



Mémoire de Stage

RECALAGE DE SQUELETTES MATHÉMATIQUES 3D ET CRÉATION D'UN ESPACE D'ANÉVRISMES AORTIQUES

Présenté par **Matthieu NASTORG**
Numéro étudiant : 21813610

Sous la direction de Messieurs **Angelo IOLLO** et **Tommaso TADDEI**

Mémoire de Master Mathématiques Appliquées et Statistiques de l'Université de Bordeaux

Remerciements

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner tous mes remerciements.

Je tiens à exprimer toute ma reconnaissance à mon tuteur de stage, Monsieur Angelo IOLLO, professeur en mathématiques appliquées à l'Université de Bordeaux et Inria Bordeaux Sud-Ouest. Je le remercie pour sa disponibilité, sa confiance, ses précieux conseils théoriques et pratiques et son important soutien lors de mes recherches de thèse.

Je remercie également Messieurs Florian BERNARD et Romain LEGUAY, co-fondateurs de la startup NUREA, qui ont pris le temps de me recevoir régulièrement pour répondre à mes questions et qui, par leurs judicieux conseils, n'ont jamais cessé de guider mes réflexions.

Je tiens également à témoigner toute ma reconnaissance aux personnes suivantes :

Monsieur Tommaso TADDEI, chargé de recherche à Inria Bordeaux Sud-Ouest, pour son écoute, son aide et ses recommandations.

Mon ami, Thomas PHILIBERT, de nos premiers projets universitaires à notre collaboration sur ce passionnant sujet. Merci pour ta présence, morale et intellectuelle, pour nos "Skype" quotidiens venant égayer de (parfois) longues journées de confinement. Je te souhaite bon courage, et tout le meilleur pour la suite!

Mes parents, pour leurs encouragements et leur soutien, avec une mention particulière pour mon petit frère qui a dû me supporter lors de ces mois particuliers de télétravail.

Pour finir, j'aimerais dédicacer ce mémoire à ma grand-mère, qui, où qu'elle soit, continue de veiller sur moi.

"À Thé!"

Table des matières

1	Introduction	3
2	Recalage de squelettes mathématiques 3D	5
2.1	Introduction sur les squelettes mathématiques	5
2.2	Problématique	7
2.3	Description des données fournies	8
2.4	Recalage optimal pour squelettes mathématiques	9
2.4.1	Description de la méthode	9
2.4.2	Étape 1 : Création du nuage de points	10
2.4.3	Étape 2 : Reconstruction du squelette	11
2.4.4	Étape 3 : Recalage optimal	12
2.4.5	Étape 4 : Mise à jour et évolution de l'erreur	18
2.5	Limites et temps de calculs	19
3	Espace d'anévrismes	21
3.1	Paramétrisation de l'anévrisme aortique	21
3.1.1	Approximation de la paroi	21
3.1.2	Reconstruction et erreur	23
3.2	Bases réduites pour la génération d'anévrismes aortiques	25
3.2.1	Création des matrices de données brutes	25
3.2.2	Réduction de dimension	26
3.2.3	Applications aux données	28
4	Conclusion	30
A	Éléments de géométrie différentielles pour courbes paramétrées	31
B	Illustrations pour la méthode de recalage 2	34

CHAPITRE 1

Introduction

L'aorte est la plus grande et grosse artère du corps humain. Reliée au ventricule gauche du cœur, elle transporte le sang dans les différentes branches du système sanguin, alimentant ainsi tous les organes du corps. Elle peut se décomposer en deux parties par rapport à la position du diaphragme : *l'aorte thoracique* pour la partie supérieure et *l'aorte abdominale* pour la partie inférieure.

Les maladies aortiques sont des affections complexes qui peuvent survenir à tout âge et nécessitent une surveillance constante. La pathologie de l'anévrisme consiste en la dilatation localisée des parois de l'aorte abdominale et, plus précisément, lorsque le diamètre transversal maximal du vaisseau est supérieur à une fois et demi la normale. Le risque principal est la rupture qui consiste en la déchirure de la paroi, entraînant une complication gravissime et souvent fatale (mortalité globale atteint les 90%). Bien qu'il s'agisse d'une maladie fréquente, la majorité d'entre eux sont de petites tailles et touchent principalement les patients de plus de 50 ans (la moyenne d'âge des patients opérés est d'environ 70 ans). Ainsi, dans bien des cas, c'est à l'occasion d'un test d'imagerie, réalisé pour une toute autre raison, que le diagnostic est posé.

Lorsque les spécialistes détectent une grosseur significative d'un début d'anévrisme, un suivi temporel régulier et précis du patient est nécessaire afin d'anticiper et prévenir une éventuelle complication, traitée, la plupart du temps, grâce à des interventions chirurgicales. De nos jours, les chirurgiens cardio-vasculaires disposent malheureusement de très peu de données provenant de l'analyse des scanners. Motivés par cette idée, Messieurs Florian BERNARD et Romain LEGUAY ont conçu et développé la startup NUREA en 2018, accompagnés par les appuis scientifiques du Professeur Angelo IOLLO, professeur en mathématiques appliquées à l'Université de Bordeaux et Inria Bordeaux Sud-Ouest, et médicaux du Professeur Eric DUCASSE, chef de servie de chirurgie vasculaire et générale du CHU de Pellegrin, Bordeaux.

Leur objectif est de développer un logiciel, intuitif et directement utilisable par les praticiens, permettant d'automatiser l'extraction d'informations **pertinentes** et **fiables**, nécessaires au diagnostic et au suivi des patients. En se focalisant en premier lieu sur l'étude des anévrismes aortiques, leur technologie repose sur la combinaison des techniques mathématiques de traitement d'images, de modélisation et simulation numérique et d'intelligence artificielle. En étendant leurs méthodes à l'intégralité du système vasculaire, ils ambitionnent de favoriser la médecine préventive et, par conséquent, de diminuer le nombre d'accidents cardiovasculaires.

Les recherches menées ces six derniers mois et décrites tout au long de ce mémoire n'ont pas pour objectif d'être directement intégrable au sein du logiciel de NUREA. Toutefois, elles permettent, dans un premier temps, de confirmer, ou non, certaines hypothèses émises par les membres de la startup. Elles proposent également de nouveaux axes de recherche et ambitionnent d'introduire de nouvelles méthodes pour de futurs travaux, qui, nous l'espérons, déboucheront sur une application utile et performante.

Nous nous intéressons, en premier lieu, à une méthode de recalage pour squelettes mathématiques. Plus précisément, NUREA a développé un algorithme, rapide et efficace, pour le calcul du squelette d'une géométrie 3D. Cependant, leur méthode s'avère sensible aux petites variations topologiques et présente, parfois, quelques erreurs empêchant de bonnes analyses futures. C'est pourquoi, nous proposons dans le chapitre 2 une méthode permettant de recaler proprement le squelette au sein de la géométrie 3D et corriger ces petits défauts. Nous porterons un intérêt tout particulier au bon déplacement des points de bifurcation originaux et garderons également comme objectif de conserver la segmentation du squelette initialement fourni.

Ensuite, le chapitre 3 propose la création d'un espace mathématique pour la modélisation d'anévrismes aortiques. En partant des données du squelette mathématique et de la géométrie 3D initiale, nous essayons, tout d'abord, de paramétriser la paroi de l'anévrisme aortique de manière rapide et précise. En regroupant les paramètres issus des paramétrisations pour un certain nombre d'anévrismes aortiques fournis, on utilisera des techniques statistiques pour synthétiser les informations et ressortir un petit nombre de vecteurs formant une base pour la paramétrisation d'anévrismes aortiques. Par combinaison linéaire de ces vecteurs, nous serons en mesure de produire une infinité d'anévrismes aortiques, dont l'utilité sera discuté dans le chapitre 4.

Chaque chapitre incluera des sections dans lesquelles nous discuterons de l'état de l'art sur le sujet, de la méthode développée, de ses limites et de ses ouvertures sur d'autres axes de recherche. Nous utilisons le langage de programmation PYTHON. Les résultats sont affichés via le logiciel de visualisation PARAVIEW. Les codes relatifs au travail décrit dans ce mémoire sont présents sur la page github suivante : https://github.com/mnastorg/STAGE_ANEVRISSMES. Vous y trouverez également des informations sur leur organisation et leur fonctionnement. Finalement, est détaillé, en annexe A, de brèves notions mathématiques que nous utilisons régulièrement au sein de ce rapport et l'annexe B comportera des illustrations exhaustives et plus précises des résultats obtenus dans 2.

En vous souhaitant bonne lecture,

Matthieu Nastorg

CHAPITRE 2

Recalage de squelettes mathématiques 3D

2.1 Introduction sur les squelettes mathématiques

L'étude et la création de modèles tri-dimensionnels représentatifs de formes solides et compactes sont, de nos jours, utilisées dans de nombreuses disciplines telles que l'infographie, l'imagerie médicale, la visualisation, l'inspection digitale, la métrologie...

Alors qu'une forme géométrique est généralement décrite par ses bordures, on s'intéresse ici à la notion de *squelette mathématique* ou autrement appelé *axe médian* (medial axis en anglais). Ce dernier permet une description interne du solide en mettant en valeur les propriétés géométriques et topologiques de l'objet telles que ses connectivités, sa longueur, sa direction, sa profondeur... Si l'on couple cela avec la distance de ces points à la bordure, nous obtenons une paramétrisation nous permettant de reconstruire, approximativement, l'intégralité du solide.

En résumé, le procédé de squelettisation permet de passer d'une géométrie complexe avec, possiblement, un grand nombre de données, à une géométrie plus simple et moins coûteuse, représentative de la forme initiale et facile à analyser.

Ce concept fut d'abord introduit par HARRY BLUM pour le cas de la dimension deux. Les études sur ce sujet ne sont pas récentes, c'est pourquoi comprendre les propriétés et savoir calculer des squelettes en deux dimensions sont dorénavant des notions bien connues. Cependant, l'extension au cas de la troisième dimension n'est pas trivial. En effet, ces dernières, plus riches et complexes apportent des spécificités et propriétés intrinsèques différentes. De plus, le calcul de manière précise de ces squelettes en trois dimensions (dont les données initiales sont plus importantes) représente un challenge bien plus grand que pour la dimension inférieure.

Finalement, de nombreuses méthodes de squelettisation 3D se sont développées ces dernières années pour des problèmes spécifiques et peu de méthodes sont donc décrites de manière générale. Le travail effectué dans [4], et dont nous nous sommes grandement inspirés, développe un état de l'art complet des avancées technologiques concernant la squelettisation de géométries tri-dimensionnelles.

Dans la littérature, il existe différentes définitions de squelettes mathématiques, toutes équivalentes,

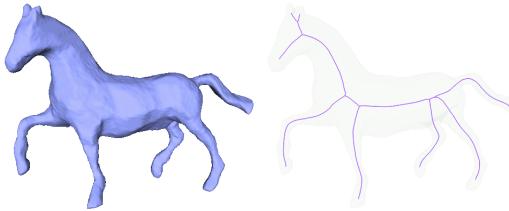


FIGURE 2.1 – Exemple d'une squelettisation 3D extraite de ([3])

menant à un grand panel d'algorithmes pour les calculer. Lorsque l'on regarde plus précisément les termes utilisées, certains auteurs considèrent "squelettes" et "axes médians" de façon identique alors que d'autres non. On peut aussi trouver des variantes telles que "straight skeletons" pour des polygones, "squelettes morphologiques", algorithmes de "thinning" et bien plus encore. Par exemple, nous observons, pour des formes 3D, des algorithmes qui agissent directement sur un maillage (voir [6]) et d'autres utiliser la notion de voxelisation (voir [7]).

Ci-après, nous présentons deux des définitions les plus courantes pour le calcul d'un squelette mathématique. Ces notions sont aussi celles que nous avons utilisées pour mener à bien le travail de ce chapitre.

Définition 2.1 (Boules Maximales).

Une boule B est dite maximale dans un ensemble A si :

- $B \subseteq A$
- Si une autre boule C contient B alors $C \not\subseteq A$

Définition 2.2 (Définition 1).

Le squelette mathématique d'un objet O est défini comme l'ensemble M des centres de toutes les boules maximales inscrites de O .

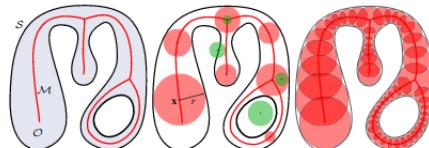


FIGURE 2.2 – Boules maximales inscrites, extrait de ([4])

La figure de gauche dans 2.2 représente le squelette M de la forme O de bordure S . Celle du milieu donne un exemple de la définition d'une boule maximale inscrite (en rouge) tandis que les vertes sont ni inscrites, ni maximales. La figure de droite donne une approximation de l'union des boules maximales inscrites représentant le squelette mathématique.

Si l'on ajoute à l'ensemble M l'ensemble des rayons correspondants R , on obtient une paire $(x, r) \in (M, R)$ et on parle de transformation par axe médian (TAM) (*Medial Axis Transform* en anglais) de O . On appelle le couple (x, r) un atome médian et les positions x sont appelées les points du squelette ou points médians. Ce couple permet de capturer de manière précise la géométrie de l'objet ainsi que sa topologie, via la notion de voisinage entre les points.

En dimension deux, le squelette est une structure 1D qui consiste en un ensemble de courbes localement centrées par rapport à la surface S de O . En dimension trois, on obtient une surface médiane qui consiste en l'ensemble des variétés avec frontière qui s'entrecroisent.

En conclusion, la TAM encode O car il est possible de reconstituer précisément la forme initiale à partir des données de la squelettisation. En effet, on a

$$O = \bigcup_{(x,r) \in (M,R)} B(x, r)$$

Construire la TAM d'une forme est appelée la *squelettisation* et l'opération inverse la *reconstruction*.

Nous disposons d'une seconde définition, équivalente à la première, pour décrire le squelette mathématique d'une géométrie O .

Définition 2.3 (Définition 2).

Le squelette mathématique d'une géométrie O est défini comme les points de discontinuité de la fonction distance associée.

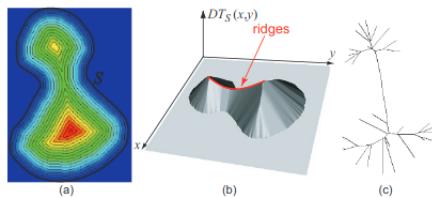


FIGURE 2.3 – Squelette via fonction distance, extrait de ([4])

Sur la figure 2.3, (a) et (b) représentent deux affichages de la fonction distance, avec, sur celle du milieu la visualisation des points de discontinuité. L'image (c) montre le squelette après extraction des points de discontinuité.

La fonction distance est une fonction qui retourne, pour tout point $x \in O$, sa distance au point le plus proche de la frontière de O . Pour son calcul, nous utiliserons un algorithme appelé le *Fast-Marching*. Pour plus d'informations sur le calcul de la fonction distance, nous invitons le lecteur à regarder notre projet de groupe effectué en Master 1, disponible à l'adresse : <https://github.com/mnastorg/GTT>.

2.2 Problématique

En utilisant la définition 2.2 et leur propre algorithme, NUREA est en mesure de nous fournir un squelette rapidement et de manière efficace. Cependant, leur méthode se heurte à certaines limites. En effet, ils extraient dans un premier temps les centres de certaines boules inscrites maximales, puis effectuent un post-traitement permettant de les relier et de les ordonner via une structure de données *d'arbre binaire*.

Lors de leur première étape, la sélection et l'extraction d'un centre (point de squelette) est très sensible aux petites variations topologiques, fréquentes pour des géométries tri-dimensionnelles, entraînant des erreurs lors de la phase de reconstruction et, par conséquent, un squelette qui n'est pas tout à fait exact.

Plus précisément, pour être en mesure de remarquer l'évolution d'un anévrisme, on peut penser comparer plusieurs squelettes d'un même patient pris à des temps différents. On souhaite alors que tous les squelettes représentent la même entité dans l'aorte. Cela signifie, dans un premier temps, être en mesure de 'couper' les géométries aux endroits où sont généralement situés les anévrismes aortiques (entre les artères rénales et lesiliaques) et, dans un second temps, passer outre les erreurs liées à l'angiographie.

Le second problème peut-être corrigé via des méthodes statistiques discutées dans le chapitre 3. Pour le premier, on peut s'inspirer des points de bifurcation du squelette. En effet, il est certain qu'il en existera au niveau des artères rénales et desiliaques. L'intérêt est donc d'obtenir des points de bifurcation qui soient *bien placés*, ce qui n'est pas systématiquement le cas via la méthode de NUREA.

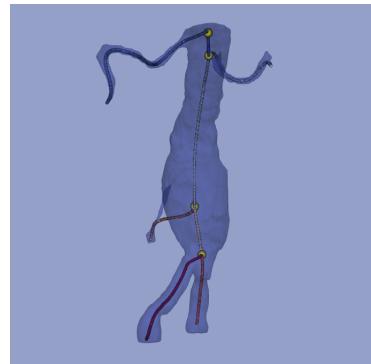


FIGURE 2.4 – Problème squelette sur une géométrie simple

La figure 2.4 nous montre une géométrie 3D et un squelette fourni par NUREA. Chaque couleur est représentative d'une segmentation. Les points jaunes représentent les points de bifurcation. Nous pouvons clairement observer que le squelette présente une erreur topologique au niveau de la mésentérique inférieure et un point de bifurcation mal placé. La suite de ce chapitre propose de développer une méthode pour recaler le squelette fourni par la startup dans la géométrie 3D et corriger, si besoin, les erreurs sur les points de bifurcation, tout en conservant la segmentation initiale.

2.3 Description des données fournies

Avant d'entrer dans les détails de notre méthode, cette section donne un aperçu des données fournies par la startup NUREA, avec lesquelles nous travaillons.

Nous sommes en possession de 10 géométries 3D et squelettes associés d'aortes abdominales. Sur les 10 fournies, 7 ne comportent pas d'anévrismes et 3 sont porteuses. Sur les 7, nous possédons une géométrie "*simple*" mais parfaitement représentative de la problématique (voir 2.4). Nous utiliserons cette géométrie pour visualiser les résultats de la méthode décrite par la suite. D'autres résultats pour quelques géométries, plus complexes, sont affichés en annexe B.

La géométrie 3D initiale est un fichier **.stl**, abréviation pour fichier *stéréolithographique*. Il s'agit d'un format de fichier qui ne décrit que la surface d'un objet. Il représente une surface fermée par une série de triangles que l'on appelle communément un maillage de surface. Pour assurer le bon fonctionnement de nos algorithmes, on demande à ce que la structure fournie n'ait qu'une seule couche de maillage et qu'elle soit complètement fermée aux extrémités.

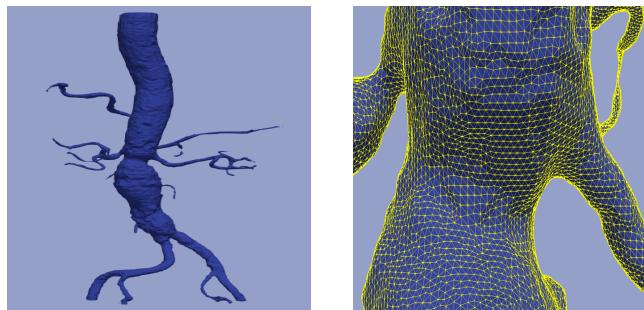


FIGURE 2.5 – A gauche une géométrie 3D complexe, à droite un zoom sur le maillage

Le squelette initial est fourni sous format (**.csv**) pour *Comma Separated Value* (format texte standard) et contient les informations suivantes, en colonne :

1. Les coordonnées des points du squelette (x, y, z)

2. Les labels de la segmentation (chaque point se voit associé un entier correspondant à la branche à laquelle il appartient)
3. La notion de points de *bifurcation*
4. La notion de points *extrêmes*

Pour les deux derniers points, les données sont décrites sous forme binaire, à savoir, 1 si le point est de bifurcation / extrême et 0 sinon.

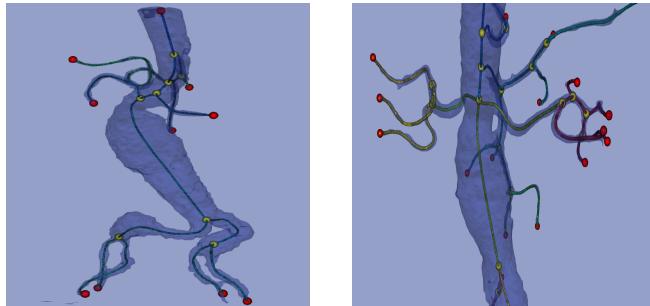


FIGURE 2.6 – Géométrie et squelette avec segmentation, en jaune bifurcation, en rouge extrêmes

2.4 Recalage optimal pour squelettes mathématiques

2.4.1 Description de la méthode

La méthode que nous développons dans ce chapitre est motivée par la volonté de garder le squelette initialement fourni par NUREA, c'est à dire conserver des informations essentielles telles que la segmentation, les points de bifurcation et les points extrêmes (voir figure 2.4). Ainsi, on ne cherche pas à reconstruire un squelette mais plutôt à conserver l'ancien tout en le replaçant correctement dans sa géométrie d'origine.

Comment définir "replacer correctement"?

En s'inspirant de la définition 2.3, nous nous sommes rendus compte que nous pouvions extraire un nuage de points grossièrement représentatif de la forme exacte (par définition) du squelette voulu. On appellera ce nuage de points la **cible**. Après avoir reconstruit le squelette de NUREA (que l'on nommera la **source**) par des courbes paramétrées de type B-Spline, nous proposons une méthode de recalage, inspirée de [8], permettant de déplacer l'intégralité du squelette reconstruit (la source), de manière optimale, dans le nuage de points (la cible). Plus précisément, cette méthode propose de minimiser une fonction coût dont l'inconnu est un vecteur de déplacement des points de contrôle des B-Splines.

En résumé, la méthode développée dans ce chapitre comporte 4 étapes qui sont :

1. L'extraction d'un nuage de point **cible** en utilisant la définition 2.3.
2. Une reconstruction du squelette fourni par NUREA en terme de courbes paramétrées ainsi que quelques analyses géométriques complémentaires (repère de FRÉNET, nombre de points, connectivités...) pour obtenir un squelette **source** prêt pour l'étape suivante.
3. Une méthode de recalage pour déplacer, de manière optimale, l'ensemble de la source dans la cible. Attention, cette étape va *désarticuler* le squelette.
4. Une mise à jour du squelette recalé (reconnection des branches) en utilisant les notions de connectivité et projection.

2.4.2 Étape 1 : Création du nuage de points

Nous nous intéressons ici à l'extraction du nuage de points **cible** de notre recalage. En prenant comme source d'inspiration la définition 2.3, nous calculons la fonction distance associée à la géométrie initiale et observons ses points de discontinuité. Ces derniers peuvent également être vus comme les points de la fonction distance qui ont la plus forte courbure. Ainsi, en appliquant un filtre *Laplacien* sur cette fonction distance, nous pouvons extraire tous les points de courbure supérieurs à un certain seuil (fixé par l'utilisateur). Nous décrivons, ci-après, nos 4 étapes pour l'extraction du nuage en se focalisant également sur la façon dont nous nous sommes adaptés aux géométries fournies.

2.4.2.1 Voxelisation du maillage

Pour l'extraction du nuage de points, nous utilisons essentiellement la géométrie 3D fournis par NUREA. Comme expliqué dans la section 2.3, nous sommes en présence d'un maillage de surface, peu utile pour effectuer une fonction distance. Pour pallier ce problème nous effectuons une *voxelisation* du maillage de surface. Nous utilisons les fonctions disponibles au sein de la librairie *trimesh* de PYTHON. Cette dernière nous permet, en insérant seulement la taille voulue des voxels, de créer une grille cartésienne de voxels cubiques entourant la géométrie. Les voxels correspondants aux points du maillage de surface seront automatiquement activés. Nous demandons également à ce que les voxels internes à la géométrie soient activés (d'où la nécessité d'avoir une géométrie initiale complètement fermée).

2.4.2.2 Calcul de la fonction distance

Maintenant que l'on a transformé notre maillage de surface en une voxelisation, nous pouvons appliquer un algorithme de *Fast-Marching* pour calculer la fonction distance. Nous ajoutons également une convolution avec un noyau gaussien 3D pour lisser le résultat et éviter des irrégularités lors de l'application du filtre *Laplacien*. Le résultat est une matrice comprenant les points de coordonnées des centres des voxels et la valeur de la fonction distance correspondante (voir 2.7).

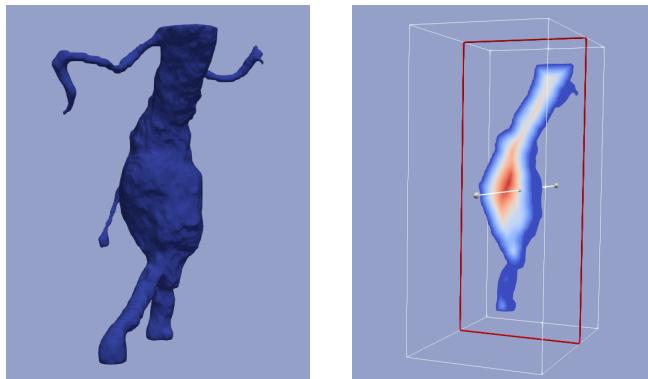


FIGURE 2.7 – Géométrie initiale (gauche) et coupe fonction distance (droite)

2.4.2.3 Application du filtre Laplacien

Ensuite, nous effectuons une convolution avec un noyau Laplacien afin d'exprimer la notion de courbure de la fonction distance. Les plus hauts "pics" du Laplacien de la fonction distance seront considérés comme nos points représentatifs du squelette à obtenir. En traitement d'image, un noyau (filtre) est une petite matrice utilisée pour le floutage, l'amélioration de la netteté, le gaufrage, détection de contours, etc... Une description rapide et simple du fonctionnement d'une convolution peut-être trouvé dans [2]. Dans notre cas, nous utilisons comme noyau de convolution Laplacien la matrice matrice ($3 \times 3 \times 3$) prenant la forme suivante :

$$\left[\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & -6 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right]$$

2.4.2.4 Extraction du nuage cible

Pour extraire le nuage de points cible, nous allons sélectionner tous les points qui sont supérieurs à un certain seuil, c'est à dire que l'on considère tous les points dont la courbure de la fonction distance est supérieure à une certaine valeur fixée. Dans notre algorithme, la valeur de seuil à insérer est en réalité un pourcentage. En effet, soit un seuil $s \in [0, 1]$ et m la valeur maximum de la courbure du Laplacien de la fonction distance. Alors, nous sélectionnons tous les points dont la courbure est supérieure à $v = s * m$. Cela nous permet d'obtenir un nuage de points, grossièrement représentatif de la forme "exacte" du squelette, comme illustré sur la figure 2.8.

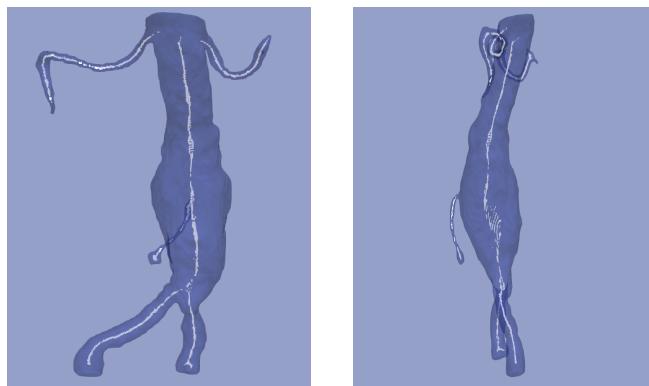


FIGURE 2.8 – Nuage de point cible extrait de la géométrie 3D initiale

2.4.2.5 Discussion sur les valeurs utilisées

Volontairement, nous n'avons pas mentionné de valeurs pour la taille d'un voxel ainsi que pour le seuil que nous utilisons lors de l'extraction du nuage de points. En effet, savoir choisir les bonnes valeurs pour ces paramètres se révèle être capital pour la suite du travail. Pour effectuer le meilleur recalage possible, il faut donc que le nuage de points cible contienne suffisamment de données. Cela induit donc d'effectuer la voxelisation la plus fine possible. Cependant, plus la voxelisation est fine, plus le nombre de points est important et, par conséquent, plus le calcul du Fast-Marching est long et coûteux.

Ces valeurs n'ont pas fait l'objet d'une étude plus approfondie et sont fixées selon notre expérience. Généralement, pour les géométries complexes (annexe B), la taille des voxels (cubiques) est fixée à 0.8mm et le seuil à 0.7 (i.e on prend tous les points dont la valeur de courbure est supérieur à 70% de la courbure maximum). Exceptionnellement, la géométrie test illsutrée dans le corps de ce mémoire, significativement plus petite que les autres, a une taille de voxels de 0.4mm.

2.4.3 Étape 2 : Reconstruction du squelette

Cette étape est cruciale pour la suite du travail et son enjeu est double.

Tout d'abord, l'intégralité de la méthode de recalage est basée sur la notion de courbes paramétrées de type B-Spline (quelques éléments disponibles dans [1]). Il faut donc, dans un premier temps, reconstruire le squelette sous la forme d'un ensemble de courbes paramétrées. L'inconnu dans notre problème de minimisation étant un vecteur de déplacement des points de contrôle, il est également préférable d'en avoir un minimum afin que le calcul soit le plus rapide possible. Cependant, il faut aussi faire attention à ce qu'il y en ait assez pour conserver la topologie (courbure, torsion, longueur..) initiale du squelette fourni. Un des objectifs sera donc de trouver le bon nombre de points de contrôle à sortir pour chaque branche.

Ensuite, nous avons la problématique des points de bifurcation. Lors du racalge nous allons déplacer les courbes B-Splines indépendamment ce qui va disloquer le squelette. La dernière étape sera donc de le reconstruire. Ainsi, il faudra différencier les branches qui se connectent (par exemple les branches

qui vont former l'aorte centrale) et celles qui vont se projeter (par exemple les artères rénales ou lesiliaques). En particulier, ce sont ces dernières qui feront bouger les points de bifurcation.

Lorsque nous analysons le squelette fourni par NUREA on se rend compte qu'un point de bifurcation possède tous les labels des branches arrivant sur cette bifurcation. Cependant, un seul de ses labels porte la mention "bifurcation" et il ne s'agit, en général, pas de la bonne branche. Il faut donc que nous trouvions un moyen, pour chaque bifurcation, d'identifier proprement quels sont les points dits *connectés* et quels sont ceux à projeter.

Pour répondre aux questions précédentes, nous proposons une reconstruction du squelette de la façon suivante :

1. Nous commençons par créer, pour chaque branche (i.e label), une courbe paramétrée de type B-Spline dont les points de contrôle sont les points du squelette initial. Plus précisément, chaque branche s'exprime par la fonction :

$$C(t) = \sum_i B_i(t) P_i$$

où $B_i(t)$ représente la fonction de base pour le paramètre t et P_i le ième point de contrôle. Sur chaque branche, on calcule également le repère de Frénet, la courbure et la longueur d'arc (voir annexe A).

2. On calcule le nombre minimum de points de contrôle par branche. Pour cela, on cherche, pour chaque branche, le produit entre sa courbure moyenne et sa longueur totale. Le résultat évalué est linéairement convertit dans un intervalle entre 3 et 10. Cela signifie que le nombre de points de contrôle varie entre 3 et 10 et dépend de la longueur et de la courbure de la courbe ce qui nous assure d'avoir suffisamment de points pour suivre la topologie initialement fournie. On reconstruit les B-Splines à partir de ces nouveaux points de contrôle en gardant le même nombre de points par branche que le squelette initial.
3. Les points de continuité sont les points qui relient deux B-Splines formant la même artère alors que les points d'intersection représentent ceux où se raccrochent une ou plusieurs artères. Pour différencier ces deux types de point, nous avons utilisé la notion de tangente. Soit X un point d'intersection où n B-Splines se rejoignent et T_1, \dots, T_n les vecteurs tangents en X pour chaque courbe qui se rejoignent en X . On va alors regarder :

$$| \langle T_i, T_j \rangle - \max_{\substack{l, k \in \{1, \dots, n\} \\ l \neq k}} \langle T_l, T_k \rangle | \text{ pour tout } (i, j) \in \{1, \dots, n\} \text{ et tel que } i \neq j$$

Tous les i, j tels que cette quantité est inférieure à 0.1 (valeur déterminée par expérience sur de nombreux tests) seront des points de continuité. Les autres points seront considérés comme points d'intersections. Nous verrons dans la section 2.4.5 comment utiliser ces informations pour reconstruire le squelette. La figure 2.9 montre (a) les points de contrôle et (b) la reconstruction en courbes de type B-Spline du squelette initialement fourni.

2.4.4 Étape 3 : Recalage optimal

2.4.4.1 Préliminaires

Notre travail s'est grandement inspiré de l'article [8]. Pour commencer, nous détaillons, ci-après, un résumé de leur méthode. L'objectif est le suivant : recalier de façon optimale une courbe paramétrée de type B-Spline dans un nuage de points représentant une forme géométrique cible.

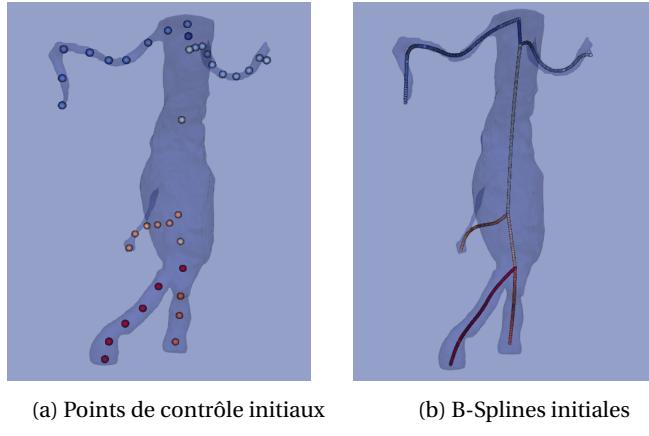


FIGURE 2.9 – Reconstruction du squelette fourni par NUREA

Soit $X_k \in \mathbb{R}^2$ ou \mathbb{R}^3 , $k = 1, \dots, n$, un nuage de points désorganisé. X_k est supposé ressembler à la forme d'une courbe inconnue, ouverte ou fermée avec la seule condition qu'elle **ne s'intersecte pas**.

Soit P_i , $i = 1, \dots, m$, les points de contrôle d'une courbe paramétrée $B(t)$ de type B-Spline dont l'expression est :

$$B(t) = \sum_{i=1}^m \phi_i(t) P_i$$

où $\phi_i(t)$ est la fonction de base au point de paramètre t , de degré fixe et vecteur de noeuds uniforme.

Considérant les X_k , on cherche les P_i telle que la fonction coût :

$$\hat{f} = \frac{1}{2} \sum_{i=1}^k \min \|B(t) - X_k\|^2 + \lambda f_s \quad (2.1)$$

soit minimale. Ici, f_s est un terme de régularisation et $\lambda > 0$ module la régularité.

Supposons que nous ayons une B-Spline initiale :

$$B(t) = \sum_{i=1}^m \phi_i(t) P_i$$

Soit maintenant $D = (D_1, \dots, D_m)$ une variable dite de mise à jour tels que les nouveaux points de contrôle s'écrivent :

$$P_+ = P + D$$

et donc la nouvelle courbe B-Spline :

$$B_+(t) = \sum_{i=1}^m \phi_i(t)(P_i + D_i) = \sum_{i=1}^m \phi_i(t)P_{i,+}$$

Généralement, nous avons $n \neq m$ (i.e le nombre de points dans le nuage est différent du nombre de points dans la courbe initiale). Il faut associer à chaque X_k , un point de B . Pour cela on va simplement sélectionner, pour chaque X_k , son point le plus proche dans la courbe (i.e lui associer un t_k). De cette façon, on peut réécrire l'équation (2.1) et nous obtenons la nouvelle fonction coût :

$$\hat{f} = \frac{1}{2} \sum_k \|B_+(t_k) - X_k\|^2 + \lambda f_s \quad (2.2)$$

La minimisation de cette fonction permet de trouver le vecteur D et de mettre à jour la nouvelle B-Spline.

$B(t_k)$ est appelé un *footpoint* de X_k et on s'intéresse à l'erreur (appelée "point distance error term") :

$$e = \|B_+(t_k) - X_k\|$$

La minimisation de cette fonction et l'optimalité du résultat trouvé dépend en partie du positionnement initial de la courbe, surtout via le calcul des footpoints. Pour améliorer cela, nous effectuons plusieurs itérations afin d'arriver à une erreur qui n'évolue plus (c'est à dire que les footpoints calculés sont toujours les mêmes).

Nous montrons sur la figure 2.10 un test que nous avons effectué en 2D. La figure 2.11 montre l'évolution de l'erreur pour 20 itérations. On remarque que, dès la troisième itération, l'erreur n'évolue plus et nous avons trouvé notre recalage optimal.

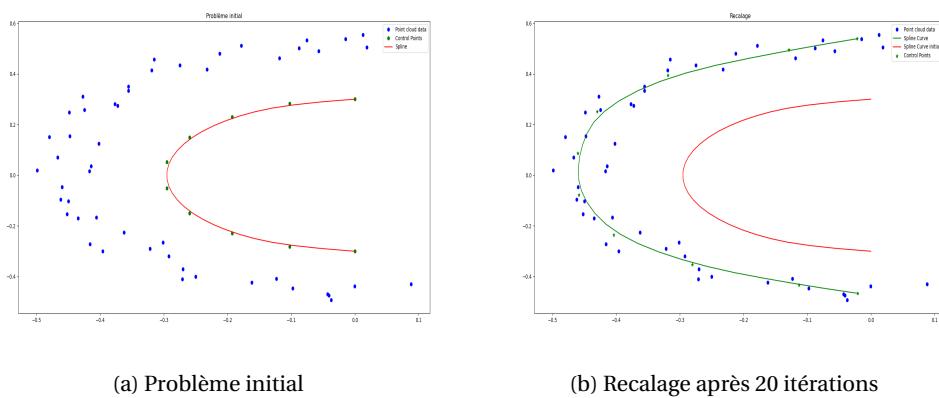


FIGURE 2.10 – Recalage d'une courbe B-Spline dans un nuage de points

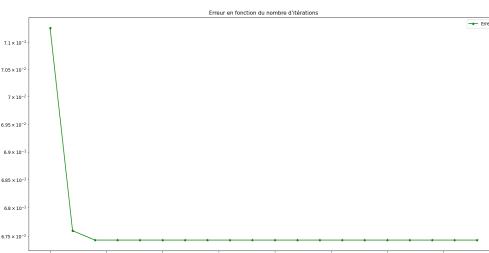


FIGURE 2.11 – Erreur en fonction du nombre d'itérations

2.4.4.2 Description de notre méthode

Considérant les géométries que l'on a, nous allons essayer d'utiliser la méthode décrite précédemment pour l'adapter, au mieux, à notre problème. Le squelette initial reconstruit dans la section 2.4.3 nous donne tous les outils nécessaires pour y parvenir. La difficulté majeure se trouve dans le fait que l'on considère plusieurs courbes B-Splines qui s'intersectent plutôt qu'une seule.

Notre stratégie est la suivante : nous allons créer une fonction coût, dans le même esprit que 2.2, qui va minimiser une erreur globale entre toutes les B-Splines du squelette reconstruit et le nuage de point cible. Cependant, le vecteur de mise à jour D , bien qu'il soit calculé sur la globalité du nuage de points, impliquera un déplacement local des courbes B-Splines. En procédant comme tel, on observera une dissociation des branches, que l'on corrigera dans la section 2.4.5. Un des avantages de la méthode est sa généralisation pour n'importe quelle dimension. Dans notre cas, nous nous focalisons évidemment sur la 3D.

2.4.4.3 Structure de données initiales

La construction bien particulière du squelette dans 2.4.3 n'est pas anodine et nous permet de ressortir une structure de données adaptées à notre problème. De cette façon, nous avons les données initiales suivantes :

Soit $X_k \in R^N$ et $k = \{1, \dots, n\}$ le nuage de points cible. Dans notre cas, $N = 3$ et la cible est extraite de 2.4.2.

Les données initiales suivantes sont extraites de 2.4.3. Soit $I = \{1, \dots, m\}$ un intervalle de \mathbb{N} qui représente les labels de la segmentation.

Soit une liste P_l , $l \in I$ de points de contrôle tel que :

- γ_l est le nombre de points de contrôle par branche l .
- $\forall l \in I, P_l \in \mathcal{M}_{(\gamma_l, N)}$.
- $\Gamma = \sum_l \gamma_l$ est le nombre total de points de contrôle dans tout le squelette.

Soit une liste B_l , $l \in I$ de B-Splines initiales tel que :

- δ_l est le nombre de points de la B-Spline d'indice l .
- $\forall l \in I, B_l \in \mathcal{M}_{(\delta_l, N)}$
- $\Delta = \sum_{l \in I} \delta_l$ est le nombre total de points dans le squelette.

Soit K_l et T_l deux listes représentant, respectivement, les vecteurs de noeud et les paramètres de discré-tisation $t \in [0, 1]$ pour chaque branche l . Le degré est fixé en début de l'algorithme (généralement à 2 ou 3).

Les listes sont **ordonnées**. C'est à dire que la B-Spline d'indice l est créée grâce au vecteur de noeud, des points de contrôle et des paramètres de discré-tisation d'indice l dans leur liste respective. De cette façon :

$$\forall l \in I, \quad B_l(t) = \sum_{i=1}^{\gamma_l} \phi_{i,l}(t) P_{i,l}$$

où $\phi_{i,l}(t)$ est la fonction de base associée à K_l et au degré fixé. La connaissance de tous les paramètres de discré-tisation de T_l ainsi que des vecteurs de noeuds nous permettent également une écriture de la B-Spline sous forme matricielle $B_l = \Phi_l P_l$.

2.4.4.4 Matrice des fonctions de base des footpoints

La première étape est de parvenir à associer, pour chaque point de X_k , son point le plus proche parmi tous les points de toutes les B-Splines. On cherche à ordonner dans une matrice F ce que l'on va appeler les fonctions de base des *footpoints*.

Tout d'abord, on introduit la matrice C regroupant tous les points de contrôle du squelette, ordonnée de la façon suivante :

$$C = \begin{bmatrix} P_{1,1} \\ P_{1,2} \\ \vdots \\ P_{1,\gamma_1} \\ P_{2,1} \\ \vdots \\ P_{m,\gamma_m} \end{bmatrix} \in \mathcal{M}_{(\Gamma, N)}$$

En particulier, le but est de construire une matrice F tel que l'on ait $B(t_k) = FC$. La matrice finale F

aura donc autant de lignes qu'il y a de points dans le nuage et autant de colonnes qu'il y a de points de contrôle dans tout le squelette.

Soit X_i un point du nuage cible. On cherche, parmi toutes les B-Splines du squelette reconstruit, son point le plus proche Y_i . Une fois identifié, on extrait son label (i.e son " l ") et, comme toutes les listes sont ordonnées, nous sommes en mesure de retrouver toutes les informations nécessaires pour construire la fonction de base associée à la construction initiale de Y_i (vecteur de noeud, paramètre de discréétisation, points de contrôle, etc...). Il suffit, pour finir, de placer la fonction au bon endroit dans la matrice F , à savoir par rapport aux points de contrôle de la matrice C .

Prenons un exemple : On regarde deux points du nuage cible X_1 et X_2 .

Supposons que le point le plus proche de X_1 soit un point de la B-Spline d'indice $l = 2$. On peut identifier le vecteur de noeud, le point de discréétisation t_1 , les points de contrôle qui le gouvernent et, par conséquent, trouver les $\phi_{i,1}(t_1)$, $i \in \{1, \dots, \gamma_1\}$

Supposons que le point le plus proche de X_2 soit un point de la B-Spline d'indice $l = 1$. On peut identifier le vecteur de noeud, le point de discréétisation t_2 , les points de contrôle qui le gouvernent et, par conséquent, trouver les $\phi_{j,2}(t_2)$, $j \in \{1, \dots, \gamma_2\}$

Ainsi, on remplit la matrice de la façon suivante, où $F \in \mathcal{M}_{(n,\Gamma)}$ et $C \in \mathcal{M}_{(\Gamma,N)}$:

$$F = \begin{bmatrix} 0 & \dots & 0 & \phi_{1,2}(t_2) & \dots & \phi_{\gamma_2,2}(t_2) & 0 & \dots & 0 \\ \phi_{1,1}(t_1) & \dots & \phi_{\gamma_1,1}(t_1) & 0 & \dots & \dots & \dots & \dots & 0 \\ \vdots & \dots & \vdots \\ \vdots & \dots & \vdots \\ \vdots & \dots & \vdots \end{bmatrix} \begin{bmatrix} P_{1,1} \\ \vdots \\ P_{1,\gamma_1} \\ P_{2,1} \\ \vdots \\ P_{m,\gamma_m} \end{bmatrix} = C$$

2.4.4.5 Fonction de coût

On créée une variable de mise à jour $D \in \mathcal{M}_{(\Gamma,N)}$ et on note $\tilde{D} \in \mathbb{R}^{N\Gamma}$. On réécrit les variables :

$$\tilde{F} = \begin{bmatrix} F & 0 & 0 \\ 0 & F & 0 \\ 0 & 0 & F \end{bmatrix} \in \mathcal{M}_{(Nn,N\Gamma)}, \quad \tilde{C} = \begin{bmatrix} C_x \\ C_y \\ C_z \end{bmatrix} \in \mathbb{R}^{N\Gamma}$$

De cette façon on peut réécrire le problème de la façon suivante :

$$\hat{f}(\tilde{D}) = \frac{1}{2} \|\tilde{F}(\tilde{C} + \tilde{D}) - X_k\|^2 + \frac{\lambda}{2} f_s$$

Le terme f_s est un terme nous permettant de contrôler la rigidité des B-Splines lors du recalage. Il est défini comme étant la norme de la dérivée seconde des B-Splines après recalage, régulée par le paramètre $\lambda \geq 0$.

Dans la section 2.4.3, on extrait également V_l , $l \in I$ une liste composée, pour une branche l , de la dérivée seconde des fonctions de base, tel que $\forall l \in I$, $V_l = \ddot{\Phi}_l$ et de cette façon $\ddot{B}_l = \ddot{\Phi}_l P_l = V_l P_l$.

On écrit :

$$V = \begin{bmatrix} V_1 & 0 & \dots & \dots & 0 \\ 0 & V_2 & 0 & \dots & 0 \\ 0 & \dots & \ddots & \dots & 0 \\ 0 & \dots & \dots & \ddots & 0 \\ 0 & \dots & \dots & \dots & V_m \end{bmatrix} \in \mathcal{M}_{(\Delta,\Gamma)} \text{ et } \tilde{V} = \begin{bmatrix} V & 0 & 0 \\ 0 & V & 0 \\ 0 & 0 & V \end{bmatrix} \in \mathcal{M}_{(N\Delta,N\Gamma)}$$

On a alors :

$$f_s = \frac{\lambda}{2} \|\tilde{V}(\tilde{C} + \tilde{D})\|^2$$

Le fonction coût à minimiser devient :

$$\hat{f}(\tilde{D}) = \frac{1}{2} \|\tilde{F}(\tilde{C} + \tilde{D}) - X_k\|^2 + \frac{\lambda}{2} \|\tilde{V}(\tilde{C} + \tilde{D})\|^2 \quad (2.3)$$

En réécrivant l'équation 2.3, le problème s'écrit : Trouver $\bar{D} \in \mathbb{R}^{NT}$ tel que :

$$\bar{D} = \underset{\tilde{D} \in \mathbb{R}^{NT}}{\operatorname{argmin}} \frac{1}{2} \|\tilde{F}\tilde{D} - (X_k - \tilde{F}\tilde{C})\|^2 + \frac{\lambda}{2} \|\tilde{V}\tilde{D} + \tilde{V}\tilde{C}\|^2 \quad (2.4)$$

Maintenant, nous allons essayer d'écrire ce problème sous la forme d'un problème des *moindres carrés*. Pour cela, on écrit :

$$A = \begin{bmatrix} \tilde{F} \\ \sqrt{\lambda} \tilde{V} \end{bmatrix} \in \mathcal{M}_{(N(n+\Delta), NT)} \quad \text{et} \quad b = \begin{bmatrix} X_k - \tilde{F}\tilde{C} \\ -\sqrt{\lambda} \tilde{V}\tilde{C} \end{bmatrix} \in \mathbb{R}^{NT}$$

et le problème 2.4 se réécrit : Trouver $\bar{D} \in \mathbb{R}^{NT}$ tel que :

$$\bar{D} = \underset{\tilde{D} \in \mathbb{R}^{NT}}{\operatorname{argmin}} \|A\tilde{D} - b\|^2 \quad (2.5)$$

On sait que la solution de ce problème est équivalente à la résolution des équations normales :

$${}^t A A \bar{D} = {}^t A b$$

Bien généralement, la matrice A n'est pas de rang maximal. Cela provient en partie du fait que F ne l'est pas obligatoirement. Lors du calcul des footpoints, deux points du nuage peuvent être associés au même point parmi toutes les B-Splines. De plus, il est également possible que V ne soit pas de rang maximal, c'est à dire que la dérivée seconde des fonctions de base pour 2 points différents soient identiques (vecteur de noeuds uniformes, paramètre de discréétisation identiques...).

Si A est de rang maximal ($= NT$) alors le problème possède une unique solution, la matrice ${}^t A A$ est inversible et

$$\bar{D} = ({}^t A A)^{-1} {}^t A b$$

Si A n'est pas de rang maximal (i.e $< NT$), alors il existe une infinité de solutions. Dans la littérature, on considère que, parmi toutes les solutions possibles, celle offerte par la SVD est *optimale* dans le sens où elle minimise la norme du vecteur solution.

On commence par calculer la SVD de $A \in \mathcal{M}_{(N(n+\Delta), NT)}$ que l'on suppose de rang $r < NT$:

$$A = U \Sigma V$$

où U et V sont des matrices carrées orthogonales de dimension respective $N(n + \Delta)$ et NT . Σ est une matrice pseudo-diagonale de même taille que A et telle que $\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0)$, σ_i représentant les valeurs singulières de A . On a :

$${}^t A A = V \Sigma {}^t U U \Sigma V = V H {}^t V \quad \text{où} \quad H = {}^t \Sigma \Sigma = \operatorname{diag}(\sigma_1^2, \dots, \sigma_r^2, 0, \dots, 0)$$

En reprenant les équations normales :

$${}^t A A \bar{D} = {}^t A b \iff V H {}^t V \bar{D} = V {}^t \Sigma {}^t U b \iff H {}^t V \bar{D} = {}^t \Sigma {}^t U b$$

Comme :

$$\begin{aligned} H^{-1} {}^t \Sigma &= \text{diag}\left(\frac{1}{\sigma_1^2}, \dots, \frac{1}{\sigma_r^2}, 0, \dots, 0\right) * \text{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0) \\ &= \text{diag}\left(\frac{1}{\sigma_1}, \dots, \frac{1}{\sigma_r}, 0, \dots, 0\right) \\ &= \Sigma^+ \end{aligned}$$

On obtient :

$${}^t V \bar{D} = \Sigma^+ {}^t U b$$

Et finalement on obtient la solution optimale de notre problème 2.5 :

$$\bar{D} = V \Sigma^+ {}^t U b$$

La figure 2.12 montre le recalage du squelette reconstruit au bout de 5 itérations (on expliquera l'évolution de l'erreur dans 2.4.5). On remarque que les branches se sont proprement déplacées mais qu'elles se sont séparées (de peu), comme l'on pouvait s'y attendre.

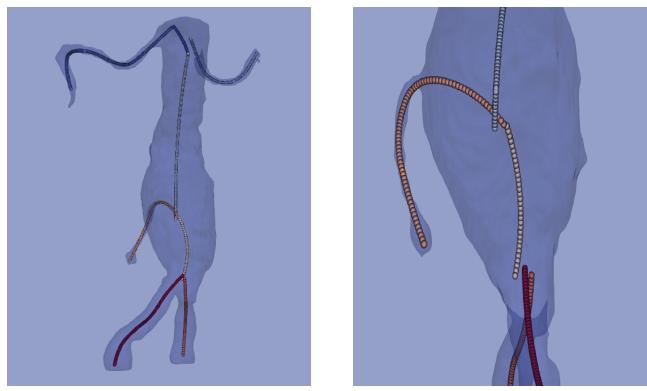


FIGURE 2.12 – Recalage d'une courbe B-Spline dans un nuage de points

2.4.5 Étape 4 : Mise à jour et évolution de l'erreur

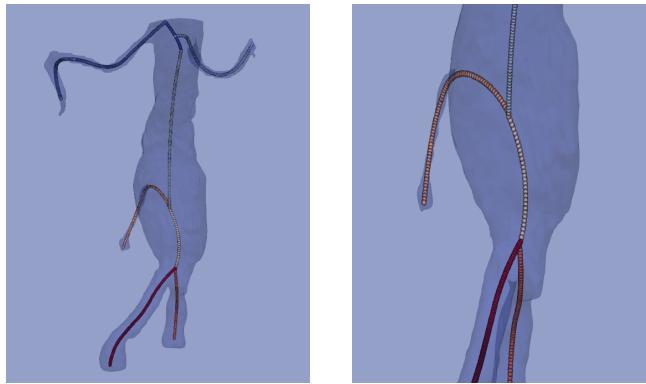
2.4.5.1 Mise à jour

Comme nous l'avons constaté dans 2.4.4, les branches du squelette après le recalage sont dissociées : nous devons donc proprement le restructurer. Pour cela, grâce à l'étape de reconstruction 2.4.3, nous savons, pour chacun des deux points extrêmes d'une branche, s'il sont des points de *continuité*, de *bifurcation* ou bien *extrême* du squelette. En vérité, seules les deux notions de continuité et de bifurcation nous intéressent.

Si deux points sont *connectés* (s'ils représentent la même artère par exemple), alors on crée un nouveau point définit comme leur barycentre. Puis, on relie les branches concernées par ce point barycentrique.

À l'inverse, si deux (ou plusieurs) points sont de *bifurcation*, on cherche, pour chacun d'eux, leur point le plus proche, parmi tous les points de toutes les B-Splines. Une fois identifié, on remplace ce point de bifurcation par le point de la branche la plus proche et on reconstruit les courbes.

De cette manière, on reconstruit, au mieux, le squelette après son recalage, comme illustré dans la figure 2.13.



(a) Squelette après le recalage

(b) Zoom de (a)

FIGURE 2.13 – Restructuration du squelette après recalage

2.4.5.2 Évolution de l'erreur

Dans les préliminaires 2.4.4.1, nous avions montré l'intérêt de faire évoluer l'erreur jusqu'à ce que les footpoints calculés soient toujours les mêmes (que l'erreur 2.4.4.1 n'évolue plus). Nous verrons que cela n'est pas optimal dans notre cas et proposons deux approches pour effectuer nos itérations.

La première est de reproduire exactement ce que nous faisions pour recalier une seule B-Spline dans un nuage de point, c'est à dire itérer le recalage optimal tant que la différence d'erreur entre deux itérations soit inférieure à un seuil de tolérance. Cette approche est fonctionnelle mais n'est pas à nos yeux optimale. En effet, plus on itère, plus les branches (qui restent dissociées entre deux itérations) ont tendance à se séparer. Cela est criant lorsque l'on manque de points dans le nuage. En effet, si lors du calcul des footpoints, il manque des points dans le nuage, certaines branches ont tendance à naturellement être attirées par des points éloignés qui, lors de la reconstruction, entraîneront des incohérences topologiques, bien que l'erreur finale soit minime.

L'autre approche que nous proposons est de restructurer le squelette **à chaque itération**. Lorsque nous faisons cela, on force l'algorithme à redémarrer d'un squelette parfaitement associé et les calculs via les prochains footpoints seront plus cohérents par rapport à la topologie générale du squelette. À l'inverse, l'erreur entre deux itérations ne sera donc pas forcément décroissante, étant donné que l'on change sa forme initiale. Cependant, nous avons remarqué qu'en appliquant cette méthode entre 5 et 10 itérations, nous obtenons un squelette proprement déplacé. C'est pourquoi nous privilégeons cette approche.

La figure 2.14 illustre la différence entre le squelette initial après sa reconstruction (vert) et le squelette recalé (rouge).

2.5 Limites et temps de calculs

Cette méthode s'est avérée robuste sur l'ensemble des 6 géométries dont nous étions en possession. Des illustrations complémentaires pour des géométries plus complexes que notre figure "test" sont disponibles en annexe B. Nous proposons, dans cette section, d'exposer ses limites actuelles ainsi que quelques axes de recherche à développer.

Tout d'abord, un bon recalage dépend avant tout d'un bon nuage de point. Cela signifie trouver une valeur pour la taille des voxels qui permette, à la fois de garder autant d'informations possible sur les branches étroites, et de pouvoir calculer le Fast-Marching le plus rapidement possible. Une autre solution pour pouvoir baisser la taille des voxels serait d'avoir une géométrie initiale fournie plus petite (la couper au maximum en longueur et largeur).

Ensuite, il est important que le squelette fourni soit cohérent par rapport à la géométrie 3D. En effet,

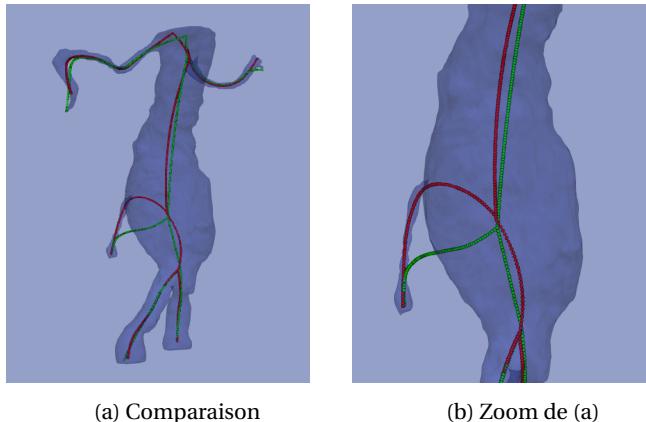


FIGURE 2.14 – Différence entre squelette initial (vert) et recalé (rouge)

le but est de recaler le squelette ce qui induit, par exemple, d'avoir le bon nombre de branches à l'origine. Si ce n'est pas le cas, certaines branches sont attirées par des points du nuage qui ne leur sont pas destinées, entraînant un mauvais recalage (erreur provoquée sur une géométrie que nous avons reçue).

Certains axes à développer seraient donc l'étude des paramètres optimaux pour la création d'un bon nuage de points, une amélioration de la rapidité de calcul du Fast-Marching (notre méthode est issue d'une librairie PYTHON...probablement pas optimale!).

Nous pourrions également penser à remplacer l'étape de mise à jour 2.4.5 du squelette par un ensemble de contraintes dans le problème de minimisation (recaler le squelette de manière optimale en contrignant ses branches à rester connecté ... tout en autorisant les points de bifurcation à bouger librement).

Pour finir, les calculs prennent généralement peu de temps. L'étape de création du nuage est clairement la plus longue, particulièrement à cause du Fast-Marching qui, si le nombre de points est élevé peut prendre plusieurs minutes, voir arrêter l'algorithme. Une itération dure maximum 2 secondes pour une géométrie complexe avec de nombreux points de contrôle (=inconnus du système).

Dans le cas de notre petite géométrie test, le temps total du calcul est de 21.66 secondes. Pour une géométrie plus complexe, elle est de 50 secondes avec une taille de voxels de 0.8 et 10 itérations.

Finalement, le paramètre de rigidité λ est fixé à $1.e - 6$. Il est nécessaire dans le sens où il permet de ne pas trop accentuer les courbes lors du recalage et contraint les branches à ne pas rechercher des points "lointains", problématique lorsqu'il manque des points dans le nuage.

CHAPITRE 3

Espace d'anévrismes

L'objectif de cette section est de créer une (des) base(s) réduite(s) pour la génération d'une infinité d'anévrismes aortiques. Pour cela, il faut, dans un premier temps, que nous soyons en mesure de reconstruire précisément les géométries initiales, ce qui est développé dans (3.1).

Puis, en s'appuyant sur les données extraites des paramétrisations d'un ensemble de géométries 3D fournies, nous utilisons une méthode de réduction de dimension pour construire des bases orthonormées synthétisant les informations brutes initiales. Par combinaison linéaire des vecteurs de ces bases, nous serons alors en mesure de générer, aléatoirement, une infinité d'anévrismes aortiques, comme présenté dans 3.2.

Dans ce chapitre, on s'intéresse essentiellement à l'endroit où se trouve l'anévrisme aortique, c'est à dire entre les artères rénales et lesiliaques. Après avoir effectué notre méthode de recalage sur l'ensemble des géométries, nous extrayons manuellement les points de contrôle des labels (i.e branches) représentatifs de la partie "rénales-iliaque". Si plusieurs labels forment cette partie, on peut penser effectuer une seconde méthode de recalage afin d'obtenir une centerline lisse.

3.1 Paramétrisation de l'anévrisme aortique

La paramétrisation de la paroi aortique va s'articuler autour de deux notions.

Tout d'abord, nous pouvons reconstruire la centerline grâce aux points de contrôles. En effet, pour $t \in [0, 1]$, nous avons les coordonnées $X(t)$, $Y(t)$ et $Z(t)$ de la centerline.

Puis, la connaissance de ces points de coordonnées et de la géométrie 3D initiale va nous permettre de construire une fonction $R_t(\theta)$ représentative de la paroi de l'anévrisme aortique étudiée au point t . La dernière étape cherchera comment évaluer l'erreur entre notre reconstruction et la géométrie initiale.

3.1.1 Approximation de la paroi

Pour être en mesure d'approximer la paroi de l'anévrisme aortique nous commençons par calculer, pour tout t , son repère de FRENET (annexe A). Puis, nous regardons l'intersection entre le plan formé par le

couple de vecteur *Normal - Binormal* à un point t (que l'on nommera plan N-B) et la paroi de l'anévrisme aortique.

La première difficulté se pose lorsque l'on effectue la coupe. En effet, la fonction que nous utilisons renvoie tous les points d'intersection entre le plan et la géométrie, ce qui implique que si ce plan intersecte une artère annexe, la fonction nous renverra également ses points d'intersection. Nous devons donc faire un tri et différencier les points l'aorte principale et ceux des branches annexes.

Nous développons dans 3.1.1.1 une méthode provisoire en attendant que la startup NUREA présente un algorithme plus robuste.

3.1.1.1 Extraction des contours de l'aorte principale

Pour tout point t de la centerline, nous lui associons les points issus de l'intersection du plan N-B avec la géométrie initiale.

Soit $X_t = (x_t, y_t, z_t)$ le point de la centerline pour le paramètre t et $N_t = (Nx_t, Ny_t, Nz_t)$, $B_t = (Bx_t, By_t, Bz_t)$, $T_t = (Tx_t, Ty_t, Tz_t)$ les vecteurs normaux, binormaux et tangents en ce point de la courbe. Soient $X_t^1 = (x_t^1, y_t^1, z_t^1), \dots, X_t^n = (x_t^n, y_t^n, z_t^n)$ les points d'intersection de ce plan avec la géométrie initiale.

Nous construisons la matrice de changement de base P et celle des coordonnées X :

$$P = \begin{bmatrix} Nx_t & Bx_t & Tx_t \\ Ny_t & By_t & Ty_t \\ Nz_t & Bz_t & Tz_t \end{bmatrix}, \quad X = \begin{bmatrix} x_t & y_t & z_t \\ x_t^1 & y_t^1 & z_t^1 \\ \vdots & \vdots & \vdots \\ x_t^n & y_t^n & z_t^n \end{bmatrix}$$

On appelle Y la matrice des coordonnées de X dans la nouvelle base tel que $Y = XP$. Nous utilisons ensuite un algorithme d'analyse en composante principale (détailé dans 3.2.2) afin d'obtenir une nouvelle base expliquant au mieux la dispersion des données et l'on projette Y dans cette nouvelle base pour obtenir :

$$\tilde{Y} = \begin{bmatrix} \tilde{x}_t & \tilde{y}_t & \tilde{z}_t \\ \tilde{x}_t^1 & \tilde{y}_t^1 & \tilde{z}_t^1 \\ \vdots & \vdots & \vdots \\ \tilde{x}_t^n & \tilde{y}_t^n & \tilde{z}_t^n \end{bmatrix}$$

La troisième coordonnées restant identique par le fait que nous sommes dans le plan N-B, nous discrétonnons uniquement les axes \tilde{x} et \tilde{y} et on regarde les fonctions de densités (i.e on compte combien nous avons de points pour chaque discrétisation dans les deux directions) :

$$f(x) = \sum_{x_i=\tilde{x}_t} \mathbb{1}_{[\min \tilde{x}_t, x]}(x_i) \quad \text{et} \quad g(y) = \sum_{y_i=\tilde{y}_t} \mathbb{1}_{[\min \tilde{y}_t, y]}(y_i)$$

En dérivant ces fonctions, on s'aperçoit que les points où la dérivée s'annule sont ceux correspondants aux "trous" entre les cercles initiaux. On calcule la distance moyenne entre chaque section (bloc où la dérivée est non nulle) et le point central $(\tilde{x}_t, \tilde{y}_t, \tilde{z}_t)$. Nous supprimons tous les points correspondants au bloc dont la distance moyenne est la plus grande. On itère ce processus, en regardant, à chaque fois les deux directions en x et y jusqu'à ce que la dérivée des fonctions de densité ne soit plus nulle (i.e il ne reste plus qu'un cercle qui est celui correspondant à l'aorte principale).

3.1.1.2 Paramétrisation de la paroi

Après avoir appliqué la méthode précédente, nous avons, pour chaque point de la centerline, les coordonnées des points de la paroi issus de l'intersection entre le plan N-B à ce point et la géométrie initiale.

On cherche à construire une fonction $R_t(\theta)$ qui va approximer la paroi. Plus précisément, R représente la norme du vecteur, dans le plan N-B, centré en le point de la centerline et d'angle θ par rapport au vecteur normal.

Soit $X_t = (x_t, y_t, z_t)$ le point de la centerline au paramètre t . Soit $X_t^i = (x_t^i, y_t^i, z_t^i)$, $i \in I = \{1, \dots, n_t\}$, les points du nuage extrait par le plan N-B.

On exprime ces points dans le plan N-B en considérant la matrice de passage de la base canonique au repère de FRENET (N-B-T) au point t :

$$P = \begin{bmatrix} Nx_t & Bx_t & Tx_t \\ Ny_t & By_t & Ty_t \\ Nz_t & Bz_t & Tz_t \end{bmatrix}$$

On obtient les nouveaux points $\tilde{X}_t = XP$ et $\tilde{X}_t^i = X_t^i P$. On remarque que la troisième colonne est identique, ce qui est normal car l'on se place dans le plan N-B.

Pour tout point du nuage \tilde{X}_t^i , on ressort et classe dans une matrice $F \in \mathcal{M}_{(n_t, 2)}$:

1. En première colonne, la distance entre un point \tilde{X}_t et \tilde{X}_t^i en calculant

$$R_i = \left\| \tilde{X}_t^i - \tilde{X}_t \right\|$$

2. En seconde colonne, l'angle entre le vecteur $\overrightarrow{\tilde{X}_t \tilde{X}_t^i}$ et le vecteur normal \vec{N} tel que

$$\theta_i = \arccos \left(\frac{\langle \overrightarrow{\tilde{X}_t \tilde{X}_t^i}, \vec{N} \rangle}{\left\| \overrightarrow{\tilde{X}_t \tilde{X}_t^i} \right\|} \right)$$

Nous calculons la fonction $R_t(\theta)$ en effectuant une approximation par série de FOURIER d'ordre m de F tel que :

$$R_t(\theta) = a_0 + \sum_{i=1}^m a_i \cos(i\theta) + b_i \sin(i\theta)$$

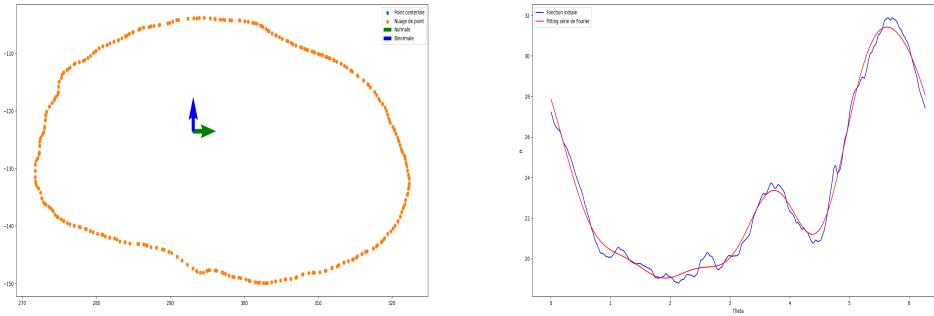
La figure 3.1 illustre en (a), dans le plan normal (vert)-binormal (bleu) à un point t , les points d'intersection entre ce plan et la géométrie initiale. En (b), nous illustrons en bleu les éléments de F liés à (a) et en rouge son approximation par série de FOURIER d'ordre 5. Nous avons choisi de reconstruire la courbe rouge pour 150 valeurs de $\theta \in [0, 2\pi]$.

3.1.2 Reconstruction et erreur

Lorsque nous avons la fonction $R_t(\theta)$, pour tout point de paramètre t , nous sommes en mesure de reconstruire la paroi. Pour tout point t de la centerline, on se place dans le plan N-B. On effectue une rotation d'angle $\theta \in \Theta$ du vecteur normal pour obtenir un nouveau vecteur, unitaire (car le vecteur normal l'est!). On multiplie ce nouveau vecteur par la distance R associée à l'angle θ ce qui nous permet de déterminer le point de coordonnée de la paroi dans le plan N-B. On effectue le passage de la base du repère de FRENET à la base canonique via tP et on peut, en suivant ce procédé, décrire toute la paroi.

La figure 3.2 représente en (b) la paramétrisation de l'anévrisme (a) dans la zone "rénale-iliaque". Nous avons utilisé 200 points de discréttisation entre 0 et 1 (i.e 200 coupes), 300 θ entre 0 et 2π et une approximation de FOURIER à l'ordre 5. Le temps total pour le calcul de cette paramétrisation est d'environ 75 secondes.

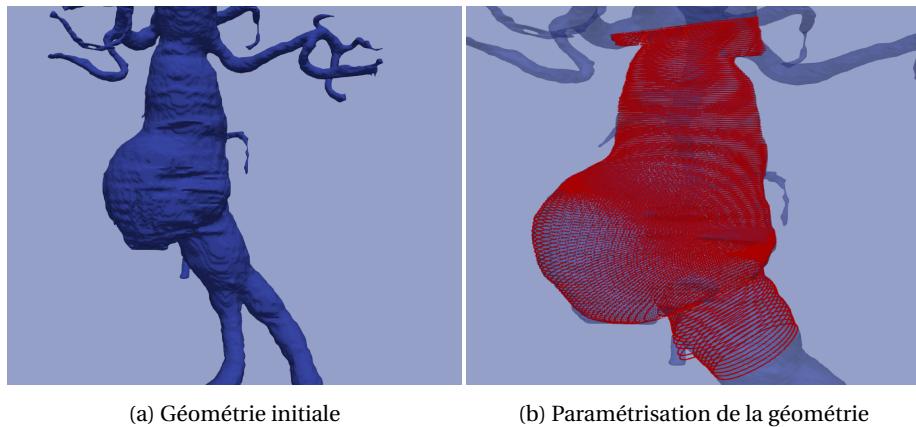
Nous pouvons nous poser la question de déterminer l'erreur entre la géométrie initiale et notre reconstruction. Pour cela, nous proposons une méthode, pas forcément optimale mais qui permet de se rendre



(a) Intersection plan N-B et géométrie

(b) Approximation par série de Fourier de F

FIGURE 3.1 – Calcul du $R_t(\theta)$



(a) Géométrie initiale

(b) Paramétrisation de la géométrie

FIGURE 3.2 – Résultat final paramétrisation

compte de la qualité de notre reconstruction. Nous voxelisons, finement, la paroi originale et, grâce à une fonction fournie par la librairie PYTHON, nous regardons le pourcentage de voxels atteints lors de la reconstruction.

Cette méthode n'est pas optimale dans le sens où l'extraction de la partie 'rénale-iliaque", n'est pas parfaite, pour les mêmes raisons que 3.1.1.1. Cela induit des voxels (même s'ils sont en faibles quantités) non liés à la partie qui nous intéresse. De plus, cela dépend également de la finesse de la voxelisation. Plus elle est fine, plus il va nous falloir de coupes pour chercher un maximum de points. Nous prenons le parti d'effectuer une voxelisation moyenne (i.e des voxels de l'ordre de 0.7mm).

Dans le figure 3.3, on s'intéresse en (a) au nombre de voxels atteint par rapport au nombre de coupes, l'ordre de l'approximation de Fourier est fixé dans ce cas à 7 et le degré d'approximation de la centerline à 3. Sans surprise, le graphe est linéaire : plus le nombre de coupes est important plus on touche un grand nombre de voxels.

Pour (b), on fixe le nombre de coupes à 100 et on regarde le coefficient de détermination moyen en fonction du nombre de modes de FOURIER. Le coefficient de détermination est une valeur statistique permettant d'apprécier la qualité d'une approximation où 1 représente la meilleure qualité possible. On remarque logiquement que, plus le nombre de modes augmente, plus l'approximation moyenne des parois est précise (tend vers 1).

En résumé, le graphique (a) regarde l'erreur sur l'intégralité de la paroi et le graphique (b) l'erreur moyenne sur une coupe.

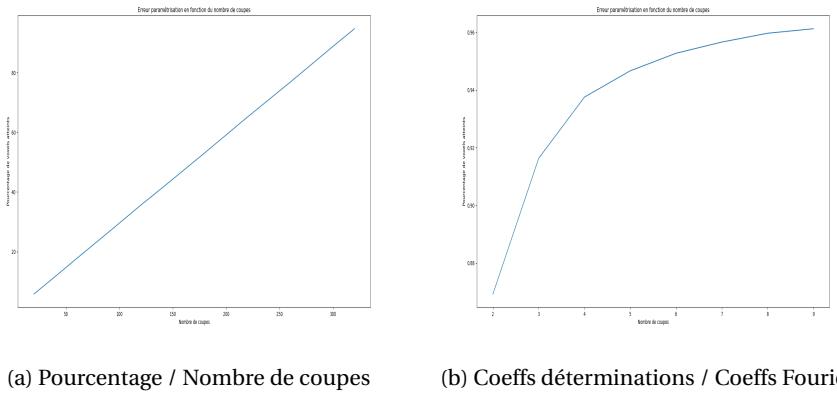


FIGURE 3.3 – Erreur sur la reconstruction

3.2 Bases réduites pour la génération d'anévrismes aortiques

La paramétrisation effectuée dans la section 3.1 nous permet de reconstituer, avec précision, la paroi d'un anévrisme aortique. Si l'on souhaite générer aléatoirement des géométries en s'appuyant sur les paramétrisations précédentes, on suppose que l'on a besoin :

- D'une base réduite pour exprimer les *coordonnées* de la centerline.
- D'une base réduite pour exprimer la *forme* de la paroi.
- D'une base réduite pour exprimer la *répartition des rayons* le long de la paroi.

L'objectif est donc de paramétriser les $N = 6$ géométries fournies, d'extraire les paramètres (données brutes) et de les organiser en 3 matrices. Chaque matrice subira une opération de réduction de dimension pour extraire une base réduite synthétisant, de la meilleure façon possible, les informations brutes initiales. De cette façon, nous obtiendrons 3 bases : une pour le calcul des coordonnées de la centerline, une pour la forme de la paroi et une pour la répartition des rayons.

3.2.1 Crédation des matrices de données brutes

3.2.1.1 Matrice pour la centerline

Tout d'abord, il est important de remarquer que les coordonnées des centerlines varient lors de l'acquisition des données (angiographie et reconstruction de la géométrie 3D). Comme on cherche à les comparer, il est nécessaire de les mettre à l'échelle et de gommer toutes les informations non pertinentes.

Pour cela, on effectue une *analyse procrasténne*. Il s'agit d'une technique d'imagerie pour la comparaison de formes géométriques. On l'utilise pour déformer un objet afin de le rendre autant que possible semblable à une référence. Cette déformation supprime les différences qui ne sont pas dues à la forme intrinsèque de l'objet. Cette transformation efface les notions de rotation, translation et procède à une mise à l'échelle. On applique cette transformation sur les points de contrôle des centerlines tout en conservant le coefficient de "scaling" pour 3.2.3.

Ensuite, nous reconstruisons intégralement toutes les centerlines (B-Splines), avec un vecteur de noeud uniforme, un degré fixe et un grand nombre de points. Cela nous permet d'effectuer une approximation polynomiale de degré n , pour chaque dimension de chaque B-Spline. Ainsi, nous obtenons les coordon-

nées de chaque centerline par :

$$\begin{aligned} X(t) &= x_0 + x_1 t + x_2 t^2 + \dots + x_n t^n \\ Y(t) &= y_0 + y_1 t + y_2 t^2 + \dots + y_n t^n \\ Z(t) &= z_0 + z_1 t + z_2 t^2 + \dots + z_n t^n \end{aligned}$$

Finalement, on crée la matrice $C \in \mathcal{M}_{(N,3n)}$ qui contient, en ligne, les coefficients des approximations polynômiales pour chaque géométrie :

$$C = \begin{bmatrix} x_{1,0} & x_{1,1} & \dots & x_{1,n} & y_{1,0} & \dots & y_{1,n} & z_{1,0} & \dots & z_{1,n} \\ \dots & \dots \\ \dots & \dots \\ \dots & \dots \\ x_{N,0} & x_{N,1} & \dots & x_{N,n} & y_{N,0} & \dots & y_{N,n} & z_{N,0} & \dots & z_{N,n} \end{bmatrix}$$

3.2.1.2 Matrice pour la paroi et la répartition des rayons

On choisit, pour toute centerline, un nombre T de coupes à effectuer. La fonction $R(t, \theta)$ est obtenue en effectuant une approximation par série de fourier d'ordre m des données extraites par le plan "Normal - Binormal" (voir???). Par exemple, pour t_1 , on a :

$$R(t_1, \theta) = a_{1,0} + \sum_{i=1}^m a_{1,i} \cos(i\theta) + b_{1,i} \sin(i\theta)$$

On crée la matrice $P \in \mathcal{M}_{(TN,m)}$ contenant, en ligne, les coefficients de la série de FOURIER pour chaque coupe.

$$P = \begin{bmatrix} a_{1,0} & a_{1,1} & \dots & a_{1,m} & b_{1,1} & \dots & b_{1,m} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{T,0} & a_{T,1} & \dots & a_{T,m} & b_{T,1} & \dots & b_{T,m} \end{bmatrix}$$

Pour la matrice de répartition des rayons, on conserve, pour chaque coupe, la valeur a_0 qui est la valeur moyenne de la série de Fourier (et donc, dans notre cas, le rayon moyen) et on crée la matrice $R \in \mathcal{M}_{(N,T)}$ où chaque ligne correspond au rayon moyen pour chaque coupe de chaque géométrie.

$$R = \begin{bmatrix} a0_{1,1} & a0_{1,2} & \dots & a0_{1,T} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a0_{N,1} & a0_{N,2} & \dots & a_{N,T} \end{bmatrix}$$

3.2.2 Réduction de dimension

Les trois matrices C , P et R regroupent les données brutes issues des paramétrisations de nos 6 anévrismes. Nous allons synthétiser ces informations afin d'extraire des vecteurs orthonormaux qui les expliquent le mieux possible. Pour cela, la méthode que nous allons utiliser est celle de l'analyse en composante principale (ACP) ou autrement appelée décomposition orthogonale aux valeurs propres (POD) en ingénierie mécanique.

L'idée centrale de l'ACP est de réduire la dimension d'un ensemble de données contenant un grand nombre de variables, en retenant le maximum de variations possibles. Cela passe par la transformation de la matrice des informations brutes en un nouvel ensemble, les composantes principales, qui ne sont

pas corrélées et ordonnées de telle sorte que les vecteurs retenus représentent la plus grande part de variation dans les données initiales.

Soit $A \in \mathcal{M}_{n,d}$ où n représente les individus (géométries, coupes...) et d les variables (coefficients dans notre cas). On suppose que A est **centrée** (i.e on a retiré, à chaque colonne sa moyenne). Supposons maintenant que l'on veuille projeter les données sur un axe v . On obtient alors le vecteur $p = Av$ où chaque élément est la longueur de la projection pour chaque point n .

On peut alors calculer la variance sur l'axe v définit par :

$$\sigma_v^2 = {}^t p p = {}^t(Av)(Av) = {}^t v {}^t A A v$$

La matrice ${}^t A A$ est appelée la matrice de *covariance de A*. Comme on souhaite maximiser la variance, on se ramène à l'étude du problème :

$$\begin{aligned} \max \sigma_v^2 &= {}^t v {}^t A A v \\ \text{s.c. } &{}^t v v = 1 \end{aligned}$$

On pose le Lagrangien du problème $\mathcal{L}(v, \lambda) = {}^t v {}^t A A v - \lambda({}^t v v - 1)$.

Que l'on résoud :

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial v} &= 0 \iff {}^t A A v = \lambda v \\ \frac{\partial \mathcal{L}}{\partial \lambda} &= 0 \iff {}^t v v = 1 \end{aligned}$$

Lorsqu'on évalue la fonction, on trouve $\sigma_v^2 = {}^t v \lambda v = \lambda$ qui est maximale pour le vecteur propre v correspondant à la plus grande valeur propre λ_1 .

Considérons maintenant un second axe w , cela nous mène à résoudre le problème, pour w de norme 1 (contrainte 1) et non corrélé à v (contrainte 2) :

$$\begin{aligned} \max \sigma_w^2 &= {}^t w {}^t A A w \\ \text{s.c. } &\begin{cases} {}^t w w = 1 \\ {}^t w v = 0 \end{cases} \end{aligned}$$

On pose le Lagrangien du problème : $\mathcal{L}(v, \lambda, \beta) = {}^t w {}^t A A w - \lambda({}^t w w - 1) - \beta {}^t w v$.

Que l'on résoud :

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w} &= 0 \iff {}^t A A w - \lambda w - \beta v \\ \frac{\partial \mathcal{L}}{\partial \lambda} &= 0 \iff {}^t w w = 1 \\ \frac{\partial \mathcal{L}}{\partial \beta} &= 0 \iff {}^t w v = 0 \end{aligned}$$

Si on multiplie la première équation par ${}^t v$, on obtient :

$${}^t v {}^t A A w - \lambda {}^t v \lambda w - \beta {}^t v \beta v = 0 \iff -\beta {}^t v v = 0 \iff \beta = 0$$

Et donc la solution du second problème est :

$${}^t A A w = \lambda w$$

où λ est la seconde plus grande valeur propre pour les mêmes raisons que précédemment.

Si l'on généralise en cherchant les d axes, les d composants principaux que l'on cherche sont les d vecteurs propres associées aux d plus grandes valeurs propres de tAA . De plus, comme les valeurs propres dans la direction des vecteurs propres représentent leur variance, on peut calculer la part de la variance de chaque vecteur propre :

$$p_{\lambda_i} = \frac{\lambda_i}{\sum_i \lambda_i}$$

Résumé :

- On centre les données, en retirant, à chaque colonne sa moyenne.
- On calcule la matrice de covariance tAA , ses valeurs propres et vecteurs propres que l'on classe dans l'ordre décroissant.
- On choisit les k vecteurs qui expliquent 95% de la variance totale.
- On crée une nouvelle matrice $\tilde{A} \in \mathcal{M}_{(d,k)}$ où chaque colonne représente un vecteur propre (ordonné par décroissance des valeurs propres).

Nous appliquons cette méthode à nos 3 matrices pour obtenir les 3 bases réduites : $\tilde{C} \in \mathcal{M}_{(3n,k_1)}$, $\tilde{P} \in \mathcal{M}_{(m,k_2)}$ et $\tilde{R} \in \mathcal{M}_{(T,k_3)}$.

3.2.3 Applications aux données

Lorsque l'on applique la méthode 3.2.2 à nos données, nous obtenons les graphiques 3.4 représentant la convergence des valeurs propres pour chaque base. Nous avons comme paramètres :

- Base centerline : 200 points de reconstruction et un degré d'approximation polynomiale de 3
- Base contour et rayon : 20 coupes par géométrie et 7 modes de Fourier

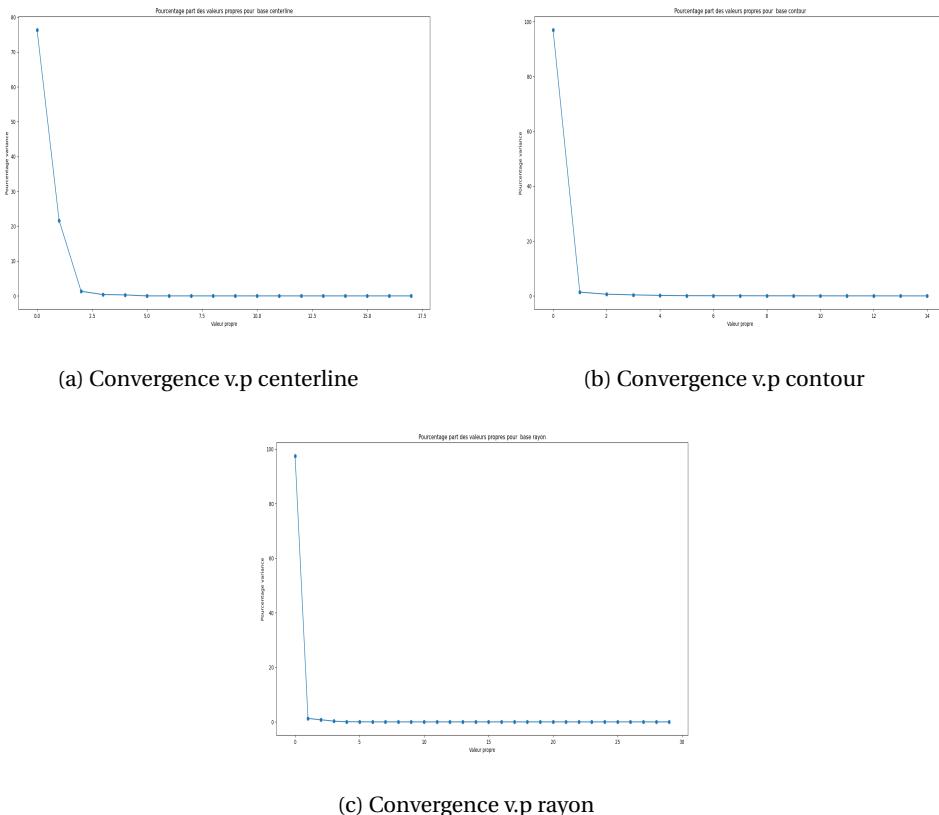


FIGURE 3.4 – Convergence des valeurs propres pour les trois bases

Finalement, nous voyons que presque une seule valeur propre est nécessaire pour synthétiser la plupart des informations. Cela est du au fait que nous n'avons pas assez de géométries différentes (seulement 6 qui se ressemblent) et que les géométries que nous utilisons ne sont pas représentatives d'anévrismes mais d'une paroi saine. On peut imaginer que dans le cas d'une grosseur, les données seront significativement différentes et les axes maximums de variances à prendre en compte seront plus nombreux.

Le temps de calcul pour la base contour est clairement la plus longue (car il faut calculer toutes les coupes pour chaque géométrie). Une fois ces bases calculées, la génération d'une géométrie est très rapide (de l'ordre de 0.35 secondes pour 200 coupes et 300 angles).

Pour finir, la figure 3.5 illustre quatres géométries générées aléatoirement par combinaison linéaire des vecteurs de base. On prendra, par géométrie, 200 points de discrétisation entre 0 et 1, un dégré d'approximation polynomiale de 3, 200 discrétisations angulaires entre 0 et 2π et 7 modes de FOURIER.

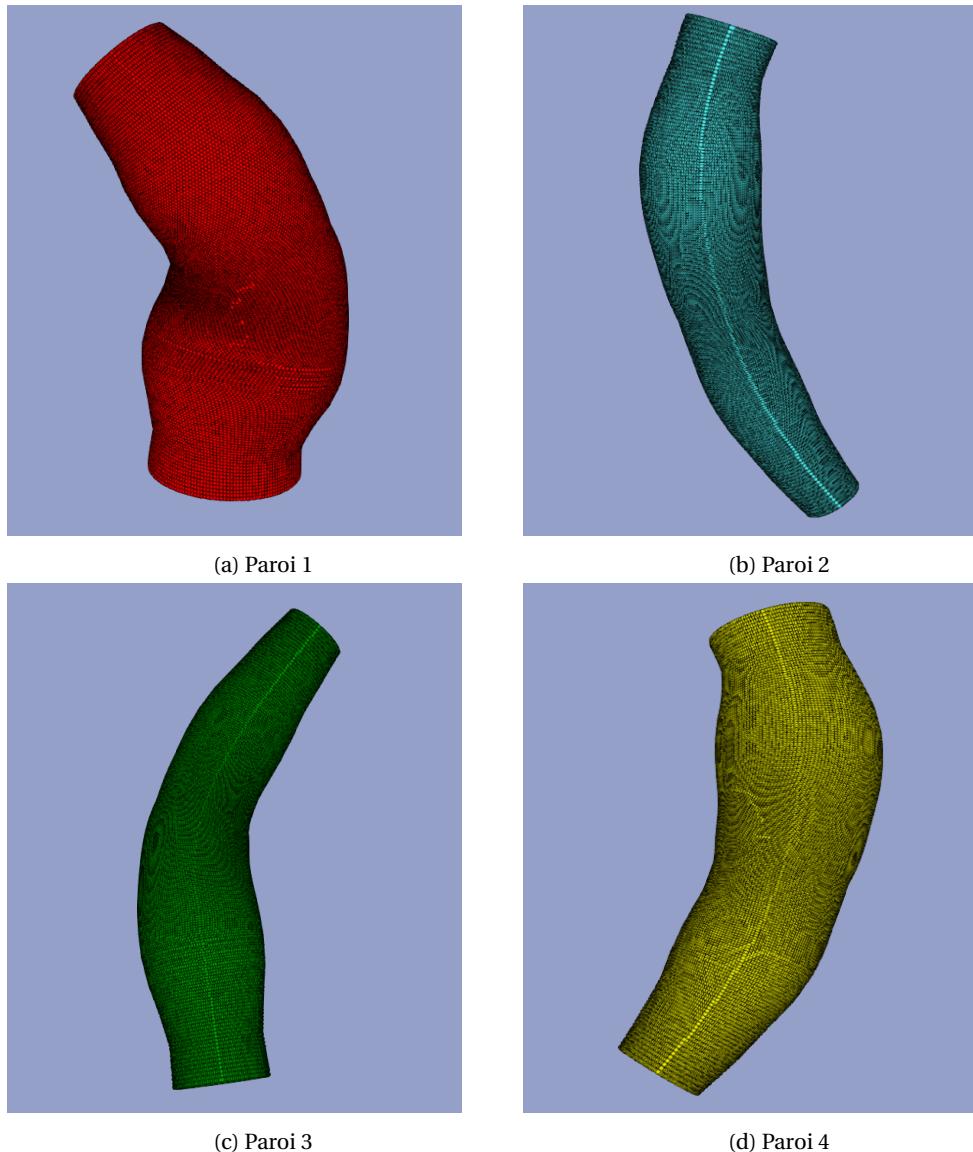


FIGURE 3.5 – 4 parois générées aléatoirement

CHAPITRE 4

Conclusion

Pour conclure, ce mémoire apporte, en 2, quelques éléments de réponse quant à savoir s'il est possible de "corriger" un squelette fourni par la startup NUREA. En résolvant un problème de minimisation dont la fonction coût est judicieusement calculée, cette méthode s'est avérée robuste sur plusieurs géométries tests. Cependant, elle se heurte encore à quelques limites qui empêchent son automatisation complète et son application directe au sein du logiciel de NUREA.

L'axe le plus important à développer serait de trouver des paramètres optimaux pour l'extraction du nuage de point cible. Préserver un maximum d'informations de la géométrie initiale tout en réduisant au minimum le temps de calcul du Fast-Marching est une étape indispensable pour un développement rapide et efficace. Il est également possible d'envisager un problème de minimisation, sous contraintes, qui permettrait, en même temps qu'effectuer le recalage, de conserver le squelette entièrement connecté.

La partie 3 développe une méthode pour la génération d'une infinité de géométries représentatives d'anévrismes aortiques. Nous avons constaté la possibilité de paramétriser une paroi de manière précise et d'utiliser ses paramètres pour créer un espace d'anévrismes.

Les applications liées à une telle méthode sont multiples. Par exemple, la possibilité de simuler un grand nombre de géométries saines et à risques, devient alors un atout lorsque l'on connaît, les capacités des algorithmes d'intelligence artificielle (deep learning, machine learning ...).

En transformant nos géométries en des maillages de surface, puis, en simulant et comparant judicieusement des données (pression sanguine, courbure, élasticité de la paroi ...) via des EDPs ou des modèles réduits, nous serions en mesure de prédire, pour une toute nouvelle géométrie, ses risques de rupture (voir [5]).

ANNEXE A

Éléments de géométrie différentielles pour courbes paramétrées

Objectifs :

- Détalier des notions autour des courbes paramétrées.

Pourquoi ? :

- Par soucis d'automatisation des algorithmes.
- Effectuer les coupes dans les modèles réduits.
- Ressortir toutes les infos nécessaires des courbes B-Splines.

Définition A.1 (courbes paramétrées de \mathbb{R}^3).

Une courbe paramétrée de \mathbb{R}^3 est une application continue

$$\gamma: \begin{array}{rcl} I & \longrightarrow & \mathbb{R}^3 \\ t & \longmapsto & \gamma(t) = (x(t), y(t), z(t)) \end{array}$$

où I est un intervalle de \mathbb{R} .

Les points $x(t)$, $y(t)$ et $z(t)$ sont appelés les coordonnées de la courbe. De plus, pour $t \in I$, on dit qu'un point $\gamma(t)$ est un point *régulier* si $\dot{\gamma}(t) \neq 0$. À l'inverse, on parle de point singulier. Si, pour tout $t \in I$, $\gamma(t)$ est une point régulier, alors on dit que γ est une courbe paramétrée régulière (CPR).

Définition A.2 (vitesse, accélération, vitesse).

Soit γ une courbe paramétrée de \mathbb{R}^3 . On définit :

- *Sa vitesse : $\dot{\gamma} : I \longrightarrow \mathbb{R}^3$*
- *Son accélération : $\ddot{\gamma} : I \longrightarrow \mathbb{R}^3$*
- *Sa vitesse : $\|\dot{\gamma}\| : I \longrightarrow \mathbb{R}^3$*

Si γ est une CPR alors sa vitesse ne disparaît jamais (elle a toujours une vitesse non nulle).

Définition A.3 (vecteur tangent).

Si γ est une courbe paramétrée alors $\dot{\gamma}(t)$ est le vecteur tangent de γ au point $\gamma(t)$.

La plupart du temps, on s'intéresse au calcul de la longueur d'un arc paramétré. Pour cela, on peut réfléchir de la façon suivante. Soit δt une quantité très petite, l'image de γ entre $\gamma(t)$ et $\gamma(t + \delta t)$ est quasiment une ligne droite dont la longueur est :

$$\|\gamma(t + \delta t) - \gamma(t)\|$$

De même si δt est très petite alors

$$\frac{\gamma(t + \delta t) - \gamma(t)}{\delta t}$$

est équivalent à $\dot{\gamma}(t)$ et donc la longueur serait approximativement

$$s(t) = \|\dot{\gamma}(t)\| \delta t$$

Motivé par cette idée, on peut découper l'arc en un grand nombre de segments, chacun correspondant à une incrémentation de δt . On calcule la longueur de chaque segment, on somme le tout puis on fait tendre δt vers 0.

Définition A.4. (*Abscisse curviligne*)

La longueur d'arc (ou abscisse curviligne) de γ démarrant à t_0 est la fonction :

$$s : \begin{cases} [t_0, +\infty[& \longrightarrow \mathbb{R} \\ t & \longmapsto s(t) = \int_{t_0}^t \|\dot{\gamma}(u)\| du \end{cases}$$

Lorsque l'on s'intéresse à la notion de géométrie différentielle et que l'on rentre plus profondément dans la théorie, on retrouve la notion de *reparamétrisation*. Cela consiste en trouver une courbe $\gamma \circ h : J \longrightarrow \mathbb{R}^3$ où $h : J \longrightarrow \mathbb{R}^3$ est régulière, surjective et de dérivée strictement positive.

Cela nous assure des propriétés très attractives lors d'études plus approfondies. En particulier, un critère très intéressant pour la suite est la notion de *unit speed curve (USP)*, c'est à dire une courbe qui, pour tout $t \in I$, $\|\dot{\gamma}(t)\| = 1$. Cela entraîne les propriétés suivantes :

- Si γ est une CPR alors sa reparamétrisation l'est aussi
- Son abscisse curviligne reste inchangée lors de sa reparamétrisation
- Chaque CPR au une reparamétrisation en USP en prenant $h = s^{-1}$ qu'on appelle reparamétrisation par longueur d'arc.

Nous n'avons pas besoin de cette notion dans nos recherches. Cependant, nous verrons plus loin pourquoi cette caractéristique est importante.

Ensuite on définit deux notions :

- **Courbure** : mesurer à quel point la courbe n'est pas contenue dans une ligne droite.
- **Torsion** : mesurer à quel point la courbe n'est pas contenue dans un plan.

Définition A.5 (Courbure).

Soit $\gamma(t)$ une courbe paramétrée régulière de \mathbb{R}^3 . Alors sa courbure est définie par la fonction :

$$\kappa : \begin{cases} I & \longrightarrow \mathbb{R} \\ t & \longmapsto \kappa(t) = \frac{\|\ddot{\gamma}(t) \times \dot{\gamma}(t)\|}{\|\dot{\gamma}(t)\|^3} \end{cases}$$

Définition A.6 (Torsion).

Soit γ une courbe paramétrée régulière de \mathbb{R}^3 de courbure qui ne disparaît pas alors on définit la torsion comme :

$$\tau : \begin{cases} I & \longrightarrow \mathbb{R} \\ t & \longmapsto \tau(t) = \frac{(\dot{\gamma}(t) \times \ddot{\gamma}(t)) \cdot \ddot{\gamma}(t)}{\|\dot{\gamma}(t) \times \ddot{\gamma}(t)\|^2} \end{cases}$$

où \times représente le produit vectoriel.

Maintenant, imaginons que nous souhaitons faire bouger une caméra le long de la courbe en 3D. Alors on a besoin de dire à la caméra dans quelle direction regarder et quelle sens suivre.

Il faut un repère de référence qui soit un repère mobile orthonormal bougeant avec la courbe. Le plus connu est dénommé repère de FRENET-SERRET ou repère de Frenet (TNB). Il est constitué essentiellement de la donnée de la vitesse et de l'accélération. Il est décrit par le vecteur tangent \vec{T} , normal \vec{N} et binormal \vec{B} , tous unitaires et définit comme suit :

$$\begin{cases} \vec{T} &= \frac{\dot{\gamma}(t)}{\|\dot{\gamma}(t)\|} \\ \vec{B} &= \frac{\dot{\gamma}(t) \times \ddot{\gamma}(t)}{\|\dot{\gamma}(t) \times \ddot{\gamma}(t)\|} \\ \vec{N} &= \vec{B} \times \vec{T} \end{cases}$$

Ce repère obéit aux équations différentielles suivantes :

$$\begin{bmatrix} \dot{\vec{T}} \\ \dot{\vec{N}} \\ \dot{\vec{B}} \end{bmatrix} = \|\dot{\gamma}(t)\| \begin{bmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{bmatrix} \begin{bmatrix} \vec{T} \\ \vec{N} \\ \vec{B} \end{bmatrix}$$

Pour montrer l'utilité de la partie sur la reparamétrisation. Si γ est une USC alors $\|\dot{\gamma}(t)\| = 1$ et donc on a le système :

$$\begin{cases} \dot{\vec{T}} &= \kappa \vec{T} \\ \dot{\vec{N}} &= -\kappa \vec{T} + \tau \vec{B} \\ \dot{\vec{B}} &= -\tau \vec{N} \end{cases}$$

Ce système a une unique solution $(\vec{T}(s), \vec{N}(s), \vec{B}(s))$. Cela signifie que la courbe paramétrée est entièrement définie par les données de κ et τ . On peut imaginer se donner essentiellement la courbure et la torsion d'une courbe correspondant à une série de paramètres et ainsi retrouver la courbe initiale et effectuer.

Nous ne détaillons pas le fonctionnement des courbes paramétrées de type B-Spline. Cependant, pour de plus amples informations, vous pouvez vous référer à [1].

ANNEXE B

Illustrations pour la méthode de recalage 2

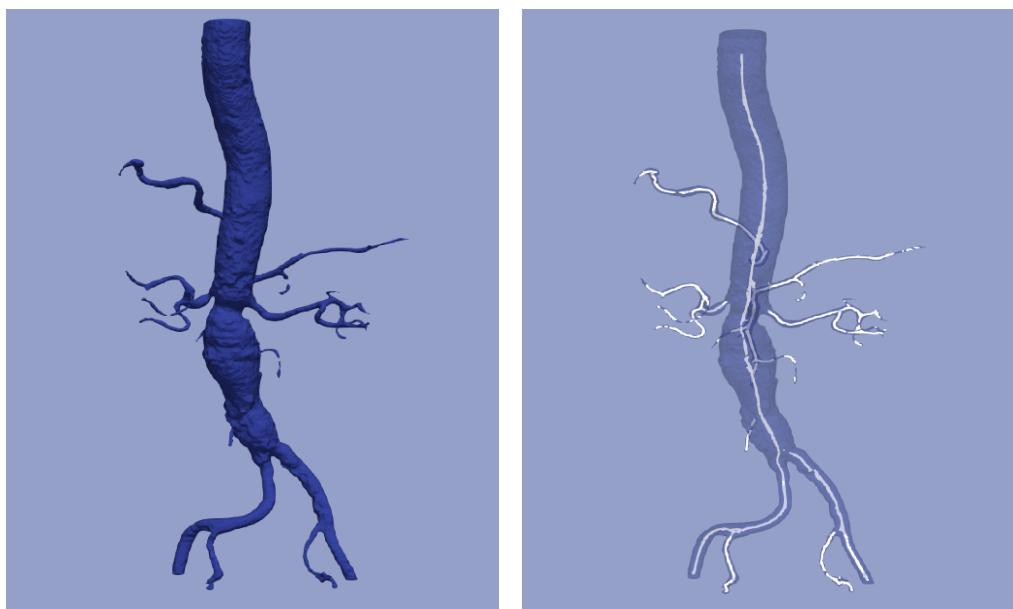


FIGURE B.1 – Géométrie 1 initiale (a) et nuage de points cible (b)

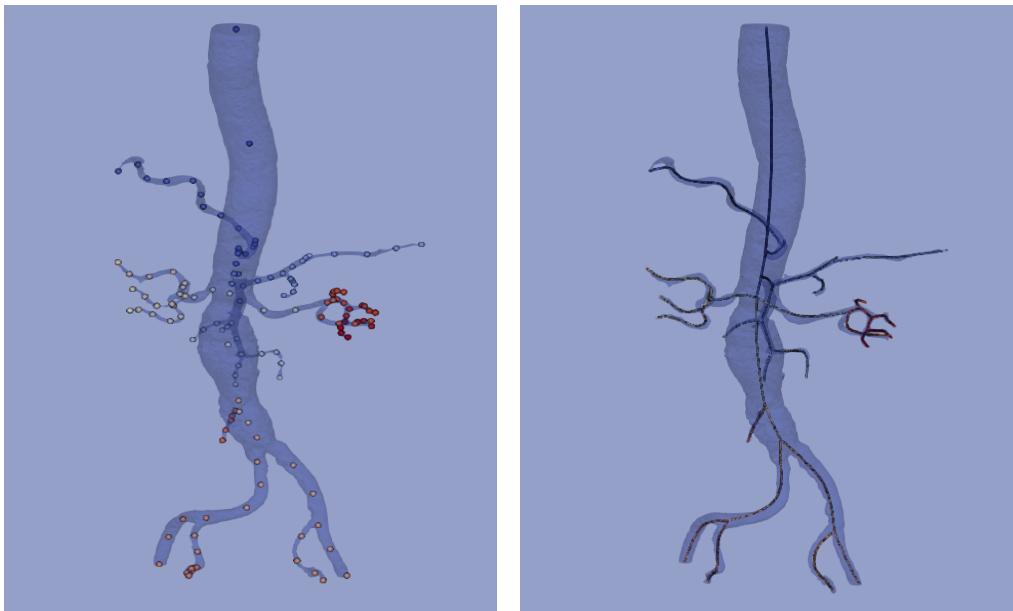


FIGURE B.2 – Points de contrôle (a) et B-Splines (b) initiaux

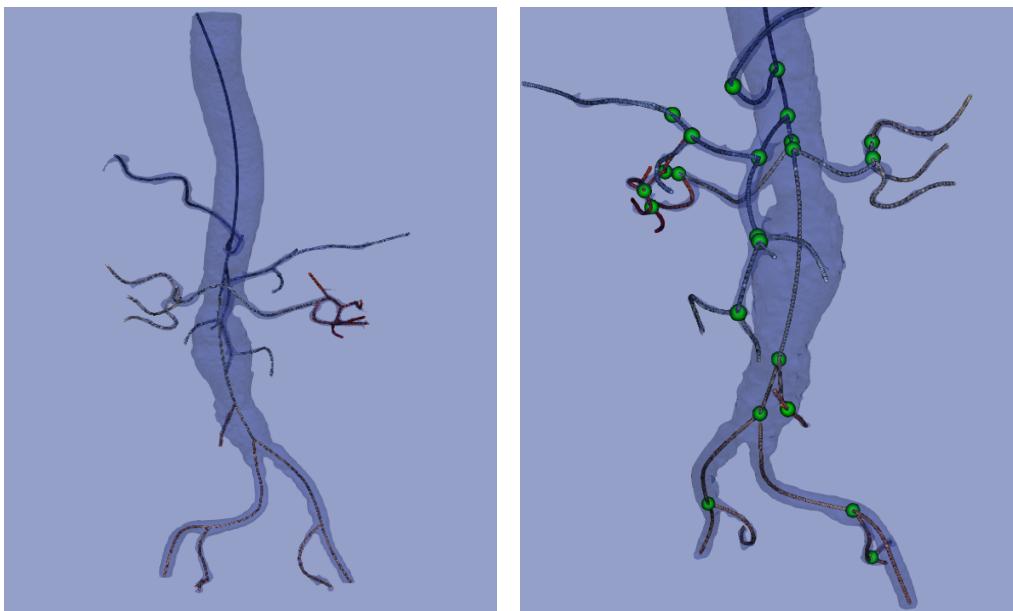


FIGURE B.3 – Recalage du squelette après 5 itérations

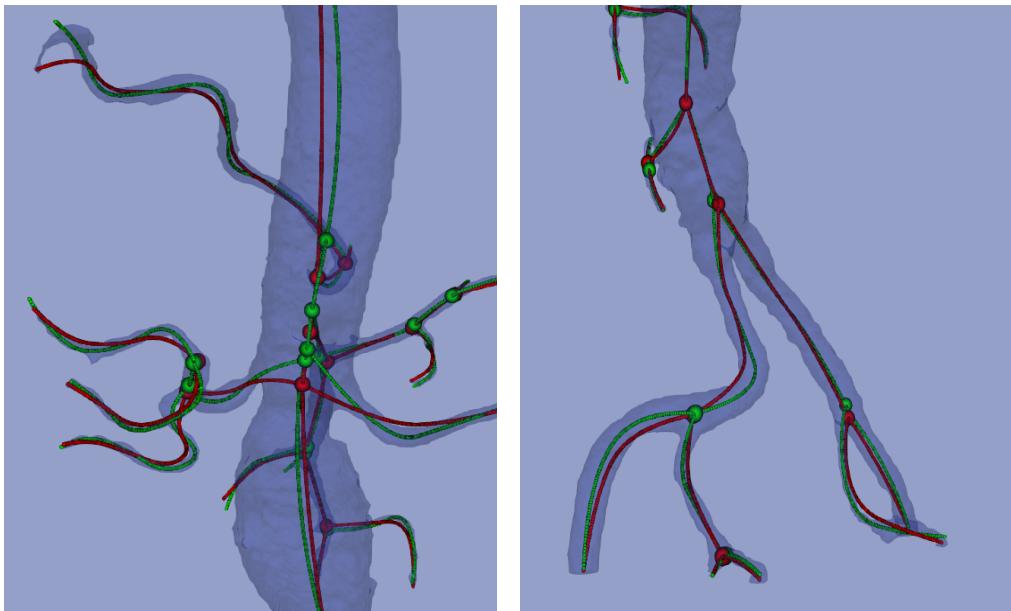


FIGURE B.4 – Comparaison squelette initial (rouge) et recalé (vert)

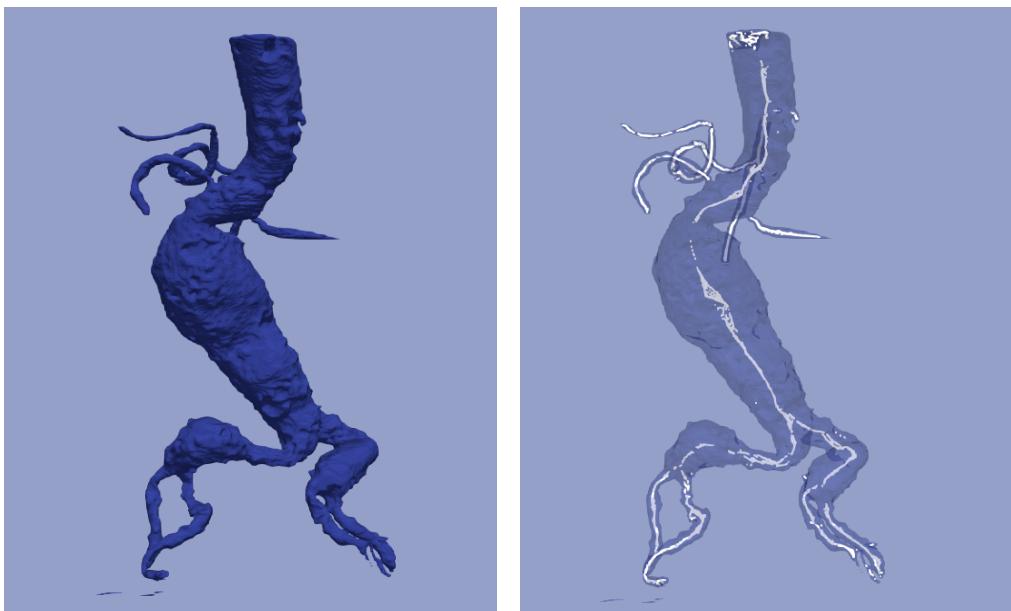


FIGURE B.5 – Géométrie 2 initiale (a) et nuage de points cible (b)

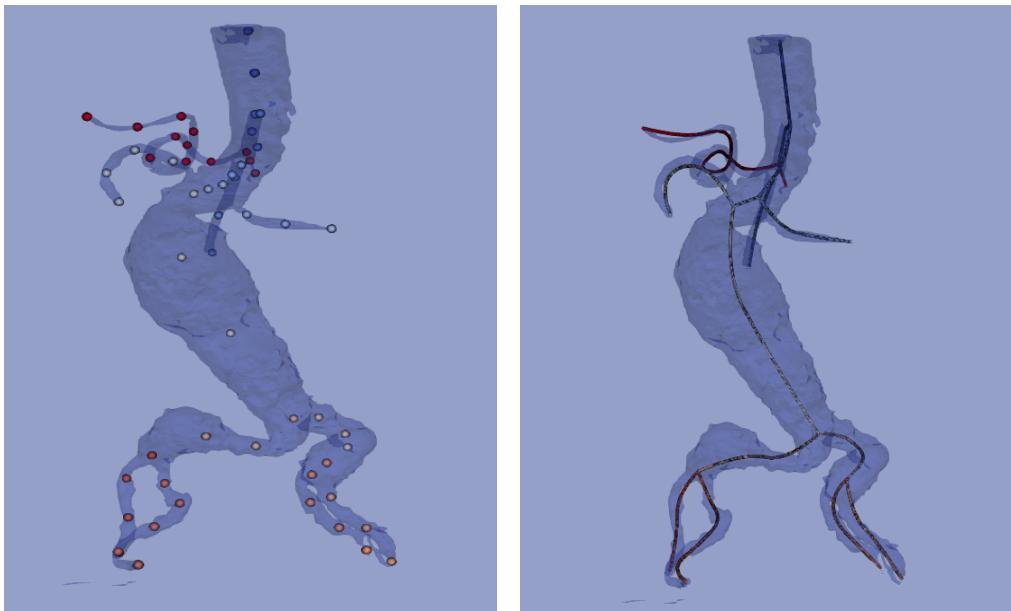


FIGURE B.6 – Points de contrôle (a) et B-Splines (b) initiaux

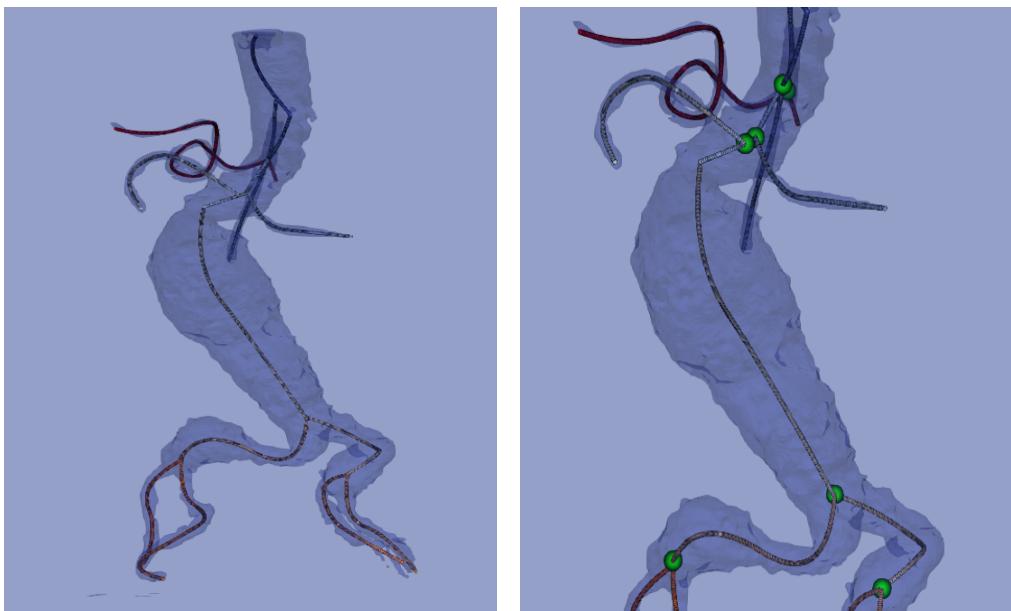


FIGURE B.7 – Recalage du squelette après 5 itérations

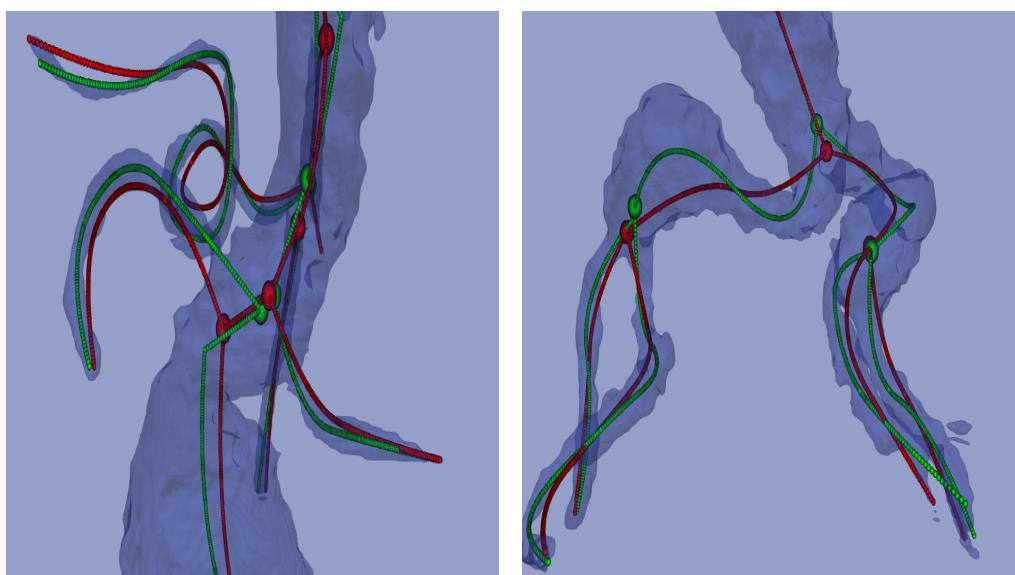


FIGURE B.8 – Comparaison squelette initial (rouge) et recalé (vert)

Bibliographie

- [1] Courbes B-Splines. <http://www.sens-neuchatel.ch/bulletin/no34/art3-34.pdf>.
- [2] Traitement d'images et convolution. <https://perso.esiee.fr/~perretb/I5FM/TAI/convolution/index.html>.
- [3] Andrea Tagliasacchi, Ibraheem Alhashim, Matt Olson, Hao Zhang. Mean curvature skeletons. (English). 2012.
- [4] Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, Alexandru Telea. 3D Skeletons : A State-of-the-Art Report. (English). 2016.
- [5] Liang Liang, Minliang Llu, Caltin Martin, Wel Sun, John A.Elefterlades. A machine learning approach to inverstigate the relationship between shape features and numerically predicted risk of ascending aortic aneurysm. (English). 2008.
- [6] Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or. Skeleton Extraction by Mesh Contraction. (English). 2008.
- [7] She, Fenghua, Chen, Ronghua, Weimin, Hodgson, Peter, Kong. Improved 3D Thinning Algorithms for Skeleton Extraction. (English). 2009.
- [8] Wenping Wang, Helmut Pottmann, Yang Liu. Fitting B-Spline Curves to Point Clouds by Curvature-Based Squared Distance Minimization. (English). 2006.