constantes sur des ellipsoïdes, d'où leur dénomination.

Nous rappelons les principales définitions et caractérisations des distributions elliptiques en 3.2.1 et donnons leurs principales propriétés en 3.2.2.

3.2.1 Définitions et caractérisations

Les distributions elliptiques admettent plusieurs définitions et caractérisations que nous exposons dans cette partie. Nous commençons par nous intéresser aux distributions de type sphérique.

Définition 3.2.1 (Distribution sphérique)**.** *Un vecteur aléatoire Y de dimension d a une distribution sphérique si*

$$QY \overset{\mathcal{L}}{=} Y \tag{3.2.1}$$

pour toute matrice orthogonale Q de taille $d \times d$. Le symbole $\overset{\mathcal{L}}{=}$ désigne l'égalité de deux distributions.

Les distributions sphériques sont un cas particulier des distributions elliptiques. Elles sont parfois appelées distributions radiales ou isotropiques. Elles ont été introduites pour la première fois dans [111]. D'après la Définition 3.2.1, de telles distributions sont invariantes par rotation. Si Y admet une densité de probabilité g, alors il est clair que sa valeur au point $y \in \mathbb{R}^d$ ne dépend que de $\|y\|$. La quantité $\|y\|$ est conservée par rotation car il s'agit d'une transformation isométrique.

Par exemple, la loi Normale $\mathcal{N}\left(0_{\mathbb{R}^d}, \sigma^2 I_d\right)$ de densité $g(y) = \dfrac{1}{(2\pi)^{d/2}\,\sigma^d} \exp\left(-\dfrac{\|y\|^2}{2\sigma^2}\right)$ appartient à la famille des distributions sphériques. Les propriétés des distributions sphériques se déduisent de celles des distributions elliptiques introduites ci-dessous.

Définition 3.2.2 (Distribution elliptique)**.** *Un vecteur aléatoire X de dimension d a une distribution elliptique de paramètres $\mu \in \mathbb{R}^d$ et $\Sigma \in \mathbb{R}^{d\times d}$ symétrique définie positive si sa densité de probabilité (quand elle existe) est de la forme*

$$|\,\Sigma\,|^{-1/2}\; g\left((x-\mu)^T \Sigma^{-1} (x-\mu)\right) \tag{3.2.2}$$

où g est une fonction positive appelée densité génératrice. On note $X \sim EC_d(\mu, \Sigma, g)$.

Les paramètres μ et Σ sont appelés respectivement paramètres de position et d'échelle. En particulier, notons que d'après la Définition 3.2.1 la distribution $EC_d(0_{\mathbb{R}^d}, I_d, g)$ est une distribution sphérique puisque sa densité de probabilité ne dépend plus que du terme $x^T x$. La loi Normale appartient à la famille des distributions elliptiques pour le choix $g(t) = \dfrac{1}{(2\pi)^{d/2}} \exp(-t/2)$. Nous avons choisi dans cette définition d'intégrer les constantes de normalisation dans la fonction g.

Notons que cette représentation n'est pas unique ([89]) puisque si $X \sim EC_d(\mu, \Sigma, g)$, alors $X \sim EC_d(\mu^*, \Sigma^*, g^*)$ si et seulement si $\exists\, a, b > 0$ tels que $\mu^* = \mu$, $\Sigma^* = a\,\Sigma$ et $g^*(t) = b\,g(a\,t)$.

La définition suivante, basée sur la fonction caractéristique, donne une caractérisation supplémentaire des distributions elliptiques.

Définition 3.2.3 (Générateur caractéristique)**.** *Un vecteur aléatoire X de dimension d a une distribution elliptique de paramètres $\mu \in \mathbb{R}^d$ et $\Sigma \in \mathbb{R}^{d\times d}$ symétrique définie positive si sa fonction caractéristique est de la forme*

$$\phi(t) = \exp\left(it^T\mu\right)\psi\left(t^T\Sigma t\right),\ t \in \mathbb{R}^d \tag{3.2.3}$$

où ψ est une fonction scalaire appelée générateur caractéristique. On note $X \sim EC_d(\mu, \Sigma, \psi)$.

Par exemple, le générateur caractéristique de la loi Normale multidimensionnelle est $\psi(u) = \exp(-u/2)$. Cette représentation n'est pas unique puisque si on considère $X \sim EC_d(\mu, \Sigma, \psi)$, alors $\forall c > 0$, $X \sim EC_d(\mu, c\Sigma, \psi\left(\frac{\cdot}{c}\right))$. Le lien entre les distributions sphériques et elliptiques peut être établi plus directement par la définition suivante.

Définition 3.2.4 (Distribution elliptique)**.** *Soit Y un vecteur aléatoire de $\mathbb{R}^d$ ayant une distribution sphérique. Alors le vecteur*

$$X = \mu + CY \tag{3.2.4}$$

admet une distribution elliptique $EC_d(\mu, \Sigma)$ où $\Sigma = C^TC$ avec C une matrice inversible de taille $d \times d$ et $\mu \in \mathbb{R}^d$.

Une distribution elliptique peut être obtenue par transformation affine d'une distribution sphérique. Intuitivement, on comprend que la forme sphérique de Y est déformée par la matrice C avant d'être translatée de μ, ce qui donne le caractère elliptique au vecteur aléatoire X.

3.2.2 Propriétés

Les variables aléatoires elliptiques étudiées dans cette partie admettent une décomposition polaire. Cette propriété est très utile pour effectuer des calculs et constitue dans nos travaux la propriété que nous utiliserons le plus. Le théorème suivant ([39]) permet de caractériser les distributions sphériques et elliptiques.

Théorème 3.2.1 (Représentation stochastique)**.** *Soient Y une variable aléatoire de distribution sphérique et $X \sim EC_d(\mu, \Sigma, g)$. Ces vecteurs aléatoires de dimension d peuvent s'écrire*

$$Y = WU^{(d)} \tag{3.2.5}$$

$$X = \mu + WAU^{(d)} \tag{3.2.6}$$

où W est une variable aléatoire positive et indépendante de $U^{(d)}$ uniformément répartie sur la sphère unité de $\mathbb{R}^d$ et $AA^T = \Sigma$. De plus, la densité de probabilité de W est de la forme

$$h(t) = \frac{2\pi^{d/2}}{\Gamma(d/2)}\,t^{d-1}g\left(t^2\right). \tag{3.2.7}$$

Notons que le théorème nous donne une équivalence entre le type de distribution (sphérique ou elliptique) et l'écriture polaire. Cette propriété permet de calculer les premiers moments de X ([81]). Il est clair que par définition

$$\mathbb{E}[U] = 0 \tag{3.2.8}$$

$$\mathbb{V}\text{ar}[U] = \frac{1}{d} I_d \tag{3.2.9}$$

et donc comme les variables W et $U^{(d)}$ sont indépendantes, on obtient les relations

$$\mathbb{E}[X] = \mu \tag{3.2.10}$$

$$\mathbb{V}\text{ar}[X] = \frac{1}{d} \mathbb{E}\left[W^2\right] I_d. \tag{3.2.11}$$

La quantité $\mathbb{E}[X]$ n'existe que si $\mathbb{E}[W]$ existe et de même $\mathbb{V}\text{ar}[X]$ n'existe que si $\mathbb{V}\text{ar}\left[W^2\right]$ existe. Enfin, le lien entre les fonctions g et h est donné par l'équation (3.2.7) et par la relation

$$g(t) = t^{(1-d)/2} h(t^{1/2}). \tag{3.2.12}$$

Si on pose $Z = \Sigma^{-1/2}(X - \mu)$, alors $\|Z\| = \sqrt{(X-\mu)^T \Sigma^{-1} (X-\mu)}$ est distribuée comme $\left\|WU^{(d)}\right\| = W$. La variable $\dfrac{Y}{\|Y\|}$ est alors uniformément distribuée sur la sphère unité de $\mathbb{R}^d$.

3.3 Construction d'un modèle adapté aux données provenant du capteur 3D

Les modèles gaussiens ont été beaucoup utilisés pour modéliser des données. Ils décrivent des distributions dont les échantillons sont concentrés autour d'une moyenne. Toutefois, la distribution normale n'est pas adaptée aux données que nous devons traiter. Les points 3D calculés à partir de l'image de profondeur renvoyée par le capteur sont situés sur la surface des objets et non pas à l'intérieur. Les points ne se concentrent pas autour d'un centre mais autour d'une surface.

Cette observation motive donc la construction d'une nouvelle distribution adaptée à notre type de donnée. Cette section est dédiée à la construction et à l'étude d'une nouvelle densité de probabilité centrée sur une surface. Puisque nous cherchons à modéliser des têtes, nous nous concentrerons dans ce chapitre sur le cas de la sphère. Nous exposerons le cheminement qui nous a permis de définir la densité proposée en section 3.3.1. Nous présenterons également quelques propriétés de la nouvelle distribution en section 3.3.2 avant de l'intégrer dans un modèle de mélange en section 3.3.3.

3.3.1 Introduction d'une nouvelle densité sphérique

La densité de probabilité d'un vecteur aléatoire gaussien X de dimension d et dont les composantes sont indépendantes et de même variance s'écrit

$$f(x) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}\left\|x-\mu\right\|^2\right) \tag{3.3.1}$$

où $\mu \in \mathbb{R}^d$ est son centre et $\sigma^2 > 0$ la variance commune à chaque composante. Un point $x \in \mathbb{R}^d$ de l'espace a une probabilité d'apparition d'autant plus élevée qu'il est proche du point μ. Cette distance à la moyenne intervient dans l'expression de la densité à travers le terme $\|x-\mu\|$. Il traduit le fait que les réalisations de la variable aléatoire suivant cette loi auront une probabilité d'apparition dépendant de leur proximité avec la moyenne. Notre idée consiste à introduire dans l'exponentielle la distance signée du point x à la surface de la sphère. Ceci revient à remplacer le paramètre μ par la projection du point x considéré sur la surface de la sphère.

Soit S une sphère de $\mathbb{R}^d$ de centre $\mu \in \mathbb{R}^d$ et de rayon $r > 0$. Nous recherchons la distance selon chaque composante entre le point $M(x)$ et la sphère. Le vecteur obtenu remplacera le terme $(x-\mu)$ dans la densité de la gaussienne. Comme cela est illustré Figure 3.3.1, le projeté de M sur la sphère est le point P de coordonnées

$$\mu + r\,\frac{(x-\mu)}{\|x-\mu\|}. \tag{3.3.2}$$

Dans le cas où M est confondu avec μ, alors la quantité $\|x-\mu\|$ est nulle et le point P est un point quelconque sur la sphère. Le terme de proximité entre un point $M(x)$ et la sphère S est donc

$$(x-\mu)\left(1-\frac{r}{\|x-\mu\|}\right) \tag{3.3.3}$$

et la distance signée correspondante vaut

$$\|x-\mu\| - r. \tag{3.3.4}$$

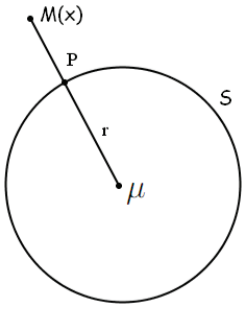


FIGURE 3.3.1 – Projeté P dans $\mathbb{R}^2$ d'un point M sur la sphère S de centre μ et de rayon r.

Tous ces éléments nous conduisent à introduire la densité définie ci-dessous

$$f(x) = K_d \exp\left(-\frac{1}{2\sigma^2}\left(\|x-\mu\| - r\right)^2\right) \tag{3.3.5}$$

avec $\mu \in \mathbb{R}^d$ le centre, $r > 0$ le rayon, $\sigma^2 > 0$ la dispersion et où K_d est une constante de normalisation donnée par

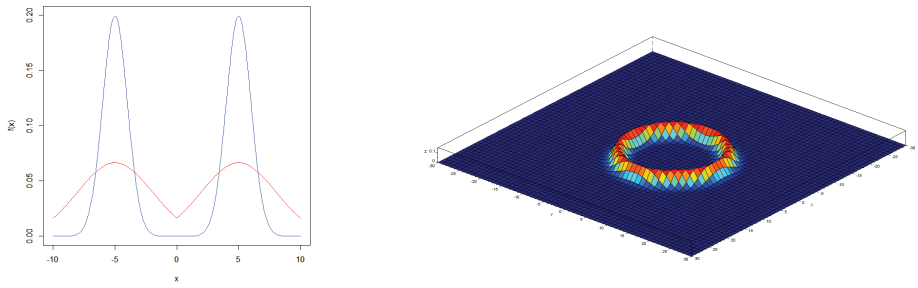
$$K_d = \frac{\Gamma(d/2)}{2\pi^{d/2}\, r^d\, J_{d-1}(\sigma/r)} \tag{3.3.6}$$

où $\Gamma(\,.\,)$ désigne la fonction Gamma et pour $q \in \mathbb{N}$ et $\alpha > 0$,

$$J_q(\alpha) = \int_0^\infty t^q \exp\left(-\frac{(t-1)^2}{2\alpha^2}\right) dt. \tag{3.3.7}$$

Comme cette distribution est conçue pour modéliser des points répartis autour d'une sphère avec des fluctuations dans la direction normale, nous ne considérons que le cas $r \gg \sigma$. En effet, lorsque cette condition n'est pas satisfaite, la distribution génère des observations dans la sphère toute entière (à l'intérieur), ce qui ne correspond pas au cas de figure souhaité. L'hypothèse $r \gg \sigma$ nous assure que la densité devient nulle (d'un point de vue numérique) lorsque l'on s'éloigne suffisamment de la surface. Les fluctuations n'interagissent pas avec le côté opposé de la sphère et les observations restent proches de la surface.

Ces considérations sont illustrées Figure 3.3.2 dans les cas $d = 1$ et $d = 2$. Lorsque $d = 1$ et $r = 5\,\sigma/3$, la densité est une somme de deux gaussiennes se chevauchant et des observations peuvent se situer vers le centre de la sphère avec une probabilité non négligeable. Si $r = 5\,\sigma$, la densité est une somme de deux densités gaussiennes centrées en $\mu - r$ et $\mu + r$ de variance σ^2 et dont les support sont restreints à leur coté du point μ . Notre hypothèse sur les paramètres r et σ revient à considérer exactement deux gaussiennes non tronquées. Plus généralement, une gaussienne de variance σ^2 est centrée sur chaque point appartenant à la surface. Cette hypothèse est raisonnable dans le cadre de notre application pratique puisque la variabilité des points autour des sphères sera faible par rapport aux rayons des têtes que l'on cherchera à modéliser.



(a) (b)

FIGURE 3.3.2 – En (a) la densité pour $d = 1$ et de paramètres $(\mu = 0, r = 5, \sigma = 1)$ en bleu et $(\mu = 0, r = 5, \sigma = 3)$ en rouge. En (b) la densité pour $d = 2$ et de paramètres $(\mu = (0,0)^T, r = 10, \sigma = 1)$.

Le calcul de l'expression exacte (3.3.6) de la constante de normalisation K_d a été détaillé en Annexe A. Pour les première valeurs de d la constante s'écrit (indépendamment de l'hypothèse sur les paramètres)

$$K_1^{-1} = 2\sigma\sqrt{2\pi}\left[1 - \Phi\left(-\frac{r}{\sigma}\right)\right] \tag{3.3.8}$$

$$K_2^{-1} = (2\pi)^{3/2}\sigma r\left[1 - \Phi\left(-\frac{r}{\sigma}\right) + \frac{\sigma}{r\sqrt{2\pi}}\exp\left(-\frac{r^2}{2\sigma^2}\right)\right] \tag{3.3.9}$$

$$K_3^{-1} = 2(2\pi)^{3/2}\sigma(r^2+\sigma^2)\left[1 - \Phi\left(-\frac{r}{\sigma}\right) + \frac{\sigma r}{\sqrt{2\pi}(r^2+\sigma^2)}\exp\left(-\frac{r^2}{2\sigma^2}\right)\right] \tag{3.3.10}$$

où Φ désigne la fonction de répartition de la loi Normale centrée réduite. Pour simplifier les calculs que nous effectuerons dans la suite de ce travail, nous proposons d'approcher la constante de normalisation. Comme cela a été discuté en Annexe A, sous l'hypothèse que $r >> \sigma$, la constante peut être approchée et on a en particulier

$$K_1^{-1} = 2\sigma\sqrt{2\pi} \tag{3.3.11}$$

$$K_2^{-1} = (2\pi)^{3/2}\sigma r \tag{3.3.12}$$

$$K_3^{-1} = 2(2\pi)^{3/2}\sigma\left(r^2+\sigma^2\right). \tag{3.3.13}$$

3.3.2 Propriétés

Nous allons maintenant nous intéresser à quelques propriétés de la densité nouvellement introduite. Lorsque $d \geq 2$, la distribution de densité f appartient à la famille des distributions elliptiques présentée en section 3.2. En effet, si on pose $\Lambda = r^2 I_d$, où I_d est la matrice identité de taille d, et pour $t \geq 0$,

$$g_\alpha(t) = \frac{\Gamma(d/2)}{2\pi^{d/2}J_{d-1}(\alpha)}\exp\left(-\frac{\left(\sqrt{t}-1\right)^2}{2\alpha^2}\right) \tag{3.3.14}$$

alors la densité f peut se réécrire sous la forme

$$f(x) = (\mid \Lambda \mid)^{-1/2} g_{\sigma/r}\left((x-\mu)^T\Lambda^{-1}(x-\mu)\right) \tag{3.3.15}$$

ce qui implique d'après la Définition 3.2.2 que f est la densité d'une distribution elliptique. Par conséquent, d'après la Définition 3.2.4, si on pose $Y = \Lambda^{-1/2}(X-\mu) = (X-\mu)/r$, alors Y a pour densité $g_{\sigma/r}(y^T y)$ et peut s'écrire sous la forme $Y = WU$ où W est une variable aléatoire positive indépendante de U uniformément distribuée sur la sphère unité de $\mathbb{R}^d$. Cette écriture correspond à un changement de variable en coordonnées polaires (voir Annexe A (A5) - (A8)). Le vecteur aléatoire X peut donc s'écrire d'après le Théorème 3.2.1

$$X = \mu + r\,W\,U \tag{3.3.16}$$

et la densité de W est de la forme

$$\varphi(t) = \frac{2\pi^{d/2}}{\Gamma(d/2)} t^{d-1} g_{\sigma/r}(t^2) = \frac{t^{d-1}}{J_{d-1}(\sigma/r)} \exp\left(-\frac{(t-1)^2}{2(\sigma/r)^2}\right) \mathbb{1}_{\{t\geq 0\}}. \tag{3.3.17}$$

Cette dernière remarque facilite le calcul des premiers moments de X. Les résultats du théorème suivant nous permettront de justifier les propriétés des estimateurs qui seront introduits en section 3.4.

Théorème 3.3.1. *Soit X un vecteur aléatoire de $\mathbb{R}^d$ ($d \geq 2$) et de densité de probabilité f introduite en (3.3.5). On pose $J_q = J_q(\sigma/r)$.*

Nous avons donc

$$\mathbb{E}[X] = \mu \qquad \textit{et} \qquad \mathbb{V}\mathrm{ar}(X) = \left(r^2 J_{d+1}\right)/\left(d\,J_{d-1}\right) I_d. \tag{3.3.18}$$

De plus,

$$\mathbb{E}[\|X-\mu\|] = r^* \qquad \textit{avec} \qquad r^* = r\,J_{d-1}^{-1}\,J_d \tag{3.3.19}$$

et

$$\mathbb{V}\mathrm{ar}\big(\|X-\mu\|\big) = \mathbb{E}\left[\big(\|X-\mu\| - r^*\big)^2\right] = r^2\,\frac{\left(J_{d+1}\,J_{d-1} - J_d^2\right)}{J_{d-1}^2} = \widetilde{\sigma}^2 \tag{3.3.20}$$

$$\mathbb{V}\mathrm{ar}\left(\big(\|X-\mu\| - r^*\big)^2\right) = r^4\left(\frac{J_{d+3}}{J_{d-1}} - \frac{4\,J_d\,J_{d+2} + J_{d+1}^2}{J_{d-1}^2} + \frac{8\,J_d^2\,J_{d+1}}{J_{d-1}^3} - \frac{4\,J_d^4}{J_{d-1}^4}\right) \tag{3.3.21}$$

Démonstration. La démonstration du théorème est directe en utilisant le fait que $(X-\mu) = r\,W\,U$ où W et U sont indépendantes, U uniformément distribuée sur la sphère unité de $\mathbb{R}^d$.

En effet, comme $\mathbb{E}[U] = 0$ et $\mathbb{V}\mathrm{ar}[U] = d^{-1} I_d$, on en déduit immédiatement que $\mathbb{E}[X] = \mu$ et $\mathbb{V}\mathrm{ar}[X] = d^{-1}\,r^2\,\mathbb{E}\left[W^2\right]\,I_d$, comme cela a été vu en (3.2.10) et (3.2.11).

En remarquant (voir annexe A) que pour tout $q \geq 0$,

$$\mathbb{E}[W^q] = J_{q+d-1}/J_{d-1},$$

on en déduit la valeur de $\mathbb{V}\mathrm{ar}[X]$.

En plus de cela, du fait que $\|U\| = 1$, il vient que $\|X-\mu\| = r\,W$ et que $\mathbb{E}[X-\mu] = r\,\mathbb{E}[W]$ ce qui est égal à r^* puisque $\mathbb{E}[W] = J_d/J_{d-1}$.

Finalement, comme $\|X-\mu\| - r^* = r\,(W - \mathbb{E}[W])$, il reste à calculer les quatre premiers moments de W. □

Les résultats du Théorème 3.3.1 sont donnés pour $d \geq 2$ et pour toutes valeurs des paramètres r et σ. La remarque suivante concerne le cas particulier $d = 3$ avec $r >> \sigma$, ce qui correspond à notre cadre d'application.

Remarque 3.3.1. *Dans le cas particulier $d = 3$ avec $r \gg \sigma$, nous pouvons approximer et clarifier quelques résultats du Théorème 3.3.1. En effet, en posant $C = \sigma\sqrt{2\pi}/r^3$, on peut montrer que $J_2 = C\,(r^2 + \sigma^2)$ et $J_3 = C\,(r^2 + 3\sigma^2)$ ce qui conduit à $r^* \simeq r$.*

De la même manière, on a $J_4 = C(r^4 + 6r^2\sigma^2 + 3\sigma^4)/r^2$ ce qui conduit à $\widetilde{\sigma}^2 \simeq \sigma^2$. Pour plus de détails, voir l'Annexe A.

Théorème 3.3.2. *Soit X un vecteur aléatoire de $\mathbb{R}^d$ ($d \geq 2$) de densité de probabilité f donnée par (3.3.5). Pour $a > 0$, on pose*

$$X^*(a) = X - a\,\frac{(X-\mu)}{\|X-\mu\|}. \tag{3.3.22}$$

1. *Pour tout $a > 0$, on a*

$$\mathbb{E}\,[X^*(a)] = \mu \qquad \textit{et} \qquad \mathbb{V}\mathrm{ar}\,(X^*(a)) = d^{-1}\big(r^2\,J_{d-1}^{-1}\,J_{d+1} - 2\,a\,r^* + a^2\big)\,I_d.$$

De plus, cette variance est minimale pour $a = r^*$ et, pour tout $a < 2r^*$, $\|\mathbb{V}\mathrm{ar}\,(X^*(a))\| < \|\mathbb{V}\mathrm{ar}\,(X)\|$.

2. *Si X est seulement distribué sur une "sphère tronquée", i.e. X a pour densité $f_{\overline{\Omega}}$ de la forme*

$$f_{\overline{\Omega}}(x) = C_{\overline{\Omega}}\,f(x)\,\mathbb{1}_{\{(x-\mu)\in\overline{\Omega}\}}$$

où $\overline{\Omega}$ et $C_{\overline{\Omega}}$ sont définis en Annexe B, alors $\mathbb{E}\,[X^*(r^*)] = \mu$ et $\|\mathbb{V}\mathrm{ar}\,(X^*(r^*))\| < \infty$.
On a de plus $\mathbb{E}\,[\|X-\mu\|] = r^*$.

Démonstration. La preuve de la première partie est immédiate en utilisant la même approche que celle utilisée dans la preuve du Théorème 3.3.1. En effet, comme $X = \mu + r\,W\,U$, alors $X^*(a) = \mu + (r\,W - a)\,U$, et il vient que $\mathbb{E}\,[X^*(a)] = \mu$ et $\mathbb{V}\mathrm{ar}\,(X^*(a)) = d^{-1}\mathbb{E}\left[(r\,W-a)^2\right]\,I_d$. Par conséquent, puisque $\mathbb{E}\,(W^q) = J_{d-1}^{-1}J_{d+q-1}$ pour tout $q \geq 0$, on obtient la valeur de $\mathbb{V}\mathrm{ar}\,(X^*(a))$ et on en déduit que la variance est minimale pour $a = r\,\mathbb{E}\,[W]$, i.e. $a = r^*$.

La preuve de la seconde partie implique la densité $f_{\overline{\Omega}}$ associée à la "sphère tronquée". A partir de cette densité, on montre en Annexe B que $\mathbb{E}\,[X^*(a)] = \mu + (r^* - a)\,V_{\overline{\Omega}}$ où $V_{\overline{\Omega}} \in \mathbb{R}^d$ est précisé dans l'Annexe. On déduit alors que $\mathbb{E}\,[X^*(r^*)] = \mu$. □

Remarque 3.3.2. La densité $f_{\overline{\Omega}}$ n'est pas elliptique, mais en notant Ω l'ensemble des points de $\overline{\Omega}$ de norme unité, on peut montrer que X peut s'écrire sous la forme $\mu + r\,W\,U_\Omega$ où W et U_Ω sont indépendantes, W a pour densité φ donnée par (3.3.17) et U_Ω est uniformément distribuée sur la surface Ω de la sphère unité d-dimensionnelle. De plus, on a $V_{\overline{\Omega}} = \mathbb{E}\,[U_\Omega]$.

Les résultats du Théorème 3.3.2 prendront du sens lorsque l'on cherchera à estimer les paramètres μ et r à partir d'un échantillon provenant d'un vecteur aléatoire de densité f où $f_{\overline{\Omega}}$. Toutefois, on peut dès à présent souligner qu'une première et importante conséquence du Théorème 3.3.2 est que l'espérance de la variable $X^* = X^*(r^*)$ vaut toujours μ, même si X n'est pas distribuée sur la "sphère complète".

3.3.3 Modèle de mélange

Nous introduisons maintenant le modèle de mélange que nous considérerons pour modéliser et segmenter les nuages de points 3D.

Un modèle de mélange fini est un modèle probabiliste dont la densité de probabilité h est une combinaison convexe de K densités. Généralement, elles appartiennent à une même famille de densités paramétriques et chaque composante du mélange est caractérisée par un vecteur de paramètres. Plus précisément, si f désigne la densité paramétrique commune et θ_k le vecteur de paramètre associé à la composante k, alors la densité de mélange est de la forme

$$h\left(x \mid \Theta\right) = \sum_{k=1}^{K} \pi_k \, f\left(x \mid \theta_k\right) \tag{3.3.23}$$

où les π_k sont les proportions du mélange telles que $\pi_k \geq 0$, $1 \leq k \leq K$ et $\sum_{k=1}^{K} \pi_k = 1$ et où $\Theta = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ est le vecteur contenant tous les paramètres du mélange.

Le choix de la densité f dépend fortement des données à modéliser. Dans la plupart des cas, elle appartient à la famille des distributions normales, de Student ou de Poisson. Étant donné que nous souhaitons modéliser des points 3D répartis autour d'une surface sphérique, f est choisie comme étant la nouvelle densité introduite en (3.3.5) dans le cas $d = 3$. Dans ce cas, le vecteur de paramètres de la composante k est $\theta_k = \left(\mu_k, r_k, \sigma_k^2\right)$ et la densité s'écrit

$$f(x \mid \theta_k) = \frac{1}{2(2\pi)^{3/2}\sigma_k(r_k^2 + \sigma_k^2)} \exp\left(-\frac{1}{2\sigma_k^2}\left(\|x - \mu_k\| - r_k\right)^2\right). \tag{3.3.24}$$

Maintenant que le modèle considéré a été spécifié, nous cherchons à en estimer les paramètres inconnus. Ce point important est traité dans la section suivante.

3.4 Estimation des paramètres du modèle

Dans cette section, nous nous intéressons à l'estimation des paramètres inconnus du modèle de mélange introduit en section 3.3.3. Nous nous plaçons comme dans notre cadre d'application dans le cas $d = 3$. Nous commencerons par traiter le cas d'un modèle à une seule composante en section 3.4.1 et la convergence des estimateurs proposés sera discutée. Nous passerons ensuite au modèle de mélange impliquant plusieurs composantes en section 3.4.2. Nous choisissons pour cela d'appliquer la méthode du maximum de vraisemblance présentée en section 3.4.2.1. La maximisation de la vraisemblance est effectuée par un algorithme EM décrit en section 3.4.2.2. Nous appliquerons enfin la méthode d'estimation à notre modèle de mélange sphérique en section 3.4.2.3.

3.4.1 Estimation du modèle à une seule composante

Dans cette partie, nous traitons le cas de d'un modèle à une seule composante ($K = 1$). Soit $(X_1, \ldots, X_n)$ un échantillon de n observations iid de $\mathbb{R}^3$ distribués selon la densité f introduite en (3.3.5) avec $d = 3$ et $r >> \sigma$. Le but est de proposer des estimateurs des paramètres μ, r et σ^2 caractérisant la distribution.

Il est clair que d'après le Théorème 3.3.1, un estimateur sans biais et convergent du centre μ est la moyenne empirique des observations notée $\overline{X}_n$. Cependant, cet estimateur a une variance plus importante qu'un autre estimateur sans biais et convergent de μ, noté $\overline{X}_n^*$ et défini par

$$\overline{X}_n^* = \overline{X}_n - \frac{r^*}{n} \sum_{i=1}^{n} \frac{(X_i - \mu)}{\|X_i - \mu\|}. \tag{3.4.1}$$

En effet, d'après les Théorèmes 3.3.1 et 3.3.2, $\| \mathbb{V}\text{ar}(\overline{X}_n^*) \| \ll \| \mathbb{V}\text{ar}(\overline{X}_n) \|$. La quantité (3.4.1) dépend bien évidemment des paramètres r^* et μ qui sont inconnus et doivent être estimés.

Considérons maintenant l'estimation du rayon r. Nous introduisons pour cela la quantité

$$\overline{R}_n = n^{-1} \sum_{i=1}^{n} \|X_i - \mu\| . \tag{3.4.2}$$

On peut montrer que $\overline{R}_n$ est un estimateur sans biais et convergent de r^*. De plus, lorsque $r \gg \sigma$, d'après la Remarque 3.3.1 on a $r^* \simeq r$. La quantité $\overline{R}_n$ peut donc être utilisée pour estimer le rayon r lorsque μ est connu.

Intéressons nous enfin à l'estimation du paramètre σ^2. Pour l'estimer, on peut utiliser

$$\overline{\sigma^2}_n = n^{-1} \sum_{i=1}^{n} (\|X_i - \mu\| - r^*)^2 . \tag{3.4.3}$$

Cet estimateur est convergent mais biaisé. Cependant, d'après la Remarque 3.3.1, quand $r \gg \sigma$, le biais se réduit à $2\sigma^4(\sigma^2 - r^2)/(\sigma^2 + r^2)^2$ et peut être considéré comme négligeable.

Finalement, pour obtenir des estimateurs des paramètres μ, r et σ^2, on applique la stratégie suivante qui fournit des estimateurs convergents avec un taux de convergence presque sûr d'ordre $\sqrt{(\log \log n)/n}$. Le centre μ est estimé par $\widehat{\mu}_n^{(0)} = \overline{X}_n$, puis le rayon r par $\widehat{r}_n^{(0)} = n^{-1} \sum_{i=1}^{n} \| X_i - \widehat{\mu}_n^{(0)} \|$ et enfin σ^2 par $\widehat{\sigma}_n^2 = n^{-1} \sum_{i=1}^{n} \left(\| X_i - \widehat{\mu}_n^{(0)} \| - \widehat{r}_n^{(0)} \right)^2$. Les résultats de convergence sont donnés Théorème 3.4.1.

Théorème 3.4.1. *Soit $(X_1, \ldots, X_n)$ un échantillon de n réalisations indépendantes et identiquement distribuées de $\mathbb{R}^3$ de densité f. On a alors*

$$\| \widehat{\mu}_n^{(0)} - \mu \| = O\left(a_n\right), \quad \left| \widehat{r}_n^{(0)} - r^* \right| = O\left(a_n\right) \text{ et } \left| \widehat{\sigma}_n^2 - \widetilde{\sigma}^2 \right| = O\left(a_n\right) \text{ p.s.}$$

où $a_n = \sqrt{(\text{loglog}\, n)/n}$.

Démonstration. La preuve est détaillée en Annexe C. □

Nous avons maintenant un estimateur sans biais de μ et des estimateurs faiblement biaisés de r et σ^2. Nous proposons maintenant une stratégie plus efficace pour estimer les différents paramètres inconnus. Comme mentionné précédemment, on sait que $\| \mathbb{V}\text{ar}(\overline{X}_n^*) \| \ll \| \mathbb{V}\text{ar}(\overline{X}_n) \|$. Par conséquent, en remplaçant respectivement dans (3.4.1) les paramètres inconnus μ et r^* par $\widehat{\mu}_n^{(0)}$ et $\widehat{r}_n^{(0)}$, on peut s'attendre à une estimation plus efficace du centre μ. Notons ce nouvel estimateur $\widehat{\mu}_n^{(1)}$ donné par

$$\widehat{\mu}_n^{(1)} = \overline{X}_n - \widehat{r}_n^{(0)} \frac{1}{n} \sum_{i=1}^{n} \frac{(X_i - \widehat{\mu}_n^{(0)})}{\left\| X_i - \widehat{\mu}_n^{(0)} \right\|}. \tag{3.4.4}$$

La convergence de $\widehat{\mu}_n^{(1)}$ est très difficile à obtenir et n'est pas encore établie. A ce jour, seule la convergence d'une version légèrement modifiée de $\widehat{\mu}_n^{(1)}$ est démontrée dans le théorème suivant.

Théorème 3.4.2. *Soit $X_1, \ldots, X_n$ un échantillon de vecteurs aléatoires de $\mathbb{R}^3$ indépendants et identiquement distribués de densité f On a alors*

$$\widehat{\mu}_n^{\star} = \overline{X}_n - \widehat{r}_n^{(0)} \frac{1}{n} \sum_{i=1}^{n} \frac{(X_i - \widehat{\mu}_{i-1}^{(0)})}{\left\| X_i - \widehat{\mu}_{i-1}^{(0)} \right\|} \quad \underset{n \to \infty}{\overset{p.s.}{\longrightarrow}} \quad \mu. \tag{3.4.5}$$

Démonstration. La preuve est détaillée en Annexe C. □

Comme d'une part $\widehat{\mu}_n^{(0)}$ est plus précis que $\widehat{\mu}_{i-1}^{(0)}$ pour $i \leq n$ et d'autre part $\widehat{\mu}_n^{\star}$ converges vers μ, on peut raisonnablement penser que que $\widehat{\mu}_n^{(1)}$ converge vers μ. La convergence de $\widehat{\mu}_n^{(1)}$ est étudiée d'un point de vue empirique en section 3.5. En particulier, on montre que $\widehat{\mu}_n^{(1)}$ est plus efficace que $\widehat{\mu}_n^{(0)}$ et que sa variance est plus faible. De la même manière, l'estimation de r^* peut être améliorée en substituant $\widehat{\mu}_n^{(0)}$ par $\widehat{\mu}_n^{(1)}$ dans $\widehat{r}_n^{(0)}$, ce qui conduit à estimer r par

$$\widehat{r}_n^{(1)} = \frac{1}{n} \sum_{i=1}^{n} \left\| X_i - \widehat{\mu}_n^{(1)} \right\|. \tag{3.4.6}$$

On peut espérer que si $\widehat{\mu}_n^{(1)}$ estime mieux μ que $\widehat{r}_n^{(0)}$, alors $\widehat{r}_n^{(1)}$ donnera une meilleure estimation de r^* que $\widehat{r}_n^{(0)}$. Cette amélioration est observée en simulation. On peut alors répéter (3.4.4) et (3.4.6) pour améliorer itérativement les estimations de μ et r^*. Les mises à jour $\widehat{\mu}_n^{(0)} = \widehat{\mu}_n^{(1)}$ et $\widehat{r}_n^{(0)} = \widehat{r}_n^{(1)}$ sont effectuées à chaque itération. En statistiques, ce processus itératif est appelé *backfitting* et a été introduit dans [37] pour estimer les paramètres de modèles additifs généralisés. Le critère d'arrêt

du processus de backfitting est basé sur la différence entre deux estimations successives. Enfin, le paramètre σ^2 est estimé par

$$\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\left\|X_i - \widehat{\mu}_n^{(1)}\right\| - \widehat{r}_n^{(1)}\right)^2 \tag{3.4.7}$$

où $\widehat{\mu}_n^{(1)}$ et $\widehat{r}_n^{(1)}$ sont les dernières estimations de μ et r^*.

On considère maintenant le cas d'observations non-uniformément réparties sur la sphère, par exemple réparties sur une demi-sphère. Le traitement des données réelles est concerné par cette caractéristique particulière puisque les capteurs 3D ne fournissent des observations que sur une face des objets. Soit $Y_1, \ldots, Y_n$ un échantillon de vecteurs aléatoires de $\mathbb{R}^3$ indépendants et identiquement distribués sur une "sphère tronquée" de centre μ et de rayon r (voir Théorème 3.3.2 pour la définition). Il est clair que la moyenne empirique $\overline{Y}_n$ n'est pas adaptée pour estimer le centre μ. Toutefois, la seconde partie du Théorème 3.3.2 indique que $\overline{Y}_n^*$ donné par (3.4.1), où X_i est remplacé par Y_i, est un estimateur sans biais et convergent du centre μ même si les observations ne sont pas distribuées sur la sphère complète. Les quantités μ et r^* sont inconnues mais nous montrerons en simulation que l'utilisation de l'algorithme du backfitting permet d'obtenir des estimations précises de μ et de r^*.

La remarque suivante permet de mieux comprendre la raison pour laquelle cette stratégie fonctionne. L'estimateur (3.4.4) est composé de deux termes : la moyenne empirique des observations et un second terme représentant un vecteur pointant dans la direction moyenne des points vers la dernière estimation du centre. Dans le cas de données seulement distribuées sur une demi-sphère, la moyenne empirique est déplacée, grâce au second terme, dans la direction du vrai centre. Chaque amélioration du centre entraîne une amélioration du rayon et ainsi de suite jusqu'à convergence. Le comportement de cette méthode d'estimation, schématisée Figure 3.4.1, sera étudié sur des données simulées en section 3.5.

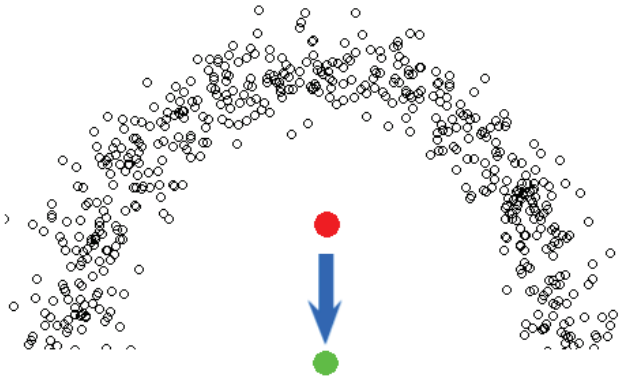


FIGURE 3.4.1 – La moyenne empirique (en rouge) est un mauvais estimateur du centre (en vert) dans le cas de la demi-sphère. Le terme correctif de l'estimateur proposé est représenté par la flèche bleue.

3.4.2 Estimation du modèle de mélange

Dans cette partie, nous cherchons à estimer le vecteur de paramètres inconnus Θ du modèle de mélange à K composantes introduit en 3.3.3. La densité de probabilité du mélange s'écrit

$$h(x, \Theta) = \sum_{k=1}^{K} \pi_k \, f(x, \theta_k) \tag{3.4.8}$$

où les π_k sont les proportions du mélange telles que $\pi_k \geq 0$, $1 \leq k \leq K$ et $\sum_{k=1}^{K} \pi_k = 1$ et où $\Theta = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ et $\theta_k = \left(\mu_k, r_k, \sigma_k^2\right)$. Nous nous plaçons dans le cas $d = 3$ sous l'hypothèse $r >> \sigma$ puisqu'il s'agit de notre cadre d'application.

Pour estimer Θ, nous choisissons d'appliquer la méthode du maximum de vraisemblance. Ce choix nous conduit naturellement à l'utilisation de l'algorithme Espérance-Maximisation appliqué aux modèles de mélange.

3.4.2.1 Méthode du maximum de vraisemblance

L'estimation d'un modèle statistique nécessite de bâtir des estimateurs des paramètres inconnus à partir d'un ensemble d'observations. Il existe différentes méthodes pour les obtenir. Dans ce travail, nous choisissons d'appliquer la méthode du maximum de vraisemblance. Cette méthode a été introduite par Fisher (voir [7]) et est très utilisée en statistiques.

Soit X un vecteur aléatoire de dimension d et de densité de probabilité h paramétrée par un vecteur de paramètres $\Theta \in \mathbb{R}^p$. Étant donné un échantillon $\mathbf{X} = (X_1, ..., X_n)$ de n observations iid de $\mathbb{R}^d$ distribué selon la densité h, le but est d'estimer le vecteur de paramètres Θ caractérisant la distribution. L'estimateur du maximum de vraisemblance $\widehat{\Theta}$ est défini par

$$\widehat{\Theta} = \arg\max_{\Theta} L\left(X, \Theta\right) \tag{3.4.9}$$

où L désigne la fonction de vraisemblance (ou de log-vraisemblance) définie par

$$L\left(X, \Theta\right) = P\left[(X_1, ..., X_n) \mid \Theta\right]. \tag{3.4.10}$$

Dans le cas particulier du modèle de mélange (3.4.8) et avec l'indépendance des observations, la log-vraisemblance (3.4.10) s'écrit

$$L\left(X, \Theta\right) = \log\left(\prod_{i=1}^{n} h(X_i, \Theta)\right) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} \pi_k \, f\left(X_i; \theta_k\right)\right). \tag{3.4.11}$$

Maximiser l'expression (3.4.11) est difficile à cause de sa non-linéarité en de nombreux paramètres. La résolution directe du problème de maximisation consistant à résoudre les équations de vraisemblance ($\partial L(X, \Theta)/\partial\theta_i = 0$ pour $i = 1, \ldots, p$) ne permet pas d'aboutir. Des techniques sophistiquées comme l'algorithme Espérance-Maximisation (EM) ([69]) ont justement été développées pour résoudre ce type de problème.

3.4.2.2 Algorithme Espérance-Maximisation appliqué aux modèles de mélange

L'algorithme Espérance-Maximisation (EM) est un processus itératif permettant de trouver le maximum de vraisemblance des paramètres d'un modèle lorsqu'il dépend de variables non-observables. Il est particulièrement utile dans le cas où les équations de vraisemblance ne peuvent être résolues directement, notamment dans le cas des modèles de mélange. Il a été introduit par A. P. Dempster et al. dans [69] et de nombreuses extensions ont été proposées (voir par exemple [130]), dont notamment les algorithmes EM généralisé (GEM) ([69]), EM stochastique (SEM) ([45]) et classification EM (CEM) ([48]).

La maximisation de la log-vraisemblance (3.4.11) n'est pas simple. Dans le cas des modèles de mélange, on suppose que chaque observation X_i provient de l'une des composantes du mélange. L'idée principale de l'algorithme EM consiste à associer à chaque observation X_i une variable dite *cachée* ou *latente* notée $z_i = (z_{i1}, \ldots, z_{iK})$ contenant l'information d'appartenance aux classes $(C_1, \ldots, C_K)$ et définie par $z_{ik} = \mathbb{1}_{\{X_i \in C_k\}}$. Les valeurs des variables z_i définissent une partition des données en classes. Les observations $\mathbf{x}$ sont appelées *données observées* et $\mathbf{z} = (z_1, \ldots, z_n)$ *données manquantes* puisque ces dernières ne sont pas observables.

La connaissance de l'information manquante $\mathbf{z}$ permettrait de résoudre entièrement le problème de maximisation puisqu'il suffirait alors de traiter chaque composante indépendamment. La variable $\mathbf{z}$ n'est pas connue et devra donc être estimée. L'introduction de la variable $\mathbf{z}$ dans l'expression (3.4.11) permet d'écrire la log-vraisemblance dite *complétée*

$$L_c(X, \Theta) = \sum_{k=1}^{K} \sum_{i=1}^{n} z_{ik} \left[\log(\pi_k) + \log(f(X_i, \theta_k)) \right]. \tag{3.4.12}$$

L'algorithme EM consiste à maximiser l'espérance conditionnelle de la log-vraisemblance complétée étant donnés l'observation $\mathbf{x}$ et l'estimation courante des paramètres $\Theta^{(j-1)}$ à l'itération précédente $j-1$

$$Q\left(\Theta, \Theta^{(j-1)}\right) = \mathbb{E}\Big[L(X, Z \mid \Theta) \mid X, \Theta^{(j-1)} \Big]. \tag{3.4.13}$$

L'algorithme se résume finalement aux deux étapes suivantes itérées successivement jusqu'à convergence

- **Étape E :** Estimer les probabilités d'appartenance $t_{i\ell}$ des observations $(X_i)_{i=1}^n$ aux classes $(C_\ell)_{\ell=1}^K$ à partir de l'estimation courante $\Theta^{(j)} = \left(\pi_1^{(j)}, \ldots, \pi_K^{(j)}, \theta_1^{(j)}, \ldots, \theta_K^{(j)}\right)$

$$t_{i\ell} = \frac{\pi_\ell^{(j)} f\left(X_i \mid \theta_\ell^{(j)}\right)}{\sum_{k=1}^{K} \pi_k^{(j)} f\left(X_i \mid \theta_k^{(j)}\right)}. \tag{3.4.14}$$

  La quantité $t_{i\ell}$ est une estimation de $\mathbb{E}\left[z_{i\ell} \mid X, \Theta^{(j)} \right]$.
- **Étape M :** Déterminer le nouveau vecteur de paramètres $\Theta^{(j+1)}$ maximisant la quantité

$$Q(\Theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} t_{ik} \big(\log(\pi_k) + \log(f(X_i \mid \theta_k)) \big). \tag{3.4.15}$$

Le processus nécessite un vecteur de paramètres initial $\Theta^{(0)}$. Il peut être obtenu par exemple à l'aide d'un algorithme K-means ([28]) fournissant des classes desquelles les paramètres peuvent être calculés. L'algorithme K-means est une méthode itérative classique de partitionnement des données en K classes basée sur la distance euclidienne entre les observations et les centres des classes.

Notons que l'étape E consiste à estimer les données manquantes et que l'étape M maximise l'espérance conditionnelle de la vraisemblance complétée étant données ces estimations. La maximisation de la fonction Q à l'étape M est effectuée analytiquement lorsque cela est possible. Dans le cas contraire, une méthode numérique peut être employée. Dans notre cas, nous préférons effectuer la maximisation analytiquement pour limiter de complexité totale de la méthode.

L'algorithme est stoppé lorsque l'un des critères suivants est satisfait

- un nombre d'itération maximal prédéfini N_{max} est atteint
- la log-vraisemblance n'augmente plus significativement : $\mid L(X, \Theta^{(j)}) - L(X, \Theta^{(j-1)}) \mid < \epsilon_L$
- les classes n'évoluent plus : $\left\| \Theta^{(j)} - \Theta^{(j-1)} \right\| < \epsilon_{param}$.

Les valeurs des paramètres estimés sont sensibles à la solution initiale $\Theta^{(0)}$ choisie pour initialiser le processus. La méthode garantit que chaque itération ne fait pas diminuer la vraisemblance, mais la solution obtenue peut n'être seulement que l'un des nombreux maxima locaux de la fonction de vraisemblance. Enfin, le nombre de composantes K doit être connu et fixé dès le début de l'algorithme. Ce dernier point fera ultérieurement l'objet d'une discussion plus détaillée en fin de chapitre et lors de la mise en œuvre pratique de la méthode.

3.4.2.3 Application au cas sphérique

Dans cette partie, nous appliquons l'algorithme EM pour estimer le vecteur de paramètres Θ du modèle de mélange (3.4.8) dans le cas $d = 3$ et sous l'hypothèse $r \gg \sigma$.

L'étape E nécessite l'évaluation la densité h pour calculer les probabilités d'appartenance et ne pose pas de difficultés particulières. La maximisation de la fonction Q lors de l'étape M est effectuée analytiquement en calculant ses dérivées partielles par rapport à chacun des paramètres. Les calculs sont détaillés en Annexe D. Pour chaque composante ℓ, les équations définissant les estimateurs sont

$$\begin{cases} \widehat{\mu}_\ell = \dfrac{1}{\sum_{i=1}^n t_{i\ell}} \left(\displaystyle\sum_{i=1}^n t_{i\ell}\, X_i - \widehat{r}_\ell \sum_{i=1}^n t_{i\ell}\, \frac{(X_i - \widehat{\mu}_\ell)}{\|X_i - \widehat{\mu}_\ell\|} \right) \\ \widehat{r}_\ell = \dfrac{\sum_{i=1}^n t_{i\ell}\, \|X_i - \widehat{\mu}_\ell\|}{\sum_{i=1}^n t_{i\ell}} \\ \widehat{\sigma}_\ell^2 = \dfrac{\sum_{i=1}^n t_{i\ell}\, (\|X_i - \widehat{\mu}_\ell\| - \widehat{r}_\ell)^2}{\sum_{i=1}^n t_{i\ell}} \\ \widehat{\pi}_\ell = \dfrac{1}{n} \displaystyle\sum_{i=1}^n t_{i\ell} \end{cases} . \tag{3.4.16}$$

Les relations (3.4.16) sont similaires à celles obtenues dans le cas de l'échantillon. Ces estimateurs n'ont cette fois encore pas de forme explicite simple. Nous utilisons donc le processus de backfitting

décrit en section 3.4.1. Dans un premier temps, le centre et le rayon sont estimés par

$$\widehat{\mu}_n^{(0)} = \frac{1}{\sum_{i=1}^n t_{i\ell}} \sum_{i=1}^n t_{iell}\, X_i \quad \text{et} \quad \widehat{r}_n^{(0)} = \frac{1}{\sum_{i=1}^n t_{iell}} \sum_{i=1}^n t_{iell} \parallel X_i - \widehat{\mu}_n^{(0)} \parallel . \tag{3.4.17}$$

Rappelons que ces estimations ne sont pas bonnes dans le cas de données non-uniformément réparties sur la surface, par exemple dans le cas d'observations situées sur une demi-sphère. Pour cette raison, les estimations sont améliorées itérativement grâce aux relations (3.4.16). Ces équations sont donc évaluées plusieurs fois à chaque étape M. Le processus de backfitting est stoppé lorsque la différence entre deux estimations successives est suffisamment faible

$$\left\| \mu_\ell^{(j)} - \mu_\ell^{(j-1)} \right\| < \epsilon_\mu \quad \text{et} \quad \left| r_\ell^{(j)} - r_\ell^{(j-1)} \right| < \epsilon_r \tag{3.4.18}$$

où lorsqu'un nombre d'itérations maximal prédéfini N_{max}^{BF} est atteint.

Comme mentionné dans le cas de l'échantillon, les estimateurs proposés sont robustes aux données manquantes et à l'inégale répartition des observations sur la surface. Le bon comportement de la méthode d'estimation proposée sera expérimenté sur des données simulées en section 3.5 et réelles en section 3.6.

3.5 Expérimentations sur des données simulées

Dans cette section, nous nous intéressons au comportement de la méthode proposée (BF) sur des données simulées. Ces expérimentations ont pour but de montrer que le processus de backfitting se comporte comme attendu et que l'algorithme EM estime correctement le modèle de mélange. La qualité des résultats obtenus sera quantifiée sur différents jeux de données. Nous commençons par traiter le cas d'une seule sphère complète en 3.5.1 et tronquée en 3.5.2 avant de passer à plusieurs sphères en 3.5.3.

La méthode d'estimation proposée sera ensuite comparée à l'algorithme RANSAC (RANdom SAmple Consensus) ([84, 169]) utilisant une méthode de moindres carrés ([176]). Cette méthode est préférée à la transformée de Hough ([3]) nécessitant une grande quantité de mémoire pour discrétiser l'espace des paramètres. Ces méthodes ont été détaillées Chapitre 1 section 1.3.5. Dans l'algorithme RANSAC utilisé, le score d'un modèle candidat est le ratio entre le nombre de points consistants au modèle et l'erreur d'ajustement des moindres carrés. Cette stratégie sera justifiée dans le chapitre suivant lors des expérimentations sur des données réelles. Les paramètres de la méthode sont le nombre de sélections aléatoires, le seuil de consistance et le score minimal d'acceptation d'un modèle. Dans cette section, il ont été respectivement fixés à 200, 20 et 100.

Nous comparons également l'algorithme BF avec les deux méthodes basiques consistant à estimer (μ, r) par $\left(\overline{X}_n, \widehat{r}_n^{(0)}\right)$ (méthode BF_0) et $\left(\widehat{\mu}_n^{(1)}, \widehat{r}_n^{(1)}\right)$ (méthode BF_1). Ces deux méthodes correspondent à l'initialisation et à la première itération de l'algorithme BF. Pour illustrer le comportement de la méthode proposée, nous considérons aussi le second itéré $\left(\widehat{\mu}_n^{(2)}, \widehat{r}_n^{(2)}\right)$.

On considère un vecteur aléatoire X de dimension $d = 3$ et de densité de probabilité f introduite en (3.3.5). Les paramètres des distributions ont été choisis de manière à se placer dans des conditions similaires à celles de notre application pratique. Le ratio r/σ et le nombre d'observations n sont choisis élevés. Les observations ont été générées par la méthode de simulation décrite en Annexe E.

3.5.1 Résultats dans le cas de la sphère complète

La première série d'expérimentations traite le cas d'une seule composante ($K = 1$). On simule $N = 200$ échantillons indépendants de réalisations de la variable aléatoire X de taille $n = 50, 100, 200, 500, 1000, 5000$ et 10000. Les paramètres de la loi sont fixés à $\mu = (0, 0, 0)^T$, $r = 50$ et $\sigma = 1$. Ces valeurs devraient conduire à des des estimations dont les biais sont négligeables (voir section 3.4.1).

Les résultats obtenus avec l'algorithme BF sont présentés Figure 3.5.1 où chaque boîte à moustaches décrit la répartition des N estimations des paramètres pour différentes tailles d'échantillon. Cette représentation nous informe à la fois sur l'erreur commise et sur la dispersion des valeurs. On observe que les estimations sont proches des valeurs exactes des paramètres et que comme attendu les biais sont négligeables. Il est clair que les estimations s'améliorent et que la variabilité diminue lorsque la taille de l'échantillon augmente. Ces premiers résultats montrent que l'algorithme fournit de bonnes estimations des paramètres même pour un faible nombre d'observations n.

Comparons maintenant la performance de l'algorithme BF avec l'algorithme RANSAC et les trois BF partiels. La Figure 3.5.2 contient les boîtes à moustaches des estimations de μ_x et r obtenues sur un échantillon de taille $n = 1000$. Il est clair que l'algorithme BF est le plus performant. L'algorithme RANSAC est presque aussi bon mais il donne de mauvaises estimations dans certains cas. Comme

cela a été mentionné en section 3.4.1, la variance de $\widehat{\mu}_n^{(1)}$ est plus faible que celle de $\overline{X}_n$: une seule itération du backfitting a réduit la variance des estimateurs. Plus généralement, on peut voir que chaque nouvelle itération améliore l'estimation et réduit la variance.

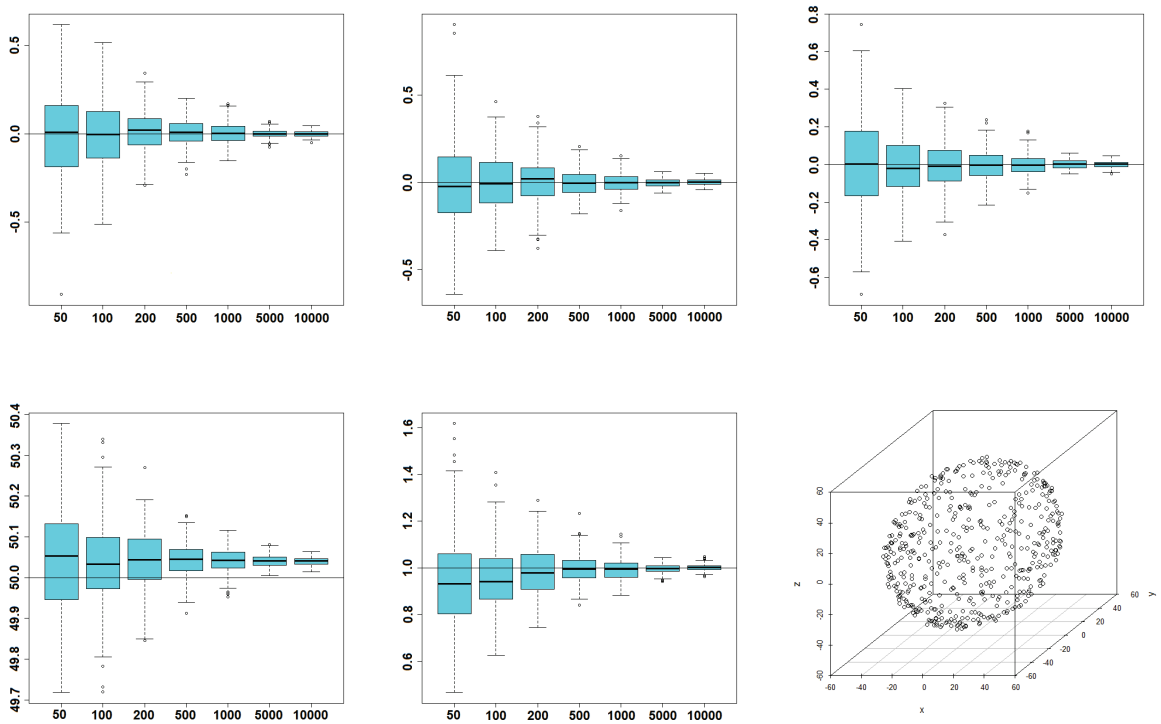


FIGURE 3.5.1 – Sur la première ligne, boîtes à moustaches des estimations (BF) de μ_x, μ_y et μ_z respectivement. Sur la seconde ligne, boîtes à moustaches des estimations (BF) de r et σ^2 respectivement. En bas à droite, un échantillon de taille $n = 1000$ de la sphère considérée.

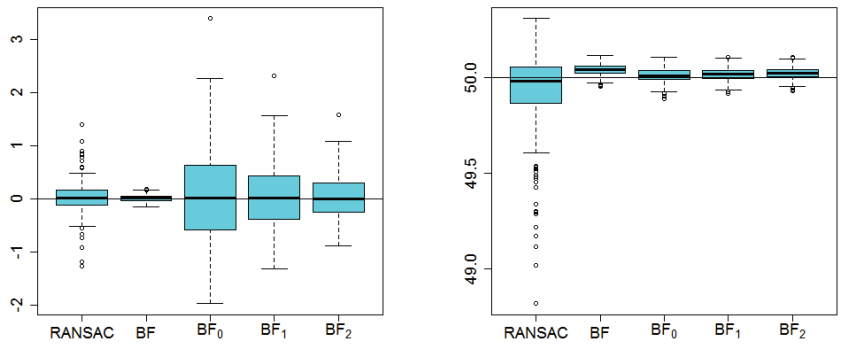


FIGURE 3.5.2 – Comparaison de BF, RANSAC et méthodes BF_0, BF_1 et BF_2. Estimations de μ_x à gauche et r à droite pour $n = 1000$.

Nous nous intéressons enfin au nombre d'itérations effectuées lors du processus de backfitting selon le nombre d'observations n. Les critères d'arrêt du backfitting ont été fixés à $\epsilon_\mu^{BF} = \epsilon_r^{BF} = 0.1$ et $N_{max}^{BF} = 100$. Les résultats sont représentés Figure 3.5.3. On observe que le nombre d'itérations effectuées diminue avec la taille de l'échantillon.

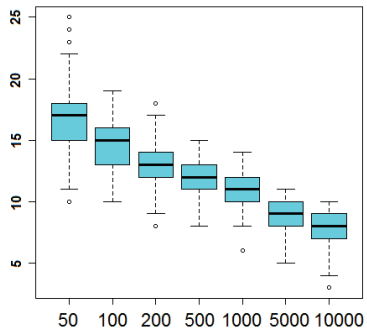


FIGURE 3.5.3 – Nombre d'itérations de backfitting effectuées en fonction du nombre d'observations n.

3.5.2 Résultats dans le cas de la demi-sphère

L'utilisation d'un seul capteur 3D produit des données réparties sur une seule face des objets. Pour se rapprocher des conditions de notre application pratique, nous simulons des observations réparties sur une demi-sphère. Plus précisément, nous ne conservons que la moitié de sphère dont les points ont une composante y positive.

Le résultats obtenus avec l'algorithme BF sont présentés Figure 3.5.4. Nous nous focalisons sur la composante μ_y pour le centre puisqu'il s'agit de la plus difficile à estimer. On observe des résultats similaires à ceux obtenus dans le cas de la sphère complète. La méthode est donc robuste à la répartition non-uniforme des données sur la surface et aux données manquantes.

Comparons maintenant les performances de l'algorithme BF avec celles de RANSAC et des trois BF partiels. Comme dans le cas de la sphère complète, l'algorithme BF est légèrement meilleur que RANSAC (Figure 3.5.5). De plus, il est clair que le centre μ (et donc le rayon r) n'est pas correctement estimé (μ_y en particulier) avec $\overline{X}_n$ ou $\widehat{\mu}_n^{(1)}$, même si $\widehat{\mu}_n^{(1)}$ est meilleur. Cette fois encore, pour une taille d'échantillon donnée, la précision des estimations s'améliore et leur variabilité décroit au fur et à mesure des itérations. Il est donc nécessaire d'itérer (3.4.4) et (3.4.6), c'est à dire d'appliquer le backfitting, pour obtenir de bonnes estimations des paramètres.

Ce dernier point est illustré Figure 3.5.6 où l'on peut observer l'évolution des estimations de μ_y et r pendant le processus itératif. Ces courbes illustrent l'amélioration des estimations à chaque itération et montrent clairement le bénéfice apporté par le backfitting. On constate que le processus itératif peut aboutir à des estimations imprécises si trop peu d'itérations sont effectuées. Il faut donc trouver un compromis entre le temps de calcul et la précision souhaitée selon l'application visée. Pour l'échantillon considéré, au moins 50 itérations sont requises pour obtenir des estimations correctes. Ce problème ne se pose pas dans le cas de la sphère complète puisque chaque étape de l'algorithme

fournit un estimateur convergent des paramètres. Ce point sera discute plus en détails lors de la mise en œuvre pratique de la méthode.

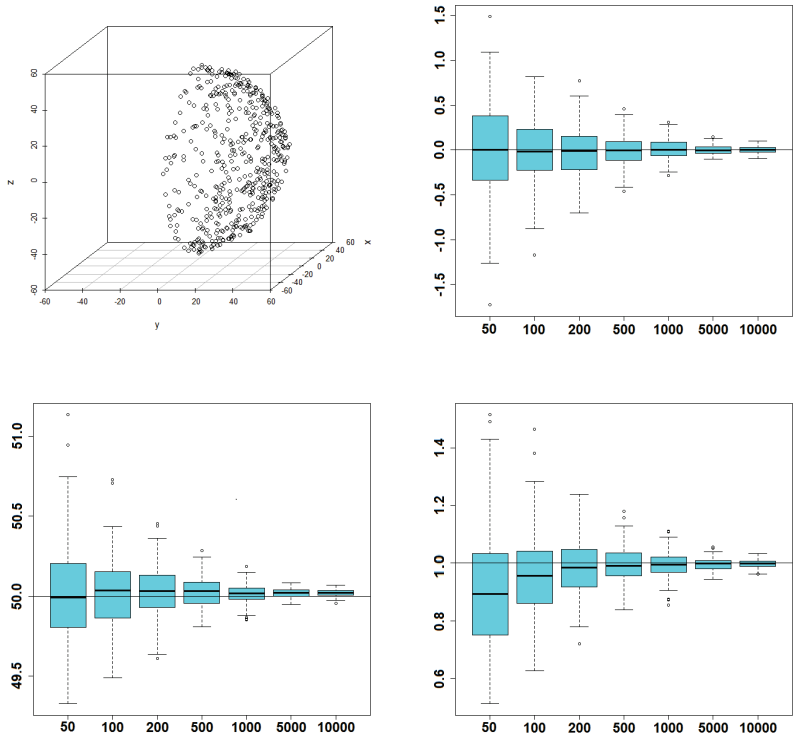


FIGURE 3.5.4 – En haut à gauche, un échantillon de taille $n = 1000$ d'une demi-sphère de paramètres $\mu = (0, 0, 0)^T$, $r = 50$ et $\sigma^2 = 1$. Boîtes à moustaches des estimations de μ_y en haut à droite, r en bas à gauche et σ^2 en bas à droite pour différentes tailles d'échantillons.

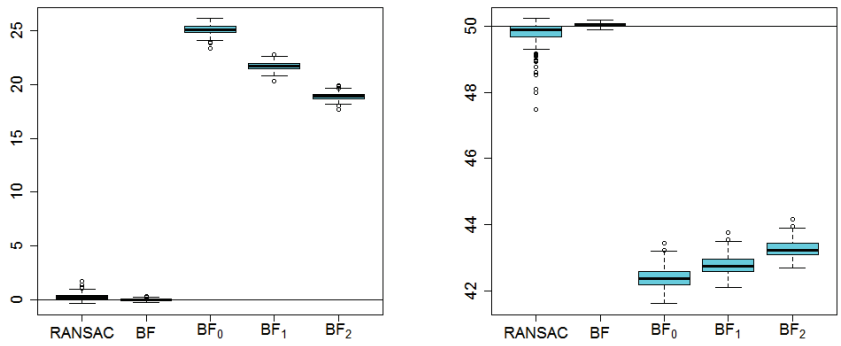


FIGURE 3.5.5 – Comparaison de BF, RANSAC et méthodes BF_0, BF_1 et BF_2. Estimations de μ_y à gauche et r à droite pour $n = 1000$.

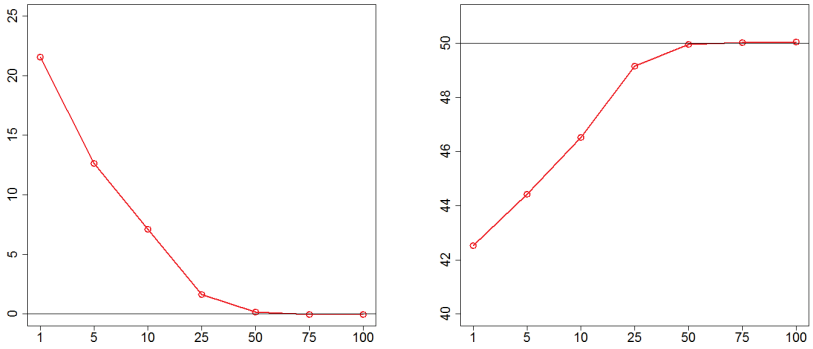


FIGURE 3.5.6 – Convergence de l'algorithme BF. A gauche, estimation de μ_y et à droite estimation de r, en fonction du nombre d'itérations effectuées.

Nous nous intéressons enfin au nombre d'itérations effectuées lors du processus de backfitting selon le nombre d'observations n. Le critère d'arrêt du backfitting ont été fixés à $\epsilon_{\mu}^{BF} = \epsilon_{r}^{BF} = 0.1$ et $N_{max}^{BF} = 100$. Les résultats sont représentés Figure 3.5.7. On observe que le nombre d'itérations effectuées est plus élevé que dans le cas de la sphère complète et qu'il ne diminue pas avec la taille de l'échantillon.

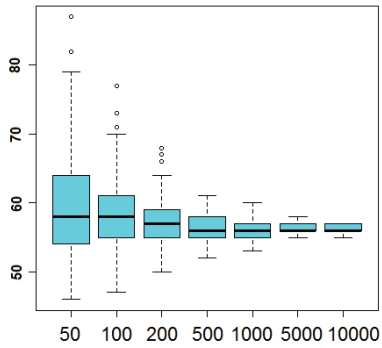


FIGURE 3.5.7 – Nombre d'itérations de backfitting effectuées en fonction de n.

Cette première série d'expérimentations nous a permis de confirmer le bon comportement de l'algorithme BF dans le cas de l'échantillon. Nous n'avons traité dans cette partie que le cas d'une seule composante ($K = 1$). Nous allons nous intéresser dans la partie suivante au comportement des estimateurs dans le cas de plusieurs composantes ($K > 1$).

3.5.3 Résultats dans le cas de plusieurs composantes

La seconde série d'expérimentations consiste à tester la méthode sur plusieurs sphères incomplètes et / ou se chevauchant. Le but est de montrer que la méthode proposée est capable de correctement séparer et modéliser plusieurs classes. Dans la suite de ces expérimentations, le vecteur de paramètres initial $\Theta^{(0)}$ du mélange est obtenu par un algorithme K-means ([28]).

Nous considérons un modèle de mélange sphérique composé de trois sphères s'intersectant et de poids égaux. On simule un échantillon de $n = 3000$ observations. Le nuage de points et les résultats sont représentés Figure 3.5.8. Les trois sphères ont été correctement séparées et modélisées. La classification induite par les probabilités d'appartenance des points aux classes montre que les observations sont globalement bien classées. Les points situés à l'intérieur d'une sphère voisine sont affectés à la bonne composante. Une incertitude demeure toutefois sur les points localisés aux intersections des surfaces.

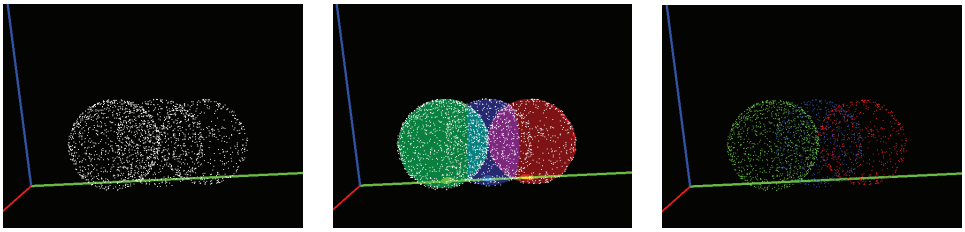
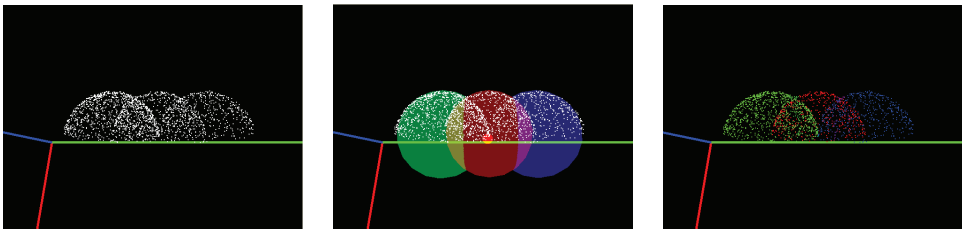


FIGURE 3.5.8 – A gauche le nuage de points, au centre les sphères estimées et à droite la classification.

Le deuxième jeu de données est généré avec les mêmes paramètres que le premier sauf que les points ne sont répartis que sur des demi-sphères. Seule les parties des surfaces dont les points ont une composante x négative ont été conservées. Les résultats obtenus sont donnés Figure 3.5.9. Malgré la répartition non-uniforme des points, les paramètres sont correctement estimés et les différents objets modélisés. La classification ne présente pas d'erreurs importantes, même pour les points situés à l'intérieur d'une demi-sphère voisine. Les résultats sont similaires à ceux obtenus sur les sphères complètes.



(a) (b) (c)

FIGURE 3.5.9 – A gauche le nuage de points, au centre les sphères estimées et à droite la classification.

La dernière expérimentation vise à analyser quantitativement la performance de l'algorithme EM. On considère un modèle de mélange à 3 composantes composé de deux sphères adjacentes et de rayons différents S_1 et S_2 et d'une demi-sphère S_3 éloignée des deux premières. Les paramètres exacts de chaque composantes sont précisés Table 3.5.1.

TABLE 3.5.1 – Valeurs exactes des paramètres.

	μ_x	μ_y	μ_z	r	σ	π
S1	0	0	0	150	3	0.5
S2	200	0	0	100	5	0.25
S3	1000	0	0	50	5	0.25

Dans cette expérimentation, le nombre de composantes K est supposé connu et fixé à 3. On simule $N = 200$ échantillons de taille $n = 500, 1000, 2000, 4000, 10000$. Comme on peut le voir Figure 3.5.10, l'algorithme fonctionne bien et les trois sphères sont correctement identifiées.

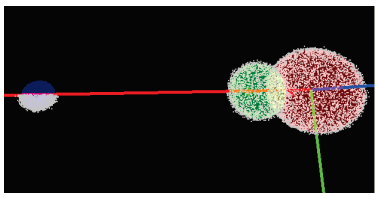


FIGURE 3.5.10 – Un échantillon de taille $n = 10000$ d'un modèle de mélange composé des 3 sphères S_1, S_2 et S_3 superposé avec les sphères estimées.

Les résultats de la Figure 3.5.11 montrent que la précision des estimations augmente avec la taille de l'échantillon n. La faible erreur d'estimation sur les deux premiers poids s'explique par une incertitude sur les points situés vers les intersections des surfaces. La très faible erreur d'estimation sur le dernier poids est due à l'échantillonnage du modèle de mélange.

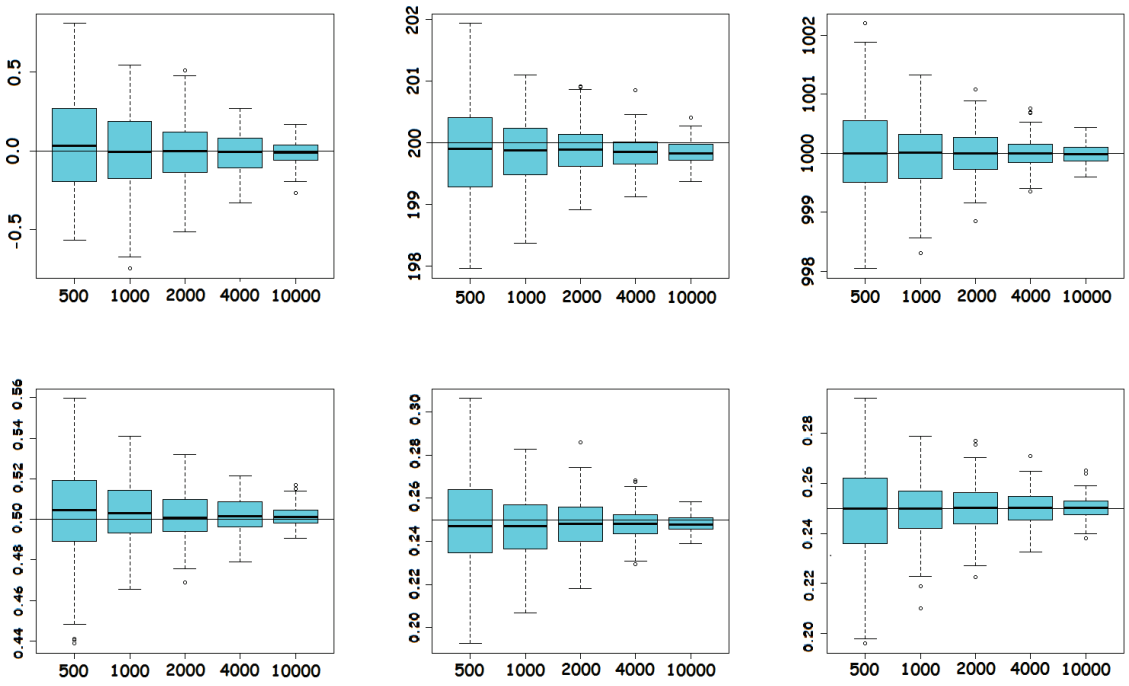


FIGURE 3.5.11 – En haut et de gauche à droite, boîtes à moustaches des estimations de μ_x, μ_y et μ_z de S_1, S_2 et S_3 respectivement. En bas et de gauche à droite, poids π_1, π_2 et π_3 pour différents n.

L'algorithme EM proposé est comparé avec l'algorithme RANSAC dans le cas d'échantillons de taille $n = 4000$. Nous nous focalisons sur les paramètres μ_x de S_1 et S_2, μ_y de S_3 et r de S_1, S_2 et S_3. La Figure 3.5.12 montre que comme dans le cas d'une seule composante notre méthode donne de meilleures estimations avec une variabilité plus faible.

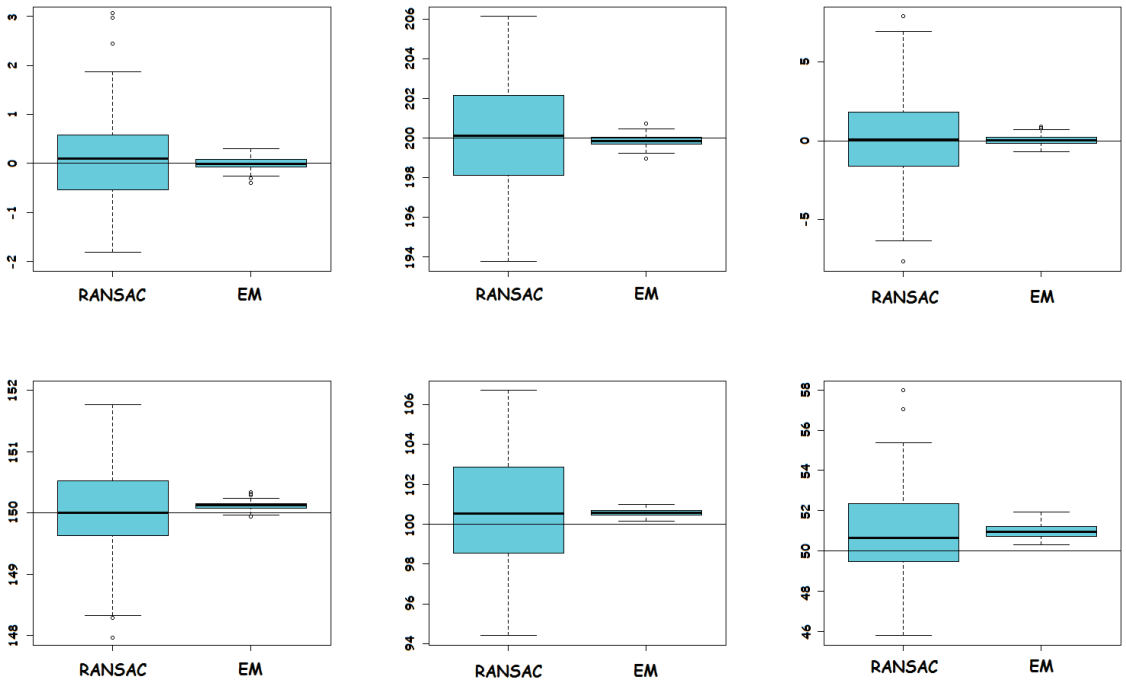


FIGURE 3.5.12 – Comparaison des algorithmes EM et RANSAC ($n = 4000$). En haut et de gauche à droite, boîtes à moustaches des estimations μ_x de S_1 et S_2 et μ_y de S_3 respectivement. En bas et de gauche à droite, boîtes à moustaches des estimations du rayon r de S_1, S_2 et S_3 respectivement.

Pour conclure cette série d'expérimentations sur données simulées, il est important de noter que malgré le fait que les estimations de r et σ^2 soient légèrement biaisées, les poids du mélange sont bien estimés, ce qui valide l'utilisation du modèle pour modéliser des nuages de points 3D. De plus, les données ont été simulées à partir de la densité exacte (3.3.5) alors que les probabilités d'appartenance sont estimées à l'aide de sa version approchée (3.3.24). Les résultats numériques obtenus sur des données simulées sont prometteurs pour les expérimentations sur des données réelles.

3.6 Expérimentations sur des données réelles

Dans cette section, nous expérimentons la méthode sur des données provenant d'un capteur 3D à lumière structurée présenté au Chapitre 1 section 1.2. L'objet d'intérêt est segmenté par un algorithme de soustraction de fond décrit au Chapitre 1 section 1.3.3.1. Nous nous intéressons ici aux résultats obtenus d'un point de vue qualitatif. Les paramètres sont choisis de manière à ce qu'un nombre suffisant d'itérations soient effectuées pour que le résultat soit satisfaisant. Nous supposerons tout d'abord en 3.6.1 que le nombre de composantes K est connu avant de proposer en 3.6.2 un choix automatique de ce paramètre.

3.6.1 Nombre de composantes connu

La mise en œuvre pratique de l'algorithme sur des données réelles nécessite certaines adaptations. Les expérimentations ont montré que l'estimation des variances σ^2 avait une grande influence sur les résultats. En effet, nous avons observé que les valeurs finales des variances estimées dépendent beaucoup de la phase d'initialisation des classes par $\Theta^{(0)}$ via l'algorithme K-means. Si une composante donnée présente une forte variance, les observations appartenant aux autres objets participeront à l'estimation des paramètres. Par conséquent, nous décidons de fixer les variances σ^2 de chacune des composantes plutôt que de les réestimer à chaque itération. Le modèle que nous choisissons d'ajuster aux données est donc accompagné d'une certaine variance fixée à priori. La valeur choisie est liée à l'éloignement maximal d'un point pour qu'il puisse influer sur l'évolution d'une composante. Une série d'essais nous ont permis de fixer empiriquement σ compris entre 5 et 7.

Cette stratégie n'est pas une limitation à l'application de la méthode. En effet, les variances peuvent être considérées comme des paramètres de l'algorithme, de la même manière que le nombre maximal d'itérations du backfitting par exemple. De plus, bien que les variances soient fixées durant la phase d'optimisation, elles peuvent être estimées a posteriori (si nécessaire) à la fin de l'algorithme EM à partir des composantes optimisées. Nous verrons que dans le cadre de notre application pratique, nous sommes surtout intéressée par l'estimation des centres et des rayons des sphères.

Nous commençons par appliquer la méthode sur des objets sphériques simples. Deux balles identiques de rayon 34 mm $\pm$ 1 mm sont placées devant le capteur et la méthode est appliquée pour $K = 2$. Les résultats obtenus sont donnés Figure 3.6.1. Un seul côté des balles est visible et une proportion importante des données est manquante. Les rayons estimés sont compris entre 32 et 33 mm ce qui constitue une bonne approximation. Nous rappelons que la précision des mesures effectuées par le capteur 3D dépendent de la distance le séparant de l'objet (voir Chapitre 1 section 1.2).

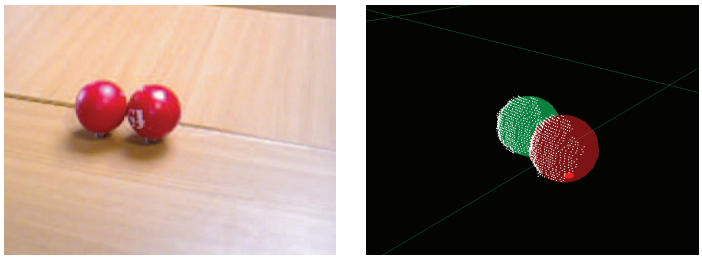
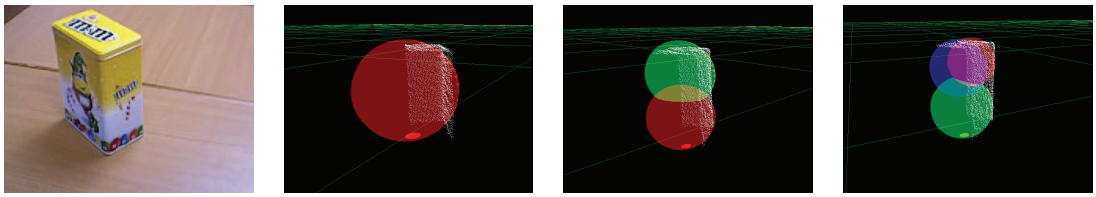


FIGURE 3.6.1 – Résultats obtenus sur deux balles de forme sphérique. A gauche, l'image RGB et à droite le nuage de points et les deux sphères estimées ($K = 2$).

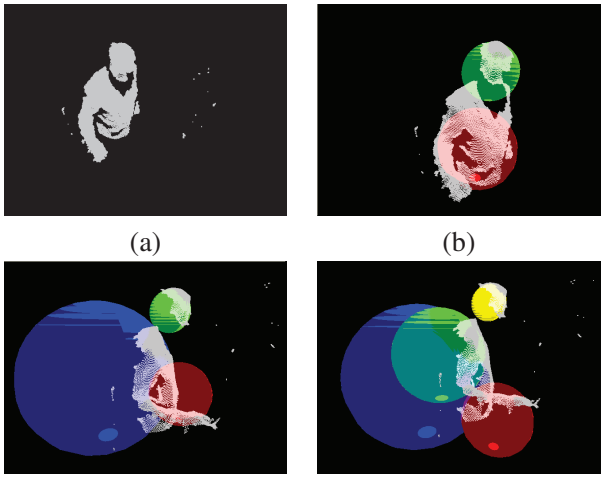
Dans une seconde expérimentation, nous évaluons la méthode dans le cas d'un objet non sphérique. Une boîte de forme rectangulaire est placée devant le capteur. Les résultats sont présentés Figure 3.6.2. Les centres des composantes optimisées sont positionnés à l'intérieur de l'objet. Les surfaces des sphères sont proches des points 3D observés. Dans cet exemple, les dimensions de la boîte peuvent laisser penser qu'une seule ou deux composantes suffiraient à modéliser correctement l'objet.



(a) (b) (c) (d)

FIGURE 3.6.2 – Résultats obtenus sur une boîte de forme rectangulaire (a) pour $K = 1$ (b), $K = 2$ (c) et $K = 3$ (d).

Enfin, la dernière expérimentation vise à tester l'algorithme sur des données plus complexes. Une personne positionnée sous le capteur est segmentée et modélisée par un mélange de K sphères. Le nombre de composantes est choisi manuellement. Les résultats sont représentés Figure 3.6.3. On remarque que $K = 2$ composantes ne permettent pas de modéliser raisonnablement la personne. En revanche, à partir de $K = 3$ composantes, la segmentation prend plus de sens. On distingue la classe du buste, de la tête et du bras. Notons que la modélisation de la tête est particulièrement intéressante puisque la forme de cette partie du corps est justement très proche de celle de la sphère.



(a) (b) (c) (d)

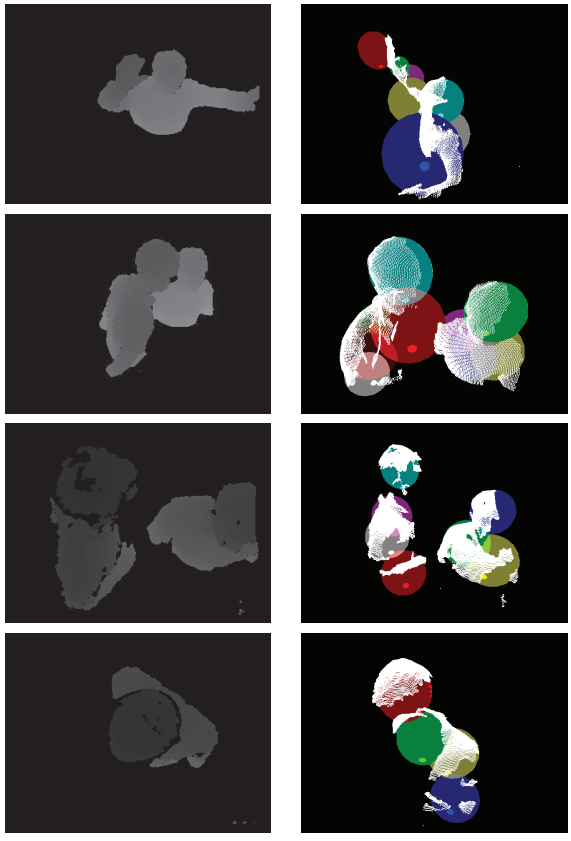
FIGURE 3.6.3 – Application de la méthode à un nuage de points représentant une personne (a) avec un modèle comportant $K = 2$ (b), $K = 3$ (c) et $K = 4$ (d) composantes.

3.6.2 Nombre de composantes inconnu

Dans la section précédente, le paramètre K était supposé connu à priori. De manière générale, le nombre de classes n'est pas connu et dépend de l'objet d'intérêt à analyser. D'un point de vue

pratique, le choix du nombre de composantes K reste donc un problème central. S'il est choisi trop faible, le modèle ne sera pas assez complexe pour correctement approcher la structure des données. Au contraire, s'il est choisi trop élevé, les données seront sur-segmentées ce qui donnera des sphères très similaires dont l'interprétation sera difficile. Nous adoptons donc une stratégie pas à pas ascendante. Le nombre de classes est initialisé à K_{min} et incrémenté jusqu'à K_{max}. Une étape donnée est acceptée si le ratio entre les vraisemblances courante et précédente est supérieur au seuil β. Dans le cas contraire, le processus est stoppé et la valeur de K est considérée comme optimale.

Un exemple de résultat est présenté Figure 3.6.4. Les nuages de points représentent de une à deux personnes et contiennent entre 8000 et 30000 points. Le capteur est positionné à une hauteur de trois mètres avec une vue quasi verticale et légèrement inclinée. Le nombre de classes initial a été fixé à $K_{min} = 2$ et incrémenté jusqu'à $K_{max} = 10$. Le seuil β a été fixé à 0.01. Les valeur de K choisies automatiquement sont plutôt bonnes et les nuages de points sont correctement segmentés. Toutefois, le temps de calcul est de l'ordre de plusieurs secondes par image. La complexité élevée de cette approche est dûe au fait que l'algorithme d'estimation est appliqué plusieurs fois pour chaque valeur de K successive. Nous proposerons dans la suite de la thèse une méthode plus efficace de choix automatique du paramètre K.



(a) (b)

FIGURE 3.6.4 – En (a) images de profondeur et en (b) nuages de points 3D et modèles estimés respectivement composés de 10, 9, 7 et 5 sphères.

3.7 Conclusion

Dans ce chapitre, nous avons proposé un nouveau modèle de mélange sphérique pour segmenter et modéliser un nuage de points 3D. Le modèle de mélange est basé sur une nouvelle densité de probabilité modélisant des points répartis autour d'une surface sphérique et appartenant à la famille des distributions elliptiques. Pour estimer les paramètres du modèle, un algorithme EM dans lequel un processus itératif a été ajouté est mis en œuvre.

Notre méthode d'estimation est simple à implémenter et son efficacité a été montrée en simulation. La méthode a été comparée à l'algorithme RANSAC. Les estimations fournies par notre méthode sont plus précises et présentent une variabilité plus faible. De plus, notre méthode repose sur des théorèmes de convergence assurant que les estimations s'améliorent avec la taille de l'échantillon. Au meilleur de notre connaissance, aucun résultat comparable n'est établit pour RANSAC.

Les résultats obtenus sont satisfaisants mais de nouvelles améliorations doivent encore être apportées pour la mise en œuvre pratique de la méthode. Le problème du choix du nombre de composantes a été soulevé et une première solution proposée. Cette solution ne s'avère cependant pas satisfaisante à cause de sa complexité trop importante. Nous explorerons d'autres approches convenant mieux à nos besoins dans les chapitres suivants.

Ce chapitre est centré sur l'étude du nouveau modèle et l'estimation de ses paramètres. Nous appliquerons la méthode développée dans le cadre d'une application concrète liée aux problématiques de Prynel dans la suite de la thèse.

Annexe

A Calcul de la constante de normalisation de la densité

Cette annexe détaille le calcul de la constante de normalisation K_d de la densité de probabilité f introduite en (3.3.5). Nous commençons tout d'abord par établir le lemme technique suivant utile pour simplifier les calculs.

Lemme A.1. *Soit pour $q \in \mathbb{N}$ et $\alpha > 0$,*

$$J_q(\alpha) \;=\; \int_0^\infty t^q \exp\left(-(t-1)^2/\left(2\alpha^2\right)\right) \mathrm{d}t. \tag{A1}$$

Alors $J_0(\alpha) = \alpha\sqrt{2\pi}\,(1-\Phi(-1/\alpha))$, $J_1(\alpha) = J_0(\alpha) + \alpha^2 \exp\left(-1/\left(2\alpha^2\right)\right)$, et pour $q \geq 2$,

$$J_q(\alpha) \;=\; J_{q-1}(\alpha) + (q-1)\alpha^2 J_{q-2}(\alpha). \tag{A2}$$

Démonstration. Posons $v(t) = -(t-1)^2/\left(2\alpha^2\right)$. Alors $v'(t) = -(t-1)/\alpha^2$ et on peut réécrire (A1) sous la forme

$$\begin{aligned} J_q(\alpha) &= \int_0^\infty \left(t^q - t^{q-1}\right) \exp\left(v(t)\right) \mathrm{d}t + J_{q-1}(\alpha) && \text{(A3)} \\ &= -\alpha^2 \int_0^\infty t^{q-1} v'(t) \exp\left(v(t)\right) \mathrm{d}t + J_{q-1}(\alpha). && \text{(A4)} \end{aligned}$$

Finalement, on obtient immédiatement (A2) par intégration par parties. Pour conclure la preuve, il est clair que $J_0(\alpha) = \alpha\sqrt{2\pi}\,(1-\Phi(-1/\alpha))$ et on obtient $J_1(\alpha)$ en utilisant (A4) avec $d = 1$. □

Nous considérons maintenant la constante K_d satisfaisant

$$\int_{\mathbb{R}^d} f(x)\mathrm{d}x \;=\; K_d \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\left(\sigma/r\right)^2}\left(\frac{\|x-\mu\|}{r} - 1\right)^2\right) \mathrm{d}x = 1 \tag{A5}$$

où $x = (x_1, \ldots, x_d)^T$ et $\mu = (\mu_1, \ldots, \mu_d)^T$. Le cas $d = 1$ est simple car l'intégrale est facilement calculable. Pour $d \geq 2$, nous utilisons le passage en coordonnées hypersphériques suivant

$$\begin{cases} x_1 = \mu_1 + \rho\, \sin(\theta_1)\, \ldots\, \sin(\theta_{d-1}) \\ x_2 = \mu_2 + \rho\, \sin(\theta_1)\, \ldots\, \sin(\theta_{d-2})\, \cos(\theta_{d-1}) \\ x_3 = \mu_3 + \rho\, \sin(\theta_1)\, \ldots\, \sin(\theta_{d-3})\, \cos(\theta_{d-2}) \\ \vdots \\ x_i = \mu_i + \rho\, \sin(\theta_1)\, \ldots\, \sin(\theta_{d-i})\, \cos(\theta_{d-i+1}) \\ \vdots \\ x_d = \mu_d + \rho\, \cos(\theta_1) \end{cases} \tag{A6}$$

où $\rho \geq 0$, $\theta_{d-1} \in [0, 2\pi]$, $\theta_1, \ldots, \theta_{d-2} \in [0, \pi]$ et dont la valeur absolue du déterminant du Jacobien est

$$\mid J \mid = \rho^{d-1} \prod_{j=1}^{d-2} \left(\sin(\theta_{d-j-1})\right)^j. \tag{A7}$$

Nous obtenons alors l'expression suivante

$$\begin{aligned} K_d^{-1} &= \int_{[0,\pi]^{d-2}} \int_{[0,2\pi]} \prod_{i=1}^{d-2} \left(\sin(\theta_{d-i-1})\right)^i d\theta_1 \dots d\theta_{d-1} \\ &\quad \times \int_0^{+\infty} \rho^{d-1} \exp\left(-\frac{1}{2\left(\sigma/r\right)^2}\left(\frac{\rho}{r}-1\right)^2\right) d\rho. \end{aligned} \tag{A8}$$

L'équation (A8) nous permet de mieux comprendre la décomposition polaire $X = \mu + r\,W\,U$ mentionnée en section 3.3.2. La première intégrale représente l'aire de la sphère unité de $\mathbb{R}^d$ et vaut $2\pi^{d/2} \,/\, \Gamma\left(d/2\right)$. La seconde intégrale est calculée par un simple changement de variable $u = \rho \,/\, r$. Finalement, on obtient

$$K_d^{-1} = \frac{2\pi^{d/2}}{\Gamma\left(d/2\right)} r^d \, J_{d-1}\left(\sigma/r\right) \tag{A9}$$

ce qui conclut le calcul de K_d. En particulier, dans les cas $d = 1, 2$ et 3, l'expression exacte de la constante est

$$\begin{aligned} K_1^{-1} &= 2\sigma\sqrt{2\pi}\left[1-\Phi\left(-\frac{r}{\sigma}\right)\right] \\ K_2^{-1} &= (2\pi)^{3/2}\sigma r\left[1-\Phi\left(-\frac{r}{\sigma}\right)+\frac{\sigma}{r\sqrt{2\pi}}\exp\left(-\frac{r^2}{2\sigma^2}\right)\right] \\ K_3^{-1} &= 2(2\pi)^{3/2}\sigma(r^2+\sigma^2)\left[1-\Phi\left(-\frac{r}{\sigma}\right)+\frac{\sigma r}{\sqrt{2\pi}(r^2+\sigma^2)}\exp\left(-\frac{r^2}{2\sigma^2}\right)\right]. \end{aligned}$$

On s'aperçoit que le ratio r/σ joue un rôle particulier dans ces expressions. Si on suppose que $r \gg \sigma$, alors on peut ignorer les termes en $\Phi(-r/\sigma)$ et $\exp\left(-r^2/(2\sigma^2)\right)$. En effet, si on analyse ces quantités comme cela a été fait Figure A1, on remarque qu'elles sont négligeables lorsque le ratio dépasse 5. Dans ce cas, la constante de normalisation peut être approchée et on a

$$K_1^{-1} = 2\sqrt{2\pi}\sigma \tag{A10}$$

$$K_2^{-1} = (2\pi)^{3/2}\sigma r \tag{A11}$$

$$K_3^{-1} = 2(2\pi)^{3/2}\sigma(r^2+\sigma^2). \tag{A12}$$

L'expression simplifiée de K_3 nous intéressera particulièrement pour la modélisation de données 3D.

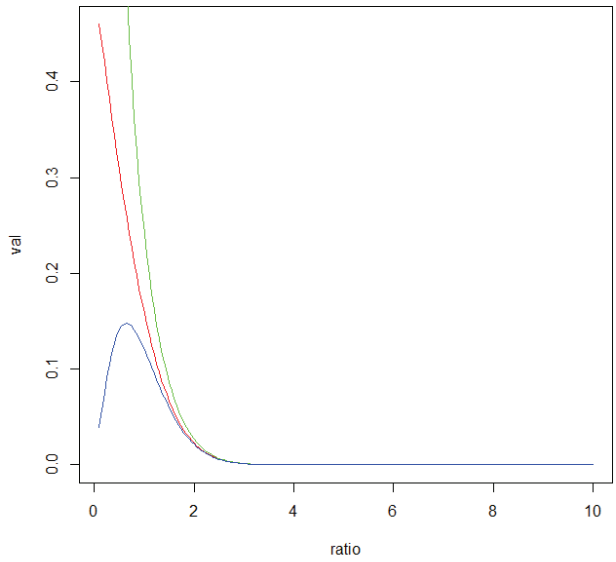


FIGURE A1 – Évolution des quantités $\Phi(-t)$ en rouge, $\frac{1}{t\sqrt{2\pi}}\exp\left(-t^2/2\right)$ en vert et $\frac{1}{(t+1/t)\sqrt{2\pi}}\exp\left(-t^2/2\right)$ en bleu.

B Preuve du Théorème 3.3.2

Cette annexe concerne la preuve de la seconde partie du Théorème 3.3.2 de la section 3.3.2. Soit Y un vecteur aléatoire de $\mathbb{R}^d$ de densité définie pour tout $y \in \mathbb{R}^d$ par

$$f_{\overline{\Omega}}(y) = K_{\overline{\Omega}} \exp\left(-\frac{1}{2\sigma^2}\left(\|y-\mu\| - r\right)^2\right) \mathbb{1}_{\{(y-\mu)\in\overline{\Omega}\}} = C_{\overline{\Omega}}\, f(y)\, \mathbb{1}_{\{(y-\mu)\in\overline{\Omega}\}} \tag{B1}$$

où $\overline{\Omega}$ est l'ensemble des points de $\mathbb{R}^d$ dont les coordonnées sphériques $(\rho, \theta_1, \ldots, \theta_{d-1})$ satisfont

$$\begin{cases} \rho \geq 0 \\ \theta_j \in \left[\theta_1^{(j)}, \theta_2^{(j)}\right] \text{ with } 0 \leq \theta_1^{(j)} < \theta_2^{(j)} \leq \pi \text{ for } 1 \leq j \leq d-2 \\ \theta_{d-1} \in \left[\theta_1^{(d-1)}, \theta_2^{(d-1)}\right] \text{ with } 0 \leq \theta_1^{(d-1)} < \theta_2^{(d-1)} \leq 2\pi \end{cases} \tag{B2}$$

où $(\theta_1^{(j)}, \theta_2^{(j)})_{j=1,\ldots,d-1}$ sont des valeurs d'angles données. Notons Θ l'ensemble

$$\Theta \;=\; \left[\theta_1^{(1)}, \theta_2^{(1)}\right] \times \ldots \times \left[\theta_1^{(d-1)}, \theta_2^{(d-1)}\right].$$

Soit Ω l'ensemble défini par l'intersection entre $\overline{\Omega}$ et la sphère unité d-dimensionnelle. Les éléments appartenant à Ω sont les éléments de $\overline{\Omega}$ dont la norme euclidienne vaut 1. On pose $K_{\overline{\Omega}} = \left(S_\Omega\, r^d\, J_{d-1}(\sigma/r)\right)^{-1}$ où S_Ω représente la surface de Ω, et on a $C_{\overline{\Omega}} = K_d^{-1}\, K_{\overline{\Omega}}$ avec K_d la constante de normalisation de la densité sphérique f.

L'objectif de cette annexe est de calculer l'espérance mathématique du vecteur aléatoire $Y^\star(a) = Y - a\,(Y-\mu)\,\|Y-\mu\|^{-1}$ pour tout $a > 0$. Pour cela, on calcule pour tout $i = 1, \ldots, d$, la $i^{\text{ème}}$ composante du vecteur $\mathbb{E}\,[Y^\star(a) - \mu]$. On a

$$\mathbb{E}\,[(Y^\star(a)-\mu)_i] \quad = \quad \int_{\mathbb{R}^d} \left((y_i - \mu_i) - a\,\frac{(y_i-\mu_i)}{\|y-\mu\|} \right) f_{\overline{\Omega}}(y)\,dy. \tag{B3}$$

Le calcul est effectué à l'aide du changement de variable hypersphérique (A6). Pour simplifier les notations, on écrit respectivement la i^{eme} équation de (A6) et la valeur absolue du déterminant du Jacobien donné en (A7) sous la forme $y_i = \mu_i + \rho\, t_i(\theta)$ et $|\,J\,| = \rho^{d-1}\,P(\theta)$, où $\theta = (\theta_1, \ldots, \theta_{d-1})^T$. Par conséquent, (B3) se réécrit

$$\mathbb{E}\,[(Y^\star(a)-\mu)_i] = K_{\overline{\Omega}} \int_0^{+\infty}\!\!\int_\Theta \rho^{d-1}(\rho - a)\,\exp\left(-\frac{(\rho-r)^2}{2\sigma^2}\right)\,t_i(\theta)\,P(\theta)d\rho\,d\theta$$

et on a alors

$$\mathbb{E}\,[(Y^\star(a)-\mu)_i] = K_{\overline{\Omega}}\,r^d\,\left(r\,J_d(\sigma/r) - a\,J_{d-1}(\sigma/r)\right)\,T_i \tag{B4}$$

où on a posé $T_i = \int_\Theta t_i(\theta)P(\theta)\,d\theta$. Remarquons que $S_\Omega = \int_\Theta P(\theta)\,d\theta$. Par suite, puisque $r^*\,J_{d-1}(\sigma/r)$ $= r\,J_d(\sigma/r)$, on obtient

$$\mathbb{E}\,[(Y^\star(a)-\mu)_i] \quad = \quad (r^* - a)\,S_\Omega^{-1}\,T_i \tag{B5}$$

et on en déduit que pour tout $a > 0$,

$$\mathbb{E}\,[Y^\star(a)] = \mu + (r^* - a)\;S_\Omega^{-1}\,T \tag{B6}$$

où $T = (T_1, \ldots, T_d)^T$. Finalement, il est clair que $\mathbb{E}\,[Y^\star(r^*)] = \mu$ et par une approche similaire, on montre facilement que $\|\mathbb{V}\text{ar}(Y^\star(r^*))\| < \infty$.

C Résultats de convergence

On présente dans cette annexe la démonstration des résultats de convergence énoncés en section 3.4.1. On considère un échantillon de variables aléatoires $(X_1, X_2, \cdots, X_n)$ indépendantes et identiquement distribuées de même densité f avec $d = 3$.

Pour tout i, on peut réécrire X_i sous la forme $X_i = \mu + r\,W_i\,U_i$ où W_i et U_i sont indépendantes, U_i est uniformément distribuée sur la surface de la sphère unité de $\mathbb{R}^d$ et W_i a pour densité (3.3.17). Tout d'abord, puisque pour tout $q \geq 0$,

$$\mathbb{E}\,[W^q] = J_{q+d-1}(\sigma/r)\,J_{d-1}^{-1}(\sigma/r) < \infty$$

où J_q est défini en (A1), et que $\|U\| = 1$, on déduit facilement que pour tout $q \geq 0$,

$$\mathbb{E}\,[\|X\|^q] < \infty.$$

Comme (X_i), (W_i) et (U_i) ont des moments finis d'ordre 2, on déduit immédiatement de la loi forte des grands nombres et des résultats des Théorèmes 3.3.1 et 3.3.2 que

$$\widehat{\mu}_n^{(0)} = \overline{X}_n = \frac{1}{n}\sum_{i=1}^n X_i \xrightarrow[n\to\infty]{p.s.} \mu \tag{C1}$$

$$\overline{R}_n = \frac{1}{n}\sum_{i=1}^n \|X_i-\mu\| = \frac{r}{n}\sum_{i=1}^n W_i \xrightarrow[n\to\infty]{p.s.} r^* = r\,\frac{J_3(\sigma/r)}{J_2(\sigma/r)} \tag{C2}$$

$$T_n = \frac{1}{n}\sum_{i=1}^n \|X_i-\mu\|^2 = \frac{r^2}{n}\sum_{i=1}^n W_i^2 \xrightarrow[n\to\infty]{p.s.} t^* = r^2\,\frac{J_4(\sigma/r)}{J_2(\sigma/r)} \tag{C3}$$

$$\overline{U}_n = \frac{1}{n}\sum_{i=1}^n \frac{(X_i-\mu)}{\|X_i-\mu\|} = \frac{1}{n}\sum_{i=1}^n U_i \xrightarrow[n\to\infty]{p.s.} 0. \tag{C4}$$

De plus, le théorème de Hartman-Winters ([183]) énonçant la loi du logarithme itéré pour les suites de variables aléatoires indépendantes et identiquement distribuées s'applique et on en déduit que

$$\left\{\begin{array}{ll} \left\|\widehat{\mu}_n^{(0)}-\mu\right\| = O\left(a_n\right), & \left|\overline{R}_n - r^*\right| = O\left(a_n\right) \\ \left|T_n - t^*\right| = O\left(a_n\right), & \left\|\overline{U}_n\right\| = O\left(a_n\right) \end{array}\right. \quad \text{p.s.} \tag{C5}$$

où $a_n = \sqrt{\text{loglog}\, n/n}$. Étudions maintenant la convergence de $\widehat{r}_n^{(0)}$. La décomposition

$$\widehat{r}_n^{(0)} - r^* = \frac{1}{n}\sum_{i=1}^n \|X_i-\mu\| - r^* + \frac{1}{n}\sum_{i=1}^n \left(\left\|X_i-\widehat{\mu}_n^{(0)}\right\| - \|X_i-\mu\|\right), \tag{C6}$$

nous permet de déduire que $\left|\widehat{r}_n^{(0)} - r^*\right| \leq \left|\overline{R}_n - r^*\right| + \left\|\widehat{\mu}_n^{(0)}-\mu\right\|$. En utilisant (C5), on obtient

$$\left|\widehat{r}_n^{(0)} - r^*\right| = O\left(a_n\right) \quad \text{p.s.} \tag{C7}$$

Étudions maintenant la convergence de $\widehat{\sigma}_n^2$. Tout d'abord, on remarque que

$$\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{i=1}^n \left(\left\|X_i-\widehat{\mu}_n^{(0)}\right\| - \widehat{r}_n^{(0)}\right)^2 = \frac{1}{n}\sum_{i=1}^n \left\|X_i-\widehat{\mu}_n^{(0)}\right\|^2 - (\widehat{r}_n^{(0)})^2. \tag{C8}$$

On peut donc écrire la décomposition

$$\widehat{\sigma}_n^2 - \widetilde{\sigma}^2 = T_n - t^* + \frac{1}{n}\sum_{i=1}^n \left(\left\|X_i-\widehat{\mu}_n^{(0)}\right\|^2 - \|X_i-\mu\|^2\right) - \left((\widehat{r}_n^{(0)})^2 - (r^*)^2\right)$$

avec $\widetilde{\sigma}^2 = t^* - (r^*)^2$ ce qui se réduit à $\sigma^2(1+3(\sigma/r)^4)/(1+(\sigma/r)^2)^2 \simeq \sigma^2$ lorsque $r \gg \sigma$. On déduit ensuite que

$$\left|\widehat{\sigma}_n^2 - \widetilde{\sigma}^2\right| \leq |T_n - t^*| + \left\|\widehat{\mu}_n^{(0)}-\mu\right\|\left(2\overline{R}_n + \left\|\widehat{\mu}_n^{(0)}-\mu\right\|\right) + \left|(\widehat{r}_n^{(0)})^2 - (r^*)^2\right|$$

et en utilisant (C5) et (C7), on obtient $\left|\widehat{\sigma}_n^2 - \widetilde{\sigma}^2\right| = O(a_n)$ p.s. ce qui conclut la preuve du Théorème 3.4.1.

Intéressons nous maintenant à la preuve du Théorème 3.4.2. On peut réécrire $(\widehat{\mu}_n^\star - \mu)$ sous la forme

$$\widehat{\mu}_n^\star - \mu = \left(\overline{X}_n - \mu\right) - \left(\widehat{r}_n^{(0)} - r^*\right) Z_n - \frac{r^*}{n}\left(M_n + Q_n\right) \tag{C9}$$

avec $Z_n = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} \frac{(X_i - \widehat{\mu}_{i-1}^{(0)})}{\left\|X_i - \widehat{\mu}_{i-1}^{(0)}\right\|}$ et

$$M_n = \sum_{i=1}^{n}\left(\frac{(X_i - \widehat{\mu}_{i-1}^{(0)})}{\left\|X_i - \widehat{\mu}_{i-1}^{(0)}\right\|} - \mathbb{E}\left[\frac{(X_i - \widehat{\mu}_{i-1}^{(0)})}{\left\|X_i - \widehat{\mu}_{i-1}^{(0)}\right\|}\Big|\mathcal{F}_{i-1}\right]\right) \tag{C10}$$

$$Q_n = \sum_{i=1}^{n}\mathbb{E}\left[\frac{(X_i - \widehat{\mu}_{i-1}^{(0)})}{\left\|X_i - \widehat{\mu}_{i-1}^{(0)}\right\|}\Big|\mathcal{F}_{i-1}\right] = \sum_{i=1}^{n}\varphi(\widehat{\mu}_{i-1}^{(0)}) \tag{C11}$$

où la fonction φ est définie pour tout $t \in \mathbb{R}^3$ par $\varphi(t) = \mathbb{E}\left[(X_1 - t)/\left\|X_1 - t\right\|\right]$. Il est clair que $\|\varphi(t)\| \leq 1$ pour tout t, et il n'est pas difficile de montrer que chaque composante de φ est continue en utilisant le théorème de convergence dominée.

Pour tout $u \in \mathbb{R}^3$ et $n \geq 1$, on pose $M_n(u) = u^T M_n$. Pour tout $n \geq 1$, on note $\mathcal{F}_n$ la σ-algèbre des évènements se produisant jusqu'à l'instant n, i.e. $\mathcal{F}_n = \sigma(X_1, \ldots, X_n)$. Puisque pour tout $n \geq 1$, $\mathbb{E}\left[M_n^2(u)\right] < \infty$ et $\mathbb{E}\left[M_{n+1}(u) \mid \mathcal{F}_n\right] = M_n(u)$, alors $(M_n(u))$ est une martingale de carré intégrable adaptée à la filtration $\mathbb{F} = (\mathcal{F}_n)_{n\geq 1}$. Calculons alors son processus croissant $<M(u)>_n$. On a

$$<M(u)>_n = \sum_{k=1}^{n}\mathbb{E}\left[\left(M_k(u) - M_{k-1}(u)\right)^2 \mid \mathcal{F}_{k-1}\right] \tag{C12}$$

$$\leq \sum_{i=1}^{n}\mathbb{E}\left[\frac{\left((X_i - \widehat{\mu}_{i-1})\ u^T\right)^2}{\left\|X_i - \widehat{\mu}_{i-1}\right\|^2} \mid \mathcal{F}_{i-1}\right] \tag{C13}$$

$$\leq n\left\|u\right\|^2. \tag{C14}$$

En utilisant alors la loi forte des grands nombres pour les martingales (voir le Théorème 4.3.16 de [74]), on obtient que pour tout u, $|M_n(u)| = o(n)$ p.s. Par conséquent, on en déduit que

$$\frac{1}{n} M_n \xrightarrow[n\to\infty]{p.s.} 0. \tag{C15}$$

Chaque composante de la fonction φ est continue et $\widehat{\mu}_n^{(0)} \xrightarrow[n\to\infty]{p.s.} \mu$. Il vient alors immédiatement que $\varphi(\widehat{\mu}_n^{(0)}) \xrightarrow[n\to\infty]{p.s.} \varphi(\mu) = 0$. Finalement, comme $\|Z_n\| \leq 1$, on conclut que

$$\left\|\widehat{\mu}_n^\star - \mu\right\| \leq \left\|\overline{X}_n - \mu\right\| + \left|\widehat{r}_n^{(0)} - r^*\right| + r^*\left(\|M_n\| + \|Q_n\|\right)/n \tag{C16}$$

et en combinant les différents résultats, on obtient la convergence presque sûre de $\widehat{\mu}_n^\star$ vers μ.

D Maximisation de la log-vraisemblance complétée

Dans cette annexe, nous justifions le choix des estimateurs utilisés dans l'étape M de l'algorithme EM décrit en 3.4.2.3. L'étape M consiste à maximiser la fonction Q. Pour faire cela, nous calculons les dérivées partielles par rapport aux paramètres μ_l, r_l, σ_l de l'expression

$$Q(\Theta) = \sum_{i=1}^{n}\sum_{k=1}^{K} t_{ik} \left(\log(\pi_k) + \log\left(f(x_i, \theta_k)\right)\right) \tag{D1}$$

avec $f(x;\theta_k) = \dfrac{1}{2(2\pi)^{3/2}\sigma_k\left(r_k^2+\sigma_k^2\right)} \exp\left(-\dfrac{1}{2\sigma_k^2}\left(\|x-\mu_k\| - r_k\right)^2\right)$ et $r_k \gg \sigma_k$.
La dérivée partielle de Q par rapport à μ_l vaut

$$\begin{aligned}\frac{\partial Q(\Theta)}{\partial \mu_\ell} &= \sum_{i=1}^{n} t_{i\ell}\left(\frac{1}{\sigma_\ell^2}(x_i-\mu_\ell)\left(1-\frac{r_\ell}{\|x_i-\mu_\ell\|}\right)\right) \\ &= \sum_{i=1}^{n} t_{i\ell}(x_i-\mu_\ell) - r_\ell\sum_{i=1}^{n} t_{i\ell}\frac{(x_i-\mu_\ell)}{\|x_i-\mu_\ell\|}.\end{aligned} \tag{D2}$$

La dérivée partielle de Q par rapport à r_ℓ vaut

$$\begin{aligned}\frac{\partial Q(\Theta)}{\partial r_\ell} &= \sum_{i=1}^{n} t_{i\ell}\left(\frac{1}{\sigma_\ell^2}\left(\|x_i-\mu_\ell\|-r_\ell\right) - \frac{2r_\ell}{r_\ell^2+\sigma_\ell^2}\right) \\ &= \frac{1}{\sigma_\ell^2}\sum_{i=1}^{n} t_{i\ell}\left(\|x_i-\mu_\ell\| - r_\ell \times \frac{3(\sigma_\ell/r_\ell)^2+1}{(\sigma_\ell/r_\ell)^2+1}\right)\end{aligned} \tag{D3}$$

et la dérivée partielle de Q par rapport à σ_l^2 s'écrit

$$\frac{\partial Q(\Theta)}{\partial \sigma_\ell} = \sum_{i=1}^{n} t_{i\ell}\left(\frac{1}{\sigma_\ell^3}\left(\|x_i-\mu_\ell\|-r_\ell\right)^2 - \frac{1}{\sigma_\ell} \times \frac{3(\sigma_\ell/r_\ell)^2+1}{(\sigma_\ell/r_\ell)^2+1}\right). \tag{D4}$$

Il n'est pas possible de résoudre analytiquement le problème de maximisation. Toutefois, si l'on suppose que $r_\ell \gg \sigma_\ell$, alors $\left(3(\sigma_\ell/r_\ell)^2+1\right) / \left((\sigma_\ell/r_\ell)^2+1\right) \simeq 1$ et les dérivées partielles (D3) et (D4) deviennent

$$\frac{\partial Q(\Theta)}{\partial r_\ell} = \frac{1}{\sigma_\ell^2}\sum_{i=1}^{n} t_{i\ell}\left(\|x_i-\mu_\ell\|-r_\ell\right) \tag{D5}$$

$$\frac{\partial Q(\Theta)}{\partial \sigma_\ell} = \sum_{i=1}^{n} t_{i\ell}\left(\frac{1}{\sigma_\ell^3}\left(\|x_i-\mu_\ell\|-r_\ell\right)^2 - \frac{1}{\sigma_\ell}\right). \tag{D6}$$

Avec ces approximations, la résolution du problème de maximisation conduit aux estimateurs présentés en 3.4.2.3 pour lesquels la convergence presque sûre a été établie. La courbe de la quantité $(1+3\,t^2)/(1+t^2)$ en fonction de t est représentée Figure D1. L'approximation effectuée est raisonnable si $t > 10$.

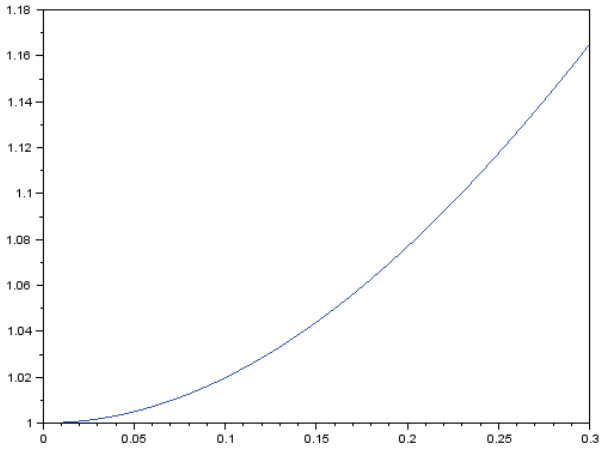


FIGURE D1 – Valeur du terme $\dfrac{1+3t^2}{1+t^2}$ en fonction de t.

L'estimateur des poids π_ℓ est le même que celui classiquement utilisé pour les modèles de mélange gaussiens (voir [128]). Les poids n'apparaissant que dans le premier terme de la quantité à maximiser (D1), le second terme peut être ignoré. L'expression du poids π_ℓ peut être obtenue par la méthode des multiplicateurs de Lagrange sous la contrainte $\sum_{k=1}^{K} \pi_k = 1$. Ceci conduit à maximiser la quantité

$$\sum_{i=1}^{n}\sum_{k=1}^{K} t_{ik} \log(\pi_k) - \lambda\left(\sum_{k=1}^{K}\pi_k - 1\right) \tag{D7}$$

où $\lambda \in \mathbb{R}$ est un multiplicateur de Lagrange et donc à résoudre le système d'équations

$$\begin{cases} \sum_{i=1}^{n} t_{i\ell} = \lambda\,\pi_\ell \\ \sum_{k=1}^{K} \pi_k = 1 \end{cases}. \tag{D8}$$

Une sommation d'indice ℓ sur chaque membre de la première équation donne $\lambda = n$ et conduit à

$$\pi_\ell = \frac{1}{n}\sum_{i=1}^{n} t_{i\ell}. \tag{D9}$$

E Méthode de simulation de la nouvelle loi

Dans cette annexe, nous décrivons le procédé d'obtention d'une réalisation d'un vecteur aléatoire X de densité f dans le cas $d = 3$. La méthode de simulation sera utilisée pour générer des données sur lesquelles la méthode d'estimation sera expérimentée. Il existe plusieurs méthodes classiques pour simuler une loi de probabilité, comme la méthode de la transformée inverse (voir par exemple [71]) ou la méthode de rejet (voir par exemple [71, 28]). Nous avons choisi d'utiliser la méthode de rejet pour simuler la composante radiale (W) et la méthode d'inversion pour simuler des observations uniformément réparties sur la sphère unité. Les méthodes de rejet et d'inversion sont présentées de manière générale en E1 et E2 avant d'être appliquées à la nouvelle loi en E3.

E1 Méthode de rejet

La méthode de rejet permet de simuler une loi lorsque cette dernière admet une densité de probabilité complexe ou non standard. Elle nécessite de majorer la densité de probabilité de la loi par celle d'une autre loi que l'on sait simuler. Un procédé d'acceptation-rejet sélectionne ensuite les observations à conserver.

Soit X un vecteur aléatoire de dimension d et de densité de probabilité f que l'on cherche à simuler. On suppose que l'on connaît un vecteur aléatoire Y de densité g sur $\mathbb{R}^d$ facilement simulable et une constante $c \in \mathbb{R}$ tels que

$$\forall x \in \mathbb{R}^d, f(x) \leq c\, g(x). \tag{E1}$$

La méthode de rejet est basée sur le résultat suivant.

Théorème E.1. *Soit X une variable aléatoire de densité f, Y de densité g et U uniforme sur l'intervalle $[0,1]$. Alors la variable aléatoire $Z = \left\{ Y,\, U \leq \dfrac{f(Y)}{c\, g(Y)} \right\}$ suit la loi de densité f.*

La méthode de simulation se résume donc à la génération indépendante de réalisations de Y et de U. La réalisation de Y est acceptée comme étant une réalisation de X si elle vérifie l'inégalité du théorème. La méthode est résumée en Algorithme E.2.

L'algorithme nécessite de faire un certain nombre de tirages avant qu'une réalisation ne soit acceptée. Son efficacité dépend donc de ce nombre, noté K, qui est une variable aléatoire. On peut montrer que la variable K suit une loi géométrique de paramètre $1/c$. La probabilité d'accepter une réalisation au bout de k tirages vaut par conséquent

$$P\left[K = k\right] = \left(1 - \frac{1}{c}\right)^{k-1} \frac{1}{c}\, 1_{k \geq 1}. \tag{E2}$$

La probabilité d'acceptation d'un tirage vaut donc $1/c$ et le nombre d'itérations moyen à effectuer est $\mathbb{E}\left[K\right] = c$. Le nombre de tirages à effectuer pour accepter une réalisation dépend donc directement de la majoration de f. Une majoration plus fine améliorera l'efficacité de la méthode.

Algorithm E.1 Méthode de rejet

Require: Densité f de la loi à simuler, densité g et constante c tels que $\forall x \in \mathbb{R}^d, f(x) \leq c\, g(x)$

1: **repeat**
2: Générer une réalisation u de loi $U_{[0,1]}$
3: Générer une réalisation y de loi de densité g
4: **until** $u > \dfrac{f(y)}{c\, g(y)}$
5: Accepter y comme réalisation de la loi de densité f

E2 Méthode de la transformée inverse

La méthode de la transformée inverse est une méthode permettant de simuler des réalisation d'une variable aléatoire X à partir de sa fonction de répartition F_X. La méthode est basée sur le résultat du théorème suivant tiré de [71].

Théorème E.2. *Soit F une fonction de répartition continue sur $\mathbb{R}$ et d'inverse F^{-1} définie par*

$$F^{-1}(u) = \inf\{x, F(x) = u, 0 < u < 1\}.$$

Si U est une variable aléatoire de loi uniforme sur $[0,1]$, alors $F^{-1}(U)$ a pour fonction de répartition F. Si X a pour fonction de répartition F, alors $F(X)$ est uniformément distribuée sur $[0,1]$.

Démonstration. Le premier résultat s'obtient en calculant la fonction de répartition de $F_X^{-1}(U)$. Pour tout $x \in \mathbb{R}$,

$$\begin{aligned} P(F_X^{-1}(U) \leq x) &= P(U \leq F_X(x)) && \text{(E3)} \\ &= F_X(x). && \text{(E4)} \end{aligned}$$

Le second résultat est démontré en calculant la fonction de répartition de la variable aléatoire $U = F_X(x)$,

$$\begin{aligned} F_U(u) = P(F_X(X) \leq u) &= P(X \leq F_X^{-1}(u) && \text{(E5)} \\ &= F_X(F_X^{-1}(u)) && \text{(E6)} \\ &= u. && \text{(E7)} \end{aligned}$$

On retrouve la fonction de répartition de la loi uniforme sur l'intervalle $[0,1]$. □

La simulation de réalisations de X passe donc par la simulation de réalisations de U auxquelles la fonction F^{-1} est appliquée.

Algorithm E.2 Méthode de la transformée inverse

Require: Fonction de répartition inverse F^{-1} de la loi à simuler (variable aléatoire X) et nombre de réalisations n à générer.

```
i = 0
for i=0 to i < n do
    Générer une réalisation u de loi U_[0,1].
    Calculer une réalisation de X comme étant y = F^{-1}(u).
end for
```

E3 Application à la nouvelle densité

Dans cette partie, nous appliquons les méthodes de simulation présentées dans les parties précédentes pour simuler des réalisations de la variable aléatoire X définie en (3.3.5) et de paramètres (μ, r, σ). Nous utilisons pour cela la décomposition établie en (3.3.16)

$$X = \mu + r\, W\, U \tag{E8}$$

où W est une variable aléatoire réelle positive indépendante de U uniformément répartie sur la sphère unité de $\mathbb{R}^d$. Nous avons vu en (3.3.17) que la densité de W s'écrit pour $t \geq 0$

$$\varphi(t) = \frac{1}{J_{d-1}(\alpha)}\, t^{d-1}\, \exp\left(-\frac{(t-1)^2}{2\alpha^2}\right) \tag{E9}$$

où $\alpha = \sigma/r$.

La variable aléatoire X peut donc être simulée en simulant indépendamment des réalisations des variables aléatoires W et U. Des réalisations de W sont obtenues par la méthode de rejet et U est simulée par une méthode d'inversion via les coordonnées polaires ([71]).

Simulation de U.

Nous ne traitons ici que le cas $d = 3$ qui correspond à notre cadre d'application. La simulation d'observations uniformément réparties sur la sphère unité de $\mathbb{R}^3$ est réalisée avec la méthode de la transformée inverse via les coordonnées sphériques. La densité de probabilité uniforme sur la sphère de centre $(0, 0, 0)^T$ et de rayon $r > 0$ de $\mathbb{R}^3$ s'écrit

$$f(x, y, z) = \frac{1}{4\pi}. \tag{E10}$$

Après le passage en coordonnées sphériques suivant

$$\begin{cases} x = \rho \cos(\theta) \sin(\phi) \\ y = \rho \sin(\theta) \sin(\phi) \\ z = \rho \cos(\phi) \end{cases}$$

avec $\rho = r$, $\theta \in [0, \pi]$ et $\phi \in [0, 2\pi]$ et dont la valeur absolue du Jacobien vaut $\mid J \mid = \rho^2 \, sin(\theta)$. Par conséquent, la densité uniforme sur la sphère unité de $\mathbb{R}^3$ s'écrit

$$f(\theta, \phi) = \frac{\sin(\theta)}{4\pi} \, 1_{\theta \in [0,\pi]} \, 1_{\phi \in [0,2\pi]}. \tag{E11}$$

Les distributions marginales ont donc pour expression

$$f_\theta(t) = \frac{\sin(t)}{2} \tag{E12}$$

$$f_\phi(t) = \frac{1}{2\pi} \tag{E13}$$

et les fonctions de répartition correspondantes s'écrivent

$$F_\theta(y) = \frac{1 - \cos(y)}{2} \tag{E14}$$

$$F_\phi(y) = \frac{y}{2\pi}. \tag{E15}$$

La méthode de la transformée inverse consiste alors à simuler des réalisations de U par les relations

$$\begin{cases} \theta = \cos^{-1}(1 - 2u) \\ \phi = 2\pi v \end{cases} \tag{E16}$$

où u et v sont des réalisations de la loi uniforme sur l'intervalle $[0, 1]$.

Simulation de W.

Pour appliquer la méthode de rejet à la variable aléatoire W, nous devons majorer la densité φ par une densité de probabilité que l'on est capable de simuler. Pour cela, nous rappelons le lemme suivant.

Lemme E.1. *Soit d un entier positif ou nul. On a l'inégalité suivante*

$$\exp(dt) \geq t^d \, \forall t \in \mathbb{R}. \tag{E17}$$

Le Lemme E.1 nous permet d'écrire les inégalités suivantes

$$\begin{aligned}
\varphi(t) &\leq \frac{1}{J_{d-1}(\alpha)} \exp((d-1)t) \exp\left(-\frac{(t-1)^2}{2\alpha^2}\right) && \text{(E18)}\\
&\leq \frac{1}{J_{d-1}(\alpha)} \exp\left((d-1)t - \frac{(t-1)^2}{2\alpha^2}\right) && \text{(E19)}\\
&\leq \frac{1}{J_{d-1}(\alpha)} \exp\left(\frac{2(d-1)t\alpha^2 - t^2 + 2t - 1}{2\alpha^2}\right) && \text{(E20)}\\
&\leq \frac{1}{J_{d-1}(\alpha)} \exp\left(-\frac{\left(t-(1+(d-1)\alpha^2)\right)^2 - 2(d-1)\alpha^2 - (d-1)^2\alpha^4}{2\alpha^2}\right) && \text{(E21)}\\
&\leq \frac{1}{J_{d-1}(\alpha)} \exp\left((d-1) + \frac{(d-1)^2\alpha^2}{2}\right) \exp\left(-\frac{\left(t-(1+(d-1)\alpha^2)\right)^2}{2\alpha^2}\right) && \text{(E22)}\\
&\leq c\,\mathcal{N}\left(1+(d-1)\alpha^2, \alpha\right) && \text{(E23)}
\end{aligned}$$

où

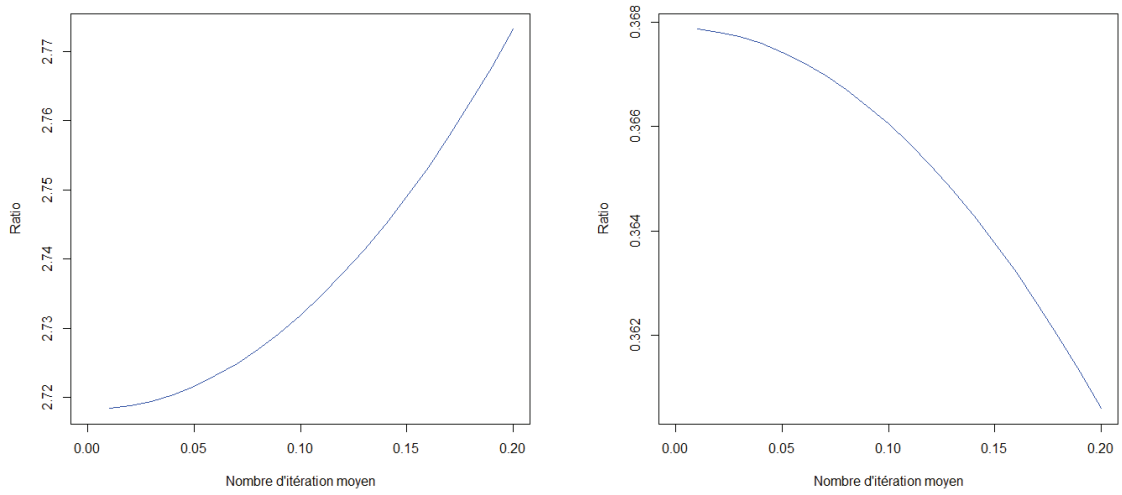
$$c = \frac{\alpha\sqrt{2\pi}\exp\left((d-1) + \frac{(d-1)^2\alpha^2}{2}\right)}{J_{d-1}(\alpha)}. \tag{E24}$$

La méthode de rejet que nous appliquons est donc basée sur la majoration $\varphi(t) \leq c\, g(t)$ où c est la constante définie en (E24) et g la densité d'une gaussienne de centre $1 + (d-1)\alpha^2$ et de variance α^2.

La majoration proposée conduit à une certaine efficacité de la méthode dépendant de la constante c. Cette constante dépend elle même du ratio α. La Figure E1 représente le nombre d'itérations moyen théorique nécessaire pour accepter une réalisation et la probabilité d'acceptation d'une réalisation en fonction de α dans les cas $d = 2$ et $d = 3$. Notons que dans ces travaux, le temps d'obtention d'un échantillon n'est pas un problème central. La majoration du Lemme E.1 est large, surtout pour des valeurs de d élevées. Dans notre cas, nous ne simulerons des sphères que pour $d = 3$ puisque nos données réelles sont des points 3D.

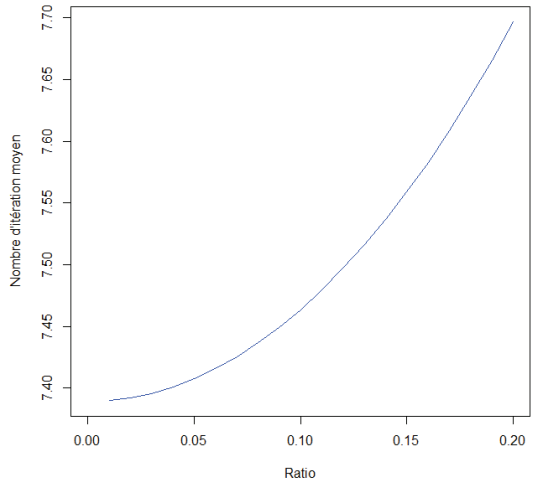
Résultats de la simulation de X.

Nous avons utilisé la méthode pour simuler des sphères de dimension 2 et 3. Les résultats sont représentés Figure E2. Les réalisations sont bien situées autour la surface de la sphère. La valeur du paramètre σ^2 contrôle la dispersion des observations. Lors de ces simulations, nous nous sommes également intéressés au nombre moyen de tirages nécessaires pour accepter une réalisation. La méthode a été appliquée pour simuler $n = 100000$ réalisations et le nombre de tirages moyen effectués a été calculé. Les valeurs obtenues sont données Table E1. Les nombres de tirages moyens constatés sont proches des valeurs théoriques attendues.

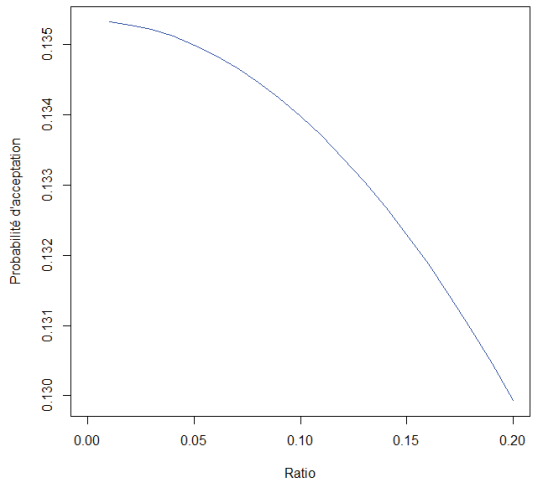


(a)

(b)



(c)

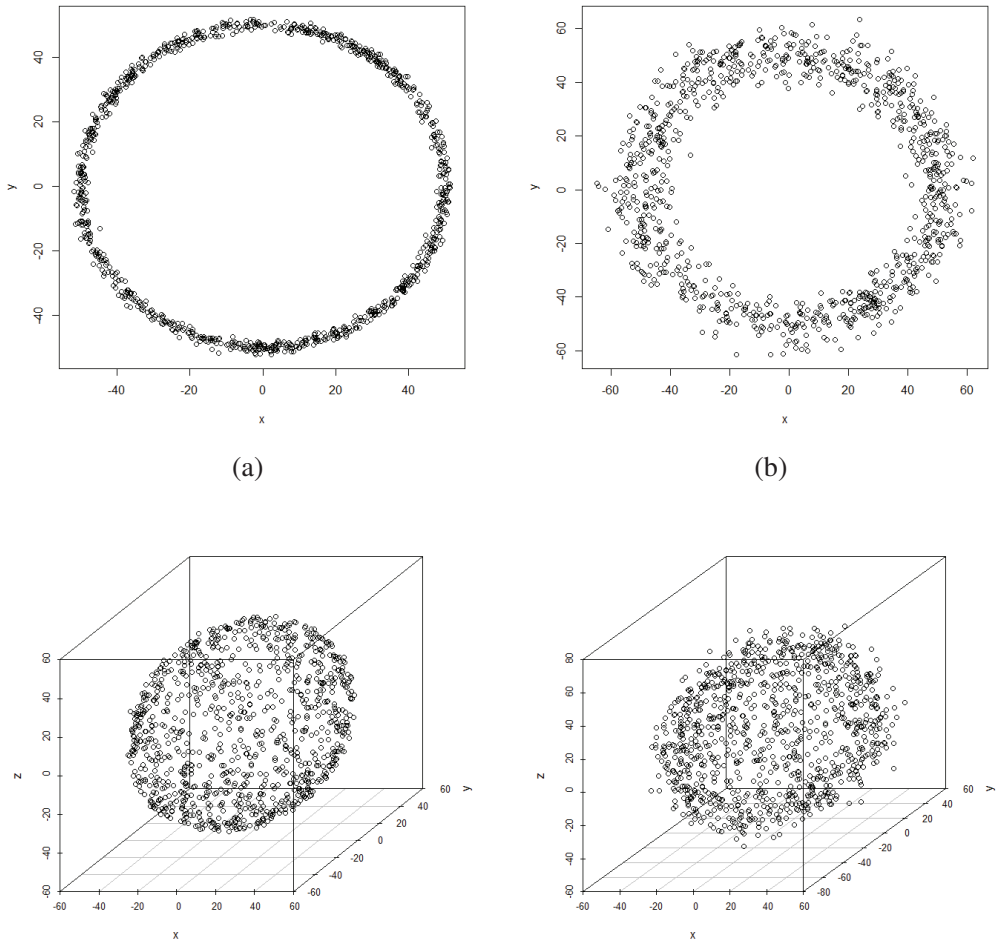


(d)

FIGURE E1 – Nombre de tirage moyen théorique pour obtenir une réalisation (a, c) et probabilité d'acceptation d'une réalisation (b, d). Les courbes (a, b) concernent le cas $d = 2$ et (c, d) le cas $d = 3$.

d	2	2	3	3
r	50	50	50	50
σ	1	5	1	5
$\overline{K}$	2.722	2.744	7.395	7.467
$\mathbb{E}[K]$	2.719	2.732	7.392	7.464

TABLE E1 – Nombres moyen et exact de tirages nécessaires à l'acceptation d'une réalisation dans différents cas. Les valeurs ont été obtenues sur une base de $n = 100000$ réalisations.



(a) (b)

(c) (d)

FIGURE E2 – Ensemble de $n = 1000$ réalisations de la loi en dimensions 2 et 3. Les sphères ont pour centre l'origine et pour rayon $r = 50$. Les écart-types sont fixés à $\sigma = 1$ (a, c) et $\sigma = 5$ (b, d).

Chapitre 4

Détection et modélisation de têtes dans un nuage de points 3D

Sommaire

4.1 Introduction

Dans ce chapitre, nous nous intéressons au problème de détection et de modélisation de têtes dans un nuage de points 3D. Le but est de détecter et d'identifier les principales caractéristiques des têtes présentes dans la scène. Ce chapitre traite de la mise en œuvre pratique des algorithmes d'estimation qui ont été développés au Chapitre 3.

La détection de têtes dans des images est une composante centrale de nombreux systèmes de vision par ordinateur comme le comptage de personnes, la vidéo surveillance, la biométrie, la vidéo conférence, la réalité virtuelle ou encore l'interaction homme machine. Ce travail s'inscrit dans le cadre d'un système de comptage de personnes pour l'accès sécurisé à une zone. Généralement, les systèmes existants sont conçus pour fonctionner dans une configuration particulière (angle de vue et position du capteur). La configuration de notre système est telle que le capteur est placé à une hauteur d'environs 2,50 mètres avec un angle de vue quasi vertical.

Une première catégorie d'algorithmes a été conçue pour traiter des images couleur. Lorsque le visage est visible dans l'image, les têtes peuvent être détectées par un algorithme de détection de visages ([196]). Une ellipse représentant la tête est suivie en utilisant des gradients d'intensité et des histogrammes de couleur dans [27]. Dans [1], la tête est modélisée par un ellipsoïde 3D qui est suivie dans une séquence d'images grâce à un filtre à particules. Les travaux cités dans ce paragraphe supposent que les personnes sont vues de face.

Les capteurs 3D fournissent une information de distance pour chaque pixel de l'image pouvant être convertie en un nuage de points 3D représentant la scène. Récemment, le lancement du capteur Kinect pour la plateforme de jeu Microsoft Xbox ([60]) a démocratisé l'utilisation de tels capteurs. La caméra capture des images RGB et de profondeur de dimensions maximales 640×480 à une fréquence de 30 images par seconde. La mesure de la distance est basée sur la déformation d'un motif infrarouge projeté sur la scène. Cette technologie présente un certain nombre d'avantages comparé à d'autres systèmes comme par exemple l'invariance à l'illumination, à la texture et la charge de calcul nécessaire à la détermination de la profondeur. Toutefois, la distance de fonctionnement est limitée de 70 centimètres et 5 mètres. Le système donne une information géométrique en plus de l'information colorimétrique. Les principes de fonctionnement des capteurs 3D ont été présentés au Chapitre 1 section 1.2. Dans ces travaux, nous ne considérerons que les images de profondeur pour les différents traitements. Les images couleur ne sont pas prises en compte.

La disponibilité de l'information 3D a été une opportunité pour améliorer les résultats de la détection de têtes. Certains travaux n'utilisent que l'image de profondeur et d'autres le nuage de points 3D organisé ou non. Le problème d'ajustement d'ellipsoïdes est décrit dans [91] et est utilisé pour suivre les mouvements de la tête. Dans [44], une ellipse est comparée au masque de mouvement. Cette ellipse est dimensionnée selon la distance de l'objet à la caméra. L'approche présentée dans [204] consiste à extraire des maxima locaux dans l'image de profondeur via un algorithme de Water Filling. Un modèle sphérique composé de zones positives et négatives est utilisé dans [187] pour déterminer de potentielles têtes. Dans [87], les images de profondeur d'une base de donnée sont sélectionnées en comparant des vecteurs caractéristiques et interpolées pour estimer la position de la tête. Une autre méthode proposée dans [80] est basée sur des des forêts aléatoires et permet de trouver la position et l'orientation des têtes. Notons que dans la plupart des méthodes, la forme ellipsoidale de la tête a été exploitée.

Dans ces travaux, nous nous basons sur un modèle probabiliste permettant de modéliser et de segmenter des nuages de points 3D. L'approche adoptée s'appuie sur le modèle de mélange sphérique introduit au Chapitre 3. Dans notre cadre d'étude, ce type de modèle présente l'avantage d'être capable de modéliser des parties du corps proches ou en contact. La mise en œuvre de la méthode sur des données réelles nécessite toutefois certaines adaptations pour être réellement utilisable. L'objet de ce chapitre est donc la mise à contribution de la méthode de segmentation et de modélisation d'un nuage de points 3D par des sphères au problème de détection de têtes.

Une fois le modèle de mélange correctement estimé et les données modélisées par un ensemble de sphères, nous cherchons à déterminer quelles composantes représentent ou non des têtes. Pour cela, un ensemble de règles de décision simples sont mises en place. Elles sont basées sur les paramètres propres à chacune des sphères et sur leurs positions relatives dans l'espace. Le travail de modélisation effectué en amont rend possible l'utilisation de règles heuristiques simples.

Le système final doit posséder un certain degré d'automatisation et ne pas nécessiter d'intervention extérieure. Nous avons souligné dans le Chapitre 3 lors de nos expérimentations que le choix du nombre de composantes K du modèle de mélange est crucial et conditionne la bonne modélisation des données. En pratique le nombre "optimal" de sphères dépend du nombre de personnes présentes dans la scène et n'est pas connu à priori. Par conséquent, le choix du paramètre K doit être intégré à l'algorithme complet de détection et être choisi automatiquement et de manière adaptative. Cette question a été illustrée lors des expérimentations du Chapitre 3 et une méthode même a été proposée. Toutefois, son coût calculatoire se révèle trop important pour une utilisation en quasi temps-réel. Après avoir examiné brièvement les techniques existantes de la littérature, nous décidons finalement de déterminer dynamiquement la valeur de ce paramètre par une stratégie descendante.

L'algorithme complet de détection doit fonctionner en quasi temps-réel. L'estimation du modèle de mélange nécessite une quantité de calculs importante (principalement concentrée à l'étape M de l'algorithme EM). La complexité de la méthode d'estimation est réduite par des améliorations algorithmiques d'une part et par une parallélisation via OpenMP d'autre part. Ces évolutions feront l'objet d'une brève étude bibliographique.

Les méthodes existantes de détection de têtes mentionnées dans cette section ne fournissent pas le même type de résultat que notre méthode, i.e. la modélisation des têtes par des sphères. Le type de résultats présenté dans [187] se rapproche du nôtre, mais la méthode proposée ne consiste pas en une estimation statistique réelle des paramètres des sphères. De plus, la méthode de segmentation du nuage de points 3D par un ensemble de sphères a été comparée à l'algorithme RANSAC au Chapitre 3. La méthode de détection de têtes proposée dans ces travaux sera donc comparée à un algorithme constitué d'une segmentation avec RANSAC puis de l'application des mêmes règles utilisées par notre méthode. L'influence de la modélisation des données sur la détection finale pourra ainsi être étudiée.

Ce chapitre est organisé de la manière suivante. Nous commencerons tout d'abord par présenter en section 4.2 le système considéré et la chaîne complète de traitements. Nous nous focaliserons sur les règles de détection de têtes retenues. Après un bref état de l'art, la méthode de choix automatique du nombre de composantes K du modèle de mélange employée sera détaillée en section 4.3. Les optimisations algorithmiques et d'implémentation apportées seront traitées en section 4.4 Nous analyserons et discuterons ensuite les résultats obtenus sur des séquences d'images de test en section 4.5. Enfin, nous conclurons le chapitre en section 4.6.

4.2 Méthode de détection de têtes dans un nuage de points 3D

Dans cette section, nous décrivons le système mis en place et l'algorithme complet de détection et de modélisation de têtes dans un nuage de points 3D. Dans un premier temps, le système sera brièvement décrit et les différentes étapes de la chaîne de traitements seront expliquées en section 4.2.1 avant de détailler dans un second temps les règles de détection de têtes en section 4.2.2.

4.2.1 Algorithme global

Avant de décrire l'algorithme complet de détection de têtes dans un nuage de points 3D, décrivons brièvement le système considéré schématisé Figure 4.2.1. Un capteur 3D est placé en vue verticale au centre de la zone d'intérêt à une hauteur d'environs 2,50 mètres et dirigé en direction du sol. Les personnes que nous souhaitons détecter et modéliser sont positionnées sous le capteur dans une pièce étroite et confinée. Les individus sont donc proches les uns des autres et observés en vue de dessus ou légèrement inclinée. Les personnes sont supposées se tenir debout dans la pièce et leurs têtes peuvent ne pas être entièrement visibles sur l'image. Ce dernier point est illustré Figure 4.2.1 et implique que les objets que nous cherchons à modéliser peuvent être tronqués et seulement partiellement visibles.

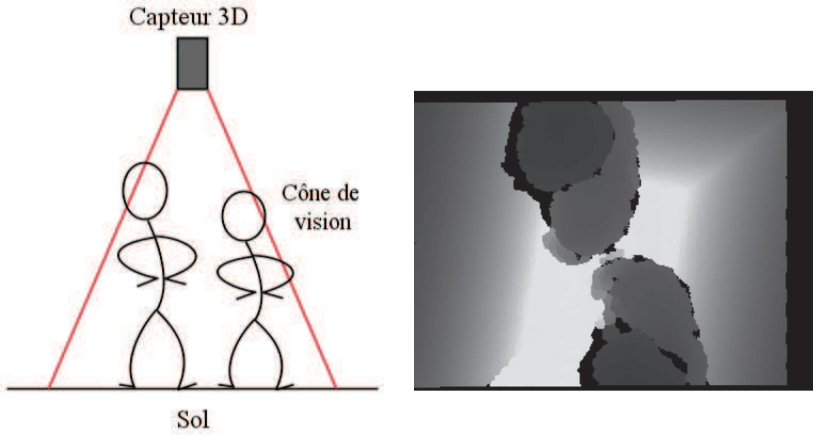


FIGURE 4.2.1 – Schéma représentant le système (a) et image de profondeur (b).

À partir ce cette installation, nous cherchons à détecter et modéliser les têtes des individus présents dans la pièce. L'algorithme complet de détection de têtes proposé est résumé par le schéma de la Figure 4.2.2 dans lequel chaque étape est illustrée par un exemple. Il est composé de trois grandes étapes : l'acquisition des données, un traitement général et un traitement spécifique à la détection de têtes auxquelles s'ajoute une calibration préalable du système.

Tout d'abord, le capteur 3D fournit une image couleur et une image de profondeur de dimensions 320×240 représentant la scène à une fréquence de 30 images par secondes. L'image couleur n'est pas considérée lors de nos traitements et nous n'utiliserons dans la suite que l'information de profondeur. Chaque pixel de la carte de profondeur contient la distance du capteur à l'objet représenté. On remarque la présence de pixels dits indéterminés représentés en noir et ne contenant aucune information de distance. Les principes de fonctionnement et les caractéristiques principales des capteurs 3D ont été présentés en détails Chapitre 1 section 1.2.

Une première segmentation de l'image est effectuée en ne conservant que les pixels représentant des objets situés dans une zone d'intérêt 3D prédéfinie. Les points situés hors de la zone de traitement sont écartés. Le procédé utilisé est décrit Chapitre 1 section 1.3.3.2. Une seconde segmentation plus complexe est ensuite appliquée et consiste en un algorithme de soustraction de fond dont le fonctionnement a été détaillé Chapitre 1 section 1.3.3.1. À l'issue de cette étape, seuls les pixels appartenant à la zone d'intérêt 3D et faisant partie de l'avant plan (objets nouveaux différents du fond de la scène) sont conservés. La seconde segmentation permet notamment de supprimer les pixels représentant les murs ou le sol. Dans l'exemple de la Figure 4.2.2, la personne a pu correctement être segmentée. Une phase de modélisation du fond de la scène durant laquelle la pièce est vide et ne contient aucun objet a été effectuée lors de la calibration du système. Rappelons que dans notre cas l'application de la soustraction de fond à l'image de profondeur apporte une plus grande robustesse qu'avec l'image couleur.

Le nuage de points 3D représentant les objets de la scène peut maintenant être calculé avec les relations données Chapitre 1 section 1.3.1. Ce calcul nécessite la connaissance des distances focales internes du capteur qui sont connues. Les points sont exprimés dans un repère 3D lié au capteur représenté Figure 4.2.3. Nous appliquons ensuite un algorithme de débruitage (Chapitre 1 section 1.3.2.1) suivi d'un changement de repère (Chapitre 1 section 1.3.6) pour finalement obtenir des points dont les coordonnées sont exprimées dans un repère lié au sol de la scène représenté Figure 4.2.3. Le calcul de la transformation rigide entre les deux systèmes de coordonnées est basé sur une modélisation du sol par un plan. L'estimation du sol est obtenue par un algorithme RANSAC (Chapitre 1 section 1.3.5) lors de la calibration du système. Notons dès à présent que ce changement de repère nous permettra par la suite de connaitre la hauteur des objets par rapport au sol. La position et l'orientation du capteur dans l'espace sont alors connus.

Le nuage de points 3D peut être stocké de manière organisée (tableau bidimensionnel agencé comme l'image 2D) ou non organisée (tableau unidimensionnel). L'algorithme appliqué lors de la prochaine étape n'utilisant pas la structure 2D de l'image, nous choisissons de travailler avec des données non organisées. Notons toutefois que d'autres traitements pourraient avoir besoin de cette structuration, par exemple pour la recherche rapide des k plus proches voisins d'un point donné. La série de traitements présentée jusqu'à maintenant reste générale et pourrait être commune à différents algorithmes travaillant sur un nuage de points 3D.

La méthode de détection de têtes que nous proposons est constituée de deux étapes. Les données sont tout d'abord modélisées par le modèle de mélange sphérique présenté au Chapitre 3. La méthode d'estimation (l'algorithme EM plus précisément) est initialisée par un algorithme K-means ([28]). En effet, le capteur étant positionné en hauteur et proche des têtes des individus, la densité de points sur les têtes est plus élevée que sur les parties du corps plus éloignées du capteur. Par conséquent, des classes seront toujours en pratique situées vers les têtes des personnes ce qui implique qu'elles pourront être correctement modélisées puis détectées.

La mise en œuvre pratique de l'algorithme d'estimation du modèle de mélange sphérique nécessite certaines adaptations pour satisfaire les contraintes d'utilisation requises. Les adaptations effectuées seront présentées dans les sections suivantes de ce chapitre. Une fois le nuage de points correctement modélisé par un ensemble de sphères, nous appliquons des règles heuristiques simples pour déterminer quelles composantes représentent des têtes. La section suivante est justement dédiée à la description des règles de détection retenues.

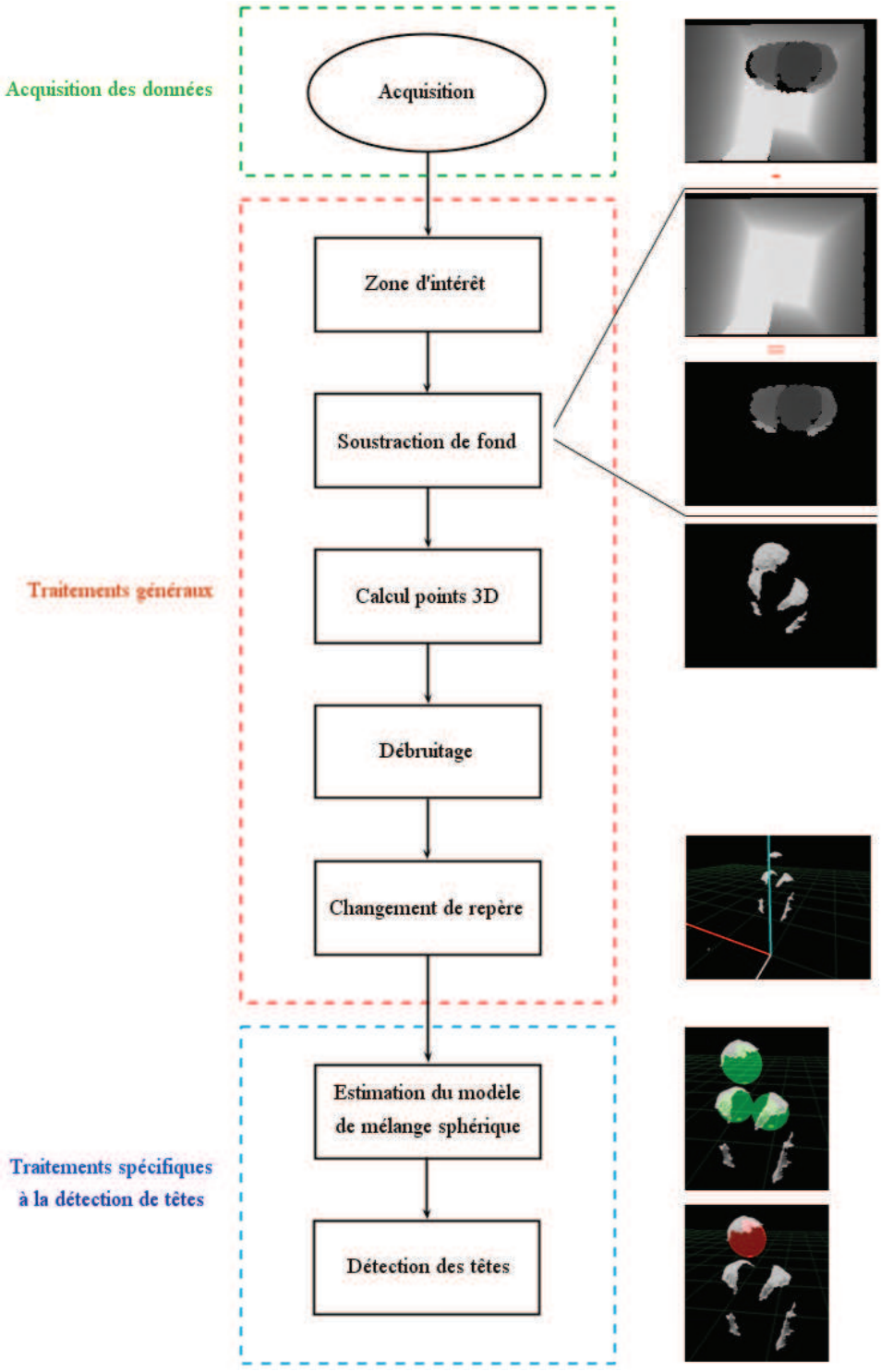


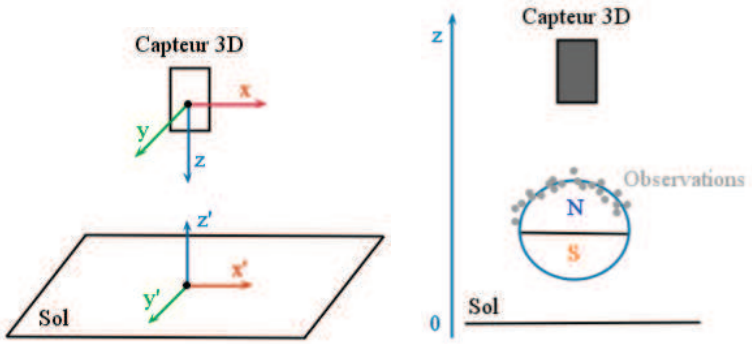
FIGURE 4.2.2 – Schéma général de la chaîne de traitements complète.

4.2.2 Règles de détection des têtes

Nous nous intéressons maintenant aux règles de décision permettant de déterminer quelles composantes du modèle de mélange représentent des têtes. Nous choisissons pour cela d'appliquer des règles heuristiques simples pour affecter à chaque composante une étiquette appartenant à l'ensemble {tête, non-tête}. Les composantes du modèle de mélange représentent des objets dont les dimensions sont des grandeurs physiques de l'espace. Nous pouvons donc nous appuyer directement sur les valeurs des paramètres du modèle estimé pour réaliser la détection.

Les règles de détection sont un ensemble de critères éliminatoires qu'une tête doit vérifier pour être détectée. Les critères sont basés d'une part sur les paramètres intrinsèques des sphères et d'autre part sur leurs positions relatives. Si l'un d'entre eux n'est pas respecté, l'étiquette *non tête* est affectée à la composante sans retour possible. Le processus de détection est résumé par le schéma de la Figure 4.2.4.

Le premier critère consiste à examiner le rayon de la sphère. En effet, le rayon de la composante doit appartenir à l'intervalle $[r_{min}, r_{max}]$ pour que cette dernière puisse être considérée comme étant une tête. La valeur de r_{max} doit être choisie suffisamment large pour prendre en compte les cas où la tête est vue de face. En pratique, malgré le fait que le capteur soit positionné verticalement, il peut arriver qu'une personne lève la tête et regarde le capteur de face. Dans ce cas, la composante modélisant la tête aura un rayon plus important que si la tête est vue de dessus puisque une partie des observations sera située sur le visage. Les valeurs de ces paramètres sont calibrées expérimentalement. En plus de cette première contrainte sur le rayon, la composante doit être suffisamment haute par rapport au sol de la scène. Comme les observations sont représentées dans un repère 3D lié au sol (voir Figure 4.2.3 (a)), ceci se traduit par le fait que la coordonnée z du centre de la sphère doit être supérieure à une limite z_{min} prédéfinie.



(a) (b)

FIGURE 4.2.3 – Les différents repères considérés (a) et la dispersion des points sur les hémisphères Nord et Sud d'une composante (b).

La deuxième série de critères prend en compte la classification des données fournie par l'algorithme EM. Les critères sont basés sur le positionnement des composantes par rapport aux points du nuage qui lui sont associés. Chaque point x_i est assigné à sa classe C_k la plus probable avec $k = \arg\max_{1 \leq l \leq K} \{t_{il}\}$ et où t_{il} représente la probabilité que le point x_i appartienne à la classe C_l. Tout d'abord, le nombre de points associés à une composante doit être suffisamment important et supérieur à n_{min}. Ce critère permet d'éliminer les sphères associées à trop peu de points, comme par

exemple une main. La valeur du paramètre n_{min} choisie doit également prendre en compte le fait que certaines têtes ne sont que partiellement visibles sur l'image. Ensuite, la distance moyenne du sous-ensemble de points à la sphère, notée E, doit être inférieure à ϵ_{erreur}. Ce critère assure que la sphère modélise correctement les données. Enfin, la répartition des observations par rapport aux hémisphères de la composante considérée est analysée. Chaque point est plus proche de l'un des deux hémisphères (nord pour celui vers le plafond et sud pour celui vers le sol) de la sphère. Si le nombre de points associés à l'hémisphère sud est plus élevé que le nombre de points associés à l'hémisphère nord, la composante ne peut être considérée comme étant une tête. Le capteur étant en position verticale, on s'attend à ce que la sphère soit située en dessous des points (selon le repère lié au sol) et non pas au dessus. Cette dernière règle permet d'écarter les sphères positionnées au dessus des observations et modélisant généralement des objets à surface plane. Pour plus de détails, voir Figure 4.2.3 (b). Les sphères vérifiant toutes les conditions précédentes sont considérées comme *têtes potentielles*.

Enfin, la dernière série de critères est basée sur les positions relatives des composantes. De part la configuration du système, deux composantes d'étiquettes *tête potentielle* ne peuvent toutes les deux représenter une tête si elles se recoupent dans le plan horizontal (Oxy) du sol. Autrement dit, les projetés des sphères sur le sol seront d'intersection vide. Lorsque ce cas de figure apparait, nous décidons de ne conserver que la composante présentant l'erreur E la plus faible. La seconde composante est alors affectée de l'étiquette *non tête*. Pour terminer, une composante d'étiquette *tête potentielle* est munie de l'étiquette *tête* si au moins une composante *non tête* représentant le corps est située juste en dessous. La sphère modélisant le corps (ou une partie) doit alors recouper horizontalement celle modélisant la tête et être située en dessous (par rapport au sol, donc on examine la coordonnée z des centres).

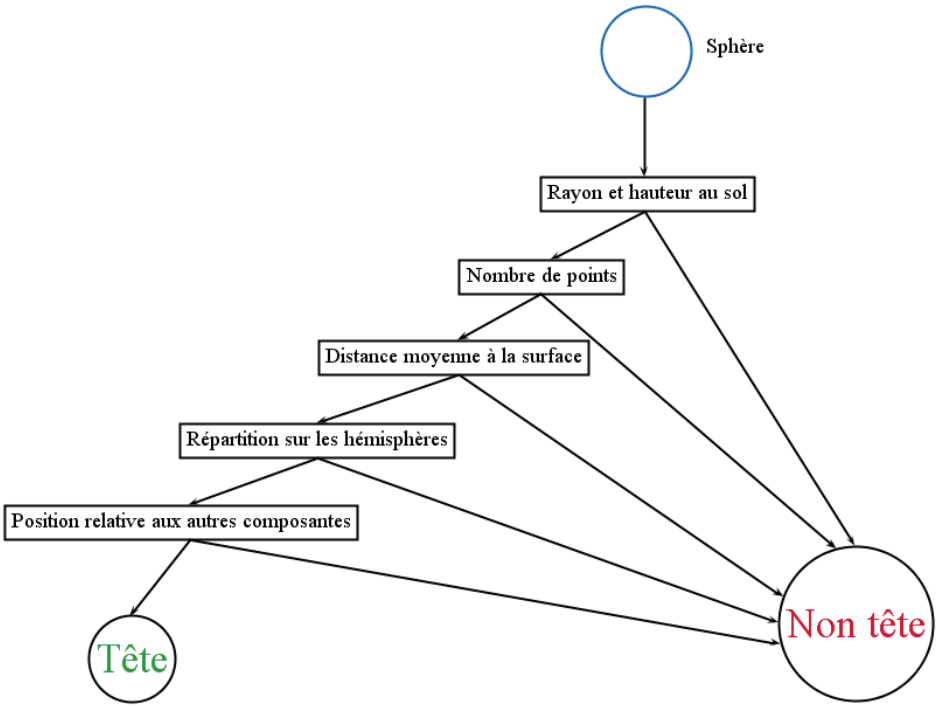


FIGURE 4.2.4 – Schéma de principe d'application des règles de détection.

Les règles de détection définies ci-dessus ont été déterminées d'une part grâce aux informations a priori (configuration du système) et d'autre part à l'aide d'expérimentations. L'utilisation de règles simples est possible et s'explique par le travail difficile de modélisation effectué à l'étape précédente de l'algorithme global. L'efficacité de la stratégie mise en œuvre sera quantifiée sur des séquences d'images et comparée à celle d'un autre algorithme basé sur une méthode différente de segmentation en section 4.5. Les valeurs des paramètres retenues seront également précisées lors des expérimentations.

4.3 Choix automatique du nombre de composantes

Dans cette section, nous traitons le problème du choix du nombre de composantes K du modèle de mélange. En pratique, la valeur de ce paramètre n'est pas connue à priori et doit être estimée en plus des paramètres des différentes classes. Dans notre cas, le nombre de classes dépend du nombre de personnes présentes dans la scène et de leur posture.

D'une part, le système de détection de têtes doit être le plus automatique possible. L'algorithme doit donc être capable de choisir automatiquement le bon nombre de composantes K. D'autre part, la valeur de ce paramètre conditionne le succès de l'algorithme entier. En effet, si aucune sphère ne modélise les points correspondant à une tête, cette dernière ne pourra être correctement détectée. A l'inverse, si une tête est sur-segmentée par plusieurs sphères, les données sont partagées en plusieurs classes et la détection échouera. L'objectif est donc de choisir convenablement et automatiquement le paramètre K de telle manière que les têtes soient chacune modélisées par une seule sphère. Nous établissons un état de l'art des méthodes existantes en section 4.3.1 et décrivons la stratégie finalement adoptée en section 4.3.2.

4.3.1 État de l'art

Le choix du nombre de composantes K d'un modèle de mélange est un problème central se posant pour tous les algorithmes de classification non-supervisée. L'augmentation du nombre de classes entraîne une augmentation mécanique de la vraisemblance et un coût calculatoire plus important. Si ce paramètre est choisi trop faible, le modèle ne sera pas assez complexe et sera mal adapté à la structure des données. Au contraire, si K est choisi trop élevé, les données seront sur-segmentées et des classes identiques ou non significatives dont l'interprétation sera difficile seront obtenues. Le nombre optimal de composantes K^* est défini comme étant le plus petit entier tel que toutes les composantes soient différentes et toutes les proportions du mélange non nulles, c'est à dire

$$K^* = \min \{K \text{ tel que } \theta_i \neq \theta_j \text{ et } \pi_i > 0, \forall 1 \leq i, j \leq K\} \tag{4.3.1}$$

où θ_k et π_k désignent respectivement le vecteur de paramètres et le poids de la composante k du mélange. Pour résoudre ce problème, différentes approches ont été proposées. Nous commençons par décrire les tests de ratio de vraisemblance en section 4.3.1.1. Nous présenterons ensuite les méthodes de sélection de modèles par critère d'information en section 4.3.1.2 avant de nous intéresser en section 4.3.1.3 à celles déterminant K pendant le processus d'optimisation du modèle. Les techniques présentées dans cette section sont abordées dans le cadre de notre problème de choix de K. Elles peuvent également être utilisées pour comparer des modèles de nature différentes et ne se réduisent pas au problème du choix du nombre de classes.

Des méthodes de Monte-Carlo par chaines de Markov ont été mises au point pour échantillonner la densité de probabilité a posteriori (voir [137, 160, 157]). Des schémas de rééchantillonage ([129]) et de validation croisée ([133]) ont aussi été utilisés. Malheureusement, le coût calculatoire de ces approches rend leur utilisation incompatibles avec notre objectif de traitement en quasi temps-réel. Nous ne traiterons donc pas en détails ces méthodes.

4.3.1.1 Test du ratio de vraisemblance

Le test du ratio de vraisemblance est un test d'hypothèse statistique permettant la comparaison de deux modèles M_0 et M_1 ajustant un même ensemble de données. Les deux modèles sont liés par le fait que le modèle M_0 est un cas particulier du modèle M_1 et que ses paramètres comportent q degrés de liberté de moins. Le problème consiste à choisir le modèle expliquant au mieux les données. Ceci conduit au test d'hypothèse suivant confrontant les hypothèses H_0 et H_1

$$\begin{cases} H_0 : M_0 \\ H_1 : M_1 \end{cases} . \tag{4.3.2}$$

Les paramètres inconnus sont estimés et les valeurs de la vraisemblance maximisée calculées. La statistique de test Λ utilisée est le ratio de vraisemblance (LRT, Likelihood Ratio Test). Elle se base sur le logarithme du ratio des vraisemblances obtenues après ajustement des deux modèles et peut s'écrire

$$\Lambda = 2\left[L_1 - L_0\right] \tag{4.3.3}$$

où L_0 et L_1 sont les valeurs de la vraisemblance maximisée de chacun des modèles. Le modèle M_1 comportant plus de paramètres aura une vraisemblance au moins aussi élevée que le modèle M_0. Sous certaines hypothèses de régularité, la statistique Λ converge vers une distribution du χ^2 à q degrés de libertés lorsque le nombre d'observations $n \to +\infty$. Une p-value est calculée et permet alors d'accepter ou de rejeter H_0, i.e. choisir M_0 ou M_1 en fonction du risque $\alpha \in [0, 1]$ choisi.

Ce test a été appliqué en particulier aux modèles de mélange dans [110] pour choisir entre deux modèles comportant un nombre de composantes différent. En pratique, des tests successifs sont effectués pour valider ou non le passage à un autre nombre de classes. La méthode d'estimation du nombre de classes K se résume alors à une série de tests d'hypothèses. Chaque test confronte l'hypothèse nulle H_0 à K_0 classes à l'hypothèse alternative H_1 à K_1 classes avec $K_0 < K_1$. Les deux modèles sont comparés avec le test suivant

$$\begin{cases} H_0 : K = K_0 \\ H_1 : K = K_1 \end{cases} . \tag{4.3.4}$$

Généralement, le test est effectué à chaque incrémentation du nombre de composantes et $K_1 = K_0 + 1$. La statistique de test devient dans ce cas

$$\Lambda = 2\left[\log\left(P\left(X \mid \Theta^{(K_1)}\right)\right) - \log\left(P\left(X \mid \Theta^{(K_0)}\right)\right)\right]. \tag{4.3.5}$$

Dans le cas des modèles de mélange, la statistique Λ ne peut alors pas être approchée par une distribution du χ^2. Une alternative possible consiste à effectuer la procédure de test à l'aide d'une procédure de Monte-Carlo ou bootstrap (voir [100, 5, 129, 180]). La distribution limite de la statistique Λ étant inconnue, la p-value ne peut être déterminée. Elle est alors estimée soit par une approche Monte-Carlo, soit par bootstrap. La précision de l'estimation de la p-value est liée au nombre d'échantillons simulés. Même si en pratique une valeur approximative est suffisante, le coût calculatoire est élevé, en particulier lorsque K_0 est grand. L'étape d'échantillonnage nécessaire à l'application de cette technique est très coûteuse en temps de calcul. Pour cette raison, nous allons nous intéresser dans la section suivante aux approches de type sélection de modèles.

4.3.1.2 Sélection de modèles et critères d'information

Les approches de type sélection de modèles consistent à appliquer l'algorithme d'estimation pour différentes valeurs de K appartenant à l'intervalle $[K_{min}, K_{max}]$. Chaque modèle est ensuite évalué selon un critère d'information. Le modèle retenu est celui minimisant la valeur du critère choisi. Plus formellement, le nombre de composantes optimal au sens d'un critère donné est

$$K^* = \arg\min_K \left\{ C\left(\widehat{\Theta}^{(K)}, K\right)\right\} \tag{4.3.6}$$

où C est un critère d'évaluation dont le calcul dépend du vecteur de paramètres optimisés $\widehat{\Theta}^{(K)}$ et du nombre de classes K. Des états de l'art sur les différentes catégories de critères existants dans la littérature ont été réalisés (par exemple) dans [128] (Chapitre 6) et [144].

Une première catégorie de critères est basée sur une **pénalisation de la log-vraisemblance** de la forme

$$C\left(\widehat{\Theta}^{(K)}, K\right) = -2\log\left(p\left(X \mid \Theta^{(K)}\right)\right) + \mathcal{P}(K) \tag{4.3.7}$$

où le premier terme est la log-vraisemblance du modèle à K composantes et $\mathcal{P}(K)$ un terme pénalisant les modèles comportant beaucoup de paramètres. La valeur de la log-vraisemblance augmente mécaniquement avec le nombre de composantes. Ceci justifie l'ajout du terme de pénalité $\mathcal{P}(K)$ pénalisant les modèles contenant plus de paramètres et donc étant plus complexes. Le critère essaie donc de trouver un bon compromis entre la qualité d'ajustement des données et la complexité du modèle. Ces critères permettent de trouver le modèle (parmi ceux testés) qui minimise l'information de Kullback-Leibler ([128]). Parmi les critères couramment utilisés, on trouve notamment

- Akaike's Information Criterion (AIC) ([6, 202])
- Bayesian Information Criterion (BIC) ([173, 40, 64, 85])
- Laplace Empirical Critetion (LEC) ([128])
- Minimum Description Length (MDL) ([161])
- Minimum Message Length criterion (MML) ([199, 142, 198])
- Informational Complexity criterion (ICOMP) ([34])

Plus récemment, l'algorithme EMCE ([2]) et le critère PHDC ([120]) ont été proposés. Nous ne donnons ici pour illustration que l'expression du critère AIC qui se présente sous la forme

$$C\left(\widehat{\Theta}^{(K)}, K\right) = -2\log\left(p\left(X \mid \Theta^{(K)}\right)\right) + 2N(K) \tag{4.3.8}$$

où $N(K)$ désigne le nombre de paramètres dans le modèle. De nombreux auteurs ont observé que ce critère a tendance à surestimer le nombre de composantes (voir par exemple [114, 49]).

Les **critères basés sur le pouvoir classifiant** constituent une deuxième catégorie de critères d'information. Ils mesurent la capacité d'un modèle à fournir des classes bien séparées. Les modèles sont évalués sur leur pouvoir discriminant et classifiant. On trouve dans cette catégorie les critères listés ci-dessous

- Critère LP et critère d'entropie (E) ([24])
- Classification Likelihood Criterion (CLC) ([26])
- Classification Log-likelohod (CL) ([26])
- Normalized Entropy Criterion (NEC) ([49])
- Integrated Classification Likelihood (ICL) ([25])

– Critère EP ([156])
– Partition Coefficient (PC) ([202]) et Partition Entropy (PE) ([23])

Par exemple, les critères LP et LE utilisés conjointement sont de la forme

$$LP = -\sum_{s=1}^{S}\sum_{n=1}^{N} z_{ns} \log(p_{ns}) \tag{4.3.9}$$

$$LE = -\sum_{n=1}^{N}\sum_{s=1}^{S} p_{ns} \log(p_{ns}) \tag{4.3.10}$$

où $P = [p_{ns}]_{n\in[1..N],s\in[1..S]}$ est la matrice des probabilités a posteriori (appartenance des observations aux classes) et z_{ns} la variable valant 1 si le point d'indice n appartient à la classe s et 0 sinon. Dans le cas où les composantes sont bien séparées, les valeurs des critères sont faibles.

Enfin, la dernière catégorie que nous citons contient les critères de **ratio d'information minimal** dont quelques uns sont listés ci-dessous

– Minimum Information Ratio (MIR) ([202]) et Minimum Information Ratio Estimation and Validation (MIREV) ([202])
– ALL et ANC ([61])
– Elbow Likelihood (EL) ([202])
– LOGVL ([8, 9])
– Cross Validation Based Information Criterion (CVIC) ([179])

A titre d'exemple, le critère MIR est défini comme étant la plus petite valeur propre du "ratio des matrice d'information" $F_c^{-1}F$ où F est la matrice d'information de Fisher et F_c la matrice d'information de Fisher des observations classifiées. Le critère MIR peut être estimé à partir du taux de convergence de l'algorithe EM. Le critère mesure la capacité des données à distinguer les différentes composantes. Sa valeur est d'autant plus élevée que la classification est réussie.

Quelque soit le type de critère utilisé, ces méthodes ramènent le choix de K à celui de l'intervalle $[K_{min}, K_{max}]$, ce qui serait acceptable dans notre cadre d'application. L'inconvénient majeur de ces techniques est, pour nos contraintes de fonctionnement, que l'algorithme d'estimation des paramètres doit être appliqué pour chaque valeur testée de K. Le coût calculatoire associé est, de part notre algorithme d'estimation et la quantité de données à traiter, important et difficilement compatible avec la contrainte de quasi temps-réel. Nous nous intéressons donc dans la section suivante aux techniques de choix dynamique de K.

4.3.1.3 Choix dynamique

Les techniques présentées dans les sections précédentes sont basées sur la comparaison de modèles comportant un nombre différent de composantes. Plusieurs modèles sont ajustés, évalués puis comparés. L'algorithme d'estimation des paramètres est dans ce cas réinitialisé pour chaque nouveau modèle. Dans cette section, nous nous intéressons à une autre stratégie consistant à déterminer la meilleure valeur du paramètre K pendant le processus d'optimisation. Le vecteur de paramètres Θ du modèle de mélange tout entier et le nombre de composantes K sont estimés simultanément. Un modèle à K_0 classes est tout d'abord estimé, puis des composantes sont par la suite ajoutées, supprimées ou fusionnées selon certains critères au fur et à mesure de l'algorithme. On parle d'approche de type

décomposition / fusion (split and merge). Ces techniques sont généralement plus efficaces que celles basées sur des recherches aléatoires, exhaustives ou génétiques ([149]). Cette manière de procéder nous amène à nous poser les questions suivantes :

– quels critères permettent de déclencher une fusion, une séparation, un ajout ou une suppression ?
– comment fusionner deux classes ?
– quels paramètres donner à une classe venant d'être ajoutée ?

Les travaux mentionnés dans cette section portent uniquement sur le mélange gaussien. On distingue globalement trois catégories d'approches qui se différencient par la manière dont le nombre de classes évolue. Il peut être amené à soit augmenter, soit diminuer soit les deux. Quelque soit la technique adoptée, les paramètres sont réestimés entre chaque série de modifications. La plupart du temps, il ne s'agit que d'une optimisation partielle (*Partial EM*) sur un sous-ensemble des composantes. La méthode d'estimation (EM dans notre cas) est initialisée avec le dernier résultat obtenu altéré par la modification apportée au nombre de classes. L'estimation du modèle de mélange n'est donc pas reprise de zéro et l'initialisation par K-means n'est effectuée que pour $K = K_0$. Le principe général est résumé sur le schéma de la Figure 4.3.1.

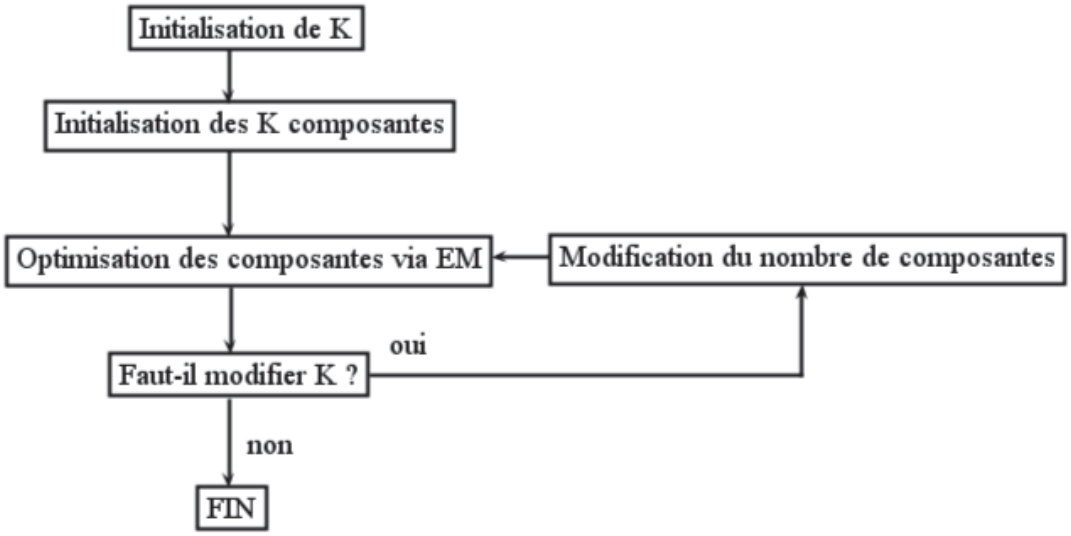


FIGURE 4.3.1 – Schéma résumant la sélection dynamique du paramètre K. Le nombre de composantes est modifié tant que le critère l'indique. Les paramètres du modèle sont réestimés à partir du résultat de l'estimation précédente (mêmes classes à la dernière modification apportée près).

La première catégorie de méthode est la plus générale puisqu'elle peut faire varier K par augmentation ou diminution. Le nombre de composantes du modèle est initialement fixé à K_0. Des critères adaptés doivent être choisis pour déterminer le meilleur choix à un moment donné.

L'algorithme heuristique SMEM (Split and Merge EM) ([194]) a été mis en place pour améliorer les performances de l'algorithme EM. D'après les auteurs, le problème de fusion de deux composantes est un problème bien posé alors que celui de la séparation est mal posé. En effet, deux classes peuvent être séparées de plusieurs manières alors que la fusion est simplement la composante résultant de l'union des classes correspondantes. Dans ces travaux, des modèles candidats sont générés par

séparations et fusions. Le critère de fusion retenu utilise les vecteurs de probabilités d'appartenance aux différente classes. Si un nombre important de points ont des probabilités similaires vis à vis de deux composantes, celles ci peuvent être fusionnées. Le critère de test de fusion des classes i et j vaut

$$\frac{P_i\left(\widehat{\Theta}\right)^T P_j\left(\widehat{\Theta}\right)}{\left\|P_i\left(\widehat{\Theta}\right)\right\|\left\|P_j\left(\widehat{\Theta}\right)\right\|} \tag{4.3.11}$$

où $\widehat{\Theta}$ est le vecteur de paramètres optimisés et $P_i\left(\widehat{\Theta}\right)$ le vecteur contenant les probabilités d'appartenance des points aux composantes. La fusion est effectuée lorsque le critère est suffisamment élevé. Le critère de séparation de la composante k est basé sur la divergence de Kullback-Leibler locale

$$\int f_k\left(x,\Theta^*\right)\log\left(\frac{f_k\left(x,\Theta^*\right)}{p_k\left(x,\theta_k^*\right)}\right) \tag{4.3.12}$$

où f_k est la densité de la composante k et p_k une estimation non paramétrique de f_k. Ce critère quantifie la distance entre la distribution des points associés à la composante k et la densité de cette même composante dans le mélange. Une séparation peut être déclenchée lorsque ce critère est trop élevé. Après chaque modification du nombre de composantes, les nouveaux paramètres doivent être réestimés. Il a été proposé dans [212] deux méthodes de séparation basées sur une décomposition en valeurs singulières et une factorisation de Cholesky.

La méthode SSMEM (Stepwise Split and Merge EM) ([200]) commence par estimer un modèle à K_0 composantes. Les classes sont ensuite séparées et fusionnées tant que cela est nécessaire. Une optimisation partielle des composantes altérées est effectuée après chaque série de modifications. Le critère de fusion de deux classes i et j est le coefficient de corrélation entre les vecteurs de probabilité d'appartenance $P_i(\widehat{\Theta})$ et $P_j(\widehat{\Theta})$ des observations à ces classes. Un coefficient de corrélation fort indique que les classes doivent être fusionnées. La distance de Kullback-Leibler similaire à (4.3.12) permet de déterminer si une séparation est nécessaire. La densité locale de chaque composante est estimée par une méthode d'estimation non paramétrique et comparée à la densité de cette même composante dans le modèle de mélange.

L'approche SAGEM (Self-Adapting Gaussian Expectation Maximization) ([92]) initialise le modèle avec un nombre de classes faible. Les composantes dont la log-vraisemblance n'augmente plus sont dédoublées ou fusionnées. Les critères de choix sont basés sur le volume occupé par la composante (déterminant de la matrice de covariance) et sur le nombre d'itérations depuis lequel aucune amélioration de sa log-vraisemblance n'a été constatée. Des seuils adaptatifs sont mis à jour à chaque itération pour décider de la modification du paramètre K.

La deuxième catégorie de méthodes débute avec un nombre de composantes faible et des classes sont ensuite ajoutées au fur et à mesure tant qu'elles améliorent la modélisation. Dans le cas où le nombre initial de classes est fixé à 1, le problème d'initialisation de l'algorithme ne se pose plus. L'initialisation des paramètres est alors immédiate ce qui rend la méthode moins sensible aux choix initial des classes. Un critère de séparation ou d'ajout bien choisi détermine à quel moment une incrémentation de K est nécessaire. Le choix de ce critère et des paramètres de la composante introduite sont de première importance.

Dans [197], une version incrémentale de EM est présentée. Il s'agit d'une approche de type algorithme glouton effectuant des choix locaux optimaux. Des composantes sont ajoutées une à une après

chaque itération de EM. Les paramètres de la nouvelle composante sont calculés par une recherche globale. Son centre μ est l'observation donnant la meilleure log-vraisemblance pour des matrices de covariance $\sigma^2 I_d$ prédéfinies. La fonction de vraisemblance est approchée par un développement de Taylor d'ordre 2. Une optimisation locale de la composante nouvellement introduite est effectuée par un algorithme EM partiel n'optimisant que les nouveaux paramètres. Une stratégie gloutonne similaire a été adoptée dans [195] où la nouvelle composante est cette fois ci sélectionnée parmi un ensemble de candidats générés dynamiquement. Chaque modèle successif correspond à une classification des données. m candidats sont alors générés pour chaque classe (par paires de deux via un algorithme k-means). Le candidat finalement sélectionné est celui maximisant la vraisemblance et son volume est inférieur à ceux des autres composantes.

La méthode FASTGMM (Fast Gaussian Mixture Modeling) ([93]) propose d'initialiser le paramètre K à 1 et de l'incrémenter au fil des itérations. Comme dans [92], l'évolution des vraisemblances est analysée et les volumes des composantes calculés. Des opérations de séparation sont déclenchées à l'aide de seuils adaptatifs.

Enfin, la troisième et dernière catégorie de méthodes est basée sur une diminution successive du nombre de classes, en choisissant K_0 grand. L'ajustement par un modèle comportant trop de composantes implique une sur-segmentation des données. Les classes redondantes ou trop peu significatives (peu de points dans la classe) sont ensuite supprimées. Cette manière de procéder assure que toutes les zones de l'espace d'observation peuvent potentiellement être occupées par une classe puisque le nombre initial de classes est grand. L'approche présentée dans [83] est basée sur le critère MML intégré à l'algorithme EM. Bien que cette méthode soit parmi les plus efficaces, son coût calculatoire reste néanmoins élevé. Dans [102], la fonction de vraisemblance est modifiée de manière à pénaliser le logarithme des proportions π_i du mélange. Il en découle de nouveaux estimateurs et les composantes ayant un poids trop proche de zéro sont supprimées au fur et à mesure de l'algorithme.

La manière dont des composantes gaussiennes peuvent être fusionnées ou séparées n'a pas été ici clairement précisée (choix des paramètres d'une nouvelle composante). Des techniques de fusion de composantes gaussiennes sont analysées dans [98]. Une mesure du recouvrement de deux composantes gaussiennes est proposée dans [184, 185]. Notons que dans nos travaux, les composantes du mélange ne sont pas gaussiennes et nous devrons trouver des procédures satisfaisantes pour notre modèle.

Nous concluons cet état de l'art en soulignant le fait que beaucoup d'approches intéressantes ont été proposées pour choisir le nombre de composantes d'un modèle de mélange. Toutefois, nous écartons les approches basées sur des tests d'hypothèses et sur des sélections de modèles à cause de leurs coût calculatoire trop élevé incompatible avec nos contraintes de quasi temps-réel. Nous décidons donc de nous inspirer des méthodes de choix dynamique de K. La distribution que nous souhaitons utiliser n'étant pas gaussienne, des critères de fusion / séparation adaptés devront être choisis.

4.3.2 Choix automatique du nombre de composantes du modèle de mélange sphérique

Pour choisir automatiquement le nombre de composantes de notre modèle de mélange sphérique, nous optons pour une méthode de choix *dynamique pas à pas descendante* décrite dans l'Algorithme 4.3.1. Cette approche est privilégiée aux méthodes de sélection de modèles pour leur efficacité calculatoire. En effet, comme cela a été mentionné dans l'étude bibliographique, la sélection de modèles nécessite d'estimer le modèle de mélange plusieurs fois pour différentes valeurs de K et ce de manière indépendante. L'algorithme d'estimation complet est appliqué pour chaque nouvelle valeur de K. En revanche, un choix dynamique offre la possibilité de réutiliser les dernières estimations des paramètres entre chaque modification de K. Cette approche offre également la possibilité de débuter l'algorithme avec un nombre de classes K_0 choisi assez grand pour que les composantes soient dispersées dans toutes les zones du nuage de points 3D. Nous espérons ainsi limiter les cas où une tête ne serait pas modélisée par une composante du modèle de mélange car cela impliquerait forcément une non détection.

L'algorithme est tout d'abord initialisé avec un nombre de composantes K_0 choisi suffisamment grand. Après chaque phase d'optimisation de l'algorithme EM, les composantes devant être supprimées ou fusionnées sont recherchées. Une seule suppression ou fusion sont autorisées à chaque phase de modification de K. Après chaque modification, des itérations de l'algorithme EM sont de nouveau effectuées pour réestimer le modèle.

Algorithm 4.3.1 Choix du nombre de composantes K

Require: Nuage de points $\mathcal{P}$ et nombre initial de classes K_0.

1: $K = K_0$
2: Estimer le modèle de mélange à K composantes sur $\mathcal{P}$ avec initialisation par K-means
3: **repeat**
4: Supprimer une composante si besoin et modifier K
5: Fusionner deux composantes si besoin et modifier K
6: Ré-estimer le modèle de mélange à K composantes sur $\mathcal{P}$ en démarrant avec les dernier paramètres estimés
7: **until** aucune suppression ou fusion ne soit effectuée

La méthode peut être accélérée en supprimant ou fusionnant plusieurs composantes à la fois, mais cela augmente le risque de trop diminuer le nombre de classes. La recherche d'une composante l_{suppr} à supprimer commence par la recherche de la composante dont le poids $\pi_{l_{suppr}}$ est minimal, c'est à dire

$$l_{suppr} = \arg \min_{1 \leq k \leq K} (\pi_k) \tag{4.3.13}$$

et est déclenchée lorsque la condition suivante est vérifiée

$$\pi_{l_{suppr}} < \frac{1}{2\,K_0}. \tag{4.3.14}$$

La composante de poids minimal est supprimée si son poids est suffisamment faible. Le seuil $(2\,K_0)^{-1}$ en dessous duquel un poids est considéré comme négligeable a été déterminé expérimentalement et fonctionne bien en pratique dans la configuration de notre système.

La fusion de deux composantes est nécessaire lorsqu'elles modélisent des formes proches. Les résultats observés sur les jeux de données nous ont conduit à nous baser sur une mesure de proximité

entre les sphères. La proximité entre deux composantes k et l, notée s_{kl}, est simplement définie par

$$s_{kl} = \|\mu_k - \mu_l\| \tag{4.3.15}$$

où μ_k, μ_l sont les centres et r_k, r_l les rayons des sphères. La fusion est déclenchée lorsque la distance minimale entre tous les couples de sphères vérifie

$$s_{kl} < \min\left(\epsilon_{fusion}, r_k + r_l\right) \tag{4.3.16}$$

où ϵ_{fusion} est un seuil à fixer. Les sphères dont les centres sont proches ont tendance à modéliser la même portion du nuage de points. En pratique et de part la nature de nos données 3D, les composantes du modèle de mélange ne sont pas incluses les unes dans les autres.

Lorsque deux composantes k et l sont fusionnées, les paramètres de la nouvelle composante (indicée par new) sont ceux de la classe formée par l'union des observations :

$$\pi_{new} = \pi_k + \pi_l \tag{4.3.17}$$

$$\mu_{new} = \frac{\pi_k\,\mu_k + \pi_l\,\mu_l}{\pi_{new}} \tag{4.3.18}$$

$$r_{new} = \frac{\pi_k\,r_k + \pi_l\,r_l}{\pi_{new}} \tag{4.3.19}$$

$$\sigma^2_{new} = \frac{\pi_k\,\sigma^2_k + \pi_l\,\sigma^2_l}{\pi_{new}}. \tag{4.3.20}$$

Le processus est stoppé lorsque aucune nouvelle modification de K n'est nécessaire. Le nombre de composantes optimal K_{opt} est alors la dernière valeur de K.

Le procédé décrit dans cette section sera mis en œuvre sur des séquences d'images en section 4.5. La modélisation du nuage de points obtenue sera comparée à celle obtenue par un algorithme de type RANSAC.

4.4 Accélération de la méthode d'estimation

Dans cette section, nous présentons les optimisations algorithmiques et d'implémentation apportées à la méthode d'estimation du modèle de mélange sphérique. La procédure d'estimation des paramètres inconnus via l'algorithme EM décrite au Chapitre 3 est modifiée afin de diminuer son coût calculatoire et de s'approcher du quasi temps-réel. Les accélérations algorithmiques sont traitées en section 4.4.1 et la parallélisation des calculs en section 4.4.2. Une initialisation plus efficace de l'étape M est proposée en section 4.4.3. Les gains obtenus par chaque amélioration seront quantifiés en section 4.5.

4.4.1 Accélération de l'algorithme EM

4.4.1.1 État de l'art des méthodes d'accélération existantes

Les optimisations algorithmiques décrites dans cette partie concernent l'algorithme EM. L'algorithme a été beaucoup étudié et des améliorations visant à réduire son coût calculatoire ont été apportées. L'algorithme EM consiste en deux étapes itérées jusqu'à convergence. Les probabilités d'appartenance des points aux différentes classes sont estimées à l'étape E. Sa complexité est linéaire par rapport au nombre de points à traiter. Dans les applications comportant un grand nombre d'observations ou dont le calcul des probabilités est coûteux, il peut être intéressant de chercher à accélérer cette étape. L'étape M est une étape de ré-estimation des paramètres à partir des probabilités d'appartenance calculées à l'étape E. Son optimisation a un impact important sur le coût total lorsque la méthode d'estimation est complexe.

Des travaux ont été menés pour combiner l'algorithme EM avec des méthodes d'optimisation classiques pour en accélérer la convergence. Les méthodes Parameterized EM ([150, 159, 131, 19, 206, 146]) intègrent EM dans une méthode de descente de gradient. L'algorithme d'estimation EM est alors traité comme une application M qui a un vecteur de paramètres θ^j associe son amélioration via EM θ_{EM}^{j+1} et pouvant s'écrire $\theta_{EM}^{j+1} = M(\theta^j)$. Après chaque itération de EM, le vecteur de paramètres est mis à jour suivant le schéma

$$\theta^{j+1} = \theta^{(j)} + p_j \left(\theta_{EM}^{(j+1)} - \theta^{(j)} \right) \tag{4.4.1}$$

où $\theta_{EM}^{(j+1)}$ est la mise à jour du vecteur de paramètres $\theta^{(j)}$ par l'algorithme EM. La direction de descente retenue est celle indiquée par EM. Selon le choix du pas p_j, la convergence peut être accélérée. Le pas de descente peut être fixe ou adaptatif. Dans le cas où $p_j = 1$, on retrouve l'algorithme EM standard.

Un procédé d'accélération d'Aitken a été proposé dans [123] pour améliorer la convergence de EM. Cette technique est applicable pour un nombre modéré de paramètres car elle nécessite des inversions matricielles. L'algorithme EM a été combiné avec une méthode de gradient conjugué dans [107] et de quasi-Newton dans [189]. La différence du vecteur de paramètres avant et après avoir appliqué EM est encore une fois exploitée pour accélérer la convergence.

Dans la suite de cette section, quelques méthodes d'accélération de l'étape E sont d'abord traitées avant de passer à celles se concentrant sur l'étape M.

Accélération de l'étape E. On distingue deux stratégies d'accélération de l'étape E. La première est basée sur l'oubli partiel d'un sous-ensemble des données et la seconde sur la compression des données traitées.

Une idée naturelle pour réduire la quantité de calcul de l'étape E est de ne pas recalculer l'intégralité des probabilités. En effet, les probabilités d'appartenance de certaines observations ne varient pas (ou très peu) d'une itération à l'autre et leur calcul n'est pas nécessaire. La méthode SpEM (Sparse EM) ([138]) cherche à réduire le coût calculatoire de l'étape E en sélectionnant les calculs à effectuer. Les probabilités d'appartenance trop faibles à une itération donnée ne sont pas recalculées durant un nombre prédéfini d'itérations. Cette technique suppose que si un objet a une probabilité d'appartenance à une classe faible, elle n'augmentera pas significativement en une seule itération. Les probabilités correspondantes sont donc fixées et non mises à jour pendant plusieurs itérations. Une étape E standard (complète) est effectuée régulièrement pour recalculer toutes les probabilités et déterminer celles qui sont trop faibles. Le schéma de la Figure 4.4.1 décrit le fonctionnement global de la méthode.

Dans le même esprit, la méthode LEM (Lazy EM) ([190]) cherche à réduire la quantité de calculs de l'étape E en se concentrant sur les objets importants. Un sous-ensemble de points est considéré important si ses probabilités d'appartenance évoluent suffisamment. Un objet dont les probabilités évoluent peu nécessite moins d'attention. L'algorithme a la même structure que SpEM à la règle définissant si une mise à jour est nécessaire près. La convergence théorique de la méthode a été démontrée. Le schéma de la Figure 4.4.1 décrit le fonctionnement global de la méthode.

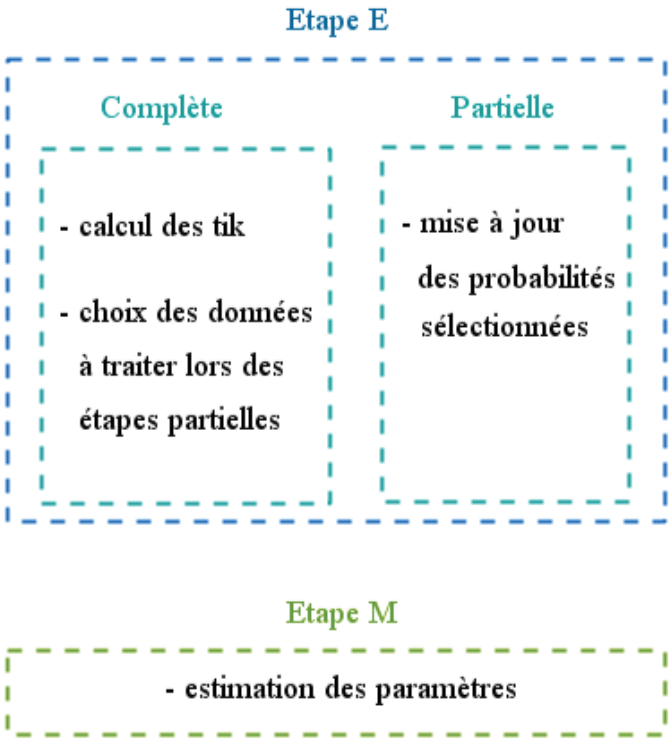


FIGURE 4.4.1 – Principe des variantes SpEM et LEM. Une étape E complète est effectuée une fois toutes les n_c itérations tandis qu'une étape partielle moins coûteuse est appliquée le reste du temps.

L'approche IEM (Incremental EM) ([138]) divise l'ensemble des observations en B blocs disjoints notés $B_1, \ldots, B_B$. Le nombre de blocs B est un paramètre de la méthode. Les blocs sont ensuite parcourus cycliquement et les probabilités correspondantes mises à jour. Une étape E est effectuée sur un seul bloc avant qu'une étape M ne soit appliquée. Une itération de IEM se résume donc à B étapes E et B étapes M intercalées. Tous les points sont visités à l'issue d'une itération. L'augmentation de la valeur du paramètre B augmente le temps de calcul. Le choix $B = 1$ revient à effectuer l'algorithme EM classique. Le principe de la méthode est illustré Figure 4.4.2. Dans [139], les auteurs se sont intéressés à la manière de choisir le nombre de blocs. La combinaison des méthodes IEM et SpEM, appellée SPIEM ([139]) a été expérimentée.

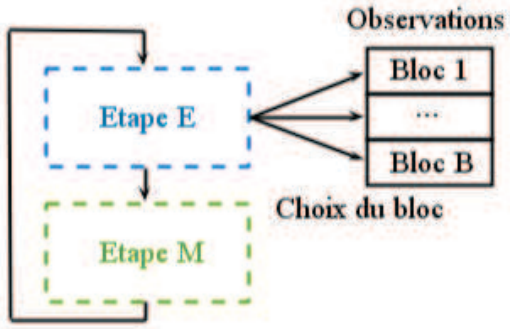


FIGURE 4.4.2 – Découpage des données en B blocs.

Une autre idée pour réduire la complexité de l'étape E consiste à compresser les données manipulées. Les observations sont compressées de manière à ce qu'un élément appelé observation compressée représente un sous-ensemble des observations. Les n observations sont alors représentées par n' observations compressées où $n' \leq n$. La quantité de données à traiter est alors plus faible et la quantité de calculs est par conséquent moins importante. En contrepartie, des observations différentes seront affectées de la même probabilité d'appartenance.

La méthode proposée dans [134] est basée sur la construction d'un arbre kd. Un arbre kd est une structure de données arborescente dans laquelle les données sont stockées efficacement. Il est construit par des divisions récursives de l'espace de travail. Dans cette structure de donnée, un ensemble de points appartenant à une même cellule de l'arbre est résumé par son sommet. La racine est le sommet contenant toutes les observations. Le principe de partitionnement des données par un arbre kd est illustré Figure 4.4.3. Les points d'un même sommet sont résumés (compressés) par leur centroïde dans [140]. Les probabilités d'appartenance de tous les points associés à un même sommet sont identiques et calculées à partir du centroïde. Ces approches mènent à une réduction importante des calculs. De manière générale, les structures de partitionnement de l'espace sont à utiliser lorsque la dimension d des points reste faible. Enfin, la méthode SPIEM mentionnée précédemment a été combinée à une compression par des arbres multi-résolutions ([140]) pour segmenter des images IRM. Une autre méthode ([35]) basée sur une compression des données a été développée et consiste à regrouper après avoir lancé l'algorithme EM les observations proches selon la distance de Mahalanobis. Dans ce cas, la convergence vers un maximum local de la fonction de vraisemblance n'est plus garantie.

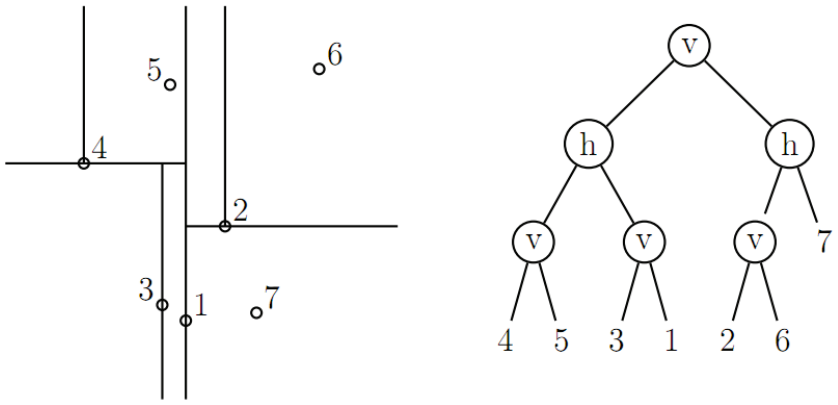


FIGURE 4.4.3 – Les points numérotés sont récursivement séparés par des hyperplans (des droites dans ce cas) horizontalement et verticalement. Le partitionnement est représenté dans un arbre dans lequel les deux fils d'un sommet représentent les points situés de chaque côté de l'hyperplan. Chaque niveau de l'arbre est une compression des données à un niveau de précision donné.

Accélération de l'étape M. Une manière d'accélérer l'étape M est de simplifier le problème de maximisation. Pour cela, les approches de type GEM (Generalized EM) ([69]) proposent simplement d'améliorer la valeur de la fonction de vraisemblance plutôt que de chercher à la maximiser. On trouve également dans cette catégorie d'approches la méthode ECM (Expectation Conditional Maximization) ([132]).

Une autre façon de réduire la complexité du problème de maximisation est de réduire le nombre de paramètres selon lesquels la vraisemblance complétée est maximisée. La méthode CEMM (Component-wise EM algorithm for Mixtures) ([46, 47]) est basée sur une décomposition du vecteur de paramètres inconnus Θ. A chaque étape M, seuls les paramètres d'une composante du mélange sont mis à jour. Ce procédé a pour but d'éviter les situations où l'algorithme EM convergence lentement. On retrouve la même idée que pour IEM mais appliquée aux paramètres et non plus aux données. Le principe est illustré Figure 4.4.4.

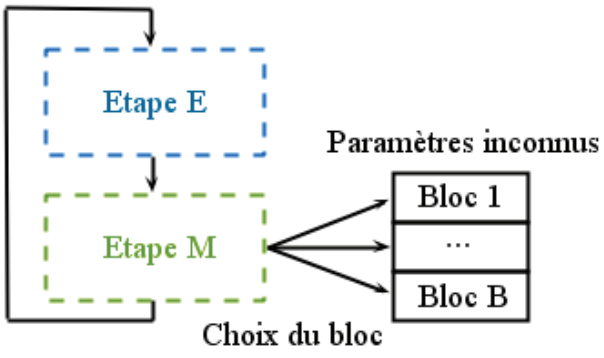


FIGURE 4.4.4 – Schéma de principe de CEMM. Seul un sous-ensemble des paramètres est mis à jour.

4.4.1.2 Application à l'estimation du modèle de mélange sphérique

Dans cette section, nous nous inspirons des méthodes présentées précédemment pour accélérer l'algorithme d'estimation dans le cadre de notre modèle de mélange sphérique. Le choix de la stratégie d'accélération est guidé par l'observation de l'évolution des composantes durant l'optimisation. La Figure 4.4.6 représente l'évolution des composantes du modèle de mélange estimé à chaque itération de EM pour un nombre de composantes fixé à $K = 7$. On s'aperçoit que certaines composantes convergent plus rapidement que d'autres et n'évoluent plus. Sur l'exemple de la Figure, les deux sphères proches des têtes n'évoluent plus après les premières itérations de EM alors que les paramètres des autres sphères sont toujours modifiés aux itérations suivantes. Par conséquent, les étapes E et M n'ont plus lieu d'être pour ces composantes particulières puisque les probabilités d'appartenance et les paramètres estimés seront toujours les mêmes. Ce phénomène est d'autant plus vérifié lorsque le nombre de composantes K est choisi automatiquement. En effet, lorsque ce paramètre est modifié, il arrive souvent que peu de composantes soient significativement touchées par la modification. Par exemple, la suppression d'une composante peut ne pas altérer les classes trop éloignées de la sphère supprimée.

Cette observation nous conduit donc à ne plus effectuer les étapes E et M des composantes considérées comme ayant déjà convergé. On considère qu'une composante a peu évolué si ses deux centres successifs $\mu^{(j)}$ et $\mu^{(j+1)}$ estimés aux itérations j et $j+1$ de EM sont proches, c'est à dire

$$\left\| \mu^{(j+1)} - \mu^{(j)} \right\| < \epsilon_{conv} \tag{4.4.2}$$

où ϵ_{conv} représente la distance en dessous de laquelle le déplacement d'une composante n'est pas significatif. Une composante dont le centre μ n'évolue plus suffisamment N_{conv} itérations de suite est considérée comme ayant convergé. Comme pour les méthodes SpEM et LEM, la composante est remise à jour lorsqu'elle ne l'a plus été pendant au moins N_{MAJ} itérations. Les entiers naturels N_{conv} et N_{MAJ} sont des paramètres de la méthode.

Le test (4.4.2) ne fait pas intervenir directement le rayon r. Toutefois, le centre et le rayon sont estimés conjointement dans le processus itératif de backfitting. Notons enfin que la distance entre les centres successifs est déjà utilisée pour tester l'arrêt de l'algorithme EM.

Sur l'exemple de la Figure 4.4.6, 14 itérations de EM ont été effectuées. La mise en œuvre de la stratégie d'accélération réduit le nombre d'itérations à 9, ce qui constitue un gain non négligeable. Les paramètres ont été fixés à $\epsilon_{conv} = 5$ mm, $N_{conv} = 2$ et $N_{MAJ} = 7$. La technique est très simple à mettre en place. En revanche, rien ne garantit que les paramètres finaux estimés seront identiques. Le résultat obtenu avec la méthode proposée est représenté Figure 4.4.5.

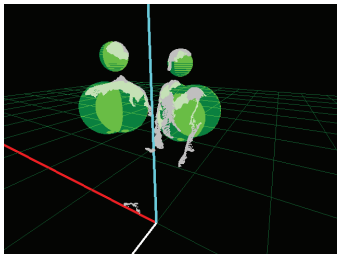


FIGURE 4.4.5 – Résultat obtenu avec l'accélération proposée après 9 itérations de EM.

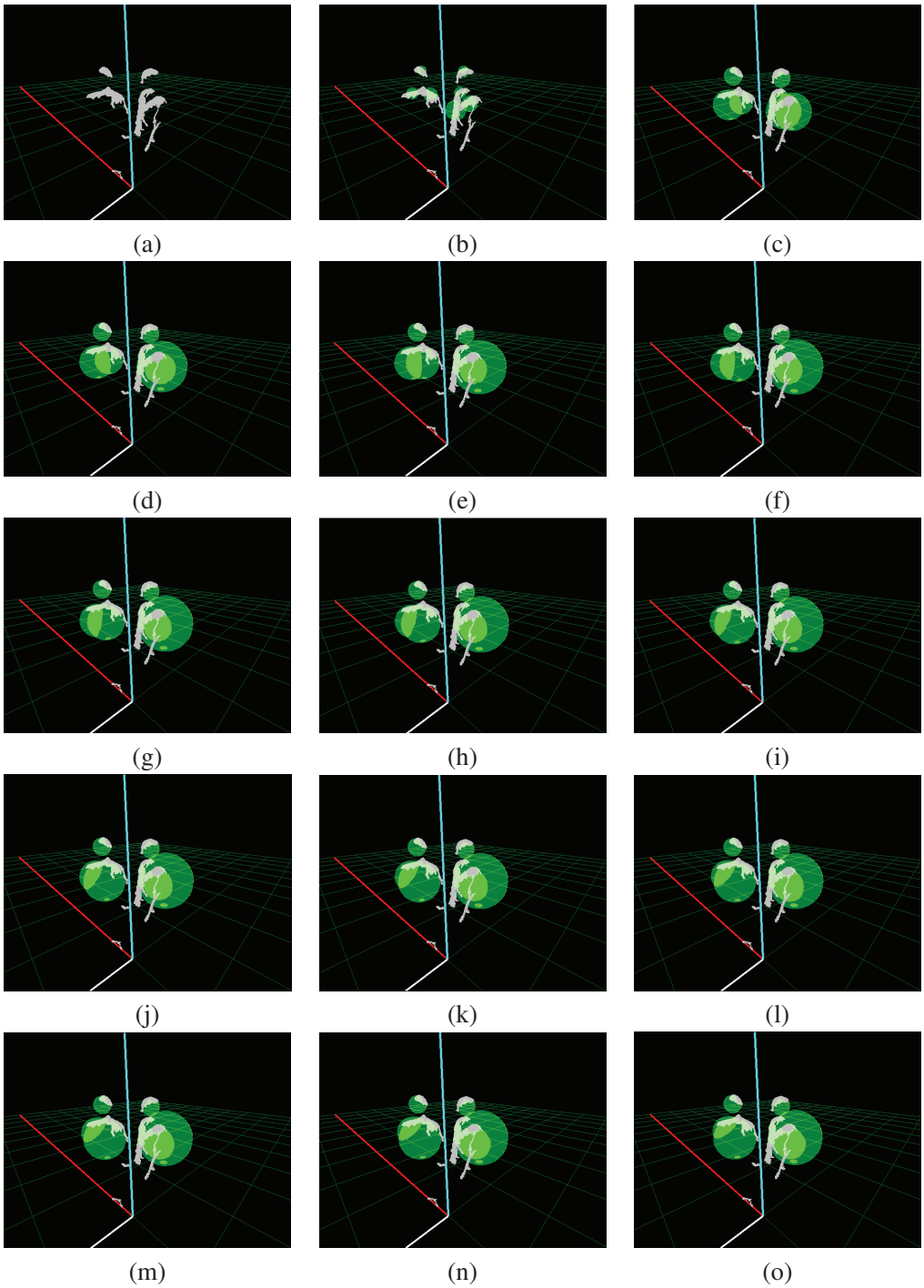


FIGURE 4.4.6 – En (a) le nuage de points 3D, en (b) l'initialisation par l'algorithme K-means et en (c-o) l'évolution du modèle à chaque itération de EM. Le nombre de composantes a été fixé à $K = 7$.

En pratique, une observation x_i éloignée d'une sphère C_ℓ présente une probabilité d'appartenance $t_{i\ell}$ numériquement très proche de zéro (mais non nulle) et sera prise en compte dans tous les calculs. Ces probabilités n'influant que très peu les résultats, nous décidons de forcer à zéro les $t_{i\ell}$ dont la valeur est trop faible, c'est à dire vérifiant

$$t_{i\ell} < \epsilon_{proba} \tag{4.4.3}$$

où ϵ_{proba} est la valeur minimale acceptée pour une probabilité d'appartenance. Le point x_i ne sera alors pas pris en compte lors des calculs concernant la composante C_ℓ. Ce simple test de nullité permet de ne pas effectuer un certain nombre d'opérations lors des étapes E et M et participe à la réduction du coût calculatoire de la méthode d'estimation.

4.4.2 Parallélisation de l'algorithme EM

La complexité calculatoire de la méthode d'estimation a été réduite algorithmiquement dans la section précédente. La parallélisation des calculs constitue une autre voie intéressante pour réduire le temps de traitement. Dans cette section, on note n le nombre d'observations du nuage de points 3D, K le nombre de composantes du modèle de mélange et N le nombre de threads concurrents disponibles (numérotés de 0 à $N-1$) pour effectuer les calculs. Les techniques les plus courantes de parallélisation de l'algorithme EM sont détaillées en section 4.4.2.1 et leur application à notre problème est exposée section 4.4.2.2.

4.4.2.1 Parallélisation des étapes E et M

Commençons par rappeler brièvement l'algorithme d'estimation du modèle de mélange sphérique dont la densité de probabilité est de la forme

$$h\left(x \mid \Theta\right) = \sum_{k=1}^{K} \pi_k \, f\left(x \mid \theta_k\right) \tag{4.4.4}$$

où f est la densité sphérique commune à chaque composante, les π_k sont les proportions du mélange telles que $\pi_k \geq 0$, $1 \leq k \leq K$ et $\sum_{k=1}^{K} \pi_k = 1$ et où $\Theta = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ est le vecteur contenant tous les paramètres du mélange. L'algorithme EM permet d'estimer Θ et consiste en deux étapes E et M itérées jusqu'à convergence.

– **Étape E :** Estimer les probabilités d'appartenance t_{il} des observations $(x_i)_{i=1}^{n}$ aux classes $(C_\ell)_{\ell=1}^{K}$ à partir de l'estimation courante $\Theta^{(j)}$

$$t_{i\ell} = \frac{\pi_\ell^{(j)} \, f\left(x_i \mid \theta_\ell^{(j)}\right)}{\sum_{k=1}^{K} \pi_k^{(j)} \, f\left(x_i \mid \theta_k^{(j)}\right)}. \tag{4.4.5}$$

– **Étape M :** Déterminer le nouveau vecteur de paramètres $\Theta^{(j+1)}$ avec les relations

$$\begin{cases} \widehat{\mu_\ell} = \dfrac{1}{\sum_{i=1}^n t_{i\ell}} \left(\displaystyle\sum_{i=1}^n t_{i\ell}\, x_i - \widehat{r_\ell} \sum_{i=1}^n t_{il}\, \frac{(x_i - \widehat{\mu_\ell})}{\|x_i - \widehat{\mu_\ell}\|} \right) \\ \widehat{r_\ell} = \dfrac{\sum_{i=1}^n t_{i\ell}\, \|x_i - \widehat{\mu_\ell}\|}{\sum_{i=1}^n t_{i\ell}} \\ \widehat{\sigma_\ell^2} = \dfrac{\sum_{i=1}^n t_{i\ell}\, (\|x_i - \widehat{\mu_\ell}\| - \widehat{r_\ell})^2}{\sum_{i=1}^n t_{i\ell}} \\ \widehat{\pi_\ell} = \dfrac{1}{n} \displaystyle\sum_{i=1}^n t_{i\ell}. \end{cases} \tag{4.4.6}$$

Le processus est initialisé par un vecteur $\Theta^{(0)}$ obtenu à l'aide d'un algorithme K-means. Les paramètres sont mis à jour à l'étape M grâce à un processus de backfitting dans lequel les estimations de μ et r (4.4.6) sont itérativement ré-évaluées.

La parallélisation des calculs de l'algorithme EM peut être effectuée du point de vue des observations $\{x_i\}_{i=1}^n$ ou des composantes $(C_k)_{k=1}^K$. Chacun des N threads disponibles peut être affecté à un sous-ensemble des données ou à un sous-ensemble des composantes. Selon la stratégie choisie, une fusion des résultats calculés par chaque processus peut être nécessaire. Nous nous intéressons dans cette section aux deux types de parallélisation.

Une version parallèle de l'algorithme EM a été proposée dans le cadre de la classification de documents dans [115]. Dans ces travaux, le nombre de documents à traiter est très important et le temps de calcul total élevé. Dans le même esprit, des accélérations ont été mises en place dans le contexte de la modélisation des contours d'une image par des segments dans [67, 68].

Parallélisation de l'étape E : L'estimation des probabilités d'appartenance lors de l'étape E est indépendante pour chaque observation et peut donc être effectuée en parallèle. Un sous-ensemble des observations est donc affecté à chaque thread. Ainsi, le thread d'indice $0 \leq l \leq N-1$ prend en charge les observations d'indices $l\ n\ /\ N$ à $(l+1)\ n\ /\ N$ et s'occupe de calculer les probabilités correspondantes pour toute composante k. Les paramètres du modèle de mélange à l'itération j doivent être connus par toutes les exécutions parallèles puisque la relation (4.4.6) fait intervenir tous les paramètres. Cette stratégie ne nécessite pas de communication entre les threads pendant le calcul car seul $\Theta^{(j)}$ est requis. Aucun problème d'accès mémoire concurrents ne se présente puisque les paramètres ne sont utilisés qu'en lecture. La méthode d'accélération est illustrée Figure 4.4.7.

Les observations sont partitionnées en N sous-ensembles chacun traité par un thread. Des paramètres intermédiaires dits locaux et notés $\Theta_{loc}^{(l)}$ peuvent êtres calculés et représentent les nouveaux paramètres estimés seulement à partir des observations assignées au thread d'indice l. Les paramètres locaux $\left\{\Theta_{loc}^{(l)}\right\}_{l=0}^N$ pourront ensuite être combinés lors de l'étape M pour calculer les nouveaux paramètres globaux $\Theta^{(j+1)}$.

La parallélisation de l'étape E sur les observations semble plus efficace qu'une parallélisation sur les composantes à cause du terme de normalisation (dénominateur de (4.4.5)). Le calcul d'une probabilité $t_{i\ell}$ ne peut être conclu tant que tous les $\{t_{ik}\}_{k=1}^K$ n'ont pas été estimés. Un second parcours

de la matrice contenant les probabilités serait nécessaire pour effectuer la normalisation, ce qui serait coûteux. Pour cette raison, une parallélisation sur les données est généralement préférée.

Parallélisation de l'étape M : Comme mentionné dans le paragraphe précédent, l'étape M peut être parallélisée en utilisant les paramètres locaux calculés durant l'étape E parallèle. Chaque thread est affecté à un certain nombre de composantes et est chargé de combiner les estimations partielles pour obtenir les nouveaux paramètres $\Theta^{(j+1)}$. Cette stratégie de parallélisation de EM est illustrée par le schéma de la Figure 4.4.7.

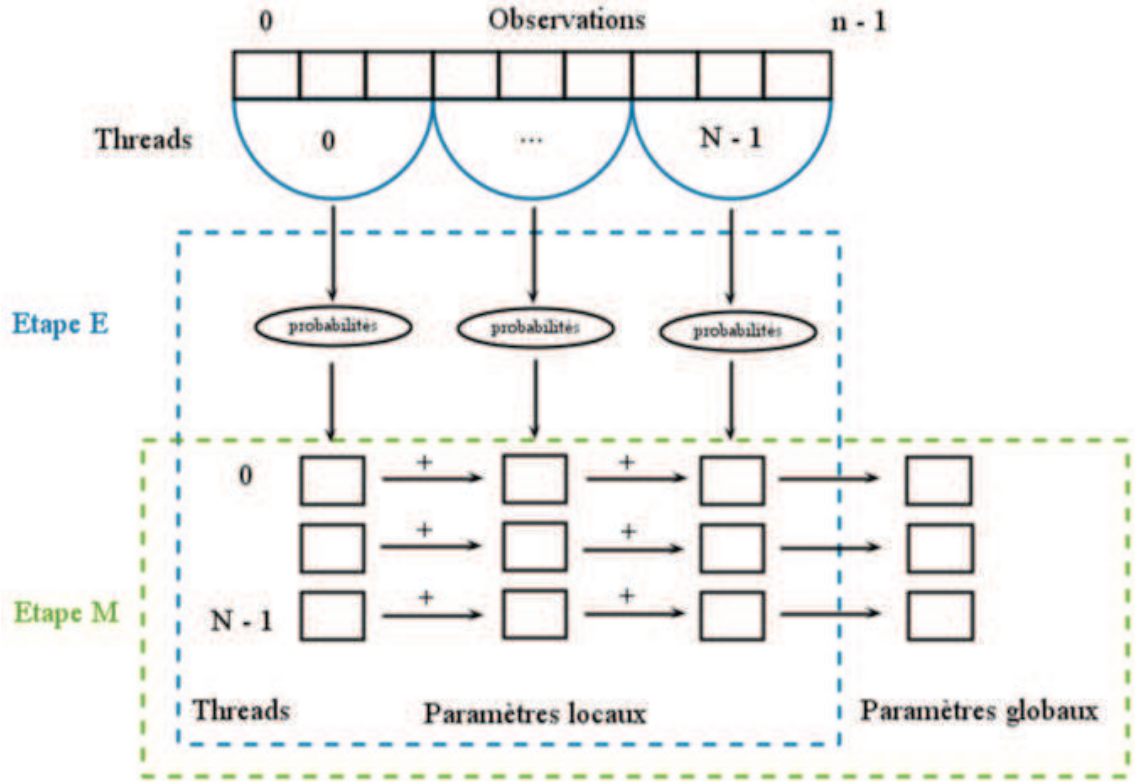


FIGURE 4.4.7 – Schéma de principe de la parallélisation de l'algorithme EM proposée dans [115].

Une version parallèle de EM minimisant la recopie des données entre les différents threads a été proposée dans [51] dans le contexte de la reconstruction d'images tomographiques. Des parallélisations GPU de EM ont également récemment été développées principalement pour les modèles de mélange gaussiens (voir par exemple [12, 151, 97]).

Quelque soit la stratégie d'accélération choisie, les étapes E et M sont chacune parallélisées. Une synchronisation de tous les threads doit être mise en place pour que l'étape M ne puisse commencer que lorsque tous les threads de l'étape E ont terminé leur travail. De même, si une combinaison des résultats est nécessaire, il faut s'assurer que tous les threads aient terminé leur calcul. Ce point soulève la question de la répartition de la quantité de calculs dans les différents threads.

4.4.2.2 Application à l'estimation du modèle de mélange sphérique

L'objet de cette section est d'appliquer les stratégies de parallélisation décrites dans la section précédente au modèle de mélange sphérique. Le schéma de la Figure 4.4.8 résume la méthode finalement retenue.

La parallélisation de l'étape E est effectuée comme décrit dans la section précédente. Chaque thread est affecté à un sous-ensemble des observations pour lesquelles les probabilités d'appartenance des points aux classes sont calculées. Les paramètres du modèle sont communs à tous les threads chargés de remplir un sous-ensemble des lignes de la matrice de probabilité.

La parallélisation de l'étape M décrite dans la section précédente pose un certain nombre de problèmes. Du fait du processus de backfitting qui a été introduit, les estimateurs n'ont pas de formulation explicite simple. La méthode consisterait alors à appliquer un backfitting sur les points associés à chacune des classes puis à combiner les différents résultats. La combinaison des estimations de chacun des threads n'est alors pas évidente.

Par conséquent, l'étape M est parallélisée selon les composantes et aucun calcul intermédiaire n'est effectué lors de l'étape E. Chaque composante est traitée indépendamment et chaque thread effectuera un backfitting sur l'ensemble des observations.

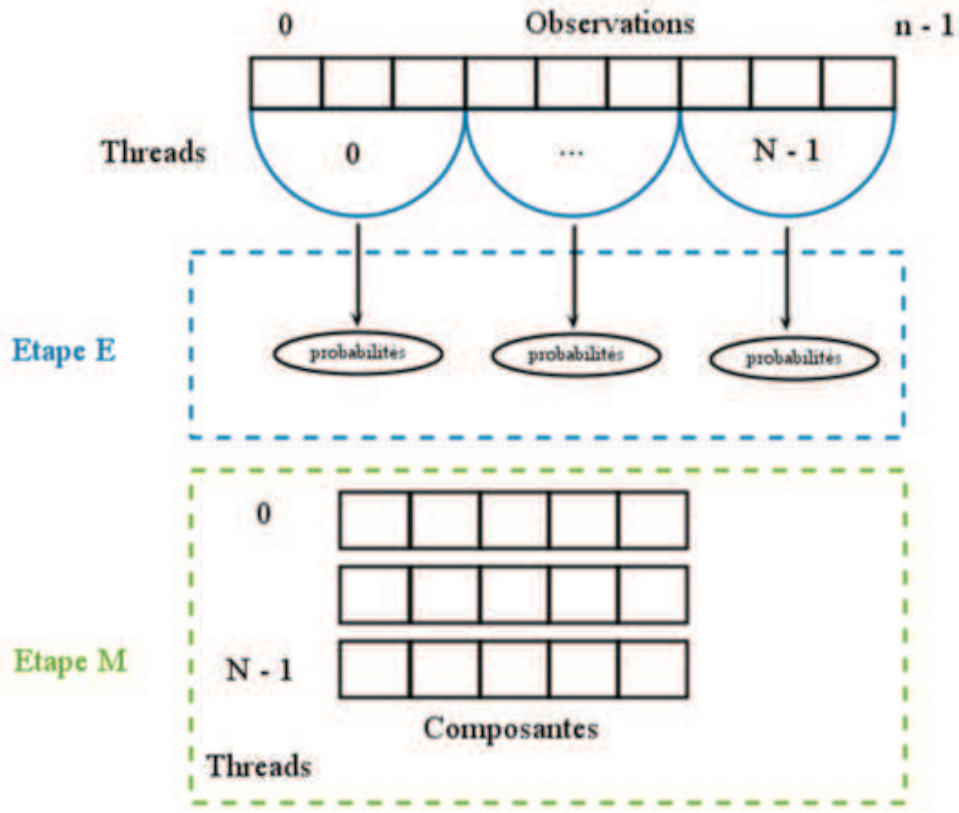


FIGURE 4.4.8 – Schéma de la méthode de parallélisation de EM retenue.

4.4.3 Initialisation efficace de l'étape M

La méthode d'estimation du modèle de mélange sphérique décrite au Chapitre 3 est basée sur un algorithme EM. L'étape de maximisation fait intervenir un algorithme itératif de type backfitting (BF) pour estimer les différents paramètres. Jusqu'à maintenant, ce processus itératif a été initialisé par la moyenne empirique des échantillons.

Dans le but de réduire le nombre d'itérations de BF, on cherche à initialiser le processus par un couple $\left(\widehat{\mu}_n^{(0)}, \widehat{r}_n^{(0)}\right)$ plus proche de la valeur optimale du paramètre. Autrement dit, on souhaite à partir d'un ensemble d'observations réparties autour d'une sphère tronquée estimer plus précisément le centre et le rayon. On peut alors s'attendre à ce que pour un critère d'arrêt donné, le nombre d'itérations de BF nécessaires soit plus faible et donc que le temps de calcul s'en trouve naturellement réduit.

Le centre $\mu \in \mathbb{R}^2$ d'un cercle $\mathcal{C}$ peut être simplement estimé par la *méthode des cordes*. Soient A, B et C trois points distincts appartenant au cercle et représentés Figure 4.4.9. Les médiatrices des cordes $[AB]$ et $[BC]$ passent par le centre μ du cercle. Trois points distincts sont donc suffisants pour déterminer le centre d'un cercle. Cette propriété des cordes du cercle nous conduit à une méthode d'estimation simple du centre. Il suffit de calculer le point d'intersection des 3 médiatrices.

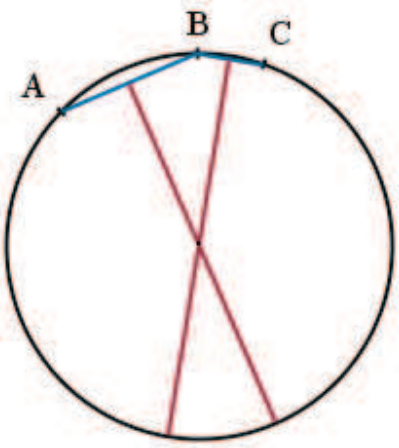


FIGURE 4.4.9 – Propriétés des cordes d'un cercle. Les médiatrices (en rouge) des cordes $[AB]$ et $[BC]$ (en bleu) passent par le centre du cercle.

La propriété des cordes peut être étendue au cas de la sphère qui nous intéresse plus particulièrement. Soient A, B, C et D 4 points non coplanaires appartenant à la sphère $\mathcal{S}$. Les plans médiateurs des cordes (par exemple) $[AB]$, $[BC]$ et $[BD]$ passent tous trois par le centre $\mu \in \mathbb{R}^3$ de la sphère. Quatre points distincts sont donc suffisant pour déterminer le centre d'une sphère. Cette propriété des cordes de la sphère nous conduit à une méthode d'estimation simple du centre. Il suffit de calculer le point d'intersection des 3 plans médiateurs.

La propriété énoncée peut être utilisée pour estimer le centre d'une sphère à partir d'un ensemble d'observations répartis sur seulement une portion de surface. On commence par choisir aléatoirement 4 points $A = (a_x, a_y, a_z)$, $B = (b_x, b_y, b_z)$, $C = (c_x, c_y, c_z)$ et $D = (d_x, d_y, d_z)$ non coplanaires dans l'échantillon. On calcule ensuite à partir de trois couples de points les équations des trois plans média-

teurs. Un plan P_i est caractérisé par quatre coefficients réels $(\alpha_i, \beta_i, \gamma_i, \delta_i)$ et son équation cartésienne s'écrit

$$\alpha_i\, x + \beta_i\, y + \gamma_i\, z + \delta_i = 0. \tag{4.4.7}$$

Par exemple, le plan médiateur de la corde $[AB]$ peut être calculé comme suit

$$(\alpha_i, \beta_i, \gamma_i) = (b_x - a_x, b_y - a_y, b_z - a_z) \tag{4.4.8}$$

$$\alpha_i\,(a_x + b_x) + \beta_i\,(a_y + b_y) + \gamma_i\,(a_z + b_z) + 2\,\delta_i = 0. \tag{4.4.9}$$

Le point d'intersection $X = (x, y, z)^T$ des plans P_1, P_2 et P_3 est obtenu en résolvant le système $M\,X = \Delta$ avec

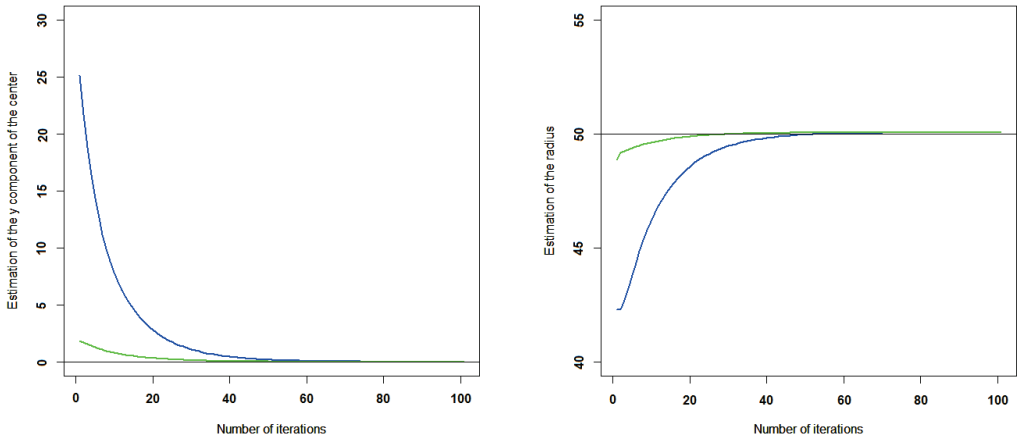
$$M = \begin{pmatrix} \alpha_1 & \beta_1 & \gamma_1 \\ \alpha_2 & \beta_2 & \gamma_2 \\ \alpha_3 & \beta_3 & \gamma_3 \end{pmatrix} \tag{4.4.10}$$

et

$$\Delta = \begin{pmatrix} -\delta_1 \\ -\delta_2 \\ -\delta_3 \end{pmatrix}. \tag{4.4.11}$$

La matrice M est inversible lorsque les plans ne sont pas parallèles entre eux, c'est à dire lorsque les points A, B, C et D sont non coplanaires.

Pour améliorer l'estimation du centre, le procédé décrit ci-dessus est répété N fois pour obtenir N estimations du centre de la sphère. Les valeurs médianes sont retenues pour chacune des coordonnées et constituent l'estimation du centre $\widehat{\mu}_n^{(0)}$. Le rayon est enfin estimé par la distance moyenne $\widehat{r}_n^{(0)}$ des observations à $\widehat{\mu}_n^{(0)}$.



(a) (b)

FIGURE 4.4.10 – Estimations des paramètres μ_y en (a) et r en (b) en fonction du nombre d'itérations de BF effectuées dans le cas de la demi-sphère et pour un critère d'arrêt donné. L'étape M a été initialisée par le centroïde (bleu) et par la méthode des cordes (vert).

Le gain en termes de nombre d'itérations est évalué par les courbes de la Figure 4.4.10. Un échantillon de taille $n = 1000$ a été généré sur la demi-sphère (tronquée selon la composante y) de paramètres $\mu = (0, 0, 0)^t$ et $r = 50$. Pour un critère d'arrêt donné, le centre et le rayon sont estimés par l'algorithme BF initialisé par le centroïde et la méthode des cordes. Il est clair que l'estimation du centre obtenue par la méthode des cordes est plus proche du centre μ que le centroïde. Le nombre d'itérations effectuées peut alors être significativement réduit.

4.5 Expérimentations sur des données réelles

Dans cette section, la méthode de détection de têtes est expérimentée sur des données réelles. L'algorithme décrit en section 4.2 est appliqué à une séquence d'images acquises par le système présenté en section 4.2.1. La séquence est composée de 1537 images représentant de 0 à 2 personnes et dont les nuages de points 3D (en entrée du module de segmentation par des sphères, après la soustraction de fond) contiennent entre 8000 et 30000 points. Dans ces expérimentations, on s'intéresse à la segmentation du nuage de points et à la bonne détection des têtes des individus.

Le nombre de composantes K du modèle de mélange sphérique est choisi automatiquement par la technique décrite en section 4.3.2 avec un nombre initial de $K_0 = 15$ classes. Cette valeur initiale nous assure que suffisamment de composantes sont disponibles pour expliquer les données. Elle convient pour notre application où la zone de traitement est étroite. La méthode est implémentée en langage C++ sur une machine équipée d'un processeur Intel(R) Core(TM) i7-2720QM à 2,20GHz avec 8Go de RAM et parallélisée avec OpenMP comme expliqué en section 4.4.2.2. L'accélération proposée en section 4.4.1.2, consistant à ne pas effectuer les étapes E et M pour les composantes considérées comme ayant déjà convergé, est appliquée avec $\epsilon_{conv} = 5$ mm, $N_{conv} = 2$, $N_{MAJ} = 7$ et $\epsilon_{proba} = 10^{-15}$.

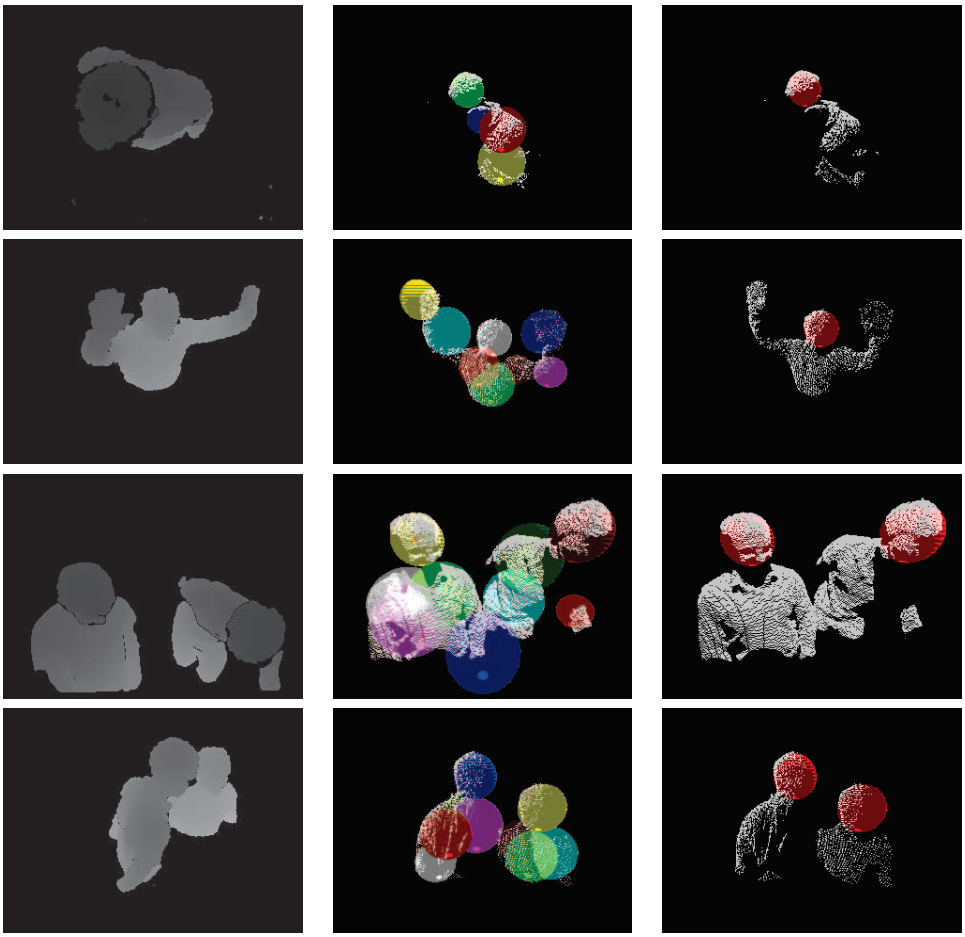
Une fois les paramètres du modèle de mélange estimés, nous appliquons les règles de décision décrites en section 4.2.2 pour détecter les têtes parmi les composantes optimisées. Les paramètres ont été fixés à $r_{min} = 70$ mm, $r_{max} = 120$ mm, $\epsilon_{erreur} = 20$ mm et $z_{min} = 1000$ mm.

Les taux de détection constatés sont donnés Table 4.5.1. Les taux de bonne détection sont élevés et les taux de détection de non têtes, c'est à dire lorsque le nombre de tête est correct mais que la modélisation est mauvaise (détection d'une épaule comme une tête par exemple) est très faible. Sur cette séquence de test, les têtes ont été correctement détectées et modélisées dans plus de 98 % des cas. Les résultats révèlent que la méthode a plus tendance à surestimer le nombre de têtes en présence d'un seul individu et à le sous estimer en présence de deux individus, mais les taux d'erreurs restent faibles.

EM		Number of detected heads				Non head detections	Total
	X	0	1	2	3		
Number of heads in the image	0	100	0	0	0	0	100
	1	0	97.92	1.98	0.1	0	100
	2	0	0.61	98.98	0	0.41	100

TABLE 4.5.1 – Taux de détection (EM) pour chaque nombre réel de têtes dans l'image. La non détection correspond aux images pour lesquelles le nombre de têtes est correct mais les sphères détectées ne sont pas sur une tête.

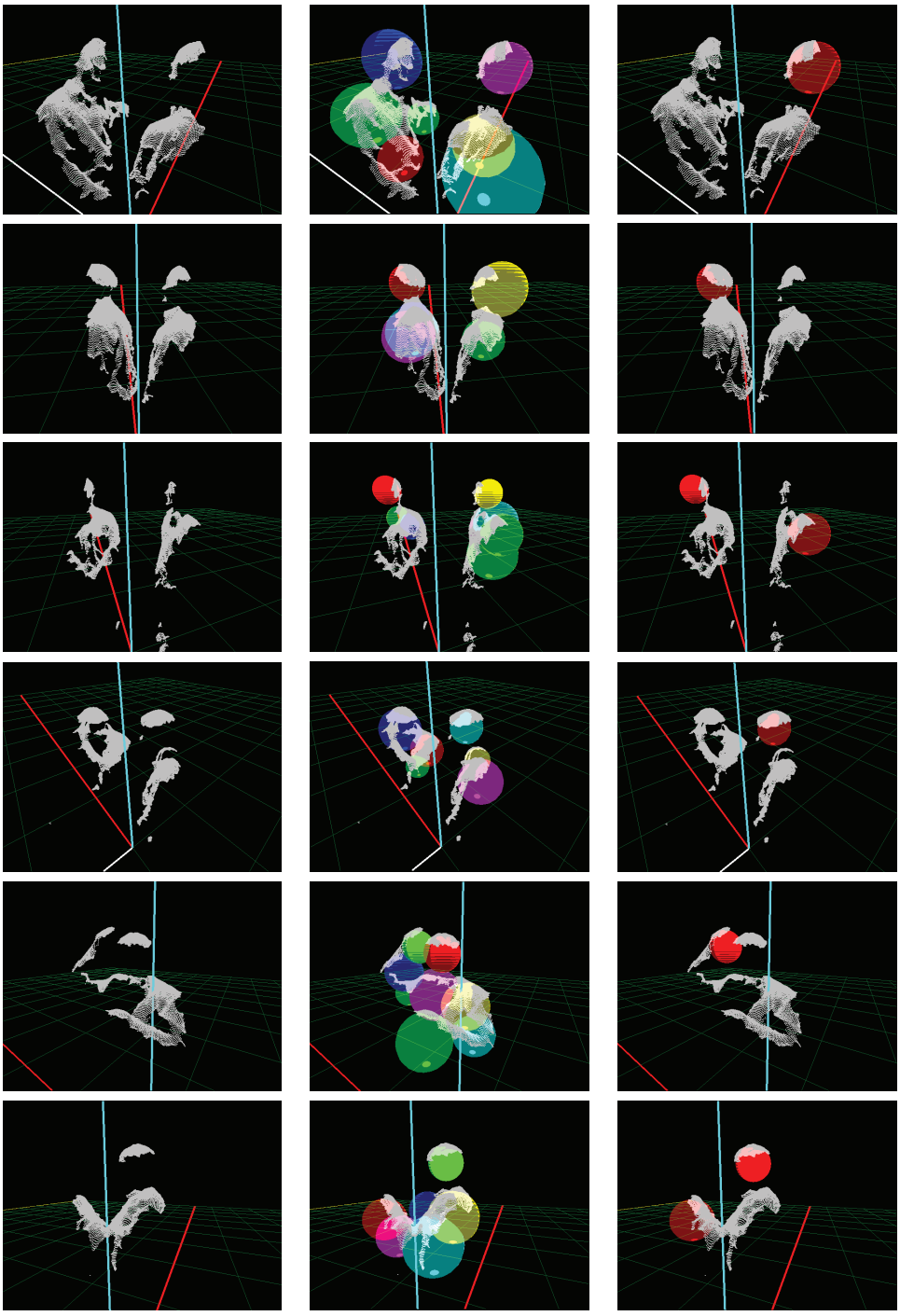
Les modèles de mélange sphérique estimés ainsi que les têtes détectées sont représentés Figure 4.5.1. Les surfaces des composantes sont proches des observations et les centres bien placés à l'intérieur des objets. Les optimisations algorithmiques mises en place ne perturbent pas la bonne estimation des paramètres. Les nombres de classes choisis automatiquement semblent corrects. Les données n'ont pas l'air à priori sur ou sous segmentées, ce qui explique en partie les bons taux de détection.



(a) (b) (c)

FIGURE 4.5.1 – Résultats obtenus avec l'algorithme EM. En (a), les nuages de points 3D, en (b) les modèles estimés et en (c) les têtes détectées.

Essayons maintenant de comprendre pourquoi la méthode échoue sur certaines images. La Figure 4.5.2 contient des cas où l'algorithme de détection échoue. Dans les quatre premiers exemples, l'une des têtes n'est pas détectée. Cela est dû dans les trois premiers cas au fait que la tête est relevée et vue de face. La valeur du rayon et la répartition des points sur les hémisphères ne respectent plus les contraintes des règles de détection. Dans le quatrième cas, une main est en contact avec la tête et le rayon de la composante est par conséquent plus important. Dans cet exemple précis, la non détection de la tête ouvre la possibilité à la composante située sur l'épaule d'être détectée comme étant une tête. La tête du cinquième exemple a bien été modélisée mais une autre sphère située sur la main, proche et légèrement plus haute a été détectée à la place. Enfin, le dernier cas illustre une fausse détection où les deux mains ont été détectées comme étant une tête.



(a) (b) (c)

FIGURE 4.5.2 – Illustration des cas où l'algorithme de détection de têtes échoue. En (a) les nuages de points 3D, en (b) les modèles estimés et en (c) les têtes détectées.

Malgré ces cas de mauvaises détections, on constate que lorsqu'une tête n'a pas été détectée, cela est plutôt dû aux règles de détection mises en place qu'à une mauvaise modélisation des données. Autrement dit, on a presque toujours en pratique une sphère sur chacune des têtes des individus. Par conséquent, une voie naturelle d'amélioration de la méthode serait de mettre en place des règles de détection plus complexes, par exemple à l'aide un algorithme d'apprentissage permettant de mieux classer les composantes.

Nous nous intéressons maintenant à la réduction du temps de traitement obtenue grâce aux diverses améliorations proposées en sections 4.4.1 et 4.4.2. Les temps d'estimation du modèle de mélange sphérique sont donnés Tables 4.5.2, 4.5.3 et 4.5.4. Les temps de calcul sont des temps moyens obtenus sur 10 exécutions de la méthode (le temps peut varier légèrement d'une exécution à l'autre). Les données sont constituées de 18027 points 3D représentant deux personnes. Les paramètres de critère d'arrêt du backfitting sont fixés à $\epsilon_{\mu}^{BF} = 1$ mm, $\epsilon_{r}^{BF} = 1$ mm et $N_{max}^{BF} = 100$ itérations et ceux de l'algorithme EM à $\epsilon_{param} = 5$ mm et $N_{max}^{EM} = 20$ itérations. Le nombre de threads est choisi le plus élevé possible (via une fonction OpenMP) et s'élève dans notre configuration à $N = 8$. Afin de se rendre compte de l'impact de chaque amélioration proposée, le temps de calcul est mesuré pour toutes les combinaisons possibles des différentes versions de la méthode et pour un nombre de classes K fixé ou variable.

Les résultats des Tables 4.5.2 et 4.5.3 révèlent que les deux améliorations proposées réduisent significativement le temps de traitement. Les gains relatifs sont proches pour un nombre de classes fixé ou variable. La parallélisation permet de diminuer le temps de calcul d'environ 53 % et l'accélération d'environ 40 %. Leur combinaison permet de le réduire encore d'avantage pour atteindre une réduction de près de 65 % du temps de calcul. Rappelons que ces valeurs ont été obtenues avec certains paramètres (nombre de classes initial, critères d'arrêt, ...) sur un certain nuage de points 3D. Ces tests valident l'utilisation des stratégies d'accélération et de parallélisation mises en œuvre.

Temps moyen	Sans accélération	Avec accélération
Sans parallélisation	263 ms	145 ms
Avec parallélisation	126 ms	96 ms

TABLE 4.5.2 – Temps de traitements moyens avec et sans les améliorations proposées pour un nombre de composantes fixé à $K = 7$.

Temps moyen	Sans accélération	Avec accélération
Sans parallélisation	300 ms	182 ms
Avec parallélisation	142 ms	105 ms

TABLE 4.5.3 – Temps de traitements moyens avec et sans les améliorations proposées pour un nombre de composantes choisi automatiquement.

Nous avons proposé en section 4.4.3 une initialisation de l'étape M par la méthode des cordes fournissant une estimation plus précise des paramètres initiaux de la sphère associée à une classe. Toutefois, comme le montrent les résultats de la Table 4.5.4, les temps de calcul sont plus élevés et n'ont pas été réduits. En observant plus précisémment le comportement de l'algorithme, on se rend compte que le nombre d'itérations de EM effectuées est beaucoup plus important. Sur le nuage de points 3D utilisé lors des différents tests et pour un nombre de classes $K = 7$ fixé, 9 itérations de

EM sont nécessaires avec l'initialisation par le centroïde et 20 avec l'initialisation par la méthode des cordes (nombre maximal d'itérations atteint). Cette différence s'explique par la sélection aléatoire des points nécessaires à l'estimation du centre et du rayon. Sur un objet assez plat, par exemple le buste, la sphère estimée par la méthode des cordes dépendra fortement des points sélectionnés. Par conséquent, la composante ne convergera pas, le critère d'arrêt de EM basé sur le déplacement maximal d'une composante ne sera jamais satisfait et le nombre maximal d'itérations sera atteint. Le coût calculatoire supplémentaire induit par le grand nombre d'itérations supplémentaires de EM est supérieur au gain en nombre d'itérations du backfitting. L'initialisation de l'étape M par la méthode des cordes n'est donc pas efficace sur nos données réelles bien qu'elle le soit dans le cas d'une seule composante sur des données simulées. Cette modification de l'algorithme n'est donc pas conservée.

Méthode	Initialisation par le centroïde	Initialisation par la méthode des cordes
$K = 7$	263 ms	488 ms
K variable	300 ms	1663 ms

TABLE 4.5.4 – Temps de traitements moyens avec les deux méthodes d'initialisation de l'étape M et pour un nombre de composantes fixé à $K = 7$ ou choisi automatiquement.

Les temps de calcul donnés jusqu'à présent ne concernent que l'estimation du modèle de mélange sphérique. Les temps de chacune des étapes de l'algorithme global sont détaillés Table 4.5.5. Le nombre de classes est choisi automatiquement et les améliorations retenues sont appliquées. Les images sont de taille de 320×240 pixels. Il est clair que l'estimation du modèle de mélange sphérique est l'étape de loin la plus coûteuse en temps de calcul.

Traitement	Temps moyen
Segmentation des individus et calcul du nuage de points 3D	2 ms
Pré-traitement du nuage de points 3D	4 ms
Estimation du modèle de mélange sphérique	105 ms
Détection des têtes	3 ms
Temps total	**114 ms**

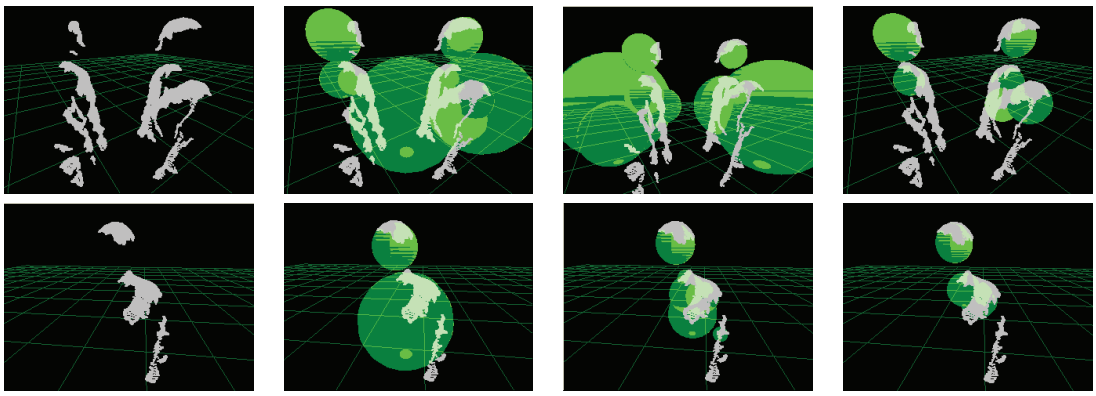
TABLE 4.5.5 – Temps de traitements moyens des différentes étapes de l'algorithme global.

La méthode de détection proposée (EM) est comparée avec une autre méthode basée sur une segmentation du nuage de points par des sphères avec l'algorithme RANSAC présenté au Chapitre 1 section 1.3.5. La chaîne de traitements est identique pour les deux méthodes comparées, sauf la phase de modélisation du nuage de points 3D. Les règles de détection de têtes utilisées sont identiques. Nous allons pouvoir mesurer l'influence de la modélisation des données sur les taux de détection.

Le paradigme RANSAC est basé sur une sélection aléatoire de points dans les données pour construire des modèles candidats qui sont ensuite évalués. Chaque modèle est évalué par un score calculé comme étant le ratio entre le nombre de points proches de la sphère (inliers) et l'erreur d'ajustement des moindres carrés. L'introduction de l'erreur d'ajustement dans le score d'un modèle candidat améliore significativement l'efficacité de la méthode de détection. Ceci est dû au fait que nos données contiennent des objets sphériques composés d'un nombre modéré d'inliers (têtes) et d'objets moins sphériques composés d'un nombre important de points. Le nombre maximal de sphères ajustées est fixé à 10, le nombre de tirages aléatoires à 50 pour chaque sphère recherchée, le seuil de

consistance à 20 mm et le score minimal pour accepter un modèle à 100.

Avant d'analyser les résultats obtenus, faisons quelques remarques sur la modélisation du nuage de points 3D. A cause de la phase aléatoire de sélection de points dans le nuage tout entier, l'algorithme RANSAC peut conduire, pour un ensemble d'observations donné, à plusieurs segmentations différentes et par conséquent modifier le résultat de la détection. On constate un manque de répétabilité dû à la nature aléatoire de la méthode. A l'inverse, la méthode EM donne toujours le même résultat et est donc plus stable. Ce point est illustré Figure 4.5.3 où différentes modélisations ont été obtenues à partir du même nuage de points. Une autre différence importante entre les deux méthodes de segmentation est que les sphères estimées par RANSAC ont, sur nos données, tendance à avoir des rayons plus importants. Cette observation est illustrée Figure 4.5.3 où le corps de la personne est modélisé par peu de sphères dont le rayon est important alors qu'il est modélisé par des sphères de rayon plus faible avec EM. Par conséquent, les composantes de rayon élevé peuvent intersecter d'autres composantes et expliquer des points appartenant à d'autres parties du corps (par exemple appartenant à une autre personne située à proximité). Les résultats obtenus avec l'algorithme RANSAC dépendent des paramètres choisis, en particulier du nombre de tirages aléatoires effectués. Pour la méthode proposée, le choix des valeurs des variances (fixées durant tout l'algorithme) explique en partie le fait que les sphères soient plus petites. Les paramètres des deux méthodes ont été choisis de manière à ce que le temps de calcul soit similaire.



(a) (b) (c) (d)

FIGURE 4.5.3 – Nuage de points 3D en (a), deux segmentations différentes obtenues avec RANSAC en (b) et (c) et segmentation obtenue avec EM en (d).

Les résultats de la Table 4.5.6 révèlent des taux de bonne détection élevés pour la méthode basée sur RANSAC. Toutefois, on observe que EM donne des résultats légèrement meilleurs, ce qui s'explique par les remarques du paragraphe précédent. De plus, le taux de détection de non têtes, c'est à dire lorsque le nombre de tête est correct mais que la modélisation est mauvaise (détection d'une épaule comme une tête par exemple), est plus faible avec EM. La modélisation du nuage de points 3D par l'algorithme proposé semble donner de meilleurs taux de détection et une modélisation plus vraisemblable des données.

RANSAC		Number of detected heads				Non head detections	Total
	X	0	1	2	3		
Number of heads in the image	0	100	0	0	0	0	100
	1	0.99	96.84	1.58	0	0.59	100
	2	0.2	5.12	90.59	0.2	3.89	100

TABLE 4.5.6 – Taux de détection (RANSAC) pour chaque nombre réel de têtes dans l'image. La non détection correspond aux images pour lesquelles le nombre de têtes est correct mais les sphères détectées ne sont pas sur une tête.

4.6 Conclusion

Dans ce chapitre, le système mis en place et l'algorithme complet de détection de têtes dans un nuage de points 3D ont été détaillés. La méthode d'estimation du modèle de mélange sphérique a été adaptée pour satisfaire les contraintes d'utilisation pratique (automatisation et complexité). Nous avons proposé une méthode de choix automatique du nombre de composantes et plusieurs modifications algorithmiques et d'implémentation visant à réduire le coût calculatoire de l'algorithme. Les règles de détection des têtes retenues ont été décrites et évaluées sur des séquences de test.

Les expérimentations ont montré que la méthode proposée donne de bons résultats sur des séquences d'images simples impliquant une ou deux personnes. Nous avons démontré la supériorité de notre algorithme de modélisation par rapport à la méthode RANSAC dans notre cadre d'application précis. La meilleure modélisation du nuage de points conduit à de meilleurs taux de détection. Les optimisations apportées nous ont permis d'atteindre un temps de calcul proche du quasi temps-réel.

La suite naturelle de ces travaux serait d'améliorer la méthode en mettant en place des règles de détection plus complexes faisant intervenir par exemple une phase d'apprentissage. Cette évolution pourrait améliorer les résultats dans les cas où la méthode actuelle a tendance à échouer, notamment lorsque les individus ont les bras levés. On pourrait également imaginer une modélisation du nuage de points par des formes géométriques plus complexes, comme par exemple l'ellipsoïde. La détection des têtes passerait alors par un test visant à décider si un ellipsoïde donné peut être apparentée à une sphère. Une tentative de généralisation du modèle sphérique au cas ellipsoïdal est justement présentée dans le chapitre suivant pour montrer que le travail réalisé présente des perspectives de recherche intéressantes.

Chapitre 5

Généralisation au modèle de mélange ellipsoïdal

Sommaire

5.1 Introduction

Dans ce chapitre, nous continuons à nous intéresser au problème de modélisation d'un nuage de points 3D par un ensemble de formes géométriques simples. Nous avons pour cela été amenés à introduire au Chapitre 3 un nouveau modèle de mélange sphérique. Une nouvelle méthode permettant de modéliser les données par un ensemble de sphères de nombre variable a été proposée. Les nuages de points provenant des capteurs 3D présentent la particularité d'être situés sur un seul côté de la surface des objets. La méthode d'estimation proposée est robuste à cette caractéristique particulière. Le choix de la forme géométrique avait été motivé par le problème de la détection de têtes.

Toutefois, la modélisation d'un nuage de points 3D par des sphères peut ne pas se révéler suffisante. En présence d'objets présentant des formes plus complexes, le modèle sphérique ne sera pas assez général. La méthode proposée précédemment aura tendance à placer une sphère de rayon important ou plusieurs sphères plus petites sur une forme non sphérique. L'application particulière visée peut aussi nécessiter d'estimer les surfaces par autre primitive géométrique que la sphère.

Par conséquent, nous cherchons dans ce chapitre à généraliser le modèle sphérique à une forme géométrique plus complexe. La généralisation naturelle de la sphère est l'ellipsoïde obtenu par déformation de cette dernière. Le nombre de degrés de liberté est plus important ce qui implique d'une part une modélisation plus souple et d'autre part un nombre de paramètres à estimer plus important. Puisque la sphère est un cas particulier de l'ellipsoïde, on espère avec un tel modèle améliorer la décomposition des objets en les décrivant par des ellipsoïdes dont certains pourront en fait s'apparenter à des sphères. Cette dernière remarque nous amène au problème du test des paramètres de l'ellipsoïde pour déterminer sa proximité avec une forme sphérique.

L'objet de ce chapitre est de construire un nouveau modèle de mélange ellipsoïdal et de proposer une méthode d'estimation de ses paramètres. L'algorithme sera expérimenté sur des données similaires à celles manipulées dans les chapitres précédents. Nous pointerons également les points communs et les différences entre les modèles sphérique et ellipsoïdal et entre les résultats obtenus. Notre démarche est similaire à celle suivie lors de l'étude du cas sphérique. La structure de ce chapitre est donc identique à celle adoptée pour le modèle sphérique.

Ce chapitre a pour but de montrer que le travail réalisé dans la thèse présente des perspectives de recherche intéressantes. Certains points sont encore incomplets et mériteraient une étude et un développement plus approfondis, en particulier la convergence des estimateurs. Nous cherchons ici à donner des pistes d'ouverture et de poursuite du travail de recherche initié.

Ce chapitre est organisé de la manière suivante. Le modèle de mélange ellipsoïdal est présenté et étudié en section 5.2. L'estimation des paramètres inconnus est traitée en section 5.3. La méthode est ensuite évaluée sur des données simulées en section 5.4 et sur des données réelles en section 5.5. Enfin, nous concluons le chapitre en section 5.6. Certains calculs sont détaillés en Annexes A, B, C et D.

5.2 Présentation du modèle de mélange ellipsoïdal

Dans le chapitre précédent, nous avons construit une densité permettant de modéliser des points répartis autour d'une surface sphérique. Nous cherchons maintenant dans cette section à généraliser le travail réalisé à une surface plus complexe : l'ellipsoïde. Pour cela, nous commençons par introduire une nouvelle densité de probabilité décrivant des points répartis autour d'une surface ellipsoïdale. Nous proposerons ensuite un modèle de mélange basé sur cette densité. Ce modèle permettrait de proposer une modélisation plus souple des données 3D. La nouvelle densité est introduite en 5.2.1 et ses principales propriétés sont énoncées en 5.2.2. Le modèle de mélange basé sur cette densité est ensuite brièvement présenté en 5.2.3.

5.2.1 Introduction de la nouvelle densité de probabilité

Notre but est de construire un modèle décrivant des points répartis autour d'une surface ellipsoïdale de $\mathbb{R}^d$. Un ellipsoïde est caractérisé par son centre $\mu \in \mathbb{R}^d$ et sa matrice de forme $\Sigma \in \mathbb{R}^{d\times d}$ symétrique définie positive. La surface ellipsoïdale est définie par

$$\left\{x \in \mathbb{R}^d, (x-\mu)^T \Sigma^{-1} (x-\mu) = 1\right\}. \tag{5.2.1}$$

Le centre μ est le point d'intersection entre les d axes et détermine la position de l'objet dans l'espace. La matrice Σ contient l'information de forme et d'orientation de la surface. Plus précisément, considérons la décomposition en éléments propres

$$\Sigma = PDP^{-1} \tag{5.2.2}$$

où D est la matrice diagonale contenant les valeurs propres $(\lambda_1, \ldots, \lambda_d)$ de Σ telles que $0 < \lambda_1 \leq \ldots \leq \lambda_d$ et P la matrice de passage dont les colonnes sont les vecteurs propres $(v_1, \ldots, v_d)$ de Σ. Les axes principaux de l'ellipsoïde sont donnés par les vecteurs propres de Σ et les longueurs des demi-axes sont les racines carrées des valeurs propres correspondantes. La Figure 5.2.1 représente une ellipse du plan notée $\mathcal{E}$ de centre $\mu = (0,0)^T$ et de matrice de forme $\Sigma = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}$. Les valeurs propres valent 1 et 4 et correspondent bien aux longueurs des demi-axes élevées au carré. Les vecteurs propres correspondants sont les axes du repère.

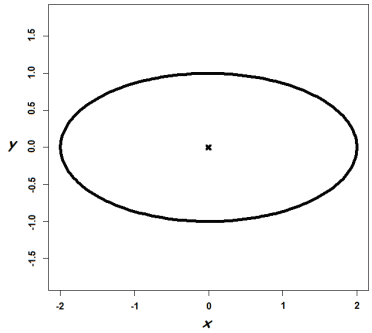


FIGURE 5.2.1 – Ellipse $\mathcal{E}$ centrée dans le plan et alignée sur les axes du repère.

Pour construire le modèle dans le cas de la sphère, nous nous étions basés sur la distance signée entre le point $x \in \mathbb{R}^d$ et la surface considérée. Malheureusement, l'expression exacte de cette distance est difficile à écrire explicitement et simplement dans le cas de l'ellipsoïde. Par conséquent, nous utilisons à la place la métrique de Mahalanobis ([124]) permettant de quantifier la distance entre le point x et l'ellipsoïde de centre μ et de matrice de forme Σ. Cette distance est définie par

$$d_{mah}(x, \mu, \Sigma) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}. \tag{5.2.3}$$

La métrique de Mahalanobis est différente de la distance Euclidienne classique et prend en compte la répartition des points dans les différentes directions de l'espace. Elle vaut 1 si x appartient à l'ellipsoïde caractérisé par (μ, Σ) et s'annule lorsque $x = \mu$. Elle est égale à la distance Euclidienne lorsque $\Sigma = I_d$. Les lignes de niveaux de la distance de Mahalanobis à l'ellipse $\mathcal{E}$ sont représentées Figure 5.2.2. On observe que la distance est différente de la distance Euclidienne classique et qu'elle varie selon les directions de l'espace. Elle constitue toutefois une information de proximité que nous allons exploiter.

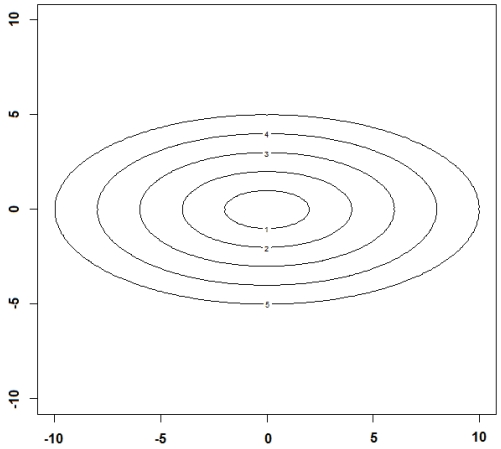


FIGURE 5.2.2 – Lignes de niveaux de la distance de Mahalanobis à l'ellipse $\mathcal{E}$.

Les remarques précédentes nous conduisent à définir la densité de probabilité suivante

$$f(x) = C_d \exp\left(-\frac{1}{2\sigma^2}\left(\sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} - 1\right)^2\right) \tag{5.2.4}$$

où $\mu \in \mathbb{R}^d$ est le centre, $\Sigma \in \mathbb{R}^{d\times d}$ la matrice de forme symétrique définie positive et $\sigma^2 > 0$ la dispersion par rapport à la surface. La constante de normalisation C_d est donnée par

$$C_d = \frac{\Gamma(d/2)}{2\pi^{d/2} \mid \Sigma \mid^{1/2} J_{d-1}(\sigma)} \tag{5.2.5}$$

où $\Gamma(\,.\,)$ désigne la fonction Gamma et pour $q \in \mathbb{N}$ et $\alpha > 0$,

$$J_q(\alpha) = \int_0^\infty t^q \exp\left(-\frac{(t-1)^2}{2\alpha^2}\right) dt. \tag{5.2.6}$$

Le centre μ caractérise la position de l'ellipsoïde, la matrice Σ sa forme et son orientation et le paramètre σ^2 contrôle la dispersion des points autour de la surface. Notons que cette densité n'est pas une généralisation au sens mathématique du terme de la densité sphérique introduite au Chapitre 3. En effet, si on pose $\Sigma = r^2\, I_d$, on ne retrouve pas exactement la densité sphérique. Ceci est dû au fait que nous n'avons pas utilisé la distance Euclidienne dans le terme exponentiel.

Cette distribution étant conçue pour modéliser des points répartis autour d'une surface ellipsoïdale, nous ne considérons que le cas où le paramètre σ est suffisamment faible pour que les fluctuations n'interagissent pas avec le côté opposé de la surface. Si le paramètre σ est choisi trop élevé, la densité génère des points situés à l'intérieur de la surface et vers le centre, ce qui n'est pas la configuration qui nous intéresse. Plus précisément, la dispersion induite par la valeur de ce paramètre doit être faible par rapport à la longueur du plus petit demi-axe de l'ellipsoïde, c'est à dire la racine carrée de la plus petite valeur propre de Σ. Cette hypothèse est analogue à celle faite dans le cas de la sphère et nous assure que la densité devient nulle (numériquement) lorsque l'on s'éloigne suffisamment de la surface. La densité associée à l'ellipse $\mathcal{E}$ (pour $d = 2$) avec $\sigma = 1$ est représentée Figure 5.2.3.

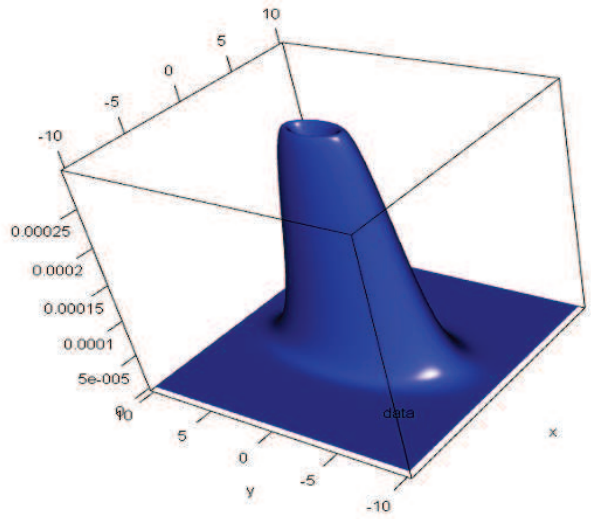


FIGURE 5.2.3 – Représentation de la densité ellipsoïdale associée à l'ellipse $\mathcal{E}$ ($d = 2$) et avec $\sigma = 1$.

Le calcul de l'expression exacte (5.2.5) de la constante de normalisation C_d a été détaillé en Annexe A. Les expressions exactes de cette constante pour les premières valeurs de d s'écrivent (indépendamment

de l'hypothèse sur les paramètres)

$$C_1^{-1} = \left(2\sigma\sqrt{2\pi} \mid \Sigma \mid^{1/2} [1 - \Phi(-1/\sigma)]\right)^{-1} \quad (5.2.7)$$

$$C_2^{-1} = \left((2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma [1 - \Phi(-1/\sigma)]\right)^{-1} \quad (5.2.8)$$

$$C_3^{-1} = \left(2(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma(1+\sigma^2) \left[1 - \Phi(-1/\sigma) + \frac{\sigma \exp\left(-1/(2\sigma^2)\right)}{(1+\sigma^2)\sqrt{2\pi}}\right]\right)^{-1} \quad (5.2.9)$$

où Φ désigne la fonction de répartition de la loi Normale centrée réduite. En utilisant le fait que le paramètre σ est choisi suffisamment faible, ces expressions peuvent être raisonnablement approximées. Comme cela a été discuté en Annexe A, l'hypothèse sur les paramètres nous permet d'écrire

$$C_1^{-1} = \left(2\sigma\sqrt{2\pi} \mid \Sigma \mid^{1/2}\right)^{-1} \quad (5.2.10)$$

$$C_2^{-1} = \left((2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma\right)^{-1} \quad (5.2.11)$$

$$C_3^{-1} = \left(2(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma(1+\sigma^2)\right)^{-1}. \quad (5.2.12)$$

5.2.2 Propriétés

Nous allons maintenant nous focaliser sur les principales propriétés de la densité (5.2.4). Lorsque $d \geq 2$, la densité ellipsoïdale appartient à la famille des distributions elliptiques présentée au Chapitre 3 section 3.2. En effet, si on considère la fonction

$$g_\alpha(t) = \frac{\Gamma(d/2)}{2\pi^{d/2} J_{d-1}(\alpha)} \exp\left(-\frac{\left(\sqrt{t}-1\right)^2}{2\alpha^2}\right), \quad (5.2.13)$$

alors la densité f peut s'écrire sous la forme

$$f(x) = \mid \Sigma \mid^{-1/2} g_\sigma\left((x-\mu)^T \Sigma^{-1} (x-\mu)\right), \quad (5.2.14)$$

ce qui implique d'après la Définition 3.2.2 du Chapitre 3 que f est la densité d'une distribution elliptique. Par conséquent, d'après la Définition 3.2.4, si on pose $Y = \Sigma^{-1/2}(X-\mu)$, alors Y a pour densité $g_\sigma(y^T y)$ et peut s'écrire sous la forme $Y = W\,U$ où W est une variable aléatoire positive indépendante de U uniformément distribuée sur la sphère unité de $\mathbb{R}^d$. Le vecteur aléatoire X peut donc s'écrire d'après le Théorème 3.2.1

$$X = \mu + \Sigma^{1/2} W\,U \quad (5.2.15)$$

et la densité de W est de la forme

$$\varphi(t) = \frac{2\pi^{d/2}}{\Gamma(d/2)} t^{d-1} g_\sigma(t^2) = \frac{t^{d-1}}{J_{d-1}(\sigma)} \exp\left(-\frac{(t-1)^2}{2\sigma^2}\right) \mathbb{1}_{\{t\geq 0\}}. \quad (5.2.16)$$

Intuitivement, l'écriture (5.2.15) consiste à bruiter avec W la sphère unité U, à la déformer via $\Sigma^{1/2}$ et à la translater jusqu'en μ. Remarquons que les fonctions g et ϕ sont identiques à celles obtenues dans le cas de la densité sphérique. Ceci implique que les fluctuations autour de la surface sont de

même nature que dans le cas sphérique. La méthode de simulation de la densité sera quasi identique à celle déjà mise en œuvre puisque les variables aléatoires W et U sont définies de la même manière. Il suffira donc d'introduire le terme en $\Sigma^{1/2}$ pour donner la forme ellipsoïdale à la distribution conformément à (5.2.15).

Les résultats exposés ci-dessus permettent de calculer assez facilement les premiers moments de X. Les résultats du théorème suivant seront utile pour justifier les propriétés des estimateurs introduits en section 5.3.

Théorème 5.2.1. *Soit X un vecteur aléatoire de $\mathbb{R}^d$ (avec $d \geq 2$) et de densité de probabilité f introduite en (5.2.4). On pose pour tout $q \geq 1$, $J_q = J_q(\sigma)$. Alors,*

$$\mathbb{E}[X] = \mu \qquad \textit{et} \qquad \mathbb{V}\text{ar}[X] = \mathbb{E}\left[(X-\mu)(X-\mu)^T\right] = \frac{J_{d+1}}{d\,J_{d-1}}\,\Sigma = d^{-1}\Sigma_* \tag{5.2.17}$$

où l'on a posé $\Sigma_* = \dfrac{J_{d+1}}{J_{d-1}}\,\Sigma$. De plus,

$$\mathbb{V}\text{ar}\left(\sqrt{(X-\mu)^T\,\Sigma_*^{-1}\,(X-\mu)}\right) \;=\; \frac{J_{d+1}J_{d-1} - J_d^2}{J_{d-1}} \;=\; \widetilde{\sigma}^2. \tag{5.2.18}$$

Démonstration. La démonstration du théorème est immédiate en utilisant la décomposition $X = \mu + \Sigma^{1/2}WU$ où W est une variable aléatoire réelle indépendante de U uniformément distribuée sur la sphère unité de $\mathbb{R}^d$. De plus, on a $\|U\| = 1$, $\mathbb{E}[U] = 0$ et $\mathbb{V}\text{ar}[U] = d^{-1}\,I_d$ et on en déduit que $\mathbb{E}[X] = \mu$ et $\mathbb{V}\text{ar}[X] = d^{-1}\,\mathbb{E}\left[W^2\right]\,\Sigma = d^{-1}\,\Sigma_*$.
On conclut en rappelant que pour tout $q \geq 0$, $\mathbb{E}[W^q] = J_{d+q-1}J_{d-1}^{-1}$. Posons

$$\xi = \sqrt{(X-\mu)^T\Sigma_*^{-1}(X-\mu)}.$$

Alors $\xi = (\mathbb{E}\left[W^2\right])^{-1/2}W$ et on en déduit que

$$\mathbb{V}\text{ar}\left(\sqrt{(X-\mu)^T\,\Sigma_*^{-1}\,(X-\mu)}\right) \;=\; \mathbb{E}\left(\xi^2\right) - \left(\mathbb{E}[\xi]\right)^2 = \;1 - \frac{\left(\mathbb{E}[W]\right)^2}{\mathbb{E}\left[W^2\right]},$$

ce qui termine la preuve du théorème. □

Remarque 5.2.1. *Les résultats du théorème sont formulés pour toute dimension $d \geq 2$ et pour toutes valeurs de σ. Dans le cas particulier où $d = 3$ et σ suffisamment faible, on peut effectuer les approximations (voir Annexe A) suivantes : $J_2 = C\left(1+\sigma^2\right)$, $J_3 = C\left(1+3\sigma^2\right)$ et $J_4 = C\left(1+6\sigma^2+3\sigma^4\right)$ avec $C = \sigma\sqrt{2\pi}$.*

Par conséquent, on a $\Sigma_* = (1+6\sigma^2+3\sigma^4)\,(1+\sigma^2)^{-1}\,\Sigma$ et $\tilde{\sigma}^2 = (1+3\sigma^4)\,(1+7\sigma^2+9\sigma^4+3\sigma^6)^{-1}\,\sigma^2$. Finalement, comme σ est supposé suffisamment petit, on a $\Sigma_* \simeq \Sigma$ et $\tilde{\sigma}^2 \simeq \sigma^2$.

Théorème 5.2.2. *Soit X un vecteur aléatoire d-dimensionnel défini comme dans le Théorème 5.2.1. Si on pose*

$$X^* = X - \frac{(X-\mu)}{\sqrt{(X-\mu)^T\,\Sigma^{-1}\,(X-\mu)}}, \tag{5.2.19}$$

alors

$$\mathbb{E}[X^*] = \mu \qquad \textit{et} \qquad \mathbb{V}\text{ar}\,(X^*) = \frac{J_{d+1} - 2\,J_d + J_{d-1}}{d\,J_{d-1}}\,\Sigma. \tag{5.2.20}$$

Démonstration. La démonstration est basée sur la décomposition $X = \mu + \Sigma^{1/2} W U$. On peut alors écrire $X^* = \mu + \Sigma^{1/2} U\ (W - 1)$. Puisque les variables W et U sont indépendantes, et comme $\mathbb{E}[U] = 0$, on conclut que $\mathbb{E}[X^*] = \mu$.

Ce premier résultat permet de calculer la variance : $\mathbb{V}\text{ar}[X^*] = \mathbb{E}\left[(X^* - \mu)(X^* - \mu)^T\right]$. Étant donné que les variables W et U sont indépendantes et que $\mathbb{V}\text{ar}[U] = d^{-1} I_d$, on obtient $\mathbb{V}\text{ar}[X^*] = d^{-1}\,\mathbb{E}\left[(W-1)^2\right]\,\Sigma$. Le résultat du théorème vient ensuite en remarquant que $\mathbb{E}[W^q] = J_{d+q-1}\, J_{d-1}^{-1}$. □

Remarque 5.2.2. *La variable aléatoire X^* peut s'écrire plus généralement en introduisant un paramètre $\Lambda \in \mathbb{R}^{d\times d}$*

$$X^* = X - \frac{X - \mu}{\sqrt{(X-\mu)^T \Lambda^{-1} (X-\mu)}}.$$

Il est clair que $\mathbb{E}[X^*] = \mu$ n'est pas vérifié pour toute matrice Λ^{-1} puisque l'on a

$$\mathbb{E}[X^*] = \mu - \frac{\Sigma^{1/2}U}{\sqrt{U^T\Sigma^{1/2}\Lambda^{-1}\Sigma^{1/2}U}}.$$

Toutefois, si on pose $\Lambda^{-1} = c\,\Sigma^{-1}$ avec $c > 0$, alors on retrouve la propriété $\mathbb{E}[X^*] = \mu$. Cette remarque est vraie en particulier pour $\Sigma_*^{-1} = J_{d-1}\, J_{d+1}^{-1}\, \Sigma^{-1}$. Cette observation sera utile lors de l'étude du comportement des estimateurs des paramètres du modèle.

Enfin, on montre facilement que $\|\mathbb{V}\text{ar}[X^*]\| < \|\mathbb{V}\text{ar}[X]\|$ en remarquant que $J_d > J_{d-1}$.

Les résultat des Théorèmes 5.2.1 et 5.2.2 et des Remarques 5.2.1 et 5.2.2 seront particulièrement utiles lors de l'estimation des paramètres de la distribution à partir d'un échantillon.

5.2.3 Modèle de mélange

Nous introduisons maintenant le modèle de mélange que nous considérerons pour modéliser les nuages de points 3D. Les modèles de mélange ont été brièvement présentés au Chapitre 3 section 3.3.3. La densité de probabilité du mélange à K composantes considéré est de la forme

$$h(x \mid \Theta) = \sum_{k=1}^{K} \pi_k\, f(x \mid \theta_k) \tag{5.2.21}$$

où les π_k sont les proportions du mélange telles que $\pi_k \geq 0,\ 1 \leq k \leq K$ et $\sum_{k=1}^{K} \pi_k = 1$ et où $\Theta = (\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K)$ est le vecteur contenant tous les paramètres du modèle. Comme nous souhaitons modéliser un nuage de points 3D par des ellipsoïdes, la densité f décrivant chaque composante est choisie comme étant la nouvelle densité ellipsoïdale définie en (5.2.4) avec $d = 3$. Dans ce cas, le vecteur de paramètres caractérisant la composante k et contenant les paramètres de l'ellipsoïde s'écrit $\theta_k = \left(\mu_k, \Sigma_k, \sigma_k^2\right)$ et la densité est de la forme

$$f(x \mid \theta_k) = \frac{1}{2(2\pi)^{3/2}\ \mid \Sigma_k \mid^{1/2} \sigma_k(1+\sigma_k^2)} \exp\left(-\frac{1}{2\sigma_k^2}\left(\sqrt{(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)} - 1\right)^2\right). \tag{5.2.22}$$

5.3 Estimation des paramètres du modèle

Dans cette section, nous nous intéressons à l'estimation des paramètres inconnus du modèle de mélange introduit en section 5.2.3 dans le cas de la dimension $d = 3$. Le modèle à une seule composante est d'abord traité en section 5.3.1. L'estimation du modèle à plusieurs composantes est ensuite considéré en section 5.3.2.

5.3.1 Estimation des paramètres dans le cas du n-échantillon

Soit $X_1, \ldots, X_n$ des vecteurs aléatoires de $\mathbb{R}^3$ de loi de densité f introduite en (5.2.4) avec $d = 3$ et sous l'hypothèse que σ est faible par rapport à la longueur du plus petit demi-axe. On cherche à estimer les paramètres $\left(\mu, \Sigma, \sigma^2\right)$ caractérisant la densité f. Une première stratégie d'estimation est présentée en section 5.3.1.1 dont une amélioration sera ensuite proposée en section 5.3.1.2.

5.3.1.1 Estimation directe des paramètres

Pour estimer le centre μ, on considère la moyenne empirique du n-échantillon

$$\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i. \tag{5.3.1}$$

Puisque, d'après le Théorème 5.2.1, $\mathbb{E}\left[X_1\right] = \mu$, il est clair que $\overline{X}_n$ est un estimateur sans biais de μ. De plus comme $\mathbb{E}[\|X_1\|^2] < \infty$, on a grâce à la loi forte des grands nombres la convergence presque sûre de $\overline{X}_n$ vers μ. De plus en appliquant la loi du logarithme itéré de Hartman et Winters ([183]), on en déduit que

$$\| \overline{X}_n - \mu \| = O\left(a_n\right) \quad \text{p.s.} \quad \text{avec } a_n = \sqrt{\log\log n / n} \tag{5.3.2}$$

avec $a_n = \sqrt{(\log\log n)/n}$. Intéressons nous maintenant à l'estimation de la matrice de forme Σ. On considère pour cela la quantité

$$S_n = \frac{1}{n} \sum_{i=1}^{n} \left(X_i - \mu\right) \left(X_i - \mu\right)^T. \tag{5.3.3}$$

En utilisant les résultats du Théorème 5.2.1 et la loi forte des grands nombres, on montre facilement que S_n est un estimateur sans biais et convergent de $\dfrac{J_4(\sigma)}{3\, J_2(\sigma)} \Sigma$. Lorsque σ est suffisamment petit, la quantité $J_4\, J_2^{-1}$ est proche de 1 (voir Remarque 5.2.1) et dans ce cas un estimateur raisonnable de Σ est $\Sigma_n = 3\, S_n$. Cependant, μ étant inconnu, on l'estime par $\overline{X}_n$ et on propose finalement d'estimer Σ par

$$\widehat{\Sigma}_n \;=\; \frac{3}{n} \sum_{i=1}^{n} \left(X_i - \overline{X}_n\right) \left(X_i - \overline{X}_n\right)^T. \tag{5.3.4}$$

En utilisant la décomposition

$$\widehat{\Sigma}_n = \Sigma_n - 3 \left(\overline{X}_n - \mu\right) \left(\overline{X}_n - \mu\right)^T, \tag{5.3.5}$$

on montre facilement que $\widehat{\Sigma}_n$ converge presque sûrement vers $\Sigma^* = J_4(\sigma)\, J_2^{-1}(\sigma)\, \Sigma$ et par conséquent lorsque σ est petit, $\widehat{\Sigma}_n$ est un estimateur raisonnable de Σ. De plus comme $\mathbb{E}[\|X_1\|^4] < \infty$, la

loi du logarithme itéré de Hartman et Winters ([183]) permet d'obtenir la majoration

$$\left\|\widehat{\Sigma}_n - \Sigma^*\right\| = O\left(a_n\right) \quad \text{p.s.} \tag{5.3.6}$$

Intéressons nous enfin à l'estimation du paramètre σ^2. On considère pour cela la quantité

$$\sigma_n^2 = \frac{1}{n}\sum_{j=1}^{n}\left(\xi_j - \mathbb{E}\left[\xi_j\right]\right)^2 \tag{5.3.7}$$

où pour $j = 1, \ldots, n$, $\xi_j = \sqrt{\left(X_j - \mu\right)^T \left(\Sigma^*\right)^{-1}\left(X_j - \mu\right)}$. En utilisant la loi forte des grands nombres et les résultats du Théorème 5.2.1, on montre que σ_n^2 converge presque sûrement vers $\widetilde{\sigma}^2 \simeq \sigma^2$ lorsque σ est petit, ce qui sera notre cadre d'application. Bien évidemment μ et Σ^* sont inconnus. On propose donc d'estimer σ^2 par

$$\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{j=1}^{n}\widehat{\xi}_j^2 - \left(\frac{1}{n}\sum_{j=1}^{n}\widehat{\xi}_j\right)^2 \tag{5.3.8}$$

où pour $j = 1, \ldots, n$, $\widehat{\xi}_j = \sqrt{\left(X_j - \overline{X}_n\right)^T \widehat{\Sigma}_n^{-1}\left(X_j - \overline{X}_n\right)}$.
On montre dans l'annexe C que cet estimateur converge vers $\widetilde{\sigma}^2$. Nous verrons dans le paragraphe suivant comment estimer de manière itérative la matrice $\left(\Sigma^*\right)^{-1}$ sans avoir à inverser la matrice $\widehat{\Sigma}_n$.

5.3.1.2 Estimation par un algorithme de type Backfitting

Nous proposons maintenant une stratégie plus performante d'estimation des différents paramètres. Nous avions proposé d'estimer le centre μ par la moyenne empirique des observations notée $\overline{X}_n$. Toutefois, cet estimateur a une variance plus élevée qu'un autre estimateur sans biais de μ noté $\overline{X}_n^*$ et défini par

$$\overline{X}_n^* = \overline{X}_n - \frac{1}{n}\sum_{i=1}^{n}\frac{\left(X_i - \mu\right)}{\sqrt{\left(X_i - \mu\right)^T \Sigma^{-1}\left(X_i - \mu\right)}}. \tag{5.3.9}$$

En effet, en utilisant les résultats du théorème 5.2.2, on montre facilement que $\overline{X}_n^*$ est un estimateur sans biais et convergent de μ. De plus, d'après la Remarque 5.2.2, on a $\|\mathbb{V}\text{ar}(\overline{X}_n^*)\| < \|\mathbb{V}\text{ar}(\overline{X}_n)\|$. Notons que puisque $\Sigma_* = J_4\, J_2^{-1}\,\Sigma$, on peut remplacer Σ par Σ_* dans (5.3.9) tout en gardant les mêmes propriétés pour $\overline{X}_n^*$.

Remarque 5.3.1. On peut remarquer que $\overline{X}_n^*$ peut se réécrire sous la forme

$$\overline{X}_n^* = \overline{X}_n - \frac{1}{n}\sum_{i=1}^{n} r(X_i)\,\frac{\left(X_i - \mu\right)}{\|X_i - \mu\|} \tag{5.3.10}$$

où $r(X_i)$ désigne le rayon de l'ellipsoïde dans la direction de X_i, c'est à dire la distance entre μ et la surface en direction de X_i. Cette écriture ressemble à celle de l'estimateur proposé dans le cas de la sphère à la différence près que l'on avait un rayon constant $r(X_i) = r$. Le rayon dépend ici du point X_i considéré et n'est pas constant.

L'estimateur $\overline{X}_n^*$ dépend bien évidemment des paramètres inconnus μ et Σ (ou Σ_*) qui doivent être estimés. Nous proposons alors la stratégie suivante pour améliorer l'estimation des paramètres μ, Σ et σ^2. On commence par estimer μ par $\widehat{\mu}_n^{(0)} = \overline{X}_n$, puis Σ_* par $\widehat{\Sigma}_n^{(0)} = \widehat{\Sigma}_n$. Puis, on améliore l'estimation de μ en considérant

$$\widehat{\mu}_n^{(1)} = \overline{X}_n - \frac{1}{n}\sum_{i=1}^{n} \frac{\left(X_i - \widehat{\mu}_n^{(0)}\right)}{\sqrt{\left(X_i - \widehat{\mu}_n^{(0)}\right)^T \left(\widehat{\Sigma}_n^{(0)}\right)^{-1} \left(X_i - \widehat{\mu}_n^{(0)}\right)}} \tag{5.3.11}$$

et celle de Σ_* par

$$\widehat{\Sigma}_n^{(1)} = \frac{3}{n}\sum_{i=1}^{n} \left(X_i - \widehat{\mu}_n^{(1)}\right)\left(X_i - \widehat{\mu}_n^{(1)}\right)^T. \tag{5.3.12}$$

La convergence de ces estimateurs n'est pour l'instant pas établie. Le processus (5.3.11)-(5.3.12) peut bien évidemment être réitéré jusqu'à convergence pour améliorer les estimations de μ et Σ_*. La mise à jour $\widehat{\mu}_n^{(0)} = \widehat{\mu}_n^{(1)}$ et $\widehat{\Sigma}_n^{(0)} = \widehat{\Sigma}_n^{(1)}$ est effectuée à la fin de chaque itération. La convergence théorique de cette stratégie itérative de backfitting n'est pas établie théoriquement mais elle a été vérifiée expérimentalement.

Le paramètre σ^2 peut ensuite être estimé par

$$\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{j=1}^{n} \widetilde{\xi}_j^2 - \left(\frac{1}{n}\sum_{j=1}^{n} \widetilde{\xi}_j\right)^2 \tag{5.3.13}$$

avec $\widetilde{\xi}_i = \sqrt{\left(X_i - \widehat{\mu}_n^{(1)}\right)^T \left(\widehat{\Sigma}_n^{(1)}\right)^{-1} \left(X_i - \widehat{\mu}_n^{(1)}\right)}$.

Nous avons proposé dans cette section une méthode d'estimation des paramètres (μ, Σ, σ) du modèle. En pratique, nous avons plutôt besoin d'estimer la matrice de forme inverse Σ^{-1}. L'inversion directe de l'estimation $\widehat{\Sigma}_n^{(1)}$ peut conduire à des instabilités numériques. Pour cette raison, nous appliquons la méthode décrite dans [74] pour estimer directement l'inverse de la matrice.

Théorème 5.3.1. *Soit $(\phi_n)_{n\geq 1}$ une suite de vecteurs de $\mathbb{R}^d$. Soit $S_n = C_0 + \ldots + C_n$ où $C_k = \phi_k\, \phi_k^T$. La matrice S_n^{-1} peut alors être calculée de manière itérative à l'aide des relations*

$$\left\{ \begin{array}{l} S_0^{-1} = I_d \\ S_k^{-1} = S_{k-1}^{-1} - \left(1 + \phi_k^T\, S_{k-1}^{-1}\, \phi_k\right)^{-1}\, S_{k-1}^{-1}\phi_k\phi_k^T S_{k-1}^{-1} \text{ pour } k = 1, \ldots, n \end{array} \right. .$$

Ce résultat est utilisé pour calculer l'inverse de la matrice $\widehat{\Sigma}_n$:

$$\left\{ \begin{array}{l} \widehat{\Sigma}_{n,0}^{-1} = I_d \\ \widehat{\Sigma}_{n,k}^{-1} = \widehat{\Sigma}_{n,k-1}^{-1} - \left(1 + \phi_k^T\, \widehat{\Sigma}_{n,k-1}^{-1}\, \phi_k\right)^{-1}\, \widehat{\Sigma}_{n,k-1}^{-1}\phi_k\phi_k^T S_{n-1}^{-1} \text{ pour } k = 1, \ldots, n \end{array} \right. \tag{5.3.14}$$

avec $\phi_k = \left(X_k - \widehat{\mu}_n^{(1)}\right)$.

5.3.2 Estimation du modèle de mélange

Cette section traite du cas d'un modèle de mélange comportant $K > 1$ composantes. Nous choisissons d'utiliser la méthode du maximum de vraisemblance. L'algorithme Espérance-Maximisation (EM) présenté au Chapitre 3 section 3.4.2.2 offre une méthode d'estimation efficace des différents paramètres inconnus. Rappelons que l'algorithme consiste en deux étapes E (Espérance) et M (Maximisation) itérées jusqu'à convergence. L'étape E consiste à calculer pour chaque observation X_i la probabilité $t_{i\ell} \in [0,1]$ qu'elle ai été générée par la composante ℓ du mélange. L'étape M maximise la log-vraisemblance étant donnés les paramètres courants du modèle (à l'itération j) et les probabilités d'appartenance. Les expressions des estimateurs proviennent de la maximisation de la log-vraisemblance complétée (Annexe B). Un vecteur de paramètres $\Theta^{(0)}$ est nécessaire pour initialiser le processus. Nous choisissons pour cela d'appliquer un algorithme K-means ([28]). La classification obtenue est utilisée pour estimer les paramètres de chacune des composantes. La méthode d'estimation se résume donc aux étapes suivantes

– **Étape E :** Estimer les probabilités d'appartenance $t_{i\ell}$ des observations $(X_i)_{i=1}^n$ aux classes $(C_l)_{l=1}^K$ à partir de l'estimation courante $\Theta^{(j)} = \left(\pi_1^{(j)}, \ldots, \pi_K^{(j)}, \theta_1^{(j)}, \ldots, \theta_K^{(j)}\right)$

$$t_{i\ell} = \frac{\pi_\ell^{(j)}\, f\left(X_i \mid \theta_\ell^{(j)}\right)}{\sum_{k=1}^K \pi_k^{(j)}\, f\left(X_i \mid \theta_k^{(j)}\right)}. \tag{5.3.15}$$

– **Étape M :** Déterminer le nouveau vecteur de paramètres $\Theta^{(j+1)}$ maximisant la log-vraisemblance (pour chaque composante ℓ)

$$\begin{cases} \widehat{\mu}_\ell = \dfrac{1}{\sum_{i=1}^n t_{i\ell}} \left[\displaystyle\sum_{i=1}^n t_{i\ell}\, X_i - \sum_{i=1}^n \frac{t_{i\ell}\,(X_i - \widehat{\mu}_\ell)}{\xi_{i,\ell}}\right] \\ \widehat{\Sigma}_\ell = \dfrac{1}{\sum_{i=1}^n t_{i\ell}} \displaystyle\sum_{i=1}^n t_{i\ell} (X_i - \widehat{\mu}_\ell)(X_i - \widehat{\mu}_\ell)^T \\ \widehat{\sigma}_\ell^2 = \dfrac{1}{\sum_{i=1}^n t_{i\ell}} \displaystyle\sum_{i=1}^n t_{i\ell} \left(\xi_{i,\ell} - \overline{\xi}_\ell\right)^2 \\ \\ \widehat{\pi}_\ell = \dfrac{1}{n} \displaystyle\sum_{i=1}^n t_{i\ell} \end{cases} \tag{5.3.16}$$

avec

$$\xi_{i,\ell} = \sqrt{(X_i - \widehat{\mu}_\ell)^T\, \widehat{\Sigma}_\ell^{-1}\, (X_i - \widehat{\mu}_\ell)} \quad \text{et} \quad \overline{\xi}_\ell = \frac{1}{n} \sum_{i=1}^n \xi_{i,\ell}.$$

En pratique, l'inverse de la matrice de covariance Σ est directement estimée par le procédé décrit dans la section précédente. Les relations définissant les estimateurs de μ et Σ nécessitent de mettre en oeuvre le processus de backfitting décrit dans le cas d'une seule composante. L'algorithme EM et le processus de backfiting ont un critère d'arrêt similaire. Ils sont stoppés lorsque les centres des composantes n'évoluent plus suffisamment entre les itérations $j-1$ et j, c'est à dire si la condition suivante est satisfaite

$$\max_{k \in [1,K]} \left(\left\| \mu_k^{(j)} - \mu_k^{(j-1)} \right\| \right) < \epsilon_\mu \tag{5.3.17}$$

où ϵ_μ est un seuil fixé à ϵ_μ^{EM} ou ϵ_μ^{BF}. Le second critère d'arrêt est un nombre maximal d'itérations effectuées fixé à N_{max}^{EM} ou N_{max}^{BF}. Le comportement de l'algorithme d'estimation sera étudié lors des expérimentations sur des données simulées.

5.4 Expérimentations sur des données simulées

Le but des expérimentations réalisées dans cette section est de montrer que la méthode d'estimation proposée se comporte comme attendu sur des données simulées. On considère un vecteur aléatoire X de dimension $d = 3$ et de densité de probabilité ellipsoïdale f introduite en (5.2.1). Les paramètres de la distribution sont choisis de manière à ce que l'hypothèse sur le paramètre σ soit vérifiée et que le nombre d'observations n soit suffisamment élevé. Les données ont été générées par la méthode de simulation décrite en annexe D. La qualité des résultats sera quantifiée dans le cas d'un modèle à une seule composante en section 5.4.1 et d'un modèle à plusieurs composantes en section 5.4.2.

5.4.1 Résultats dans le cas d'un modèle à une seule composante

La première série d'expérimentations concerne le modèle à une seule composante ($K = 1$). Un ensemble de $N = 200$ échantillons indépendants de réalisations de X pour différentes valeurs de n est simulé. Les paramètres sont fixés à $\mu = (0, 0, 0)^T$, $\Sigma = \begin{pmatrix} 100^2 & 0 & 0 \\ 0 & 50^2 & 0 \\ 0 & 0 & 50^2 \end{pmatrix}$ et $\sigma = 0.01$. Ces valeurs ont été choisies pour se rapprocher des conditions de l'application réelle.

Les résultats obtenus avec la méthode proposée (BF) sont présentés Figures 5.4.1 et 5.4.2. Il est clair que la qualité des estimations s'améliore et que la variabilité diminue lorsque la taille de l'échantillon augmente. Les estimations sont raisonnables pour un nombre d'observations assez grand.

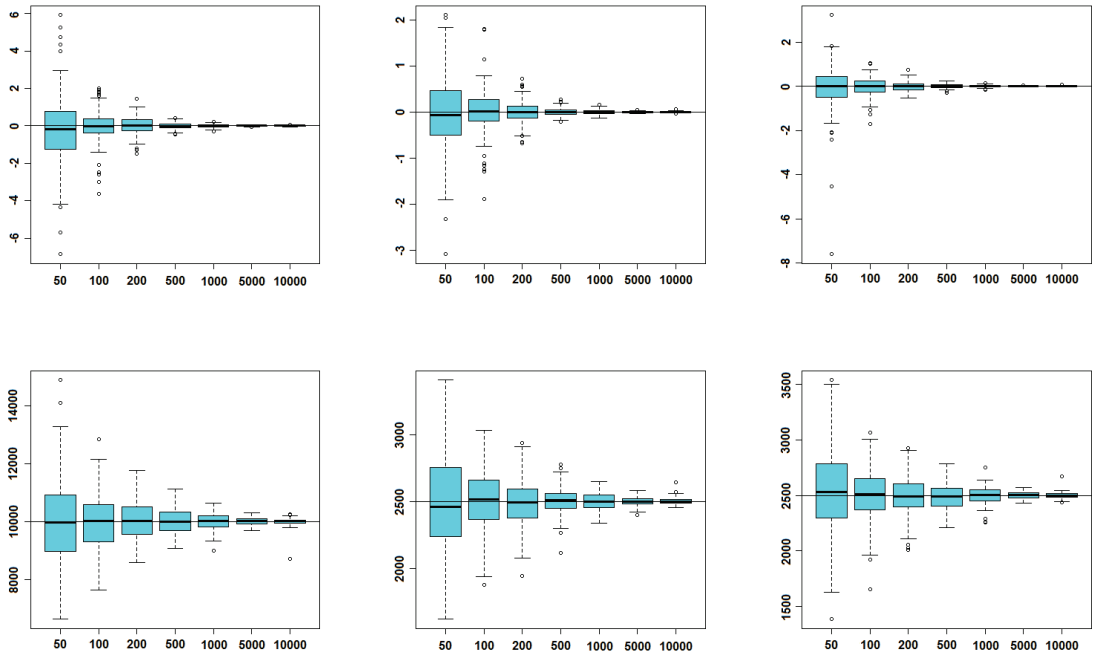


FIGURE 5.4.1 – Sur la première ligne, boîtes à moustaches des estimations de μ_x, μ_y et μ_z. Sur la seconde ligne, estimations de Σ_{11}, Σ_{22} et Σ_{33}.

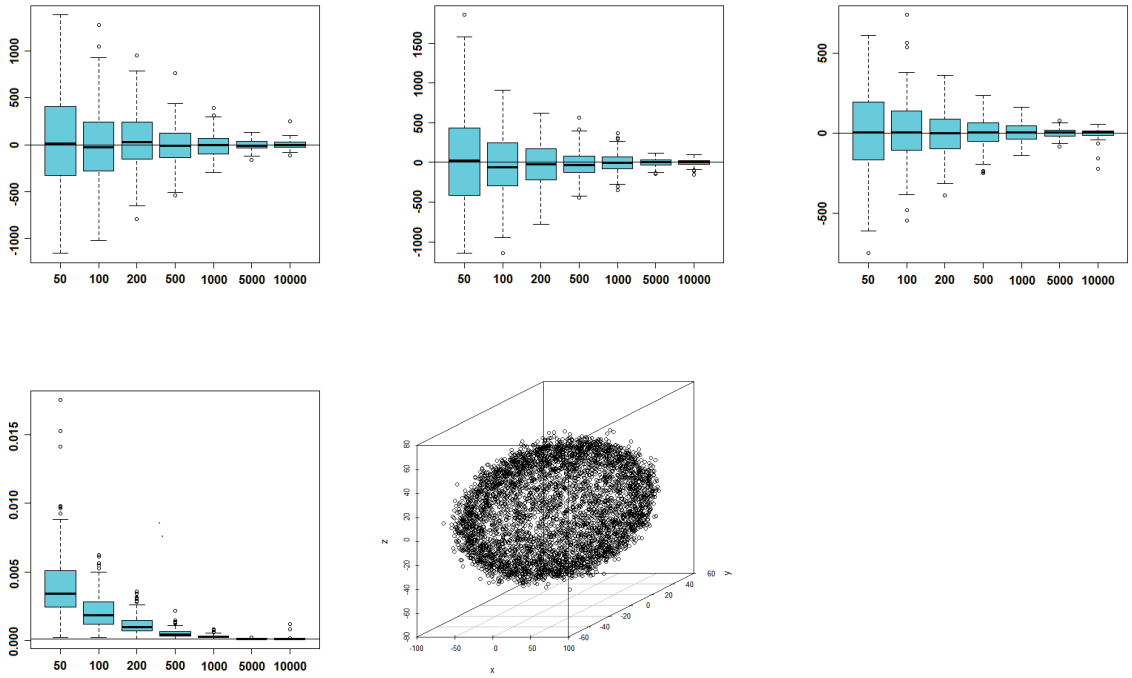


FIGURE 5.4.2 – Sur la première ligne, estimations de Σ_{12}, Σ_{13} et Σ_{23}. Sur la seconde ligne, estimation de σ et un échantillon de taille $n = 1000$.

Nous avons mentionné en section 5.3.1 que l'estimateur proposé présentait une variance inférieure à celle de la moyenne empirique. Ces deux estimateurs sont comparés sur le même jeu de données. Les résultats sont présentés Figure 5.4.3. Les estimations du centre obtenues avec la méthode BF présentent une variabilité beaucoup plus faible que celle obtenues avec la moyenne empirique.

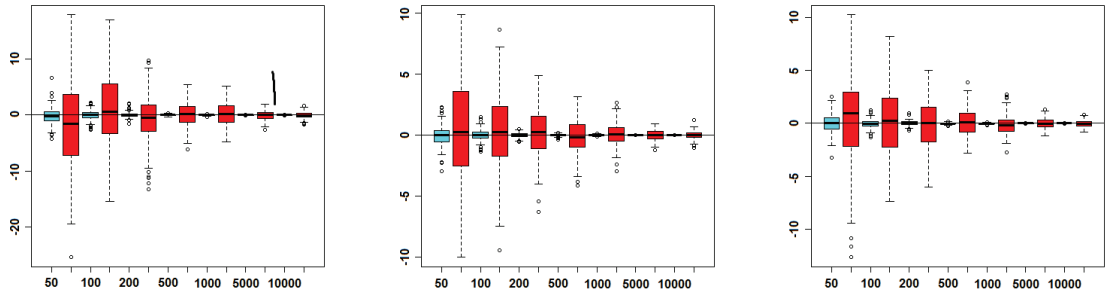


FIGURE 5.4.3 – Boites à moustaches respectives des estimations de μ_x, μ_y et μ_z selon le nombre d'observations n. La méthode BF est représentée en cyan et la moyenne empirique en rouge.

Étant donné que les nuages de points provenant du capteur 3D ne sont répartis que sur une face des objets, nous nous intéressons au cas du demi-ellipsoïde. Nous simulons des observations réparties sur une moitié de surface ellipsoïdale. Plus précisément, seuls les points dont la composante y est positive sont conservés. Les résultats obtenus avec l'algorithme BF sont donnés Figure 5.4.4. Contrairement à ce que nous avions pu observer avec le modèle sphérique, l'ellipsoïde estimé ne modélise pas correctement les données. La méthode d'estimation n'est pas robuste à la répartition non uniforme des observations sur la surface. Elle ne peut donc pas être utilisée pour modéliser un nuage de points provenant d'un seul capteur 3D. Ceci peut s'expliquer en partie grâce à la Remarque 5.2.2 dans laquelle il est précisé que la quantité

$$X^\star = X - \frac{X-\mu}{\sqrt{(X-\mu)^T \Lambda^{-1} (X-\mu)}}$$

avec $\Lambda \in \mathbb{R}^{d\times d}$ est d'espérance μ si $\Lambda = c\,\Sigma$ avec $c > 0$, mais pas dans le cas général. Par conséquent, une mauvaise estimation de la matrice Σ (non égale à une constante multiplicative près) impliquera un biais important sur l'estimateur du centre. L'erreur d'estimation est ensuite propagée durant les itérations du backfitting. Notons enfin que lorsque l'un des deux paramètres μ ou Σ est fixé à la bonne valeur, on observe que les autres paramètres sont très bien estimés. On peut donc penser que c'est la combinaison des différent estimateurs ou l'initialisation du processus itératif qui doivent être mis en cause.

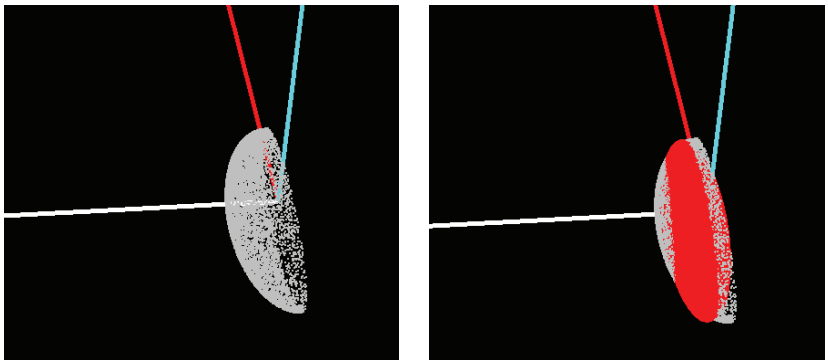


FIGURE 5.4.4 – Échantillon de taille $n = 10000$ d'un demi-ellipsoïde et modèle estimé.

Enfin, la dernière expérimentation de cette section s'intéresse à la modélisation de points répartis autour d'une sphère par une forme ellipsoïdale. En effet, on peut légitimement se demander si la matrice de forme estimée sera proche d'une matrice diagonale. Les paramètres de la densité sont fixés à $\mu = (0,0)^T$, $\Sigma = 50^2\, I_3$ et $\sigma = 0.01$. Les résultats sont présentés Figures 5.4.5 et 5.4.6. L'ellipsoïde estimé a bien une formé sphérique et modélise correctement les données. Les termes non diagonaux de la matrice de forme estimée sont proches de zéro pour un nombre suffisant d'observations. Une stratégie de test d'hypothèse pourrait être imaginée et mise en place pour décider si l'ellipsoïde peut être considéré ou non comme étant une sphère, avec un risque d'erreur fixé. Les bons résultats obtenus sur ces données peuvent s'expliquer par le fait que les estimations initiales calculées par l'algorithme K-means sont proches des valeurs réelles des paramètres.

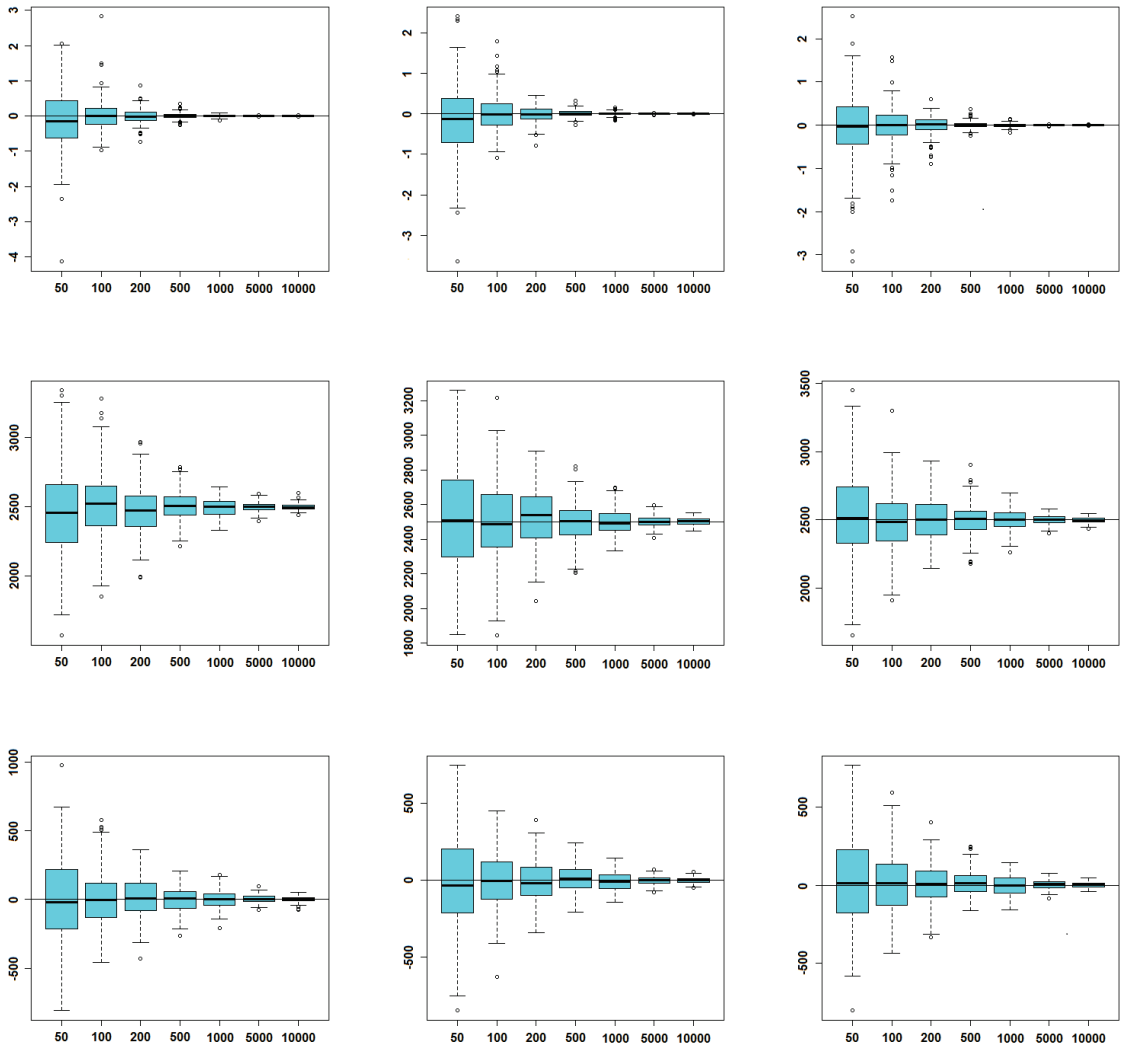


FIGURE 5.4.5 – Sur la première ligne, boîtes à moustaches des estimations de μ_x, μ_y et μ_z. Sur la deuxième ligne, estimations de Σ_{11}, Σ_{22} et Σ_{33}. Sur la dernière ligne, estimations de Σ_{12}, Σ_{13} et Σ_{23}.

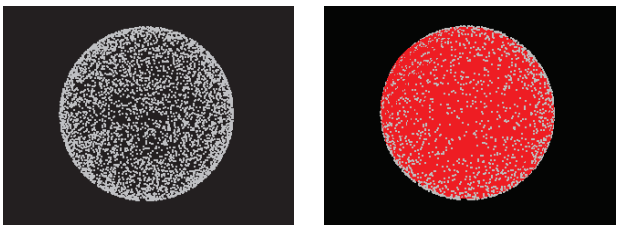


FIGURE 5.4.6 – A gauche, un échantillon de $n = 5000$ observations réparties sur une sphère et à droite le modèle estimé.

5.4.2 Résultats dans le cas d'un modèle à plusieurs composantes

La seconde série d'expérimentations vise à tester la méthode sur plusieurs ellipsoïdes se chevauchant. L'objectif est de montrer que l'algorithme EM permet de correctement séparer et modéliser plusieurs classes. Dans la suite, l'algorithme EM est initialisé par un vecteur de paramètres $\Theta^{(0)}$ obtenu après une classification des données par un algorithme K-means.

On considère un modèle de mélange composé de trois ellipsoïdes E_1, E_2 et E_3 s'intersectant et de même poids. On simule un échantillon de $n = 3000$ observations. Les paramètres μ, σ et π du modèle sont donnés Table 5.4.1. Les matrices de forme sont identiques pour les trois composantes et fixées à $\Sigma = \begin{pmatrix} 1000^2 & 0 & 0 \\ 0 & 500^2 & 0 \\ 0 & 0 & 500^2 \end{pmatrix}$.

TABLE 5.4.1 – Valeurs exactes des paramètres μ, σ et π.

	μ_x	μ_y	μ_z	σ	π
E1	0	0	0	0.01	1/3
E2	1800	0	0	0.01	1/3
E3	3600	0	0	0.01	1/3

Les résultats obtenus sont présentés Figures 5.4.7, 5.4.8, 5.4.9, 5.4.10 et 5.4.11. On constate que la variabilité des estimations diminue lorsque la taille de l'échantillon augmente. On remarque la présence d'un nombre important de mauvaises estimations lorsque le nombre d'observations est trop faible. Les poids du mélange sont correctement estimés. Les termes diagonaux de la matrice de forme sont moins bien estimés que dans le cas d'une seule composante, particulièrement pour la composante située au milieu. La méthode d'estimation semble moins performante sur ces données où les ellipsoïdes s'intersectent. L'estimation de la matrice de forme est un problème difficile et pourrait être amélioré en utilisant un estimateur plus robuste. On peut également supposer que l'initialisation des classes par un autre algorithme que K-means pourrait améliorer les résultats.

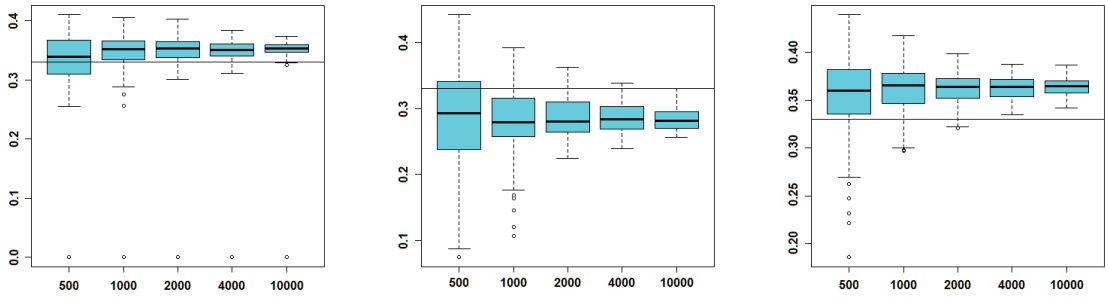


FIGURE 5.4.7 – Estimations des poids π_1, π_2 et π_3 associés aux ellipsoïdes E_1, E_2 et E_3.

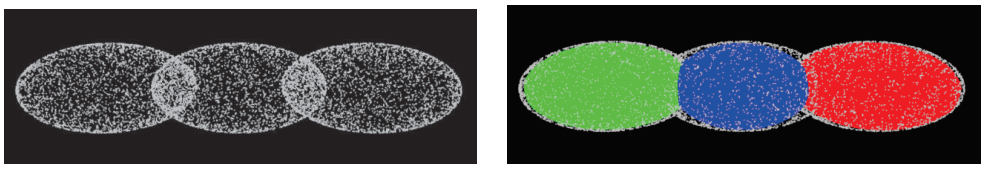


FIGURE 5.4.8 – Un échantillon de taille $n = 10000$ et le modèle estimé.

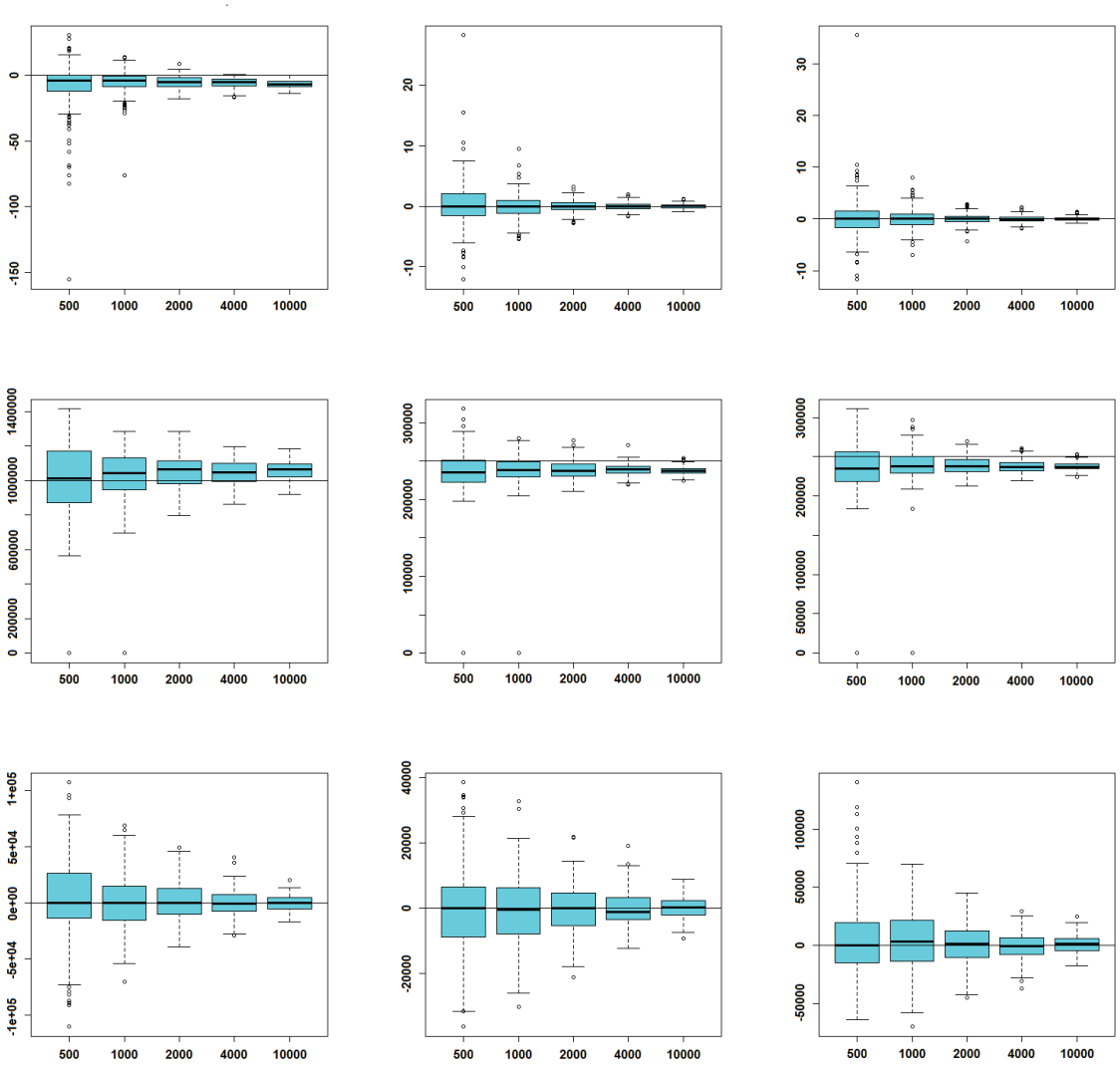


FIGURE 5.4.9 – Estimation des paramètres de E_1. Sur la première ligne, boîtes à moustaches des estimations de μ_x, μ_y et μ_z. Sur la deuxième ligne, estimations de Σ_{11}, Σ_{22} et Σ_{33}. Sur la troisième ligne, estimations de Σ_{12}, Σ_{13} et Σ_{23}.

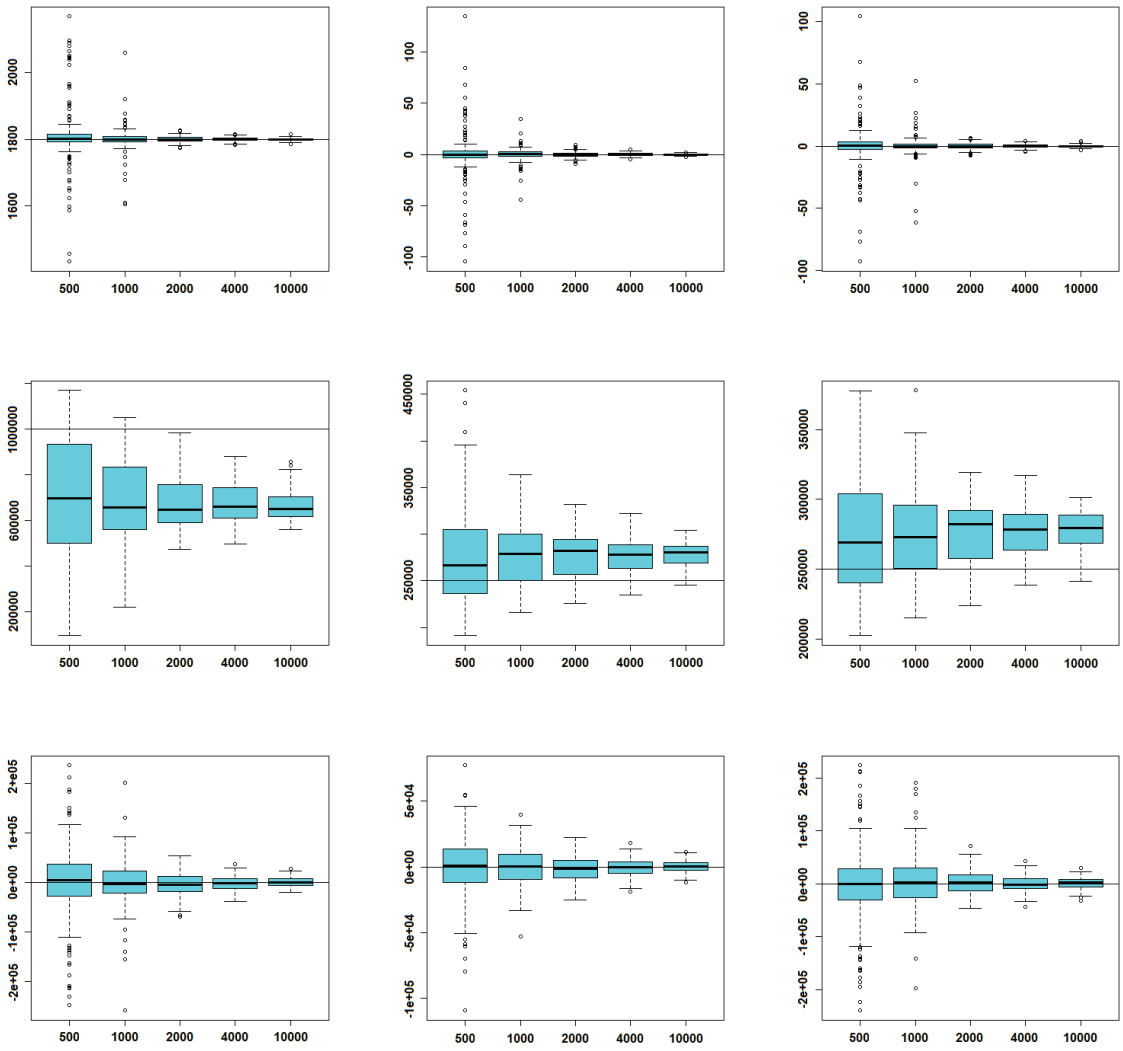


FIGURE 5.4.10 – Estimation des paramètres de E_2. Sur la première ligne, boîtes à moustaches des estimations de μ_x, μ_y et μ_z. Sur la deuxième ligne, estimations de Σ_{11}, Σ_{22} et Σ_{33}. Sur la troisième ligne, estimations de Σ_{12}, Σ_{13} et Σ_{23}.

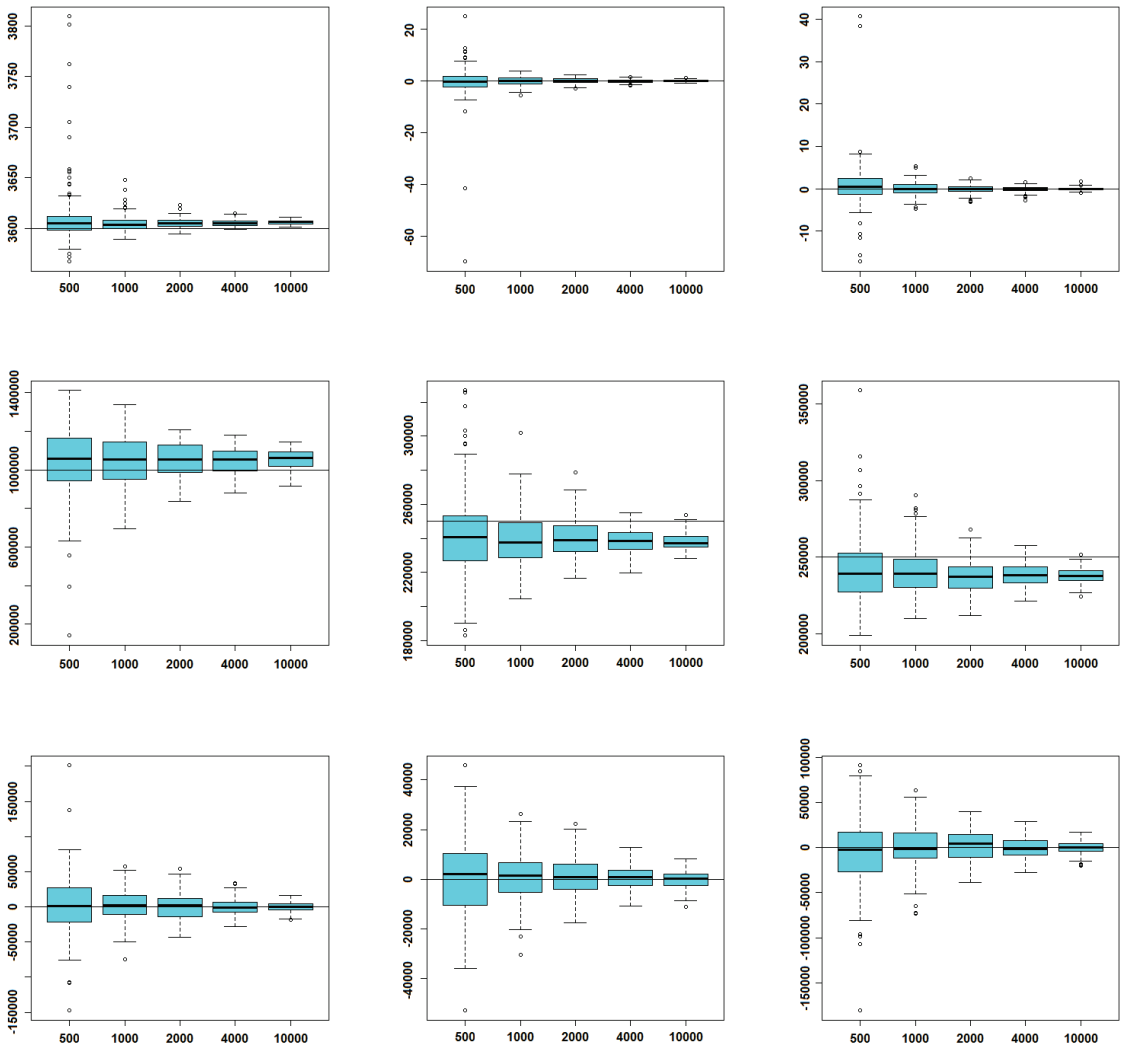


FIGURE 5.4.11 – Estimation des paramètres de E_3. Sur la première ligne, boîtes à moustaches des estimations de μ_x, μ_y et μ_z. Sur la deuxième ligne, estimations de Σ_{11}, Σ_{22} et Σ_{33}. Sur la troisième ligne, estimations de Σ_{12}, Σ_{13} et Σ_{23}.

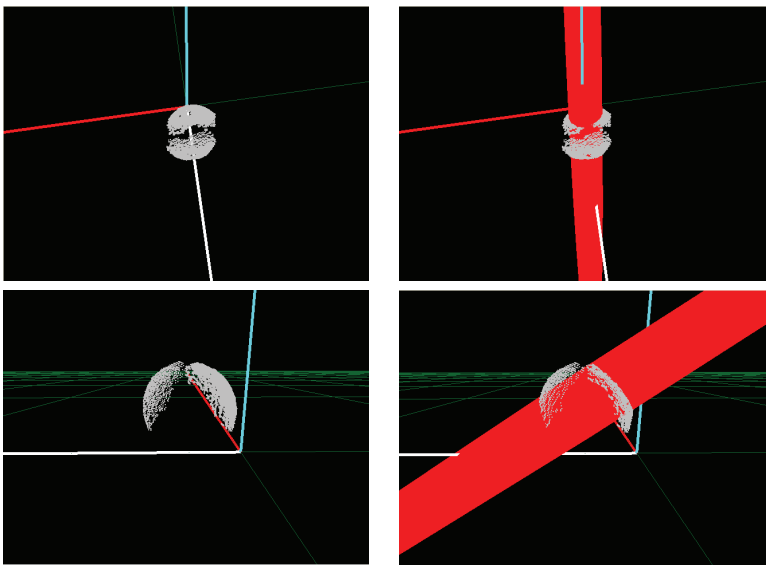
5.5 Expérimentations sur des données réelles

Nous avons vu dans la section précédente que la méthode d'estimation proposée ne fonctionnait pas bien sur des nuages de points situés sur une seule face des objets. Par conséquent, la méthode est expérimentée sur des données provenant de deux capteurs 3D. Les nuages de points 3D ont été calibrés manuellement l'un par rapport à l'autre. On constate toutefois que les capteurs ne sont pas parfaitement synchronisés entre eux et que les images n'ont pas été acquises au même instant. L'utilisation de deux capteurs implique que les objets ne sont plus vus de dessus mais depuis deux points de vue opposés. La quantité de données manquantes est alors beaucoup plus faible et des observations sont situées de chaque coté des objets.

Nous rappelons que ces expérimentations sont à but prospectif pour ouvrir des pistes de recherche futures. Le travail réalisé dans ce chapitre mériterait une étude supplémentaire plus approfondie afin d'améliorer les résultats obtenus.

Le nombre de composantes du modèle de mélange est choisi manuellement. En effet, la stratégie de choix automatique élaborée dans le cas sphérique ne peut être directement appliquée dans le cas ellipsoïdal et nécessiterait une adaptation. Les diverses stratégies de réduction du temps de traitement n'ont pas été adaptées au cas ellipsoïdal.

L'algorithme d'estimation est tout d'abord testé sur un nuage de points 3D représentant une balle. Les résultats obtenus sont présentés Figure 5.5.1. Malgré la présence de points des deux côtés, l'objet n'est pas correctement modélisé. Certaines zones ne sont toujours pas visibles sur les deux images. La méthode d'estimation n'est pas assez robuste à la répartition non uniforme de points sur la surface et aux données manquantes.

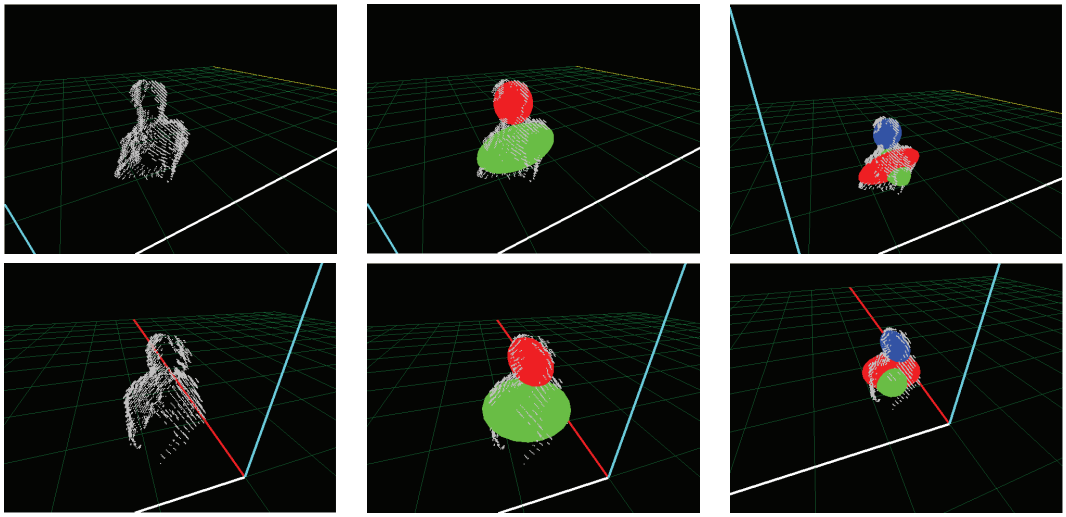


(a) (b)

FIGURE 5.5.1 – En (a) le nuage de points 3D provenant des deux capteurs et en (b) l'ellipsoïde estimé.

Malgré les mauvais résultats obtenus lors de l'expérimentation précédente, la méthode est appliquée sur un nuage de points 3D représentant une personne. Les résultats de la modélisation sont

présentés Figure 5.5.2. On note que les deux nuages de points ne sont pas parfaitement alignés. La modélisation du nuage de points 3D est plus cohérente. On constate toutefois que la composante modélisant la tête est ellipsoïdale et non sphérique. Un travail de recherche est encore nécessaire pour envisager l'utilisation du modèle de mélange ellipsoïdal dans le cadre de la détection de têtes. La robustesse de la méthode d'estimation aux données manquantes, des règles de détection adéquates et une accélération de la méthode seraient à étudier.



(a) (b) (c)

FIGURE 5.5.2 – En (a) le nuage de points 3D, en (b) le modèle ellipsoïdal à $K = 2$ composantes estimé et en (c) celui à $K = 3$ composantes, depuis deux points de vue différents.

5.6 Conclusion

Dans ce chapitre, nous avons étendu le modèle de mélange sphérique au cas ellipsoïdal. Le modèle de mélange ellipsoïdal est basé sur une nouvelle densité de probabilité modélisant des points répartis autour d'un ellipsoïde. La densité n'est pas basée sur la distance Euclidienne classique mais sur la distance de Mahalanobis. Pour estimer les paramètres du modèle de mélange, un algorithme EM comprenant un processus de backfitting est utilisé.

L'efficacité de la méthode d'estimation proposée a été étudiée en simulation. Elle n'est pas robuste à la répartition non uniforme des observations sur la surface, contrairement à ce qui avait été constaté pour le modèle sphérique. Les expérimentations sur des données provenant de deux capteurs 3D ont montré qu'un travail de recherche était encore nécessaire pour pouvoir appliquer la méthode à la détection de têtes. L'estimation de la matrice de forme est un problème difficile et nous pensons que ce point devrait être amélioré pour obtenir de meilleurs résultats. Malgré le fait que des questions théoriques et pratiques mériteraient une étude plus approfondie, les premiers résultats obtenus sont prometteurs.

Ce chapitre nous a permis de montrer que le travail réalisé dans cette partie offrait des perspectives de recherche intéressantes.

Annexe

A Calcul de la constante de normalisation de la densité

Cette annexe détaille le calcul de la constante de normalisation C_d de la densité de probabilité f introduite en (5.2.4). Nous commençons tout d'abord par établir le lemme technique suivant utile pour simplifier les calculs.

Lemme A.1. *Soit pour $q \in \mathbb{N}$ et $\alpha > 0$,*

$$J_q(\alpha) \;=\; \int_0^\infty t^q \exp\left(-(t-1)^2/\left(2\alpha^2\right)\right) \mathrm{d}t. \tag{A1}$$

Alors $J_0(\alpha) = \alpha\sqrt{2\pi}\,(1-\Phi(-1/\alpha))$, $J_1(\alpha) = J_0(\alpha) + \alpha^2 \exp\left(-1/\left(2\alpha^2\right)\right)$, et pour $q \geq 2$,

$$J_q(\alpha) \;=\; J_{q-1}(\alpha) + (q-1)\alpha^2 J_{q-2}(\alpha) \tag{A2}$$

où Φ désigne la fonction de répartition de la loi Normale centrée réduite.

Démonstration. Ce lemme a déjà été prouvé au Chapitre 3 Annexe A. □

Considérons la constante de normalisation C_d satisfaisant l'équation

$$I \;=\; C_d \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\sigma^2}\left(\sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} - 1\right)^2\right) dx = 1. \tag{A3}$$

Le changement de variable $x := x - \mu$ permet d'écrire

$$C_d \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\sigma^2}\left(\sqrt{x^T\Sigma^{-1}x} - 1\right)^2\right) dx = 1. \tag{A4}$$

La matrice de forme $\Sigma \in \mathbb{R}^{d\times d}$ est une matrice symétrique inversible à coefficients réels. Par conséquent, son inverse Σ^{-1} est également symétrique inversible donc diagonalisable à valeurs propres strictement positives. La matrice Σ^{-1} peut donc s'écrire sous la forme

$$\Sigma^{-1} = P\,D\,P^{-1} \tag{A5}$$

où P est une matrice de passage (orthogonale) dont les colonnes sont les vecteurs propres de Σ^{-1} et $D = diag\,(\lambda_1, \ldots, \lambda_d)$ une matrice diagonale contenant les valeurs propres de Σ^{-1}. Par conséquent, si on effectue le changement de variable

$$y = P^{-1}x \tag{A6}$$

dont la valeur absolue du Jacobien vaut

$$\mid J \mid = \mid P^{-1} \mid = 1, \tag{A7}$$

alors la relation (A4) devient

$$I \;=\; C_d \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\sigma^2}\left(\sqrt{y^T D y} - 1\right)^2\right) dy \tag{A8}$$

$$\;=\; C_d \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\sigma^2}\left(\sqrt{\sum_{i=1}^d \lambda_i y_i^2} - 1\right)^2\right) dy. \tag{A9}$$

On considère le changement de variable $u_i = \sqrt{\lambda_i}\, y_i$. Son jacobien vaut

$$\begin{aligned} \mid J \mid &= \mid diag\left(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_d}\right) \mid & \text{(A10)} \\ &= \sqrt{\lambda_1 \ldots \lambda_d} & \text{(A11)} \\ &= \mid \Sigma \mid^{1/2}, & \text{(A12)} \end{aligned}$$

ce qui implique que l'expression (A9) devient

$$I = C_d \mid \Sigma \mid^{1/2} \int_{\mathbb{R}^d} \exp\left(-\frac{1}{2\sigma^2}\left(\|u\| - 1\right)^2\right) du. \tag{A13}$$

Enfin, un passage en coordonnées polaires donne

$$\begin{aligned} I &= C_d \frac{2\pi^{d/2}}{\Gamma\left(d/2\right)} \mid \Sigma \mid^{1/2} \int_0^{+\infty} \rho^{d-1} \exp\left(-\frac{(\rho-1)^2}{2\sigma^2}\right) d\rho & \text{(A14)} \\ &= C_d \frac{2\pi^{d/2}}{\Gamma\left(d/2\right)} \mid \Sigma \mid^{1/2} J_{d-1}\left(\sigma\right) = 1. & \text{(A15)} \end{aligned}$$

Finalement, la constante de normalisation a pour expression

$$C_d = \frac{\Gamma\left(d/2\right)}{2\pi^{d/2} \mid \Sigma \mid^{1/2} J_{d-1}\left(\sigma\right)}. \tag{A16}$$

Les expressions exactes de cette constante pour les premières valeurs de d s'écrivent

$$\begin{aligned} C_1 &= \frac{1}{2\sigma\sqrt{2\pi} \mid \Sigma \mid^{1/2} \left[1 - \Phi\left(-1/\sigma\right)\right]} & \text{(A17)} \\ C_2 &= \frac{1}{(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma \left[1 - \Phi\left(-1/\sigma\right)\right]} & \text{(A18)} \\ C_3 &= \frac{1}{2(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma(1+\sigma^2)\left[1 - \Phi\left(-1/\sigma\right) + \dfrac{\sigma \exp\left(-1/(2\sigma^2)\right)}{(1+\sigma^2)\sqrt{2\pi}}\right]} & \text{(A19)} \end{aligned}$$

où Φ désigne la fonction de répartition de la loi Normale centrée réduite.

En utilisant l'hypothèse que le paramètre σ est suffisamment faible, ces expressions peuvent être raisonnablement approximées par

$$\begin{aligned} C_1 &= \frac{1}{2\sigma\sqrt{2\pi} \mid \Sigma \mid^{1/2}} & \text{(A20)} \\ C_2 &= \frac{1}{(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma} & \text{(A21)} \\ C_3 &= \frac{1}{2(2\pi)^{3/2} \mid \Sigma \mid^{1/2} \sigma(1+\sigma^2)}. & \text{(A22)} \end{aligned}$$

La Figure A1 représente l'évolution des termes $\Phi(-1/\sigma)$ et $\frac{t}{\sqrt{2\pi}(1+t^2)} e^{(-1/(2\sigma^2))}$ et fonction de σ. On constate que ces quantités sont négligeables pour des valeurs faibles de σ.

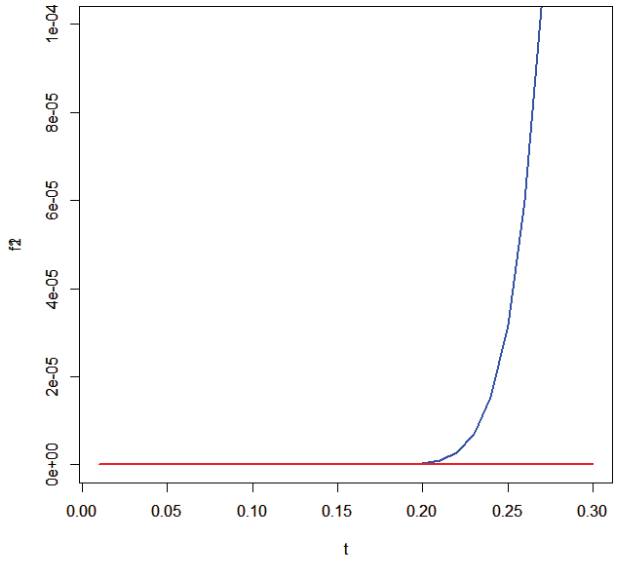


FIGURE A1 – Évolution des termes $\Phi(-1/\sigma)$ (bleu) et $\frac{t}{\sqrt{2\pi}(1+t^2)} e^{(-1/(2\sigma^2))}$ (rouge) et fonction de σ.

B Maximisation de la log-vraisemblance complétée

Dans cette Annexe, nous justifions le choix des estimateurs utilisés dans l'étape M de l'algorithme EM décrit en section 5.3.2. Pour maximiser la fonction Q, on calcule les dérivées partielles par rapport aux paramètres μ_ℓ, Σ_ℓ et σ_ℓ de la quantité

$$Q(\Theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} t_{ik} \left(\log(\pi_k) + \log(f(x_i, \theta_k)) \right) \tag{B1}$$

avec $f(x \mid \theta_k) = \dfrac{1}{2(2\pi)^{3/2} \mid \Sigma_k \mid^{1/2} \sigma_k(1+\sigma_k^2)} \exp\left(-\dfrac{1}{2\sigma_k^2} \left(\sqrt{(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)} - 1 \right)^2 \right)$
et σ suffisamment faible.

La dérivée partielle de Q par rapport à μ_ℓ vaut

$$\frac{\partial Q}{\partial \mu_\ell} = \frac{\Sigma^{-1}}{\sigma^2} \sum_{i=1}^{n} (x_i - \mu_\ell) \left[1 - \frac{1}{\sqrt{(x_i-\mu_\ell)^T \Sigma_\ell^{-1} (x_i-\mu_\ell)}} \right]. \tag{B2}$$

La dérivée partielle de Q par rapport à Σ_ℓ vaut

$$\frac{\partial Q}{\partial \Sigma_\ell} = -\frac{n}{2} \Sigma_\ell^{-1} - \frac{}{\sigma^2} \sum_{i=1}^{n} (x_i - \mu_\ell)(x_i - \mu_\ell)^T \left[\sqrt{(x_i-\mu_\ell)^T \Sigma_\ell^{-1} (x_i-\mu_\ell)} - 1 \right]. \tag{B3}$$

La dérivée partielle de Q par rapport à σ_ℓ^2 vaut

$$\frac{\partial Q}{\partial \sigma_\ell} = -n\frac{(1+3\,\sigma_\ell^2)}{\sigma_\ell(1+\sigma_\ell^2)} + \frac{1}{\sigma_\ell^3}\sum_{i=1}^{n}\left(\sqrt{(x-\mu_\ell)^T\,\Sigma_\ell^{-1}\,(x-\mu_\ell)}-1\right)^2. \tag{B4}$$

Il n'est pas possible de résoudre analytiquement le problème de maximisation. Avec l'hypothèse que σ est suffisamment faible, on a $(1+3\,\sigma^2)\,/\,(1+\sigma^2)\simeq 1$ et la dérivée partielle (B4) peut être simplifiée. On préfère d'autre part estimer la matrice de forme par l'estimateur classique $n^{-1}\,\sum_{i=1}^{n}(x_i-\mu_\ell)(x_i-\mu_\ell)^T$.

Ces approximations conduisent aux estimateurs présentés en section 5.3.2. Les poids du mélange sont estimés par les relations classiquement utilisées dans les mélanges gaussiens (voir Chapitre 3 Annexe D).

C Preuve de la convergence de σ_n^2

Cette annexe est consacrée à l'étude des propriétés de convergence de $\widehat{\sigma}_n^2$ annoncées en section 5.3.1.1. Pour cela, il nous faut d'abord établir la convergence de $\widehat{\Sigma}_n^{-1}$ vers Σ_*^{-1}. L'étude de la convergence de $\widehat{\Sigma}_n^{-1}$ est assez standard et nous reprenons ici les grandes lignes de la preuve proposée dans N'Guyen et Saracco [141]. D'après l'équation de Riccati pour l'inversion de matrice donnée par exemple dans Duflo ([74], page 96), nous pouvons écrire la décomposition suivante :

$$\widehat{\Sigma}_n^{-1} = \Sigma_*^{-1} - \Sigma_*^{-1}\left(\widehat{\Sigma}_n-\Sigma_*\right)\Sigma_*^{-1} + R_n \tag{C1}$$

où R_n est donné par

$$R_n = \Sigma_*^{-1}\left(\Sigma_*-\widehat{\Sigma}_n\right)\widehat{\Sigma}_n^{-1}\left(\Sigma_*-\widehat{\Sigma}_n\right)\Sigma_*^{-1}. \tag{C2}$$

On a immédiatement

$$\left\|\widehat{\Sigma}_n^{-1}-\Sigma_*^{-1}\right\| \leq \lambda_{\max}^2(\Sigma_*^{-1})\left\|\widehat{\Sigma}_n-\Sigma_*\right\| + \|R_n\| \tag{C3}$$

et

$$\|R_n\| \leq \lambda_{\max}\left(\widehat{\Sigma}_n^{-1}\right)\lambda_{\max}^2(\Sigma_*^{-1})\left\|\Sigma_*-\widehat{\Sigma}_n\right\|^2. \tag{C4}$$

Rappelons que pour toute matrice carrée A définie positive, on a $\lambda_{\max}(A^{-1})=\lambda_{\min}^{-1}(A)$. Par conséquent puisque $\lambda_{\min}(\Sigma)>0$ alors $\lambda_{\min}(\Sigma_*)>0$. De plus, puisque $\widehat{\Sigma}_n$ converge presque sûrement vers Σ_*, alors $\lambda_{\min}(\widehat{\Sigma}_n)>0$. On obtient alors

$$\|R_n\| = O\left(\left\|\Sigma_*-\widehat{\Sigma}_n\right\|^2\right) \quad \text{p.s.} \tag{C5}$$

En utilisant la majoration $\left\|\widehat{\Sigma}_n-\Sigma_*\right\| = O(a_n)$ obtenue dans le paragraphe 5.3.1.1 avec $a_n = \sqrt{(\log\log n)/n}$, on obtient

$$\left\|\widehat{\Sigma}_n^{-1}-\Sigma_*^{-1}\right\| = O\left(a_n\right) \quad \text{p.s.} \tag{C6}$$

Nous pouvons alors passer à l'étude de la convergence de $\widehat{\sigma}_n^2$ que l'on réécrit sous la forme $A_n - B_n^2$ avec

$$A_n = \frac{1}{n}\sum_{j=1}^{n} \widehat{\xi}_j^2 \quad \text{et} \quad B_n = \frac{1}{n}\sum_{j=1}^{n} \widehat{\xi}_j$$

où l'on rappelle que $\widehat{\xi}_j = \sqrt{(X_j - \overline{X}_n)^T \widehat{\Sigma}_n^{-1}(X_j - \overline{X}_n)}$.
Commençons par étudier la convergence de A_n que l'on décompose sous la forme

$$A_n \;=\; A_{1,n} \;+\; A_{2,n} \;-\; (\overline{X}_n - \mu)^T \Sigma_*^{-1}(\overline{X}_n - \mu) \tag{C7}$$

avec

$$A_{1,n} \;=\; \frac{1}{n}\sum_{j=1}^{n}(X_j - \overline{X}_n)^T \left(\widehat{\Sigma}_n^{-1} - \Sigma_*^{-1}\right)(X_j - \overline{X}_n) \tag{C8}$$

$$A_{2,n} \;=\; \frac{1}{n}\sum_{j=1}^{n}(X_j - \mu)^T \Sigma_*^{-1}(X_j - \mu). \tag{C9}$$

On montre facilement que $\sum_{j=1}^{n} \left\|X_j - \overline{X}_n\right\|^2 = O(n)$ p.s., et en utilisant la majoration (C6), on obtient $|A_{1,n}| = O(a_n)$ p.s.
Par ailleurs, on a

$$A_{2,n} \;=\; \frac{1}{n}\sum_{j=1}^{n} \frac{J_2(\sigma)}{J_4(\sigma} W_j^2 \quad \xrightarrow[n\to\infty]{p.s.} \quad 1 \tag{C10}$$

et puisque les (W_j) ont un moment fini d'ordre 4, la loi du logarithme itéré de Hartman et Winters [183] s'applique et on a $|A_{2,n} - 1| = O(a_n)$ p.s.
Enfin, puisque $\left\|\overline{X}_n - \mu\right\| = O(a_n)$ p.s. d'après (5.3.2), on obtient en rassemblant les différents résultats la convergence presque sûre de A_n vers 1 et la majoration

$$|A_n - 1| = O(a_n) \quad \text{p.s.} \tag{C11}$$

Étudions maintenant la convergence de B_n que l'on décompose sous la forme

$$B_n \;=\; B_{1,n} \;+\; B_{2,n} \;+\; B_{3,n} \tag{C12}$$

avec

$$B_{1,n} \;=\; \frac{1}{n}\sum_{j=1}^{n} \xi_j \tag{C13}$$

$$B_{2,n} \;=\; \frac{1}{n}\sum_{j=1}^{n} \left(\sqrt{(X_j - \mu)^T \widehat{\Sigma}_n^{-1}(X_j - \mu)} - \xi_j\right) \tag{C14}$$

$$B_{3,n} \;=\; \frac{1}{n}\sum_{j=1}^{n} \left(\widehat{\xi}_j - \sqrt{(X_j - \mu)^T \widehat{\Sigma}_n^{-1}(X_j - \mu)}\right) \tag{C15}$$

où l'on rappelle que $\xi_j = \sqrt{(X_j - \mu)^T \Sigma_*^{-1}(X_j - \mu)}$. On a immédiatement, pour tout $1 \le j \le n$,

$$(\lambda_{\min}(\Sigma_*^{-1}))^{1/2} \left\|X_j - \mu\right\| \le \xi_j \le (\lambda_{\max}(\Sigma_*^{-1}))^{1/2} \left\|X_j - \mu\right\|. \tag{C16}$$

En remarquant de plus que ξ_j peut se mettre aussi sous la forme $\xi_j = \left(\mathbb{E}W^2\right)^{-1/2} W_j$, on déduit immédiatement que

$$B_{1,n} \xrightarrow[n\to\infty]{p.s.} \frac{\mathbb{E}(W)}{(\mathbb{E}W^2)^{1/2}} \quad \text{et} \quad \left| B_{1,n} - \frac{\mathbb{E}(W)}{(\mathbb{E}W^2)^{1/2}} \right| = O\left(a_n\right) \text{ p.s.} \tag{C17}$$

En utilisant l'égalité $\sqrt{a} - \sqrt{b} = (a-b)(\sqrt{a}+\sqrt{b})^{-1}$, on déduit facilement que

$$\begin{aligned} |B_{2,n}| &\leq \frac{1}{n}\sum_{j=1}^{n} \frac{\left|(X_j-\mu)^T\widehat{\Sigma}_n^{-1}(X_j-\mu) - \xi_j^2\right|}{\sqrt{(X_j-\mu)^T\widehat{\Sigma}_n^{-1}(X_j-\mu)} + \xi_j} \\ &\leq \frac{1}{n}\sum_{j=1}^{n} \frac{\left|(X_j-\mu)^T\left(\widehat{\Sigma}_n^{-1} - \Sigma_*^{-1}\right)(X_j-\mu)\right|}{\xi_j}. \end{aligned} \tag{C18}$$

Puisque $\xi_j \geq (\lambda_{\min}(\Sigma_*^{-1}))^{1/2} \left\|X_j - \mu\right\|$, on en déduit que

$$|B_{2,n}| \leq (\lambda_{\max}(\Sigma_*))^{1/2} \left\|\widehat{\Sigma}_n^{-1} - \Sigma_*^{-1}\right\| \frac{1}{n}\sum_{j=1}^{n} \left\|X_j - \mu\right\|. \tag{C19}$$

Finalement, puisque $\sum_{j=1}^{n} \left\|X_j - \mu\right\| = O(n)$ p.s., on obtient en utilisant la majoration (C6),

$$|B_{2,n}| = O\left(a_n\right) \quad \text{p.s.} \tag{C20}$$

En utilisant l'inégalité triangulaire pour la distance de Mahanobis, on déduit immédiatement que

$$|B_{3,n}| \leq \frac{1}{n}\sum_{j=1}^{n} \sqrt{(\overline{X}_n - \mu)^T\widehat{\Sigma}_n^{-1}(\overline{X}_n - \mu)} \tag{C21}$$

et puisque pour tout $u \in \mathbb{R}^3$ et toute matrice Q définie positive, $u^TQu \leq \lambda_{\max}(Q)\left\|u\right\|^2$, on obtient

$$|B_{3,n}| \leq \left(\lambda_{\max}(\widehat{\Sigma}_n^{-1})\right)^{1/2} \left\|\overline{X}_n - \mu\right\|.$$

Finalement, puisque d'une part $\left\|\overline{X}_n - \mu\right\| = O\left(a_n\right)$ p.s. et d'autre part, $\widehat{\Sigma}_n^{-1}$ converge presque sûrement vers Σ_*^{-1} avec $\lambda_{\min}(\Sigma_*) > 0$, on obtient

$$|B_{3,n}| = O\left(a_n\right) \quad \text{p.s.} \tag{C22}$$

En combinant (C17), (C20) et (C22) avec (C12), on obtient

$$B_n \xrightarrow[n\to\infty]{p.s.} \frac{\mathbb{E}(W)}{(\mathbb{E}W^2)^{1/2}} \quad \text{et} \quad \left| B_n - \frac{\mathbb{E}(W)}{(\mathbb{E}W^2)^{1/2}} \right| = O\left(a_n\right) \text{ p.s.} \tag{C23}$$

On conclut sur la convergence de $\widehat{\sigma}_n^2 = A_n - B_n^2$ en combinant les résultats (C11) et (C23). On obtient tout d'abord

$$\widehat{\sigma}_n^2 \xrightarrow[n\to\infty]{p.s.} 1 - \frac{(\mathbb{E}(W))^2}{\mathbb{E}(W^2)} = 1 - \frac{J_3^2}{J_2 J_4} = \widetilde{\sigma}^2. \tag{C24}$$

Puis, en considérant la décomposition

$$\widehat{\sigma}_n^2 - \widetilde{\sigma}^2 \quad = \quad (A_n - 1) - (B_n - \gamma)(B_n + \gamma) \tag{C25}$$

où l'on a posé $\gamma = (\mathbb{E}W^2)^{-1/2}\mathbb{E}(W)$, et en utilisant la majoration $|B_n + \gamma| = O(1)$ p.s., on déduit facilement que

$$\left|\widehat{\sigma}_n^2 - \widetilde{\sigma}^2\right| = O(a_n) \text{ p.s.} \tag{C26}$$

ce qui achève la preuve de la convergence de $\widehat{\sigma}_n^2$.

D Méthode de simulation de la nouvelle loi

L'objet de cette annexe est de présenter la méthode de simulation de la loi de densité f introduite en (5.2.4) avec $d = 3$. Soit X un vecteur aléatoire de $\mathbb{R}^3$ de densité f et de paramètres $\mu \in \mathbb{R}^3$, $\Sigma \in \mathbb{R}^{3\times 3}$ et $\sigma^2 > 0$. Nous avons vu en section 5.2.2 que X pouvait se décomposer sous la forme

$$X = \mu + \Sigma^{1/2} W U \tag{D1}$$

où W est une variable aléatoire réelle indépendante de U uniformément distribuée sur la sphère unité de $\mathbb{R}^3$. La densité de W est connue et s'écrit pour $t \geq 0$

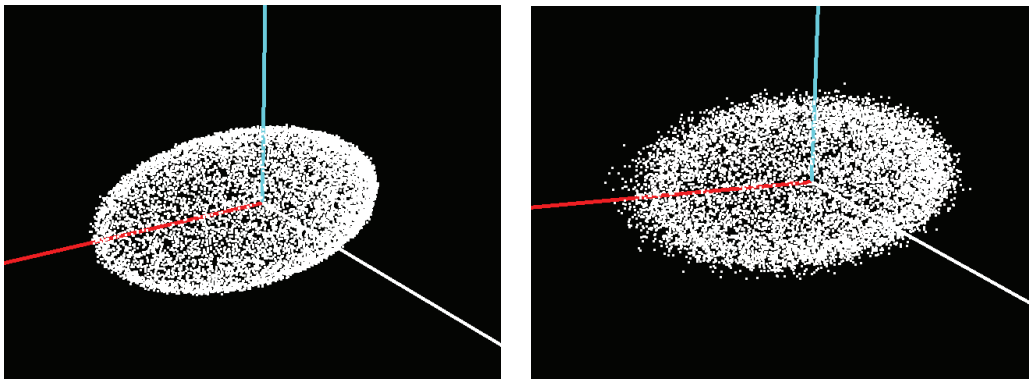
$$\varphi(t) = \frac{t^{d-1}}{J_{d-1}(\sigma)} \exp\left(-\frac{(t-1)^2}{2\sigma^2}\right).$$

Les variables aléatoires W et U sont identiques à celles intervenant dans le cas de la densité sphérique introduite Chapitre 3. Par conséquent, les méthodes de simulation de réalisations des variables aléatoires W et U sont les mêmes que celles décrites Chapitre 3 Annexe E. Des réalisations de W sont obtenues par une méthode de rejet et des réalisations de U par la méthode de la transformée inverse via un passage en coordonnées sphériques. L'algorithme de simulation de réalisations de X est résumé dans l'Algorithme D.1. Des exemples d'échantillons obtenus pour différentes valeurs de σ sont représentés Figure D1.

Algorithm D.1 Méthode de simulation de X

Require: Nombre de réalisations n à générer et paramètres (μ, Σ, σ) de la loi.

1: **for** i=1 **to** n **do**
2: Générer une réalisation $\omega > 0$ distribuée selon la loi de W.
3: Générer une réalisation $u \in \mathbb{R}^3$ distribuée selon la loi de U.
4: Calculer la réalisation de X : $x = \mu + \omega\, \Sigma^{1/2}\, u$.
5: **end for**



(a) (b)

FIGURE D1 – Échantillons issus d'un ellipsoïde centré, aligné sur les axes du repère (Ox, Oy, Oz) (rouge, vert, bleu) et de demi-axes de longueurs respectives $(1000, 500, 500)$. Le paramètre σ a été fixé à 10 en (a) et 50 en (b).

Troisième partie

Restauration d'images de profondeur présentant des fortes variations par des splines d'interpolation

Dans cette dernière partie, nous nous intéressons au problème de la restauration de données 3D, plus connu sous le nom d'*inpainting*. En effet, les systèmes d'acquisition 3D fournissent des images dans lesquelles une partie des données sont manquantes. Ce phénomène s'explique par les limites du processus de mesure de la distance dans certaines configurations. L'objectif est donc d'estimer les valeurs des pixels de l'image de profondeur ne contenant aucune information, dits indéterminés. De nombreuses méthodes ont été développées pour résoudre ce problème.

Dans cette partie, nous adoptons une approche basée sur l'approximation des données par une surface. Une fois les données modélisées, la restauration de l'image est simplement effectuée en évaluant l'approximant aux points ne contenant aucune donnée. Plus généralement, cette stratégie offre la possibilité de ré-échantillonner les données sur une grille de précision choisie. Cependant, puisque nos données contiennent de forts gradients, les méthodes d'approximation classiques conduisent à l'apparition d'oscillations parasites (phénomène de Gibbs). Pour pallier ce problème, nous nous appuyons sur les travaux d'Apprato et Gout [11] utilisant des changements d'échelle. Ces transformations permettent de contrôler les fortes variations présentes dans les données et par conséquent d'atténuer les oscillations parasites. La méthode proposée dans cette partie utilise des splines d'interpolation alors que la méthode initiale utilise des splines d'ajustement. L'objectif de cette partie est donc l'étude théorique de la convergence de la méthode de restauration et sa mise en œuvre pratique sur des données réelles.

Cette partie est composée de deux chapitres. Le Chapitre 6 rappelle des éléments fondamentaux d'analyse fonctionnelle et d'approximation de surfaces. Le Chapitre 7 détaille la méthode de restauration adoptée et présente les résultats obtenus sur des images provenant de capteurs 3D.

Chapitre 6

Rappels et notations

Sommaire

Le but de ce chapitre est de rappeler un certain nombre de définitions, propriétés et théorèmes indispensables à la bonne compréhension du contenu de cette partie. Ces rappels nous permettront dans le même temps d'introduire les notations qui seront utilisées. Le problème d'approximation de surfaces par des fonctions splines est le fil conducteur de ce chapitre. Avant cela, nous devons rappeler des définitions et résultats d'analyse fonctionnelle ainsi que des propriétés concernant les éléments finis.

Ce chapitre est organisé en trois sections. Nous commencerons dans un premier temps par rappeler les bases de l'analyse fonctionnelle en section 6.1. Nous présenterons ensuite de manière générale les éléments finis, dont quelques uns couramment utilisés en section 6.2. Enfin, nous traiterons de l'approximation de données (ajustement et interpolation) par des fonctions splines en section 6.3.

6.1 Analyse fonctionnelle

Nous commençons par préciser les notations concernant la **dérivation** qui seront employées dans cette partie. Pour plus de détails concernant le calcul différentiel, on renvoie le lecteur à H. Cartan [42]. Soient E et F deux espaces vectoriels normés, A un sous-ensemble de E et v une application définie par

$$v : A \longrightarrow F. \tag{6.1.1}$$

On suppose que la fonction v est k-fois différentiable en un point $a \in A$. La dérivée de Fréchet d'ordre k de v au point a est notée $D^k v(a)$ et sa norme $\left\|D^k v(a)\right\|$. Dans le cas particulier où $E = \mathbb{R}^n$ et $F = \mathbb{R}$, la dérivée partielle d'ordre $\alpha \in \mathbb{N}^n$ de v au point a est notée

$$\partial^\alpha v(a) = \frac{\partial^{|\alpha|} v(a)}{\partial x_1^{\alpha_1} \dots \partial x_n^{\alpha_n}} \tag{6.1.2}$$

$$= D^{|\alpha|} v(a) \cdot (\overbrace{e_1, ..., e_1}^{\alpha_1 \; fois}, ..., \overbrace{e_n, ..., e_n}^{\alpha_n \; fois}) \tag{6.1.3}$$

où $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{N}^n$ est un multi-indice, où $|\alpha| = \sum_{i=1}^n \alpha_i$ et où $\{e_j\}_{j=1}^n$désigne la base canonique de $\mathbb{R}^n$.

Nous aurons besoin dans l'annexe du Chapitre 7 le résultat de majoration suivant dû à P.G. Ciarlet et P.A. Raviart [57].

Théorème 6.1.1. *Soient X, Y et Z trois espaces vectoriels normés. Soient U un sous-ensemble ouvert de X, V un sous-ensemble ouvert de Y, et $f : U \to V$, $\varphi : V \to Z$ deux fonctions m-fois différentiables $(m \geq 1)$ au point $a \in U$ et $b = f(a)$ respectivement. La fonction $h = \varphi \circ f$ est alors m-fois différentiable au point $a \in U$ et*

$$\|D^m h(a)\| \leq C \sum_{j=1}^{m} \left\|D^l \varphi(b)\right\| \sum_{i \in I(l,m)} \|Df(a)\|^{i_1} \left\|D^2 f(a)\right\|^{i_2} \dots \left\|D^l f(a)\right\|^{i_m} \tag{6.1.4}$$

où C est une constante ne dépendant que de m et, pour $1 \leq l \leq m$,

$$I(m,l) = \{i = (i_1, i_2, \dots, i_m) \in \mathbb{N}^n, i_1 + i_2 + \dots + i_m = l \text{ et } i_1 + 2\, i_2 + \dots + m\, i_m = m\}. \tag{6.1.5}$$

Nous allons maintenant nous intéresser aux **espaces fonctionnels** et rappeler les définitions des espaces L^p et des espaces de Sobolev. Ces espaces sont à la base de toute l'analyse fonctionnelle. Pour plus de détails concernant les résultats présentés dans ce paragraphe, nous renvoyons le lecteur à R.A. Adams [4], J.L. Lions - E. Magenes [121] ou K. Yosida [209]. On considère un ouvert non vide Ω de $\mathbb{R}^n$ dont la fermeture dans $\mathbb{R}^n$ est notée $\overline{\Omega}$ et de frontière $\partial\Omega$.

Définition 6.1.1 (Espaces $L^p(\Omega)$)**.** *On note $L^p(\Omega)$ (avec $1 \leq p < \infty$) l'ensemble des fonctions v (plus précisément des classes d'équivalence de fonctions, où on identifie deux fonctions égales presque partout) mesurables sur Ω (au sens de la mesure de Lebesgue) et à valeurs dans $\mathbb{R}$ telles que $|v|^p$ soit intégrable sur Ω, c'est à dire*

$$\int_\Omega |v(x)|^p \, dx < +\infty. \tag{6.1.6}$$

L'espace $L^p(\Omega)$ est un espace de Banach muni de la norme

$$\|v\|_{L^p(\Omega)} = \left(\int_\Omega |v(x)|^p \, dx \right)^{1/p}. \tag{6.1.7}$$

Dans le cas particulier $p = 2$, l'espace $L^2(\Omega)$ est un espace de Hilbert pour le produit scalaire défini par

$$\forall u, v \in L^2(\Omega), \quad \langle u, v \rangle_{L^2(\Omega)} = \int_\Omega u(x)\, v(x) \, dx. \tag{6.1.8}$$

Définition 6.1.2 (Espace $L^\infty(\Omega)$)**.** *L'espace $L^\infty(\Omega)$ est l'ensemble des fonctions v (plus précisément des classes d'équivalence de fonctions, où on identifie deux fonctions égales presque partout) mesurables sur Ω pour lesquelles il existe une constante positive C telle que $|v(x)| \leq C$ presque partout sur Ω. L'espace $L^\infty(\Omega)$ muni de la norme*

$$\|v\|_{L^\infty(\Omega)} = \inf \left(C, 0 \leq C \leq \infty, |v| \leq C \text{ presque partout} \right) \tag{6.1.9}$$

est un espace de Banach.

Définition 6.1.3 (Espaces de Sobolev $W^{m,p}(\Omega)$)**.** *Soient m un entier positif ou nul et $p \in [1, +\infty]$. On appelle espace de Sobolev l'espace fonctionnel défini par*

$$W^{m,p}(\Omega) = \{ v \in L^p(\Omega), \partial^\alpha v \in L^p(\Omega), |\alpha| \leq m \} \tag{6.1.10}$$

avec $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ un multi-indice et où $|\alpha| = \sum_{i=1}^n \alpha_i$. L'espace $W^{m,p}(\Omega)$ est un espace de Banach pour la norme définie par

$$\begin{cases} \|v\|_{W^{m,p}(\Omega)} = \left(\displaystyle\sum_{|\alpha| \leq m} \|\partial^\alpha v\|^p_{L^p(\Omega)} \right)^{1/p} & \text{, si } 1 \leq p < \infty \\ \|v\|_{W^{m,\infty}(\Omega)} = \displaystyle\max_{|\alpha| \leq m} \left\{ \sup_{x \in \Omega} |\partial^\alpha v(x)| \right\} & \text{, si } p = \infty \end{cases} \tag{6.1.11}$$

et les semi-normes correspondantes sont notée

$$\begin{cases} |v|_{W^{m,p}(\Omega)} = \left(\displaystyle\sum_{|\alpha| = m} \int_\Omega |\partial^\alpha v|^p \, dx \right)^{1/p} & \text{, si } 1 \leq p < \infty \\ |v|_{W^{m,\infty}(\Omega)} = \displaystyle\max_{|\alpha| = m} \left\{ \sup_{x \in \Omega} |\partial^\alpha v(x)| \right\} & \text{, si } p = \infty \end{cases}. \tag{6.1.12}$$

Définition 6.1.4 (Espace $H^m(\Omega)$)**.** *Dans le cas particulier où $p = 2$, l'espace $W^{m,2}(\Omega)$ est plus simplement noté $H^m(\Omega)$. L'espace de Sobolev $H^m(\Omega)$ est donc défini par*

$$H^m(\Omega) = \left\{ v \in L^2(\Omega), \partial^\alpha v \in L^2(\Omega), \forall \alpha \in \mathbb{N}^n, |\alpha| \leq m \right\} \tag{6.1.13}$$

et est un espace de Hilbert pour le produit scalaire

$$\forall u, v \in H^m(\Omega), \quad \langle u, v \rangle_{H^m(\Omega)} = \sum_{|\alpha| \leq m} \langle \partial^\alpha u, \partial^\alpha v \rangle_{L^2(\Omega)} \tag{6.1.14}$$

dont la norme associée s'écrit

$$\forall v \in H^m(\Omega), \quad \|v\|_{H^m(\Omega)} = \left(\sum_{|\alpha| \leq m} \|\partial^\alpha v\|^2_{L^2(\Omega)} \right)^{1/2} \tag{6.1.15}$$

et la semi-norme correspondante

$$|v|_{H^m(\Omega)} = \left(\sum_{|\alpha| = m} \|\partial^\alpha v\|^2_{L^2(\Omega)} \right)^{1/2}. \tag{6.1.16}$$

La norme et la semi-norme de $H^m(\Omega)$ définies ci-dessus seront parfois respectivement notées $\|\cdot\|_{m,\Omega}$ et $|\cdot|_{m,\Omega}$.

Définition 6.1.5 (Espaces $C^k(\Omega)$, $C^k(\overline{\Omega})$ et $C^{m,\lambda}(\overline{\Omega})$)**.**

- *L'espace des fonctions continues à valeurs réelles définies sur Ω et dont les dérivées partielles jusqu'à l'ordre k sont continues sur Ω est noté $C^k(\Omega)$.*
- *L'espace $C^k(\overline{\Omega})$ est composé des fonctions $v \in C^k(\Omega)$ telles que pour chaque multi-indice α avec $|\alpha| \leq k$, l'application $x \in \Omega \mapsto \partial^\alpha v(x)$ se prolonge continûment sur $\overline{\Omega}$. Cet espace est muni de la norme suivante*

$$\forall v \in C^k(\overline{\Omega}), \quad \|v\|_{C^k(\overline{\Omega})} = \max_{|\alpha| \leq k} \sup_{x \in \Omega} |\partial^\alpha v(x)|. \tag{6.1.17}$$

- *Pour tout entier $m \geq 0$ et tout réel $\lambda \in]0, 1]$, on appelle $C^{m,\lambda}(\overline{\Omega})$ l'espace des fonctions $v \in C^m(\overline{\Omega})$ qui sont λ-höldériennes sur $\overline{\Omega}$ ainsi que toutes leurs dérivées partielles $\partial^\alpha v$ d'ordre $|\alpha| \leq m$, c'est à dire vérifiant*

$$\exists C > 0, \forall x, y \in \overline{\Omega}, \forall\, |\alpha| \leq m, \quad |\partial^\alpha v(x) - \partial^\alpha v(y)| \leq C\, \|x - y\|^\lambda_{\mathbb{R}^n}. \tag{6.1.18}$$

Théorème 6.1.2. *L'espace $C^{m,\lambda}(\overline{\Omega})$ est un espace de Banach pour la norme définie par*

$$\begin{aligned} &\forall v \in C^{m,\lambda}(\overline{\Omega}), \\ &\|v\|_{C^{m,\lambda}(\overline{\Omega})} = \|v\|_{C^m(\overline{\Omega})} + \max_{|\alpha|=m} \left(\sup_{x,y \in \Omega, x \neq y} \frac{|\partial^\alpha v(x) - \partial^\alpha v(y)|}{\|x - y\|^\lambda_{\mathbb{R}^n}} \right). \end{aligned} \tag{6.1.19}$$

Nous allons maintenant rappeler les **théorèmes d'injection** dans les espaces de Sobolev. On suppose que Ω est connexe et à frontière Lipschitzienne (voir J. Necas [43]). L'injection continue de E dans F sera notée $E \hookrightarrow F$ et l'injection compacte $E \subset^c F$.

Définition 6.1.6 (Injection continue)**.** *Soient E et F deux espaces vectoriels normés de normes respectives $\|\cdot\|_E$ et $\|\cdot\|_F$ avec $E \subseteq F$. On dit que E s'injecte continûment dans F si pour tout $x \in E$, l'application*

$$\begin{aligned} id: \quad & E \longrightarrow F \\ & x \mapsto id(x) = x \end{aligned}$$

est continue. Cette propriété se traduit par le fait qu'il existe une constante $C \geq 0$ telle que

$$\forall x \in E, \quad \|x\|_F \leq C \ \|x\|_E . \tag{6.1.20}$$

Théorème 6.1.3. *Soient $m, n \in \mathbb{N}^\star$ et $k \in \mathbb{N}$. Si $m > k + \dfrac{n}{2}$, alors on a l'injection continue suivante*

$$H^m(\Omega) \hookrightarrow C^k(\overline{\Omega}). \tag{6.1.21}$$

Théorème 6.1.4. *Pour tout entier $m \geq 0$ et tout entier p tel que $1 \leq p \leq \infty$, on a*

- $W^{m,p}(\Omega) \hookrightarrow L^{p^\star}(\Omega)$ *avec* $\frac{1}{p^\star} = \frac{1}{p} - \frac{m}{n}$ *si* $m < \frac{n}{p}$
- $W^{m,p}(\Omega) \hookrightarrow L^q(\Omega)$ *pour tout* $q \in [1, \infty[$ *si* $m = \frac{n}{p}$
- $W^{m,p}(\Omega) \hookrightarrow C^{0, m - \frac{n}{p}}(\overline{\Omega})$ *si* $\frac{n}{p} < m < \frac{n}{p} + 1$
- $W^{m,p}(\Omega) \hookrightarrow C^{0,\lambda}(\overline{\Omega})$ *pour tout* $\lambda \in]0, 1[$ *si* $m = \frac{n}{p} + 1$
- $W^{m,p}(\Omega) \hookrightarrow C^{0,1}(\overline{\Omega})$ *si* $m > \frac{n}{p} + 1$.

Définition 6.1.7 (Injection compacte)**.** *Soient E et F deux espaces vectoriels normés de normes respectives $\|\cdot\|_E$ et $\|\cdot\|_F$ avec $E \subseteq F$. On dit que E s'injecte de manière compacte dans F si*

- *E s'injecte continûment dans F,*
- *l'injection de E dans F est un opérateur compact : elle est continue et envoie toute partie bornée A de E sur une partie relativement compacte de F.*

Théorème 6.1.5. *Pour tout $1 \leq p \leq \infty$, on a les injections compactes suivantes*

- $W^{m,p}(\Omega) \subset^c L^q(\Omega)$ *pour tout* $1 \leq q \leq p^\star$ *avec* $\frac{1}{p^\star} = \frac{1}{p} - \frac{m}{n}$ *si* $m < \frac{n}{p}$
- $W^{m,p}(\Omega) \subset^c L^q(\Omega)$ *pour tout* $q \in [1, +\infty[$ *si* $m = \frac{n}{p}$
- $W^{m,p}(\Omega) \subset^c C^0(\overline{\Omega})$ *si* $m > \frac{n}{p}$.

Théorème 6.1.6. *Soient $m, n \in \mathbb{N}^\star$ et $k \in \mathbb{N}$. Soit Ω un ouvert connexe non vide à frontière Lipschitzienne. On a alors l'injection compacte suivante*

$$H^m(\Omega) \subset^c H^{m-1}(\Omega). \tag{6.1.22}$$

De plus, si $m > k + \frac{n}{2}$, alors on a l'injection compacte suivante

$$H^m(\Omega) \subset^c C^k(\overline{\Omega}). \tag{6.1.23}$$

Théorème 6.1.7. *Pour tout $r \in \mathbb{N}^\star$ et pour tout $\theta \in]0, r[$, on a l'injection compacte*

$$H^r(\Omega) \subset^c H^{r-\theta}(\Omega). \tag{6.1.24}$$

Nous rappelons enfin un théorème d'existence et d'unicité de la solution du problème de minimisation d'une fonctionnelle J sur un Hilbert. Il donne également une caractérisation de la solution.

Théorème 6.1.8 (Théorème de Stampacchia)**.** *Soit V un espace de Hilbert et $J : V \to \mathbb{R}$ une fonctionnelle de la forme*

$$J(v) = \frac{1}{2}\, a(v, v) - f(v) \tag{6.1.25}$$

où a est une forme bilinéaire, continue, symétrique, V-elliptique et $f : V \to \mathbb{R}$ une forme linéaire continue.
Soit K une partie non-vide, convexe et fermée de V. Alors il existe un unique élément $u \in K$ tel que

$$J(u) = \inf_{v \in K} J(v). \tag{6.1.26}$$

La solution u du problème (6.1.26) vérifie de façon équivalente

$$\forall v \in K, \quad a(u, v - u) \geq f(v - u) \tag{6.1.27}$$

et réciproquement, si $u \in K$ vérifie l'inégalité (6.1.27), alors u est solution de (6.1.26).
Dans le cas où K est un espace vectoriel, la solution du problème est caractérisée de manière équivalente par

$$\forall v \in K, \quad a(u, v) = f(v). \tag{6.1.28}$$

6.2 Eléments finis

Dans cette section, nous rappelons ce que sont les éléments finis et introduisons quelques types d'éléments couramment employés. La notion d'élément fini intervient notamment dans la méthode des éléments finis permettant de résoudre numériquement des équations aux dérivées partielles. Les éléments finis sont utilisés pour discrétiser un domaine continu (pour résoudre une EDP par exemple). Pour plus de détails concernant les notations introduites ainsi que les résultats présentés dans cette section, on renvoie le lecteurs à P.G. Ciarlet [56].

6.2.1 Eléments finis généraux

Nous commençons par donner une définition assez générale des éléments finis.

Définition 6.2.1 (Elément fini)**.** *Un élément fini est un triplet (K, P, Σ) où*

- *K est une partie compacte, connexe, d'intérieur non vide et à frontière Lipschitzienne de $\mathbb{R}^n$*
- *P est un espace vectoriel de fonctions sur K à valeurs dans $\mathbb{R}$*
- *Σ est un ensemble de formes linéaires $\{\sigma_1, \ldots, \sigma_N\}$ sur P. Ces formes linéaires sont appelées degrés de liberté de l'élément fini.*

L'ensemble K est un domaine géométrique : par exemple un segment en dimension 1, un triangle ou un carré en dimension 2, un tétraètre ou un cube en dimension 3. Généralement, l'espace P est choisi comme étant un espace de fonctions polynomiales. Les fonctions de P peuvent également être choisies dans un espace de fonctions à valeurs vectorielles et non plus scalaires. Ces différents éléments constituant un élément fini seront précisés dans la suite de cette section. Nous rappelons maintenant la définition de l'unisolvance.

Définition 6.2.2 (Unisolvance)**.** *L'ensemble Σ est dit P-unisolvant si l'application linéaire*

$$p \in P \longmapsto (\sigma_1(p), \ldots, \sigma_N(p))^T \in \mathbb{R}^N \tag{6.2.1}$$

est bijective. Ceci signifie en particulier que

$$\forall\, (\alpha_1, \ldots, \alpha_N) \in \mathbb{R}^N, \exists! p \in P \text{ tel que } \sigma_i(p) = \alpha_i \text{ pour } 1 \leq i \leq N. \tag{6.2.2}$$

Dans ce cas, l'élément fini (K, P, Σ) est dit unisolvant.

Remarque 6.2.1. *On trouve parfois des définitions pour lesquelles un élément fini doit forcément vérifier la propriété d'unisolvance. Lorsque l'unisolvance n'est pas mentionnée dans la définition, il s'agit d'une propriété supplémentaire et on parle alors d'élément fini unisolvant.*

Définition 6.2.3 (Eléments finis affine-équivalents)**.** *Deux éléments finis (K, P, Σ) et $(\widehat{K}, \widehat{P}, \widehat{\Sigma})$ sont affine-équivalents s'il existe une fonction affine F inversible telle que $K = F(\widehat{K})$, $\Sigma = F(\widehat{\Sigma}) = \{\sigma_i, \sigma_i(p) = \widehat{\sigma}_i(p \circ F)\}$ et $P = F(\widehat{P}) = \left\{\widehat{p} \circ F^{-1}, \widehat{p} \in \widehat{P}\right\}$.*

La notion d'éléments finis affine-équivalents est résumée par le schéma de la Figure 6.2.1. Une famille affine d'éléments finis est une famille d'éléments finis tous affine-équivalents à un même élément fini de référence. L'avantage de travailler sur une famille affine d'éléments finis est que les calculs (d'intégrales par exemple) peuvent être ramenés à des calculs sur l'élément fini de référence.

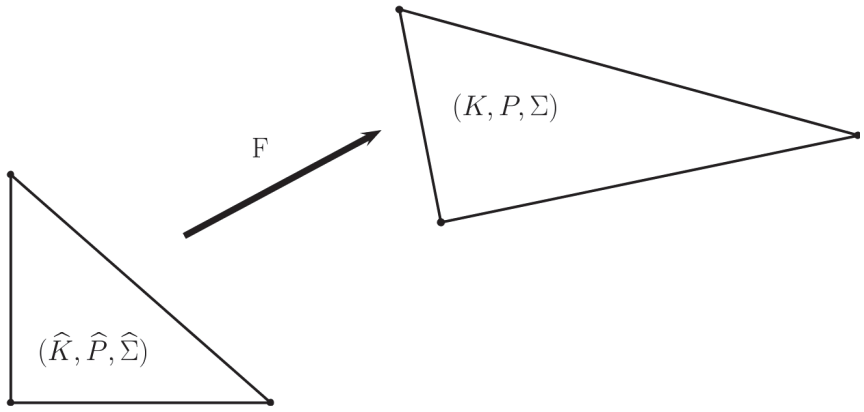


FIGURE 6.2.1 – L'élément fini (K, P, Σ) est obtenu par transformation affine de l'élément fini $(\widehat{K}, \widehat{P}, \widehat{\Sigma})$ (choisi ici triangulaire).

D'après la Définition 6.2.1, un élément fini est caractérisé par trois éléments : son domaine géométrique K, l'espace de fonctions P (généralement polynomial) et ses degrés de libertés Σ. Les notations employées concernant les espaces polynomiaux seront précisées en section 6.2.1.1. Deux grandes classes d'éléments finis seront introduits en section 6.2.1.2 : les éléments finis simpliciaux et les éléments finis parallélotopes. Enfin, nous verrons dans la section suivante que le choix de Σ conduit à plusieurs types d'éléments finis.

6.2.1.1 Espaces de polynômes

L'espace de fonctions P d'un élément fini (K, P, Σ) est généralement un espace polynomial. Nous précisons maintenant les notations retenues pour ces espaces polynomiaux.

Définition 6.2.4 (Espace $\mathbb{P}_k$)**.** *On désigne par $\mathbb{P}_k$ l'espace des polynômes de $\mathbb{R}^n$ dans $\mathbb{R}$ dont le degré total est au plus k. Un polynôme $p \in \mathbb{P}_k$ est alors de la forme*

$$p(x) = \sum_{|i| \leq k} \alpha_i \, x^i \quad \forall x \in \mathbb{R}^n \tag{6.2.3}$$

où $i = (i_1, \ldots, i_n) \in [0, k]^n$ est un multi-indice, $|i| = \sum_{j=1}^n i_j$ et $x^i = x^{i_1} \ldots x^{i_n}$.
La dimension de cet espace est $\begin{pmatrix} n+k \\ k \end{pmatrix}$.

Définition 6.2.5 (Espace $\mathbb{Q}_k$)**.** *On désigne par $\mathbb{Q}_k$ l'espace des polynômes de $\mathbb{R}^n$ dans $\mathbb{R}$ dont le degré par rapport à chaque variable est au plus k. Un polynôme $q \in \mathbb{Q}_k$ est alors de la forme*

$$q(x) = \sum_{i \in [0,k]^n} \alpha_i \, x^i \quad \forall x \in \mathbb{R}^n \tag{6.2.4}$$

où $i = (i_1, \ldots, i_n) \in [0, k]^n$ est un multi-indice et $x^i = x^{i_1} \ldots x^{i_n}$.
La dimension de cet espace est $(k+1)^n$.

Remarque 6.2.2. *On remarque immédiatement que l'on a* $\mathbb{P}_k \subset \mathbb{Q}_k$ *et* $\mathbb{P}_1 = \mathbb{Q}_1$.

Les espaces de polynômes introduits ci-dessus seront utiles lorsque des types particuliers d'éléments finis (Lagrange, Hermite) seront traités dans les sections suivantes. Selon la géométrie adoptée, la propriété d'unisolvance d'un élément fini requière le choix d'un espace de polynôme particulier.

6.2.1.2 Eléments finis simpliciaux et parallélotopes

Le domaine géométrique K d'un élément fini de $\mathbb{R}^n$ peut être choisi comme étant un n-simplexe (triangle, tétraèdre, ...) ou un n-parallélotope (carré, cube, ...). Ces choix conduisent à deux classes distinctes d'éléments finis, impliquant nous le verrons des choix différents de l'espace de fonctions P. Par conséquent, nous rappelons ici des points importants concernant ces deux types de géométrie.

On commence par s'intéresser aux éléments finis dont le domaine géométrique K est un n**-simplexe**.

Définition 6.2.6 (n-simplexe)**.** *Soit* $A = \{a_j\}_{j=1}^{n+1}$ *un ensemble de* $n+1$ *points de* $\mathbb{R}^n$ *non situés dans un même hyperplan de* $\mathbb{R}^n$. *Le* n*-simplexe* K *de sommets* A *est l'enveloppe convexe des points de* A.

En particulier, un 2-simplexe est un triangle et un 3-simplexe est un tétraèdre (voir Figure 6.2.2). Un point x de $\mathbb{R}^n$ peut être caractérisé par ses coordonnées barycentriques par rapport au simplexe K définies ci-dessous.

Définition 6.2.7 (Coordonnées barycentriques)**.** *Soit* K *un* n*-simplexe de* $\mathbb{R}^n$ *de sommets* $A = \{a_j\}_{j=1}^{n+1}$ *et un point* $x = (x(1), \ldots, x(n))^T \in \mathbb{R}^n$. *Le point* x *est caractérisé par* $(n+1)$ *réels* $(\lambda_j)_{j=1}^{n+1}$ *solutions du système linéaire*

$$\begin{cases} \displaystyle\sum_{j=1}^{n+1} a_j(i)\,\lambda_j = x(i), \quad 1 \le i \le n \\ \displaystyle\sum_{j=1}^{n+1} \lambda_j = 1 \end{cases} \tag{6.2.5}$$

où $a_j(i)$ *et* $x(i)$ *désignent respectivement la* i*ème composante des points* a_j *et* x. *Les* $(\lambda_j)_{j=1}^{n+1}$ *sont appelés coordonnées barycentriques de* x *par rapport aux sommets de* A.

La matrice du système (6.2.5) est inversible puisque les points de A ne sont pas situés dans un même hyperplan. Un n-simplexe K de $\mathbb{R}^n$ de sommets $\{a_j\}_{j=1}^{n+1}$ est caractérisé par

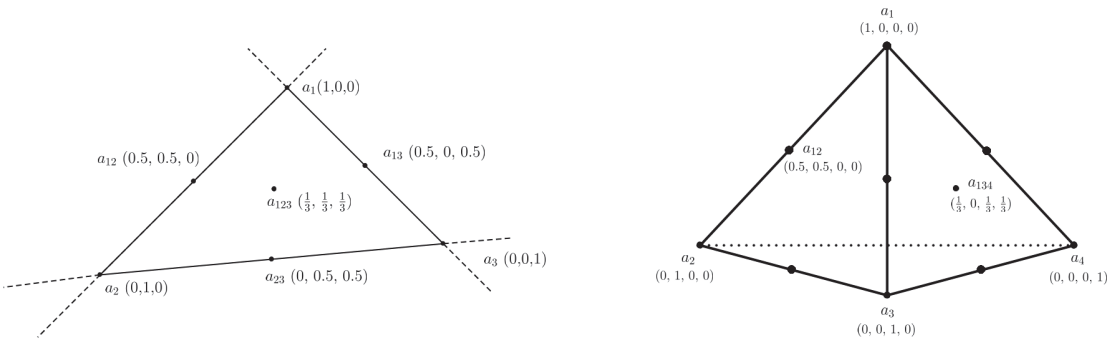
$$K = \{x \in \mathbb{R}^n, \quad 0 \le \lambda_j \le 1, \quad 1 \le j \le n+1\}, \tag{6.2.6}$$

puisque cet ensemble décrit l'enveloppe convexe des points a_j. Les coordonnées barycentriques dans un 2-simplexe (triangle) et dans un 3-simplexe (tétraèdre) sont illustrées Figure 6.2.2.

Définition 6.2.8 (Treillis principal d'ordre k)**.** *Soit* K *un* n*-simplexe de* $\mathbb{R}^n$ *de sommets* $A = \{a_j\}_{j=1}^{n+1}$. *On appelle treillis principal d'ordre* $k \ge 1$ *du simplexe* K *l'ensemble* A_k *des points de* $\mathbb{R}^n$ *dont les coordonnées barycentriques (par rapport à* A*) vérifient*

$$\lambda_j \in \left\{0, \frac{1}{k}, \ldots, \frac{k-1}{k}, 1\right\} \text{ pour } 1 \le j \le n+1 \quad \text{et} \quad \sum_{j=1}^{n+1} \lambda_j = 1. \tag{6.2.7}$$

Le treillis principal A_k *est alors composé de* $\begin{pmatrix} n+k \\ k \end{pmatrix}$ *points.*

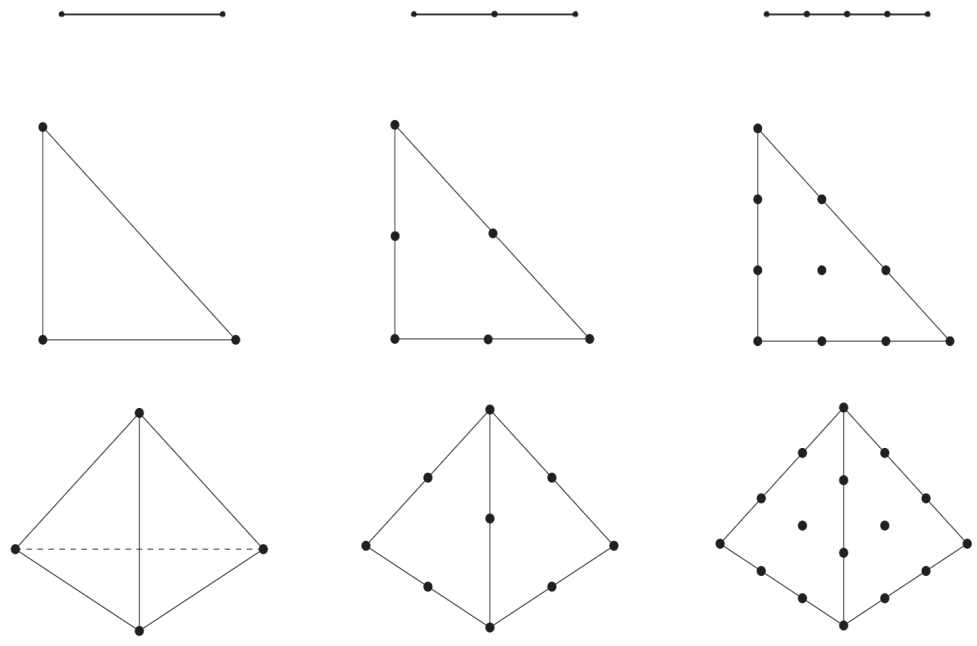


(a) (b)

FIGURE 6.2.2 – Coordonnées barycentriques dans un triangle (a) et dans un tétraèdre (b).

Théorème 6.2.1. *On note $\mathbb{P}_k$ l'ensemble des polynômes définis sur $\mathbb{R}^n$ dont le degré total est au plus k. Alors le treillis principal d'ordre k est $\mathbb{P}_k$-unisolvant et on a* $\dim(\mathbb{P}_k) = card(A_k) = \begin{pmatrix} n+k \\ k \end{pmatrix}$.

La Figure 6.2.3 représente différents treillis principaux pour les premières valeurs de k et en dimensions $n = 1$, $n = 2$ et $n = 3$.



(a) (b) (c)

FIGURE 6.2.3 – Treillis principaux A_1 en (a), A_2 en (b), et A_3 en (c) sur un segment ($n = 1$), un triangle ($n = 2$) et un tétraèdre ($n = 3$).

Définition 6.2.9 (n-simplexe de type (k))**.** *On appelle n-simplexe de type (k) ($k \geq 0$) un élément fini (K, P, Σ) où K est un n-simplexe de $\mathbb{R}^n$ dont l'ensemble des noeuds est le treillis principal A_k.*

Remarque 6.2.3. *Les définitions et théorèmes énoncés dans cette première partie de paragraphe sont très importants et justifient l'utilisation de l'espace de polynômes $\mathbb{P}_k$ pour les éléments finis de type (k).*

Théorème 6.2.2. *Deux éléments finis n-simplexes de type (k) sont affine équivalents.*

La seconde partie de cette section est consacrée à la classe d'éléments finis dont le domaine géométrique K est un n**-parallélotope**.

Définition 6.2.10 (n-parallélotope)**.** *Soit $A = \{a_j\}_{j=1}^{2^n}$ un ensemble de 2^n points de $\mathbb{R}^n$. L'enveloppe convexe de A est un n-parallélotope s'il existe une application affine inversible F de $\mathbb{R}^n$ dans $\mathbb{R}^n$ telle que*

$$a_j = F(\widehat{a_j}), \quad 1 \leq j \leq 2^n \tag{6.2.8}$$

où les $\widehat{a_j}$ sont les sommets de l'hypercube unité de $\mathbb{R}^n$.

En particulier, comme cela est illustré Figure 6.2.4, un 2-parallélotope est un parallélogramme et un 3-parallélotope un parallépipède.

Définition 6.2.11 (Treillis principal d'ordre k)**.** *Soit K un hypercube de $\mathbb{R}^n$ et $\{a_j\}_{j=1}^{2^n}$ un ensemble de points de K. On appelle treillis principal d'ordre $k \geq 1$ du n-parallélotope K l'ensemble A_k des points de $\mathbb{R}^n$ dont les coordonnées locales dans K (image la base canonique de $\mathbb{R}^n$ par F) sont*

$$\left(\frac{i_1}{k}, \ldots, \frac{i_n}{k}\right), \quad 0 \leq i_1, \ldots, i_n \leq k. \tag{6.2.9}$$

Le treillis principal d'ordre k, A_k, est alors composé de $(k+1)^n$ points.

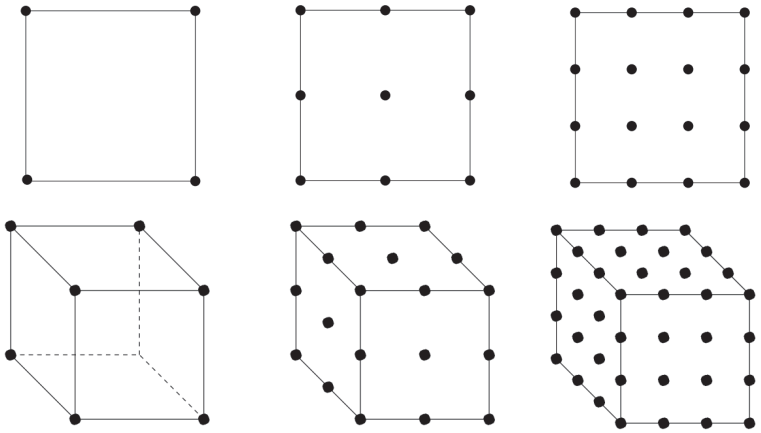
Théorème 6.2.3. *Le treillis principal d'ordre k, A_k, est $\mathbb{Q}_k$-unisolvant et on a $\dim(\mathbb{Q}_k) = card(A_k) = (k+1)^n$.*

La Figure 6.2.4 représente des treillis principaux pour les premières valeurs de k en dimensions $n = 2$ et $n = 3$.

Définition 6.2.12 (n-parallélotope de type (k))**.** *On appelle n-parallélotope de type (k) un élément fini (K, P, Σ) où K est un n-parallélotope de $\mathbb{R}^n$ dont l'ensemble des noeuds est le treillis principal d'ordre k.*

Remarque 6.2.4. *Comme dans le cas des éléments finis simpliciaux, les définitions et théorèmes de cette seconde partie de paragraphe justifient l'utilisation de l'espace de polynômes $\mathbb{Q}_k$ pour les n-parallélotopes de type (k).*

La Définition 6.2.1 décrivant un élément fini (K, P, Σ) est très générale. Nous avons apporté des précisions les éléments K et P. Dans les sections suivantes, nous présenterons quelques types d'éléments finis couramment utilisés. Nous commencerons par introduire les éléments finis de Lagrange en section 6.2.2, puis les éléments finis d'Hermite en section 6.2.3 et enfin les éléments finis de Bogner-Fox-Schmit en section 6.2.4. Avant cela, nous rappelons une définition simple et intuitive d'un maillage.



(a) (b) (c)

FIGURE 6.2.4 – Treillis principaux A_1 en (a), A_2 en (b), et A_3 en (c) sur un carré ($n = 2$) et un cube ($n = 3$).

6.2.1.3 Construction d'un maillage

L'interpolation de données sur un ensemble de taille importante est moins efficace que sur un ensemble de taille plus réduite. Par conséquent, on préfère diviser le domaine Ω en sous-domaines sur lesquels l'interpolation sera effectuée.

On considère un ouvert borné $\Omega \subset \mathbb{R}^n$. On suppose que $\overline{\Omega}$ peut s'écrire comme une union finie d'éléments géométriques polyédriques disjoints de $\mathbb{R}^n$.

Définition 6.2.13 (Triangulation)**.** *Soit Ω un domaine polyédrique de $\mathbb{R}^n$. Une triangulation T_h de Ω est une partition finie*

$$\overline{\Omega} = \bigcup_{K \in T_h} \overset{\circ}{K} \tag{6.2.10}$$

vérifiant

- *chaque élément K appartenant à T_h est un polyèdre d'intérieur non-vide de $\mathbb{R}^n$;*
- *les intérieurs de deux polyèdres distincts de T_h sont d'intersection vide ;*
- *une face d'un polyèdre K de T_h est soit une face d'un autre polyèdre de T_h, soit une partie de la frontière de Ω.*

Le paramètre h est défini par

$$h = \max_{K \in T_h} h(K) \tag{6.2.11}$$

où $h(K)$ est le diamètre de K.

Les polyèdres $K \in T_h$ peuvent être par exemple des n-simplexes ou des n-parallélotopes. La dénomination de "triangulation" ne limite pas à la seule utilisation de triangles. L'ensemble T_h constitue alors un **maillage** de Ω. Le maillage est une discrétisation spatiale du domaine continu Ω. Sa finesse est contrôlée par le paramètre h. La classe des éléments finis dépend de la régularité des fonctions de base entre plusieurs éléments.

6.2.2 Eléments finis de Lagrange

Les éléments finis de Lagrange sont couramment utilisés en pratique et constituent le type d'éléments finis le plus simple. Ils ont la caractéristique d'imposer un seul degré de liberté en chacun de leurs noeuds.

Définition 6.2.14 (Elément fini de Lagrange)**.** *Un élément fini de Lagrange est un triplet* (K, P, Σ) *où*

- K *est un polyèdre borné connexe et d'intérieur non vide de* $\mathbb{R}^n$ *(segment, triangle, rectangle, tétraèdre, cube, parallélépipède).*
- P *est un espace vectoriel de dimension finie de fonctions définies sur* K *et à valeurs dans* $\mathbb{R}$ *(généralement polynomiales).*
- $\Sigma = \{\sigma_1, \ldots, \sigma_N\}$ *est un ensemble de* N *formes linéaires sur* P *définies par*

$$\sigma_i : p \in P \longmapsto \sigma_i(p) = p(a_i) \in \mathbb{R} \tag{6.2.12}$$

où $A = \{a_1, \ldots, a_N\}$ *est un ensemble de* N *points distincts de* K*.*

Remarque 6.2.5. *Les points* $\{a_j\}_{j=1}^N$ *sont appelée noeuds de l'élément fini.*

Un élément fini (K, P, Σ) est un élément fini de de type Lagrange si Σ est P-unisolvant. La bijectivité de l'application linéaire de la Définition 6.2.2 signifie qu'étant donnés N scalaires quelconques $\{\alpha_j\}_{j=1}^N$, il existe une unique fonction $p \in P$ vérifiant

$$p(a_j) = \alpha_j, \quad 1 \leq j \leq N. \tag{6.2.13}$$

Autrement dit, si l'on fixe des valeurs α_j en chacun des points a_j de A, alors il existe une unique fonction de P passant par les points $(a_j, \alpha_j)_{j=1}^N$. On peut montrer que ceci équivaut à la propriété suivante

$$\begin{cases} dim(P) = card(\Sigma) = N \\ \forall p \in P, (p(a_i) = 0, 1 \leq i \leq N) \quad \Rightarrow \quad (p = 0) \end{cases}. \tag{6.2.14}$$

Soit (K, P, Σ) un élément fini de Lagrange. La propriété d'unisolvance implique alors qu'il existe un unique ensemble de N fonctions $p_i \in P$, $1 \leq i \leq N$ telles que

$$\sigma_j(p_i) = p_i(a_j) = \delta_{ij} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{sinon} \end{cases}. \tag{6.2.15}$$

Pour toute fonction $v \in P$, il existe un unique $p \in P$ interpolant v sur A, c'est à dire vérifiant

$$p(a_j) = v(a_j), \quad 1 \leq j \leq N, \tag{6.2.16}$$

puisqu'il suffit de choisir $p = \sum_{i=1}^N v(a_i)\, p_i$. Les fonctions p_i sont appelées fonctions de base de l'élément fini.

Définition 6.2.15 (Fonctions de base)**.** *Les* N *fonctions* $\{p_i\}_{i=1}^N$ *sont appelées fonctions de base de l'élément fini de Lagrange* (K, P, Σ)*. L'opérateur* $\prod : v \longmapsto \prod v = \sum_{i=1}^N v(a_i)\, p_i$ *est l'opérateur de* P*-interpolation de Lagrange sur* K *et* $\prod v$ *est appelé le* P*-interpolé de* v *sur* K*.*

Exemple en dimension 1. Soit $\Omega = [a, b]$ un intervalle non vide de $\mathbb{R}$. L'espace Ω est discrétisé en N intervalles $I_i = [a_i, a_{i+1}]$ où

$$a = a_0 < a_1 < \ldots < a_N = b$$

avec $a_{i+1} - a_i = \dfrac{b-a}{N} = h$. On considère l'élément fini

$$\left\{ \begin{array}{l} K = I_i \\ P = \mathbb{P}_N \\ \Sigma = \{\sigma_i : p \in P \longmapsto \sigma_i(p) = p(a_i) \in \mathbb{R}\}_{i=0}^{N} \text{ et } A = \{a_j\}_{j=0}^{N} \end{array} \right. . \tag{6.2.17}$$

Les fonctions de base de l'élément fini de Lagrange (K, P, Σ) s'écrivent dans ce cas

$$p_i(x) = \left\{ \begin{array}{ll} \frac{x-a_{i-1}}{h} & \text{si } x \in [a_{i-1}, a_i] \\ \frac{a_{i+1}-x}{h} & \text{si } x \in [a_i, a_{i+1}] \\ 0 & \text{sinon} \end{array} \right. , \text{ pour } 1 \leq i \leq N-1 \tag{6.2.18}$$

et

$$\begin{array}{l} p_0(x) = \left\{ \begin{array}{ll} \frac{a_1-x}{h} & \text{si } x \in [a_0, a_1] \\ 0 & \text{sinon} \end{array} \right. \\ p_N(x) = \left\{ \begin{array}{ll} \frac{x-a_{N-1}}{h} & \text{si } x \in [a_{N-1}, a_N] \\ 0 & \text{sinon} \end{array} \right. \end{array} \tag{6.2.19}$$

et sont appelées fonctions chapeau de part leur représentation graphique donnée Figure 6.2.5.

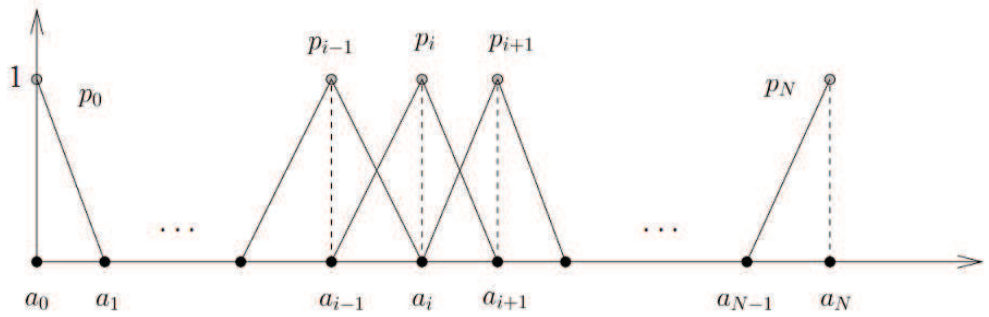


FIGURE 6.2.5 – Fonctions de base d'un ensemble d'éléments finis discrétisant un intervalle de $\mathbb{R}$.

Exemple dimension 2 (triangle). Soit $\Omega = \mathbb{R}^2$. On considère l'élément fini

$$\left\{ \begin{array}{l} K = \text{ le triangle de sommets } \{a_1, a_2, a_3\} \\ P = \mathbb{P}_1 \\ \Sigma = \{\sigma_i : p \in P \longmapsto \sigma_i(p) = p(a_i) \in \mathbb{R}\}_{i=0}^{N} \text{ et } A = A_1 \end{array} \right. . \tag{6.2.20}$$

Les fonctions de base $\{p_1, p_2, p_2\}$ de cet élément fini telles que $p_i(a_j) = \delta_{ij}$ sont les fonctions coordonnées barycentriques, i.e.

$$p_i(x) = \lambda_i(x), \quad 1 \leq i \leq 3. \tag{6.2.21}$$

On considère maintenant l'élément fini

$$\left\{\begin{array}{l} K = \text{ le triangle de sommets } \{a_1, a_2, a_3\} \\ P = \mathbb{P}_2 \\ \Sigma = \{\sigma_i : p \in P \longmapsto \sigma_i(p) = p(a_i) \in \mathbb{R}\}_{i=0}^{N} \\ \text{ et } A = A_2 = \{a_j, a_{ij}\}_{i,j=0, i<j}^{3}, \text{ avec } a_{ij} = \dfrac{a_i + a_j}{2} \end{array}\right. . \tag{6.2.22}$$

Les fonctions de base de cet élément fini s'écrivent à partir des fonctions coordonnées barycentriques, soit

$$\left\{\begin{array}{l} p_i(x) = \lambda_i(x)(2\,\lambda_i(x) - 1), \quad 1 \leq i \leq 3 \\ p_{ij} = 4\,\lambda_i(x)\,\lambda_j(x) \end{array}\right. . \tag{6.2.23}$$

Exemple dimension 2 (rectangle). On considère maintenant l'élément fini

$$\left\{\begin{array}{l} K = \text{ le rectangle de sommets } \{a_1, a_2, a_3, a_4\} \\ P = \mathbb{Q}_1 \\ \Sigma = \{\sigma_i : p \in P \longmapsto \sigma_i(p) = p(a_i) \in \mathbb{R}\}_{i=0}^{N} \\ \text{ et } A = A_1 \end{array}\right. . \tag{6.2.24}$$

Les fonctions de base de cet élément fini sont de la forme

$$p_i(x, y) = \frac{(x - x_j)\ (y - y_i)}{(x_i - x_j)\ (y_i - y_j)} \tag{6.2.25}$$

où les couples (x_i, y_i) sont les coordonnées des points a_i et où a_j est le sommet opposé à a_i.

6.2.3 Eléments finis de Hermite

Dans cette section, nous présentons l'élément fini de Hermite. Ce type d'élément fini a pour particularité de prendre en compte les valeurs des dérivées de la fonction à interpoler sur les noeuds. Le nombre de fonctions de base est donc plus élevé que pour l'élément fini de Lagrange puisque l'on a plusieurs degrés de liberté par noeud.

Définition 6.2.16 (Elément fini de Hermite)**.** *Un élément fini de Hermite est un triplet (K, P, Σ) où*

- *K est un polyèdre borné connexe et d'intérieur non vide de $\mathbb{R}^n$ (segment, triangle, rectangle, tétraèdre, cube, parallélépipède).*
- *P est un espace vectoriel de dimension finie de fonctions définies sur K et à valeurs dans $\mathbb{R}$ (généralement polynomiales).*
- *$\Sigma = \left\{\sigma_i^l\right\}_{i=1,\ldots,N}^{l=0,\ldots,k}$ un ensemble P-unisolvant de $(k+1)\,N$ formes linéaires sur P définies par*

$$\sigma_i^l : p \in P \longmapsto \sigma_i^l(p) = p^{(l)}(a_i) \in \mathbb{R}, l = \{0, 1, \ldots, k\}\,, 1 \leq i \leq N, \tag{6.2.26}$$

où $A = \{a_1, \ldots, a_N\}$ un ensemble de N points distincts de K.

Remarque 6.2.6. *Les éléments finis de Lagrange introduits en section 6.2.2 sont un cas particulier des éléments finis de Hermite pour $k = 0$.*

Exemple en dimension 1.

Sur l'intervalle $[0,1]$**.** On se place dans le cas $\Omega = [0,1]$. On considère l'élément fini suivant

$$\left\{ \begin{array}{l} K = [0,1] \\ P = P_{2k+1} \\ \Sigma = \sigma : v \mapsto v^{(l)}(s) \text{ où } s = 0,1 \text{ et } l = 0,\ldots,k \end{array} \right. . \tag{6.2.27}$$

Les fonctions de base de l'élément fini de Hermite, notées $q^s_{l,m}$, sont définies par

$$\forall t = 0,1, \forall m = 0,\ldots,k, \quad \frac{d^m\, q^s_{l,k}}{dx^m}(t) = \delta_{lm}\, \delta_{st} \tag{6.2.28}$$

pour $s = 0,1$ et $l = 0,\ldots,k$. On a de plus, pour tout $u \in [0,1]$,

$$\left\{ \begin{array}{l} q^0_{0m}(u) + q^1_{0m}(u) = 1 \\ q^1_{0m}(u) = u - q^0_{1m}(u) - q^1_{1m}(u) \\ q^0_{1m}(u) = -q^1_{1m}(1-u) \end{array} \right. . \tag{6.2.29}$$

Pour $k = 1$, les fonctions de base de l'élément fini de Hermite ont pour expression

$$\begin{array}{ccc} l & s=0 & s=1 \\ 0 & (1+2x)(x-1)^2 & x^2(3-2x) \\ 1 & x(x-1)^2 & x^2(x-1) \end{array} \tag{6.2.30}$$

et pour $k = 2$ on a

$$\begin{array}{ccc} l & s=0 & s=1 \\ 0 & (1+3x+6x^2)(1-x)^3 & x^3(10-15x+6x^2) \\ 1 & x(1+3x)(1-x)^3 & x^3(1-x)(3x+4) \\ 2 & \frac{1}{2}x^2(1-x)^3 & \frac{1}{2}x^3(1-x)^2 \end{array} . \tag{6.2.31}$$

Sur l'intervalle $[a_i, a_{i+1}]$**.** On se place maintenant sur un intervalle $[a_i, a_{i+1}] \subset \mathbb{R}$. On considère l'élément fini suivant

$$\left\{ \begin{array}{l} K_i = [a_i, a_{i+1}] \\ P_i = \mathbb{P}_{2k+1} \\ \Sigma_i = \sigma : v \mapsto v^{(l)}(a_i), v^{(l)}(a_{i+1}) \text{ où } l = 0,\ldots,k \end{array} \right. . \tag{6.2.32}$$

Les éléments finis (K_i, P_i, Σ_i) et (K, P, Σ) (sur $[0,1]$) sont affine-équivalents. L'ensemble noté $\left\{ q^{0,i}_{lk}, q^{1,i}_{lk}, l = 0,\ldots,k \right\}$ des fonctions de base de l'élément fini (K_i, P_i, Σ_i) est défini par

$$\forall l = 0,\ldots,k, \quad \left\{ \begin{array}{l} q^{0,i}_{lk} = q^0_{lk} \circ F_i^{-1} \\ q^{1,i}_{lk} = q^1_{lk} \circ F_i^{-1} \end{array} \right. \tag{6.2.33}$$

où

$$\left\{ \begin{array}{rccl} F_i : & [0,1] & \longrightarrow & [a_i, a_{i+1}] \\ & u & \longmapsto & F_i(u) = a_i + h\, u \text{ avec } h = a_{i+1} - a_i \end{array} \right. \tag{6.2.34}$$

où les q^s_{lk} sont les fonctions de base de l'élémnt fini (K, P, Σ).

6.2.4 Eléments finis de Bogner-Fox-Schmit

Nous introduisons dans cette section les éléments finis de Bogner-Fox-Schmit (notés BFS). On considère pour cela un rectangle de $\mathbb{R}^2$ noté K de dimensions $h_1, h_2 > 0$ défini par les sommets

$$b_\beta = b_{00} + \beta_1\, h_1\, e_1 + \beta_2\, h_2\, e_2 \tag{6.2.35}$$

où $b_{00} = \left(b_{00}^1, b_{00}^2\right)$ est un point de $\mathbb{R}^2$, $\beta = (\beta_1, \beta_2) \in \mathbb{N}^2$ tel que $0 \leq \beta_1, \beta_2 \leq 1$ et (e_1, e_2) la base canonique de $\mathbb{R}^2$.

On considère alors le triplet

$$\left\{\begin{array}{l} K \text{ est le rectangle de sommets } b_\beta \\ P_K = \mathbb{Q}_{2k+1} \\ \Sigma_K = \left\{\sigma_\alpha^\beta : v \mapsto \partial^\alpha v(b_\beta) \text{ avec } \alpha = (\alpha_1, \alpha_2) \in \mathbb{N}^2, 0 \leq \alpha_1 \leq k \text{ et } 0 \leq \alpha_2 \leq k\right\} \end{array}\right. . \tag{6.2.36}$$

Ce triplet est un élément fini de classe C^k appelé élément fini de Bogner-Fox-Schmit. Pour tout $\alpha = (\alpha_1, \alpha_2) \in \mathbb{N}^2$ tel que $0 \leq \alpha_1, \alpha_2 \leq k$ et $\beta = (\beta_1, \beta_2) \in \mathbb{N}^2$ tel que $0 \leq \beta_1, \beta_2 \leq k1$, soient

$$p_{\alpha k}^\beta \subset \mathbb{Q}_{2k+1} \tag{6.2.37}$$

les fonctions de base de l'élément fini de Hermite (1D) définies pour tout $(x, y) \in K$ par

$$p_{\alpha k}^\beta(z) = h_1^{\alpha_1}\, h_2^{\alpha_2}\, q_{\alpha_1 k}^{\beta_1}\left(\frac{x_1 - b_{00}^1}{h_1}\right) q_{\alpha_2 k}^{\beta_2}\left(\frac{x_2 - b_{00}^2}{h_2}\right). \tag{6.2.38}$$

Exemple d'élément fini BFS de classe C^1 sur $[0,1] \times [0,1]$. On considère l'élément fini BFS de classe C^1 suivant

$$\left\{\begin{array}{l} K = [0,1] \times [0,1] \\ P = \mathbb{Q}_3 \\ \Sigma = \left\{\sigma_\alpha^\beta : v \mapsto \partial^\alpha v(b_\beta)\right\} \end{array}\right. .$$

A chacun des sommets du rectangle sont donc associés 4 fonctions de base correspondants aux degrés de liberté qui sont la valeur de la fonction, de sa dérivée partielle première en x_1, en x_2 et de la dérivée croisée. Si l'on note $\{q_i\}_{i=1}^4$ les fonctions de base de l'élément fini de Hermite (cas $k = 1$), alors les 16 fonctions de base de l'élément fini BFS de classe C^1 sont données par

$$\left[\begin{array}{l} p_{0,0,1}^{0,0}(x_1, x_2) = q_1(x_1)\, q_1(x_2) \\ p_{1,0,1}^{0,0}(x_1, x_2) = q_3(x_1)\, q_1(x_2) \\ p_{0,1,1}^{0,0}(x_1, x_2) = q_1(x_1)\, q_3(x_2) \\ p_{1,1,1}^{0,0}(x_1, x_2) = q_3(x_1)\, q_3(x_2) \\ \text{pour le point } (0,0) \end{array}\right. \tag{6.2.39}$$

$$\left[\begin{array}{l} p_{0,0,1}^{1,1}(x_1, x_2) = q_2(x_1)\, q_2(x_2) \\ p_{1,0,1}^{1,1}(x_1, x_2) = q_4(x_1)\, q_2(x_2) \\ p_{0,1,1}^{1,1}(x_1, x_2) = q_2(x_1)\, q_4(x_2) \\ p_{1,1,1}^{1,1}(x_1, x_2) = q_4(x_1)\, q_4(x_2) \\ \text{pour le point } (1,1) \end{array}\right. \tag{6.2.41}$$

$$\left[\begin{array}{l} p_{0,0,1}^{1,0}(x_1, x_2) = q_2(x_1)\, q_1(x_2) \\ p_{1,0,1}^{1,0}(x_1, x_2) = q_4(x_1)\, q_1(x_2) \\ p_{0,1,1}^{1,0}(x_1, x_2) = q_2(x_1)\, q_3(x_2) \\ p_{1,1,1}^{1,0}(x_1, x_2) = q_4(x_1)\, q_3(x_2) \\ \text{pour le point } (1,0) \end{array}\right. \tag{6.2.40}$$

$$\left[\begin{array}{l} p_{0,0,1}^{0,1}(x_1, x_2) = q_1(x_1)\, q_2(x_2) \\ p_{1,0,1}^{0,1}(x_1, x_2) = q_3(x_1)\, q_2(x_2) \\ p_{0,1,1}^{0,1}(x_1, x_2) = q_1(x_1)\, q_4(x_2) \\ p_{1,1,1}^{0,1}(x_1, x_2) = q_3(x_1)\, q_4(x_2) \\ \text{pour le point } (0,1) \end{array}\right. \tag{6.2.42}$$

6.3 Approximation de surfaces

On considère le problème d'approximation d'une fonction f définie sur un domaine borné Ω de $\mathbb{R}^n$, à valeurs réelles et suffisamment régulière. On cherche à déterminer une approximation Φ de classe C^k de f à partir de données de Lagrange constituées de N points $\{a_i, \beta_i = f(a_i)\}_{i=1}^N$ où $a_i \in \overline{\Omega}$. Lors de l'approximation de surfaces, deux conditions sont à prendre en compte

- des conditions quantitatives : la surface peut être approchée par ajustement (passage au plus près des données) ou par interpolation (passage exactement sur les données) ;
- des conditions qualitatives : l'approximation est régulière (de classe C^k).

Les fonctions splines permettent de répondre à ces deux conditions. Les fonctions splines sont des fonctions régulières effectuant l'ajustement ou l'interpolation de données. Elles peuvent être définies (1) par la minimisation d'un critère dans un espace de Hilbert ou (2) par des fonctions polynomiales par morceaux. La définition des splines de type (1) a été formulée par Attéia [15, 16] alors que celles de type (2) apparaissent surtout dans la littérature américaine. En dimension $n = 1$, les splines possèdent les caractéristiques (1) et (2). En dimension supérieure, les deux propriétés ne peuvent être conservées.

Si l'on souhaite conserver le caractère polynomial des splines, on est amené à s'intéresser aux B-splines (voir L.L. Schumaker [172], C. de Boor et K. Höllig [66], W. Dahmen et C.A. Michelli [62]). Si l'on se place du point de vue de la minimisation d'un critère, on est amené à étudier les D^m-splines de J. Duchon [73]. Les D^m-splines conduisent à la minimisation d'une fonctionnelle sur un ouvert borné régulier Ω de $\mathbb{R}^n$. La fonctionnelle à minimiser est composée d'un terme d'attache aux données et d'un terme de régularité.

Les B-splines sont bien adaptées aux problèmes de CAO tandis que les D^m-splines sont un bon outil pour effectuer l'interpolation ou l'ajustement de données. Nous nous focalisons donc dans cette section sur les splines définies par la minimisation d'un critère sur un Hilbert. De part l'application pratique détaillée dans le chapitre suivant, nous ne traiterons que le cas de données de type **Lagrange**. Les splines d'interpolation seront abordées en section 6.3.1 et les splines d'ajustement en section 6.3.2. Les D^m-splines seront finalement introduites en section 6.3.3.

Plus précisément, nous considèrerons dans la suite de cette section les éléments suivants

- m et n deux entiers strictement positifs tels que $m > \frac{n}{2}$
- Ω un ouvert borné, connexe, non vide de $\mathbb{R}^n$ et à frontière Lipschitzienne au sens de J. Nečas [43]
- $A = \{a_1, \ldots, a_N\}$ un ensemble de N points distincts de $\overline{\Omega}$ tel que A contienne un sous ensemble P_{m-1}- unisolvant (voir Définition 6.2.2).
- un vecteur de données $\beta = (\beta_1, \ldots, \beta_N)$ de $\mathbb{R}^N$ où $\beta_i = f(a_i)$ pour $1 \leq i \leq N$.

D'après les théorèmes d'injection de la section 6.2, les hypothèses ci-dessus impliquent que l'on a l'injection continue suivante

$$H^m(\Omega) \hookrightarrow C^0\left(\overline{\Omega}\right). \tag{6.3.1}$$

On définit l'opérateur continu $\rho \in L\left(H^m(\Omega), \mathbb{R}^N\right)$ par

$$\forall v \in H^m(\Omega), \quad \rho v = (v(a_1), \ldots, v(a_N)) \in \mathbb{R}^N \tag{6.3.2}$$

qui associe à une fonction $v \in H^m(\Omega)$ sa valeur en chacun des points de A.

Enfin, on peut montrer [43] que l'application

$$|||\cdot|||_{m,\Omega} : \begin{cases} H^m(\Omega) \longrightarrow \mathbb{R}^+ \\ u \longrightarrow |||u|||_{m,\Omega} = \left(\|\rho\, u\|^2_{\mathbb{R}^n} + |u|^2_{m,\Omega} \right)^{1/2} \end{cases} \tag{6.3.3}$$

est une norme hilbertienne équivalente à la norme usuelle $\|\cdot\|_{m,\Omega}$ de $H^m(\Omega)$ introduite à la Définition 6.1.4.

6.3.1 Splines d'interpolation

L'interpolation consiste à approcher la fonction f par une fonction Φ de classe C^k telle que

$$\Phi(a_i) = f(a_i) = \beta_i, \quad 1 \leq i \leq N. \tag{6.3.4}$$

On définit la variété linéaire affine des fonctions de $H^m(\Omega)$ ayant les mêmes valeurs que f sur A

$$K_\beta = \{u \in H^m(\Omega), \rho\, u = \beta\} \tag{6.3.5}$$

et le sous-espace vectoriel associé

$$K_0 = \{u \in H^m(\Omega), \rho\, u = 0\}. \tag{6.3.6}$$

On peut vérifier que les ensembles K_β et K_0 sont fermés dans $H^m(\Omega)$ et que K_β est un convexe non-vide.

L'interpolation des données $(a_i, \beta_i = f(a_i))_{i=1}^N$ peut se formuler comme un problème de minimisation. On appelle **fonction spline d'interpolation** sur Ω relativement à ρ et à β la solution du problème de minimisation suivant

$$\begin{cases} \text{Trouver } \sigma(f) \in K_\beta \text{ tel que } \forall v \in K_\beta, \\ |\sigma(f)|_{m,\Omega} \leq |v|_{m,\Omega}. \end{cases} \tag{6.3.7}$$

Les solutions sont les fonctions ayant les mêmes valeurs que f sur A dont la semi-norme de $H^m(\Omega)$ est minimale. On peut montrer que le problème (6.3.7) est équivalent au problème suivant

$$\begin{cases} \text{Trouver } \sigma(f) \in K_\beta \text{ tel que } \forall v \in K_\beta, \\ \|\rho\, \sigma(f) - \beta\|^2_{\mathbb{R}^N} + |\sigma(f)|^2_{m,\Omega} \leq \|\rho\, v - \beta\|^2_{\mathbb{R}^N} + |v|^2_{m,\Omega} \end{cases} \tag{6.3.8}$$

qui peut lui même se réécrire

$$\begin{cases} \text{Trouver } \sigma(f) \in K_\beta \text{ tel que } \forall v \in K_\beta, \\ |||\sigma(f)|||^2_{m,\Omega} - 2\langle \rho\, \sigma(f), \beta \rangle_{\mathbb{R}^N} \leq |||v|||^2_{m,\Omega} - 2\langle \rho\, v, \beta \rangle_{\mathbb{R}^N}. \end{cases} \tag{6.3.9}$$

Proposition 6.3.1. *Les problèmes équivalents (6.3.7), (6.3.8) et (6.3.9) admettent une solution unique appelée* ***spline d'interpolation*** *sur Ω relative à ρ et β. La solution est caractérisée par*

$$\begin{cases} \sigma(f) \in K_\beta \\ \forall v \in K_0, \langle \sigma(f), v \rangle_{m,\Omega} = 0 \end{cases}. \tag{6.3.10}$$

La démonstration de cette proposition est basée sur le théorème de Stampacchia (théorème 6.1.8) présenté en section 6.1.

On introduit donc le sous ensemble suivant

$$S = \left\{ u \in H^m(\Omega), \forall v \in K_0, \quad \langle u, v \rangle_{m,\Omega} = 0 \right\} \tag{6.3.11}$$

appelé **espace des fonctions splines** sur Ω. La solution unique $\sigma(f)$ du problème d'interpolation est donc caractérisée par

$$\sigma(f) \in \{K_\beta \cap S\}. \tag{6.3.12}$$

Proposition 6.3.2. *La restriction ρ_S de ρ à S est un isomorphisme de S sur $\mathbb{R}^N$. Pour tout vecteur $\beta \in \mathbb{R}^N$, $\rho_S^{-1}(\beta)$ n'est autre que la spline d'interpolation $\sigma(f)$ sur Ω relative à ρ et à β.*

Démonstration. La démonstration de cette proposition est immédiate en utilisant la Proposition 6.3.1. □

Un isomorphisme étant une application linéaire bijective, l'espace des fonctions splines est donc de dimension N. Les fonctions splines de base, notées σ_j, $1 \leq j \leq N$, sont les splines d'interpolation relatives à l'opérateur ρ et au vecteur $\beta = e_j$ où e_j est le j^{me} élément de la base canonique de $\mathbb{R}^N$. On a alors

$$\sigma_j = \rho_S^{-1}(e_j), \quad 1 \leq j \leq N. \tag{6.3.13}$$

Étant donné que tout vecteur $\beta \in \mathbb{R}^N$ peut s'écrire comme combinaison linéaire des e_j, c'est à dire $\beta = \sum_{j=1}^N \beta_j e_j$, on peut écrire que

$$\forall \beta \in \mathbb{R}^N, \quad \sigma(f) = \rho_S^{-1}(\beta) = \sum_{j=1}^N \beta_j \sigma_j. \tag{6.3.14}$$

La famille $(\sigma_1, \ldots, \sigma_N)$ est donc une base de S.

Résultats de convergence. On s'intéresse à la convergence dans $H^m(\Omega)$ de la spline d'interpolation $\sigma(f)$ relative à ρ et à β vers la fonction f lorsque $N \to +\infty$. On considère pour cela les éléments suivants :

- un ensemble $D \subset \mathbb{R}^{+*}$ admettant 0 comme point d'accumulation ;
- pour tout $d \in D$, un ensemble ordonné A^d composé de $N(d)$ points de $\overline{\Omega}$ contenant un ensemble P_{m-1}-unisolvant et tel que
$$\sup_{x \in \Omega} \delta\left(x, A^d\right) = d \tag{6.3.15}$$
où δ désigne la distance euclidienne dans $\mathbb{R}^n$. Le paramètre d représente le rayon de la plus grande boule incluse dans Ω ne contenant aucun point de A^d ;
- pour tout $d \in D$, l'opérateur $\rho^d \in L\left(H^m(\Omega), \mathbb{R}^N\right)$ défini par
$$\forall v \in H^m(\Omega), \quad \rho^d v = (v(a))_{a \in A^d}, \tag{6.3.16}$$
associant à une fonction $v \in H^m(\Omega)$ sa valeur en chaque point de A^d.
- l'ensemble K^d par
$$K^d = \left\{ u \in H^m(\Omega), \rho^d u = \rho^d f \right\}. \tag{6.3.17}$$

Pour tout $d \in D$, on note $\sigma^d(f)$ la spline d'interpolation approximant f sur Ω relative à ρ^d et $\rho^d f$.
De plus, pour tout $d \in D$, on considère l'application

$$v \in H^m(\Omega) \longrightarrow |||v|||_{d,m,\Omega} = \left(\left\| \rho^d v \right\|^2_{\mathbb{R}^N} + |v|^2_{m,\Omega} \right)^{1/2}. \tag{6.3.18}$$

D'après un théorème d'équivalence de J. Nečas [43], on sait que cette application est une norme équivalente à la norme usuelle $\|\cdot\|_{m,\Omega}$ dans $H^m(\Omega)$.

Soit maintenant A_0 un ensemble P_{m-1}-unisolvant fixé quelconque de $\overline{\Omega}$. Les hypothèses $0 \in \overline{D}$ et (6.3.15) impliquent que

$$\forall a \in A_0, \exists \left(a^d\right)_{d\in D}, \quad \left(\forall d \in D, a^d \in A^d\right) \quad \text{et} \quad \left(a = \lim_{d\to 0} a^d\right). \tag{6.3.19}$$

Alors pour tout $d \in D$, on désigne par A_0^d l'ensemble des points de A^d ainsi associés à A_0 et on pose

$$|||v|||_d^0 = \left(\sum_{a^d \in A_0^d} v^2(a^d) + |v|^2_{m,\Omega} \right)^{1/2}. \tag{6.3.20}$$

On a alors le résultat d'équivalence de norme suivant.

Lemme 6.3.1. *Il existe $\eta > 0$ tel que pour tout $d \in D$, $d \le \eta$, l'application $|||\cdot|||_d^0$ soit une norme sur $H^m(\Omega)$ uniformément équivalente sur $D \cap]0, \eta]$ à la norme $\|\cdot\|_{m,\Omega}$.*

Démonstration. (voir R. Arcangéli [13]). □

Le résultat de convergence suivant est alors vérifié.

Théorème 6.3.1. *Soit $f \in H^m(\Omega)$ et $\sigma^d(f)$ la spline d'interpolation associée. On a alors*

$$\lim_{d\to 0} \left\| \sigma^d(f) - f \right\|_{m,\Omega} = 0. \tag{6.3.21}$$

Démonstration. (voir R. Arcangéli [13]). □

6.3.2 Splines d'ajustement

On reprend les notations de la section précédente. L'ajustement consiste à approcher la fonction f par une approximation Φ de classe C^k dont la valeur en chaque point a_i est proche (selon un certain critère) de la donnée $f(a_i)$. Les fonctions f et Φ n'ont donc pas nécessairement les même valeurs sur A. La proximité entre les données disponibles et l'approximation est définie selon un certain critère. Pour cela, on introduit la fonctionnelle suivante de paramètre $\epsilon > 0$,

$$J_\epsilon(v) = \|\rho\, v - \beta\|^2_{\mathbb{R}^N} + \epsilon |v|^2_{m,\Omega} \tag{6.3.22}$$

où le premier terme assure la fidélité aux données et où le second est un terme de lissage. Le paramètre ϵ est appelé paramètre de lissage.

Le problème d'ajustement de données $(a_i, \beta_i = f(a_i))_{i=1}^N$ par une fonction spline peut être formulé sous la forme d'un problème de minimisation. On appelle **fonction spline d'ajustement** sur Ω

relative à la semi-norme $|\cdot|_{m,\Omega}$, à l'opérateur ρ, au vecteur β et au paramètre ϵ les solutions (si elles existent) $\sigma_\epsilon(f)$ du problème de minimisation suivant

$$\begin{cases} \text{Trouver } \sigma_\epsilon(f) \in H^m(\Omega) \text{ tel que } \forall v \in H^m(\Omega), \\ J_\epsilon(\sigma_\epsilon(f)) \leq J_\epsilon(v). \end{cases} \tag{6.3.23}$$

La spline d'ajustement recherchée est la fonction minimisant le critère J_ϵ.

On peut montrer que le problème de minimisation (6.3.23) admet une unique solution $\sigma_\epsilon(f) \in H^m(\Omega)$. La spline d'ajustement $\sigma_\epsilon(f)$ appartient à l'espace des fonctions splines S introduit en (6.3.11) et est également solution du problème variationnel suivant

$$\begin{cases} \text{Trouver } \sigma_\epsilon(f) \in H^m(\Omega) \text{ tel que } \forall v \in H^m(\Omega), \\ \langle \rho\, \sigma_\epsilon(f), \rho\, v\rangle_{\mathbb{R}^N} + \epsilon \langle \sigma_\epsilon(f), v\rangle_{m,\Omega} = \langle \beta, \rho\, v\rangle_{\mathbb{R}^N}. \end{cases} \tag{6.3.24}$$

L'espace des fonctions splines S contient donc à la fois les fonctions splines d'interpolation et d'ajustement. Il ne dépend que de la semi-norme $|\cdot|_{m,\Omega}$ et de l'opérateur ρ.

Résultats de convergence. On s'intéresse dans ce paragraphe à la convergence de l'approximation.

Le premier résultat montre que lorsque $\epsilon \to 0$, la spline d'ajustement sur Ω, $\sigma_\epsilon(f)$, relative à ρ, β et ϵ est une approximation de la spline d'interpolation sur Ω, $\sigma(f)$, relative à ρ et β.

Théorème 6.3.2. *Lorsque $\epsilon \to 0$, on a : $\|\sigma_\epsilon(f) - \sigma(f)\|_{m,\Omega} = O(\epsilon)$.*

Démonstration. (voir R. Arcangéli [13]). □

Enfin, on peut montrer que comme dans le cas de l'interpolation, la spline d'ajustement $\sigma_\epsilon(f)$ converge vers f dans $H^m(\Omega)$ lorsque $N \to +\infty$.

Pour tout $d \in D$, on note $\sigma_\epsilon^d(f)$ la spline d'ajustement relative à ρ^d, $\rho^d f$ et ϵ. Le résultat de convergence suivant est alors établi.

Théorème 6.3.3. *Soit $f \in H^m(\Omega)$ et $\sigma_\epsilon^d(f)$ la spline d'ajustement associée. On a alors*

$$\lim_{d\to 0} \sigma_\epsilon^d(f) = f \text{ dans } H^m(\Omega). \tag{6.3.25}$$

Démonstration. (voir R. Arcangéli [13]). □

6.3.3 Approximation par D^m-splines

Nous allons maintenant voir comment utiliser les D^m-splines d'interpolation et d'ajustement pour approcher la fonction f. Comme on ne sait pas exprimer les D^m-splines de manière explicite, il est nécessaire de les discrétiser par la méthode des éléments finis.

On considère

- un sous-ensemble borné $H \in \mathbb{R}^{+*}$ admettant 0 comme point d'accumulation
- un ouvert borné polyédrique $\tilde{\Omega}$ de $\mathbb{R}^n$ tel que $\Omega \subset \tilde{\Omega}$
- pour tout $h \in H$, une triangulation $\tilde{T}_h$ de $\overline{\tilde{\Omega}}$ par des éléments notés K de diamètre $h_K \leq h$

– un espace de type éléments finis $\tilde{V}_h$ construit sur $\tilde{T}_h$ de dimension finie tel que

$$\tilde{V}_h \subset H^m(\tilde{\Omega}) \cap C^k(\overline{\tilde{\Omega}}) \text{ avec } k = 1 \text{ ou } 2. \tag{6.3.26}$$

On suppose en plus de cela que la famille $(\tilde{T}_h)_{h\in H}$ est régulière (au sens de P.G Ciarlet et P.A. Raviart [57]) et que l'élément fini générique (K, P_K, Σ_K) de la famille $(\tilde{V}_h)_{h\in H}$ est tel que $P_m(K) \subset P_K$. Si ces deux conditions sont vérifiées, alors il existe une famille d'opérateurs $(\tilde{\prod}_h)_{h\in H} \subset L(H^m(\tilde{\Omega}), \tilde{V}_h)$ satisfaisant l'hypothèse classique de la théorie des éléments finis

– $\exists C > 0, \forall h \in H, \forall l = 0, \ldots, m, \forall v \in H^m(\tilde{\Omega}), \quad \left|v - \tilde{\Pi}_h\, v\right|_{l,\Omega} \leq C\, h^{m-1}\, |v|_{m,\Omega},$

– $\forall v \in H^m(\tilde{\Omega}), \quad \lim_{h\to 0} \left|v - \tilde{\Pi}_h\, v\right|_{m,\Omega} = 0.$

Précisons maintenant l'espace de discrétisation V_h considéré. Pour tout $h \in H$, on considère l'ouvert Ω_h défini comme étant l'intérieur de l'union des éléments K de T_h tels que $K \cap \Omega \neq \emptyset$. Il est clair que la famille $(\Omega_h)_{h\in H}$ vérifie les relations

$$\left\{ \begin{array}{l} \forall h \in H, \quad \Omega \subset \Omega_h \subset \tilde{\Omega} \\ \lim_{h\to 0} mesure(\Omega_h \setminus \Omega) = 0 \end{array} \right. . \tag{6.3.27}$$

Pour tout $h \in H$, on définit alors l'espace V_h par l'espace des restrictions à Ω_h des fonctions de $\tilde{V}_h$. Il est clair que V_h est un sous-espace de dimension finie $M = M(h)$ de $H^m(\Omega)$.

La discrétisation des splines d'interpolation ou d'ajustement comporte deux étapes :

– formulation variationnelle du problème dans Ω_h à frontière polygonale tel que $\Omega \subset \Omega_h$. La solution du problème est alors dans $H^m(\Omega_h)$. Il suffit de remplacer Ω par Ω_h dans les équations.

– discrétisation du problème dans le sous-espace V_h de dimension finie tel que $V_h \subset H^m(\Omega_h) \cap C^k\left(\overline{\Omega}_h\right)$. L'approximant de la fonction inconnue f est la solution du problème discrétisé restreinte à Ω.

Splines d'interpolation

Commençons par traiter le cas des splines d'interpolation. Soit f une fonction de $H^m(\Omega)$. Pour tout $h \in H$, on pose

$$K_{f_h} = \{v_h \in V_h, \rho\, v_h = \rho\, f\} \quad \text{et} \quad K_{0h} = \{v_h \in V_h, \rho\, v_h = 0\}. \tag{6.3.28}$$

On considère alors le problème suivant

$$\left\{ \begin{array}{l} \text{Trouver } \sigma_h \in K_{f_h} \text{ tel que} \\ \forall v_h \in K_{0h}, \langle \sigma_h, v_h \rangle_{m,\Omega_h} = 0. \end{array} \right. \tag{6.3.29}$$

Le problème (6.3.29) est une discrétisation du problème (6.3.10) vu précédemment. On peut montrer que ce problème admet une solution unique notée σ_h (démonstration similaire à celle de la Proposition 6.3.1).

On note $\{w_j\}_{j=1}^M$ une base de V_h vérifiant $w_j(a_j) = 1$ pour $1 \leq j \leq N$. Si on pose $\beta = (\beta_j)_{j=1}^N = \rho\, f$, alors il est évident que $\sum_{j=1}^N \beta_j\, w_j \in K_{f_h}$. De plus, la famille $\{w_j\}_{j=N+1}^M$ est une base de K_{0h}. On peut donc exprimer cette solution dans la base d'éléments finis sous la forme

$$\sigma_h = \sum_{j=1}^{N} \beta_j\, w_j + \sum_{j=N+1}^{M} \alpha_j\, w_j \tag{6.3.30}$$

avec $\alpha = (\alpha_{N+1}, \ldots, \alpha_M) \in \mathbb{R}^{M-N}$. Le vecteur de coefficients α est obtenu par résolution du système linéaire (issu de (6.3.29))

$$\sum_{j=N+1}^{M} \langle w_j, w_i \rangle_{m,\Omega_h} \alpha_j = -\sum_{j=1}^{N} \beta_j \langle w_j, w_i \rangle_{m,\Omega_h} \quad \text{pour } N+1 \leq i \leq M \tag{6.3.31}$$

dont la matrice est inversible.

Enfin, on s'intéresse à la convergence de l'approximation en reprenant les notations des sections précédentes. Pour cela, on pose pour tout $h \in H$ et $d \in D$,

$$K_{fh}^d = \left\{ v_h \in V_h, \rho^d v_h = \rho^d f \right\} \quad \text{et} \quad K_{0h}^d = \left\{ v_h \in V_h, \rho^d v_h = 0 \right\}. \tag{6.3.32}$$

Pour tout $(d,h) \in D \times H$, on considère le problème (6.3.29) en remplaçant σ_h par σ_h^d, K_{fh} par K_{fh}^d et K_{0h} par K_{0h}^d. La solution unique de ce nouveau problème est notée σ_h^d. La convergence de σ_h^d vers f lorsque $(h,d) \to (0,0)$ est assurée par le théorème suivant.

Théorème 6.3.4. *Soit $f \in H^m(\Omega)$ et σ_h^d la solution du problème précédent pour tout $(d,h) \in D \times H$. Alors*

$$\lim_{(d,h)\to(0,0)} \left\| f - \sigma_h^d \right\|_{m,\Omega} = 0. \tag{6.3.33}$$

Démonstration. R. Arcangéli [13]. □

Splines d'ajustement

On peut également discrétiser le problème (6.3.24) lié aux splines d'ajustement sur le sous-espace $V_h \subset H^m(\Omega) \cap C^k(\overline{\Omega}_h)$. Ceci conduit alors au problème suivant

$$\left\{ \begin{array}{l} \text{Trouver } \sigma_{\epsilon,h} \in V_h \text{ tel que} \\ \forall v_h \in V_h, \langle \rho\, \sigma_{\epsilon,h}, \rho\, v_h \rangle_{\mathbb{R}^N} + \epsilon \langle \sigma_{\epsilon,h}, v_h \rangle_{m,\Omega} = \langle \rho\, f, \rho\, v_h \rangle_{\mathbb{R}^N}. \end{array} \right. \tag{6.3.34}$$

On peut montrer que ce problème admet une solution unique (comme pour (6.3.24)). L'approximant de la fonction inconnue f est la solution $\sigma_{\epsilon,h}$ du problème (6.3.34) restreinte à Ω.

On note $(w_i)_{i=1}^M$ une base de V_h. On peut alors écrire $\sigma_{\epsilon,h}$ sous la forme

$$\sigma_{\epsilon,h} = \sum_{j=1}^{M} \gamma_j\, w_j \tag{6.3.35}$$

où les γ_j sont des coefficients réels.

En prenant en particulier $v_h = w_k, 1 \leq k \leq M$, on s'aperçoit que le problème (6.3.34) est équivalent au problème suivant

$$\left\{ \begin{array}{l} \text{Trouver } \gamma = (\gamma_1, \ldots, \gamma_M) \in \mathbb{R}^M \text{ solution du système linéaire} \\ \left(A^T A + \epsilon R\right) \gamma = A^T \rho\, f \end{array} \right. \tag{6.3.36}$$

avec

$$A = (w_j(a_i))_{1\leq i\leq N, 1\leq j\leq M} \quad \text{et} \quad R = \left(\langle w_i, w_j \rangle_{m,\Omega_h} \right)_{1\leq i,j\leq M}. \tag{6.3.37}$$

Enfin, on s'intéresse à la convergence de l'approximation. Pour cela, on suppose que $\epsilon \in]0, \epsilon_0]$, où ϵ_0 est un nombre strictement positif. On suppose également que

$$\exists C > 0, \exists \theta > 0, \forall d \in D, d \leq \theta, \quad N(d) \leq \frac{C}{d^2}. \tag{6.3.38}$$

Cette dernière propriété traduit la régularité asymptotique de la répartition des points de A^d dans $\overline{\Omega}$.

Soit $f \in H^m(\Omega)$. Pour tout $(d, h, \epsilon) \in D \times H \times]0, \epsilon_0]$, on considère le problème variationnel (6.3.34) en remplaçant $\sigma_{\epsilon,h}$ par $\sigma^d_{\epsilon,h}$ et ρ par ρ^d. Ce nouveau problème discrétisé admet une solution unique $\sigma^d_{\epsilon,h}$.

Soit $\tilde{f} \in H^m(\tilde{\Omega})$ un prolongement de f par continuité. On s'intéresse alors au problème

$$\left\{ \begin{array}{l} \text{Trouver } \sigma_{\epsilon,h} \in V_h \text{ tel que} \\ \forall v_h \in V_h, \left\langle \rho^d \, \sigma^d_{\epsilon,h}, \rho^d \, v_h \right\rangle_{\mathbb{R}^N} + \epsilon \left(\sigma^d_{\epsilon,h}, v_h \right)_{m,\Omega} = \left\langle \rho^d \, \tilde{f}, \rho^d \, v_h \right\rangle_{\mathbb{R}^N} \end{array} \right. \tag{6.3.39}$$

admettant une solution unique $\sigma^d_{\epsilon,h}$. La convergence de $\sigma^d_{\epsilon,h}$ vers f est assurée par le théorème suivant.

Théorème 6.3.5. *Soit $f \in H^m(\Omega)$ et $\sigma^d_{\epsilon,h}$ la solution de (6.3.39). Alors*

$$\lim_{(d,h)\to(0,0)} \left\| f - \sigma^d_{\epsilon,h} \right\|_{m,\Omega} = 0. \tag{6.3.40}$$

Démonstration. R. Arcangéli [13]. □

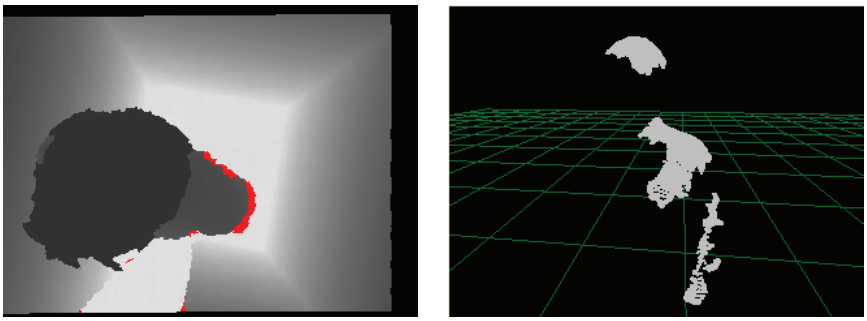
Chapitre 7

Restauration d'images de profondeur par des splines d'interpolation

Sommaire

7.1 Introduction

Dans ce chapitre, nous nous intéressons au problème de restauration des images de profondeur. Comme cela a été souligné dans les chapitres précédents (1 et 2), les images de profondeur retournées par le capteur 3D peuvent contenir des pixels ne contenant aucune information de distance. L'absence de données est généralement due aux propriétés ou à l'orientation de la surface de l'objet. Par exemple, les systèmes de mesure classiques ne sont en général pas capable d'estimer la distance d'objets en verre, de surfaces parallèles avec l'axe optique du capteur ou des zones trop exposées à la lumière naturelle. Les images présentées Figure 7.1.1 illustrent le problème des données manquantes. Les zones rouges correspondent aux pixels dits indéterminés ne contenant aucune information de distance. En plus de cela, il est avéré que les valeurs de distance estimées par les capteur 3D à lumière structurée (que nous utilisons) contiennent une erreur de mesure s'amplifiant de manière quadratique avec l'éloignement de l'objet (voir par exemple [112]).



(a) (b)

FIGURE 7.1.1 – En (a), une image de profondeur présentant des données manquantes. Les pixels indéterminés sont représentés en rouge pour plus de clarté. En (b), le nuage de points 3D représentant la personne seule. Les données contiennent des fortes variations, par exemple entre l'épaule et la tête ou entre le corps et le sol (grille verte).

Il est clair que la qualité des données manipulées conditionne le bon fonctionnement des algorithmes de traitement d'image. Nous cherchons donc dans ce chapitre à restaurer les images de profondeur tout en respectant les deux contraintes suivantes :

- conserver les pixels valides
- combler les zones présentant des données manquantes

Le problème de reconstruction d'images présentant des zones indéterminées est bien connu dans le domaine du traitement d'image (sous le nom d'**inpainting**). Il existe tout un ensemble de méthodes classiques pour résoudre ce problème. On distingue deux classes d'algorithmes : une première basée sur des techniques de filtrage et une seconde basée sur la résolution de problèmes variationnels.

Une méthode simple consiste à estimer les valeurs manquantes par la valeur moyenne des pixels appartenant au 4-voisinage (ou 8-voisinage) du pixel indéterminé considéré ([86] par exemple). Une technique de filtrage bilatéral est proposée dans [41]. Les valeurs des pixels restaurés sont calculées comme une combinaison des valeurs valides voisines. Les pixels adjacents sont pondérés selon leur intensité (information colorimétrique), leur profondeur (information de distance) et selon une carte de consistance temporelle. Des techniques de filtrage non-local basées sur la couleur et la distance ont

été expérimentées (voir par exemple [152]). On remarque qu'une partie des méthodes de la littérature combinent les informations des cartes de couleur et de profondeur afin d'améliorer les résultats. Étant donné que certains capteurs ne fournissent pas d'image couleur, seule l'image de profondeur sera prise en compte dans nos travaux.

Des méthodes variationnelles ont également été proposées pour restaurer des images. Le travail initial de Bertalmio et al. [21] a posé des bases pour la résolution de ce problème. Le modèle utilisé est basé sur des équations aux dérivées partielles non linéaires. Il s'inspire des techniques employées par les artistes spécialisés en restauration. En particulier, ces travaux partent du principe qu'un bon algorithme de restauration doit propager les contours forts (gradients élevés) situés autour de la zone dégradée. Ceci peut être effectué en connectant les isocontours de l'image en niveaux de gris entre eux à l'intérieur de la zone à reconstruire (voir également Masnou et Morel [126] et Masnou [125]) de telle sorte que les niveaux de gris soient étendus continûment à l'intérieur de la zone d'intérêt. A. Bertozzi et al. [22] utilisent un modèle basé sur l'équation de Cahn-Hilliard, permettant la restauration rapide et efficace de textes. Le modèle introduit par Chan et Shen [50] utilise un modèle de débruitage basé sur variation totale de Rudin, Osher et Fatemi [162] dans le cadre de la restauration d'images. Ce modèle est en effet capable de propager les contours forts dans la zone endommagée. Toutefois, du fait du terme de régularisation, le modèle impose une pénalité sur la longueur des contours et ne peut pas par conséquent connecter des contours sur des distances trop importantes. Un autre inconvénient est que ce modèle n'étend pas continûment la direction des isocontours aux bords du domaine à traiter.

Cependant, nous avons choisi dans ce travail d'utiliser une approche issue de l'**approximation de surfaces** (à partir de données de Lagrange) consistant à approcher les données par une surface régulière. L'approximation de la surface des objets permet d'une part de combler les zones de données manquantes et d'autre part de rééchantilloner le nuage de points 3D sur une grille de précision choisie. Ce dernier point constitue un avantage non négligeable par rapport à la plupart des méthodes de la littérature.

Une image est obtenue par un mécanisme de projection de la scène réelle sur le capteur. Chaque pixel mesure une information (couleur, distance) sur une portion de l'objet correspondant. Par conséquent, dans le cas d'une image de profondeur, les gradients de l'image sont forts lorsque deux objets adjacents sont situés à des profondeurs différentes, i.e. quand la distance varie brutalement. On peut observer ce phénomène Figure 7.1.1 où l'épaule de la personne et le sol sont proches dans l'image mais éloignés dans l'espace 3D. Leurs distances au capteur, et donc les valeurs associées à chaque pixel, sont très différentes. D'autre part, si l'on considère un faible déplacement sur le sol, la hauteur peut varier brutalement lorsque l'on rencontre un objet. Nous sommes donc confrontés au traitement de données pouvant présenter de **fortes variations**.

Le problème soulevé est bien connu en théorie de l'approximation. Sans aucune information a priori sur les fortes variations (localisation et amplitude), l'application de méthodes usuelles d'approximation conduit à l'apparition d'oscillations parasites pouvant dénaturer localement ou globalement l'approximation. On parle alors de phénomène de Gibbs. Un exemple d'oscillations parasites obtenues lors de l'ajustement d'une courbe est représenté Figure 7.1.2. Dans le cadre de la restauration d'images de profondeur, ces oscillations conduisent à une mauvaise estimation des valeurs manquantes au voisinage des zones de fort gradient.

Par conséquent, nous nous appuyons sur une méthode d'approximation de surface proposée par Apparato et Gout [11] faisant intervenir des **changements d'échelle**. Ce type de transformation est courant en approximation de surfaces et permet d'atténuer l'influence des fortes variations. L'opérateur

d'approximation utilisé dans [11] est un opérateur spline d'ajustement. Dans ce travail, nous avons choisi d'utiliser un opérateur spline d'interpolation afin d'obtenir une surface passant exactement par les données. Le but de ce travail est donc d'appliquer la méthode au problème de restauration de cartes de profondeur provenant de capteurs 3D. La convergence de la méthode est étudiée d'un point de vue théorique.

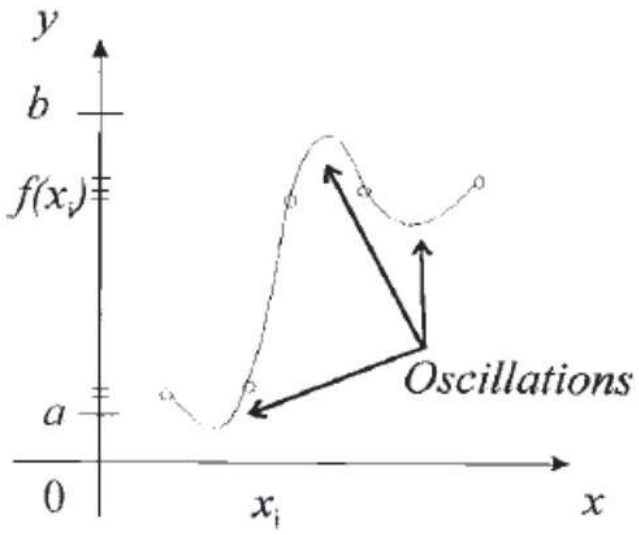


FIGURE 7.1.2 – Exemple d'oscillations parasites lorsque l'on approche, avec une méthode usuelle, une fonction présentant de fortes variations.

Ce chapitre est organisé de la manière suivante. La modélisation du problème et le principe général de la méthode d'approximation seront présentés en section 7.2. La construction des changements d'échelle sera ensuite décrite en section 7.3. Nous rappellerons les caractéristiques principales des D^m-splines en section 7.4. La méthode sera expérimentée sur des données réelles en section 7.5. Nous conclurons enfin le chapitre en section 7.6. La démonstration de la convergence de la méthode sera détaillée en Annexe A.

7.2 Principe général de la méthode d'approximation

La méthode de reconstruction étudiée dans ce chapitre est basée sur les travaux d'Apprato et Gout [11] faisant intervenir deux changements d'échelle. Le principe général de la méthode est exposé en section 7.2.1. Les notations et hypothèses seront ensuite précisées en section 7.2.2.

7.2.1 Changements d'échelle

L'algorithme proposé présente l'avantage de ne pas créer d'oscillations parasites lorsque les données varient brutalement. L'objectif est donc d'approcher une fonction explicite

$$f : \overline{\Omega} \longrightarrow \mathbb{R} \tag{7.2.1}$$

à partir de données de Lagrange $(x_i, f(x_i))_{i=1}^n$, où Ω est un ouvert de $\mathbb{R}^n$, et ce sans préjuger a priori de la présence ou non de fortes variations sur f. De plus, la méthode ne doit pas dépendre de l'opérateur d'approximation choisi. Elle doit, en supposant quelques hypothèses, pouvoir s'appliquer aux opérateurs d'approximation usuels.

L'algorithme a été développé dans le cadre de l'ajustement de données. Dans le cadre de la restauration d'images, étant donné que l'on souhaite conserver les valeurs valides, nous nous intéressons au problème d'interpolation plutôt qu'à celui de l'ajustement. Notons toutefois que l'ajustement des données impliquait un lissage atténuant le bruit de mesure.

L'idée centrale de la méthode est d'appliquer deux changements d'échelle afin de réduire les fortes variations. Le premier intervient en pré-traitement sur les données $(f(x_i))$ avant que toute approximation ne soit effectuée. Le second changement d'échelle est quant à lui appliqué en post-traitement de l'approximation des données pré-traitées.

Pour illustrer ce principe, on se place dans le cas de l'approximation d'une courbe d'équation $z = f(x)$. Soit $(x_i, f(x_i))_{i=1}^N$ un ensemble quelconque de points de $[0,1] \times [a,b]$ constituant les données. L'idée est d'atténuer les variations éventuelles en déplaçant les valeurs $(f(x_i))$. On considère pour cela un premier changement d'échelle noté φ_d transformant les valeurs $\{f(x_i)\}_{i=1}^N$ en valeurs $\{u_i\}_{i=1}^N$ régulièrement réparties dans un intervalle $[\alpha, \beta]$ arbitrairement choisi :

$$\varphi_d : [a,b] \longrightarrow [\alpha, \beta]\,. \tag{7.2.2}$$

Le rôle de la transformation φ_d est de contrôler les variations éventuelles de f. Le résultat de l'application de ce premier changement d'échelle aux données est illustré Figure 7.2.1. Regardons plus précisément l'impact de la transformation sur les variations de f. Le rapport de deux pentes discrètes successives vis à vis des données pré-traitées par le changement d'échelle φ_d vaut

$$\frac{x_{i+1} - x_i}{x_i - x_{i-1}} \tag{7.2.3}$$

alors qu'il valait

$$\frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \, \frac{x_{i+1} - x_i}{f(x_{i+1}) - f(x_i)} \tag{7.2.4}$$

sans pré-traitement. Autrement dit, la variation du gradient de la fonction f ne dépend plus, après le changement d'échelle, des valeurs de la fonction (et donc de ses possibles variations importantes).

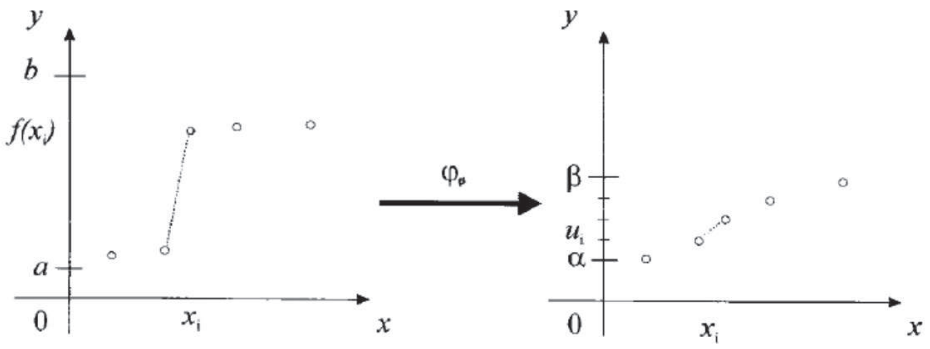


FIGURE 7.2.1 – Les données sont réparties régulièrement dans l'intervalle $[\alpha, \beta]$ pour atténuer les fortes variations.

L'approximation de f par un opérateur d'approximation, par exemple une spline d'interpolation, sur les données pré-traitées donne une courbe ne présentant pas ou peu d'oscillations parasites puisque les fortes variations ont été atténuées par le changement d'échelle φ_d. La courbe obtenue à partir des données pré-traitées est représentée Figure 7.2.2. Il s'agit d'une approximation des données de Lagrange pré-traitées $(x_i, u_i)_{i=1}^N$ mais pas des données initiales $(x_i, f(x_i))$.

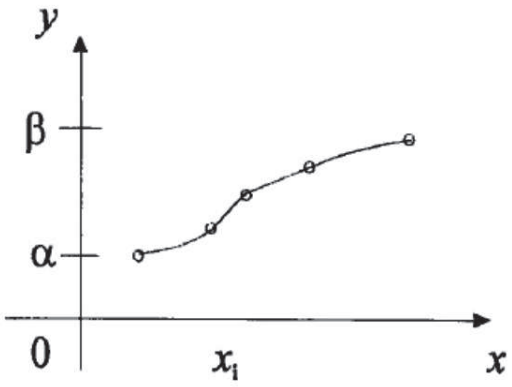


FIGURE 7.2.2 – La courbe approchant les données pré-traitées ne présente pas d'oscillations parasites.

Dans le but de revenir dans l'espace de travail initial, on considère un second changement d'échelle noté ψ_d, lié à φ_d, de la forme

$$\psi_d : [\alpha, \beta] \longrightarrow [a, b]\,. \tag{7.2.5}$$

Cette transformation est appliquée en post-traitement de l'algorithme d'approximation. On obtient alors une courbe approchant les données initiales $(x_i, f(x_i))_{i=1}^N$ sans oscillations parasites, dans la mesure où la fonction ψ_d n'en crée pas. La courbe finale approchant la fonction f est représentée Figure 7.2.3.

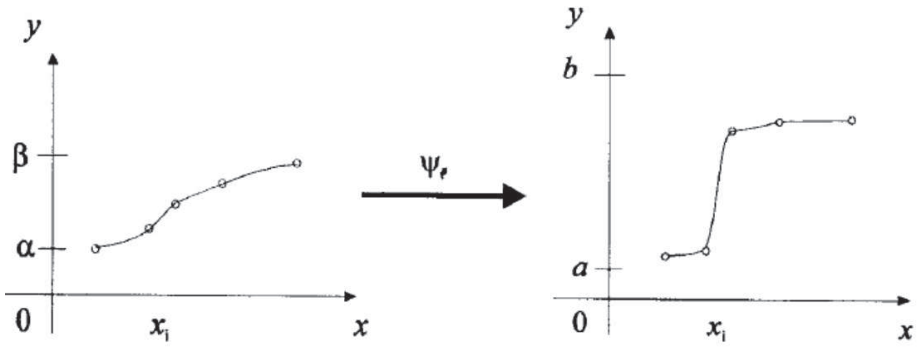


FIGURE 7.2.3 – L'approximant est ramené dans l'espace de travail initial via le changement d'échelle ψ_d. La courbe résultante ne présente pas d'oscillations indésirables.

Afin de vérifier la validité de la méthode proposée, la convergence de l'approximation construite avec les changements d'échelle vers la fonction f inconnue est étudiée. Avant cela, les notations et hypothèses sont précisées dans la section suivante.

7.2.2 Notations et hypothèses

Nous introduisons dans cette section quelques notations spécifiques au problème d'approximation de surface considéré dans ce chapitre. Nous commençons par définir précisément la fonction f inconnue que l'on souhaite approcher. Soient

- Ω un ouvert borné connexe à frontière Lipschitzienne de $\mathbb{R}^n$, $n \in \mathbb{N}$
- deux entiers $m \in \mathbb{N}^*$, $m \geq 2$ et $m' \in \mathbb{N}$ tel que $m' > m + \frac{n}{2}$
- une fonction $f : \overline{\Omega} \longrightarrow \mathbb{R}$ fixée telle que $f \in H^{m'}(\Omega)$.

Nous définissons maintenant plus précisément les données disponibles à partir desquelles la fonction f inconnue sera approchée. Pour cela, on considère

- un sous-ensemble D de réels strictement positifs admettant 0 comme point d'accumulation
- pour tout $d \in D$, un sous-ensemble A^d composé de $N = N(d)$ points de données distincts de $\overline{\Omega}$ tels que

$$\sup_{x \in \overline{\Omega}} \left(\inf_{y \in A^d} \delta(x, y) \right) = d \tag{7.2.6}$$

  où δ désigne la distance euclidienne de $\mathbb{R}^n$. Le paramètre d représente le rayon de la plus grande boule ouverte incluse dans Ω ne contenant aucun points de A^d. L'ensemble A^d correspond aux points de $\overline{\Omega}$ pour lesquels la valeur de la fonction f est connue.
- pour tout $d \in D$, l'ensemble Z_1^d des valeurs échantillonnées sur la surface définie par f et défini par

$$Z_1^d = f\left(A^d\right). \tag{7.2.7}$$

  L'ensemble Z_1^d représente les valeurs de la fonction f aux points de A^d.

– pour tout $d \in D$, le sous-ensemble Z_2^d de Z_1^d défini comme étant les $p(d)$ éléments distincts de Z_1^d ordonnés de manière croissante

$$z_1^d < z_2^d < \ldots < z_{p(d)}^d. \tag{7.2.8}$$

Nous formulons maintenant un ensemble d'hypothèses ([11]) sur les changements d'échelle φ_d, ψ_d et sur l'opérateur d'approximation T^d. On suppose que la famille de changements d'échelle (φ_d) vérifie

$$\begin{cases} (i) \quad \varphi_d(z_i^d) = u_i^d, \quad 1 \leq i \leq p(d) \\ (ii) \ \ \varphi_d \in C^m\left([a,b],[\alpha,\beta]\right) \\ (iii) \ \lim\limits_{d\to 0} \varphi_d = \varphi \in C^m\left([a,b],[\alpha,\beta]\right) \end{cases} \tag{7.2.9}$$

où les $\left(u_i^d\right)_{i=1}^{p(d)}$ constituent une subdivision quelconque d'un intervalle $[\alpha,\beta] \subset \mathbb{R}$, $[a,b] = Im(f)$ et φ est une fonction régulière. De même, on suppose que la famille de changements d'échelle (ψ_d) vérifie pour tout $d \in D$,

$$\begin{cases} (i) \quad \psi_d(v_i^d) = w_i^d, \quad 1 \leq i \leq p(d) \\ (ii) \ \ \psi_d \in C^m\left([r,s],[\delta,\gamma]\right) \\ (iii) \ \lim\limits_{d\to 0} \psi_d = \psi \in C^m\left([r,s],[\delta,\gamma]\right) \end{cases} \tag{7.2.10}$$

où les $\left(v_i^d\right)_{i=1}^{p(d)}$ constituent une subdivision quelconque d'un intervalle $[r,s] \subset \mathbb{R}$, les $(w_i)_{i=1}^{p(d)}$ constituent une subdivision quelconque d'un intervalle $[\delta,\gamma] \subset \mathbb{R}$ et ψ est une fonction régulière.

Les changements d'échelle considérés sont supposés être monotones sur $\mathbb{R}$.

Enfin, on considère un opérateur d'approximation de données de type Lagrange noté T^d pour tout $d \in D$. On peut établir la convergence de l'approximation quand $d \to 0$, lorsque l'on applique un opérateur T^d à la fonction $\varphi_d \circ f$ (pré-traitement des données) et que l'on compose le résultat de l'approximation $T^d\left(\varphi_d \circ f\right)$ avec la fonction ψ_d (post-traitement du résultat de l'approximation). Pour cela, il est nécessaire de faire les hypothèses suivantes sur la famille d'opérateurs $\left(T^d\right)_{d\in D}$

$$\begin{cases} (i) \quad T^d \in \mathcal{L}_c\left(H^m\left(\Omega,[\alpha,\beta]\right), H^m\left(\Omega,[r,s]\right)\right) \\ (ii) \ \ \left(T^d\left(\varphi_d \circ f\right)\right) \text{ est bornée dans } H^m\left(\Omega\right) \\ (iii)\, T^d\left(\varphi_d \circ f\right) \underset{d\to 0}{\longrightarrow} \varphi \circ f \text{ dans } C^0\left(\overline{\Omega}\right) \end{cases} \tag{7.2.11}$$

où $[\alpha,\beta]$, $[r,s]$, φ_d et φ ont été introduits en (7.2.9) et (7.2.10).

On suppose l'hypothèse (7.2.9) satisfaite. Le choix $f \in H^{m'}\left(\Omega,[a,b]\right)$ avec $m' > m + \frac{n}{2}$ implique que l'on ait l'injection continue $H^{m'}(\Omega) \hookrightarrow C^m(\overline{\Omega})$. Alors on peut montrer que la famille de fonctions $\left(\varphi_d \circ f\right)_{d\in D}$ est bornée dans $C^m\left(\overline{\Omega}\right)$. Ce résultat sera utile lors de l'étude de la convergence de la méthode.

La convergence de la méthode d'approximation peut être établie sous les hypothèses (7.2.9), (7.2.10) et (7.2.11). Le résultat est donné par le théorème suivant.

Théorème 7.2.1. *Sous les hypothèses (7.2.9), (7.2.10) et (7.2.11), on a*

$$\lim_{d\to 0}\left\{\psi_d \circ T^d(\varphi_d \circ f)\right\} = (\psi \circ \varphi \circ f) \ \textit{dans}\ H^{r-\theta}\left(\Omega\right) \tag{7.2.12}$$

pour tout $r \in \mathbb{N}$ *tel que*

$$r < m - \frac{n}{2} \qquad (\textit{donc } H^m(\Omega) \textit{ s'injecte continûement dans } C^r(\overline{\Omega}))$$

et pour tout $\theta > 0$ *tel que*

$$\theta < r - \frac{n}{2} \qquad (\textit{donc } H^{r-\theta}(\Omega) \textit{ s'injecte continûment dans } C^0(\overline{\Omega})).$$

Démonstration. La preuve du théorème est détaillée en annexe A pour les changements d'échelle décrits en section 7.3 et pour l'opérateur d'interpolation spline présenté en section 7.4. □

La relation (7.2.12) constitue un résultat théorique permettant de montrer la convergence de l'approximation en utilisant les changements d'échelle φ_d et ψ_d. Si l'on choisit $\psi = \varphi^{-1}$, la convergence vers la fonction f inconnue est établie. Dans ce cas, la convergence est démontrée pour tout choix de φ. Remarquons de plus que dans le cas $n = 2$ et $m = 3$, la convergence de l'approximation a lieu dans $H^{2-\theta} \subset C^0\left(\overline{\Omega}\right)$ pour tout $\theta > 0$ suffisamment petit.

7.3 Construction des changements d'échelle

Cette partie traite de la construction des familles de changements d'échelle (φ_d) et (ψ_d). La section 7.3.1 établit quelques résultats préliminaires. La construction des familles de changements d'échelle sera ensuite traitée en section 7.3.2.

7.3.1 Résultats préliminaires pour les familles de changements d'échelle

Dans cette section, nous établissons un ensemble de résultats préliminaires qui seront utiles dans la suite du chapitre pour construire les familles de changements d'échelle (φ_d) et (ψ_d) de telle sorte que la convergence de l'approximation soit assurée.

Les deux premières propositions concernent les propriétés des ensembles de données A^d et Z_2^d lorsque le paramètre d tend vers zéro, c'est à dire quand le nombre de points de données tend vers l'infini.

On rappelle que l'ensemble A^d a été défini comme un ensemble de $N(d)$ points distincts de $\overline{\Omega}$ tels que

$$\sup_{x\in\overline{\Omega}}\left(\inf_{y\in A^d}\delta\left(x,y\right)\right)=d$$

où δ désigne la distance euclidienne dans $\mathbb{R}^n$.

Proposition 7.3.1. *Le nombre d'éléments de A^d vérifie la propriété suivante*

$$card\left(A^d\right)=N(d)\underset{d\to 0}{\longrightarrow}+\infty. \tag{7.3.1}$$

Démonstration. La preuve de cette proposition est basée sur un raisonnement par l'absurde. Supposons que $N(d)$ ne tend pas vers une quantité infinie, i.e.

$$N(d)\underset{d\to 0}{\nrightarrow}+\infty$$

ce qui se traduit par

$$\exists M>0,\forall\eta>0,\exists d\in D,\quad d\leq\eta\ \Rightarrow\ N(d)\leq M.$$

Soit alors F un ensemble formé de $(E(M)+1)$ points distincts $\{x_i\}_{i=1}^{E(M)+1}$ de $\overline{\Omega}$, où $E(M)$ désigne la partie entière de M. Puisque l'on a

$$\sup_{x\in\overline{\Omega}}\left(\inf_{y\in A^d}\delta\left(x,y\right)\right)=d,$$

alors on déduit que pour tout $x_i\in F$,

$$\forall d\in D,\exists(x_i^d)\in A^d\text{ tel que}\qquad\lim_{d\to 0}x_i^d=x_i\in F$$

Par conséquent, lorsque $d\leq\eta$, on a

$$card\left(x_i^d\right)_{i=1}^{E(M)+1}=card(F)=E(M)+1$$

et on en déduit que

$$N(d)\geq card\left(x_i^d\right)_{i=1}^{E(M)+1}=E(M)+1>M,$$

ce qui est impossible et conclu la démonstration. □

On rappelle que l'ensemble Z_2^d est défini comme un ensemble ordonné de $p(d)$ éléments distincts (de Z_1^d),

$$z_1^d < z_2^d < \ldots < z_{p(d)}^d.$$

Proposition 7.3.2. *Soit $f \in H^{m'}(\Omega)$ avec $m' > m + \frac{n}{2}$. Alors pour tout $z \in Im(f)$, il existe $(f(x^d) = z^d)_{d\in D}$ tel que*

$$\forall d \in D, \quad z^d \in Z_2^d \text{ et } \lim_{d\to 0} z^d = z. \tag{7.3.2}$$

Démonstration. Soit $z \in Im(f)$. Alors il existe un élément $x \in \overline{\Omega}$ tel que $z = f(x)$. Or, comme

$$\sup_{x\in\overline{\Omega}} \delta\left(x, A^d\right) = d,$$

alors il existe $(x^d)_{d\in D}$ vérifiant

$$\forall d \in D, \quad x^d \in A^d \text{ et } \lim_{d\to 0} x^d = x.$$

Par l'injection continue de $H^{m'}(\Omega)$ dans $C^{0,1}\left(\overline{\Omega}\right)$, nous avons $f \in C^{0,1}(\overline{\Omega})$. Il existe alors une constante $C > 0$ telle que

$$\forall d \in D, \forall x^d \in A^d, \quad \left|z - f(x^d)\right| = \left|f(x) - f(x^d)\right| \leq C \left\|x - x^d\right\|_{\mathbb{R}^n},$$

ce qui prouve le résultat de la proposition. □

Remarque 7.3.1. *Ce résultat signifie que lorsque $d \to 0$, l'ensemble Z_2^d "remplit" de plus en plus l'ensemble $Im(f)$.*

Proposition 7.3.3. *Par définition de l'ensemble Z_2^d, on a*

$$\forall d \in D, \quad card(Z_2^d) = p(d) \leq N(d).$$

Ainsi,

$$card(Z_2^d) = p(d) \underset{d\to 0}{\longrightarrow} +\infty. \tag{7.3.3}$$

Démonstration. Il est clair que les définitions des ensembles Z_1^d et Z_2^d données en (7.2.7) et (7.2.8) impliquent immédiatement que

$$p(d) \leq N(d).$$

De plus, en appliquant le même raisonnement que dans la Proposition 7.3.1, en remplaçant Ω par $Im(f)$ et en utilisant le résultat de la Proposition 7.3.2, il vient que

$$card(Z_2^d) = p(d) \underset{d\to 0}{\longrightarrow} +\infty$$

et le résultat attendu est ainsi démontré. □

Nous rappelons que nous avons fait l'hypothèse suivante

$$f \in H^{m'}(\Omega) \quad \text{avec } m' > m + \frac{n}{2},$$

ce qui implique l'injection continue

$$H^{m'}(\Omega) \hookrightarrow C^{0,1}(\overline{\Omega}).$$

En introduisant la quantité

$$C_f = \sup_{x,y \in \overline{\Omega}, x \neq y} \left(\frac{|f(x) - f(y)|}{\|x - y\|_{\mathbb{R}^n}} \right) \geq 0, \tag{7.3.4}$$

alors on a le lemme suivant.

Lemme 7.3.1. *Sous l'hypothèse (7.2.6) sur d et en considérant la quantité C_f introduite en (7.3.4), alors l'inégalité suivante est vérifiée*

$$\forall d \in D, \quad \sup_{z \in Im(f)} \delta\left(z, Z_2^d\right) \leq C_f \, d. \tag{7.3.5}$$

Démonstration. Soit z un élément quelconque appartenant à l'image de f. Il existe donc un élément $x \in \overline{\Omega}$ tel que $z = f(x)$. D'après (7.2.6), pour tout $d \in D$, il existe $x_0^d \in A^d$ tel que

$$\left| x - x_0^d \right| = \inf_{x^d \in A^d} \left| x - x^d \right| = \delta\left(x, A^d\right) \leq d.$$

De la définition de C_f (7.3.4), on déduit que

$$\forall d \in D, \exists x_0^d \in A^d \text{ tel que } \quad \left| f(x) - f(x_0^d) \right| \leq C_f \, \left\| x - x_0^d \right\|_{\mathbb{R}^n} \leq C_f \, d.$$

Par conséquent,

$$\forall d \in D, \inf_{x^d \in A^d} \left| z - f(x^d) \right| \leq C_f \, d.$$

Cette dernière inégalité étant vérifiée pour tout $z \in Im(f)$, on en déduit que

$$\forall d \in D, \sup_{z \in Im(f)} \left(\inf_{x^d \in A^d} \left| z - f(x^d) \right| \right) \leq C_f \, d$$

et le résultat escompté est ainsi démontré. □

Remarque 7.3.2. *Comme nous allons le voir dans le corollaire suivant, le résultat du Lemme 7.3.1 implique que le plus grand intervalle que l'on puisse inclure dans $Im(f)$ n'intersectant pas l'ensemble Z_2^d est au plus de longueur $2\, C_f \, d$. Ce résultat est similaire à la définition (7.2.6) du paramètre d.*

Dans la suite de cette partie, on suppose que

$$Im(f) = [a, b]$$

où $[a, b]$ est un intervalle inclus dans $\mathbb{R}$, $a \leq b$. On rappelle que les éléments de Z_2^d sont triés par ordre croissant

$$a \leq z_1^d \leq \ldots, z_{p(d)}^d \leq b. \tag{7.3.6}$$

A partir du Lemme 7.3.1 que nous venons de démontrer, on obtient le corollaire suivant.

Corollaire 7.3.1. *Sous les hypothèses du Lemme 7.3.1 et avec le résultat (7.3.6) de la Proposition 7.3.1, il vient que pour tout $d \in D$,*

$$\begin{cases} \left|z_1^d - a\right| \leq C_f\, d, \qquad \left|b - z_{p(d)}^d\right| \leq C_f\, d \\ \text{et } \left|z_{i+1}^d - z_i^d\right| \leq 2\, C_f\, d \text{ pour } 1 \leq i \leq p(d) - 1. \end{cases} \tag{7.3.7}$$

Démonstration. La preuve du corollaire est basée sur un raisonnement par l'absurde. Supposons qu'il existe un $d \in D$ tel que

$$\left| \begin{array}{l} \left|z_1^d - a\right| > C_f\, d \\ \text{ou } \left|b - z_{p(d)}^d\right| > C_f\, d \\ \text{ou } \exists i_0 \in \{1, \ldots, p(d) - 1\} \text{ tel que } \left|z_{i_0+1}^d - z_{i_0}^d\right| > 2\, C_f\, d. \end{array} \right.$$

Cas *i)*. Si $\left|z_1^d - a\right| > C_f\, d$, alors on a d'après la définition (7.3.6) de Z_2^d

$$\inf_{x^d \in A^d} \left|a - f(x^d)\right| = \left|z_1^d - a\right|$$

et donc

$$\inf_{x^d \in A^d} \left|a - f(x^d)\right| > C_f\, d.$$

D'où, puisque $a \in Im(f)$,

$$\sup_{z \in Im(f) = [a,b]} \left(\inf_{x^d \in A^d} \left|z - f(x^d)\right| \right) \geq \inf_{x^d \in A^d} \left|a - f(x^d)\right| > C_f\, d,$$

ce qui contredit le résultat du Lemme 7.3.1.

Cas *ii)*. Lorsque $\left|b - z_{p(d)}^d\right| > C_f\, d$, on procède de la même manière que pour $i)$.

Cas *iii)*. Dans le cas où il existe $i_0 \in \{1, \ldots, p(d) - 1\}$ tel que

$$\left|z_{i_0+1}^d - z_{i_0}^d\right| > 2\, C_f\, d,$$

on s'intéresse au milieu du segment $\left[z_{i_0}^d, z_{i_0+1}^d\right]$,

$$z_{i_0, i_0+1}^d = \frac{z_{i_0+1}^d - z_{i_0}^d}{2}.$$

Ce point appartient à $Im(f) = [a, b]$ et l'on a

$$\inf_{x^d \in A^d} \left|z_{i_0, i_0+1}^d - f(x^d)\right| = \left|z_{i_0+1}^d - z_{i_0, i_0+1}^d\right| = \left|z_{i_0, i_0+1}^d - z_{i_0}^d\right|,$$

ce qui implique que

$$\inf_{x^d \in A^d} \left|z_{i_0, i_0+1}^d - f(x^d)\right| = \frac{z_{i_0+1}^d - z_{i_0}^d}{2} > C_f\, d.$$

On a donc

$$\sup_{z \in Im(f)} \left(\inf_{x^d \in A^d} \left|z - f(x^d)\right| \right) \geq \inf_{x^d \in A^d} \left|z_{i_0, i_0+1}^d - f(x^d)\right| > C_f\, d,$$

ce qui est contraire au résultat du Lemme 7.3.1. □

7.3.2 Construction des changements d'échelle

Dans cette section, nous donnons une présentation classique des familles de changements d'échelle $(\varphi_d)_{d\in D}$ et $(\psi_d)_{d\in D}$ satisfaisant les conditions (7.2.9) et (7.2.10) posées en début de chapitre. Les transformations proposées conservent bien évidemment l'ordre des valeurs comme cela est le cas pour les changements d'échelle classiques.

On suppose qu'il existe un réel $\eta > 0$ tel que pour tout $d \in D$, $d \leq \eta$,

$$a = z_1^d \quad \text{et} \quad b = z_{p(d)}^d. \tag{7.3.8}$$

Nous proposons une construction des familles de changements d'échelle qui utilise à la fois un schéma de différences finies et une base d'éléments finis. D'autres constructions peuvent être imaginées, comme par exemple des fonctions splines sous tension. On rappelle que l'ensemble Z_2^d est ordonné de la manière suivante

$$a = z_1^d < z_2^d < \ldots < z_{p(d)-1}^d < z_{p(d)} = b. \tag{7.3.9}$$

Le choix du changement d'échelle revient à choisir la nouvelle distribution des données. Puisque nous cherchons à atténuer les fortes variations éventuellement présentes dans les données initiales, on considère une subdivision **régulière** $\{u_i\}_{i=1}^{p(d)}$ d'un intervalle $[\alpha, \beta]$ de $\mathbb{R}$ en $p(d) - 1$ intervalles adjacents satisfaisant

$$u_1^d = \alpha \quad \text{et} \quad u_{i+1}^d - u_i^d = \frac{\beta - \alpha}{p(d) - 1}. \tag{7.3.10}$$

On introduit la fonction φ_d suivante définie pour un entier i, $1 \leq i \leq p(d) - 1$ et pour tout $z \in \left[z_i^d, z_{i+1}^d\right]$ par

$$\begin{aligned} \varphi_d(z) = \quad & u_i^d \, q_{0m}^0 \left[\frac{z - z_i^d}{z_{i+1}^d - z_i^d}\right] + u_{i+1}^d \, q_{0m}^1 \left[\frac{z - z_i^d}{z_{i+1}^d - z_i^d}\right] \\ & + \zeta_1\left(z_i^d\right)\left(z_{i+1}^d - z_i^d\right) q_{1m}^0 \left[\frac{z - z_i^d}{z_{i+1}^d - z_i^d}\right] \\ & + \zeta_1\left(z_{i+1}^d\right)\left(z_{i+1}^d - z_i^d\right) q_{1m}^1 \left[\frac{z - z_i^d}{z_{i+1}^d - z_i^d}\right] \end{aligned} \tag{7.3.11}$$

où les fonctions q_{lm}^k désignent pour $k = 0, 1$ et $l = 0, 1$ les fonctions de base de l'élément fini de Hermite et où pour un entier j, $1 \leq j \leq p(d) - 1$,

$$\zeta_1(z_j) = \frac{u_{j+1}^d - u_j^d}{z_{j+1}^d - z_j^d} \quad \text{et} \quad \zeta_1\left(z_{p(d)}^d\right) = \zeta_1\left(z_{p(d)-1}^d\right). \tag{7.3.12}$$

La monotonie des données $\left\{z_i^d\right\}_{i=1}^{p(d)}$ par le changement d'échelle φ_d est vérifiée si l'on fait l'hypothèse suivante

$$\left|\frac{z_{i+1}^d - z_i^d}{z_{i+2}^d - z_{i+1}^d} - 1\right| < \min\left(\frac{1}{C_1}, \frac{1}{C_2}\right) \tag{7.3.13}$$

où

$$C_1 = \left|\inf_{[0,1]} \left(q_{1m}^1\right)'\right| \quad \text{et } C_2 = \sup_{[0,1]} \left(q_{1m}^1\right)' \geq 1. \tag{7.3.14}$$

On peut alors montrer que la transformation φ_d satisfait (7.2.9) avec

$$\varphi(z) = \frac{\beta - \alpha}{b - a}\,(z - a) + \alpha. \tag{7.3.15}$$

De la même manière que nous avons construit le changement d'échelle φ_d, nous définissons une famille ψ_d de changements d'échelle qui sera appliquée en sortie de l'approximation. Soit $[r, s]$ un intervalle de $\mathbb{R}$ et $\{u_i\}_{i=1}^{p(d)}$ la subdivision régulière suivante

$$r = u_1^d < u_2^d < \ldots < u_{p(d)-1}^d < u_{p(d)}^d = s \quad \text{et} \quad u_{i+1}^d - u_i^d = \frac{r - s}{p(d) - 1}. \tag{7.3.16}$$

On définit la fonction ψ_d pour un entier i, $1 \leq i \leq p(d) - 1$ et pour tout $u \in \left[u_i^d, u_{i+1}^d\right]$ par

$$\begin{aligned} \psi_d(u) = \quad & z_i^d\, q_{0m}^0 \left[\frac{u - u_i^d}{u_{i+1}^d - u_i^d}\right] + z_{i+1}^d\, q_{0m}^1 \left[\frac{u - u_i^d}{u_{i+1}^d - u_i^d}\right] \\ & + \left(u_{i+1}^d - u_i^d\right)\, \xi_1\left(u_i^d\right)\, q_{1m}^0 \left[\frac{u - u_i^d}{u_{i+1}^d - u_i^d}\right] \\ & + \left(u_{i+1}^d - u_i^d\right)\, \xi_1\left(u_{i+1}^d\right)\, q_{1m}^1 \left[\frac{u - u_i^d}{u_{i+1}^d - u_i^d}\right] \end{aligned} \tag{7.3.17}$$

où les fonctions q_{lm}^k désignent pour $k = 0, 1$ et $l = 0, 1$ les fonctions de base de l'élément fini de Hermite et où pour un entier j, $1 \leq j \leq p(d) - 1$,

$$\xi_1\left(u_j^d\right) = \frac{z_{j+1}^d - z_j^d}{u_{j+1}^d - u_j^d} \quad \text{et} \quad \xi_1\left(u_{p(d)}^d\right) = \xi_1\left(u_{p(d)-1}^d\right). \tag{7.3.18}$$

On peut alors facilement montrer que la transformation ψ_d vérifie les hypothèses (7.2.10) et que

$$\psi = \varphi^{-1}. \tag{7.3.19}$$

Les familles $(\varphi_d)_{d \in D}$ et $(\psi_d)_{d \in D}$ définies ci-dessus vérifient les hypothèses (7.2.9) et (7.2.10). Ces hypothèses seront utiles lors de l'étude de la convergence de la méthode d'approximation combinée avec les changements d'échelle. En pratique, seules les transformations φ et ψ sont utilisées pour ré-échelonner les données.

7.4 D^m-spline d'interpolation

Contrairement aux travaux d'Apprato et Gout [11], on choisit d'utiliser dans ce travail une spline d'interpolation au lieu d'une spline d'ajustement. D'une part, la surface passera exactement par les données valides issues de la carte de profondeur. D'autre part, le nombre modéré de points de données ne justifie pas l'utilisation d'une spline d'ajustement moins coûteuse en temps de calcul (contrairement aux applications géophysique par exemple).

La théorie des splines est due au travaux de Schoenberg [171]. En dimension 1, une spline est une fonction polynomiale par morceaux dont les dérivées sont continues jusqu'à l'ordre m. En dimension 1, la même fonction peut être obtenue par minimisation d'une fonctionnelle comme $\int_a^b \left(f^{(m)}(t)\right)^2 \mathrm{d}t$ avec des contraintes d'interpolation de type Lagrange ou Hermite. En dimension supérieure (2 ou plus), deux approches différentes peuvent être considérées.

La première approche (appelée interpolation locale) nécessite de construire une triangulation sur le domaine de $\mathbb{R}^n$ considéré. Les fonctions polynomiales sont ensuite calculées sur chaque n-simplexe, avec des conditions aux bords (pour assurer le critère de régularité souhaité). Le problème de cette approche est de traiter ces conditions aux bords sur chaque simplexe. Le choix de la triangulation du domaine se pose également. La fonction spline obtenue ne satisfait pas de critères de minimisation. Pour plus de détails, voir de Boor [65].

La seconde approche (appelée interpolation globale) est basée sur la minimisation d'un critère sous des contraintes d'interpolation de type Lagrange ou Hermite. Pour s'assurer de l'existence, de l'unicité et de la régularité de la fonction spline, on la choisi dans un espace de Hilbert et on cherche à minimiser sa norme dans cet espace. Cette méthode utilise les espaces de Hilbert et les propriétés des noyaux reproduisants (voir N. Aronsajn [14]). L'inconvénient de cette approche est d'obtenir une caractérisation de la spline, aboutissant à la résolution d'une équation aux dérivées partielles difficile.

Dans ce travail, on utilise les D^m-splines d'interpolation [13] présentées au Chapitre 6 dont nous rappelons ici les principales caractéristiques. Les D^m-splines d'interpolation discrétisées sur un espace de type éléments finis comportent de nombreux avantages : la mise en place d'un raffinement local dans le maillage est envisageable, la matrice du système linéaire obtenu est creuse et la convergence de l'approximation peut être étudiée. Dans certaines applications, les splines d'ajustement sont parfois préférées aux splines d'interpolation pour des raisons de coût calculatoire. Par exemple, dans le domaine de la géophysique, la quantité de données est très importante, ce qui incite à choisir des splines d'ajustement.

Étant donné un ensemble de données de Lagrange $(x_i, (\varphi_d \circ f)(x_i))$, on s'intéresse au problème classique de construction d'un approximant de classe C^k. Pour tout $d \in D$, on considère :

- un opérateur $\rho^d \in L\left(H^m(\Omega), \mathbb{R}^{N(d)}\right)$ défini par
$$\forall v \in H^m(\Omega), \rho^d v = (v(a))_{a \in A^d} \tag{7.4.1}$$
associant à une fonction v sa valeur en chaque point de A^d.
- l'ensemble
$$K^d = \left\{u \in H^m(\Omega), \rho^d u = \rho^d(\varphi_d \circ f)\right\} \tag{7.4.2}$$
où φ_d est le changement d'échelle introduit précédemment.
- le sous espace vectoriel
$$K_0^d = \left\{u \in H^m(\Omega), \rho^d u = 0\right\}. \tag{7.4.3}$$

On considère alors le problème de minimisation suivant

$$\begin{cases} \text{Trouver } \sigma^d(\varphi_d \circ f) \in K_d \text{ tel que } \forall v \in K^d, \\ |\sigma^d(\varphi_d \circ f)|_{m,\Omega} \leq |v|_{m,\Omega}. \end{cases} \tag{7.4.4}$$

On peut montrer que le problème (7.4.4) admet une solution unique appelée spline d'interpolation sur Ω associée à A^d.

7.5 Expérimentations

Dans cette section, la méthode de restauration d'images de profondeur est expérimentée sur des données réelles provenant de capteurs 3D à lumière structurée de type Kinect. Le capteur 3D est fixé au plafond à une hauteur de Z_{sol} mm et orienté verticalement. Les données représentent une personne en position verticale dans une pièce en vue de dessus. L'image de profondeur, notée I_D, est de largeur $L = 48$ pixels et de hauteur $H = 81$ pixels. L'installation utilisée pour obtenir les images est identique à celle décrite au Chapitre 4. Le schéma représentant le système est rappelé Figure 7.5.1.

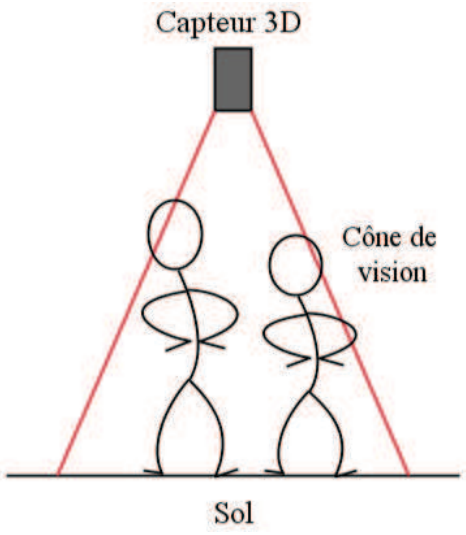


FIGURE 7.5.1 – Système d'acquisition utilisé lors des expérimentations.

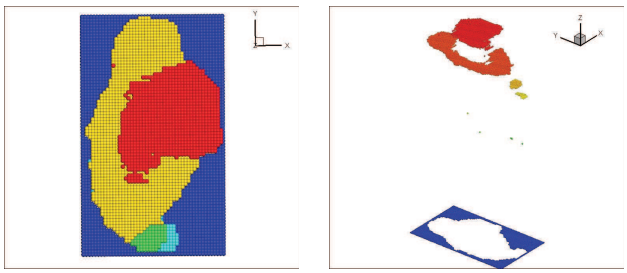
Dans ces expérimentations, la fonction inconnue f à approcher, définie sur $[1, L] \times [1, H]$, correspond à la surface modélisant la personne et le sol de la pièce. Les données (de Lagrange) sont composées des pixels $x_i \in [1, L] \times [1, H]$ et des côtes $f(x_i) = Z_{sol} - I_D(x_i)$. Les valeurs des côtes sont proches des hauteurs des points par rapport au sol. L'image de profondeur et le nuage de points 3D considérés pour tester les performances de la méthode sont représentés Figure 7.5.2. Comme mentionné dans l'introduction, les données présentent des fortes variations. En effet, un faible déplacement $\Delta(xy)$ sur l'image (ou de manière équivalente sur le sol, parallèle au plan image) peut conduire à une variation brutale de la hauteur $\Delta(z)$ par rapport au sol.

Les paramètres α et β ont été respectivement fixés à 0 et 1. L'opérateur d'approximation T^d utilisé est l'opérateur D^m-spline d'interpolation noté σ_h^d. Le plan (Oxy) de l'image est discrétisé par des éléments finis rectangulaires de type Bogner-Fox-Schmit de classe C^1 présentés au Chapitre 6. Le nombre de rectangles a été fixé à 15 en largeur et 25 en hauteur. Notons que l'approximant obtenu, exprimé dans la base d'éléments finis, peut être évalué sur une grille de précision choisie par l'utilisateur. Les données peuvent ainsi être très facilement ré-échantillonnées. Dans nos tests, la grille d'évaluation, donc l'image restaurée, a une taille de 49 en largeur et 82 en hauteur, soit une résolution légèrement supérieure à la résolution originale. Si cela avait été nécessaire, il aurait été facile d'évaluer, par exemple, l'approximant sur une grille de dimensions 490 par 820, soit cent fois plus de noeuds. L'augmentation de la résolution de l'image est très simple à mettre en œuvre.

La première expérimentation consiste à supprimer manuellement une partie des données. Pour cela, une bande transversale est définie et les points y appartenant sont supprimés. L'image de profondeur détériorée est représentée Figure 7.5.3. Sur cet exemple, 463 pixels ont été altérés. Dans ces expérimentations, nous avons choisi de supprimer manuellement une partie des données. En effet, la connaissance des valeurs de l'image non détériorée nous permettra par la suite de quantifier la différence entre l'image initiale et l'image restaurée. On calcule pour cela l'erreur quadratique suivante

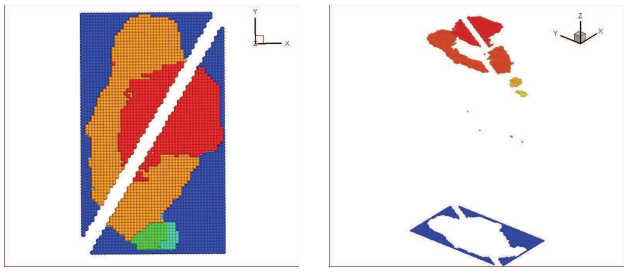
$$E = \left(\frac{\sum_{i=1}^{L \times H} (\tilde{z}_i - z_i)^2}{\sum_{i=1}^{L \times H} z_i^2} \right)^{1/2} \tag{7.5.1}$$

où les $(z_i)_{i=1}^{L \times H}$ sont les valeurs des pixels de l'image non détériorée et les $(\tilde{z}_i)_{i=1}^{L \times H}$ sont les valeurs restaurées (évaluées sur la grille initiale).



(a) (b)

FIGURE 7.5.2 – Image de profondeur (données de Lagrange) représentant une personne en position verticale dans une pièce en (a) et représentation 3D en (b). Le nombre de points s'élève à 3888.

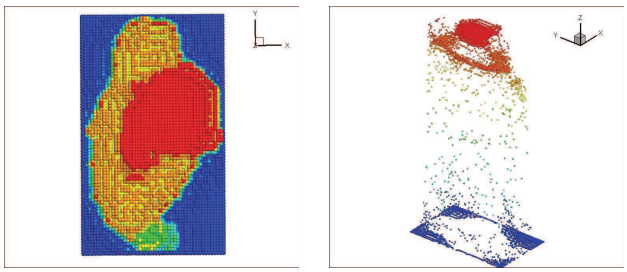


(a) (b)

FIGURE 7.5.3 – Image de profondeur présentant des données manquantes en (a) et représentation 3D en (b). Le nombre de points s'élève à 3425.

Les résultats obtenus sont représentés Figure 7.5.4. On observe que les données manquantes ont été correctement restaurées. La représentation 3D de l'image reconstruite révèle que des points sont ajoutés entre le haut du corps et le sol. Ceci est dû au fait que la grille d'évaluation choisie pour construire l'image est plus fine que celle de l'image initiale, afin d'obtenir une résolution plus élevée. Il serait aisément possible, moyennant un coût calculatoire négligeable, d'évaluer la surface sur une

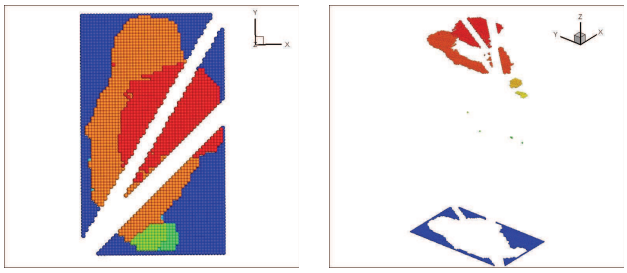
grille beaucoup plus fine, si un post-traitement le justifie. L'erreur mesurée lors de cette première expérimentation s'élève à $E = 7,9\,10^{-4}$, ce qui est un très bon résultat dans ce contexte.



(a) (b)

FIGURE 7.5.4 – En (a) l'image de profondeur restaurée et en (b) le nuage de points 3D correspondant.

La seconde expérimentation est basée sur le même jeu de données auquel une seconde bande de pixels a été supprimée. Le nombre de pixels indéterminés s'élève cette fois ci à 832, soit près de deux fois plus que dans la première expérimentation. L'image de profondeur et le nuage de points 3D dégradés sont représentés Figure 7.5.5.

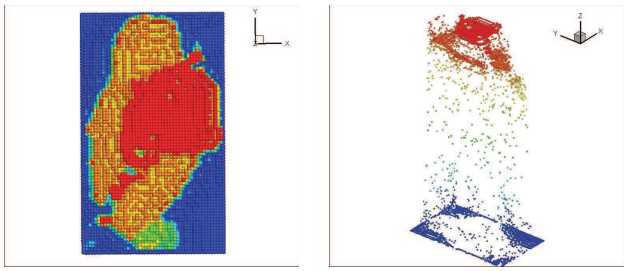


(a) (b)

FIGURE 7.5.5 – Image de profondeur présentant des données manquantes en (a) et représentation 3D en (b). Le nombre de points s'élève à 3056.

Les résultats obtenus sont représentés Figure 7.5.6. Les paramètres ont été choisis comme dans la première expérimentation. Malgré la proportion de données manquantes plus élevée, on observe que les zones dégradées ont bien été restaurées. L'erreur mesurée lors de cette seconde expérimentation s'élève à $E = 4,6\,10^{-3}$. Les erreurs de reconstruction constatées dans ces deux expérimentations sont suffisamment faibles pour pouvoir appliquer des algorithmes (de détection et de suivi d'objets par exemple) sur les images pré-traitées.

L'algorithme a été implémenté en Fortran et en C sur une machine équipée d'un processeur Intel Core i7 avec 8 Go de RAM. Le temps de calcul de la méthode est de l'ordre de 30 secondes, ce qui n'est clairement pas compatible avec un fonctionnement en temps-réel. L'étape d'évaluation de l'approximant n'est pas la plus coûteuse si la résolution de la grille reste raisonnable. Par conséquent, un effort d'optimisation ou de parallélisation de la méthode d'approximation permettrait de réduire le coût calculatoire.



(a) (b)

FIGURE 7.5.6 – En (a) l'image de profondeur restaurée et en (b) le nuage de points 3D correspondant.

7.6 Conclusion

Dans cette partie, nous nous sommes penchés sur le problème de la restauration d'images de profondeur. Les images de profondeur dégradées, dont une partie des données sont manquantes, ont (généralement) la caractéristique de présenter des fortes variations. Pour cette raison, nous avons appliqué une méthode d'approximation de surface adaptée faisant intervenir deux changements d'échelle. Nous avons étudié la convergence de la méthode complète d'approximation dans le cas d'un opérateur D^m-spline d'interpolation alors que les travaux initiaux utilisaient une spline d'ajustement, l'avantage étant que les démonstrations sont analogues. L'originalité de ce chapitre réside d'une part dans le choix d'une telle approche pour la restauration d'images et d'autre part dans le choix de l'opérateur.

Les expérimentations ont montré que la méthode permettait de reconstruire assez efficacement les images dégradées. L'erreur de reconstruction constatée est suffisamment faible pour pouvoir appliquer des algorithmes complexes sur les images restaurées. De plus, puisque les données sont approchées par une surface, il est très facile de ré-échantillonner le signal pour obtenir une image de résolution supérieure. En effet, à partir du moment où la solution est connue (via ses coefficients dans la base d'éléments finis, obtenus par résolution du système linéaire), le coût calculatoire pour évaluer la surface solution est négligeable, y compris si l'on choisit une grille d'évaluation de plusieurs millions de noeuds.

Cependant, le temps de calcul de cette méthode est trop élevé pour un fonctionnement en temps-réel. Une perspective de poursuite des travaux serait donc de mettre au point une méthode moins coûteuse, notamment grâce à un code de calcul parallélisé. De même, il eut été souhaitable de définir des maillages éléments finis prenant en compte les fortes variations, détectées par exemple lors d'une segmentation préalable. Le maillage pourrait donc être localement raffiné sur certaines zones d'intérêt. Un travail supplémentaire consistant à faire varier les différents paramètres (changements d'échelle choisis, type d'éléments finis) permettrait de mieux étudier leur influence. En conclusion, cette partie donne de bonnes bases afin d'initier une approche plus aboutie permettant la restauration de cartes de profondeurs.

Annexe

A Convergence de la méthode d'approximation

Dans cette annexe, nous démontrons le résultat de convergence de la méthode d'approximation (Théorème 7.2.1) faisant intervenir les changements d'échelle introduits en section 7.3 et l'opérateur D^m-spline d'interpolation introduit en section 7.4. Le théorème est rappelé ci-dessous.

Théorème A.1. *Sous les hypothèses (7.2.9), (7.2.10) et (7.2.11), on a*

$$\lim_{d\to 0}\left\{\psi_d \circ \sigma^d(\varphi_d \circ f)\right\} = (\psi \circ \varphi \circ f) \in H^{r-\theta}(\Omega) \tag{A1}$$

pour tout $r \in \mathbb{N}$ *tel que*

$$r < m - \frac{n}{2} \qquad (\Rightarrow H^m(\Omega) \hookrightarrow C^r(\overline{\Omega}))$$

et pour tout $\theta > 0$ *tel que*

$$\theta < r - \frac{n}{2} \qquad (\Rightarrow H^{r-\theta}(\Omega) \hookrightarrow C^0(\overline{\Omega})).$$

La démonstration de ce théorème est analogue à celle d'Apprato et Gout [11], la différence étant que l'on travaille ici avec une spline d'interpolation σ^d. La preuve est divisée en trois étapes.

Étape 1. Tout d'abord, on sait que pour tout $d \in D$, $\psi_d \in C^m([r,s])$ et

$$\left(\sigma^d(\varphi_d \circ f)\right) \in H^m(\Omega,[r,s]) \hookrightarrow C^r\left(\overline{\Omega},[r,s]\right) \tag{A2}$$

du fait de l'hypothèse (7.2.11)-(i) et du fait que

$$(\varphi_d \circ f) \in C^m\left(\overline{\Omega},[\alpha,\beta]\right) \subset H^m(\Omega,[\alpha,\beta]). \tag{A3}$$

Par conséquent, on en déduit que pour tout $d \in D$,

$$\psi_d \circ \sigma^d(\varphi_d \circ f) \in C^r\left(\overline{\Omega}\right). \tag{A4}$$

Dans la suite, par soucis de concision, on posera $R_d = \psi_d \circ \sigma^d(\varphi_d \circ f)$.

D'autre part, on a aussi : $\forall d \in D, \forall l = 1,\ldots,r, \forall x \in \overline{\Omega}, \forall \zeta \in \Omega$,

$$\begin{aligned} D^l R_d(x)\cdot(\zeta)^l = \quad & l!\sum_{j=1}^{l}\sum_{s\in S(j,l)}\frac{1}{j!}D^j R_d(x)\cdot \\ & \left(\frac{1}{s_1!}D^{s_1}(\sigma^d(\varphi_d\circ f))(x)\cdot(\zeta)^{s_1},\ldots,\frac{1}{s_j!}D^{s_j}(\sigma^d(\varphi_d\circ f))(x)\cdot(\zeta)^{s_j}\right) \end{aligned} \tag{A5}$$

où l'on a posé

$$S(j,l) = \left\{s = (s_1,\ldots,s_j) \in \mathbb{N}^j, 1 \le s_1,\ldots,s_j \le l, s_1+\ldots+s_j = l\right\}. \tag{A6}$$

Donc, en appliquant le résultat de P.G. Ciarlet et P.A. Raviart [57] rappelé au Chapitre 6, on obtient : $\forall d \in D, \forall l = 1,\ldots,r, \forall x \in \overline{\Omega}$,

$$\left\|D^l R_d(x)\right\| \le \alpha \sum_{j=1}^{l}\left\|D^j R_d(x)\right\| \sum_{i\in I(j,l)}\left\|D\,\sigma^d(\varphi_d\circ f)(x)\right\|^{i_1}\ldots\left\|D^l\,\sigma^d(\varphi_d\circ f)(x)\right\|^{i_l} \tag{A7}$$

avec $\alpha = \alpha(l)$ et

$$I(j,l) = \left\{ i = (i_1, \ldots, i_j) \in \mathbb{N}^j, i_1 + \ldots + i_l = j,\ i_1 + 2\, i_2 + \ldots + l\, i_l = l \right\}. \tag{A8}$$

Par conséquent, il vient la majoration suivante

$$\left\| D^l\, R_d(x) \right\| \leq \alpha \sum_{j=1}^{l} \eta_{jl}\ \left\| D^j R_d(x) \right\| \tag{A9}$$

avec

$$\eta_{jl} = \sum_{i \in I(j,l)} \left\| \sigma^d(\varphi_d \circ f) \right\|_{1,\infty,\Omega}^{i_1} \ldots \left\| \sigma^d(\varphi_d \circ f) \right\|_{l,\infty,\Omega}^{i_l}. \tag{A10}$$

En utilisant l'injection continue

$$\sigma^d\left(\varphi_d \circ f\right) \in H^m\left(\Omega\right) \hookrightarrow C^r(\overline{\Omega}), \tag{A11}$$

on obtient la majoration suivante

$$\forall d \in D, \forall l = 1, \ldots, r, \quad \left\| D^l\, R_d(x) \right\| \leq \alpha \sum_{j=1}^{l} \eta_{jl}\ \left\| \psi_d \right\|_{C^r([r,s])}. \tag{A12}$$

De plus, la famille $\left(\sigma^d\left(\varphi_s \circ f\right)\right)_{d \in D}$ est bornée dans $H^m\left(\Omega\right)$ indépendamment de d, donc dans $C^r\left(\overline{\Omega}\right)$. Il existe alors une constante $C_1 > 0$ telle que

$$\eta_{jl} \leq card\left(I(j,l)\right)\ C_1^j\ \left\| \sigma^d(\varphi_d \circ f) \right\|_{C^r(\overline{\Omega})}^{j} \tag{A13}$$

et on en déduit qu'il existe une constante $C_1^j > 0$ telle que pour tout entier j, $1 \leq j \leq l$, pour tout $d \in D$ et pour tout entier l, $1 \leq l \leq r$,

$$\left\| D^l\, R_d(x) \right\| \leq \alpha \left(\sum_{j=1}^{l} card\left(I(j,l)\right)\ C_1^j\, C^j \right)\ \left\| \psi_d \right\|_{C^r([r,s])}. \tag{A14}$$

Ensuite, l'hypothèse (7.2.10) nous assure que la famille $(\psi_d)_{d \in D}$ est bornée dans $C^m\left([r,s]\right)$ ($\hookrightarrow C^r\left([r,s]\right)$). Il existe donc une constante $C > 0$ telle que pour tout entier l, $1 \leq l \leq r$, et pour tout $d \in D$, on a

$$\left\| D^l\, R_d(x) \right\| \leq C. \tag{A15}$$

De plus, les hypothèses (7.2.9), (7.2.10) et (7.2.11)-(ii) impliquent que il existe une constante $C > 0$ telle que pour tout $d \in D$,

$$\left\| R_d(x) \right\|_{C^0\left(\overline{\Omega}\right)} \leq C.$$

La famille $\left(\psi_d \circ \sigma^d\left(\varphi_d \circ f\right)\right)_{d \in D}$ est don bornée dans $C^r\left(\overline{\Omega}\right)$, donc dans $H^r\left(\Omega\right)$.

On en déduit qu'il existe une sous-suite extraite notée $\left\{\psi_{d_n} \circ \sigma^{d_n}\left(\varphi_{d_n} \circ f\right)\right\}_{n \in \mathbb{N}}$ et une fonction $g \in H^r\left(\Omega\right)$ telles que

$$\lim_{d \to 0} d_n = 0 \quad \text{et} \quad \lim_{d \to 0} \left(\psi_{d_n} \circ \sigma^{d_n}\left(\varphi_{d_n} \circ f\right) \right) = g \quad \text{dans } H^{r-\theta}\left(\Omega\right) \tag{A16}$$

pour tout $\theta > 0$ suffisamment petit pour que $H^r(\Omega) \subset_c H^{r-\theta}(\Omega)$.

Étape 2. La deuxième étape de la démonstration consiste à montrer que toute sous-suite extraite de la famille $\left(\psi_d \circ \sigma^d(\varphi_d \circ f)\right)_{d\in D}$, convergeant fortement dans $H^{r-\theta}(\Omega)$, converge vers la fonction $\psi \circ (\varphi \circ f)$.

Soit $\left\{\psi_{d_n^\star} \circ \sigma^{d_n^\star}\left(\varphi_{d_n^\star} \circ f\right)\right\}_{n\in\mathbb{N}}$ une suite extraite de $\left(\psi_d \circ \sigma^d(\varphi_d \circ f)\right)_{d\in D}$ telle que

$$\lim_{n\to+\infty} d_n^\star = 0 \quad \text{et} \quad \lim_{n\to+\infty} \left(\psi_{d_n^\star} \circ \sigma^{d_n^\star}\left(\varphi_{d_n^\star} \circ f\right)\right) \text{ existe dans } H^{r-\theta}(\Omega). \tag{A17}$$

On a alors

$$\lim_{n\to+\infty} \left(\psi_{d_n^\star} \circ \sigma^{d_n^\star}\left(\varphi_{d_n^\star} \circ f\right)\right) = \psi \circ (\varphi \circ f). \tag{A18}$$

En effet, soit x un point fixé mais quelconque de Ω. On a pour tout $n \in \mathbb{N}$,

$$\begin{aligned}\psi_{d_n^\star}\left[\sigma^{d_n^\star}(\varphi_{d_n^\star} \circ f)(x)\right] - \psi\left[\varphi(f(x))\right] = \quad & \psi_{d_n^\star}\left[\left(\sigma^{d_n^\star}\left(\varphi_{d_n^\star} \circ f\right)\right)(x)\right] - \psi_{d_n^\star}\left[\varphi(f(x))\right] \\ & + \psi_{d_n^\star}\left[\varphi(f(x))\right] - \psi\left[\varphi(f(x))\right].\end{aligned} \tag{A19}$$

On obtient alors pour tout $n \in \mathbb{N}$,

$$\begin{aligned}\left|\psi_{d_n^\star}\left[\left(\sigma^{d_n^\star}(\varphi_{d_n^\star} \circ f)\right)(x)\right] - \psi\left[\varphi(f(x))\right]\right| \leq \quad & \left|\psi_{d_n^\star}\left[\left(\sigma^{d_n^\star}\left(\varphi_{d_n^\star} \circ f\right)\right)(x)\right] - \psi_{d_n^\star}\left[\varphi(f(x))\right]\right| \\ & + \left|\psi_{d_n^\star}\left[\varphi(f(x))\right] - \psi\left[\varphi(f(x))\right]\right|.\end{aligned} \tag{A20}$$

Comme la famille $(\psi_d)_{d\in D}$ est dans $C^m([r,s])$ elle est également dans $H^m([r,s]) \hookrightarrow C^{0,\alpha}([r,s])$ puisque $m \geq 2$. Donc il existe une constante $C > 0$ et un réel $\alpha \in]0,1]$ tel que pour tout $n \in \mathbb{N}$,

$$\left|\psi_{d_n^\star}\left[\sigma^{d_n^\star}(\varphi_{d_n^\star} \circ f)(x)\right] - \psi_{d_n^\star}\left[\varphi(f(x))\right]\right| \leq C\left|\sigma^{d_n^\star}(\varphi_{d_n^\star} \circ f)(x) - \varphi(f(x))\right|^\alpha \left\|\psi_{d_n^\star}\right\|_{C^m([r,s])}. \tag{A21}$$

Puisque l'opérateur σ^d satisfait l'hypothèse 7.2.11-(iii), on a

$$\lim_{n\to+\infty} \left(\sigma^{d_n^\star}(\varphi_d \circ f)\right)(x) = \varphi \circ f(x) \text{ pour tout } x \in \Omega \tag{A22}$$

car

$$H^m(\Omega, [\alpha,\beta]) \hookrightarrow C^0\left(\overline{\Omega}, [\alpha,\beta]\right) \tag{A23}$$

et on sait également que $(\psi_d)_{d\in D}$ est bornée dans $C^m([r,s])$ donc dans $H^m([r,s])$. Par conséquent, on a

$$\lim_{n\to+\infty} \left|\psi_{d_n^\star}\left[\left(\sigma^{d_n^\star}(\varphi_{d_n^\star} \circ f)\right)(x)\right] - \psi_{d_n^\star}\left[\varphi \circ f(x)\right]\right| = 0 \tag{A24}$$

puisque

$$\lim_{n\to+\infty} d_n^\star = 0. \tag{A25}$$

En plus de cela, l'hypothèse (7.2.10) implique que

$$\lim_{d\to 0} \psi_d = \psi \text{ dans } C^m([\alpha,\beta]) \tag{A26}$$

donc dans $C^0([\alpha,\beta])$. Ainsi,

$$\lim_{n\to+\infty} \left|\psi_{d_n^\star}\left[\varphi \circ f(x)\right] - \psi\left[\varphi \circ f(x)\right]\right| = 0 \tag{A27}$$

puisque

$$\lim_{n\to+\infty} d_n^\star = 0. \tag{A28}$$

D'où

$$\lim_{n\to+\infty} \left|\psi_{d_n^\star}\left[\sigma^{d_n^\star}\left(\varphi_{d_n^\star}\circ f\right)\right](x) - \psi\left[\varphi\circ f(x)\right]\right| = 0. \tag{A29}$$

Finalement, pour tout $\theta > 0$ tel que $H^{r-\theta}(\Omega) \hookrightarrow C^0\left(\overline{\Omega}\right)$, on obtient par des arguments de continuité que

$$\lim_{n\to+\infty} \left(\psi_{d_n^\star}\circ\sigma^{d_n^\star}\left(\varphi_{d_n^\star}\circ f\right)\right) = \psi\circ(\varphi\circ f) \text{ dans } \Omega. \tag{A30}$$

Étape 3. Dans cette dernière étape, on montre que la famille $\left(\psi_d\circ\sigma^d\left(\varphi_d\circ f\right)\right)_{d\in D}$ converge vers $\psi\circ(\varphi\circ f)$ dans $H^{r-\theta}(\Omega)$ lorsque $d\to 0$.

Pour montrer cela, on raisonne par l'absurde. On suppose qu'il existe $\eta > 0$ tel que pour tout $\mu > 0$, il existe $d \in D$ avec $d \le \mu$, tel que

$$\left\|\psi_d\circ\sigma^d\left(\varphi_d\circ f\right) - \psi\circ(\varphi\circ f)\right\|_{H^{r-\theta}(\Omega)} > \eta.$$

En particulier, il existe $\eta > 0$ tel que pour tout $n \in \mathbb{N}$, il existe $d_n^{\star\star} \in D$, $d_n^{\star\star} \le n^{-1}$ vérifiant

$$\left\|\psi_{d_n^{\star\star}}\circ\sigma^d\left(\varphi_{d_n^{\star\star}}\circ f\right) - \psi\circ(\varphi\circ f)\right\|_{H^{r-\theta}(\Omega)} > \eta. \tag{A31}$$

Mais la séquence $\psi_{d_n^{\star\star}}\circ\sigma^d\left(\varphi_{d_n^{\star\star}}\circ f\right)_{n\in\mathbb{N}}$ est borné dans $H^{r-\theta}(\Omega)$ car la famille $\left(\psi_d\circ\sigma^d\left(\varphi_d\circ f\right)\right)_{d\in D}$ est également bornée dans $H^{r-\theta}(\Omega)$ et que $\lim_{d\to 0} d_n^{\star\star} = 0$.
Donc en raisonnant comme dans l'étape 1, on peut extraire une sous-suite qui, selon l'étape 2, est fortement convergente vers $\psi\circ(\varphi\circ f)$. La propriété (A31) est alors contredite.

Pour conclure, on utilise (7.3.19) ce qui donne

$$\psi\circ\varphi\circ f = f. \tag{A32}$$

Conclusion et perspectives

Les travaux présentés dans cette thèse s'articulent autour des problématiques de détection de personnes dans des données 3D et de restauration de cartes de profondeur dégradées. Nous avons commencé par nous intéresser aux méthodes référencées dans la littérature avant de nous concentrer sur des améliorations possibles à leur apporter. Nous nous sommes penchés dans un premier temps sur le traitement direct des images de profondeur. Dans un second temps, nous avons préféré travailler sur les nuages de points 3D représentant la scène. L'application directe à des problématiques industrielles concrètes des algorithmes développés a été une source majeure d'avancement des travaux, dont les principales contributions sont ici résumées.

Notre première contribution concerne l'amélioration d'une méthode existante de détection et de suivi de personnes dans une séquence d'images de profondeur. Les modifications apportées permettent de détecter les personnes partiellement masquées par les bords de l'image ou par leur trop grande proximité avec le capteur 3D. Les tests effectués traduisent le bon comportement de la méthode. Ce travail a donné lieu à une publication dans *IEEE Euvip (2014)*.
La deuxième contribution, la principale de ce manuscrit, est l'introduction d'un nouveau modèle de mélange sphérique basé sur une nouvelle densité de probabilité. Ce modèle permet de décrire des points répartis autour de surfaces sphériques. Nous avons proposé une méthode d'estimation des différents paramètres de ce modèle. La méthode a été appliquée avec succès au problème de détection de têtes dans un nuage de points 3D. Elle a ensuite été optimisée pour réduire son coût calculatoire et fonctionner en quasi temps-réel. Nous avons montré que pour ce problème, notre méthode était plus efficace que l'algorithme RANSAC, très utilisé dans la littérature. Ce travail a donné lieu à une publication dans *SIAM Journal on Imaging Sciences (2014)* et à deux communications aux *Journées de Statistique (2014)* et à la conférence internationale *Curves and Surfaces (2014)*.
Enfin, la dernière contribution de ces travaux est l'application d'un algorithme d'approximation de surfaces pour restaurer et éventuellement ré-échantillonner des images de profondeur dégradées. La méthode retenue fait intervenir des changements d'échelle pour atténuer les fortes variations présentes dans les données. Les tests effectués traduisent le bon comportement de la méthode. Ce travail a donné lieu à une publication dans *IEEE Euvip (2014)* et à une autre publication en cours de soumission dans une revue.

Finalement, les expérimentations effectuées, notamment celles présentées dans ce manuscrit, montrent le bon fonctionnement global des algorithmes développés.

Pour conclure, les différents objectifs fixés en début de thèse ont été remplis, que ce soit du point de vue académique ou industriel. En effet, chacune des parties de ce manuscrit a fait l'objet de publications ou communications scientifiques. De plus, une partie des algorithmes développés dans ces travaux ont été intégrés au logiciel propriétaire de la société partenaire et expérimentés sur site en conditions réelles. Ces travaux ont montré que les capteurs 3D constituent un outil capable de solutionner

des problèmes complexes difficiles à traiter avec des caméras classiques. Ces caméras particulières ont donc toute leur place dans le panel des équipements utilisés dans le domaine de la vidéo-surveillance.

Les travaux exposés dans ce manuscrit offrent des perspectives de recherche intéressantes. Nous pensons en particulier à la généralisation des modèles de mélange introduits dans la deuxième partie. De part le problème de détection de têtes, nous nous sommes naturellement intéressés à des densités décrivant des objets de forme sphérique. Les techniques d'estimation des paramètres proposées pourraient certainement être améliorées en adoptant d'autres approches (que le backfitting). Il se trouve que la sphère est un objet 3D simple comportant assez peu de paramètres en comparaison des autres surfaces. On pourrait imaginer exploiter la même idée pour modéliser des surfaces plus complexes. Une première généralisation à l'ellipsoïde a été explorée, mais un travail de recherche important est encore nécessaire pour construire un algorithme robuste. Nous nous sommes aperçu à cette occasion qu'il n'est pas toujours simple d'exprimer de manière explicite la distance à la forme géométrique étudiée. Le problème du suivi temporel des têtes n'a pas été considéré dans cette partie. La connaissance de la position et du déplacement des têtes aux instants précédents sont des éléments qui pourraient guider l'initialisation du modèle de mélange. Enfin, d'un point de vue applicatif, il est clair que la fusion des données provenant de plusieurs capteurs 3D constitue une suite logique aux travaux présentés. Plus généralement, l'intégration de l'information de couleur, ignorée dans ce manuscrit, pourrait aussi améliorer les différentes méthodes.

De manière analogue, comme mentionné dans la conclusion de la partie 3, l'algorithme de restauration d'images de profondeur mériterait des améliorations. Le maillage pourrait être construit de manière adaptative grâce à une segmentation préalable des données. Un raffinement local pourrait être effectué et une réduction du coût calculatoire obtenue par une parallélisation de la méthode. Ces pistes prometteuses seraient susceptibles de mener à une méthode de restauration et d'augmentation de la résolution des données.

Nous pensons que tous ces éléments constituent des fils conducteurs pour des recherches futures.

Liste des publications et communications

Publications

[P1] D. Brazey and C. Gout, 3D sensor depth map restoration using smoothing spline aproximation : theory and applications, *submitted*, 2014.

[P2] D. Brazey and C. Gout, An algorithm for automatic people tracking from depth map sequences, *5th European Workshop on Visual Information Processing (IEEE EUVIP)*, pp. 1-6, 2014.

[P3] D. Brazey and C. Gout, Kinect depth map inpainting using spline approximation, *5th European Workshop on Visual Information Processing (IEEE EUVIP)*, pp. 1-6, 2014.

[P4] D. Brazey and B. Portier, A new spherical mixture model for head detection in depth images, *SIAM Journal on Imaging Sciences*, 7(4), pp. 2423-2447, 2014.

Communications

[C1] D. Brazey, T. Chalumeau, B. Portier, C. Gout and N. Fortier, 3D head detection from unorganized point cloud using a spherical mixture model, *8th International Conference on Curves and Surfaces*, ENSAM Paris, June 2014.

[C2] D. Brazey et B. Portier, Détection de têtes dans un nuage de points 3D à l'aide d'un modèle de mélange sphérique, *46ième Journées de Statistique de la SFdS*, Rennes, Juin 2014.

[C3] D. Brazey et B. Portier, Un nouveau modèle de mélange sphérique pour la modélisation 3D, *Séminaire SPOC, Institut de Mathématiques de Bourgogne*, Dijon, 2014.

Bibliographie

[1] C. Rougier J. Meunier A. St-Arnaud, J. Rousseau. 3d head tracking for fall detection using a single calibrated camera. *Image and Vision Computing*, 31 :246–254, 2013.

[2] A. R. Abas. On determining efficient finite mixture models with compact and essential components for clustering data. *Egyptian Informatics Journal*, 14 :79–88, 2013.

[3] A. Abuzaina, M.S. Nixon, and J.N. Carter. Sphere detection in kinect point clouds via the 3d hough transform. *Lecture Notes in Computer Science*, pages 290–297, 2013.

[4] R.A. Adams. *Sobolev spaces*. Academic Press, New York, 1975.

[5] M. Aitkin, D. Anderson, and J. Hinde. Statistical modelling of data on teaching styles (with discussion). *Journal of the Royal Statistical Society*, 144 :419–461, 1981.

[6] H. Akaike. Information theory and an extension of the maximum likelihood principle. *Second International Symposium on Information Theory*, 1973.

[7] J. Aldrich. Ra fisher and the making of maximum likelihood 1912-1922. *Statistical Science*, 12 :162–176, 1997.

[8] R. L. Andrews and I. S. Currim. A comparison of segment retention criteria for finite mixture logit models. *Journal of Marketing Research*, pages 235–243, 2003a.

[9] R. L. Andrews and I. S. Currim. Recovering and profiling the true segmentation structure in markets : an empirical investigation. *International Journal of Research in Marketing*, 20 :177–192, 2003b.

[10] B. Antic, D. Letic, D. Culibrk, and V. Crnojevic. K-means based segmentation for real-time zenithal people counting. *Proc. IEEE ICIP*, pages 2537–2540, 2009.

[11] D. Apprato and C. Gout. A result about scale transformation families in approximation : application to surface fitting from rapidly varying data. *Numerical Algoithms*, 23(2-3) :263–279, 2000.

[12] G.F. Araujo, H.T. Maced, M.T. Chella, C.A.E. Montesco, and M.V.O. Medeiros. Parallel implementation of expectation-maximisation algorithm for the training of gaussian mixture models. *Journal of Computer Science*, 10 :2124–2134, 2014.

[13] R. Arcangéli, M. Cruz de Silanes, and J.J. Torrens. *Multidimensional Minimizing Splines : Theory and Applications*. Springer, 2004.

[14] N. Aronsajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68 :337–404, 1950.

[15] M. Attéia. Fonctions "spline" définies sur un ensemble convexe. *Numer. Math.*, 12 :192–210, 1968.

[16] M. Attéia. Fonctions "spline" et noyaux reproduisants d'aronszajn-bergman. *RIRO*, R-3 :31–43, 1970.

[17] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Readings in computer vision : issues, problems, principles, and paradigms*, pages 714–725, 1987.

[18] J. Barandiaran, B. Murguia, and F. Boto. Real-time people counting using multiple lines. *Proc. Image Analysis for Multimedia Interactive Services Conference*, pages 159–162, 2008.

[19] E. Bauer, D. Koller, and Y. Singer. Update rules for parameter estimation in bayesian networks. *Proceedings of the Tirteenth Conference on Uncertainty in Artificial Intelligence*, pages 3–13, 1997.

[20] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19, 2010.

[21] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. *SIGGRAPH 2000*, pages 417–424, 2000.

[22] A.L. Bertozzi, S. Esedoglu, and A. Gillette. Inpainting of binary images using the cahn-hilliard equation. *IEEE Transactions on Image Processing*, 16(1) :285–291, 2007.

[23] J. C. Bezdek, W. Q. Li, Y. Attikiouzel, and M. Windham. A geometric approach to cluster validity for normal mixtures. *Soft Computing*, 1 :166–179, 1997.

[24] C. Biernacki. *Choix de Modeles en Classification*. PhD thesis, Universite de Technologie de Compiegne, 1997.

[25] C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22 :719–725, 2000.

[26] C. Biernacki and G. Govaert. Using the classification likelihood to choose the number of clusters. *Computing Science and Statistics*, 29 :451–457, 1997.

[27] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. *Computer Vision and Pattern Recognition*, pages 232–237, 1998.

[28] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

[29] E. Borovikov and L. Davis. 3d shape estimation based on density driven model fitting. *3D Data Processing Visualization and Transmission*, pages 116–125, 2002.

[30] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter. A data structure for the 3d hough transform for plane detection. *IFAC*, 2010.

[31] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter. The 3d hough transform for plane detection in point clouds : a review and a new accumulator design. *3D Research*, 2 :1–13, 2011.

[32] T. Bouwmans. Recent advanced statistical background modeling for foreground detection : A systematic survey. *Recent Patents on Computer Science*, 4 :147–176, 2011.

[33] T. Bouwmans, F. El Baf, and B. Vachon. Background modeling using mixture of gaussians for foreground detection - a survey. *Recent Patents on Computer Science*, 3 :219–237, 2008.

[34] H. Bozdogan. Choosing the number of component clusters in the mixture model using a new informational complexity criterion of the inverse-fisher information matrix. *Information and Classification, Springer-Verlag*, pages 40–54, 1993.

[35] P. Bradley, U. Fayyad, and C. Reina. Scaling em (expectation-maximization) clustering for large databases. *Rapport Technique MSR-TR-98-35, Microsoft research*, 1998.

[36] G. Bradski and A. Kaebler. *Learning OpenCV : Computer Vision with the OpenCV Library*. O'Reilly.

[37] L. Breiman and J. H. Friedman. Estimating optimal transformations for multiple regression and correlations (with discussion). *Journal of the American Statistical Association*, 80 :580–619, 1985.

[38] N. Greggio A. Bernardino C. Laschi, P. Dario and J. Santos-Victor. Fast estimation of gaussian mixture models for image segmentation. *Machine Vision and Applications*, 2 :1246–1251, 2012.

[39] S. Cambanis, S. Huang, and G. Simons. On the theory of elliptically contoured distributions. *Journal of Multivariate Analysis*, 11 :368–385, 1981.

[40] J. Campbell, C. Fraley, F. Murtagh, and A. Raftery. Linear flaw detection in woven textiles using model-based clustering. *Pattern Recognition Letters*, 18 :1539–1548, 1997.

[41] M. Camplani and L. Salgado. Efficient spatio-temporal hole filling strategy for kinect depth maps. *Three-Dimensional Image Processing and Applications II*, 2012.

[42] H. Cartan. *Cours de Calcul Différentiel*. Hermann, Paris, 1977.

[43] J. Nečas. *Les méthodes directes en théorie des équations elliptiques*. Masson, Paris, 1967.

[44] J.R. Casas and J. Ruiz-Hidalgo. Real-time head and hand tracking based on 2.5d data. *International Conference on Multimedia and Expo*, 2011.

[45] G. Celeux, D. Chauveau, and J. Diebolt. Stochastic versions of the em algorithm : an experimental study in the mixture case. *Journal of Statistical Computation and Simulation*, 55 :287–314, 1996.

[46] G. Celeux, S. Chrétien, F. Forbe, and A. Mkhadri. A component-wise em algorithm for mixtures. Technical report, Inria 3746, 1999.

[47] G. Celeux, S. Chrétien, and F. Forbes. A component-wise em algorithm for mixtures. *Journal of Computational and Graphical Statistics*, 2012.

[48] G. Celeux and G. Govaert. A classification em algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14 :315–332, 1992.

[49] G. Celeux and G. Soromenho. An entropy criterion for assessing the number of clusters in a mixture model. *Classification journal*, 13 :195–212, 1996.

[50] T. Chan, J. Shen, and S. Kang. Euler's elastica and curvature-based image inpainting. *SIAM Journal on Applied Mathematics*, 63(2) :564–592, 2002.

[51] C. M. Chen and S. Y. Lee. A new parallel em algorithm with the optimal data replication on a hypercube multiprocessor for 3d pet image reconstruction. *Engineering in Medicine and Biology Society*, 1 :626–627, 1994.

[52] G. Cheung, T. Kanade, J. Y. Bouguet, and M. Holler. A real-time system for robust 3d voxel reconstruction of human motions. *IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[53] S. S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic. *Visual Communications and Image Processing*, 5308 :881–892, 2004.

[54] O. Chum and J. Matas. Matching with prosac " progressive sample consensus. *CVPR*, pages 220–226, 2005.

[55] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. *lecture Notes in Computer Science*, 2781 :236–243, 2003.

[56] P.G. Ciarlet. *The finite element method for elliptic problems*. North Holland, Amsterdam, 1978.

[57] P.G. Ciarlet and P.A. Raviart. General lagrange and hermite interpolation in irn with application to finite element methods. *Arch. Mech. Anal.*, 46 :177–199, 1972.

[58] Asus corporation. Xtion sensor. *http ://www.asus.com*, 2011.

[59] Intel corporation and Creative corporation. Creative interactive gesture camera toolkit. *http ://www.intel.com*.

[60] Microsoft corporation. Kinect for xbox 360. *http ://www.microsoft.com*.

[61] A. Cutler and M. P. Windham. Information-based validity functionals for mixture analysis. *Proceedings of the First US/Japan Conference on the Frontier Statistical Modeling : An Informational Approach*, pages 149–170, 1994.

[62] W. Dahmen and C.A. Michelli. *Recent progress in multivariate splines*. Academic Press, New York, 1983.

[63] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[64] A. Dasgupta and A. Raftery. Detecting features in spatial point patterns with clutter via model-based clustering. *Journal of the American Statistical Association*, 93 :294–302, 1998.

[65] C. de Boor. *A practical guide to splines*. Springer Verlag, Berlin-Heidelberg, 1978.

[66] C. de Boor nad K. Höllig. B-splines from parallepipeds. *J. Anal. Math.*, 42 :99–115, 1982-83.

[67] P. E. López de Teruel, J. M. García, and M. Acacio. The parallel em algorithm and its applications in computer vision. *International Conference on Parallel and Distributed Processing Techniques and Applications*, 1 :571–577, 1999.

[68] P. E. López de Teruel and A. Ruiz. A parallel algorithm for tracking of segments in noisy edge images. *Proceedings of the International Conference on Pattern Recognition*, 4 :807–811, 2000.

[69] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39 :1–38, 1977.

[70] K. G. Derpanis. Overview of the ransac algorithm. 2010.

[71] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, 1986.

[72] P. Dollar, C. Wojek, B. Schiele, and P. Perone. Pedestrian detection : a benchmark. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[73] J. Duchon. Splines minimizing rotation-invariant semi-norms in sobolev spaces. *Lecture Notes in Mathematics*, 571 :85–100, 1977.

[74] M. Duflo. *Random Iterative Models*. Springer Verlag, Berlin, 1997.

[75] T. Dutta. Evaluation of the kinect sensor for 3d kinematic measurement in the workplace. *Applied Ergonomics*, 2011.

[76] C. Stauffer E. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 246–252, 1999.

[77] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. *IEEE Frame-Rate Workshop*, pages 751–767, 2000.

[78] M. Enzweiler and D. Gavrila. Monocular pedestrian detection : survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 :2179–2195, 2009.

[79] C. Eveland, K. Konolige, and R. C. Bolles. Background modeling for segmentation of video-rate stereo sequences. *Computer Vision and Pattern Recognition*, pages 266–271, 1998.

[80] G. Fanelli, T. Weise, J. Gall, and L. Van Gool. Real time head pose estimation from consumer depth cameras. *International Conference on Pattern Recognition*, pages 101–110, 2011.

[81] K. Fang and Y. Zhang. *Generalized multivariate analysis*. Springer-Verlag, 1990.

[82] K.T. Fang, S. Kotz, and K.W. Ng. *Symmetric multivariate and related distributions*. Chapman and Hall, 1990.

[83] M.A.T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analys and Machine Intelligence*, 24, 2002.

[84] M. A. Fischler and R. C. Bolles. Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 :381–395, 1981.

[85] C. Fraley and A. Raftery. How many clusters ? which clustering method ? answers via model-based cluster analysis. Technical report, Dept. Statistics Univ. Washington, 1998.

[86] F. Galcik and R. Gargalik. Real-time depth map based people counting. *Advanced Concepts for Intelligent Vision Systems*, pages 330–341, 2013.

[87] S. B. Gokturk and C. Tomasi. 3d head tracking based on recognition and interpolation using a time-of-flight depth sensor. *Computer Vision and Pattern Recognition*, 2 :211–217, 2004.

[88] E. Gomez, M. A. Gomez-Villegas, , and J. M. Marin. A survey on continuous elliptical vector distributions. *Revista matematica complutense*, 16 :345–361, 2003.

[89] E. Gomez, M. A. Gomez-Villegas, and J. M. Marin. Continuous elliptical and exponential power linear dynamic models. *Journal of Multivariate Analysis*, 83 :22–36, 2002.

[90] G. Gordon, T. Darell, M. Harville, and J. Woodfill. Background estimation and removal based on range and color. *Computer Vision and Pattern Recognition*, pages 2459–2464, 1999.

[91] N. Grammalidis and M. G. Strintzis. Head detection and tracking by 2-d and 3-d ellipsoid fitting. *International Conference on Computer Graphics*, pages 221–226, 2000.

[92] N. Greggio, A. Bernardino, and J. Santos-Victor. A practical method for self-adapting gaussian expectation maximization. *ICINCO*, pages 36–44, 2010.

[93] N. Greggio, A. Bernardino, and J. Santos-Victor. Unsupervised learning of finite gaussian mixture models (gmms) : A greedy approach. *Informatics in Control, Automation and Robotics*, 89 :105–120, 2011.

[94] A. K. Gupta and T. Varga. *Elliptically contoured models in statistics*. Kluwer Ac. Press. London, 1993.

[95] M. Harville, G. Gordon, and J. Woodfill. Adaptive video background modeling using color and depth. *ICIP*, 3 :90–93, 2001.

[96] M. Harville, G. Gordon, and J. Woodfill. Foreground segmentation using adaptive mixture models in color and depth. *Workshop on Detection and Recognition of Events in Video*, pages 3–11, 2001.

[97] A. Barthel H.D. Tagare and F.J. Sigworth. Adaptive expectation-maximization algorithm with gpu implementation for electron cryomicroscopy. *Journal of Structural Biology*, 171 :256–265, 2010.

[98] C. Hennig. Methods for merging gaussian mixture components. *Advances in Data Analysis and Classification*, 4 :3–34, 2010.

[99] D. Hernandez, M. Castrillon, and J. Lorenzo. People counting with re-identification using depth cameras. *Proc. Imaging for Crime Detection and Prevention*, pages 1–6, 2011.

[100] A. Hope. A simplified monte carlo significance test procedure. *Journal of the Royal Statistical Society*, 30 :582–598, 1968.

[101] P. Hough. Method and means for recognizing complex patterns. *U.S. Patent*, 1962.

[102] T. Huang, H. Peng, and K. Zhang. Model selection for gaussian mixture models. 2013.

[103] IDS. *http ://fr.ids-imaging.com*.

[104] S. Ikemura and H. Fujiyoshi. Real-time human detection using relational depth similarity features. *Asian Conference on Computer Vision*, pages 25–38, 2010.

[105] J. Illingworth and J. Kittler. A survey of the hough transform. *Computer Vision, Graphics and Image Processing*, 44 :87–116, 1988.

[106] Mesa Imaging. Swissranger. *http ://www.mesa-imaging.ch*.

[107] M. Jamshidian and R. I. Jennrich. Conjugate gradient acceleration of the em algorithm. *Journal of the American Statistical Society*, 88 :221–228, 1993.

[108] B. Jian and B. C. Vemuri. A robust algorithm for point set registration using mixture of gaussians. *International Conference on Computer Vision*, 2 :1246–1251, 2005.

[109] D. Harwood K. Kim, T. H. Chalidabhongse and L. Davis. Background modeling and subtraction by codebook construction. *International Conference on Image Processing*, pages 3061–3064, 2011.

[110] H. Kasahara and K. Shimotsu. Testing the number of components in finite mixture models. *CIRJE*, 2012.

[111] D. Kelker. Distribution theory of spherical distribution and a location-scale parameter generalization. *The Indian Journal of Statistics*, 32 :419–430, 1970.

[112] K. Khoshelham. Accuracy analysis of kinect depth data. *ISPRS Workshop on Laser Scanning*, 38, 2011.

[113] J.W. Kim, K.S. Choi, B.D. Choi, and S.J. Ko. Real-time vision-based people counting system for the security door. *Proc. International Technical Conference On Circuits Systems Computers and Communications*, pages 1416–1419, 2002.

[114] A.B. Koehler and E.H. Murphree. A comparison of the akaike and schwarz criteria for selecting model order. *Applied Statistics*, 37 :187–195, 1988.

[115] C. Kruengkrai and C. Jaruskulchai. A parallel learning algorithm for text classification. *The Eight ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 201–206, 2002.

[116] H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2 :83–97, 1955.

[117] A. Ladikos, S. Benhimane, and N. Navab. Efficient visual hull computation for real-time 3d reconstruction using cuda. *Computer Vision and Pattern Recognition*, 2008.

[118] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2) :150–162, 1994.

[119] N. Lazaros, G. C. Sirakoulis, and A. Gasteratos. Review of stereo vision algorithms : from software to hardware. *International Journal of Optomechatronics*, pages 435–462, 2008.

[120] W. Lin, Y. Wang, Y. Zhuang, and K. S. Zhang. Evaluate the number of clusters in finite mixture models with compact and essential components for clustering data. *Journal of Process Control*, 23 :1052–1062, 2013.

[121] J.L. Lions and E. Magenes. *Problèmes aux limites non homogènes et applications*. Dunod, Paris, 1968.

[122] M. A. Livingston, J. Sebastian, Ai. Zhumin, and J. W. Decker. Performance measurement for the microsoft kinect skeleton. *Virtual Reality Short Papers and Posters*, pages 119–120, 2012.

[123] T. A. Louis. Finding the observed information matrix when using the em algorithm. *Journal of the Royal Statistical Society*, pages 226–233, 1982.

[124] P. C. Mahalanobis. On the generalized distance in statistics. *Proceedings National Institute of Science*, 2 :49–55, 1936.

[125] S. Masnou. Disocclusion : a variational approach using level lines. *IEEE Transactions on Image Processing*, 11(2) :68–76, 2002.

[126] S. Masnou and J. Morel. Level lines based disocclusion. *IEEE International Conference on Image Processing*, 1998.

[127] J. Matas and O. Chum. Randomized ransac with sequential probability ratio test. *ICCV*, 2 :1727–1732, 2005.

[128] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley and Sons, 2000.

[129] G. J. McLachlan. On bootstrapping the likelihood ratio test statistic for the number of components in a normal mixture. *Applied Statistics*, 36 :318–324, 1987.

[130] G. J. McLachlan and K. Thriyambakam. *The EM Algorithm and Extensions*. WILEY-Interscience, 2008.

[131] I. Meilijson. A fast improvement to the em algorithm on its own terms. *Journal of the Royal Statistical Society*, 51 :127–138, 1989.

[132] X. L. Meng and D. van Dyk. The em algorithm - an old folksong sung to a fast new tune (with discussions). *Journal of the Royal Statistical Society*, 59 :511–567, 1997.

[133] M. Miloslavsky, M. Laan, and M. J. van der Laan. Fitting of mixtures with unspecified number of components using cross validation distance estimate. *Comput. Statist. Data Anal.*, 41 :413–428, 2003.

[134] A. Moore. Very fast em-based mixture model clustering using multiresolution kd-trees. *AAdvances in Neural Information Processing Systems*, 11 :543–549, 1999.

[135] M. Blanke M.R. Blas, R.B. Rusu and M. Beetz. Fault-tolerant 3d mapping with application to orchard robot. *International Symposium on Fault Detection, Supervision and Safety of Technical Processes*, pages 893–898, 2009.

[136] S. Mukherjee, B. Saha, I. Jamal, R. Leclerc, and N. Ray. A novel framework for automatic passenger counting. *Proc. ICIP*, pages 2969–2972, 2011.

[137] R. Neal. Bayesian mixture modeling. *Proc. 11th Int'l Workshop Maximum Entropy and Bayesian Methods of Statistical Analysis*, pages 197–211, 1992.

[138] R. Neal and G. Hinton. A view of the em algorithm that justifies incremental, sparse and other variants. *Learning in Graphical Models*, pages 355–371, 1998.

[139] S. K. Ng and G. J. McLachan. On the choice of the number of blocks with the incremental em algorithm for the fitting of normal mixtures. *Statistics and Computing*, 13 :45–55, 2003.

[140] S. K. Ng and G. J. McLachan. Speeding up the em algorithm for mixture model-based segmentation of magnetic resonance images. *Pattern Recognition*, 37 :1573–1589, 2004.

[141] T.M.N. Nguyen and J. Saracco. Estimation récursive en régression inverse par tranches (sliced inverse regression). *Journal de la Société Française de Statistique*, 151 :19–46, 2010.

[142] J. Oliver, R. Baxter, and C. Wallace. Unsupervised learning using mml. *Proc. 13th Int. Conf. Machine Learning*, pages 364–372, 1996.

[143] N. M. Oliver, B. Rosario, and A. P. Petland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence*, 22 :831–843, 2000.

[144] A. Oliviera-Brochado and F.V. Martins. Assessing the number of components in mixture models : a review. *Universidade do Porto, Faculdade de Economia do Porto*, 2005.

[145] opencv.org. Opencv.

[146] L. E. Ortiz and L. P. Kaelbling. Accelerating em : an empirical study. *On Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, pages 512–521, 1999.

[147] R. Rocha M. Campos P. Núñez, P. Drews Jr and J. Dias. Novelty detection and 3d shape retrieval based on gaussian mixture models for autonomous surveillance robotics. *International Conference on Intelligent Robots and Systems*, pages 4724–4730, 2009.

[148] C. Papazov and D. Burschka. An efficient ransac for 3d object recognition in noisy and occluded scenes. *Proceedings of the 10th Asian Conference on Computer Vision ACCV*, pages 135–148, 2011.

[149] F. Pernkopf and D. Bouchaffra. Genetic-based em algorithm for learning gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27 :1344–1348, 2005.

[150] B. C Peters and H. F. Walker. An iterative procedure for obtaining maximum-likelihood estimates of the parameters for a mixture of normal distributions. *SIAM Journal of Applied Mathematics*, 35 :362–378, 1978.

[151] V. Pham, P. Vo, and V. Hung. Gpu implementation of extended gaussian mixture model for background subtraction. *Proceedings of the International Conference on Computing and Communication Technology Research, Innovation and Vision for the Future*, pages 1–4, 2010.

[152] F. Qi, J. Han, P. Wang, G. Shi, and F. Li. Structure guided fusion for depth map inpainting. *Pattern Recognition Letters*, 34 :70–76, 2013.

[153] T. Rabbani and F. Van Den Heuvel. Efficient hough transform for automatic detection of cylinders in point clouds. *ISPRS*, 3 :60–65, 2005.

[154] T. Rabbani, F. Van Den Heuvel, and G. Vosselmann. Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 36 :1–6, 2006.

[155] R. Raguram, J.M. Frahm, and M. Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *European Conference on Computer Vision*, 2008.

[156] V. Ramaswamy, W.S. DeSarbo, D.J. Reibstein, and W.T. Robinson. An empirical pooling approach for estimating marketing mix elasticities with pims data. *Marketing Science*, 12 :103–124, 1993.

[157] C. Rasmussen. The infinite gaussian mixture model. *Advances in Neural Information Processing Systems*, 12 :554–560, 2000.

[158] M. Rauter. Reliable human detection and tracking in top-view depth images. *Proc. CVPR*, pages 529–534, 2013.

[159] R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood an the em algorithm. *SIAM*, 26 :195–239, 1984.

[160] S. Richardson and P. J. Green. On bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society*, pages 731–758, 1997.

[161] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. Singapore : World Scientific, 1989.

[162] L.I. Rudin, S. Osher, and E. Fatemi. Non linear total variation based noise removal algorithms. *Phys. D*, 60(1-4) :259–268, 1992.

[163] R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Institut fur Informatik des Technischen Universitat Munchen, 2009.

[164] R.B. Rusu. Functional object mapping of kitchen environments. *IROS*, pages 3525–3532, 2008.

[165] R.B. Rusu and S. Cousins. 3d is here : Point cloud library (pcl). *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[166] R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3d point cloud based object maps for household environments. *Robot. Auton. Syst.*, 56 :927–941, 2008.

[167] P.P. Sapkota. *Segmentation of coloured point cloud data*. PhD thesis, International Institute for Geo-Information Science and Earth Observation, 2008.

[168] R. Schnabel, R. Wahl, and R. Klein. Shape detection in point clouds. *SIGGRAPH*, 2006.

[169] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point cloud shape detection. *Computer Graphics Forum*, 26 :214–226, 2007.

[170] R. Schnabel, R. Wahl, R. Wessel, and R. Klein. Shape recognition in 3d point clouds. *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2008.

[171] I. J. Schoenberg. Contribution to the problem of approximation of equidistant data by analytic functions. *Quart. of Appl. Math.*, 4 :45–99 and 112–141, 1960.

[172] L.L. Schumaker. *Spline Functions : Basic Theory*. Wiley, New York, 1981.

[173] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6 :461–464, 1978.

[174] S. Se and M. Brady. Ground plane estimation, error analysis and applications. *Robotics and Autonomous Systems*, 39 :59–71, 2002.

[175] S. M. Seitz. A comparison and evaluation of multi-view stereo reconstruction algorithms. *Computer Vision and Pattern Recognition*, pages 519–528, 2006.

[176] C. M. Shakarji. Least-squares fitting algorithms of the nist algorithm testing system. *Journal of Research of the National Institute of Standards and Technology*, 103 :633–641, 1998.

[177] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. *CVPR*, pages 1297–1304, 2011.

[178] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image supplementary material. *CVPR*, 2011.

[179] P. Smith. Model selection for probabilistic clustering using cross-validated likelihood. *Statistics and Computing*, 10 :63–72, 2000.

[180] G. De Soete and W. S. DeSarbo. A latent class probit model for analysing pick any/n data. *Journal of Classification*, 8 :45–63, 1991.

[181] SoftKinetic. Depthsense. *www.softkinetic.com*.

[182] L. Spinello and K. Arras. People detection in rgb-d data. *IEEE International Conference on Intelligent Robots*, pages 3838–3843, 2011.

[183] W.F. Stout. *Almost sure convergence*. A Series of Monographs and Textbooks in Probability and Statistics, 1974.

[184] H. Sun and S. Wang. Measuring the component overlapping in the gaussian mixture model. *Data Mining and Knowledge Discovery*, 23 :479–502, 2011.

[185] H. J. Sun, M. Sun, and S. R. Wang. A measurement of overlap rate between gaussian components. *International Conference on Machine Learning and Cybernetics*, 4 :2373–2378, 2007.

[186] H. Sunyoto. A comparative study of fast dense stereo vision algorithms. *Intelligent Vehicles Symposium*, pages 319–3247, 2004.

[187] S. Bakkes T. van Oosterhout and B. Krose. Head detection in stereo data for people counting and segmentation. *VISAPP*, pages 620–625, 2011.

[188] F. Tarsha-Kurdi, T. Landes, and P. Grussenmeyer. Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from lidar data. *ISPRS Workshop on Laser Scanning*, 2007.

[189] B. Thiesson. Accelerated quantification of bayesian networks with incomplete data. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 306–311, 1995.

[190] M. Thiesson, B. Meek, and D. Heckerman. Accelerating em for large databases. *Machine Learning*, 45 :279–299, 2001.

[191] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *ICCV*, 1998.

[192] F. Tombari and L. Di Stefano. Object recognition in 3d scenes with occlusions and clutter by hough voting. *PSIVT*, pages 349–355, 2010.

[193] P.H.S. Torr and D.W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24 :271–300, 1997.

[194] N. Ueda, R. Nakano, Z. Ghahramani, and G.E. Hiton. Smem algorithm for mixture models. *Neural Computing*, 12 :2109–2128, 2000.

[195] J. Verbeek, N. Vlassis, and B. Krose. Efficient greedy learning of gaussian mixture models. *Neural Computation*, 15 :469–485, 2003.

[196] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[197] N. Vlasis and A. Likas. A greedy em algorithm for gaussian mixture learning. *Neural Processing Letters*, 15 :77–87, 2002.

[198] C. Wallace and D. Dowe. Minimum message length and kolmogorov complexity. *The Computer Journal*, 42 :270–283, 1999.

[199] C. Wallace and P. Freeman. Estimation and inference via compact coding. *Journal of Royal Statistical Society*, 49 :241–252, 1987.

[200] H. Wang, B. Luo, Q. Zhang, and S. Wei. Estimation for the number of components in a mixture model using stepwise split and merge em algorithm. *Pattern Recognition Letters*, 25 :1799–1809, 2004.

[201] S. Weik. A passive full body scanner using shape from silhouettes. *Pattern Recognition*, 1 :750–753, 2000.

[202] M. Whindham and A. Cutler. Information ratios for validating mixture analysis. *Journal of the American Statistical Association*, 87 :1188–1192, 1992.

[203] M. Yao X. Zhang, R. He and F. Zhu. The elliptical contoured mixture model for image segmentation. *Journal of Computational Information Systems*, pages 7847–7855, 2012.

[204] S. Feng Z. Lei D. Yi X. Zhang, J. Yan and S. Z. Li. Water filling : Unsupervised people counting via vertical kinect sensor. *AVSS*, 2012.

[205] L. Xia, C.C. Chen, and J.K. Aggarwal. Human detection using depth information by kinect. *Computer Vision and Pattern Recognition*, pages 15–22, 2011.

[206] L. Xu. Comparative analysis on convergence rates of the em algorithm and its two modifications for gaussian mixtures. *Neural Processing Letters*, 6 :69–76, 1997.

[207] T. Yahiaoui, C. Meurie, L. Khoudour, and F. Cabestaing. A people counting system based on dense and close stereovision. *Proceedings of the 3rd international conference on image and signal processing*, pages 59–66, 2008.

[208] M. Ying Yang and W. Förstner. Plane detection in point cloud data. Technical report, University of Bonn, 2010.

[209] K. Yosida. *Functionnal Analysis*. Springer-Verlag, Berlin, 1974.

[210] S. Yous, H. Laga, and K. Chihara. People detection and tracking with world-z map from a single stereo camera. *International Workshop on Visual Surveillance*, 2008.

[211] Q. Zhan, Y. Liang, and Y. Xiao. Color-based segmentation of point clouds. *Laser Scanning*, pages 248–252, 2009.

[212] Z. Zhang, C. Chen, J. Sun, and K. L.Chan. Em algorithms for gaussian mixtures with split-and-merge operation. *Pattern Recognition*, 36 :1973–1983, 2003.

[213] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per pixel image for the task of background subtraction. *Pattern Recognition Letters*, 27 :773–780, 2006.

[214] M. Zuliani. Ransac for dummies. Technical report, 2011.