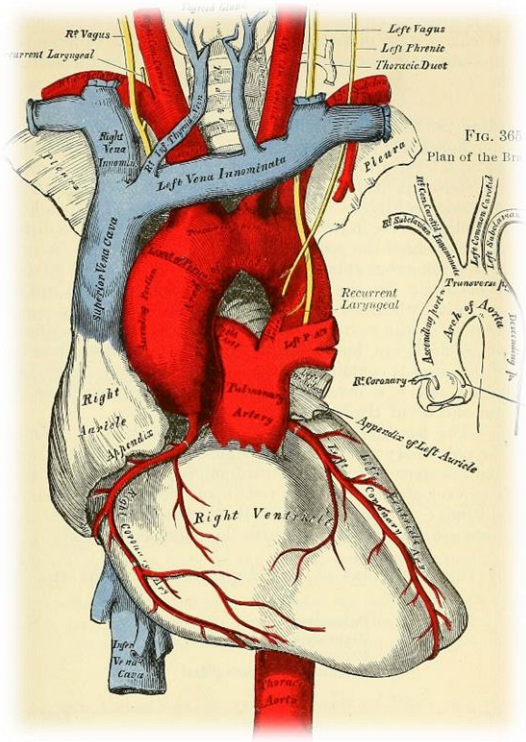
Reconstruction 3D de l'arc aortique à partir de résultats d'IRM

Rapport de projet ~ Biotech et Numérique



MAY 27, 2025

ALEXANDRE LANDREIN & WANDRILLE BUREAU
Encadrant : Mounir LAHLOUH

~Remerciements~

Nous tenons tout d'abord à remercier en un premier lieu notre encadrant, Mounir LAHLOUH, qui nous a guidé et conseillé avec expertise tout le long de notre projet. Sa présence et sa disponibilité nous ont permis de mener à bien ce projet.

Dans un second temps, nous tenons à remercier vivement notre responsable de majeur, Yasmina Chenoune, qui nous a suivi durant ces derniers mois. Sa bienveillance et ses conseils nous ont aidés à progresser lors de ce semestre.

Enfin, nous tenons à remercier l'ensemble des enseignants et intervenants de la majeur Biotech et Numérique qui nous ont fait découvrir et aimer l'ingénierie appliqué à la santé.

Wandrille et Alexandre

Sommaire

1. Contexte et problématique

Introduction

Le projet consiste à effectuer une modélisation 3D de l'arc aortique à partir de résultats d'imageries par résonnance magnétique fournis par notre encadrant. Dans un second temps, il nous est demandé de pouvoir visualiser cette reconstruction à l'aide d'un logiciel et enfin d'imprimer ce modèle 3D. Ainsi nous aurons une visualisation précise de ces structures vasculaires en 3D pour des applications médicales telles que la chirurgie vasculaire ou l'impression 3D pour des simulations d'interventions à but éducatif

Ce projet a pour objectif de nous faire manipuler des fichiers d'imagerie médicale et de découvrir des algorithmes de reconstruction 3D à partir de ces dernières.

Contexte médical

L'arc aortique ou crosse aortique (Figure 1) est la portion courbée de l'aorte qui assure la transition entre l'aorte ascendante et l'aorte descendante. Son rôle est de distribuer le sang oxygéné vers la tête, le cou et les membres supérieurs grâce à trois grosses branches : le tronc brachiocéphalique, l'artère carotide commune gauche et l'artère subclavière gauche comme sur la figure 1.

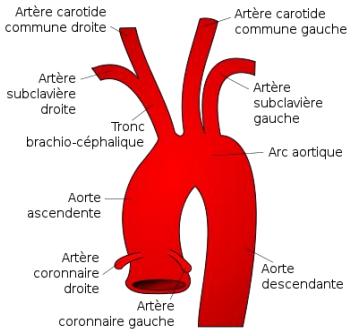


Figure 1 - Schéma de l'arc aortique

Il garantit ainsi un apport efficace en oxygène au cerveau et aux bras, essentiels au bon fonctionnement du corps. Toute anomalie de l'arc aortique peut entraîner des troubles graves de la circulation sanguine. Ces anomalies sont facilement identifiables par la forme de l'arc ou la disposition des artères. Il est donc utile et important de pouvoir visualiser correctement cet arc pour de la prévention médicale.

Problématique

Avant d'entamer la réalisation de ce projet, nous nous sommes posé plusieurs questions et problématiques que nous tâcherons d'éclaircir tout au long du semestre. Parmi celles-ci nous pouvons trouver, comment obtenir un modèle précis de l'arc aortique ? Quels formats de fichiers médicaux seront les plus adaptés à la réalisation de notre projet ? Quels sont les algorithmes permettant une reconstruction 3D et lesquels nous permettrons d'obtenir une reconstruction de qualité ? Comment évaluer cette qualité de reconstruction ? Enfin, comment garantir la reproductibilité pour un usage clinique et donc exigeant ?

2. Formats de données médicales : DICOM vs NIfTI
Choix du format

Notre projet de reconstruction d'un modèle d'arc aortique en 3D se basant sur des fichiers d'imagerie médicale, nous avons dû nous poser la question de quel format choisir pour ces fichiers ? Après quelques recherches, nous en avons retenus deux : DICOM (Figure 2) et NIfTI (Figure 3).

Les fichiers DICOM pour Digital Imaging and Communications in Medicine, sont des fichiers provenant des appareils cliniques d'imagerie médicale. Ils ont l'avantage de contenir une grande quantité de métadonnées notamment sur le patient ou l'acquisition. Cependant la manipulation de ces données est laborieuse et complexe car il y'a une grande quantité de fichiers, plus d'une centaine par examen, et que leur orientation est manuelle.

Les fichiers NIfTI pour Neuroimaging Informatics Technology Initiative, est un format de fichier provenant des DICOM pour donner suite à une conversion et est donc plus simple. En effet, il ne contient qu'un seul fichier et a l'avantage d'avoir la matrice d'affinité intégrée. De par cet unique fichier et des métadonnées plus légères, ne contenant que les techniques, les fichiers NIfTI sont bien plus légers que les fichiers DICOM et ainsi la segmentation dans notre cadre est optimale. De plus il existe des librairies python compatibles facilitant notre travail.

Notre choix s'est donc porté sur l'utilisation des fichiers NIfTI pour tous les avantages donnés précédemment.

Figure 2 - Logo du format DICON

Figure 3 - Logo du format NIfTI

Pour le bon déroulé de notre projet, notre encadrant nous a fournis une banque de fichiers NIfTI. Pour chaque arc aortique, nous avons accès à une « image » et à un « label ». Le premier est l'image brute résultant de l'examen d'imagerie par résonnance magnétique. Le second est la segmentation de l'arc aortique. Pour les visualiser, nous utilisons le logiciel *3D Slicer*. Ce logiciel nous permet d'observer la superposition de l'image et du label selon trois angles de coupe (figure 4) : la coupe sagittale, la coupe frontale et la coupe transversale, comme indiqué sur la figure 4 ci-contre.

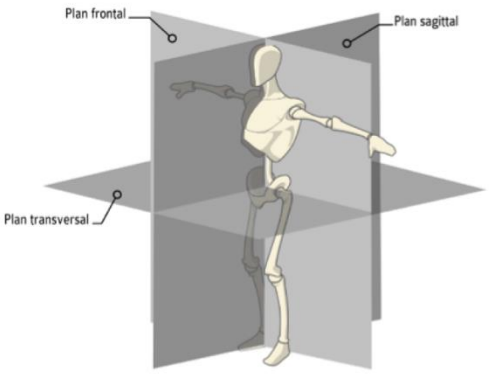


Figure 4 - les différentes coupes

En plus de pouvoir visualiser les différentes coupes, nous pouvons faire défiler les différentes tranches et ainsi se donner une idée de la forme de l'arc aortique. Sur la figure 5, nous pouvons donc observer les trois coupes du fichier « 05 – AORTE » fournit par notre encadrant (Figure 5).

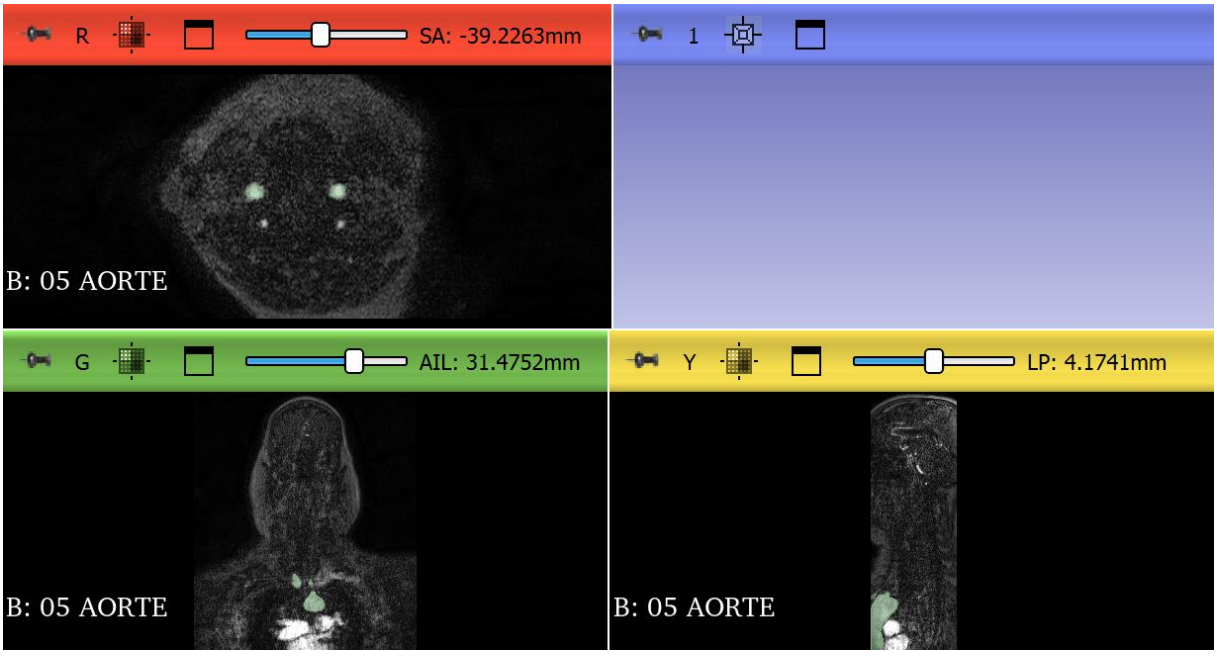


Figure 5 - Visualisation du fichier "05 - AORTE" sur 3D Slicer

3. Etat de l'art des méthodes de reconstruction 3D

Une fois le type de fichier choisi, nous pouvons nous intéresser à l'état de l'art des méthodes de reconstruction 3D. Pour notre projet, nous avons dû nous concentrer sur les méthodes permettant de reconstruire une surface 3D à partir d'un volume médical segmenté.

Nous avons commencé nos recherches en voulant une méthode de reconstruction à la fois rapide et précise. Nous avons donc étudié différentes méthodes classiques de reconstruction 3D, et rapidement identifié quatre approches : Marching Cubes ou Lofting pour la génération initiale de la surface et Poisson Surface Reconstruction ou Neural Radiance Field pour son affinage.

Marching Cubes

La méthode Marching Cubes (Figure 6) repose sur le découpage de l'espace en petits cubes, dont chaque sommet est évalué par rapport à un seuil. En fonction de l'état des sommets, des triangles sont générés pour former une délimitation entre le vide et la surface. L'ensemble de ces triangles constitue alors un maillage 3D.

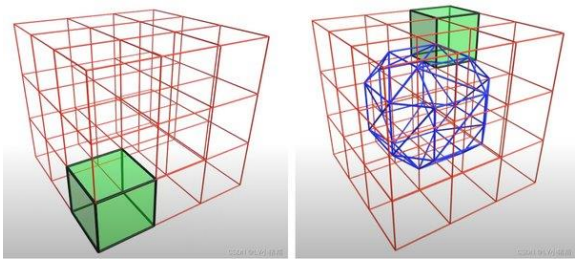


Figure 6 - Marching Cubes

La méthode Lofting (Figure 7) permet de sélectionner deux surfaces 2D et de les rassembler afin de créer une surface 3D les combinant. Cependant, cette méthode n'est pas adaptée aux courbures irrégulières que peut avoir l'arc aortique et a de très grande probabilité de supprimer certains détails sur la modélisation. Nous avons donc décidé de nous tourner vers la méthode Marching Cubes qui est bien plus adaptée pour modéliser une surface irrégulière. Notre sélection de Marching Cubes provient aussi du fait que cette méthode permet de meilleurs résultats au traitement des fichiers NIfTI de sa simplicité d'implémentation.

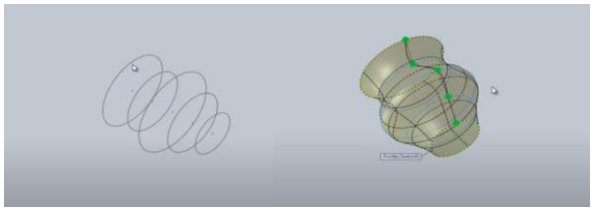


Figure 7 - Lofting

Cependant cette méthode génère des modèles potentiellement bruités et comportant des irrégularités de surface et des imperfections comme des artéfacts cubiques. Dans le cadre de notre projet, nous souhaiterions modéliser un modèle précis et exploitable au

niveau médical. Afin d'améliorer la qualité des surfaces obtenues par Marching Cubes, nous avons étudié la méthode de Poisson Surface Reconstruction et Neural Radiance Field.

Poisson Surface Reconstruction

La méthode de Poisson Surface Reconstruction (Figure 8) se base sur la résolution d'une équation de Poisson et génère un ensemble de nuage de points échantillonnés avec leurs normales.

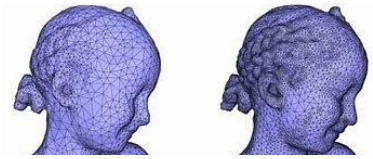


Figure 8 - Poisson Surface Reconstruction

La méthode Neural Radiance Field (Figure 9) utilise une fonction qui fait intervenir l'intelligence artificielle pour générer notre modèle. Seulement, la méthode Neural Radiance Field est assez complexe, et bien que le lissage soit très précis, le temps de construction du lissage augmentera grandement et ajouterait un niveau de complexité élevé, non nécessaire.

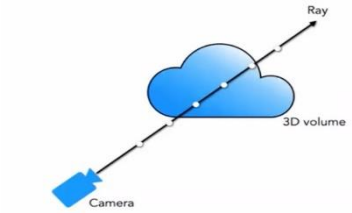


Figure 9 - Neural Radiance Field

Ainsi, nous avons décidé d'utiliser la méthode de Poisson Surface Reconstruction, notre modèle se retrouve donc avec une surface lissée et sans artéfacts. Nous avions d'autres raisons de compléter notre programme avec Poisson :

- Amélioration de la qualité visuelle de notre modélisation grâce au lissage

- Permet un contrôle précis des détails grâce à des paramètres comme le nombre de points et la profondeur.

Coupe des extrémités du modèle

Une fois les algorithmes de reconstruction 3D choisis, pour améliorer notre modèle, nous avions besoin de couper les extrémités de notre modélisation pour la rendre complètement creuse. Nous avons donc cherché des moyens et des algorithmes pour supprimer les extrémités des aortes ascendantes et descendantes.

Nous avons commencé nos recherches en nous intéressant, dans un premier temps, à la fonction bounding_box (Figure 10).

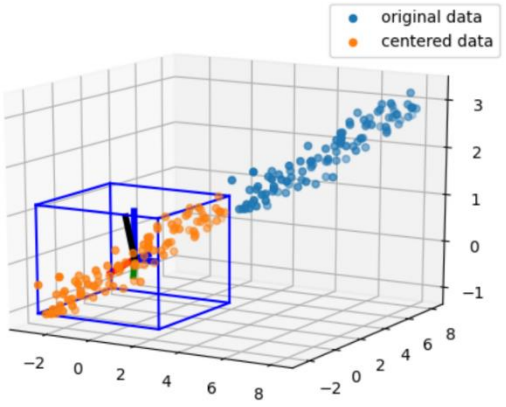


Figure 10 - Bounding_box

Cette fonction permet de définir un volume 3D rectangulaire dans lequel on conserve les points du maillage voulu. Ainsi, cette fonction permet d'extraire toutes les parties de la modélisation en dehors de ce volume 3D de bounding_box. Dans le but de ne couper que les aortes ascendantes et descendantes, nous avons seulement paramétré la fonction bounding_box en y. Cependant, cette méthode présente un inconvénient. Elle agit sur les 3 axes en même temps, ce qui peut conduire à une altération de l'ensemble du maillage. En effet, même les points de construction qui sont à distance de la surface de modélisation sont importants pour une bonne modélisation globale. Les éléments les moins épais et dense de notre modèle s'en retrouvaient fortement impacté. Nous étions alors contraints d'augmenter grandement nos paramètres de constructions ce qui augmentaient le temps de calcul, le volume et le temps de la modélisation.

Nous nous sommes alors tournés vers une méthode plus simple, la méthode de coupe par seuil quantile sur l'axe y. Cette méthode consiste à caractériser la surface de la modélisation par un pourcentage avec la fonction "percentile". L'intégralité du modèle correspond à 100%, on va alors pouvoir choisir le pourcentage appelé quantile que l'on veut garder ainsi que l'axe par laquelle on veut que la coupe s'effectue. Cette méthode agit donc de manière ciblée sur une seule direction, ce qui permet de supprimer précisément l'extrémité voulu du modèle tout en préservant l'intégrité du reste de la structure.

4. Implémentation et explication des scripts

Pour réaliser notre programme et nos modélisations avec Marching Cubes et Poisson Surface Reconstruction, nous avons dû utiliser plusieurs bibliothèques disponibles avec Python.

Présentation des bibliothèques et des fonctions utilisées

Notamment les bibliothèques Nibabel, Numpy, Scikit-Image et Numpy-STL utilisées pour l'implémentation de Marching Cubes. Ces bibliothèques ont toutes leur rôle dans le programme. Nibabel permet d'utiliser des fonctions pouvant Charger le volume NIfTI médical segmenté. Numpy permet la manipulation des matrices de données volumétriques. Scikit-Image est utilisé pour avoir accès aux fonctions de Marching Cubes. Par exemple, la fonction marching_cubes(volume, level, spacing) qui va générer notre surface 3D à partir de notre fichier NIfTI et Mesh(np.zeros(faces.shape[0])) qui va créer un maillage STL prêt à être sauvegardé. Ces fonctions possèdent des paramètres qui nous seront utiles pour rendre notre modèle encore plus précis. Et enfin, Numpy-STL peut exporter un modèle 3D reconstruit en format STL. Ainsi nous pouvons passer à l'application de Poisson Surface Reconstruction avec notre nouveau format STL.

Ainsi, le nouveau format STL peut être traité et affiné grâce à Poisson Surface Reconstruction. Cette méthode utilise aussi une bibliothèque particulière, Open3D, qui permet l'utilisation des fonctions de Poisson Surface Reconstruction comme :

- io.read_triangle_mesh() qui charge le modèle en STL précédemment traité par Marching Cubes.

- create_from_point_cloud_poisson(pcd, depth) qui va recréer une surface lissée sur notre modélisation grâce aux nuages de points et à la profondeur qui sont en paramètre.

- sample_points_poisson_disk(number_of_points) qui répartit uniformément des points sur la surface de notre modèle.

- visualization.draw_geometries() qui permet la visualisation de notre modélisation dans une fenêtre.

De ce fait, nous avons pu générer un modèle lissé avec des détails permettant une potentielle exploitation grâce à sa précision. De plus, le temps de modélisation est relativement rapide et les paramètres choisis permettent une impression peu coûteuse.

Une fois la reconstruction de la surface lissée grâce à la méthode de Poisson Surface Reconstruction, nous souhaitions appliquer notre fonction personnalisée coupe_extremite().

La fonction coupe_extremite() est composé principalement :

- np.percentile() qui calcule une valeur de la surface correspondant à 100%.

- quantile qui représente la proportion en % de la surface que l'on veut garder.

Ainsi, pour résumer le principe de la fonction, les structures du maillage 3D sont transformées en tableaux NumPy pour faciliter les traitements. Le tableau sommets = np.asarray(maillage.vertices) contient les coordonnées (x, y, z) de chaque point du modèle. triangles = np.asarray(maillage.triangles) stocke les indices des sommets formant chaque triangle. On utilise np.all() pour créer un masque qui sélectionne uniquement les triangles dont tous les sommets sont en dessous d'un seuil calculé avec np.percentile(). Par la suite np.unique() extrait les sommets encore utilisés. Enfin, une table est créée pour réindexer les triangles, mis à jour dans une boucle. Pour conclure, le maillage est reconstruit en utilisant les nouveaux sommets et triangles.

En conclusion de notre programme, nous avons utilisé des fonctions de nettoyage comme remove_duplicated_vertices(), remove_degenerate_triangles() et remove_unreferenced_vertices(). Afin d'éliminer les erreurs géométriques et d'assurer l'intégralité du maillage final.

5. Rendus obtenus

Une fois notre modélisation réussie, nous devions nous assurer que nous utilisions bien les meilleurs paramètres.

Les paramètres de notre algorithme

Ces paramètres sont le seuil, le nombre de points et la profondeur. Ces paramètres vont grandement influencer le volume et la précision de notre maillage, nous allons donc les définir :

- Le paramètre de seuil détermine la valeur d'intensité à partir de laquelle le volume du modèle va être extrait. Le volume du modèle, créé par Marching cubes, est constitué de voxel ayant une valeur d'intensité variant généralement entre 0 et 1. Le seuil fixe donc l'intensité exacte à partir de laquelle la surface du maillage sera générée. Si le seuil est trop bas, la surface sera bruitée et si le seuil est trop grand, le modèle risque de perdre en précision.

- Le paramètre du nombre de points détermine combien de points seront utilisés pour la construction du nuage de points de l'algorithme de Poisson Surface Reconstruction sur la surface du maillage initial. Si le nombre de points est trop élevé, le lissage du modèle sera trop important et donc l'algorithme perdrait en précision. Si le nombre de points est trop bas, le lissage peut dégrader le modèle et engendrer des défauts ou des trous.

- Le paramètre de profondeur contrôle le niveau de détail de la modélisation de Poisson Surface Reconstruction. Par exemple, une profondeur de 8 signifie que chaque axe (x, y, z) est divisé en 2^8=256 unités. L'espace 3D est découpé en une grille de 256×256×256 cellules. Plus cette valeur est grande, plus la surface correspond avec précision aux détails du nuage de points. Cependant le temps de calcul et la volume du maillage augmenteront en conséquence. Si cette valeur est trop basse, les détails les plus précis comme les courbures fines seront mal modélisées.

Recherche des meilleurs paramètres

Nous avons donc effectué plusieurs tests pour observer les effets et l'influence de nos paramètres : seuil, nombre de points et profondeur.

Nous pouvons d'abord tester le seuil avec Marching Cubes pour trouver notre valeur de seuil optimal :

Figure 13 - Seuilà 0.3

Figure 12 - Seuil à 0.6

Figure 11 - Seuil à 0.7

On peut observer qu'un seuil d'une valeur faible va provoquer une surface bruitée (Figure 13) et une valeur trop haute va provoquer un risque de perte de certains détails. (Figure 11)

On a donc décidé de choisir un seuil de 0.6 pour la suite de nos tests avec Poisson Surface Reconstruction. (Figure 12)

Tableau de variations des paramètres :

Test	Seuil (Marching Cubes)	Nombre de points (Poisson)	Profondeur (Poisson)	Explication du rendu
1	0.6	10000	6	La surface du modèle est peu précise avec des trous dans le maillage visible, il n'y a pas assez de point pour une reconstruction fiable (fig.15)
2	0.6	10000	8	La surface du modèle manque encore de précision à cause de son manque de densité et de points de construction (fig.14)
3	0.6	30000	6	Le surface du modèle présente des défauts sur les petits reliefs à cause d'une valeur de profondeur trop faible (fig.17)
4	0.6	30000	8	La surface est bien lissée et précise. Il y a un bon équilibre entre les détails et la densité du modèle(fig.16)
5	0.6	50000	6	La surface reste peu détaillée malgré le nombre important de point à cause du manque de profondeur (fig.19)
6	0.6	50000	8	La surface est très précise et satisfaisante mais le modèle devient lourd à manipuler. Finalement, le test 4 est préférable (fig.18)
7	0.6	50000	10	La surface est bien très complexe avec un maillage très dense, ce qui n'est pas idéal pour une impression fluide et rapide (fig.21)
8	0.6	30000	10	La surface est plutôt satisfaisante mais est sur-détaillée ce qui engendrer un lissage trop important et donc une perte dans les détails du modèle (fig.20)
9	0.	10000	10	La surface possède une profondeur bien trop élevée pour son nombre de point ce qui rend le modèle inexploitable (fig.22)

En conclusion, le test 4 semble être équilibré entre précision pour l'exploitation médicale et densité pour une impression fluide.

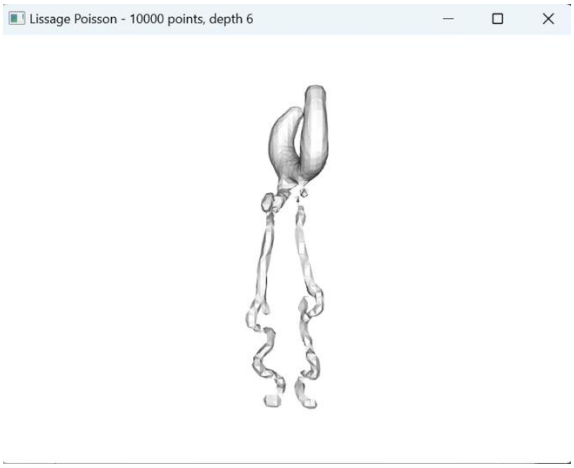


Figure 15 ~ Test 1

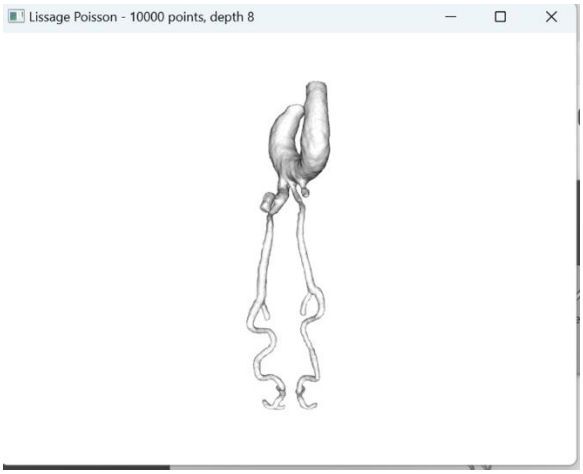


Figure 14 ~ Test 2

Figure 17 ~ Test 3

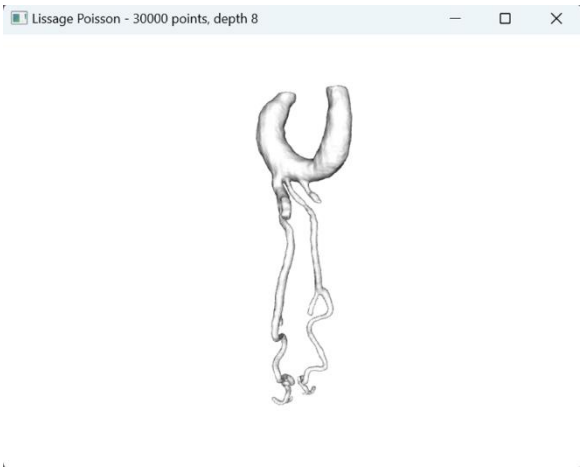


Figure 16 ~Test 4

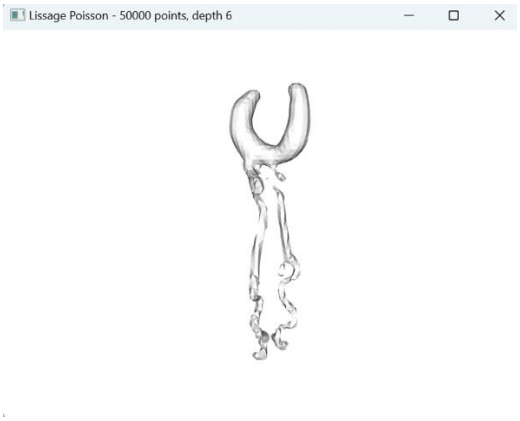


Figure 19 ~ Test 5

Figure 18~ Test 6

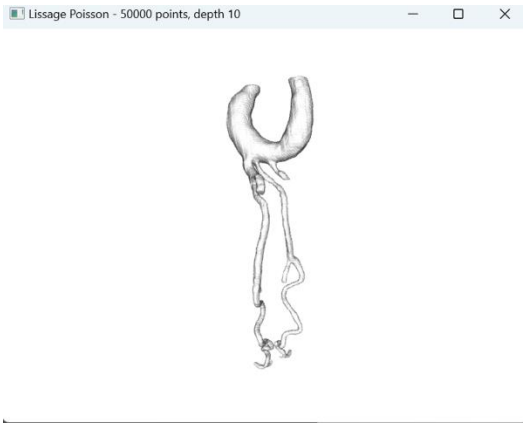


Figure 21~ Test 7

Figure 20~ Test 8

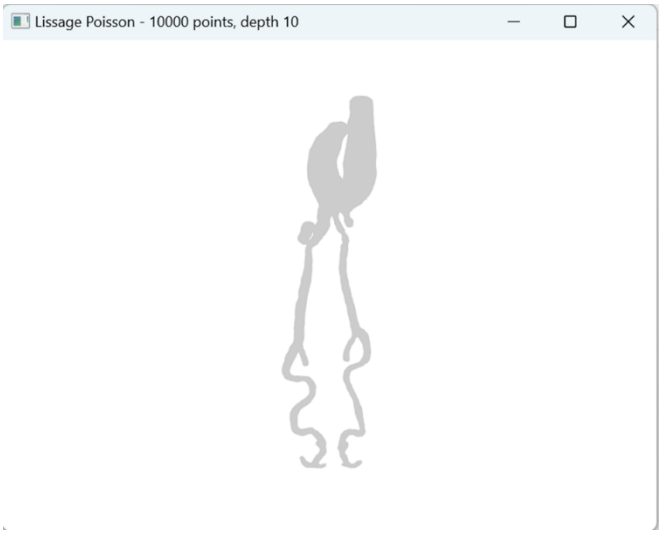


Figure 22~ Test 9

Comparaison des résultats

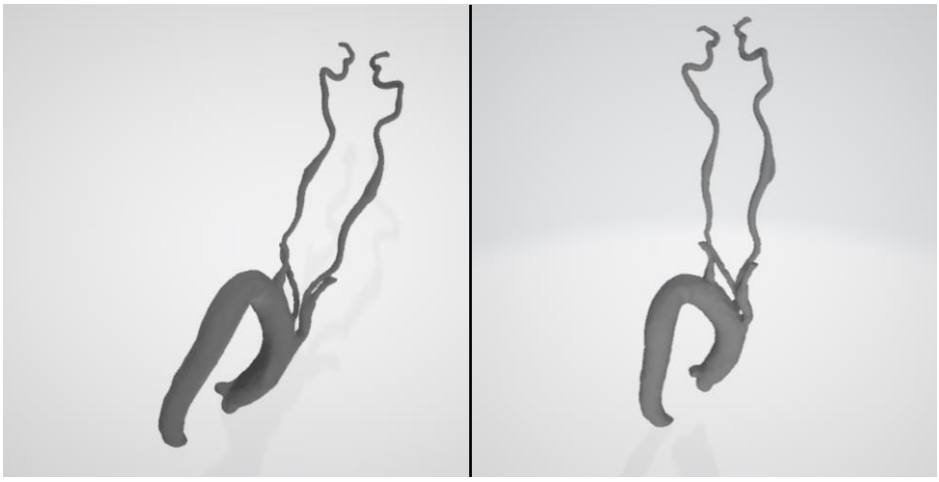


Figure 22 ~ Comparaison de l'aorte 1

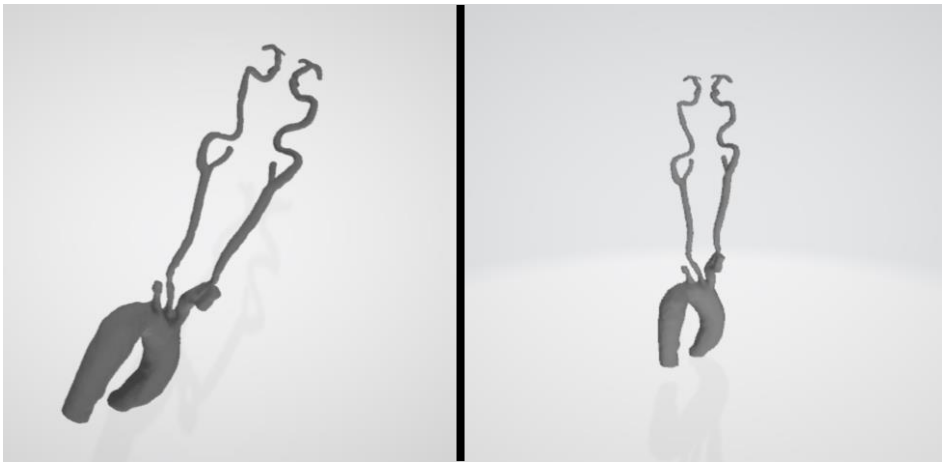


Figure 23 ~ Comparaison de l'aorte 5

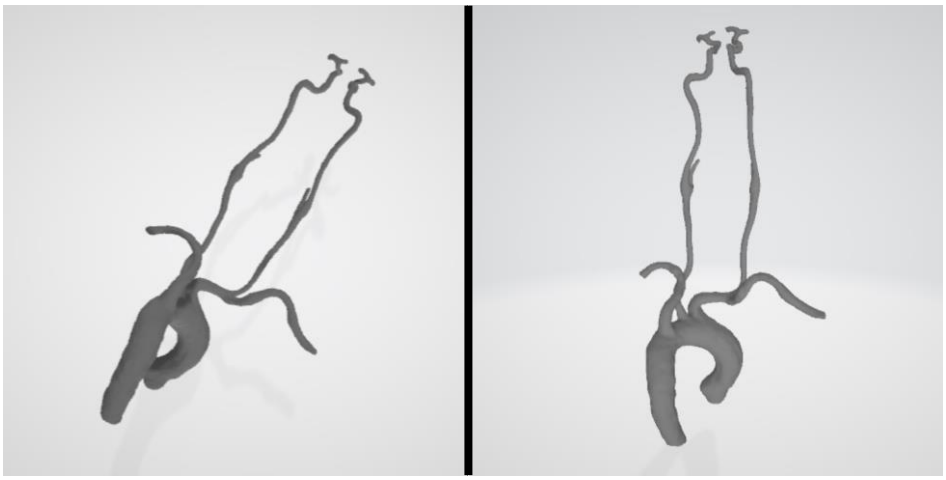


Figure 24 ~ Comparaison de l'aorte 9

Grâce à des modèles 3D des aortes correspondantes aux datas que nous avions, nous avons pu évaluer et constater la qualité de notre reconstruction 3D. En effet, il est aisé

de les reconnaitre. Sur la partie gauche, on peut observer le modèle que nous avons reconstruit et sur la partie droite, le modèle fournit par notre encadrant.

Chaque signe distinctif des différentes aortes sont présents et les dimensions sont égales. Depuis un logiciel de visualisation 3D, nous pouvons également observer une qualité de lissage similaire entre les deux modèles. (Figure 23, 24 et 25)

6. Interface homme-machine

Après avoir mis au point ces algorithmes de reconstruction et après avoir obtenu des résultats très satisfaisants, il fallait développer une IHM qui permettrait de visualiser le modèle 3D obtenu et l'exporter en format .stl. L'idéal serait que cette IHM puisse également communiquer des informations sur la crosse aortique générée. Dans un premier temps, il était question de créer une IHM à l'aide de la bibliothèque python PyQt5 mais il aurait fallu un ordinateur bien plus puissant que nos ordinateurs personnels pour pouvoir afficher un modèle 3D. Nous nous sommes donc tournés vers la mise en place d'un serveur Flask.

Voici sa structure :

```
projet_esme_final/
├── app.py                          ==>Serveur Flask
├── reconstruction_avec_coupe.py    ==>Algorithme de reconstruction avec coupe
├── reconstruction_sans_coupe.py    ==>Algorithme de reconstruction sans coupe
├── requirements.txt                ==>Fichier contenant les bibliothèques à installer
├── README.md                       ==>Fichier expliquant la mise en place de l'IHM
├── static/
│   ├── style.css                   ==>Fichier CSS
│   ├── uploads/
├── templates/
│   └── index2.html                 ==>Fichier HTML
```

Serveur Flask

Flask est une bibliothèque Python de type micro-framework web. Dans le cadre de notre projet, elle nous permet de manière très simple de servir une interface HTML.

En effet, le serveur Flask va héberger une application web locale qui permettra à l'utilisateur d'interagir via une interface web créée en HTML. Le serveur permet également d'exécuter des scripts pythons et ainsi afficher des résultats dynamiques.

Sa simplicité de mise en place et d'exécution nous a donc permis de créer ce début d'IHM, le serveur Flask en étant le noyau.

HTML et CSS

Le fichier HTML définit la structure de l'interface web. Si le serveur a le rôle de cerveau, le fichier HTML a le rôle de squelette. En effet, il va structurer toutes les informations et fonctionnalités que l'on souhaite avoir pour notre IHM. Il comprendra notamment le

formulaire d'import des fichiers NIfTI, les informations concernant le modèle 3D et surtout la visualisation 3D. Par lui, Flask va donc pouvoir mettre en place l'IHM.

Grâce à du JavaScript ajouté au fichier HTML, nous avons pu récupérer les informations dynamiques et grâce à la bibliothèque Three.js, afficher la scène 3D et la manipuler.

Cependant, HTML tout seul ne permet que d'obtenir le fond de l'IHM, donnant un rendu non structuré et très peu ergonomique. C'est là que le fichier CSS rentre en jeu.

Ainsi, le fichier CSS va venir fournir la forme du projet. Permettant ainsi d'avoir une interface visuellement agréable, structurée en colonnes. On peut donc ainsi donner à notre projet un aspect plus professionnel et bien plus présentable.

Fonctionnalités de l'IHM

Figure 25 ~ Affichage de l'IHM

La figure 26 correspond au résultat de la reconstruction du fichier 05_AORTE.nii.gz et de sa visualisation grâce à notre IHM.

Dans la colonne de gauche, cette IHM permet donc d'importer un fichier au format .nii ou .nii.gz. Il est possible de choisir avec quel algorithme nous voulons le modéliser. Dans ce même formulaire, il est possible à l'aide de boutons de lancer la modélisation mais également de télécharger le fichier .stl obtenu grâce à cette reconstruction. En dessous, différents widgets sont mis à disposition de l'utilisateur pour interagir avec la scène 3D. Il peut notamment zoomer ou recentrer la scène mais aussi choisir de ne pas afficher les axes.

Au centre de la page web, c'est là que le modèle 3D va apparaître à la suite de sa construction. Il est possible à l'aide de la souris, de le déplacer, faire pivoter ou zoomer.

Enfin, dans la colonne de droite, nous pouvons visualiser diverses informations sur le modèle généré comme le nom du fichier .nii, ses dimensions ou encore l'algorithme utilisé.

Accessibilité du projet

Cette IHM a pour but de rendre notre projet accessible et de regrouper dans une seule interface la modélisation, la visualisation et l'exportation du modèle 3D. Pour ce faire nous avons tenu à garder des fonctionnalités et widget simples pour que l'utilisateur ne soit pas perdu.

Du point de vue de l'installation. Nous avons ajouté un fichier .txt comprenant les différentes bibliothèques à installer pour n'en rater aucune et ne pas perdre de temps.

L'installation nécessitant la création d'un environnement virtuel, pas forcément intuitif pour les utilisateurs, nous avons rédigé un README détaillant l'installation pas à pas avec toutes les commandes pour la création de l'environnement virtuel, l'installation des bibliothèques et le lancement du serveur Flask. Ce fichier explique également comment réagir face aux différents problèmes que l'utilisateur pourrait rencontrer

7. Perspectives et problématiques de l'impression 3D

Problématique de coupe des extrémités

La problématique de notre algorithme de coupe repose sur le manque de localisation précise de la coupe. En effet, la méthode de coupe par seuil quantile n'extrait le maillage que suivant un ou plusieurs axes de directions x, y ou z. Cette méthode n'offre pas un contrôle local sur les zones que l'on veut supprimées. De ce fait, sur certains fichiers que nous avions, les coupes étaient mal réalisées ou irréalisables pour avoir un résultat satisfaisant. Notamment les modèles avec une aortes ascendantes ou descendantes plus longues que l'autre. Une des deux aortes était coupée mais pas l'autre. Certaines structures importantes, comme les artères coronaires droite et gauche, les artères carotides communes, ainsi que les artères subclavières ne pouvaient être coupées à cause de leur forme ou de leur orientation et position dans l'espace.
Cependant, pour pallier cette limitation, nous pourrions envisager d'utiliser une méthode de clustering de surface. Ainsi, cette méthode permettrait d'analyser les propriétés géométriques locales du maillage pour regrouper ces zones sous formes de clusters. Les coupes pourront alors être appliquées de manière précises et localisées sans altérer le reste du modèle.
Le clustering de surface repose sur des techniques d'apprentissage non supervisé comme k-means. Cette méthode offrirait une modélisation plus intelligente et ciblée et permettrait d'obtenir un arc aortique complet.

Problème rencontré pour l'impression 3D

L'impression 3D de ce modèle obtenu grâce à notre algorithme est également l'un de nos objectifs finaux. Cette dernière aurait avant tout un but éducatif. En effet, avec un modèle 3D en main, les étudiants en médecine pourraient s'entraîner à des actions et opérations cliniques réalistes. Les chirurgiens ou autres praticiens pourraient également s'en servir pour préparer leur opération et ainsi écarter des risques.

Pour ce faire, l'impression doit donc être creuse, ce qui pose un problème. Pendant l'impression, l'imprimante va devoir créer des supports à l'intérieur pour que l'impression se fasse correctement. Mais étant donné la forme de l'arc, il sera compliqué de les enlever. Les solutions possibles sont le découpage du modèle en plusieurs parties pour les imprimer séparément puis en les assemblant à la main. Mais le découpage reste complexe à effectuer. Une autre solution est l'utilisation d'une imprimante par suspension comme sur la figure 27 si-dessous, utilisant du gel et qui permet de se passer des supports. Cependant l'école n'en a pas à disposition.

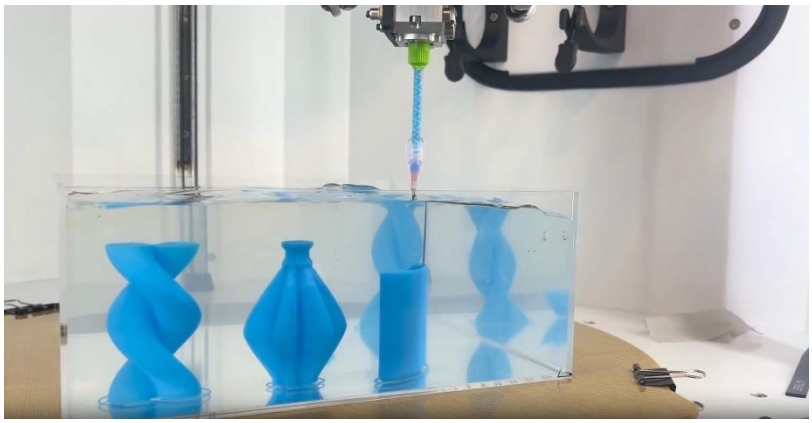


Figure 26 - Imprimante par suspension

8. Analyse de risque

Type	Risques induits	Causes	Gravité (/5)	Détectabilité (/5)	Occurrence (/5)	Cotation	Remédiation
Design	Mauvaise reconstruction et maillage inexploitables	Données provenant de l'hôpital corrompu ou endommagé	4	2	4	32	Vérification automatique de la qualité à l'import
Conception	Mauvaise interprétation anatomique	Choix d'un seuil de segmentation inadapté	4	2	3	24	Algorithmes robustes et feedback utilisateur
Pratique	Perte de données, fuite de données personnelles	Failles de sécurité, absence d'authentification et de protection aux attaques informatiques	5	3	2	30	Authentification forte, pare-feu, mise à jour régulière et demande à des spécialistes
Conception	Reconstruction non fidèle	Logique de la fonction bounding box mal paramétrée	3	2	3	18	Ajustement manuel ou outil d'édition, visualisation préalable
Design	Interruption de l'application	Machine peu puissante ou algorithme trop complexe	3	1	4	12	Optimisation du code ou pré-filtrage
Conception	Impossibilité d'export ou de visualisation	Paramètres Poisson mal adaptés ou mauvais lissage	4	3	3	36	Réduction automatique de complexité, vérification du maillage
Pratique	Utilisateur peut commettre des erreurs	Interface du logiciel peu intuitive	2	2	3	12	Interface simple, messages d'erreur clairs
Pratique	Problème accessibilité pour certains utilisateurs	Dépendances système ou composants obsolètes	3	2	2	12	Tests multi-plateformes, recommandations techniques claires
Design	Maillage inexploitable, mauvaise reconstruction visuelle	Paramètres inadéquats ou bruit dans les données	3	3	3	27	Paramètres ajustables pour l'utilisateur

Echelle :

Gravité :

1 : Aucun impact médical ni technique. La visualisation est possible malgré le défaut.

2 : Légères erreurs visuelles ou pertes de temps, mais le modèle reste exploitable.

3 : Mauvaise qualité du modèle, nécessitant retouche ou retraitement. Risque de report du projet.

4 : Reconstruction inutilisable ou interprétation erronée des structures anatomiques.

5 : Risque pour un patient (mauvaise préparation d'intervention) ou perte irrémédiable de données.

Détectabilité :

1 : Problème détecté automatiquement (ex : vérif de qualité à l'import, erreurs système visibles).

2 : Détection rapide par un utilisateur entraîné ou lors d'une visualisation de routine.

3 : Détection possible après quelques manipulations ou tests manuels.

4 : Nécessite une revue approfondie, souvent détecté trop tard (ex : export ou impression).

5 : Le défaut passe inaperçu jusqu'à l'utilisation finale du modèle.

Occurrence :

1 : Aucun cas rencontré pendant les tests ou projets similaires.

2 : Déjà observé mais peu fréquent avec de bonnes pratiques (ex : bon format, code stable).

3 : Problème apparaissant régulièrement selon la qualité des données ou outils utilisés.

4 : Problème courant, notamment si automatisation insuffisante ou dépendance forte aux entrées.

5 : Presque systématique sans actions correctives ou vérifications spécifiques.

9. Conclusion

Notre projet a donc pour objectif de réaliser une reconstruction 3D du réseau vasculaire de l'arc aortique à partir d'imagerie à résonnance magnétique. Ce projet a plusieurs objectifs, les principaux étant la manipulation de données médicales, la découverte et l'utilisation des méthodes de reconstruction 3D, le développement d'un interface homme-machine et l'impression de notre modèle 3D.

Les potentielles utilisations de notre projet en milieu médical sont nombreuses. En effet, il permettra au praticien d'avoir accès à une visualisation anatomique précise, d'avoir une aide opératoire et une navigation en temps réel. Il permettra également d'effectuer une simulation pré-opératoire, limitant ainsi de potentiels risques durant l'opération. Notre projet pourrait également avoir une utilisation à but éducative. Les étudiants en médecine pourront s'exercer sur les modèles imprimés en 3D dans des conditions proches de la réalité.

Concernant la modélisation, pour notre projet, nous avons utilisé les méthodes de reconstructions 3D Marching Cubes et Poisson Surface Reconstruction. Après plusieurs tests, nous avons conclus que les paramètres optimaux pour notre modèle sont :

- Seuil = 0.6

- Nombre de points = 30000

- Profondeur = 8

On devrait avoir une modélisation à la fois précise et avec un minimum de contrainte d'impression.

Pour améliorer le réalisme de notre modèle, nous avons utilisé la méthode de coupe par seuil quantile, afin de supprimer les extrémités de notre modèle.

Pour la partie IHM, nous avons mis en place un serveur Flask, nous permettant à l'aide des algorithmes de reconstruction, d'un code en HTML et un en CSS, de reconstruire, visualiser et exporter le modèle 3D de façon très simple. De plus, notre IHM nous permet de visualiser différentes informations sur la crosse aortique générée et de la manipuler dans l'espace 3D.

Ainsi, nous estimons avoir mené à bien notre projet en remplissant les objectifs donnés par notre encadrant et sommes satisfaits du travail réalisé. Seul point noir, nous n'avons pas pu imprimer le modèle généré, mais nous le ferons sans aucun doute de notre côté maintenant que nous avons toutes les clés en main.

10. Annexe

Etat de l'art :

Méthode de reconstruction :

Marching cubes : Marching cubes - Wikipedia / Marching cubes: A high resolution 3D surface construction algorithm | ACM SIGGRAPH Computer Graphics / GitHub - nihaljn/marching-cubes: C++ implementation of the Marching Cubes algorithm

Lofting : [2202.06330] Optimal lofted B-spline surface interpolation based on serial closed contours

Poisson Surface Reconstruction : GitHub - HugoRdet/Python_Poisson_Surface_Reconstruction: Implementation of the Poisson Surface Reconstruction paper using only PyTorch. / Poisson surface reconstruction | Proceedings of the fourth Eurographics symposium on Geometry processing

Neural Radiance Field : [2003.08934] NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis / GitHub - bmild/nerf: Code release for NeRF (Neural Radiance Fields)

Bounding_box : Minimum bounding box - Wikipedia

Bibliothèques utilisées :

Python & Écosystème :

Flask : https://flask.palletsprojects.com/

nibabel : https://nipy.org/nibabel/

numpy : https://numpy.org/doc/

scikit-image : https://scikit-image.org/docs/stable/

numpy-stl : https://pypi.org/project/numpy-stl/

Open3D : http://www.open3d.org/docs/

setuptools : https://setuptools.pypa.io/en/latest/

Technologies Web :

HTML : https://developer.mozilla.org/fr/docs/Web/HTML

CSS : https://developer.mozilla.org/fr/docs/Web/CSS

JavaScript : https://developer.mozilla.org/fr/docs/Web/JavaScript

Three.js : https://threejs.org/docs/)

https://threejs.org/docs/#examples/en/loaders/STLLoader)

Autres outils utilisés:

Visual Studio Code : https://code.visualstudio.com/

Python3.10 : https://www.python.org/downloads/release/python-3100/

Pip : https://pip.pypa.io/en/stable/

Ressources complémentaires :

NIfTI format info](https://nifti.nimh.nih.gov/

Three.js STL examples](https://threejs.org/examples/#webgl_loader_stl/

Open3D Tutorials](http://www.open3d.org/docs/latest/tutorial/