POLYPUBLIE
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Mise en correspondance spatio-temporelle de séquences angiographiques par apprentissage profond
Auteur: Author:	Raphaël Royer-Rivard
Date:	2021
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Royer-Rivard, R. (2021). Mise en correspondance spatio-temporelle de séquences angiographiques par apprentissage profond [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/5611/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/5611/
Directeurs de recherche: Advisors:	Farida Cheriet, & Nagib S. Dahdah
Programme: Program:	Génie informatique

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Mise en correspondance spatio-temporelle de séquences angiographiques par apprentissage profond

RAPHAËL ROYER-RIVARD
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Février 2021

© Raphaël Royer-Rivard, 2021.

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Mise en correspondance spatio-temporelle de séquences angiographiques par apprentissage profond

présenté par **Raphaël ROYER-RIVARD**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Thomas HURTUT, président
Farida CHERIET, membre et directrice de recherche
Nagib S. DAHDAH, membre et codirecteur de recherche
Guillaume-Alexandre BILODEAU, membre

DÉDICACE

À tous mes amis du labos,
vous me manquerez...

REMERCIEMENTS

Je désire tout d'abord remercier ma directrice de recherche Farida Cheriet qui m'a généreusement offert conseils, encouragements, support financier, contacts professionnels et aide pour compléter plusieurs documents administratifs (à ne pas sous-estimer !). Sa véritable bienveillance et son profond désir de faire tout en son possible pour permettre à ses étudiants de réussir leurs études supérieures ont sans nul doute mérité mon respect.

Je tiens à remercier les autres membres de mon comité conseil, mon codirecteur et cardiologue Dr. Nagib Dahdah qui a cru en moi et qui a vaillamment répondu à toutes mes questions en rapport aux techniques de diagnostic des maladies coronariennes ainsi que Fantin Girard qui m'a supporté en me proposant des méthodes pour améliorer l'entraînement de mon modèle de réseau de neurones.

J'aimerais aussi remercier l'associé de recherche du laboratoire LiV4D Philippe Debanné, qui a toujours été disponible pour m'indiquer les démarches nécessaires pour régler mes pépins techniques, pour répondre à mes questions concernant les rédactions scientifiques, pour avoir généreusement partagé son temps pour corriger mon article de conférence et pour toutes les conversations variées que nous avons eu durant les pauses de dîner.

Je dois aussi souligner l'aide que m'a apportée le cardiologue Dr. Pierre Tessier de l'Hôpital du Sacré-Coeur de Montréal en me laissant assister à des examens d'angiographie coronaire. Je suis reconnaissant de son enthousiasme envers mon projet qui m'a encouragé ainsi que de sa précieuse aide pour me fournir des données anonymisées d'angiographies de patients adultes.

Je ne pourrais oublier de mentionner l'énorme support moral et technique que m'ont donné les deux étudiants au doctorat Gabriel Lepetit-Aimon et Clément Playout lors de nos innombrables journées de travail au laboratoire ainsi que par messagerie lors de la pandémie de Covid-19.

Je remercie aussi TransMedTech et CRSNG pour le financement de ma maîtrise sans lequel je n'aurais pu étudier durant ces deux années.

Finalement, j'aimerais remercier ma charmante conjointe Marie-Ève Dumas, qui m'a convaincu de reprendre mes études, qui m'a toujours encouragé et qui n'a jamais cessé de démontrer son intérêt envers mon travail qu'elle ne pouvait pas toujours comprendre.

RÉSUMÉ

Les maladies coronariennes sont la deuxième plus importante cause de décès au Canada. Pour diagnostiquer celles-ci, les cardiologues se fient principalement sur l'électrocardiographie et l'angiographie coronaire invasive (ICA). Cette procédure consiste en l'injection d'un agent de contraste dans les artères coronaires à partir d'un cathéter et visualisée grâce à une machine à rayons X. En analysant ces images, il est possible de mesurer la taille de la lumière des artères coronaires. Mais puisque la plaque causée par une accumulation de dépôts de graisses ne se forme pas de manière uniforme, il est nécessaire de capturer plusieurs séquences à partir d'angles différents pour évaluer la sévérité de la maladie.

Lors de l'opération de projection de 3D à 2D faite par l'imagerie à rayons X, une quantité considérable d'information est perdue et une sélection sous-optimale des angles de projection peut mener à une sous-estimation ou surestimation de la sévérité de la maladie. Pour cette raison, la tâche de reconstruction 3D de l'arbre coronarien à partir d'images angiographiques 2D a été explorée par plusieurs chercheurs, principalement dans les deux dernières décennies.

Ce mémoire décrit le travail effectué pour deux étapes nécessaires dans le processus de reconstruction 3D des artères coronaires, soient la synchronisation des séquences angiographiques ainsi que la mise en correspondance des segments d'artères d'une vue à l'autre.

La première étape est implémentée grâce à une approche par apprentissage profond basée sur des séquences angiographiques multi-vues brutes pour décrire la phase cardiaque à partir de groupes de 3 images consécutives. Les vecteurs de caractéristiques générés par le réseau de neurones à partir des images de deux vues sont par la suite comparés entre eux afin de créer une matrice de distance. Un algorithme de recherche de chemin est ensuite appliqué sur la matrice afin de trouver la meilleure synchronisation entre les deux vues, ce qui est essentiel pour une bonne reconstruction 4D (3D + t) de l'arbre coronarien. Le modèle d'apprentissage profond est entraîné et testé sur un total de 505 séquences angiographiques provenant de 39 patients pédiatriques et de 26 patients adultes dont certains sont atteints de maladie coronarienne. Le modèle est évalué grâce à un score de synchronisation basé sur les données d'ECG qui sont acquises pendant l'angiographie. Il obtient un score d'alignement de 96.04% lorsque entraîné et testé seulement sur des données de patients pédiatriques et obtient un score de 94.24% lorsque entraîné et testé sur un jeu de données composé avec les séquences de patients pédiatriques et adultes.

La deuxième étape, qui vise à trouver des correspondances des segments d'artères dans différentes vues, est implémentée grâce à une technique par apprentissage profond basée sur

des graphes orientés. Ceux-ci sont extraits de squelettes générés à partir de segmentations d'images angiographiques. Pour extraire ces graphes orientés, il est nécessaire d'avoir un point de départ dans le squelette et il est logique que celui-ci soit la base de l'artère, c'est-à-dire l'ostium. Pour détecter l'ostium, un algorithme permettant d'identifier les segments du cathéter est utilisé. En suivant ces segments de cathéter, il est possible de localiser le début de l'artère en détectant un changement brusque de largeur ou d'angle. Les résultats de détection de l'ostium sont évalués en comparant la position de ceux-ci avec des données *ground truth* où l'ostium a été manuellement identifié dans 71 images et la moyenne de distance est de 30.29 pixels pour des images de taille 1024x1024. Pour extraire les graphes orientés, il est aussi nécessaire de détecter les croisements de vaisseaux, car les algorithmes de squelettisation transforment les croisements en deux bifurcations. Un algorithme de détection de croisement a donc été implémenté en se fiant sur la largeur, l'angle et la distance des vaisseaux pour trouver la meilleure continuité. L'algorithme produit parfois des faux-positifs de détection de croisement, mais réussit à extraire la majorité des branches dans la bonne direction. En moyenne, 83.2% des branches finales (finissant en cul-de-sac) sont présentes dans les graphes orientés. Une fois ceux-ci extraits, un réseau de neurones siamois par graphe est entraîné pour apprendre à générer des vecteurs de caractéristiques qui représentent suffisamment bien les bifurcations afin de trouver des paires dans deux vues. Bien qu'imparfaite, cette preuve de concept montre que la méthode est viable pour résoudre la tâche de mise en correspondance spatiale des segments d'artères.

ABSTRACT

Coronary Artery Disease (CAD) is the second leading cause of death in Canada. Electrocardiography and Invasive Coronary Angiography (ICA) are generally used by cardiologists to diagnose CAD. This procedure consists of the injection of a contrast agent in the coronary arteries from a catheter and visualized with the help of an X-ray device. By analyzing the generated images, it is possible to roughly measure the lumen size of the coronary arteries. But since the plaque caused by an accumulation of fat deposits do not form in a uniform way, multiple sequences must be acquired from different angles to effectively evaluate its severity.

Because of the 3D to 2D projection operation done with X-ray imaging, a considerable amount of information is lost and suboptimal projection angle selection can lead to an underestimation or overestimation of the CAD severity. For this reason, the task of 3D reconstruction of the coronary tree from 2D angiography has been explored by several researchers, mostly in the last two decades.

This master's thesis describes the work done for two necessary steps in the 3D reconstruction process of the coronary arteries, which are the synchronization of angiographic sequences and the spatial matching of artery segments from one view to the other.

The first step is implemented with a deep learning approach based on raw angiographic multi-view sequences to describe the cardiac phase from groups of 3 consecutive images. The feature vectors generated by the neural network from the images of two views are then compared to generate a distance matrix. Next, a pathfinding algorithm is applied on the matrix to find the best synchronization between the two views, which is essential for a good 4D (3D + t) reconstruction of the coronary tree. The deep learning model is trained and tested on a total of 505 angiographic sequences acquired on 39 pediatric patients and 26 adult patients, some of whom are affected by coronary artery disease. The model is evaluated with a synchronization score based on ECG signals that are acquired during the angiography procedures. It obtains an alignment score of 96.04% when trained and tested on pediatric data only and a score of 94.24% when trained and tested on a dataset composed of sequences from pediatric and adult patients.

The second step, which aims to find matches between artery segments in different views, is implemented with a deep learning technique based on oriented graphs. They are extracted from skeletons generated from segmented angiographic images. To extract these oriented graphs, it is necessary to have a starting point in the skeleton and it makes sense to choose the base of the artery, which is the ostium. To detect the ostium, an algorithm that detects the

catheter segments is developed. By following these catheter segments, it is possible to localise the base of the artery by detecting a sudden change in the width or angle of the segment. The results of the ostium detection are evaluated by comparing the detected ostium positions with the ground truth data where the ostium has been manually identified in 71 images and the average distance is 30.29 pixels for 1024x1024 images. To extract the oriented graphs, it is also necessary to detect vessel crossings since the skeletonization algorithms transform each crossings into two bifurcations. A crossing detection algorithm has then been implemented by relying on the width, angle and distance of the vessels to find the best continuity. The algorithm sometimes produces crossing detection false positives, but extracts most of the branches in the right direction. On average, 83.2% of the ending branches (which have a dead-end) are present in the oriented graphs. Once they are extracted, a siamese graph neural network model is trained to generate feature vectors that represent sufficiently well the bifurcations to find pairs in the two views. Although imperfect, this proof of concept shows that the method is viable to tackle the task of spatial matching of the coronary artery segments.

TABLE DES MATIÈRES

DÉDICACE iii

REMERCIEMENTS iv

RÉSUMÉ v

ABSTRACT vii

TABLE DES MATIÈRES ix

LISTE DES TABLEAUX xi

LISTE DES FIGURES xii

LISTE DES SIGLES ET ABRÉVIATIONS xv

CHAPITRE 1 INTRODUCTION 1

CHAPITRE 2 REVUE DE LITTÉRATURE 5
- 2.1 Système d'acquisition des séquences angiographiques 5
- 2.2 Concepts de base pour la reconstruction 3D 8
- 2.3 Électrocardiographie 10
- 2.4 Synchronisation 11
- 2.5 Détection de correspondances spatiales 13
- 2.6 Apprentissage profond et réseaux de neurones convolutifs 15
  - 2.6.1 Transfert d'apprentissage 18
  - 2.6.2 Apprentissage profond géométrique 18
- 2.7 Approche proposée 23

CHAPITRE 3 MÉTHODOLOGIE 24
- 3.1 Synchronisation des séquences angiographiques 24
  - 3.1.1 Préparation des données 24
  - 3.1.2 Choix du modèle 28
  - 3.1.3 Génération des matrices de similarité 28
  - 3.1.4 Recherche de chemin 30
  - 3.1.5 Stratégie d'entraînement 31

3.2 Détection de correspondances spatiales 32
3.2.1 Extraction de graphe 33
3.2.2 Processus d'apprentissage 42

CHAPITRE 4 RÉSULTATS 45
4.1 Synchronisation des séquences angiographiques 45
4.1.1 Calcul du score des synchronisations 46
4.1.2 Analyse des résultats de synchronisation 46
4.1.3 Hyperparamètres d'entraînement 48
4.2 Détection de correspondances spatiales 50
4.2.1 Segmentation 50
4.2.2 Description des valeurs des hyperparamètres employés 51
4.2.3 Analyse des résultats de la détection d'ostium 54
4.2.4 Analyse des résultats d'extraction de graphe 55
4.2.5 Entraînement d'un réseau de neurones par graphe pour distinguer les nœuds 56
4.2.6 Résultats de l'entraînement pour distinguer les nœuds 57
4.2.7 Résultats de l'entraînement du modèle siamois 57

CHAPITRE 5 DISCUSSION GÉNÉRALE 61
5.1 Synchronisation des séquences angiographiques 61
5.2 Détection de correspondances spatiales 62
5.2.1 Squelettisation 62
5.2.2 Extraction du graphe orienté 62
5.2.3 Génération de données *ground truth* pour les paires de nœuds 63
5.2.4 Analyse des résultats du GCN siamois 65

CHAPITRE 6 CONCLUSION 67
6.1 Recommandations 68

RÉFÉRENCES 70

LISTE DES TABLEAUX

4.1 Meilleurs hyperparamètres trouvés pour l'entraînement sur données de patients pédiatriques. 49
4.2 Meilleurs hyperparamètres trouvés pour l'entraînement sur données de patients pédiatriques et adultes. 50
4.3 Résultats de mise en correspondance des nœuds à partir de paires de graphes grâce au modèle siamois. 59
4.4 Résultats de mise en correspondance des nœuds à partir de paires de graphes grâce au modèle siamois après un surentraînement de 10 000 époques. 60

LISTE DES FIGURES

1.1 Gauche : C-arm de l'Hôpital Sainte-Justine. Droite : exemple d'image obtenue par angiographie coronaire invasive. 2

2.1 Angles de vue des angiographies coronariennes. Figure tirée du cours GBM6700 enseigné par la professeure Farida Cheriet. 6

2.2 Exemple des paramètres listés dans le fichier DICOM généré lors d'une angiographie. 7

2.3 Exemple d'utilisation de géométrie épipolaire pour estimer la position d'un point X en 3 dimensions à partir de deux vues. O_G et O_D représentent les centres de projection des deux caméras. Les points jaunes e_G et e_D représentent les épipoles. La ligne rouge représente la droite épipolaire. Image libre de droits. Wikimedia Commons (2020). 9

2.4 Apparence normale des ondes présentes dans un ECG. Image libre de droits modifiée. Wikimedia Commons (2020). 10

2.5 Exemple de la méthode RANSAC. Image libre de droits. Wikimedia Commons (2020). 15

2.6 Réseau de neurones artificiels. Image libre de droits. Wikimedia Commons (2020). 17

2.7 CNN typique. Image libre de droits. Wikimedia Commons (2020). . . 17

2.8 Exemple d'un graphe orienté cyclique. Image libre de droits. Wikimedia Commons (2020). 19

2.9 Différentes représentations matricielles d'un graphe orienté. 20

3.1 Sélection des images pertinentes d'une séquence selon leur contraste global. Pour chaque image, la valeur moyenne du gradient est calculée. On leur soustrait ensuite la valeur moyenne du gradient calculée sur toute la séquence, puis les images dont le gradient est au-dessus de la moyenne (section en vert où les valeurs sont supérieures à la ligne rouge) sont gardées pour l'entraînement. 25

3.2 Étapes pour l'identification automatique des ondes R. **A** : valeurs brutes en bleu, moyenne mobile grossière en jaune, moyenne mobile fine en vert. **B** : valeurs normalisées et lissées en bleu, pics potentiels en jaune. **C** : gradient des valeurs normalisées en bleu, maximums locaux en vert et minimums locaux en rouge. **D** : valeurs normalisées en bleu, pics filtrés en rouge. 27

3.3 Identification automatique des ondes R avec exemple de rejet dû à un pic manquant. 28
3.4 Exemple de matrice de données *ground truth*. Chaque élément de la matrice représente le niveau de synchronisation entre l'image du vidéo A et celle du vidéo B. Les lignes rouges représentent les ondes R extraites des ECG. 29
3.5 Exemples de matrices de distance, de similarité et une composition des deux. Les lignes diagonales, où la distance est la plus faible, représentent les paires d'images décrivant un moment du cycle cardiaque similaire. 30
3.6 Exemple du résultat final de la recherche de chemin. Les points de chaque chemin sont représentés par l'ajout de grandes valeurs dans la matrice afin de les faire contraster. 31
3.7 Schéma des artères coronaires. Les ostiums sont identifiés par les flèches vertes. Image libre de droits modifiée. Wikipédia (2020). 33
3.8 Exemple d'artefacts sous forme de fausses branches apparaissant dans le squelette en raison de pixels manquants dans la segmentation. La troisième image montre le résultat obtenu lorsqu'il n'y a plus de pixels manquants. 35
3.9 Exemples de squelettisation. Image libre de droits. Par Robotyczna Owca. 36
3.10 Illustration du problème de croisement de lignes lors de la squelettisation. 37
3.11 Exemples de points de bifurcations d'un squelette. 38
3.12 Exemple de détection de croisement de vaisseaux. Ici, on voit un cathéter superposé à un vaisseau qui présente une vraie bifurcation. . . 39
3.13 Exemple de segments du squelette qui atteignent le haut de l'image. . 40
3.14 Exemples de segments de cathéter près de l'ostium. 41
3.15 Outil d'identification manuel de paires de bifurcations et de points de terminaison. 43
3.16 Vue d'ensemble du modèle de réseau de neurones siamois par graphe. 44
4.1 Animation des résultats de synchronisation sur des séquences angiographiques de patients pédiatriques. La vidéo est disponible ici. . . . 45
4.2 Histogramme cumulatif des scores de synchronisation du meilleur modèle. 47
4.3 Superposition du score des points du chemin (en bleu) et des signaux d'ECG (en rouge). 48

4.4 Exemples d'images angiographiques provenant de patients adultes (haut) et de patients pédiatriques (bas). 49
4.5 Exemple de segmentation automatique imparfaite. Le nombre dans le coin gauche représente le seuil de probabilité de vasculature sur 255. . 51
4.6 Exemple de segmentation corrigée manuellement avec superposition du squelette. 52
4.7 Exemple d'élagage de branches du squelette des vaisseaux. 52
4.8 Histogramme cumulatif des distances entre la position de l'ostium détectée par l'algorithme et celle identifiée manuellement. 54
4.9 Image angiographique où l'algorithme de détection d'ostium performe le moins bien. Le point vert est la position identifiée par l'utilisateur et le point rouge est celle détectée par l'algorithme. 55
4.10 Histogramme cumulatif du ratio du nombre de branches finales des graphes orientés par rapport au nombre de branches finales des graphes non-orientés. 56
4.11 Exemple d'extraction de graphe à partir d'une segmentation. 56
4.12 Vecteurs de caractéristiques à 2 dimensions normalisés générés par le GCN pour chaque nœud d'un graphe. Rangée du haut : valeurs des données d'entraînement avant et après l'apprentissage. Rangée du bas : valeurs des données de validation et de test après l'apprentissage. . . 58
4.13 Progression de la *loss* lors de l'entraînement du GCN siamois pour la mise en correspondance des nœuds d'une paire de graphe. La ligne rouge représente l'époque où la *loss* de validation est à son minimum. 59
5.1 Explication d'un faux positif de détection de croisement. Les segments par lesquels passe la ligne rouge sont détectés comme étant un seul et même segment car un croisement est détecté pour la branche montant vers le haut à partir du centre de l'image alors qu'il s'agit d'une bifurcation. 63
5.2 Exemple de segments du squelette explorés dans la mauvaise direction. Les ronds au bout des lignes droites colorées indiquent la direction d'exploration des segments. Le trait rouge indique le chemin exploré par l'algorithme d'extraction de graphe. 64
5.3 Vecteurs de caractéristiques des nœuds générés par le GCN siamois après l'entraînement. 65

LISTE DES SIGLES ET ABRÉVIATIONS

3DRA	Angiographie coronarienne rotationnelle 3D (Three-Dimensional Rotational Angiography)
ACS	Syndrome coronarien aigu (Acute Coronary Syndrome)
BRIEF	Binary Robust Independent Elementary Features
CAD	Maladie de l'artère coronaire (Coronary Artery Disease)
CHU	Centre hospitalier universitaire
CCTA	Angiographie coronaire par tomodensitométrie (Coronary Computed Tomography Angiography)
CNN	Réseau de neurones convolutifs (Convolutional Neural Network)
DCNN	Diffusion-Convolution Neural Network
DICOM	Digital Imaging and Communications in Medecine
ECG	Électrocardiogramme
GCN	Réseau de neurones par graphe (Graph Neural Network)
ICA	Angiographie coronarienne invasive (Invasive Coronary Angiography)
LCA	Artère coronaire gauche (Left Coronary Artery)
Liv4D	Laboratoire d'Imagerie et de Vision 4D
LSS	Local Self-Similarity
MST	Arbre couvran de poids minimal (Minimal Spanning Tree)
RANSAC	Random Sample Consensus
RCA	Artère coronaire droite (Right Coronary Artery)
ReLU	Rectified Linear Unit
RGB	Rouge vert bleu (Red Green Blue)
SIFT	Scale Invariant Feature Transform
SURF	Speeded Up Robust Features

CHAPITRE 1 INTRODUCTION

Les maladies du cœur sont la deuxième plus importante cause de décès au Canada et affectent près de 1 Canadien adulte sur 12. Elles apparaissent lorsque l'apport sanguin au cœur diminue, souvent causé par le rétrécissement des artères coronaires, ce qui peut entraîner un infarctus du myocarde. Aussi appelée athérosclérose, cette condition est habituellement causée par l'accumulation de plaque formée par la combinaison de calcium et cholestérol sur les parois intérieures des vaisseaux sanguins. Cette accumulation réduit le flux sanguin et peut parfois produire des caillots lors de ruptures de la plaque. Lorsque l'accumulation est sévère, elle peut entraîner un syndrome coronarien aigu et peut causer des dommages aux tissus du cœur, une crise cardiaque ou une insuffisance cardiaque.

Pour diagnostiquer les maladies coronariennes, les cardiologues se fient sur l'électrocardiographie. Cette technique permet de capturer l'influx nerveux envoyé au cœur grâce à un électrocardiographe (12 électrodes placées à différents endroits sur le corps du patient). Cette activité électrique est affichée dans un électrocardiogramme (ECG) sous forme de signal continu sur un graphique. En plus de mesurer le rythme cardiaque du patient, l'ECG permet de détecter une multitude de problèmes cardiaques.

Pour évaluer le degré de sévérité de la maladie, l'angiographie coronaire invasive (ICA) est souvent choisie. Cette procédure, utilisée depuis presque 60 ans, consiste en l'injection d'un agent de contraste dans les artères coronaires à partir d'un cathéter fémoral et visualisée grâce à une machine à rayons X de type C-arm (nommée ainsi car la source et le récepteur sont placés d'une part et d'autre du patient et liés ensemble par un bras courbé, lui donnant l'allure d'un "C") et présentée dans la figure 1.1. Certains modèles permettent d'acquérir des séquences bi-plans à l'aide de deux sources et deux récepteurs placés de manière perpendiculaire afin d'avoir une différence de 90 degrés entre les deux vues. Une technique d'utilisation avancée de ces machines, nommée angiographie coronarienne rotationnelle 3D (3DRA), permet d'acquérir une séquence sous une multitude d'angles en pivotant la source et le récepteur autour du patient pendant l'acquisition. Par la suite, un logiciel s'occupe de combiner certaines images spécifiques d'une séquence (synchronisées sur la phase cardiaque) pour en faire la reconstruction 3D. Cette technique propose plusieurs améliorations telles que la diminution du total des radiations infligées au patient et de la quantité d'agent de contraste utilisé [1]. Bien qu'il soit possible de créer une reconstruction 3D de l'arbre coronaire à partir d'une seule séquence grâce à l'angiographie rotationnelle [2], son usage clinique n'est pas encore répandu puisque cette technique est plutôt récente et qu'elle pose des défis

liés au mouvement cardiorespiratoire intra-scan [3].

Parfois, l'angiographie coronaire par tomodensitométrie est aussi utilisée sur des patients soupçonnés d'être atteints d'athérosclérose. Celle-ci est une autre méthode basée sur l'utilisation de rayons X, mais plutôt que d'employer une source et un récepteur positionnés de manière statique, elle exploite un système rotationnel permettant à la source et au récepteur de tourner rapidement autour du patient afin d'obtenir des images sous plusieurs angles qui sont par la suite fusionnées à l'aide d'un ordinateur pour former un volume 3D. Bien que cette technologie ait grandement évolué depuis son apparition au début des années 2000, les images qu'elle produit demeurent d'une résolution spatiale et temporelle réduite en comparaison avec ICA pour une quantité similaire d'irradiation [4].

En analysant les images générées par l'ICA, il est possible de mesurer la taille de la lumière des artères coronaires grâce à la trajectoire de l'agent de contraste qui est visible sur les images. Mais puisque la plaque ne se forme pas de manière uniforme, il est nécessaire de capturer plusieurs séquences à partir d'angles différents pour évaluer la sévérité de la maladie. De plus, plusieurs artéfacts courants se retrouvent dans ce type d'image. Par exemple, les structures osseuses et les organes sont en superposition avec les vaisseaux, les vaisseaux se superposent aussi entre eux, le gras ajoute du flou à l'image et une quantité insuffisante d'agent de contraste peut rendre difficile l'interprétation des images.

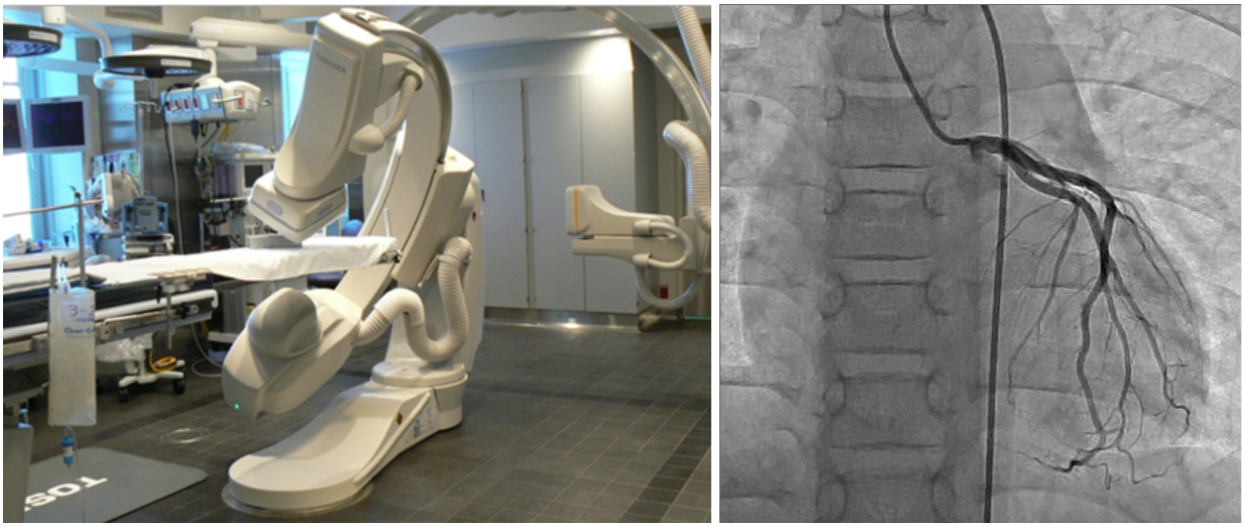


Figure 1.1 Gauche : C-arm de l'Hôpital Sainte-Justine. Droite : exemple d'image obtenue par angiographie coronaire invasive.

Puisque l'angiographie 2D possède son lot de désavantages et limitations, l'évaluation fidèle de l'anatomie et de la pathologie de la vasculature coronarienne est parfois difficile. Afin d'aider les cardiologues à diagnostiquer les maladies coronariennes, une reconstruction 3D+t

de l'arbre coronaire peut s'avérer utile. La reconstruction 3D pourrait permettre le calcul des angles de vue optimaux, la sélection des équipements interventionnels (tel que la taille de stent pour restaurer le flux sanguin dans les artères sténosées), la simulation des procédures cardiovasculaires, fournir un système de navigation sous fluoroscopie et bien plus [5] [6]. L'aspect dynamique de la reconstruction pourrait permettre le calcul automatique de l'élastoplasticité des artères et le suivi de la progression de l'agent de contraste dans l'arbre coronarien. La raison pour laquelle l'utilisation de techniques de reconstruction 3D n'est pas encore courante en clinique est qu'elle doit être à la fois robuste, précise et temps-réel, ce qui représente un grand défi [6]. Pour cette raison, la tâche de reconstruction 3D ou même 3D+t demeure pertinente aujourd'hui. D'autre part, une reconstruction 3D durant les acquisitions permettrait d'optimiser le nombre de projections afin d'éviter le cumul d'irradiation au patient et au personnel de santé. La reconstruction 3D+t est d'ailleurs l'objectif à long terme de ce projet. Afin de réaliser cet objectif, une mise en correspondance spatio-temporelle des séquences est essentielle et constitue l'objectif principal de ce projet de recherche. Afin de répondre au besoin d'une reconstruction en ligne durant l'acquisition des images, des techniques d'apprentissage profond ont été choisies étant donné leur rapidité d'exécution et leur performance liées aux tâches de vision par ordinateur.

Les objectifs spécifiques qui doivent être atteints pour la mise en correspondance temporelle des séquences sont les suivants :

1. Construction d'une banque de données d'entraînement à partir de signaux ECG.
2. Choix d'un modèle d'apprentissage profond.
3. Définition d'une métrique de synchronisation de séquences angiographiques.
4. Évaluation des performances du modèle proposé.

Les objectifs spécifiques qui doivent être atteints pour la mise en correspondance spatiale sont les suivants :

1. Représentation de l'arbre coronarien sous forme de graphe orienté.
2. Construction d'une banque de données d'entraînement à partir des points de bifurcations de l'arbre coronarien.
3. Réalisation une preuve de concept pour la mise en correspondance des nœuds d'une paire de graphes orientés à l'aide d'un réseau de neurones par graphe.

Ce mémoire est présenté sous forme classique. Il contient 6 chapitres, incluant l'introduction et la conclusion. Au chapitre 2, une revue de littérature décrit les différentes recherches pertinentes consultées à propos de plusieurs sujets en lien avec l'angiographie coronaire, la reconstruction 3D et l'apprentissage profond. Au chapitre 3, le travail effectué pour atteindre

les objectifs de recherche est expliqué en détail. Au chapitre 4, les résultats des expériences sont listés puis analysés. Au chapitre 5, les résultats décrits au chapitre précédent sont discutés et les limitations des algorithmes développés sont mises en lumière. Finalement, la conclusion revient sur l'atteinte des objectifs et recommande des pistes de solutions aux limitations présentées dans la discussion pour les travaux futurs.

CHAPITRE 2 REVUE DE LITTÉRATURE

Cette section présente un résumé de la littérature à propos des différents sujets en lien avec le projet. Premièrement, le système d'acquisition des images angiographiques est décrit, suivi d'une description de l'utilité de l'électrocardiographie pour l'appariement d'images angiographiques. Ensuite, une revue de différentes méthodes pour la synchronisation de séquences ainsi que pour la détection de correspondances spatiales est présentée. Finalement, un résumé de l'apprentissage profond, de l'utilité du transfert d'apprentissage et de l'usage des réseaux de neurones sur des données sous forme de graphes est documenté.

2.1 Système d'acquisition des séquences angiographiques

Lors de l'acquisition des séquences angiographiques, plusieurs angles de vue sont utilisés pour bien visualiser les artères. Les plus fréquents sont sans doute l'angle antéro-postérieur (AP) et l'angle latéral gauche (LAT), dont l'incidence des rayons X est parallèle au sol, passant de l'arrière du patient vers l'avant et du côté droit du patient vers son côté gauche, respectivement. Tel que montré par la figure 2.1, les autres vues sont identifiées par leur angle primaire sur le plan transverse, nommé latéral LAO (*Left Anterior Oblique*) s'il est positif ou RAO (*Right Anterior Oblique*) s'il est négatif et par leur angle secondaire sur le plan sagital, nommé crânial s'il est positif ou caudal s'il est négatif.

En plus des angles de vue, d'autres paramètres sont disponibles dans les fichiers DICOM générés par les C-arms. Par exemple, on peut y trouver la distance entre la source et le détecteur, la distance entre la source et le patient, la durée entre chaque trame, la taille de chaque pixel et bien plus. Un exemple est présenté par la figure 2.2.

Pour former les images lors d'une angiographie coronaire invasive, le C-arm génère des rayons X en accélérant des électrons grâce à un tube à rayons X à haut voltage. Cette source est placée devant le patient qui est ensuite traversé par ces électrons. Les tissus denses tels que les os du patient absorbent une partie des rayons X alors que les tissus moins denses tels que les organes laissent passer la majorité de ceux-ci. L'agent de contraste injecté dans les artères est lui aussi d'une densité élevée et donc absorbe les rayons X. Le récepteur, placé du côté opposé au patient, récolte les rayons X qui n'ont pas été absorbés et génère des signaux électriques lors de la réception des électrons. Ces signaux sont finalement interprétés par un ordinateur pour générer une image digitale composée de tons de gris, où un pixel clair signifie que plusieurs électrons ont été reçus à cet endroit.

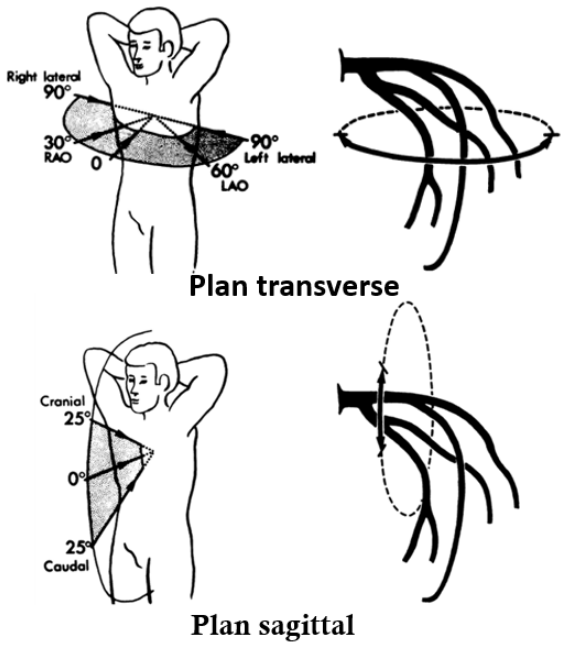


Figure 2.1 Angles de vue des angiographies coronariennes. Figure tirée du cours GBM6700 enseigné par la professeure Farida Cheriet.

Cette procédure n'est préférablement appliquée que dans les cas qui la nécessite, car elle pose quelques dangers pour le patient. L'agent de contraste, typiquement composé d'iode, est toxique pour les reins, ce qui peut forcer les cliniciens à limiter la durée de l'intervention. De plus, une exposition prolongée aux rayons X entraîne des risques de cancer accrus. Finalement, le caractère invasif de la procédure amène des risques d'infections.

Le modèle de formation des images angiographiques est similaire à celui du sténopé, qui se base sur les paramètres intrinsèques et extrinsèques de la configuration du système définis lors de la phase de calibrage de celle-ci. Les paramètres intrinsèques forment la matrice K de taille 3x4 qui contient principalement la distance principale d qui définie la distance entre la source aux rayons X et le détecteur ainsi que les coordonnées de projection du centre optique de la source aux rayons X sur le plan image (u_0, v_0). Les paramètres extrinsèques, quant à eux, forment la matrice RT qui contient la matrice de rotation R de taille 3x3 qui permet d'indiquer l'angle d'incidence par rapport à la scène ainsi que le vecteur de translation T de 3 dimensions permettant de définir la position de la source aux rayons X par rapport à la scène. Une multiplication de ces deux matrices avec les coordonnées 3D exprimées sous forme homogène $(x, y, z, 1)$ d'un point dans la scène permet de déterminer les coordonnées 2D exprimées elles aussi sous forme homogène $(u, v, 1)$ du même point dans le plan image sur le détecteur, tel que présenté par l'équation (2.1).

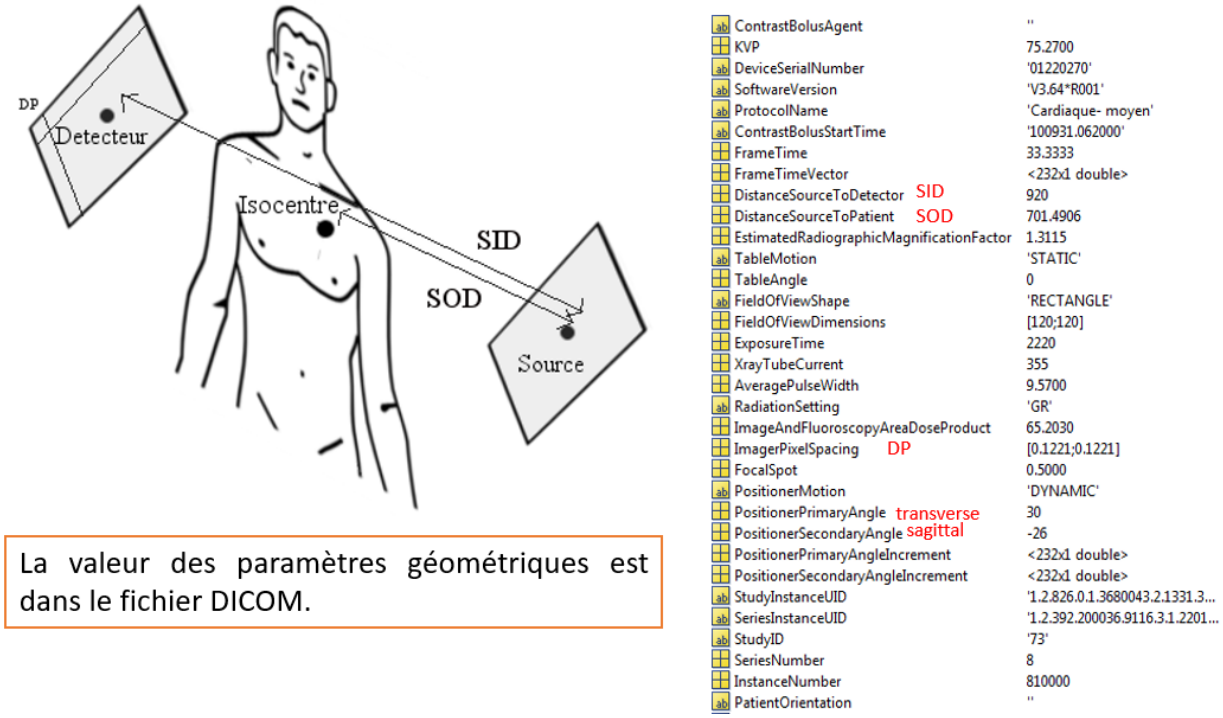


Figure 2.2 Exemple des paramètres listés dans le fichier DICOM généré lors d'une angiographie.

$$
\begin{aligned}
\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} &= K \begin{bmatrix} R & T \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \\
K &= \begin{bmatrix} d & 0 & u_0 & 0 \\ 0 & d & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\
\begin{bmatrix} R & T \end{bmatrix} &= \begin{bmatrix} & & & t_x \\ & R_{3\times3} & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{2.1}
$$

Une fois le système calibré, la position 3D d'une structure anatomique peut être retrouvée à partir de deux images acquises à deux incidences différentes en utilisant une triangulation stéréoscopique.

2.2 Concepts de base pour la reconstruction 3D

La tâche de reconstruction 3D à partir d'images 2D appartient au domaine de vision par ordinateur. Pour y arriver, plusieurs techniques sont viables. Celles-ci peuvent être classées dans deux groupes, c'est-à-dire les méthodes basées sur la projection vers l'avant et celles basées sur la rétroprojection [6].

Les méthodes du premier groupe fonctionnent en déformant un modèle 3D pour l'adapter aux structures des vaisseaux extraites des images 2D en optimisant des fonctions d'énergie basées sur le contour des vaisseaux ou l'intensité des pixels. Bien que certaines de ces méthodes ne dépendent pas d'une segmentation robuste ni de mise en correspondence entre les lignes centrales des vaisseaux d'une vue à l'autre, celles-ci se fient généralement sur une sélection manuelle de chaque branche d'artère, ce qui peut être long et susceptible d'erreurs.

Les méthodes du deuxième groupe, exploitant la rétroprojection, fonctionnent soit grâce à une rétroprojection directe des vaisseaux à partir de plusieurs vues, soit grâce à des mises en correspondance spatiales 2D entre deux vues. Celles qui utilisent une rétroprojection directe permettent l'utilisation d'une multitude de vues en simultané pour effectuer la reconstruction 3D. Un exemple de ce genre de méthode est celle de *shape from silhouette*, qui reconstruit la vasculature à partir de la silhouette des artères et raffine le résultat en ajoutant plus de vues. Bien que cette méthode soit automatique et suffisamment rapide pour du temps réel, elle est très sensible aux erreurs de calibrage [7]. Celles qui se fient sur les mises en correspondance spatiales commencent par une synchronisation des images angiographiques, suivi d'une segmentation des artères et d'une identification de points clés. Elles terminent ensuite par une mise en correspondance de ceux-ci et d'une méthode de triangulation pour les placer dans l'espace 3D.

La détection de correspondances entre deux vues d'angiographie coronarienne est un problème difficile à résoudre pour plusieurs raisons. Premièrement, puisque la différence de l'angle d'incidence entre les deux vues peut être très grande, la forme de la vasculature peut grandement différer d'une vue à l'autre. Deuxièmement, puisque les vaisseaux visibles dans les images possèdent peu de détails, les caractéristiques des points sont toutes semblables. Troisièmement, en raison de la semi-transparence des structures visualisées par l'imagerie par rayons X, les segments d'artères à l'avant et en arrière du cœur sont visibles à la fois, ce qui cause une superposition des vaisseaux, en plus d'ajouter des artéfacts occasionnés par la présence de la cage thoracique et des organes.

La géométrie épipolaire est alors utile pour réduire le domaine de recherche de correspondances si l'on connaît les paramètres intrinsèques et extrinsèques du système radiographique.

En effet, à partir de deux images de la même scène acquises selon des angles différents, il est possible de tracer une droite dans chaque image sur laquelle un point identifié dans l'autre image se doit d'être (voir figure 2.3). Cette droite se nomme la droite épipolaire et passe par l'épipole, point d'intersection du plan image et d'une droite passant par les centres de projection des deux caméras. Puisque l'on sait que le point doit passer par cette droite, il devient possible de limiter grandement le nombre de candidats potentiels pour la mise en correspondance du point identifié dans la première image. Par contre, la contrainte épipolaire n'est pas suffisante pour lever l'ambiguïté au niveau de la mise en correspondance des structures dû à la semi-transparence du modèle d'acquisition qui implique une superposition de toutes les structures anatomiques sur l'image. De plus, les systèmes radiographiques doivent être parfaitement calibrés pour que le point se trouve sur la droite épipolaire. Plus le calibrage comporte d'erreurs, moins cette technique est fiable. Finalement, si l'objet dans la scène s'est déplacé entre la capture des deux images, des erreurs additionnelles sont introduites.

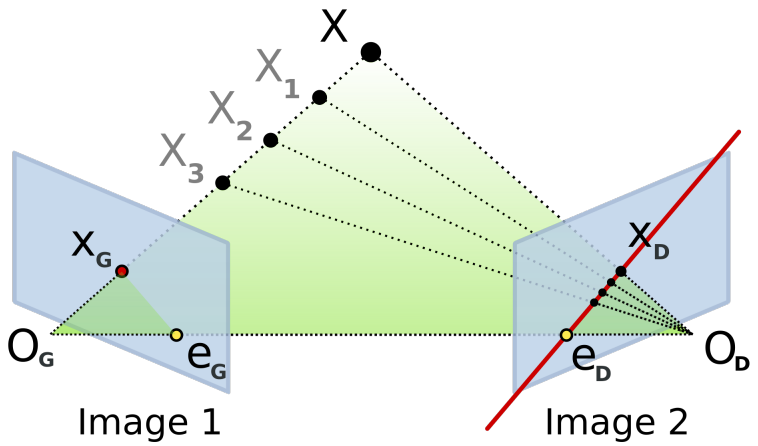


Figure 2.3 Exemple d'utilisation de géométrie épipolaire pour estimer la position d'un point X en 3 dimensions à partir de deux vues. O_G et O_D représentent les centres de projection des deux caméras. Les points jaunes e_G et e_D représentent les épipoles. La ligne rouge représente la droite épipolaire. Image libre de droits. Wikimedia Commons (2020).

Pour obtenir une reconstruction 3D fidèle à partir d'une méthode basée sur la mise en correspondance de points clés, il est donc nécessaire de disposer d'une synchronisation précise des images pour réduire au maximum l'impact du mouvement cardiorespiratoire sur la position des artères. Cette tâche est habituellement faite grâce à l'électrocardiographie.

2.3 Électrocardiographie

Tel que mentionné dans l'introduction, l'électrocardiographie est habituellement utilisée avant même d'avoir recours à l'angiographie. Elle est d'ailleurs fréquemment utilisée pendant l'angiographie. Les techniques standards de reconstruction 3D exploitent bien souvent les signaux générés par l'électrocardiographe, car le mouvement cardiaque cause un déplacement des artères qui rend difficile la reconstruction 3D. Pour éviter ce problème, il est possible de recourir au *gating*, qui consiste en l'utilisation d'images spécifiques choisies lorsque le mouvement du cœur est à son plus faible, à la fin de la phase diastolique. Pour se faire, on doit détecter la fin de la phase diastolique à partir des ondes de l'ECG. Sur les courbes générées, on peut y reconnaître 5 types d'ondes (voir Figure 2.4). Le complexe QRS, composé des ondes Q, R et S, est lié à la dépolarisation ventriculaire ainsi qu'à la repolarisation auriculaire. En d'autres mots, il s'agit du moment où les ventricules sont gorgés de sang, juste avant qu'ils se contractent, ce qui consiste à la transition entre les phases diastolique et systolique. L'onde P, quant à elle, est de faible amplitude et résulte de la dépolarisation auriculaire. L'onde T est généralement deux fois plus large que l'onde P et est causée par la repolarisation ventriculaire.

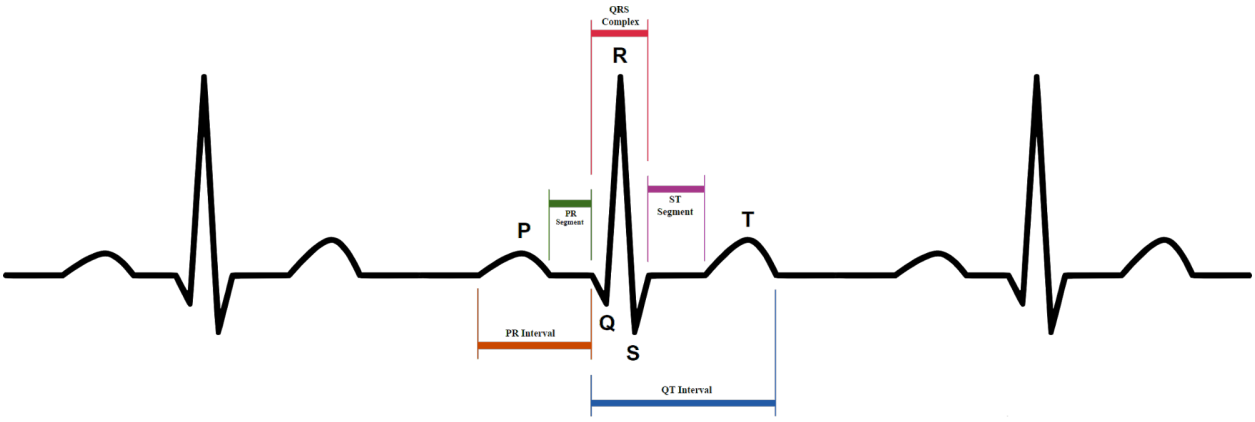


Figure 2.4 Apparence normale des ondes présentes dans un ECG. Image libre de droits modifiée. Wikimedia Commons (2020).

En plus de permettre le *gating*, il est possible d'analyser les ondes présentes dans un ECG pour identifier un infarctus du myocarde. Une élévation du segment ST peut indiquer ce genre de maladie. Par contre, l'ECG n'est pas suffisamment précis pour diagnostiquer des sténoses significatives à lui seul [8]. Si l'ECG n'affiche pas d'élévation du segment ST mais que le patient présente des douleurs thoraciques ou autres symptômes d'angine, d'autres tests sont nécessaires, tels qu'une analyse sanguine et un test de stress cardiovasculaire. Il peut aussi arriver que l'ECG présente une dépression du segment ST ou une inversion de l'onde T, ce qui peut aussi indiquer un infarctus du myocarde.

Bien que très utile pour diagnostiquer plusieurs maladies cardiovasculaires, l'ECG n'est pas exempt d'artéfacts. Ceux-ci sont généralement causés par des interférences provenant de signaux électriques, parfois externes, parfois internes. Des exemples de causes externes peuvent inclure des dispositifs très près du module de capteur tels que des téléphones cellulaires, lits électriques ou lampes chirurgicales. Des exemples de causes internes peuvent inclure des mouvements du patient, des spasmes musculaires, un nettoyage inapproprié de la peau, un excès de gel conducteur précordial ou encore un mauvais positionnement des électrodes. Ces interférences sont généralement représentées dans l'ECG par des artéfacts ayant pour effet de déformer les ondes, tel qu'une inversion de l'onde R, une augmentation d'intensité de l'onde T, un ajout de pics en forme de dents de scie et bien d'autres [9].

2.4 Synchronisation

La tâche de reconstruction 3D à partir de séquences d'objets en mouvement nécessite des images synchronisées pour réduire les erreurs et le *gating* peut échouer si des artefacts sont présents dans les ECG. De plus, le fait de ne garder que quelques images d'une séquence cause une perte d'information temporelle. Pour ces raisons, plusieurs chercheurs se sont penchés sur le problème de synchronisation de trames de vidéos à partir des pixels des images. Certains travaux utilisent des notions de suivi d'objets, d'autres se fient sur des méthodes statistiques basées sur la fluctuation de l'intensité des pixels alors que plusieurs autres tirent profit des réseaux de neurones convolutifs pour extraire des descripteurs d'images.

Pour synchroniser des séquences à partir du suivi d'objets, plusieurs options sont disponibles. Il est possible d'identifier des trames clés à partir de points d'inflexion dans la trajectoire des objets afin de pouvoir aligner les séquences [10]. Une autre méthode nécessite de détecter spatialement des correspondances entre la première image des différentes vues. Ensuite, on doit comparer les trajectoires de ces points à l'aide d'une matrice d'observation basée sur la matrice de projection de la caméra et permettant d'extraire des mesures invariantes à l'angle de vue. Grâce à la matrice fondamentale définie par les correspondances entre les points, il est possible de calculer une mesure de similarité entre les trajectoires. On peut ensuite effectuer la déformation temporelle dynamique des vidéos en optimisant une fonction d'erreur d'alignement qui utilise ces similarités de trajectoires [11]. Une technique plus récente, applicable seulement pour la synchronisation de séquences angiographiques des artères coronaires, détecte et calcule la trajectoire de points clés situés sur les vaisseaux. Selon la vélocité du mouvement de ces points clés, la phase cardiaque est estimée et utilisée pour synchroniser les séquences [12]. Ces méthodes reposent entièrement sur un suivi précis et continu d'éléments dans les images angiographiques, ce qui n'est pas toujours possible (par exemple, lorsque

l'agent de contraste n'est pas injecté de manière uniforme).

Certains chercheurs préfèrent recourir à des méthodes statistiques plutôt que de faire du suivi d'objets pour résoudre la tâche de synchronisation des séquences angiographiques. En calculant le déphasage de différentes fréquences issues d'une transformation de Fourier sur l'image complète, il est possible de déterminer le mouvement cardiaque ainsi que le mouvement respiratoire, permettant ensuite de synchroniser différentes séquences [13]. Une autre méthode applique l'analyse par composantes principales sur les pixels appartenant aux vaisseaux. Cela permet d'identifier les vecteurs propres qui, croisés avec les valeurs des pixels, servent à déterminer les signaux respiratoires et cardiaques. Ces signaux sont ensuite utilisés pour effectuer du gating cardiorespiratoire [14]. Ces méthodes, bien que très rapides, échouent elles aussi lorsque l'agent de contraste est injecté de manière non uniforme.

L'autre catégorie de travaux s'attaquant au problème de synchronisation repose sur les réseaux de neurones profonds. Un premier propose un *framework* permettant de synchroniser deux vidéos en extrayant un descripteur à 1000 valeurs pour chaque image à partir d'un ResNet-50 pré-entraîné. Les descripteurs du premier vidéo sont ensuite comparés à ceux du deuxième pour générer une matrice de coûts. Finalement, l'algorithme de Dijkstra est appliqué sur la matrice de coûts pour trouver un chemin représentant les paires d'images synchronisées. Le modèle est entraîné à partir de données non annotées de vidéos de trajets d'une voiture en ville à différentes saisons enregistrés par une caméra placée sur un tableau de bord [15]. Un deuxième propose lui aussi de comparer les descripteurs des images de deux vidéos. La différence principale avec le premier est la manière dont les paires d'images sont choisies pour la synchronisation. Plutôt que de recourir à la recherche de chemin dans une matrice de coût, cette méthode repose sur le double usage de l'algorithme du plus proche voisin pour s'assurer que deux trames sont similaires d'une vidéo à l'autre et vice-versa. Ce modèle est entraîné sur des banques de courtes séquences d'actions similaires (par exemple verser du jus dans un verre) où certaines trames clés sont manuellement identifiées [16]. Bien que plutôt précise, cette méthode est faite pour synchroniser des séquences présentant une action ayant un début et une fin, ce qui n'est pas le cas pour les séquences angiographiques puisque le battement cardiaque est cyclique. Un troisième propose de détecter la progression de la phase cardiaque directement à partir d'une séquence d'images angiographiques grâce à un réseau de neurones convolutif prenant 10 images consécutives en entrée. La sortie du modèle est une valeur entre 0 et 1 où une valeur en dessous de 0.5 signifie que l'image au centre de la séquence extraite est en phase de systole, ou en phase de diastole si sa valeur est au dessus de 0.5. Ce modèle obtient une précision de 97.6% grâce à son entraînement fait sur un jeu de données composé de 17800 séquences angiographiques provenant de 3900 patients. Son apprentissage est alimenté par une fonction de perte basée sur l'identification

des phases cardiaques à partir des signaux ECG enregistrés parallèlement aux séquences angiographiques [17]. Ce modèle de synchronisation présente d'impressionnants résultats, mais dépend d'une immense quantité de données qui n'est pas toujours disponible en clinique.

2.5 Détection de correspondances spatiales

La grande majorité des articles récents issus de recherches en matière de détection de correspondances entre deux images (communément appelé "matching") sont basées sur les réseaux de neurones profonds. Par contre, cette tâche a été abordée par les chercheurs bien avant la montée en popularité des réseaux de neurones. Il y a donc plusieurs travaux ayant marqué le domaine qui valent la peine d'être mentionnés puisqu'ils demeurent une référence même pour les méthodes récentes.

Le descripteur local SIFT (acronyme de "Scale Invariant Feature Transform") est un incontournable, puisqu'il est très robuste. L'algorithme présenté dans l'article [18] propose de détecter des points clés d'une image dans plusieurs résolutions créées à partir de différences de gaussiennes. Un pixel est défini comme point clé si son intensité est un maxima ou un minima parmi les 8 pixels qui l'entourent et les neufs pixels voisins pour les deux résolutions adjacentes (deux fois plus fine et deux fois plus grossière). Par la suite, un filtrage des points clés est fait pour ne garder que les points ayant un contraste élevé. Pour détecter les coins, une matrice Hessienne 2x2 est calculée à la position du point clé et si le gradient est élevé dans deux directions perpendiculaires, le pixel est considéré comme un coin. Par la suite, une invariance à la rotation est assurée en donnant une orientation au point clé, définie par la direction principale du gradient dans le voisinage du point clé. Pour déterminer cette direction, on doit d'abord calculer l'orientation et l'intensité du gradient pour chaque pixel dans une région de 16x16 autour du point clé et placer celles-ci dans un histogramme à 36 classes (10 degrés par classe). La classe ayant la plus forte amplitude détermine l'orientation du point clé. Finalement, pour créer le descripteur caractérisant le point clé, les orientations et intensités de gradient des points du voisinage sont agrégées par groupes de 4x4 pour 8 directions, ce qui crée 16 sous-régions de 8 valeurs pour un vecteur de taille 128 [18].

Par la suite, une multitude de nouveaux descripteurs ont été créés, avec chacun leurs forces et faiblesses. Le descripteur SURF (acronyme de "Speeded Up Robust Features") est très similaire à SIFT mais approxime les différences de gaussiennes avec des *box filters* et utilise les ondelettes pour créer le vecteur de caractéristiques, ce qui lui permet d'être moins lourd en termes de calculs [19]. Le descripteur DAISY (à cause de sa représentation visuelle s'apparentant à une marguerite) calcule le gradient dans plusieurs régions autour du point clé qui augmentent de taille proportionnellement à leur distance avec le centre, ce qui donne un

descripteur plus précis lorsque l'image change peu, ce qui est idéal pour les problèmes de vision stéréo [20]. Le descripteur BRIEF (acronyme de "Binary Robust Independent Elementary Features") effectue des comparaisons entre des pixels aléatoires dans le voisinage pour créer un vecteur de caractéristiques binaires lui permettant d'être encore plus rapide [21]. Le descripteur LSS (acronyme de "Local Self-Similarity") fait abstraction des couleurs et textures pour représenter uniquement la forme [22]. La méthode DeepMatching offre une robustesse par rapport aux effets des mouvements non rigides en calculant la corrélation entre des groupes de pixels sur plusieurs niveaux de résolution à l'aide de convolutions [23].

Une autre catégorie de descripteurs ayant vu le jour dans la dernière décennie est celle faisant usage des réseaux de neurones convolutifs pour apprendre des descripteurs à partir de grandes banques de données. Ces méthodes permettent d'apprendre un espace de caractéristiques qui préserve la similarité sémantique, ce qui peut servir à générer des descripteurs capables de trouver des correspondances parmi différentes instances d'une même classe plutôt que de simplement trouver des correspondances de la même instance [24]. Certains modèles font usage d'apprentissage non supervisé à l'aide de séquences vidéos pour permettre à leur réseau de neurones d'apprendre à faire des correspondances entre les images consécutives [25]. Un autre travail notable est celui des "Neighborhood Consensus Networks" qui apprend à trouver des correspondances de manière non supervisée à partir de paires d'images en calculant des cartes de correspondances denses. Par la suite, les correspondances sont ajustées pour réduire l'importance de celles n'étant pas des plus proches voisins mutuels et des opérations de convolutions sont faites pour détecter les groupes de correspondances fortes [26].

Générer des vecteurs de caractéristiques n'est que la première étape de la tâche de détection de correspondances. Il est nécessaire de les comparer entre eux par la suite afin de déterminer lesquels se ressemblent le plus. Pour se faire, plusieurs métriques de distance sont viables dont certaines plus appropriées que d'autres dans certains scénarios. La distance de Manhattan est probablement la plus basique, puisqu'il s'agit simplement de la somme des différences entre les valeurs des deux vecteurs. La distance Euclidienne est très semblable, calcule la distance en ligne droite entre deux points plutôt que de faire la somme des distances pour chaque dimension. Lorsque l'amplitude des vecteurs n'est pas aussi importante que leur orientation, la similarité cosinus est pertinente puisqu'elle ne fait que calculer l'angle entre les deux vecteurs grâce à un produit scalaire normalisé. La distance de Hamming, quant à elle, est faite pour comparer des vecteurs de caractéristiques dont les valeurs sont catégoriques puisqu'elle somme le nombre de valeurs qui sont différentes.

La dernière étape de la détection de correspondances est l'identification des paires concordantes et des paires aberrantes. Pour se faire, il est nécessaire d'estimer les paramètres du

modèle. Une méthode simple est celle des moindres carrés, mais celle-ci n'est pas bien adaptée à cette tâche puisqu'elle est sensible aux données aberrantes. Bien qu'il soit possible par diverses manières d'identifier des données aberrantes en comparant chaque valeur avec l'ensemble de données pour ignorer celles qui ont un score z trop élevé, cette technique est plutôt difficile à appliquer dans un contexte où la proportion de données aberrantes est élevée. La méthode RANSAC (acronyme de "Random Sample Consensus"), pour sa part, est très robuste dans ces situations et c'est pour cette raison qu'elle est souvent choisie pour la tâche de détection de correspondances. Elle fonctionne en sélectionnant un minimum de paires pour déterminer les paramètres du modèle et ensuite compter le nombre de paires plausibles selon ces paramètres. Cette opération est refaite un grand nombre de fois afin de s'assurer qu'au moins un ensemble de départ ne contienne que des paires plausibles puis les paramètres du modèle dont le nombre de paires plausibles est le plus élevé est gardé [27].

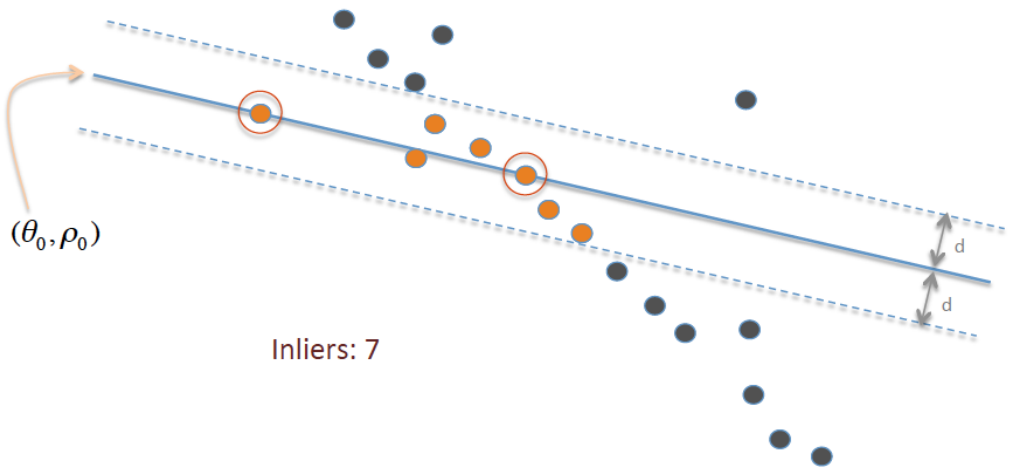


Figure 2.5 Exemple de la méthode RANSAC. Image libre de droits. Wikimedia Commons (2020).

2.6 Apprentissage profond et réseaux de neurones convolutifs

Depuis AlexNet [28], toutes les compétitions de traitement d'images ont été remportées par des réseaux de neurones profonds convolutifs. Il est donc pertinent de s'attarder à ceux-ci pour comprendre comment ils peuvent être utiles.

L'apprentissage automatique est une forme d'intelligence artificielle qui fonctionne en entraînant un modèle à effectuer une tâche spécifique à partir de données, plutôt que de reposer sur des connaissances d'expert codées directement dans un logiciel. Plusieurs méthodes appar-

tiennent à la classe d'apprentissage automatique, telles que les machines à vecteurs de support, les arbres de décision, la régression logistique et bien d'autres. L'apprentissage profond est aussi une sous-classe de l'apprentissage automatique et a beaucoup gagné en popularité dans la dernière décennie. Bien que l'idée d'utiliser des neurones artificiels date des années 40, quelques problèmes techniques ainsi que le manque de puissance des ordinateurs de l'époque a grandement ralenti la progression de cette branche de recherche. Mais avec la constante amélioration des microprocesseurs et des cartes graphiques, l'utilisation des réseaux de neurones profonds est redevenu un sujet de recherche d'actualité. On voit maintenant qu'une grande puissance de calcul leur permet de bien s'adapter aux données utilisées lors de l'entraînement et que leur performance ainsi que leur capacité de généralisation tend à augmenter avec la quantité de données.

Les réseaux de neurones peuvent avoir différentes structures, mais sont en général composés de plusieurs couches de neurones connectées les unes aux autres (ces liens entre les couches sont souvent appelés des poids) et utilisées sous forme de multiplications de matrices. Ces couches sont habituellement séparées par des opérations non linéaires, dont la plus populaire est sans doute l'opération de ReLU (Rectified Linear Unit) qui ne fait que garder les valeurs positives et transformer les valeurs négatives en 0. Ces opérations non linéaires aident à manipuler l'espace des données afin de mieux séparer celles-ci, ce qui permet aux réseaux de neurones d'identifier et faire ressortir des tendances dans les données. Pour conférer l'habileté d'apprentissage au modèle, l'optimisation des poids se fait grâce à la rétropropagation qui est basée sur des dérivées partielles en chaîne. La rétropropagation est nourrie par une fonction de perte différentiable qui spécifie à quel point les résultats du réseau sont erronés par rapport à une entrée spécifique. Pendant l'apprentissage, le modèle est alimenté par des données provenant de l'ensemble d'entraînement et évalué sur des données de l'ensemble de validation. Plus l'entraînement avance, plus le modèle performe généralement bien sur les données de l'ensemble d'entraînement ainsi que sur les données de l'ensemble de validation. Après plusieurs époques (terme référant habituellement à l'utilisation complète de l'ensemble de données), les résultats sur l'ensemble de validation tendent à se dégrader car le modèle fait du surapprentissage sur les données de l'ensemble d'entraînement. Ce qui définit la fin de l'entraînement est un nombre maximal d'époques ou l'arrêt prématuré du processus basé sur la performance du modèle sur les données de l'ensemble d'évaluation. Une fois l'entraînement terminé, on choisit la version des poids où le modèle a performé le mieux sur les données de l'ensemble de validation pour définir le modèle entraîné.

Parmi les différentes structures de réseaux de neurones, celles avec des couches de convolution sont présentement les mieux adaptées pour résoudre les tâches de vision par ordinateur. Inventés par Yann LeCun dans les années 1980, les réseaux de neurones convolutifs (CNN)

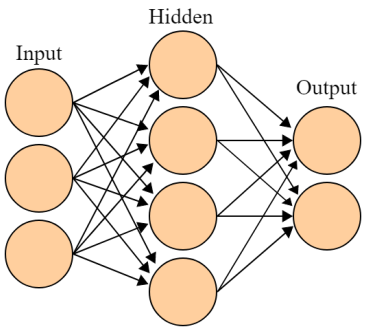


Figure 2.6 Réseau de neurones artificiels. Image libre de droits. Wikimedia Commons (2020).

sont composés de multiples couches de neurones artificielles sur lesquelles des opérations de convolution sont effectuées. Ils prennent en entrée une image de dimensions largeur par hauteur par nombre de canaux (habituellement 1 pour les images en noir et blanc ou 3 pour les images en couleurs, soit rouge, vert et bleu) où chaque valeur représente l'intensité d'un pixel entre 0 et 1. La sortie des CNN, quant à elle, diffère selon la tâche qu'ils tentent de résoudre. Lorsque la tâche est de simplement décrire l'image, un descripteur sous forme de vecteur de nombres à virgule flottante représente la sortie du réseau. Pour une tâche de classification, une couche de type *softmax* est ajoutée afin de déterminer à quelle classe l'image appartient. Dans ce cas, le vecteur en sortie représente une sorte de niveau d'appartenance de l'image à chaque classe. LeNet [29] et AlexNet, les réseaux de Yann LeCun et Alex Krizhevsky, font partie de cette catégorie de réseaux puisqu'ils ont pour but de classifier des images de chiffres écrits à la main et les images d'ImageNet appartenant à 1000 classes différentes, respectivement. Pour une tâche de segmentation, la sortie est de même taille que l'image d'entrée, mais plutôt que de posséder 3 canaux, elle est composée d'un canal par classe où chaque valeur est un booléen indiquant si le pixel appartient ou non à la classe en question.

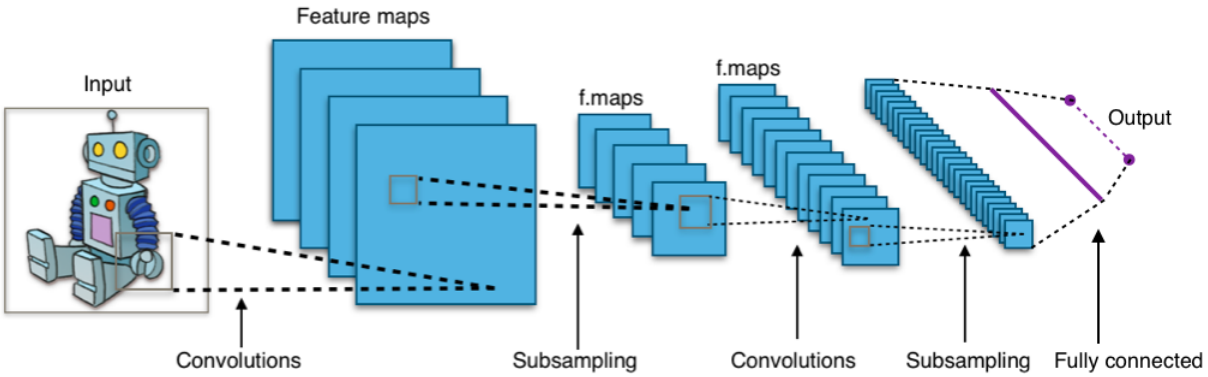


Figure 2.7 CNN typique. Image libre de droits. Wikimedia Commons (2020).

2.6.1 Transfert d'apprentissage

Puisque entraîner un réseau de neurones convolutif de zéro peut prendre beaucoup de temps et nécessiter l'utilisation de plusieurs cartes graphiques haut de gamme et qu'une très grande quantité d'images est nécessaire pour le rendre robuste, il n'est pas rare aujourd'hui de réutiliser des poids de réseaux déjà entraînés sur ImageNet. De plus, il est parfois utile de réutiliser la structure des modèles de l'état de l'art plutôt que de créer sa propre structure de réseau de neurones qui bien souvent ne permettrait pas d'obtenir d'aussi bons résultats. En prenant avantage d'un modèle pré-entraîné, il devient possible de profiter d'une structure de réseau sofistiquée tout en réduisant considérablement le temps d'entraînement requis. Lorsqu'on utilise un *framework* d'apprentissage automatique tel que PyTorch [30], il est très simple d'importer un modèle pré-entraîné. Par exemple, la bibliothèque "torchvision.models" contient plusieurs modèles pré-entraînés tels que les ResNets [31] et les MobileNets [32]. Lorsque l'on veut utiliser ceux-ci à partir de cette bibliothèque, leurs poids se téléchargent automatiquement pour qu'ils soient prêts à être utilisés.

Pour résoudre une tâche qui diffère de celle pour laquelle le modèle a été pré-entraîné (habituellement une classification d'images d'ImageNet), il suffit de retirer la dernière couche entièrement connectée située à la fin du réseau et la remplacer par une nouvelle réinitialisée aléatoirement. Par la suite, un réglage fin est nécessaire pour adapter le modèle pré-entraîné aux données de la nouvelle tâche que l'on souhaite résoudre [33]. Cet entraînement est généralement beaucoup plus rapide que l'entraînement complet du réseau, puisque les dernières couches du réseau sont moins longues à entraîner que les précédentes étant donné le problème du gradient disparaissant [34].

2.6.2 Apprentissage profond géométrique

Les réseaux de neurones classiques sont faits pour traiter des données provenant d'un espace Euclidien. Le son et le langage sont des données unidimensionnelles, les images sont bidimensionnelles ou tridimensionnelles, mais toujours structurées de manière à ce que chaque valeur soit à équidistance de ses voisins. Structurer les données ainsi peut engendrer une perte d'information, puisque cela ne tient aucunement compte des relations entre les éléments. L'apprentissage profond géométrique arrive à pallier ce problème en proposant des architectures capables de traiter des données sous forme de graphe, tels qu'un réseau relationnel ou un nuage de points.

Représentation sous forme de graphes

Pour bien comprendre comment l'apprentissage profond géométrique fonctionne, il est important de connaître comment les graphes sont décrits. La théorie des graphes est une branche des mathématiques et de l'informatique qui étudie les graphes, qui sont des réseaux composés de nœuds (entités) connectés avec des liens (relations). Lorsque les liens sont considérés comme bidirectionnels, on parle d'un graphe non orienté, alors que lorsque les liens peuvent pointer dans une seule direction, on parle d'un graphe orienté. Si le graphe possède un chemin d'un nœud vers ce même nœud, on dit qu'il contient un cycle.

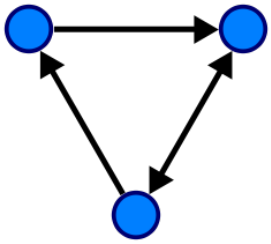


Figure 2.8 Exemple d'un graphe orienté cyclique. Image libre de droits. Wikimedia Commons (2020).

Les nœuds et les liens ont parfois des valeurs qui leur sont associées. Par exemple, dans un contexte de structures de données, un arbre servant à stocker des données triées possède des nœuds qui contiennent des valeurs et des liens orientés qui eux ne possèdent pas de valeur. Dans un contexte de recherche de chemin, les nœuds sont souvent sans valeur autre qu'un identifiant alors que les liens possèdent une valeur représentant un coût.

Un arbre couvrant de poids minimal (ou MST pour "Minimal Spanning Tree") est un concept important dans la théorie des graphes. Le MST d'un graphe désigne un graphe possédant les mêmes nœuds que le graphe de base mais ne gardant qu'un sous-ensemble des liens connectant tous les nœuds au moindre coût.

Il est parfois utile de représenter un graphe sous forme matricielle. Plusieurs représentations sont possibles, telles que les matrices d'incidences, les matrices de degrés, les matrices d'adjacences et les matrices laplaciennes. La matrice d'incidence est de taille n x p où n est le nombre de nœuds et p le nombre de liens. Elle liste les nœuds i à partir desquels les liens j sortent. Dans le cas d'un graphe orienté, un lien sortant est identifié par -1 plutôt que 1. La matrice des degrés est une matrice semi-définie positive diagonale n x n dont les valeurs représentent le nombre de liens qui pointent vers chaque nœud. La matrice d'adjacence est aussi une matrice semi-définie positive de taille n x n. Elle liste l'adjacence des nœuds entre eux, où 1 signifie que le nœud i a un lien vers le nœud j. La diagonale est composée de 0,

mais possède des 1 lorsque des nœuds ont un lien vers eux-mêmes. Dans le cas d'un graphe non orienté, la matrice est symétrique. La matrice laplacienne est définie par la soustraction de la matrice d'adjacence à la matrice des degrés.

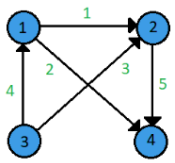
Graphe	Matrice d'incidence	Matrice de degrés	Matrice d'adjacence	Matrice laplacienne
	$\begin{bmatrix} -1 & -1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & -1 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & -1 & 0 & -1 \\ 0 & 2 & 0 & -1 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$

Figure 2.9 Différentes représentations matricielles d'un graphe orienté.

La matrice laplacienne peut aussi être représentée par sa forme normalisée (appelée "weighted" ou "normalized" en anglais), dans tel cas elle sera définie tel que décrite par l'équation (2.2) où d représente la matrice des degrés.

$$\mathcal{L}_{ij}(G) = \begin{cases} 1 & \text{si } i = j \text{ et } d_{ij} \neq 0 \\ -\frac{1}{\sqrt{d_{ii}d_{jj}}} & \text{si les nœuds aux index } i \text{ et } j \text{ sont adjacents} \\ 0 & \text{sinon.} \end{cases} \tag{2.2}$$

Convolutions sur graphes

L'opération de convolution faite par les CNNs utilise un champ réceptif de taille fixe (bien souvent 3x3) où chaque pixel a une position prédéfinie grâce à la structure spatiale en forme de grille qui est présente dans les images. Cela permet de glisser les filtres de caractéristiques de gauche à droite et de haut en bas pour parcourir l'image de manière structurée. Mais si l'on souhaite définir le champ réceptif de l'opération de convolution sur des données non euclidiennes, un certain ordre dans les données se doit d'être imposé. Cette opération se nomme l'étiquetage de graphe ("graph labeling" en anglais) et assigne une valeur à chaque nœud de manière à ce qu'il devienne possible d'ordonner un ensemble de nœuds de manière unique. La méthode d'assignation de valeur à chaque nœud doit faire en sorte que les nœuds de deux sous-graphes similaires soient placés dans le même ordre lorsque traduits de leur représentation graphique à leur représentation vectorielle.

Les réseaux de neurones convolutifs sur graphes (GCN en anglais pour "Graph Convolutional Networks") se divisent en deux classes. Ceux opérant dans le domaine spectral et ceux opérant dans le domaine spatial. La première fonctionne en effectuant d'abord une décomposition

propre de la matrice laplacienne normalisée à l'aide de l'équation (2.3) où L est la matrice laplacienne normalisée, V les vecteurs propres et Λ les valeurs propres.

$$L = V \Lambda V^T \tag{2.3}$$

Pour transférer les poids du réseau et les caractéristiques des nœuds dans le domaine spectral, un simple produit scalaire de ceux-ci avec les valeurs propres est nécessaire. Ensuite, une multiplication élément par élément entre ces deux résultats suivi d'un autre produit scalaire avec les valeurs propres permet de simuler la convolution [35].

$$x *_G y = V((V^T x) \odot (V^T y)) \tag{2.4}$$

Dans cette équation, x représente un signal d'entrée et y représente un filtre de convolution.

Un avantage des convolutions spectrales est qu'elles peuvent détecter facilement des patrons globaux dans le graphe alors que la méthode spatiale a besoin de plusieurs couches successives de convolutions pour avoir une vision plus large. Par contre, les convolutions spectrales sont moins efficaces computationnellement puisqu'elles doivent extraire les valeurs propres des matrices laplaciennes tandis que les convolutions spatiales sont faites directement sur le graphe en propageant l'information des nœuds. De plus, les convolutions spectrales généralisent mal sur des nouveaux graphes car des perturbations apportées au graphe entraînent des changements des espaces propres, alors que les convolutions spatiales effectuent des convolutions localement en réutilisant les poids à différentes positions dans le graphe. Finalement, les convolutions spectrales ne fonctionnent qu'avec les graphes non-orientés, car la matrice aplacienne se doit d'être symétrique pour extraire les vecteurs propres. Les convolutions spatiales ne possèdent pas cette restriction [36].

Une grande quantité de travaux faisant usage des convolutions spatiales se sont démarqués au cours des dernières années, en voici quelques-uns.

Un travail propose Patchy-San, qui consiste en une série d'algorithmes permettant de sélectionner formellement des nœuds et leurs voisinage sur lesquels des convolutions sont à faire. Il traduit les pas de déplacement des filtres (*strides* en anglais) en passant par-dessus certains nœuds dans l'algorithme qui choisit sur quels nœuds faire des convolutions. Le champ réceptif de l'opération de convolution est traduit par un assemblage des plus proches voisins de manière ordonnée afin de produire un vecteur de nœuds de taille fixe sur lequel les filtres de convolutions seront appliqués. Cet ordre dans le vecteur des plus proches voisins est défini par une fonction de distance entre les nœuds qui peut être soit générale, soit adaptée au

domaine du graphe [37].

DCNN (pour “Diffusion-Convolution Neural Network”) utilise les séries entières pour classifier les graphes, les nœuds ou même les liens en les convertissant en des nœuds et en transformant les nœuds en liens. Il propose de diffuser l’information des noeuds à leurs voisins de 3 à 5 fois avant de classifier ceux-ci, mais n’effectue pas de *pooling*, ce qui limite la propagation de l’information à un voisinage relativement restreint pour chaque noeud [38].

Un autre travail notable propose PointNet, une architecture capable de traiter des nuages de points. Il évite de convertir ces nuages de points en grilles de voxels 3D pour résoudre des tâches de classification et segmentation. Le modèle fonctionne en groupant un point et ses plus proches voisins puis effectue une opération de *max-pooling* pour agréger les caractéristiques extraites de ces points afin de produire une classification. Ensuite, il combine les informations locales et globales de chaque point pour permettre une segmentation précise [39].

Un travail récent introduit les convolutions sur les liens plutôt que sur les nœuds (nommées “EdgeConv”) et l’implémente sur l’architecture de PointNet. Les convolutions sur les liens permettent d’assurer une invariance à la translation et une non-localité de l’opération. De plus, le modèle utilise à la fois la position absolue des points ainsi que la position relative des plus proches voisins du point central à la convolution. Une autre fonctionnalité intéressante amenée par ce travail est la reconstruction dynamique du graphe après chaque couche de convolution. Les plus proches voisins sont recalculés à l’aide d’une fonction de distance faite dans l’espace des caractéristiques, ce qui permet au modèle d’apprendre à construire des graphes qui lient les points selon leur représentation sémantique plutôt qu’à partir de leur position spatiale [40].

Un travail lié à l’utilisation de GCN dans le traitement d’images de vasculatures applique les convolutions de graphe sur des images de fond d’œil pour améliorer le résultat de segmentation. Le graphe est construit à partir de points échantillonnés de manière équidistante sur la ligne centrale des vaisseaux générée grâce à une squelettisation. Un lien est créé si les deux points sont directement liés dans le squelette. Chaque nœud est composé d’un vecteur de caractéristiques extrait d’un CNN aux coordonnées du pixel. La représentation cachée de chaque nœud reflète leur probabilité conditionnelle d’appartenir à un segment de vaisseau, sachant la probabilité de ses voisins [41].

Une très récente étude comparative évalue différents modèles pour la tâche d’alignement de graphes. Cette étude définit ce problème comme étant de trouver une matrice d’alignement S où $S(u, v)$ représente la similarité entre u et v, qui sont des nœuds appartenant respectivement aux graphes G_s et G_t. Parmi les méthodes comparées, 3 sont basées sur l’apprentissage des représentations. Bien que chacune ait leur particularité, celles-ci ont toutes une chose en

commun : un procédé en deux étapes incluant la génération de vecteurs de caractéristiques pour chaque nœud des graphes suivi de la génération d'une matrice d'alignement. Ces deux étapes sont faites grâce à des réseaux de neurones distincts, mais aucun d'entre eux ne possède une structure basée sur la propagation de l'information des nœuds par les liens qui les relient [42].

2.7 Approche proposée

Le chapitre suivant présente les méthodes utilisées pour les tâches de synchronisation et mise en correspondance spatiale. En premier, il propose une technique similaire à [17], mais plutôt que d'entraîner un réseau de neurones convolutif de zéro à partir d'une grande quantité de séquences angiographiques, elle exploite le transfert d'apprentissage pour arriver à résoudre la tâche de synchronisation avec une quantité modeste de données. De plus, plutôt que de tenter d'estimer la progression du cycle cardiaque directement pour ensuite effectuer la synchronisation, elle calcule la ressemblance entre les différentes trames des deux séquences pour générer des matrices sur lesquels un algorithme de recherche de chemin permet de trouver la meilleure synchronisation, de manière similaire à [15]. Ensuite, il documente les efforts réalisés pour explorer une méthode d'apprentissage profond sur des graphes pour résoudre la tâche de mise de correspondance spatiale des bifurcations des artères coronaires.

CHAPITRE 3 MÉTHODOLOGIE

Cette section du mémoire présente les détails des travaux effectués en deux étapes. La première décrit la méthode d'apprentissage profond utilisée pour synchroniser des séquences angiographiques sur le cycle cardiaque alors que la deuxième décrit une preuve de concept pour la détection de correspondances spatiales de bifurcations des artères coronaires dans des images multi-vues à l'aide d'apprentissage profond sur graphes.

3.1 Synchronisation des séquences angiographiques

Cette étape du projet vise à trouver des correspondances temporelles sur toute la durée des séquences multi-vues. Un bon alignement sur le cycle cardiaque permet une qualité maximale de reconstruction 3D puisque les artères devraient être positionnées au même endroit en 3D si l'on ignore le mouvement respiratoire.

3.1.1 Préparation des données

Afin de pouvoir entraîner un modèle à synchroniser les différentes séquences angiographiques d'un patient, il est impératif de s'équiper de bons jeux de données. Un premier jeu de données a été créé à partir de 271 séquences angiographiques des artères coronaires gauches et droites en respiration libre provenant de 39 patients pédiatriques distincts. En moyenne, ces séquences sont composées de près de 120 trames acquises pendant une durée de près de 9 cycles cardiaques. Elles ont été acquises à partir d'examens effectués par des cardiologues du CHU Sainte-Justine à l'aide d'un C-Arm de modèle Toshiba Infinix-CFI BP. Cet appareil permet d'acquérir des séquences bi-plans (2 vues perpendiculaires acquises quasi simultanément, en alternance) à 15 trames par seconde pour chaque plan, mais seulement une petite portion des séquences sont obtenues de cette manière et la grande majorité sont acquises en série à 30 trames par seconde dans un seul plan. Les C-Arm génèrent des fichiers DICOM dans lesquels les séquences vidéos sont stockées, ainsi que les lectures de l'électrocardiogramme, si utilisé lors de l'examen. Par la suite, un second jeu de données a été créé, cette fois-ci en combinant les données du jeu précédent avec des données de patients adultes pour vérifier la capacité de généralisation du modèle, avec 234 nouvelles séquences acquises sur 26 patients. En moyenne, ces séquences sont composées de près de 80 trames acquises pendant une durée de près de 7 cycles cardiaques. Ces données ont été générées par un C-Arm de modèle Philips Allura Xper à l'Hôpital du Sacré-Coeur de Montréal, pour une vitesse d'acquisition de 15 trames par

seconde. Pour ce qui est de la séparation des jeux de données en ensembles d'entraînement, de validation et de test, le premier garde 39 séquences provenenant de 5 patients pour créer l'ensemble de validation et 30 séquences provenant de 5 autres patients pour créer l'ensemble de test. Le deuxième jeu de données garde 40 séquences de 5 patients pédiatriques et 45 séquences de 5 patients adultes pour créer l'ensemble de validation ainsi que 47 séquences de 6 patients pédiatriques et 31 séquences de 5 patients adultes pour créer l'ensemble de test.

Séquences d'entraînement

Bien que les images brutes des séquences soient utilisées directement pour l'entraînement du modèle, ce ne sont pas toutes les images qui sont utilisées. Seules les images ayant un contraste plus élevé que la moyenne de la séquence sont gardées. Ce choix est explicable par le fait que les artères ne sont visibles que pour environ la moitié des trames de la séquence, lorsque l'agent de contraste circule dans celles-ci. En général, les séquences commencent avant que l'agent de contraste soit injecté dans l'arbre coronarien et terminent après qu'il soit évacué. La figure 3.1 montre la courbe du gradient moyen global dont ses valeurs sont utilisées pour choisir les trames à garder.

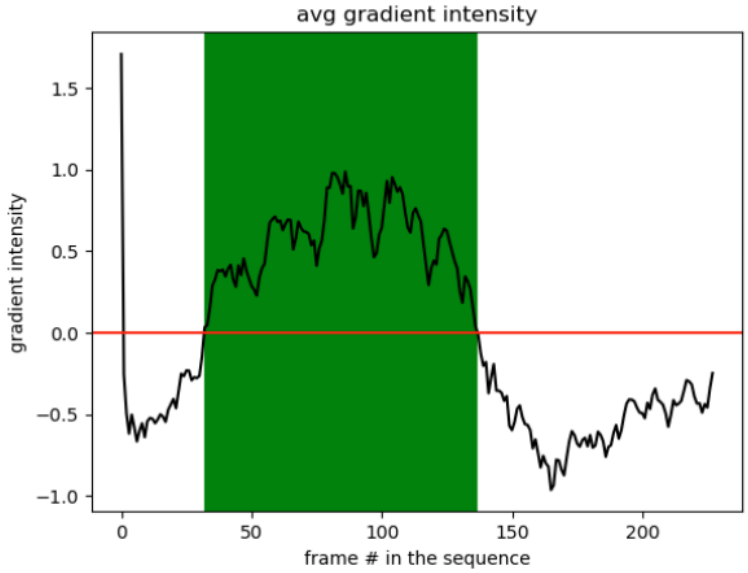


Figure 3.1 Sélection des images pertinentes d'une séquence selon leur contraste global. Pour chaque image, la valeur moyenne du gradient est calculée. On leur soustrait ensuite la valeur moyenne du gradient calculée sur toute la séquence, puis les images dont le gradient est au-dessus de la moyenne (section en vert où les valeurs sont supérieures à la ligne rouge) sont gardées pour l'entraînement.

Signaux ECG

Afin de donner une direction à l'apprentissage du réseau de neurones pour la tâche de synchronisation de manière supervisée, il est essentiel de préparer des données de vérité terrain (communément appelées données *ground truth*). Dans ce cas-ci, l'analyse des électrocardiogrammes permet de générer ce genre de données, car il est relativement simple d'identifier un moment clé du cycle cardiaque grâce à ceux-ci. Les ondes R sont les plus faciles à identifier, car elles forment un pic d'intensité élevée. Pour détecter précisément leur position dans la séquence, il est possible d'utiliser une librairie python telle que *BioSPPy*, mais pour des raisons de difficultés techniques, une méthode faite à la main est proposée. Celle-ci procède en plusieurs étapes et les valeurs énoncées dans l'algorithme ont été choisies de manière empirique.

Premièrement, on lisse la courbe à l'aide d'une moyenne mobile fine d'environ un cinquième de seconde pour retirer le bruit, on soustrait une moyenne mobile grossière d'environ quatre cinquièmes de seconde pour égaliser la courbe au long de la séquence, puis on normalise les valeurs. Un exemple est affiché à la figure 3.2 où la courbe brute est présentée en A et où B montre le résultat de la première étape. Cette étape permet d'avoir des courbes similaires d'une lecture à l'autre. Deuxièmement, on trouve les pics potentiels en identifiant les maximums locaux sur les valeurs normalisées, tel que présenté en B. Cela trouve une grande quantité de pics potentiels, qu'on doit filtrer à la dernière étape. Troisièmement, on identifie les minimums et maximums locaux sur le gradient des valeurs normalisées, tel que présenté en C. Ceux-ci servent à filtrer les pics potentiels à la prochaine étape. Finalement, les pics potentiels sont filtrés de 3 manières différentes successivement. Les maximums et minimums trouvés sur le gradient sont utilisés pour filtrer tout pic potentiel n'étant pas positionné après un pic de gradient positif et suivi d'un pic de gradient négatif. Les pics potentiels restants sont ensuite filtrés par leur largeur pour rejeter les pics trop larges (de largeur supérieure à un cinquième de seconde), puis les pics étant trop collés les uns sur les autres sont filtrés pour ne garder que le plus grand pic. Les pics restants sont ceux vis-à-vis des ondes R, tel que présenté en D.

Par la suite, une validation est faite sur les ondes R identifiées en mesurant l'écart-type de distance entre les pics pour s'assurer qu'il n'y ait pas de détection manquante. Si ce n'est pas le cas, la séquence est rejetée puisqu'elle ne pourrait pas être utilisée pour entraîner ou évaluer le modèle.

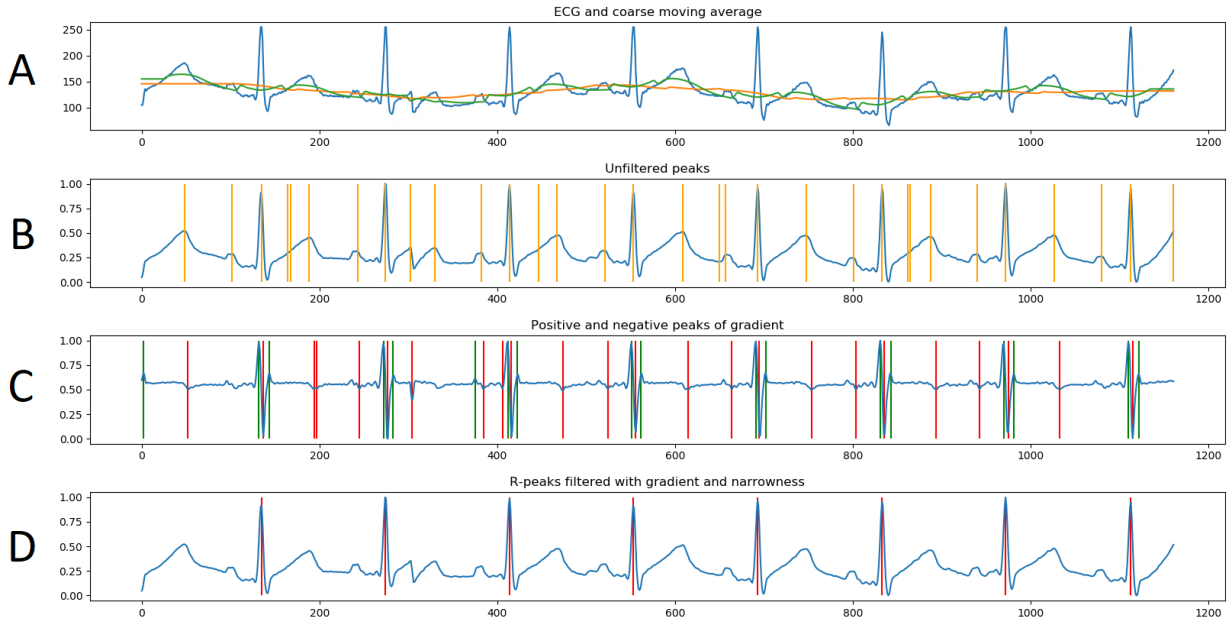


Figure 3.2 Étapes pour l'identification automatique des ondes R. **A** : valeurs brutes en bleu, moyenne mobile grossière en jaune, moyenne mobile fine en vert. **B** : valeurs normalisées et lissées en bleu, pics potentiels en jaune. **C** : gradient des valeurs normalisées en bleu, maximums locaux en vert et minimums locaux en rouge. **D** : valeurs normalisées en bleu, pics filtrés en rouge.

Données *ground truth* à partir des ECG

Les ondes R extraites des électrocardiogrammes sont utilisées pour générer des matrices de données *ground truth* (un exemple est présenté à la figure 3.4) où chaque élément (t, t') représente le niveau de synchronisation variant entre 0 et 1 pour la paire d'images t de la vidéo A et t' de la vidéo B. Un niveau de synchronisation de 1 signifie que les images t et t' sont positionnées exactement au même endroit dans leur cycle cardiaque respectif et 0 signifie qu'il y a un demi cycle cardiaque de distance entre leur phase cardiaque. La phase cardiaque représente la progression du cycle cardiaque et est calculée en utilisant les ondes R extraites des signaux des ECG. Plus précisément, elle est la fraction calculée selon la position de t entre les ondes R précédente (R_0) et suivante (R_1).

$$Phase = \frac{t - R_0}{R_1 - R_0} \tag{3.1}$$

Cette façon de créer la matrice de données *ground truth* à partir des signaux des ECG des deux vidéos assure une bonne correspondance temporelle à la fin de la phase diastolique, même lorsque la différence entre le rythme cardiaque de celles-ci est grande.

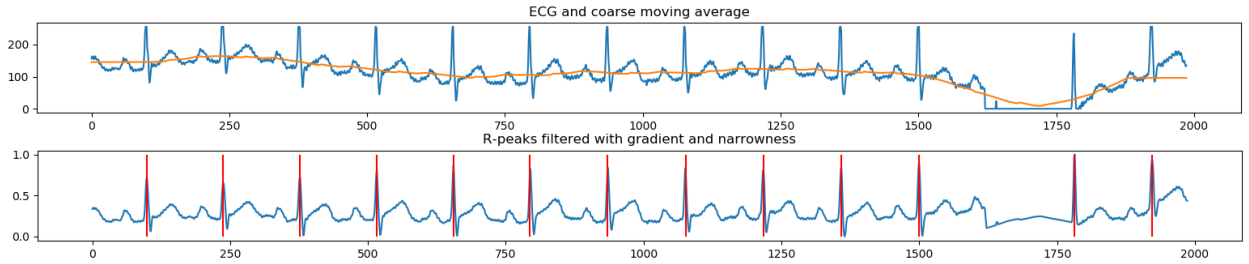


Figure 3.3 Identification automatique des ondes R avec exemple de rejet dû à un pic manquant.

3.1.2 Choix du modèle

Puisque le modèle doit synchroniser les séquences à partir des images, les réseaux de neurones convolutifs sont bien placés pour réaliser la tâche. Ceux-ci peuvent prendre des images en entrée et créer des vecteurs de caractéristiques en sortie, qui peuvent être utilisés pour déterminer un degré de synchronisation entre les images. Puisque la quantité de séquences d'entraînement disponible est relativement faible, démarrer l'entraînement à partir d'un modèle pré-entraîné est préférable. Dans ce travail, plusieurs modèles existants ont été testés, dont les notables ResNet-50, MobileNet-v2 et EfficientNet. Ces modèles ont été choisis grâce à leur performance sur ImageNet, le populaire défi de reconnaissance d'images.

Par contre, ces modèles sont faits pour traiter des images couleurs à 3 canaux alors que les images des séquences angiographiques sont en noir et blanc et donc ne possèdent qu'un seul canal. Pour remédier à ce problème, il est possible d'empiler 3 images consécutives de la séquence pour créer une image à 3 canaux (donc traitable par ces modèles pré-entraînés) et cela ajoute même une dimension temporelle aux images d'entrées. Cela permet au réseau de détecter et d'utiliser le mouvement des artères afin de donner en sortie un vecteur de caractéristiques qui tient compte de l'aspect temporel.

3.1.3 Génération des matrices de similarité

Pour chaque triplet d'images consécutives passé en entrée lors de l'inférence du modèle, un vecteur de caractéristiques décrivant leur position dans le cycle cardiaque est généré. Lorsque cela est fait pour deux séquences angiographiques, on obtient deux listes de vecteurs A et B. Les vecteurs de ces listes peuvent être comparés les uns aux autres afin de déterminer leur ressemblance. Pour se faire, la méthode proposée est une composition de la similarité cosinus et de la distance L1. Cette composition est définie par l'équation (3.2) où N est la taille des vecteurs de caractéristiques.

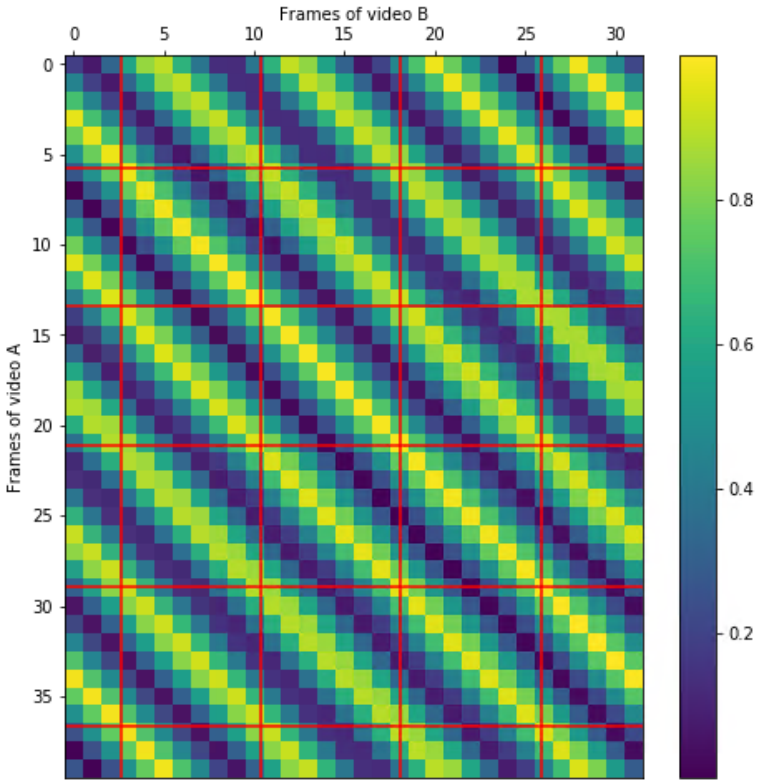


Figure 3.4 Exemple de matrice de données *ground truth*. Chaque élément de la matrice représente le niveau de synchronisation entre l'image du vidéo A et celle du vidéo B. Les lignes rouges représentent les ondes R extraites des ECG.

$$C_{i,j} = \left(2 - \frac{A_i \cdot B_j}{\|A_i\|\|B_j\|}\right) \sum_{k=0}^{N} |A_{i_k} - B_{j_k}| \tag{3.2}$$

Selon les expériences effectuées, la combinaison des matrices de distance et de similarité permet de bénéficier des avantages qu'offrent les deux tout en évitant leurs inconvénients respectifs. D'après les résultats obtenus en cours de développement, la matrice de distance est habituellement plus précise mais comporte parfois des aberrations, alors que la matrice de similarité est légèrement moins précise mais plus constante. En les combinant, on obtient une fonction de distance à la fois précise et robuste. Cette fonction permet donc de créer une matrice exposant la différence entre chaque image des deux séquences. Dans cette matrice, on peut voir des lignes diagonales qui représentent les meilleures synchronisations.

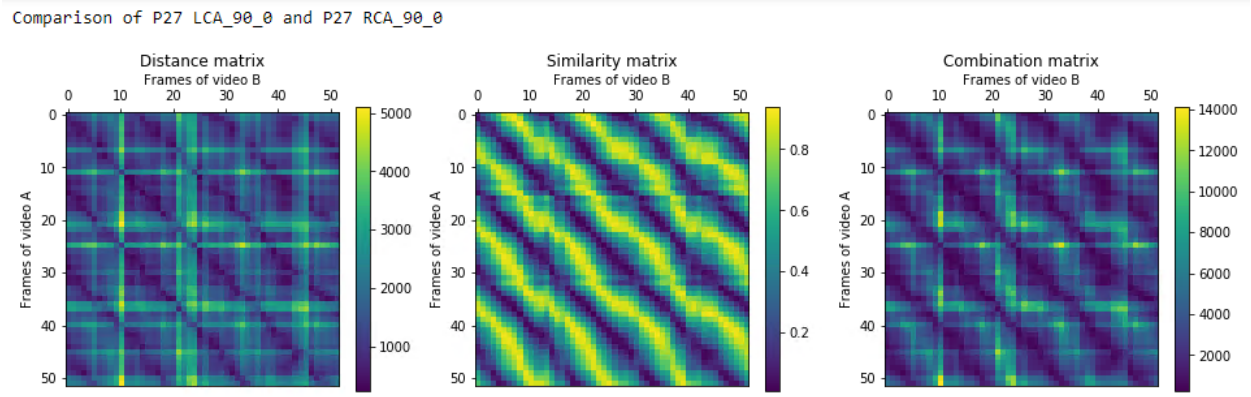


Figure 3.5 Exemples de matrices de distance, de similarité et une composition des deux. Les lignes diagonales, où la distance est la plus faible, représentent les paires d'images décrivant un moment du cycle cardiaque similaire.

3.1.4 Recherche de chemin

Une fois la matrice de distance générée, il ne reste qu'à identifier les lignes diagonales pour trouver les possibles synchronisations sur le cycle cardiaque. Pour se faire, l'utilisation de l'algorithme de Dijkstra pour la recherche de chemin est proposée. Cet algorithme permet d'explorer un espace grâce à une recherche en largeur plutôt qu'en profondeur, en priorisant les chemins dont le coût est le plus bas. Cela permet de réduire considérablement le nombre d'états à explorer en comparaison avec un algorithme de force brute qui explorerait tous les états possibles. Les valeurs utilisées par l'algorithme décrit dans le prochain paragraphe ont été trouvées de manière empirique.

On commence par démarrer une recherche de chemin sur la première ligne et la première colonne en espaçant chaque essai par quelques éléments. Les actions disponibles sont d'avancer d'un élément verticalement, horizontalement ou diagonalement, sans retour en arrière. Une limite de quelques actions consécutives verticales et horizontales est imposée afin d'éviter que l'algorithme ne fasse progresser qu'un seul des vidéos pour trop de trames consécutives dans le cas où la matrice présente des aberrations. Les recherches de chemins se terminent lorsque la dernière colonne ou la dernière ligne de la matrice est atteinte. Une fois toutes les recherches de chemin complétées, on en démarre une nouvelle à chaque point de fin des recherches de l'étape précédente. Ces dernières ont les mêmes actions disponibles, mais à sens contraire. On compare ensuite le nombre de points de fin distincts à la fin de cette deuxième étape avec le nombre de recherches de chemin effectuées. Tant que ces valeurs sont différentes, on répète cette dernière étape dans la direction opposée. Le résultat final est une collection de chemins qui passent par les lignes diagonales aperçues précédemment dans les matrices

de distances.

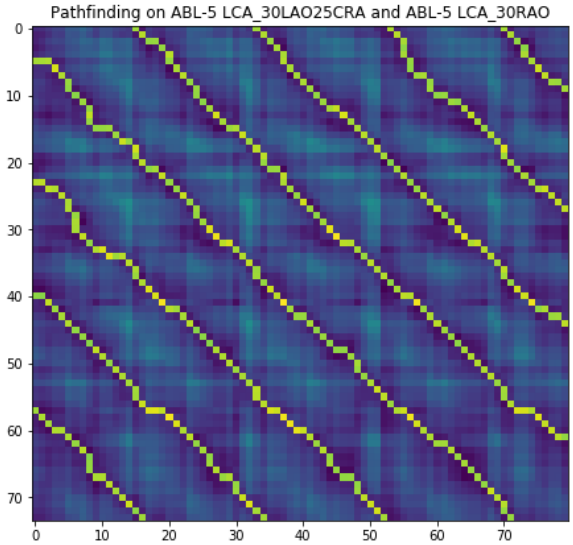


Figure 3.6 Exemple du résultat final de la recherche de chemin. Les points de chaque chemin sont représentés par l'ajout de grandes valeurs dans la matrice afin de les faire contraster.

Par la suite, on doit tenter de trouver le "meilleur" chemin pour ne garder que la meilleure synchronisation. Parmi les plus longs chemins, c'est-à-dire ceux dont la longueur est supérieure ou égale à 90% de la longueur du plus long chemin, celui dont la rectitude est la plus élevée est désigné comme étant le meilleur chemin. La rectitude d'un chemin est calculée en divisant la distance entre les points de départ et de fin par la longueur du chemin. Le meilleur chemin est ensuite utilisé pour définir quelles images des deux séquences sont associées pour avoir la meilleure synchronisation possible.

3.1.5 Stratégie d'entraînement

Même si le modèle choisi est pré-entraîné sur des millions d'images provenant d'ImageNet, il ne pourra pas générer des matrices de distances précises puisqu'il n'a pas été entraîné sur les séquences d'images angiographiques. Pour se faire, on doit recourir à une phase de *fine-tuning* dans laquelle une fonction de perte alimentée par les données *ground truth* permet au modèle d'ajuster ses poids pour mieux répondre à la tâche de synchronisation.

Fonction de perte

Pendant l'entraînement, pour chaque *mini-batch* M, N images sont échantillonnées dans une vidéo choisie au hasard pour créer un vecteur d'images, où N est la taille de la *mini-batch*. Il

est à noter que les images échantillonnées peuvent aussi provenir de deux vidéos plutôt que d'un seul et que les deux méthodes ont été évaluées lors de la recherche d'hyperparamètres. Pour chaque image échantillonnée, l'image d'entrée du réseau à 3 canaux est créée en superposant l'image courante et les deux images précédentes dans le vidéo. Étant donné i et j, deux images d'entrées échantillonnées de la *mini-batch*, deux vecteurs de caractéristiques sont générés à partir de l'inférence du réseau de neurones siamois. La méthode de similarité cosinus est utilisée pour calculer la similarité entre ces deux vecteurs de caractéristiques. La fonction de perte utilisée pour le *fine-tuning* du réseau de neurones est définie par la moyenne des distances au carré entre la similarité cosinus de chaque paire de vecteur de caractéristiques avec sa valeur respective dans la matrice de données *ground truth* y. Pour l'équation (3.4), la notation M_i et M_j est utilisée plutôt que A_i et B_j puisqu'il s'agit d'opérations faites sur des images de la même *mini-batch* plutôt que sur des images provenant de séquences différentes.

$$loss = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{(0.5(cos(\theta)_{i,j} + 1) - y_{i,j})^2}{N^2} \tag{3.3}$$

$$cos(\theta)_{i,j} = \frac{M_i \cdot M_j}{\|M_i\| \|M_j\|} \tag{3.4}$$

Les fonctions de perte habituellement utilisées par les réseaux de neurones siamois, telles que celles par paires ou triplets, génèrent un nombre limité de combinaisons valides par *mini-batch* car chaque exemple d'entraînement nécessite un exemple positif et un exemple négatif. La fonction de perte proposée, appelée "soft pair loss" ou "paires souples" en français, génère un signal d'apprentissage plus riche à partir des vecteurs de caractéristiques échantillonnés puisqu'ils peuvent tous être appariés entre eux.

3.2 Détection de correspondances spatiales

La mise en correspondance temporelle des images multi-vues sur la phase cardiaque n'est que la première étape de la reconstruction 3D. La mise en correspondance spatiale des segments d'artères est encore plus importante, et surtout, plus complexe. La superposition des structures vasculaires fait en sorte que la contrainte épipolaire n'est pas suffisante pour associer les segments d'artères d'une vue à l'autre. De plus, les contraintes basées sur l'intensité des pixels ne peuvent pas être utilisées puisque les segments d'artères dans les images angiographiques ne possèdent pas de caractéristique suffisamment discriminatoire pour les distinguer. Même pour un humain, il est souvent assez difficile de reconnaître chaque segment d'artère

d'une vue à l'autre. Pour s'aider, plutôt que de simplement tenter d'identifier des similarités dans la forme des artères, il est possible de parcourir l'arbre coronarien à partir des ostiums (identifiés dans la figure 3.7). Cela nous permet de placer chaque segment d'artère par rapport à son parent dans l'arbre coronaire, ce qui aide à détecter les correspondances spatiales. Cette section présente le travail effectué pour répliquer cette logique à l'aide d'algorithmes, en représentant l'arborescence des segments d'artères à l'aide de graphes extraits de façon ad hoc. Par la suite, une preuve de concept décrivant une technique d'apprentissage par réseau de neurones profond sur graphe est présentée, dans le but de reconnaître des segments d'artères dans différentes vues en identifiant des correspondances spatiales entre les bifurcations et les points de terminaison des artères.

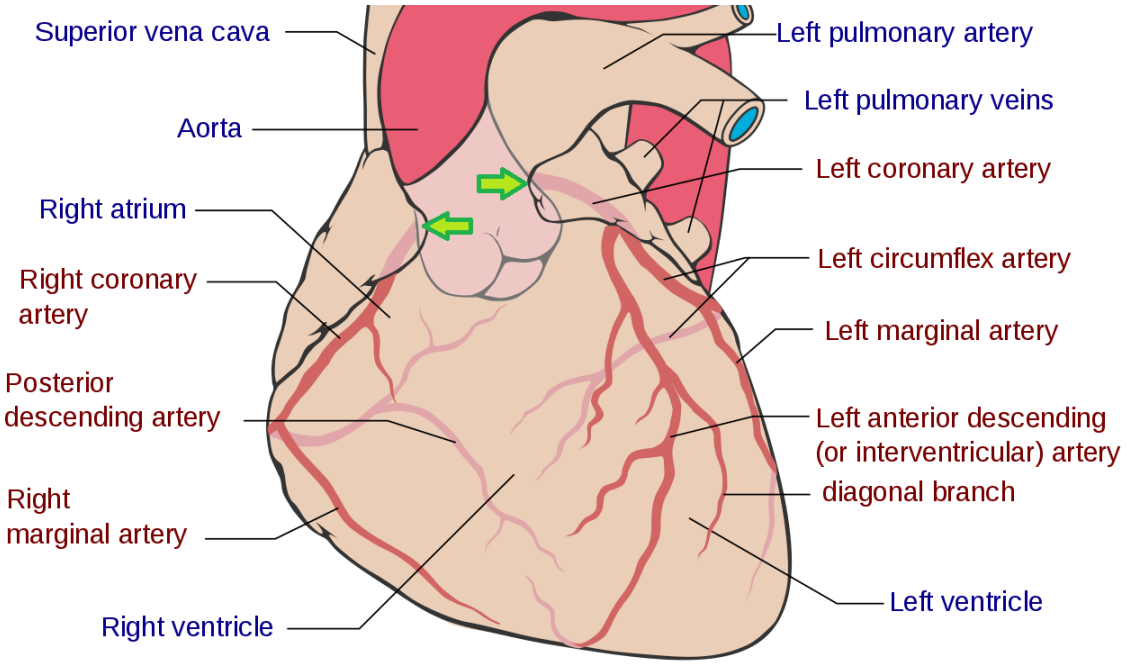


Figure 3.7 Schéma des artères coronaires. Les ostiums sont identifiés par les flèches vertes. Image libre de droits modifiée. Wikipédia (2020).

3.2.1 Extraction de graphe

Pour extraire un graphe de l'arbre coronarien à partir d'une image angiographique, plusieurs étapes sont nécessaires. La première est l'extraction des vaisseaux sanguins grâce à la segmentation. Habituellement faite grâce à des réseaux de neurones convolutifs, cette tâche se doit d'être très précise et robuste afin d'éviter d'identifier de faux segments ou de générer des trous sur de vrais segments. L'image en sortie est une carte binaire où les pixels allumés représentent les segments de vaisseaux. Ensuite, la squelettisation des vaisseaux segmentés

permet d'identifier la ligne centrale de chaque vaisseau. Cette tâche est habituellement faite à l'aide d'un algorithme qui effectue un retrait itératif des pixels extérieurs d'une image segmentée, jusqu'à ce qu'il ne reste que des lignes d'un seul pixel d'épaisseur. À partir des squelettes, il devient possible de détecter les bifurcations et les croisements des segments d'artères. Par contre, cette étape est très difficile, puisque les algorithmes de squelettisation ne sont pas adaptés pour extraire correctement la ligne centrale de segments qui se croisent. De plus, la superposition des vaisseaux complexifie davantage la tâche d'identification des croisements d'artères. Finalement, pour extraire le graphe de manière orientée, il est nécessaire d'identifier un point de départ, qui est l'ostium dans ce cas-ci. Celui-ci est détecté à partir du cathéter pour ensuite pouvoir parcourir l'arbre coronarien de manière orientée. Lors du parcours de l'arbre, les vraies bifurcations sont identifiées pour créer un graphe à partir duquel un réseau de neurones par graphe peut être entraîné pour caractériser les bifurcations. Tel que mentionné précédemment, les algorithmes décrits dans cette section sont développés de façon ad hoc puisque l'extraction de graphe est une étape intermédiaire à la détection de correspondances spatiales et ne constitue pas l'objectif de ce projet.

Segmentation

Bien que peu d'efforts ont été consacrés à l'obtention de bonnes cartes de segmentations dans le contexte de ce projet, il est tout de même intéressant de se renseigner sur cette tâche. De nos jours, toute tâche de segmentation est faite à partir de réseaux de neurones convolutifs. La structure de ces réseaux est habituellement basée sur le U-Net [43]. L'augmentation de données est cruciale pour générer une segmentation robuste, particulièrement lorsque la quantité de données disponible est plutôt limitée. Par exemple, les auteurs du U-Net présentent des transformations affines telles que la translation et la rotation, mais aussi d'autres types de transformations, telles que la variation d'intensité des pixels ou la déformation élastique aléatoire qui, selon eux, est un concept clé pour permettre au modèle de bien généraliser avec peu de données. Pour la tâche spécifique de segmentation de vaisseaux, le prétraitement des données peut aider à améliorer les performances du modèle. Par exemple, plutôt que de simplement alimenter le réseau avec une image à 3 canaux, on peut ajouter 3 autres canaux contenant les valeurs RGB normalisées pour corriger les disproportions d'illumination [44].

Sur une autre note, certains travaux s'appuient sur l'exploitation d'information tirée d'images consécutives provenant de séquences vidéos. Un premier améliore la qualité de segmentation en faisant du réglage fin des poids sur une image segmentée du vidéo avant de segmenter les suivantes [45]. Un autre arrive à apprendre à faire la segmentation (entre autre) de manière auto-supervisée en forçant le modèle à reconstruire les images complètes de vidéos [46].

Un article récent propose de traiter le problème de segmentation d'artères coronaires comme un problème de décomposition de matrice structurée de rang inférieur pour distinguer l'avant-plan de l'arrière-plan grâce au mouvement des vaisseaux [47].

Mais puisque l'étape de segmentation ne fait pas directement partie des objectifs de recherche, la solution la plus simple a été choisie. Celle-ci se trouve à être l'utilisation de segmentations générées automatiquement puis corrigées à la main pour en retirer les défauts. Ces segmentations proviennent du travail d'un ancien étudiant du laboratoire qui les a générées grâce à un module d'extraction de ligne curviligne gaussienne anisotrope récursif et orientable, après avoir amélioré la qualité photométrique des séquences angiographiques à l'aide d'une combinaison d'étapes de transformation en ondelettes complexes à double arbre et de filtrage homomorphe [48].

Mais avant de procéder à la squelettisation des vaisseaux, il est important de peaufiner la carte de segmentation. Si certains pixels sont manquants, cela peut causer des artefacts dans le squelette sous forme de branches faisant le tour des pixels manquants, tel que présenté dans la figure 3.8. Pour remplir ces pixels manquants, une image temporaire est créée en copiant la carte de segmentation originale et en remplaçant les pixels dont l'intensité est égale à 0 par des -1. Ensuite, un filtre 3x3 composé d'éléments à valeur 1 sur les côtés et avec -1 au centre est convolué sur l'image. Lorsque la réponse est maximale (égale à 9), la position du pixel au centre du filtre est utilisée pour allumer le pixel dans la segmentation originale.

Figure 3.8 Exemple d'artefacts sous forme de fausses branches apparaissant dans le squelette en raison de pixels manquants dans la segmentation. La troisième image montre le résultat obtenu lorsqu'il n'y a plus de pixels manquants.

Squelettisation

L'opération de squelettisation, qui sert à extraire les lignes centrales d'une image binaire, est utilisée pour une multitude d'applications liées au traitement d'images et vision par ordinateur, comprenant la reconnaissance de forme et son analyse, la décomposition de forme, la reconnaissance de caractères, l'analyse d'empreintes digitales, l'animation, le suivi de mouvement, le recalage, l'interpolation, le suivi de chemins, le traitement d'images médicales et bien plus [49]. Dans ce domaine de recherche, deux travaux se sont considérablement démarqués. Le premier introduit un algorithme de squelettisation parallélisable qui fonctionne en retirant les pixels du contour de l'image de manière itérative jusqu'à ce qu'il n'en reste qu'un seul d'épaisseur, en prenant deux fois moins de temps que ses prédécesseurs [50]. Le deuxième propose lui aussi un algorithme de squelettisation parallélisable mais celui-ci utilise un *octree* de taille 3x3x3 pour examiner le voisinage 3D de chaque pixel itérativement pour déterminer si ceux-ci peuvent être retirés, jusqu'à ce qu'une itération ne puisse retirer aucun pixel. Une opération d'élagage est ensuite exécutée en post-traitement, où les branches de taille inférieure à une longueur prédéterminée sont retirées afin de corriger les erreurs introduites par le bruit [51].

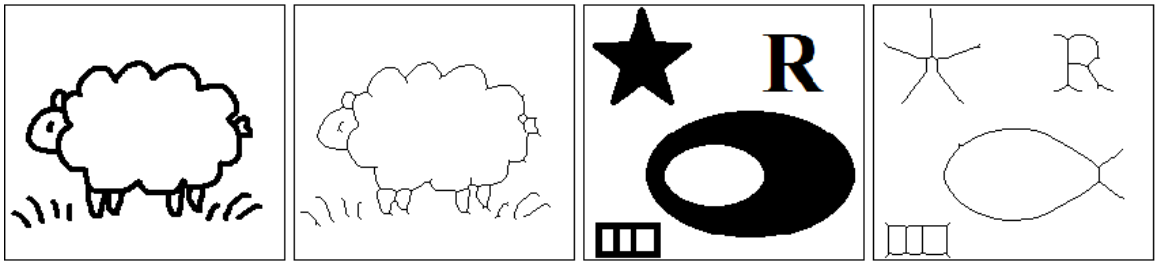


Figure 3.9 Exemples de squelettisation. Image libre de droits. Par Robotyczna Owca.

Une tâche particulièrement difficile liée à l'analyse de squelettes est la détection de croisements. La plupart du temps, à l'endroit du croisement, plutôt que de créer un pixel central avec 4 pixels connectés aux coins ou d'avoir 4 pixels centraux connectés par leurs coins, l'opération de squelettisation crée deux bifurcations séparées par un segment au centre du croisement.

Certains chercheurs se sont donc penchés sur le problème pour trouver une solution. Un travail introduit un autre algorithme de squelettisation afin de détecter les croisements de segments de courbes avant de procéder à l'extraction des lignes centrales en calculant l'épaisseur verticale et horizontale de ceux-ci. Il se trouve que l'épaisseur verticale ou horizontale des segments qui se croisent augmente soudainement lorsqu'ils commencent et terminent le croisement. Un algorithme est exécuté pour connecter les différents segments qui se croisent

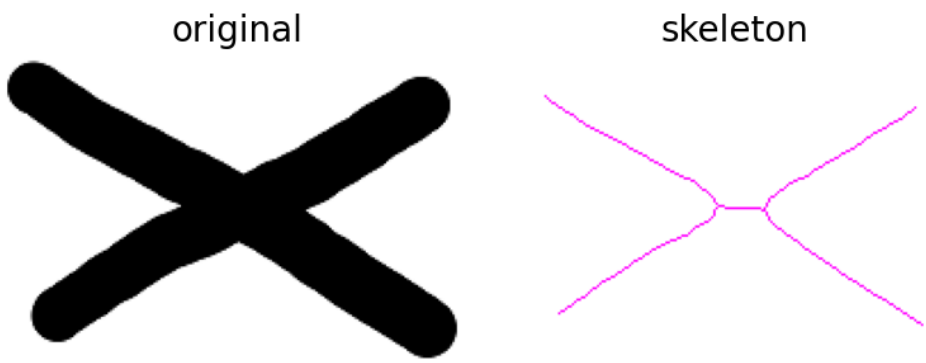


Figure 3.10 Illustration du problème de croisement de lignes lors de la squelettisation.

lors de l'extraction des lignes centrales [52]. Un autre travail propose de modifier les points de bifurcations durant la phase de post-traitement en plaçant une fenêtre de taille fixe centrée sur chaque point de bifurcation et de compter le nombre de segments qui atteignent le cadre de la fenêtre. Lorsque 3 segments sont comptés, aucune modification n'est faite. Lorsque seulement 2 segments se rendent aux bords de la fenêtre, le troisième est considéré comme une branche indésirable et est donc retiré, ce qui fait disparaître le point de bifurcation. Finalement, lorsque 4 segments sont répertoriés, les deux bifurcations sont fusionnées pour n'avoir que 2 segments qui se croisent [53]. Par contre, ces algorithmes ne permettent pas de gérer des croisements multiples ou de longs croisements comme ceux qu'on retrouve dans des images d'angiograhie coronaire.

Pour ce projet, le squelette est généré grâce à la méthode de Lee (Lee et al. (1994)) implémentée dans la librairie python "scikit-image", car celle-ci calcule la largeur des vaisseaux lors de la phase de squelettisation. Une fois le squelette obtenu, il est pertinent de retirer certaines branches indésirables. Les algorithmes de squelettisation fonctionnent de manière à ce que lorsqu'une protubérance d'un vaisseau est visible dans la carte de segmentation, une branche est créée au centre de celle-ci. Mais puisque ces petites branches ne sont pas des vrais segments de vaisseaux, elles peuvent être considérées comme des artefacts dans le graphe. Un algorithme d'élagage est donc nécessaire pour obtenir un squelette propre.

La méthode proposée est de considérer la largeur et la longueur de chaque branche se terminant en cul-de-sac pour décider si elles devraient être retirées ou non, en procédant itérativement. La formule (3.5) est utilisée pour choisir quelles branches sont à couper.

$$\frac{largeur}{longueur} > \alpha + \beta \cdot longueur \tag{3.5}$$

α et β sont des hyperparamètres servant à ajuster le seuil. α permet de choisir la valeur de base du seuil et β augmente le seuil proportionnellement à la longueur de la branche afin que les branches plus longues aient besoin d'avoir un ratio largeur/longueur plus élevé pour être coupées. Une fois toutes les branches analysées, tant qu'il y a au moins une d'entre elles qui a été coupée, on continue à itérer en analysant à nouveau toutes les branches se terminant en cul-de-sac.

Détection des bifurcations et croisements

Une fois le squelette élagué, l'étape suivante pour extraire le graphe est de détecter les bifurcations. Celles-ci sont définies par le pixel à partir duquel trois branches émergent. Par contre, ces bifurcations ne sont pas toutes de vraies bifurcations de vaisseaux puisque l'algorithme de squelettisation transforme les croisements de vaisseaux en deux bifurcations. Si l'on veut extraire un graphe qui ne comporte que de vraies bifurcations, il devient nécessaire de détecter ces fausses bifurcations pour les considérer comme des croisements. La méthode proposée est de parcourir les segments du squelette et de vérifier si chaque bifurcation pourrait être un croisement en comparant l'angle et la largeur du segment avec celles des autres segments du squelette dans le voisinage de la bifurcation.

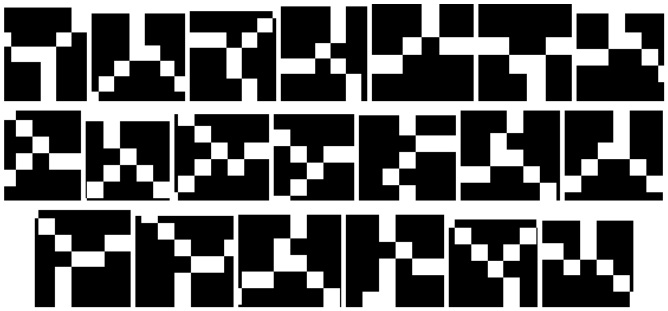


Figure 3.11 Exemples de points de bifurcations d'un squelette.

L'algorithme démarre à partir d'un point donné du squelette puis suit les pixels jusqu'à arriver à une bifurcation. Il calcule ensuite un vecteur de direction basé sur les pixels précédant le point de bifurcation. Les X pixels les plus près de la bifurcation, où X est la largeur au point de bifurcation, sont pondérés à la baisse puisqu'ils changent parfois de direction par rapport au reste du vaisseau. La largeur du segment est aussi calculée à partir des pixels précédant le point de bifurcation.

Une fois la direction et la largeur du segment calculées, l'algorithme poursuit l'exploration du squelette en priorisant les branches les plus proches et dans la même direction. Chaque branche issue d'une bifurcation se trouvant dans le voisinage de la bifurcation initiale est

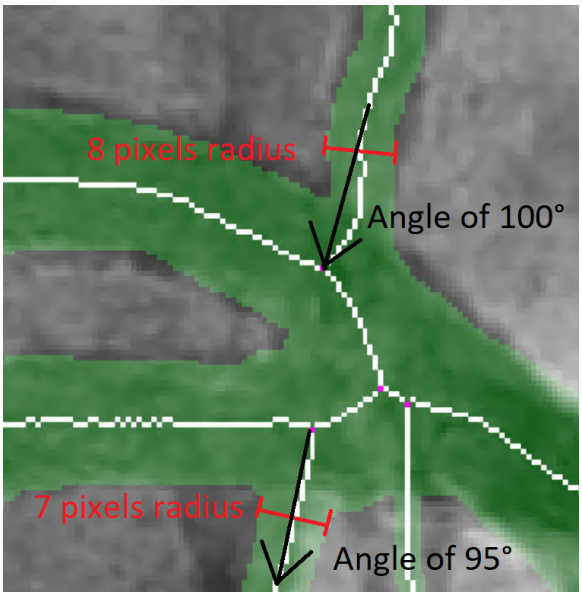


Figure 3.12 Exemple de détection de croisement de vaisseaux. Ici, on voit un cathéter superposé à un vaisseau qui présente une vraie bifurcation.

explorée afin de trouver toutes les branches qui pourraient être une continuité. Une bifurcation est considérée comme étant dans le voisinage si sa distance avec la bifurcation initiale est inférieure à un maximum prédéfini et si l'angle entre la direction du segment initial et la ligne reliant les deux bifurcations est inférieur au seuil défini par l'équation (3.6).

$$seuil = \alpha - \tau(\alpha - \beta) \tag{3.6}$$

Dans cette équation, α représente l'angle maximal permis, β représente l'angle minimal et τ représente la fraction de la distance de la bifurcation sur la distance maximale. Ce seuil fait en sorte que l'angle peut être large pour une bifurcation à proximité mais plus la bifurcation est éloignée, plus le seuil d'angle diminue, jusqu'à atteindre l'angle minimal lorsque la distance est égale au maximum.

Par la suite, la direction et la largeur de chaque branche explorée sont calculées et comparées à celles du segment initial. Pour considérer une branche comme étant une continuité probable du segment initial, la différence d'angle entre les deux directions et le ratio de la différence de largeur des vaisseaux doivent être inférieurs à des valeurs prédéfinies.

Lorsque toutes les branches du voisinage ont été explorées, l'équation (3.7) définissant une formule de score est utilisée pour déterminer laquelle est la plus probable d'être la vraie continuité.

$$score = 1 - \frac{\theta}{\alpha} + max\left(0, 1 - \frac{\tau}{\beta}\right) + 1 - \frac{\delta}{\gamma} \tag{3.7}$$

Dans cette équation, θ est l'angle, α est l'angle maximal permis, τ est le ratio de largeur, β est le ratio de largeur maximal permis, δ est la distance et γ est la distance maximale permise. La branche avec le score le plus élevé est choisie comme étant la continuité du segment initial et l'algorithme continue d'explorer le squelette à partir de celle-ci pour trouver de nouveaux croisements.

Détection de l'ostium à partir du cathéter

Afin de pouvoir extraire un graphe orienté à partir du squelette, il est primordial d'identifier la racine du graphe. Celle-ci est logiquement définie par le point d'entrée de l'artère coronaire, c'est-à-dire l'ostium. La méthode proposée pour trouver l'ostium à partir du squelette est de suivre le cathéter jusqu'à ce qu'il aboutisse à l'ostium. La première étape est de détecter les segments du squelette qui appartiennent au cathéter. Pour se faire, la méthode la plus simple est d'identifier les segments qui atteignent ou presque une extrémité de l'image. Dans le jeu de données de segmentations utilisées pour générer les squelettes, le cathéter passe près du haut de l'image pour chaque segmentation. Si plus de deux segments sont détectés de cette manière, seuls les deux ayant l'intensité la plus faible (gris plus foncé) sont gardés.

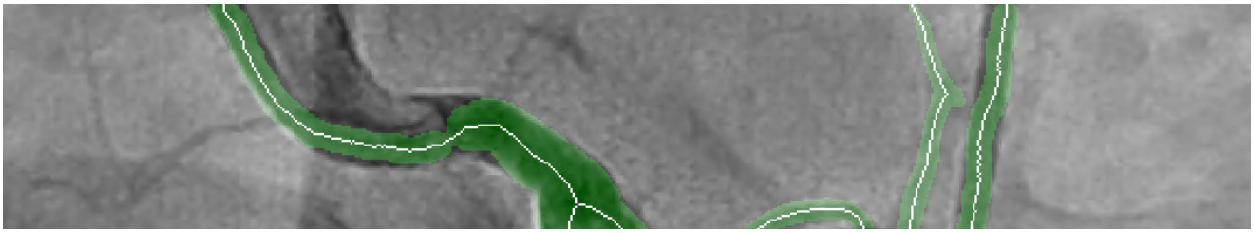


Figure 3.13 Exemple de segments du squelette qui atteignent le haut de l'image.

Ensuite, pour chaque segment de cathéter, on suit les pixels de ceux-ci en vérifiant constamment certaines informations. Lorsqu'un cul-de-sac est détecté dans le bas de l'image, cela signifie que ce segment de cathéter est celui qui monte vers l'aorte. C'est plutôt l'autre qui est utile, celui qui descend de l'aorte vers l'ostium. Lorsque l'on suit les pixels de celui-ci, plusieurs indicateurs peuvent aider à détecter la position de l'ostium. Un changement brusque de direction, un élargissement brusque du segment, un cul-de-sac ou une vraie bifurcation peuvent tous révéler la position de l'ostium.

Pour détecter un changement brusque de direction, il est possible de calculer une progression

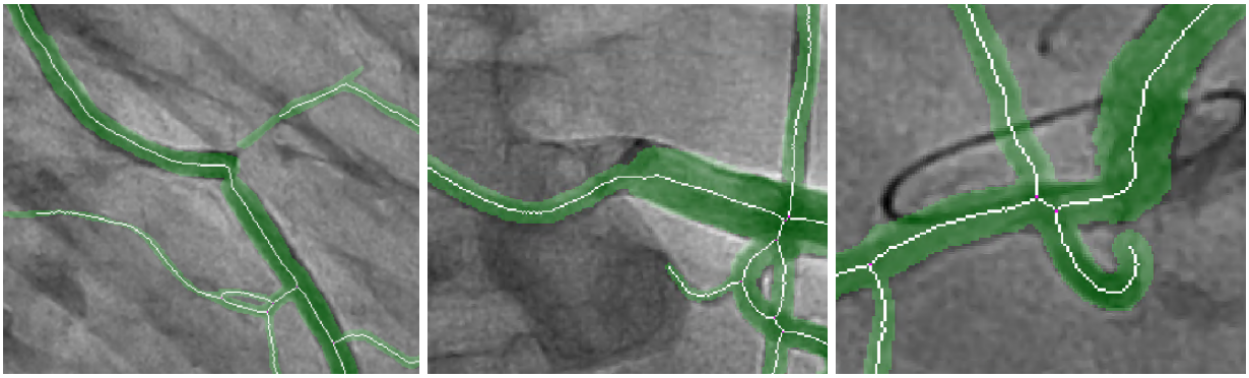


Figure 3.14 Exemples de segments de cathéter près de l'ostium.

de la direction du segment à chaque pixel (en degrés, à partir de quelques pixels consécutifs), d'adoucir la courbe de cette progression avec un filtre gaussien, de prendre la valeur absolue du gradient de cette progression puis de sélectionner la valeur de gradient la plus élevée. Si celle-ci est supérieure ou égale à une valeur prédéfinie et qu'elle ne se trouve pas parmi une longue séquence de gradients élevés, on considère que ce pixel du segment du cathéter est la position de l'ostium.

Pour détecter un élargissement brusque du segment, il est possible d'analyser la variation de la largeur du vaisseau dans un intervalle de plusieurs pixels. Si le ratio est supérieur ou égal à une valeur prédéfinie, il est probable qu'il s'agisse de l'ostium.

Lorsqu'aucun des deux indicateurs précédents n'est détecté, l'exploration du segment peut se terminer par une vraie bifurcation (qui peut être considérée comme la position de l'ostium) ou par un cul-de-sac. La manière la plus simple de détecter l'ostium à partir d'un segment du cathéter finissant en cul-de-sac est de trouver le point du squelette le plus proche de ce cul-de-sac (et ne faisant pas partie du segment du cathéter, évidemment). La troisième image de la figure 3.14 en montre un exemple.

Extraction des bifurcations et segments de vaisseaux de manière orientée

Une fois la position de l'ostium trouvée, il devient possible de générer un graphe orienté à partir du squelette. L'algorithme commence en parcourant les segments du squelette en prenant la position de l'ostium comme point de départ. Chaque bifurcation identifiée comme étant un croisement est ignorée pour que seulement les vraies bifurcations soient sauvegardées en tant que nœuds du graphe. Les segments reliant ces vraies bifurcations sont enregistrés en tant que liens dans le graphe. Plusieurs propriétés des bifurcations et segments sont conservées afin de caractériser les nœuds et liens du graphe. Pour les nœuds, les informations gardées sont

leur position, leur largeur, leur intensité et s'ils sont l'ostium ou non. Pour les liens, on garde leur longueur, leur largeur moyenne ainsi que leur intensité moyenne. L'ordre d'exploration des branches est choisi par rapport à la taille de la bifurcation de laquelle elle est issue. Cela fait généralement en sorte que les vaisseaux les plus larges sont explorés en premier, ce qui leur donne plus de chance d'être explorés dans la bonne direction.

3.2.2 Processus d'apprentissage

Pour entraîner un réseau de neurones par graphe à identifier des paires de bifurcations dans deux images différentes, il est nécessaire d'avoir des données à partir desquelles le modèle pourrait apprendre. Par contre, ce genre de données est très difficile à générer, car on doit avoir non seulement des images segmentées de la même artère dans différentes vues, mais on doit aussi savoir quelles bifurcations sont les mêmes dans les deux vues. Puisqu'il n'existe pas de jeu de données contenant ce genre d'information, créer celui-ci avant de pouvoir entraîner le modèle est inévitable et cela prend beaucoup de temps. Une fois ce jeu de données créé, il devient possible d'entraîner un réseau de neurones siamois par graphe pour générer des descripteurs de nœuds permettant d'effectuer des mises en correspondance.

Création de paires de bifurcations

Puisque le modèle de réseau de neurones siamois par graphe a besoin des données *ground truth* pour avoir des paires positives, il est nécessaire de créer un jeu de données dans lequel ces paires sont identifiées. Pour se faire, il est possible de créer un outil mettant côte-à-côte deux images de la même artère sous différents angles. Cet outil, codé avec la librairie PyQt5 en python et présenté dans la figure 3.15, permet de charger deux images, de préférence des images dont la segmentation et le squelette sont superposés à l'image angiographique brute, puisque cela permet à l'utilisateur de savoir où sont les bifurcations identifiées par l'algorithme de squelettisation. Il est ensuite possible de tracer des lignes entre ces bifurcations d'une image à l'autre. Puisqu'il est peu probable que chaque bifurcation d'une image puisse être liée à une bifurcation de l'autre image, ces paires peuvent être identifiées de manière éparse. Le processus d'entraînement du modèle siamois n'a ensuite qu'à utiliser les bifurcations qui ont été identifiées par cet outil et ignorer les bifurcations du squelette qui ne font pas partie de la liste des paires. Pour lier ces paires aux noeuds des graphes, il suffit de vérifier la position des paires dans les images et sélectionner le noeud le plus près à l'intérieur d'un rayon de quelques pixels (afin d'éviter qu'une bifurcation identifiée par l'outil ne fasse pas partie de la liste des noeuds extraits du squelette, dans le cas d'un graphe orienté dans lequel la bifurcation aurait été identifiée comme un croisement).

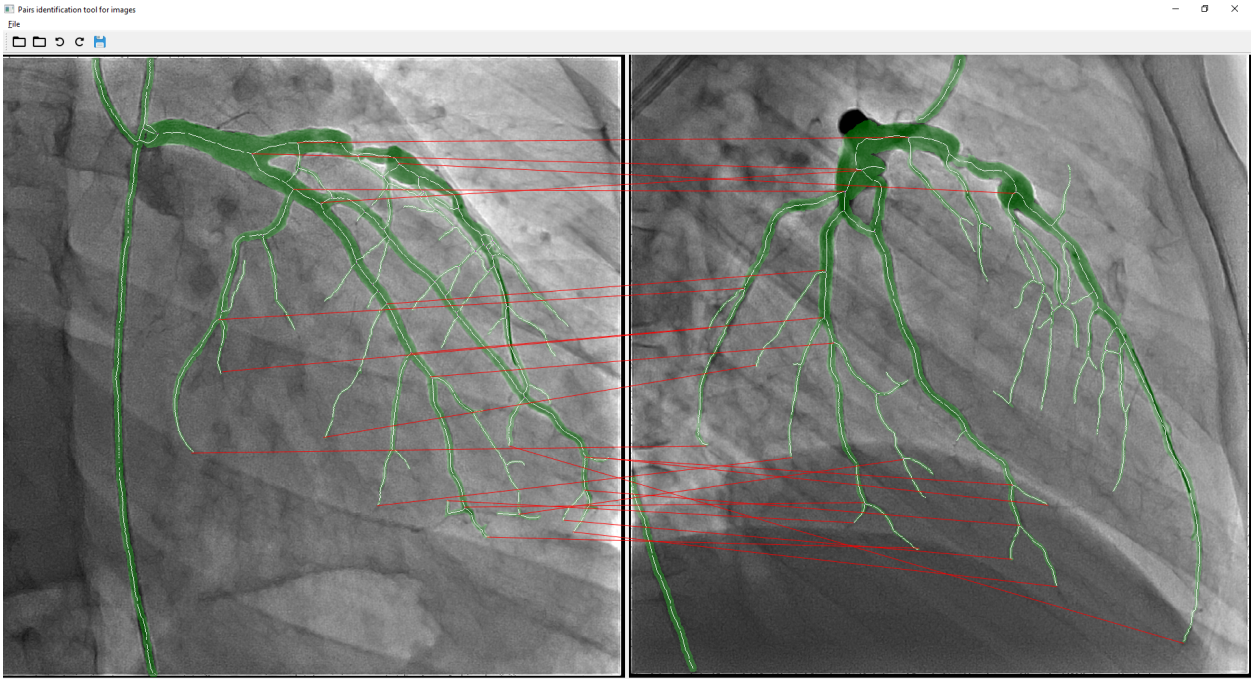


Figure 3.15 Outil d'identification manuel de paires de bifurcations et de points de terminaison.

Étant donné la quantité limitée de segmentations disponibles, seulement 3 paires d'images ont pu être annotées à l'aide de cet outil. À partir de ces 3 paires, 23, 18 et 13 paires valides ont pu être identifiées pour créer le jeu de données d'entraînement et de validation.

Réseau de neurones siamois par graphe

Pour trouver des paires de nœuds dans deux graphes, un réseau de neurones siamois par graphe est utilisé. Les réseaux de neurones convolutifs sur graphe (ConvGNN) permettent de propager l'information de chaque nœud vers ses voisins lors de chaque couche de convolution, ce qui leur permet d'obtenir des informations à la fois locales et globales lorsqu'ils génèrent les vecteurs de caractéristiques des nœuds après plusieurs convolutions. Cela permet donc de considérer une bonne partie de la structure de l'arbre coronarien pour décrire chaque bifurcation. Les modèles de type siamois permettent de définir une fonction de perte qui les encourage à réduire la distance entre certaines paires de nœuds et maximiser la distance pour d'autres, à condition d'avoir des données *ground truth* listant ces paires positives et négatives. Il n'est pas nécessaire d'avoir toutes les bifurcations des graphes présentes dans les paires de données *ground truth*, car seulement celles présentes sont utilisées pour générer des paires positives et négatives. Théoriquement, avec suffisamment d'information des nœuds se propageant dans le réseau de neurones, ce modèle devrait arriver à générer des vecteurs de

caractéristiques qui décrivent suffisamment bien chaque bifurcation pour pouvoir trouver des paires entre deux vues de la même artère. Une vue d'ensemble du modèle est présentée à la figure 3.16.

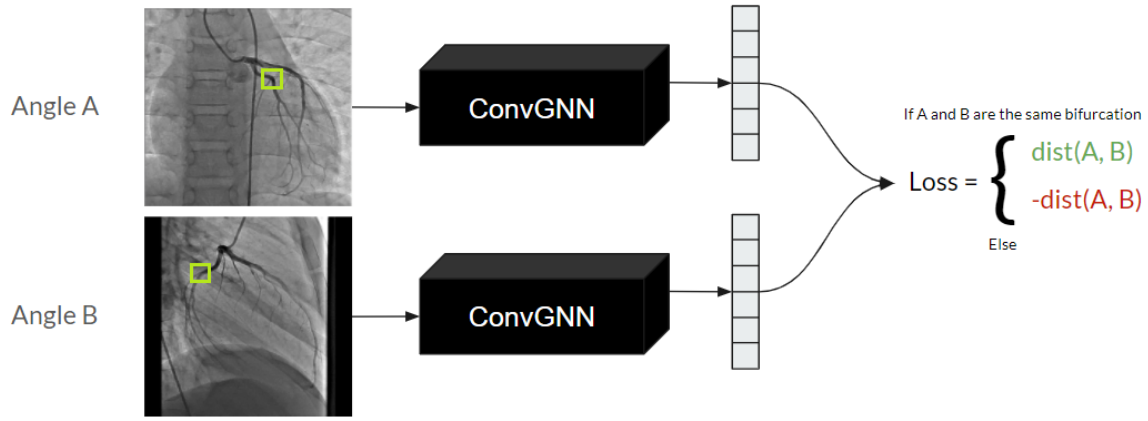


Figure 3.16 Vue d'ensemble du modèle de réseau de neurones siamois par graphe.

Puisqu'il est possible de créer beaucoup plus de paires négatives que de paires positives, la fonction de perte doit tenir compte de ce débalancement afin d'attribuer une importance similaire à la minimisation de la distance entre les vecteurs de caractéristiques des nœuds des paires positives et la maximisation de la distance entre ceux des paires négatives.

CHAPITRE 4 RÉSULTATS

Cette section du mémoire présente et analyse les résultats obtenus par les méthodes présentées dans le chapitre précédent. Elle montre les résultats du modèle d'apprentissage profond pour la tâche de synchronisation, mentionne les valeurs des différents paramètres pour les algorithmes relatifs à la l'extraction du graphe orienté à partir du squelette et présente les résultats préliminaires de la preuve de concept pour la tâche de mise en correspondance spatiale.

4.1 Synchronisation des séquences angiographiques

Les résultats de la tâche de synchronisation sont quantifiables en les comparant aux données *ground truth*. Pour se faire, la méthode propose un calcul de score de synchronisation basé sur l'alignement avec les signaux d'ECG. Ce score permet aussi de sélectionner les meilleurs hyperparamètres du modèle, qui sont explorés de manière aléatoire. Dans un premier temps, le modèle est entraîné et testé sur le jeu de données de patients pédiatriques. Par la suite, le meilleur modèle est testé sur le second jeu de données composé des données du premier jeu ainsi que de nouvelles données provenant d'un autre hôpital, afin de vérifier s'il peut généraliser à des patients adultes. Finalement, un nouveau modèle est entraîné à partir du deuxième jeu de données puis évalué pour comparer les résultats avec celui entraîné seulement sur les données de patients pédiatriques.

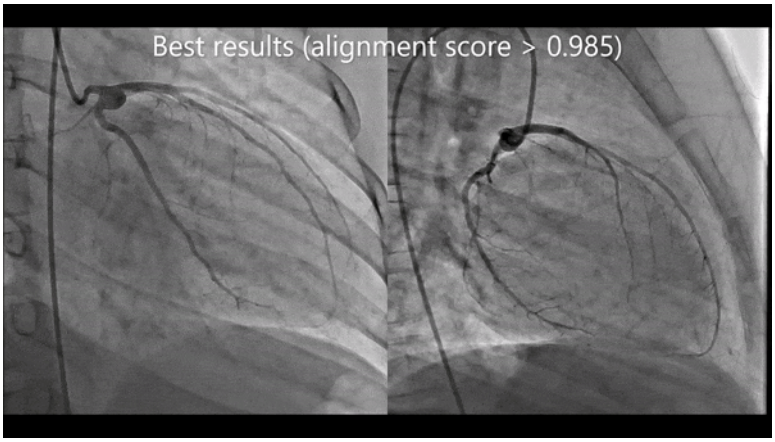


Figure 4.1 Animation des résultats de synchronisation sur des séquences angiographiques de patients pédiatriques. La vidéo est disponible ici.

4.1.1 Calcul du score des synchronisations

Après la recherche de chemin sur les matrices combinant la distance et la similarité cosinus, les valeurs des éléments sélectionnés sont comparées à ceux de la matrice de données *ground truth* générée à partir des ondes R extraites des signaux d'ECG. Pour calculer le score d'une synchronisation, il suffit de prendre la moyenne des valeurs du chemin sur la matrice de données *ground truth*. Cette valeur se situe entre 0 et 1, où 0 signifie une désynchronisation totale (½ cycle cardiaque de distance tout au long du chemin) et 1 signifie une synchronisation parfaite. Par contre, une valeur de 1 n'est pas toujours possible, puisque seulement un nombre fini d'images composent un cycle cardiaque. Cela fait en sorte que le meilleur score possible pour un index particulier est souvent légèrement inférieur à 1. Afin d'obtenir un score de synchronisation qui considère cette limite, on peut normaliser celui-ci en le divisant par le score obtenu grâce à une recherche de chemin faite directement sur la matrice de données *ground truth*. Pour évaluer un modèle, on prend la moyenne des scores de synchronisation sur toutes les données de l'ensemble de test.

4.1.2 Analyse des résultats de synchronisation

Le meilleur modèle entraîné obtient un score moyen de 0.9604, mais la médiane est plus haute à 0.979. Pour avoir une meilleure idée de la distribution des scores de synchronisation obtenus à partir d'un modèle, l'usage d'un histogramme cumulatif est approprié. Cela permet de détecter s'il y a des anomalies dans les résultats dans le cas où quelques synchronisations auraient un score très bas.

Grâce à l'histogramme cumulatif, on peut voir qu'environ 5% des synchronisations ont un score inférieur à 0.85. Lorsque l'on s'attarde aux résultats ayant un score inférieur à la moyenne, on peut s'apercevoir que la plupart des valeurs inférieures à la moyenne se trouvent entre les ondes R, tel qu'affiché dans la figure 4.3. Cela peut s'expliquer par le fait que la phase diastolique ne démarre pas nécessairement exactement au centre des deux ondes R l'entourant. Puisque les données *ground truth* ne s'assurent d'aligner que le début de la phase systolique et que le modèle apprend à synchroniser les images à partir de leur mouvement, il arrive qu'une divergence de l'alignement présente dans les données *ground truth* au niveau du début de la phase diastolique cause une diminution du score. Le modèle semble réussir à synchroniser la phase diastolique suffisamment bien pour que la différence avec les données *ground truth* n'affecte le score que pour quelques paires de séquences.

Parmi les 124 paires de vidéos de l'ensemble de test, 8 d'entre elles sont des séquences acquises de manière simultanée par un C-arm biplan à 15 trames par seconde pour chaque plan.

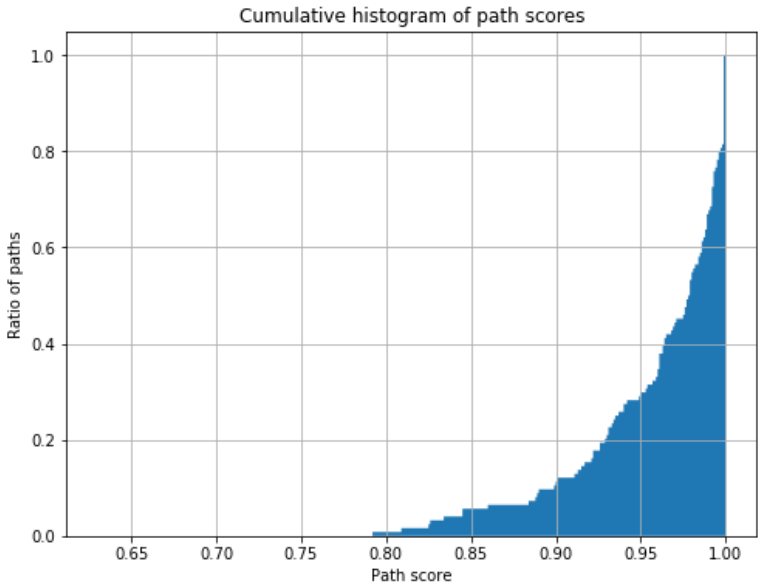


Figure 4.2 Histogramme cumulatif des scores de synchronisation du meilleur modèle.

Comme on pourrait le prévoir, ces paires ont un score plus haut que la moyenne, à 0.9857. Les paires de vidéos combinant des séquences à 15 et 30 trames par seconde obtiennent des résultats similaires à la moyenne.

Suite à l'entraînement sur le premier jeu de données, le modèle a été testé sur le second. Puisque la distribution des pixels des images provenant des vidéos de patients adultes diffère grandement de celle sur laquelle le modèle a été entraîné, celui-ci ne performe pas très bien sur les nouvelles données, avec un score de 0.754. Tel que présenté par la figure 4.4, les images du jeu de données générées avec des patients adultes semblent plus floues et les structures vasulaires ne sont pas toujours entièrement visibles en comparaison avec celles des patients pédiatriques. Il est donc nécessaire de réentraîner le modèle avec ces nouvelles données puisqu'elles diffèrent trop de celles de l'entraînement initial. Suite à la recherche d'hyperparamètres du modèle entraîné sur le nouveau jeu de données composé des deux précédents combinés, le score du meilleur modèle est de 0.9424. La pire synchronisation du nouvel ensemble de test provient d'une séquence de l'artère coronaire gauche acquise avec un angle crânial de 30 degrés et un angle latéral droit de 40 degrés, ce qui est très peu courant dans l'ensemble du jeu de données. La rareté de ce genre de séquences peut expliquer la difficulté que le modèle a à synchroniser celles-ci.

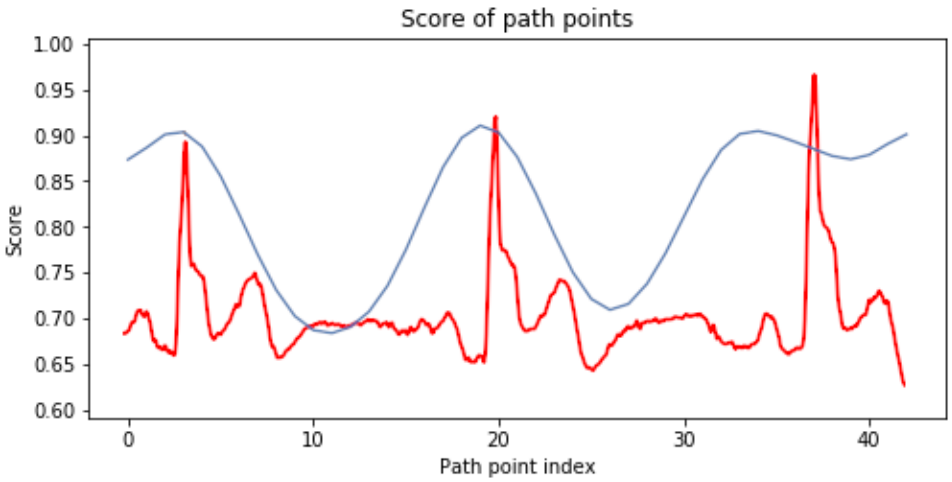


Figure 4.3 Superposition du score des points du chemin (en bleu) et des signaux d'ECG (en rouge).

4.1.3 Hyperparamètres d'entraînement

Pour optimiser les hyperparamètres du modèle, une recherche aléatoire est exécutée. Parmi les paramètres explorés, plusieurs sont classiques, tels que le choix du modèle (ResNet-50, MobileNet-v2, EfficientNet-b0 et EfficientNet-b1), le taux d'apprentissage, la taille de la dernière couche dense, la taille de la *mini-batch*, le taux de *dropout*, la fréquence de diminution et réinitialisation du taux d'apprentissage, la normalisation ou non du jeu de données ainsi que l'utilisation d'augmentation de données ou non. D'autres sont plus adaptés au problème de synchronisation des séquences angiographiques, tels que l'utilisation de paires créées à partir de différentes séquences ou provenant exclusivement de la même séquence ainsi que l'utilisation d'un maximum de nombre de cycles cardiaques de distance pour l'utilisation de paires provenant de la même séquence. Lors de l'entraînement sur le jeu de données contenant exclusivement des séquences de patients pédiatriques, la recherche aléatoire d'hyperparamètres a trouvé les valeurs listées dans le tableau 4.1.

On peut constater que les valeurs des hyperparamètres varient grandement d'un entraînement à l'autre tout en affichant un score similaire, ce qui peut laisser croire que ceux-ci ne sont pas si importants. Par contre, certains entraînements terminent avec un score beaucoup plus bas, parfois près de 0.6. On peut donc conclure qu'il existe un très grand nombre de combinaisons d'hyperparamètres permettant au modèle de bien performer, mais qu'il en existe aussi d'autres qui nuisent au modèle.

Pour ce qui est de l'entraînement sur le jeu de données regroupant les patients pédiatriques et adultes, les hyperparamètres du tableau 4.2 sont ceux qui ont performé le mieux sur

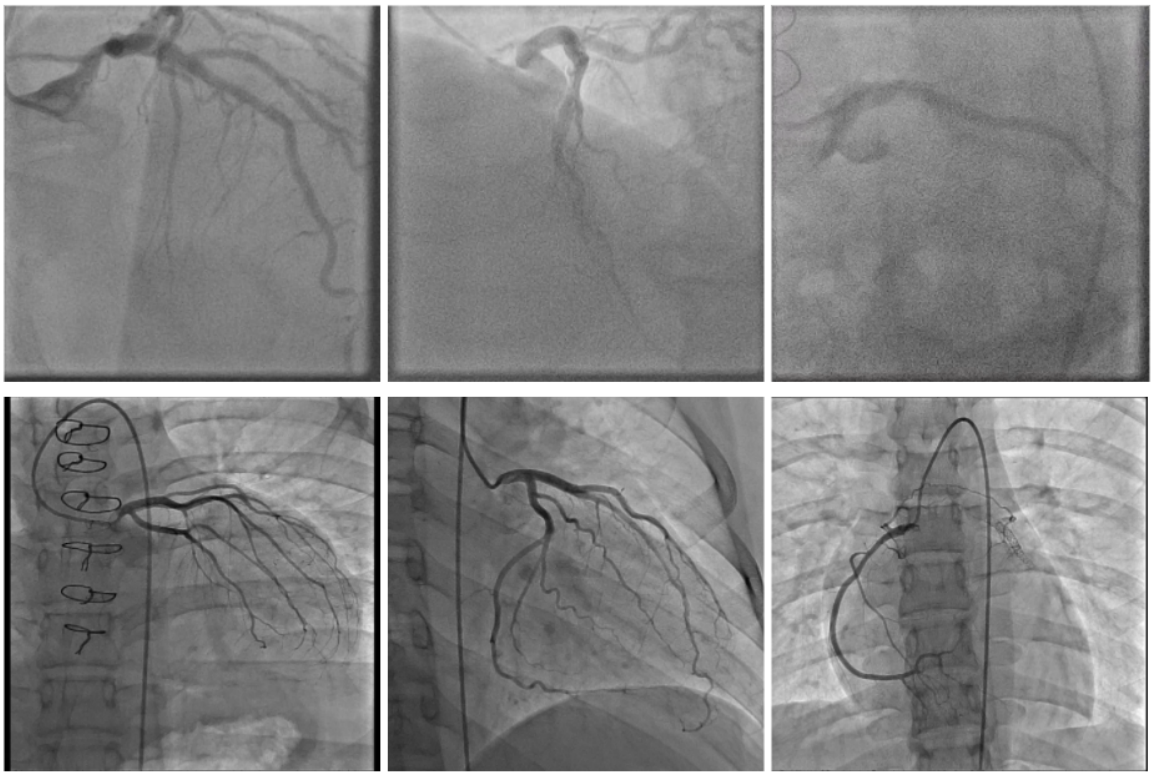


Figure 4.4 Exemples d'images angiographiques provenant de patients adultes (haut) et de patients pédiatriques (bas).

Tableau 4.1 Meilleurs hyperparamètres trouvés pour l'entraînement sur données de patients pédiatriques.

AM	TDCD	TMB	TD	NMC	PIV	AD	Score
ResNet-50	64	32	0.183	N/A	non	oui	0.9549
MobileNet	128	32	0	2.02	non	non	0.9558
EfficientNet-b0	8	16	0	N/A	oui	oui	0.9581
EfficientNet-b1	4	12	0	N/A	non	non	**0.9604**

Tableau 4.1 **AM** : architecture du modèle, **TDCD** : taille de la dernière couche dense, **TMB** : taille des *mini-batchs*, **TD** : taux de *dropout*, **NMC** : nombre maximal de cycles cardiaques de distance, **PIV** : paires inter-vidéos, **AD** : augmentation de données.

l'ensemble de test.

La durée moyenne du *fine-tuning* des modèles en 20 époques est d'environ 2h sur une carte graphique GeForce RTX 2070.

Tableau 4.2 Meilleurs hyperparamètres trouvés pour l'entraînement sur données de patients pédiatriques et adultes.

AM	TDCD	TMB	TD	NMC	PIV	AD	Score
EfficientNet-b0	32	24	0.453	N/A	non	oui	0.9424

Tableau 4.2 **AM** : architecture du modèle, **TDCD** : taille de la dernière couche dense, **TMB** : taille des *mini-batchs*, **TD** : taux de *dropout*, **NMC** : nombre maximal de cycles cardiaques de distance, **PIV** : paires inter-vidéos, **AD** : augmentation de données.

4.2 Détection de correspondances spatiales

L'efficacité de l'algorithme de détection d'ostium peut être quantifiée car un jeu de données *ground truth* a été créé. L'efficacité de l'algorithme d'extraction de graphe, quant à elle, ne peut pas être totalement quantifiée puisqu'il est très difficile de savoir quelles bifurcations sont des croisements et d'avoir la vraie direction des segments. Par contre, il est possible de créer une métrique semblable à la spécificité, c'est-à-dire qui peut déterminer si l'algorithme génère des faux-positifs ou non. On peut aussi visualiser les résultats de l'algorithme pour juger de sa performance et identifier ses failles. Pour ce qui est de l'entraînement du modèle de réseau de neurones par graphe, il est possible d'analyser les vecteurs de caractéristiques générés à partir des nœuds d'un même graphe pour juger de son efficacité à distinguer ceux-ci. De plus, grâce aux paires de bifurcations identifiées manuellement pour des paires de graphes, il est possible de quantifier la performance du modèle siamois à identifier les correspondances entre les nœuds d'une paire de graphes.

4.2.1 Segmentation

Tel que mentionné dans la méthodologie, les segmentations utilisées pour générer les squelettes sont celles ayant été corrigées manuellement, puisque les segmentations obtenues de manière complètement automatique ne permettent pas de créer des squelettes exempts d'imperfections. La figure 4.5 montre un exemple de segmentation automatique qui présente de gros segments de vaisseaux non identifiés. Puisque ce modèle génère une probabilité de vasculature pour chaque pixel, il est possible de le faire varier pour évaluer son effet sur la segmentation finale. Lorsque celui-ci est trop élevé, de gros segments d'artère sont manquants. Lorsqu'il est réduit, non seulement les segments manquant le sont toujours, mais des portions de l'image ne présentant pas de vaisseaux sont faussement identifiées. Il est donc

préférable d'utiliser des segmentations manuellement corrigées. La figure 4.6 en montre un exemple.

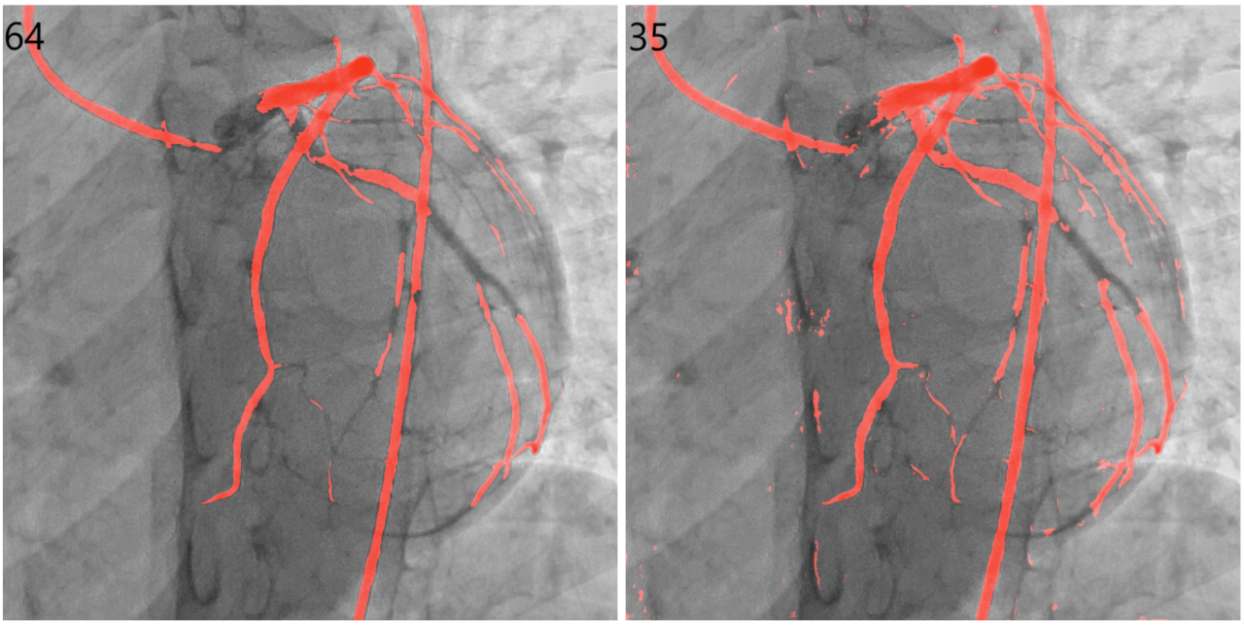


Figure 4.5 Exemple de segmentation automatique imparfaite. Le nombre dans le coin gauche représente le seuil de probabilité de vasculature sur 255.

4.2.2 Description des valeurs des hyperparamètres employés

Plusieurs hyperparamètres sont nécessaires pour le bon fonctionnement des différents algorithmes utilisés pour générer les données nécessaires à l'entraînement d'un modèle pour la détection de correspondances. L'optimisation de ces hyperparamètres n'a pas été effectuée de manière exhaustive, étant donné que l'objectif est d'effectuer une preuve de concept pour la mise en correspondance spatiale des vaisseaux à l'aide de réseaux de neurones sur graphes. Leur valeurs ont toutes été choisies de manière empirique, en se fiant soit à des résultats visuels, soit à une métrique décrivant la performance de l'algorithme sur tout le jeu de données. Les techniques d'évaluations utilisées sont décrites dans les sections 4.2.3 et 4.2.4.

Squelettisation

Dans le chapitre précédent, une méthode d'élagage de branches d'un squelette a été présentée par la formule (3.5). Les valeurs des hyperparamètres α et β inclus dans la formule, qui servent respectivement à ajuster la valeur de base du seuil et à ajuster le seuil proportionnellement à la longueur de la branche, ont été choisies de manière empirique. Leur valeur respective de

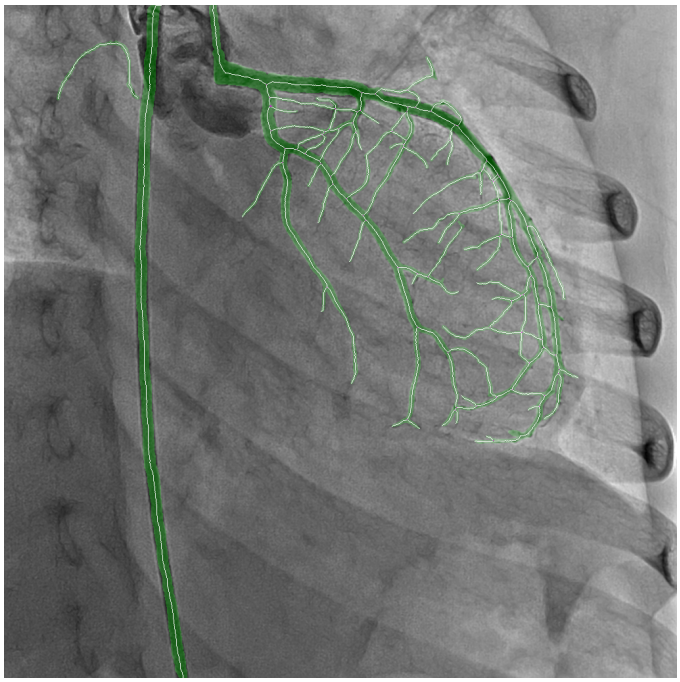


Figure 4.6 Exemple de segmentation corrigée manuellement avec superposition du squelette.

0.1 et 0.01 sont celles qui ont donné les meilleurs résultats. Un exemple d'élagage est présenté à la figure 4.7.

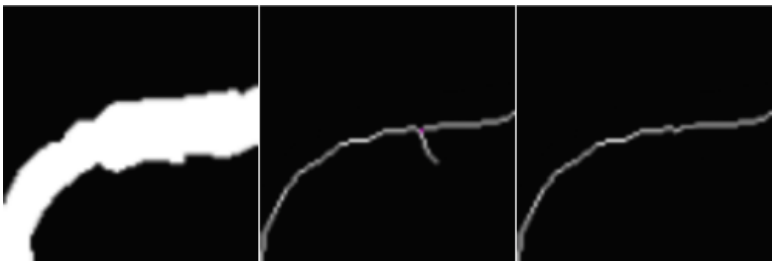


Figure 4.7 Exemple d'élagage de branches du squelette des vaisseaux.

Détection des bifurcations et croisements

Afin de permettre aux algorithmes de mieux généraliser sur des images de différentes tailles, les valeurs des hyperparamètres ayant un lien avec une distance en pixels sont toutes proportionnelles à la taille de l'image. Les valeurs mentionnées ci-dessous sont valides pour des images de taille 1024x1024.

Lors de l'explication de l'algorithme pour détecter les croisements, le nombre de pixels utilisés pour calculer la direction et la taille du segment a été omis. Lors du développement de

l'algorithme, une valeur de 20 pixels a été choisie empiriquement, car c'est ce qui semble donner les meilleures performances sur l'ensemble du jeu de données. De plus, pour calculer la largeur du segment, plutôt que de prendre directement la moyenne de largeur à chaque pixel, ces largeurs sont triées et le premier quartile est choisi pour représenter la largeur du segment. La logique derrière ce choix est que la segmentation des pixels près de la bifurcation a tendance à avoir une largeur plus grande que ceux plus loin car la superposition des vaisseaux qui se croisent augmente la largeur. De plus, prendre la valeur minimale des 20 pixels aurait pour effet de trop souvent refléter une irrégularité de la segmentation.

Pour ce qui est de la sélection du seuil d'angle présenté par la formule (3.6) et utilisé pour choisir quelles bifurcations sont des candidats potentiels pour la détection de croisement, trois hyperparamètres ont été eux aussi choisis empiriquement. Pour α, représentant l'angle maximal permis, une valeur de 90 degrés lui est donnée. Pour β, qui lui représente l'angle maximal accepté lorsque la bifurcation est à la distance maximale, une valeur de 15 degrés lui est donnée. Finalement, pour la distance maximale utilisée par τ, représentant la fraction de la distance de la bifurcation sur la distance maximale, une valeur de 100 pixels est choisie.

Pour filtrer davantage les candidats potentiels pour la détection de croisements, les valeurs de deux autres hyperparamètres ont été choisies de manière empirique. Ceux-ci sont la différence d'angle maximale entre les deux segments (avant la première bifurcation et après le candidat potentiel) ainsi que le ratio de largeur entre les deux segments. Des valeurs de 30 degrés et de 1.45 pour le ratio ont été choisies.

Pour la formule de calcul de score pour un candidat potentiel (formule 3.7), les hyperparamètres α, β et γ possèdent tous des valeurs mentionnées précédemment, qui sont de 30 degrés pour l'angle maximal, 1.45 pour le ratio maximal de largeur et 100 pixels pour la distance maximale.

Détection d'ostium

Pour calculer la direction d'un segment de cathéter, un vecteur de direction est calculé pour chaque pixel à partir de 5 pixels consécutifs. Ce nombre permet d'avoir une direction assez précise de la portion du segment sur laquelle le pixel se trouve, sans trop l'être, ce qui aurait pour effet de produire des vecteurs de direction qui changent trop brusquement. Ensuite, on cherche à trouver des valeurs de gradient de la direction supérieures à 2.75 (valeur trouvée de manière empirique). Si une valeur dépasse ce seuil, on vérifie si elle fait partie d'une séquence de 30 pixels consécutifs ou plus dont le gradient est supérieur au seuil. Si c'est le cas, il est plus probable que cette courbe soit un crochet dans le cathéter plutôt que la fin du cathéter et le début de l'artère.

Lorsqu'on tente de détecter un élargissement brusque du segment du cathéter pour trouver le début de l'artère, on compare la largeur du pixel courant et du pixel positionné 30 points avant. Si ce ratio est supérieur ou égal à 1.6, on considère que le pixel courant est l'ostium.

4.2.3 Analyse des résultats de la détection d'ostium

L'algorithme de détection d'ostium dépend d'une grande quantité d'hyperparamètres et faire varier les valeurs de ceux-ci peut avoir une grande influence sur le résultat de détection d'ostium. Il est donc important d'avoir une métrique à consulter à chaque changement pour savoir si celui-ci est bénéfique ou nuisible. Dans cette optique, un simple programme python a été développé pour permettre à un utilisateur de cliquer sur une image angiographique afin d'identifier l'endroit où il croit que l'ostium se trouve. À l'aide de ce programme, il est possible de créer un jeu de données *ground truth* permettant de comparer la position de l'ostium détectée par l'algorithme et celle identifiée par l'utilisateur du programme. Pour évaluer l'efficacité de l'algorithme après avoir effectué un changement, on peut calculer la moyenne des distances pour chaque image du jeu de données et ainsi savoir grâce à cette métrique si le changement est efficace. Pour les 71 images du jeu de données, de taille 1024x1024, la distance moyenne entre la position de l'ostium détectée par l'algorithme et celle identifiée par l'utilisateur est de 30.29 pixels, ce qui en général est relativement près.

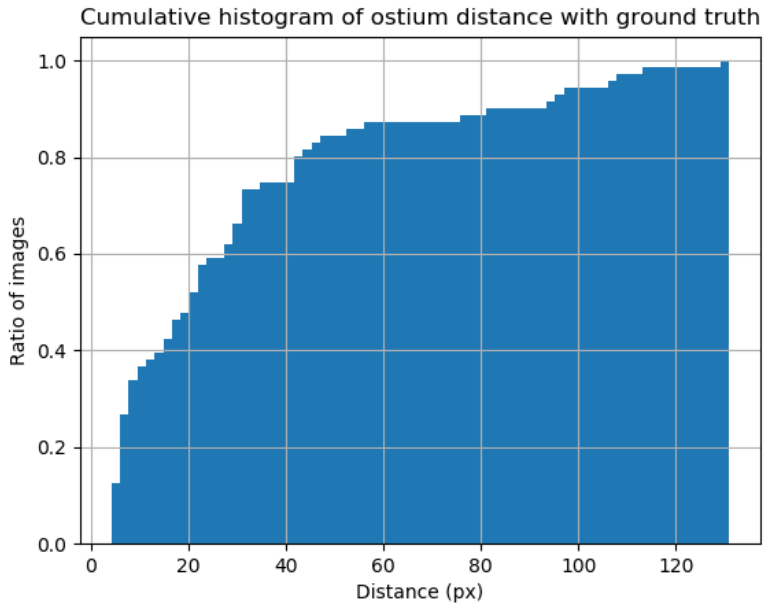


Figure 4.8 Histogramme cumulatif des distances entre la position de l'ostium détectée par l'algorithme et celle identifiée manuellement.

Quelques images où l'ostium est détecté à plus de 100 pixels de distance font grandement

augmenter la moyenne et la majorité de celles-ci ne peuvent pas facilement être améliorées à l'aide d'un algorithme basé sur le suivi du cathéter car celui-ci est superposé au début de l'artère sur une longue distance.

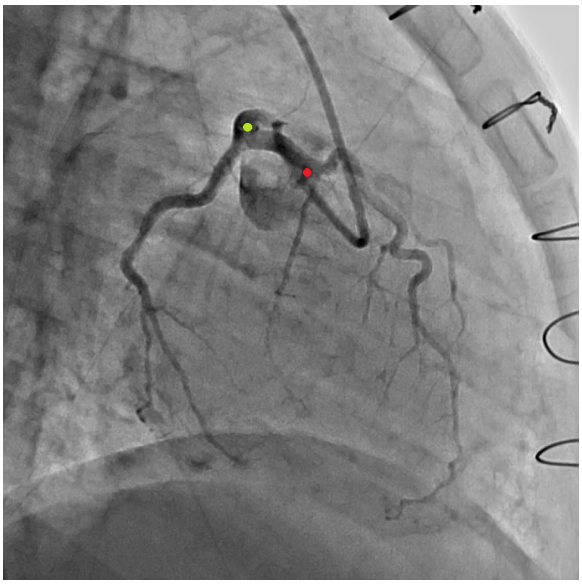


Figure 4.9 Image angiographique où l'algorithme de détection d'ostium performe le moins bien. Le point vert est la position identifiée par l'utilisateur et le point rouge est celle détectée par l'algorithme.

4.2.4 Analyse des résultats d'extraction de graphe

Tel que mentionné plus tôt, il n'y a pas de données *ground truth* pour l'extraction du graphe orienté à partir du squelette. La seule façon de juger de sa performance est de créer une métrique similaire à la spécificité basée sur le nombre de branches finales (qui terminent en cul-de-sac) présentes dans le graphe orienté par rapport à celui non-orienté. La figure 4.10 présente l'histogramme cumulatif de cette métrique. On peut voir quelques cas où il y a un grand nombre de faux-positifs de détection de croisement (ce qui cause des branches manquantes) mais aussi qu'environ la moitié des graphes possèdent 85% ou plus des branches présentes dans les graphes non-orientés. La moyenne est comparable, à 83.2%.

Il est aussi possible de visualiser les graphes générés pour juger de l'efficacité de l'algorithme. La figure 4.11 présente une comparaison des graphes orientés et non orientés extraits du squelette. La direction d'un lien dans le graphe orienté est représentée par un point au bout du segment. En inspectant le graphe orienté, on peut s'apercevoir qu'il ne semble pas y avoir de croisements faussement identifiés comme des bifurcations, contrairement au graphe non-orienté où chaque croisement est représenté par des bifurcations, et les segments pointent

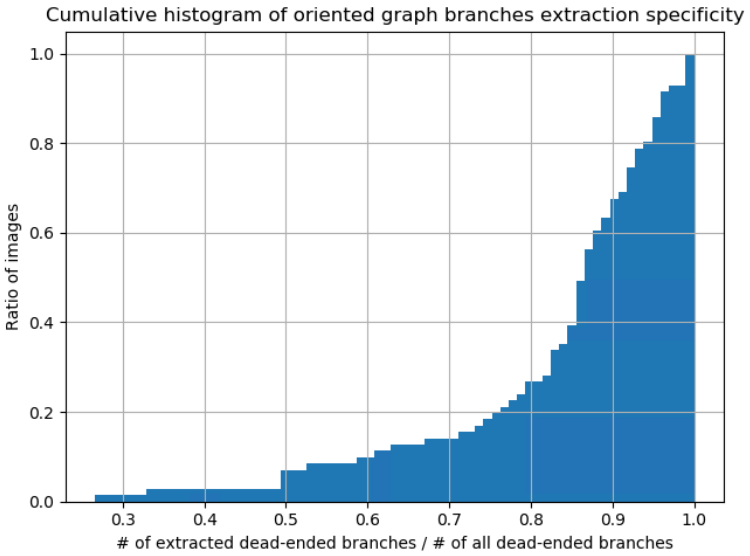


Figure 4.10 Histogramme cumulatif du ratio du nombre de branches finales des graphes orientés par rapport au nombre de branches finales des graphes non-orientés.

généralement dans la bonne direction.

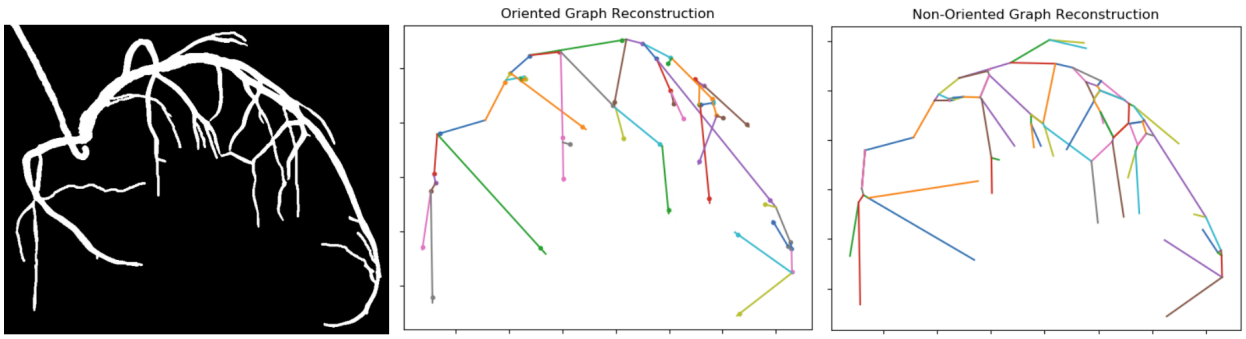


Figure 4.11 Exemple d'extraction de graphe à partir d'une segmentation.

4.2.5 Entraînement d'un réseau de neurones par graphe pour distinguer les nœuds

Dans la librairie python "Pytorch Geometric", une opération de convolution sur les nœuds d'un graphe (GCNConv) est déjà implémentée et il est possible de les empiler pour créer un modèle capable de propager l'information des noeuds à leurs voisins. Ce genre de modèle peut apprendre à distinguer les nœuds à partir d'un seul graphe à la fois grâce à une fonction de perte (définie par l'équation (4.1)) qui maximise la distance entre les vecteurs de caractéristiques générés par le réseau de neurones.

$$loss = -\frac{1}{|N|^2}\sum_{i=1}^{|N|}\sum_{j=1}^{|N|}\sum_{k=1}^{V} min\left(0.25, \left(|N_i|_k - |N_j|_k\right)^2\right) \tag{4.1}$$

Dans cette équation, N est l'ensemble des nœuds du graphe et V est la taille des vecteurs de caractéristiques générés par le réseau de neurones. L'idée derrière l'utilisation de la normalisation des vecteurs de caractéristiques est que si cette restriction de taille n'est pas donnée, le modèle n'aurait qu'à générer des vecteurs avec des valeurs extrêmement grandes pour maximiser la distance entre ceux-ci. Pour ce qui est de l'utilisation de la valeur maximale de 0.25 si la distance est plus grande, cela permet de prévenir le réseau de générer seulement deux grappes de points où la distance des points de la première grappe avec ceux de la deuxième est maximale mais où celle-ci est nulle entre les points d'une même grappe.

Bien que maximiser la distance entre les vecteurs de caractéristiques des nœuds d'un graphe ne permet pas de trouver des paires de nœuds dans deux graphes différents, cela permet de valider qu'un réseau de neurones par graphe arrive à apprendre à distinguer les nœuds d'un même graphe, ce qui s'approche de la tâche finale.

4.2.6 Résultats de l'entraînement pour distinguer les nœuds

En affichant les vecteurs de caractéristiques à deux dimensions générés par le réseau pour chaque nœud, on peut juger de l'efficacité du modèle à distinguer ceux-ci. Tel que présenté par la figure 4.12, les valeurs normalisées des vecteurs de caractéristiques à deux dimensions générés par le modèle avant l'entraînement sont très similaires, ce qui ne permet pas de bien distinguer les différents nœuds, puisque la distance entre les différentes paires possibles est très petite. Une fois l'entraînement terminé, les valeurs normalisées sont beaucoup mieux réparties sur l'ensemble des valeurs possibles, autant pour les nœuds provenant des 3 graphes de l'ensemble d'entraînement que pour ceux des 2 graphes de l'ensemble de validation ou du graphe de l'ensemble de test. Ces résultats laissent croire que ce genre de modèle peut effectivement apprendre à générer des vecteurs de caractéristiques différents pour différents nœuds, ce qui est utile pour une tâche de mise en correspondance.

4.2.7 Résultats de l'entraînement du modèle siamois

Puisque le jeu de données est très petit, deux des trois paires de graphes sont utilisées comme données d'entraînement alors que l'autre compose l'ensemble de validation. Le modèle composé de 3 couches de GCNConv est entraîné pendant 2000 époques à l'aide de la fonction d'optimisation Adam et un taux d'apprentissage de 0.001. Tel que présenté par la figure 4.13,

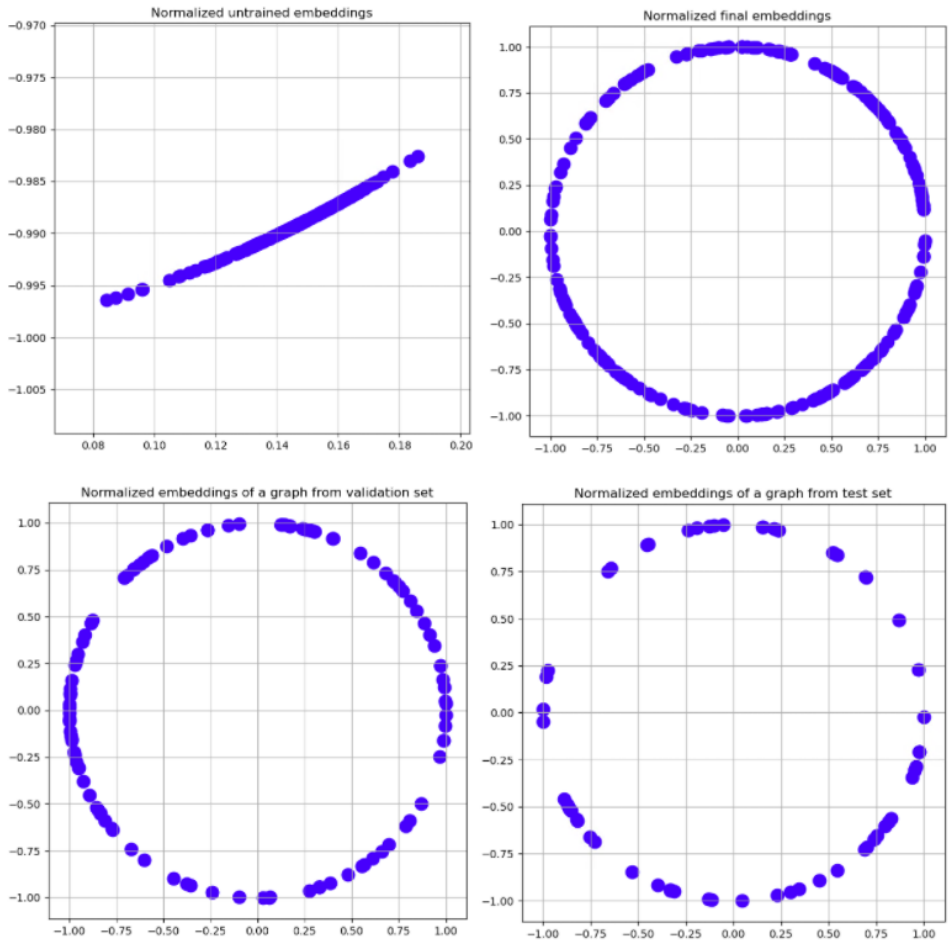


Figure 4.12 Vecteurs de caractéristiques à 2 dimensions normalisés générés par le GCN pour chaque nœud d'un graphe. Rangée du haut : valeurs des données d'entraînement avant et après l'apprentissage. Rangée du bas : valeurs des données de validation et de test après l'apprentissage.

le modèle arrive à optimiser la fonction de perte qui minimise la distance entre les paires de nœuds positives et qui maximise la distance entre les paires de nœuds négatives. Plus l'entraînement avance, plus le modèle fait du surapprentissage sur les données d'entraînement, ce qui diminue sa performance sur les données de validation.

Afin de juger de la performance du modèle, il est possible de calculer la distance entre chaque nœud des paires de graphes pour voir si le modèle arrive à générer des vecteurs de caractéristiques similaires pour les paires positives. À l'aide de ces distances, on peut compter le nombre de paires bien identifiées comme positives en les comparant aux données *ground truth* faites à la main. Pour chaque paire de graphe, on parcourt chaque nœud du graphe A pour calculer la distance entre son vecteur de caractéristiques à deux dimensions avec celui

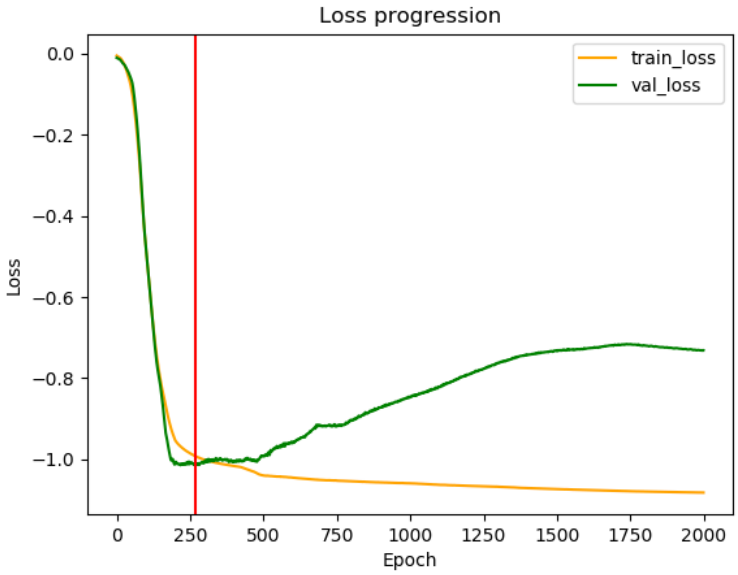


Figure 4.13 Progression de la *loss* lors de l'entraînement du GCN siamois pour la mise en correspondance des nœuds d'une paire de graphe. La ligne rouge représente l'époque où la *loss* de validation est à son minimum.

de chaque nœud du graphe B. Ces distances sont ensuite ordonnées de manière croissante afin de vérifier la position de chaque paire positive dans les listes. Les résultats sont affichés dans le tableau 4.3. La colonne Top-1 montre le nombre de paires positives dont la distance est la plus petite de la liste et la colonne Top-5 montre la quantité dont la distance est parmi les 5 plus petites. La dernière colonne montre la position moyenne des paires positives dans les listes de distances et la valeur entre parenthèses représente le percentile de ces positions. Une valeur parfaite de 1 représenterait un cas où toutes les paires positives seraient les plus petites de leur liste respective et celles-ci seraient dans le 100e percentile. En comparaison, un modèle aléatoire aurait en moyenne la position des paires positives au centre des listes, ce qui représenterait le 50e percentile.

Tableau 4.3 Résultats de mise en correspondance des nœuds à partir de paires de graphes grâce au modèle siamois.

Paire de graphes	Ensemble	Top-1	Top-5	Position moyenne
#1	Entraînement	6/23 (26.08%)	14/23 (60.86%)	7.8/67 (89.85%)
#2	Entraînement	5/18 (27.77%)	10/18 (55.55%)	11.1/56 (81.96%)
#3	Validation	2/13 (15.38%)	6/13 (46.15%)	7/55 (89.09%)

Afin de voir si le modèle est bien adapté à la tâche de mise en correspondance des nœuds des graphes, il est pertinent de tenter de le surentraîner et de comparer les résultats obtenus. Lorsqu'on entraîne le modèle pendant 10 000 époques, on peut voir la performance augmenter sur les graphes de l'ensemble d'entraînement et diminuer sur le graphe de l'ensemble de validation, tel que présenté par le tableau 4.4.

Tableau 4.4 Résultats de mise en correspondance des nœuds à partir de paires de graphes grâce au modèle siamois après un surentraînement de 10 000 époques.

Paire de graphes	Ensemble	Top-1	Top-5	Position moyenne
#1	Entraînement	13/23 (56.52%)	20/23 (86.95%)	3.3/67 (96.57%)
#2	Entraînement	6/18 (33.33%)	14/18 (77.77%)	3.4/56 (95.71%)
#3	Validation	0/13 (0%)	3/13 (23.07%)	12.6/55 (78.91%)

CHAPITRE 5 DISCUSSION GÉNÉRALE

Ce chapitre présente des réflexions par rapport aux résultats obtenus et met en lumière les limitations des différentes méthodes employées pour les étapes de synchronisation de séquences angiographiques et de détection de correspondances spatiales.

5.1 Synchronisation des séquences angiographiques

Lors de la sélection des images à garder pour l'entraînement basée sur le gradient global moyen, seulement les images dont la valeur est supérieure à la moyenne sont gardées, mais l'impact de ce choix n'a pas été mesuré. Il serait pertinent de vérifier si l'utilisation d'images où les artères ne sont pas visibles pour l'entraînement permettrait d'améliorer les résultats.

Lors de la création des données *ground truth*, l'algorithme de détection automatique des ondes R dans les ECG a permis d'identifier des points clés dans les séquences. Les résultats de l'algorithme ont été manuellement inspectés pour s'assurer de son bon fonctionnement. Les valeurs des paramètres utilisés dans l'algorithme de validation des détections automatiques faisant usage de l'écart-type entre les pics ont été choisies de manière empirique en s'assurant que celles-ci permettaient de repérer tous les cas où des pics étaient manquants tout en laissant passer les cas ne présentant aucune erreur de détection.

Les réseaux de neurones récurrents (RNN) sont bien adaptés aux données temporelles et un modèle de ce genre aurait pu être utilisé pour résoudre la tâche de synchronisation des séquences angiographiques. Par contre, les RNN sont plus complexes à créer et à entraîner et il n'existe pas de modèle pré-entraîné qui aurait pu être utilisé pour faciliter l'apprentissage sur une quantité limitée de données. C'est donc pour ces raisons qu'un CNN pré-entraîné a été choisi à la place.

Tel que présenté dans la section de l'analyse des résultats de synchronisation, le score obtenu par le modèle entraîné sur les données de patients pédiatriques est relativement bas lorsque testé sur les données de patients adultes. Ceci indique donc que le modèle a de la difficulté à généraliser sur des données ne provenant pas du même jeu de données que pour l'entraînement. Il est probable que l'importante différence dans la distribution des pixels des images des deux jeux de données explique la mauvaise performance du modèle. En effet, les images obtenues avec des patients pédiatriques ont tendance à afficher les vaisseaux, organes et cage thoracique de manière beaucoup plus nette que celles obtenues avec des patients adultes, qui ont tendance à être plus floues en raison de la plus forte présence de tissus adipeux.

Bien que le score de synchronisation obtenu par les modèles entraînés sur les jeux de données sont relativement haut (0.9604 pour les patients pédiatriques seulement et 0.9424 pour les patients pédiatriques et adultes), il est probable que ce résultat serait encore meilleur si les données *ground truth* faites à partir des ECG étaient plus précises en incluant une détection de la fin de la phase systolique. En regardant la vidéo présentée dans la description de la figure 4.1, on peut voir que les paires de séquences avec un score faible ont l'air assez bien synchronisées. Et tel que montré par un exemple à la figure 4.3, la plupart des scores inférieurs à la moyenne sont dus à des différences de synchronisation entre les ondes R du *ground truth*. Par contre, ajouter une détection automatique de la fin de la phase systolique est très complexe, puisque la courbe de l'onde T n'est pas aussi abrupte que l'onde R. De plus, il est fréquent que l'onde T change d'apparence puisque sa forme est affectée par certaines maladies cardiaques.

Par rapport au modèle choisi, le tableau 4.1 montre que celui-ci a peu d'impact sur le score final. Il semble toujours y avoir une combinaison d'hyperparamètres pour chaque type de modèle qui leur permette d'obtenir un score similaire au meilleur. Le meilleur choix de modèle serait donc possiblement lié au contexte d'utilisation, par exemple si les performances des composantes de l'ordinateur sur lequel le modèle est exécuté sont plutôt limitées, il serait peut-être avantageux d'utiliser un modèle plus léger, tel que le MobileNet.

5.2 Détection de correspondances spatiales

5.2.1 Squelettisation

Pour une image de taille 1024x1024, l'algorithme de squelettisation implémenté dans la librairie "scikit-image" prend en moyenne 250 ms à s'exécuter avec un processeur i7-4790, ce qui n'est pas tout à fait suffisant pour être utilisé en temps réel. De plus, l'algorithme d'élagage des branches prend en moyenne 600 ms à s'exécuter, ce qui est beaucoup trop lent pour du temps réel.

5.2.2 Extraction du graphe orienté

Lors du développement des algorithmes, la valeur de chaque hyperparamètre a été choisie de manière empirique, ce qui n'est pas l'idéal. Il serait préférable de procéder d'une manière exhaustive pour choisir ces valeurs.

On peut aussi voir que certaines branches sont manquantes et cela est probablement causé par des faux positifs dans la détection de croisements, tel que présenté par la figure 5.1. Dans

d'autres exemples, on peut voir que certains segments sont explorés dans le mauvais sens (voir figure 5.2).

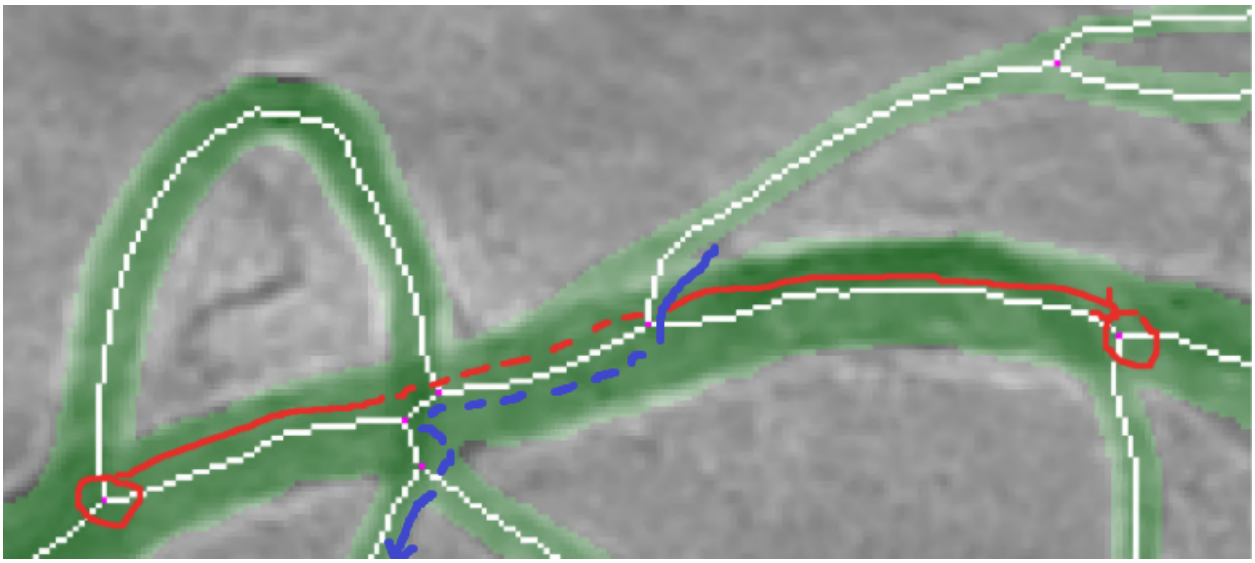


Figure 5.1 Explication d'un faux positif de détection de croisement. Les segments par lesquels passe la ligne rouge sont détectés comme étant un seul et même segment car un croisement est détecté pour la branche montant vers le haut à partir du centre de l'image alors qu'il s'agit d'une bifurcation.

En ce qui concerne la création du graphe orienté à partir du squelette, des caractéristiques additionnelles pourraient être ajoutées aux nœuds. Par exemple, il serait possible d'utiliser un CNN pour générer un vecteur de caractéristiques décrivant une région de pixels autour d'une bifurcation. En utilisant un CNN pré-entraîné, ce vecteur de caractéristiques pourrait contenir des informations potentiellement intéressantes pour le modèle de GCN siamois. Par contre, si ce CNN ne fait pas partie du modèle, il ne pourra pas s'améliorer pendant l'entraînement pour générer des vecteurs de caractéristiques contenant des informations plus pertinentes. Intégrer celui-ci dans le GCN siamois serait probablement mieux, mais aussi plus complexe à implémenter et entraîner. Une autre caractéristique qui serait bon à considérer est l'angle de vue de l'image angiographique. Celle-ci pourrait être simple à ajouter directement au vecteur de caractéristiques des nœuds.

5.2.3 Génération de données *ground truth* pour les paires de nœuds

Puisque l'entraînement du modèle siamois ne peut pas être fait sans un jeu de données étoffé, la prochaine étape est de générer plus de données *ground truth* pour les paires de bifurcations. Par contre, ce n'est pas tout de créer ces paires de bifurcations, on doit aussi

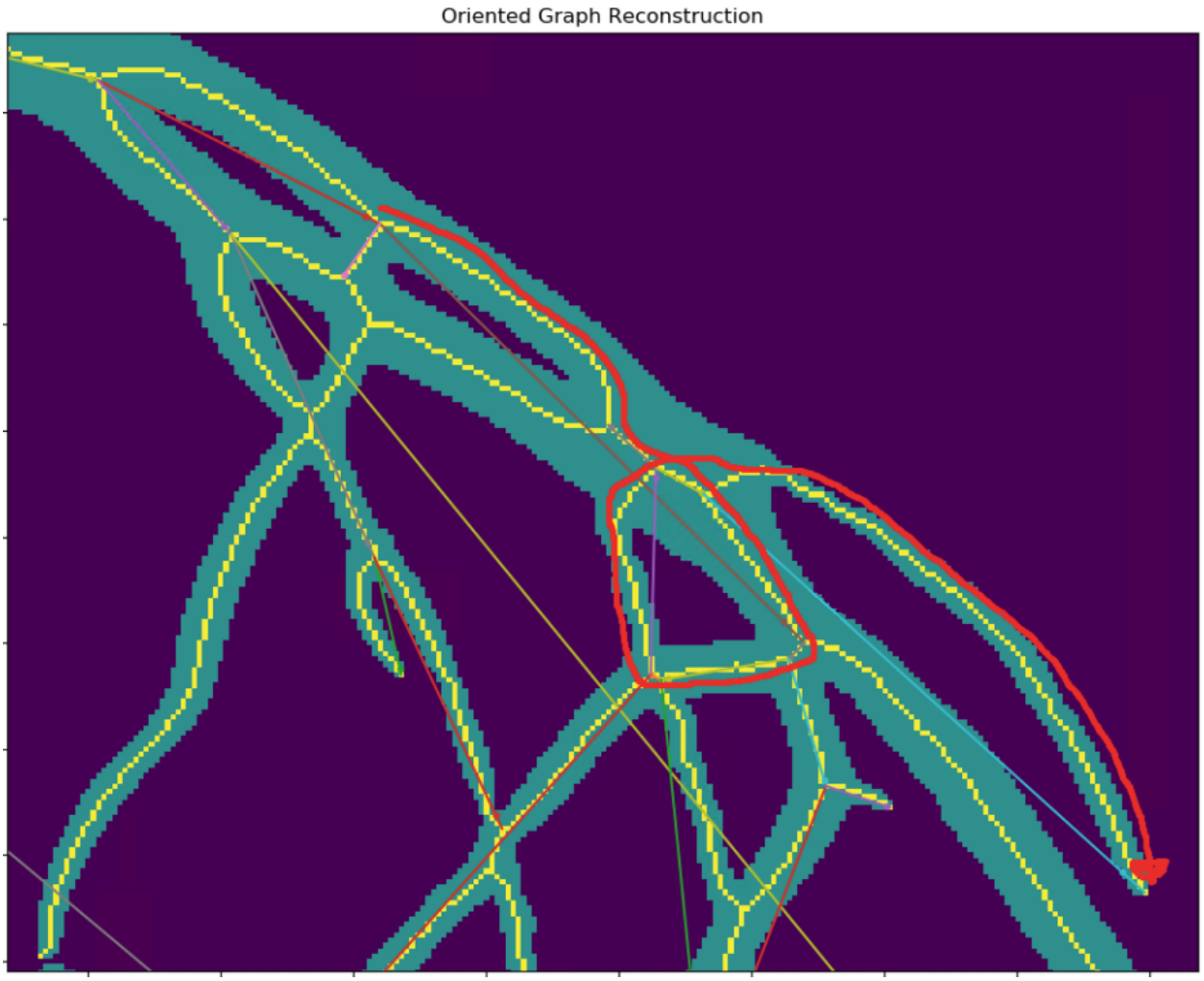


Figure 5.2 Exemple de segments du squelette explorés dans la mauvaise direction. Les ronds au bout des lignes droites colorées indiquent la direction d'exploration des segments. Le trait rouge indique le chemin exploré par l'algorithme d'extraction de graphe.

avoir les segmentations pour générer les squelettes avec lesquels on peut créer les graphes orientés pour l'entraînement. Afin de permettre la création de paires de bifurcations, ces segmentations doivent provenir d'images non seulement du même patient, mais aussi de la même artère (gauche ou droite). Le nombre de combinaisons possibles augmente de manière exponentielle par rapport au nombre d'images segmentées de la même artère. Il serait donc préférable de segmenter plusieurs images d'un même patient plutôt que de segmenter des images d'un plus grand nombre de patients. Il pourrait aussi être intéressant de segmenter des images d'une même séquence afin de voir si le réseau a plus de facilité à générer des vecteurs de caractéristiques similaires pour les mêmes bifurcations d'un même angle de vue.

5.2.4 Analyse des résultats du GCN siamois

Bien que le modèle siamois performe beaucoup mieux qu'un modèle aléatoire, ses performances pourraient être meilleures. Seulement un faible pourcentage des paires positives sont classées au premier rang dans les listes de distances entre les nœuds, ce qui n'est pas suffisant pour un usage clinique. D'ailleurs, la très faible quantité de données disponibles pour l'entraînement et la validation empêche l'évaluation fidèle du modèle, ce qui ne permet pas vraiment de tirer de conclusions.

Lorsque l'on s'attarde aux valeurs des vecteurs de caractéristiques 2D générés par le modèle pour chaque nœud, on peut voir qu'une des deux caractéristiques semble dépendre de l'autre, ce qui la rend plutôt inutile (voir figure 5.3). Il est difficile d'expliquer exactement pourquoi le modèle n'utilise pas les deux dimensions des vecteurs de caractéristiques, mais il est probable que la fonction de perte soit en cause et qu'une autre mieux adaptée pourrait l'encourager à exploiter pleinement les deux dimensions du vecteur de caractéristiques en sortie.

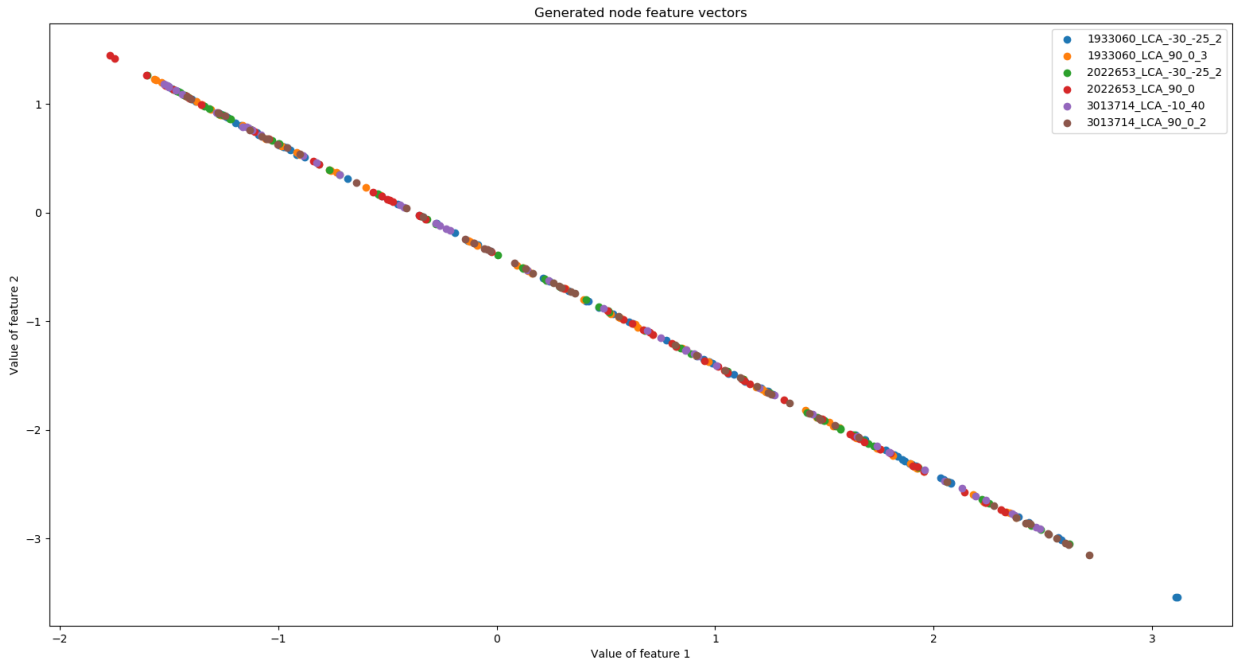


Figure 5.3 Vecteurs de caractéristiques des nœuds générés par le GCN siamois après l'entraînement.

Quelques améliorations pourraient être amenées à la représentation du squelette en graphe orienté ainsi qu'à l'architechture du réseau afin de lui permettre d'obtenir de meilleurs résultats. Par exemple, puisque le modèle propage les caractéristiques des nœuds seulement dans la direction des liens du graphe orienté, celui-ci n'a pas accès à certaines informations, telles

que la proximité du nœud courant à la fin d'une branche. Ce genre d'information pourrait l'aider à identifier les paires de nœuds se trouvant loin de l'ostium.

Pour ce qui est de l'architecture du GCN siamois, celle-ci ne propage que les vecteurs de caractéristiques des nœuds et ignore ceux des liens. Il est donc probable que de meilleurs résultats pourraient être obtenus si le modèle pouvait utiliser les informations des liens lorsqu'il génère un vecteur de caractéristiques pour un nœud.

Une fois que plus de données sont préparées et que le modèle possède le bon type d'opération de convolution, il serait intéressant de tester plusieurs choses lors des entraînements. Par exemple, différentes quantités de couches de convolution pourraient être comparées afin de voir combien de sauts les informations des nœuds doivent faire pour obtenir les meilleurs résultats. Aussi, il serait pertinent de comparer les résultats des graphes orientés avec ceux des graphes non-orientés pour déterminer s'il est vraiment nécessaire d'extraire seulement les nœuds qui sont de vraies bifurcations.

CHAPITRE 6 CONCLUSION

L'objectif pricipal de ce projet de recherche étant la mise en correspondance spatio-temporelle des séquences angiographiques, les contributions de ce travail visent toutes à remplir celui-ci.

Par rapport à la tâche de mise en correspondance temporelle, la création d'une banque de données d'entraînement à partir de séquences angiographiques multi-vues de patients pédiatriques et adultes a permis l'entraînement d'un modèle de synchronisation des séquences basé sur l'apprentissage profond et exploitant le transfert d'apprentissage afin d'obtenir une synchronisation précise à partir d'une quantité modeste de données d'entraînement. L'identification automatique des ondes R dans les ECG a été l'élément clé pour la définition du *ground truth*. De plus, la conception d'un score de synchronisation basé sur la distance de la progression du cycle cardiaque de deux séquences, couplée à la comparaison de différents modèles de réseaux de neurones convolutifs pré-entraînés puis *fine-tuned*, a permis de montrer l'efficacité de la méthode pour résoudre la tâche de synchronisation, peu importe l'architechture du modèle choisi. Finalement, l'évaluation quantitative et l'analyse des performances des modèles entraînés a montré que le modèle arrive généralement à synchroniser les séquences encore mieux que les ECG se basant seulement sur les ondes R.

La plupart des techniques de reconstruction 3D présentées dans la revue de littérature utilisent simplement la contrainte épipolaire pour obtenir des mises en correspondances, ce qui n'est pas suffisant pour assurer une reconstruction fiable puisque cette méthode ne permet pas de lever les ambiguïtés amenées par la superposition des vaisseaux. Pour répondre à ce problème, ce travail a présenté une preuve de concept pour la mise en correspondances des nœuds d'une paire de graphes orientés à l'aide d'un réseau de neurones par graphe. La génération du graphe orienté de l'arbre coronarien à partir d'une segmentation d'image angiographique coronarienne en analysant le squelette des vaisseaux a rendu possible l'extraction automatique d'information pertinente à l'entraînement du modèle de réseau de neurones par graphe. Ensuite, le développement d'un outil d'identification de paires de bifurcations a permis la construction d'une très modeste banque de données, rendant possible la création de la preuve de concept à l'aide d'un réseau de neurones siamois par graphe. Finalement, l'analyse des résultats obtenus grâce à ce type de modèle a montré que bien que plusieurs améliorations seront nécessaire pour atteindre une performance suffisante pour un usage clinique, ce genre de modèle peut apprendre à effectuer la mise en correspondance des segments d'artères coronaires.

Puisque les méthodes explorées sont basées sur des réseaux de neurones, leurs performances

en terme de temps de calcul lors de l'inférence sont très bonnes, ce qui pourra permettre une reconstruction 3D+t pendant l'acquisition des séquences.

6.1 Recommandations

Dans le chapitre précédent, plusieurs limitations des algorithmes proposés ont été énoncées. Des recommandations pour pallier ces problèmes sont listées dans cette section.

En ce qui concerne la difficulté du modèle de synchronisation à généraliser, il faudrait entraîner et tester le modèle sur davantage de jeux de données provenant de différentes sources, tels que des centres hospitaliers qui utiliseraient des C-arms de modèle différent. Une plus large sélection de séquences angiographiques d'apparence différente permettrait certainement de mieux généraliser.

Quant à la vitesse d'exécution de l'algorithme de squelettisation n'étant pas tout à fait suffisante pour une utilisation temps réel, une réduction de la taille des images aiderait certainement à rendre la méthode suffisamment rapide. Et pour l'algorithme d'élagage, optimiser l'algorithme en le parallélisant et en choisissant mieux les branches à considérer lors des nouvelles itérations permettrait de rendre celui-ci très rapide.

Au sujet des hyperparamètres des algorithmes développés de manière ad hoc pour l'extraction de graphe orienté, maintenant que les métriques de distance pour la détection d'ostium et de pourcentage de branches gardées pour l'extraction du graphe orienté sont disponibles, il serait préférable d'effectuer une recherche d'hyperparamètres, similaire à ce qui a été fait pour l'optimisation du modèle de synchronisation, pour s'assurer que les algorithmes soient les mieux adaptés possible aux données disponibles.

À propos de l'algorithme de détection de croisements, il faudrait améliorer la logique de détection pour prévenir les faux positifs, afin d'obtenir des graphes orientés de meilleure qualité, où moins de branches sont manquantes. De plus, il serait possible de prévenir le parcours des branches dans le mauvais sens en améliorant l'ordre d'exploration des branches. Pour l'instant, ce sont les segments les plus larges qui sont explorés en premier, mais la logique d'ordre d'exploration pourrait aussi considérer la direction des branches afin de décourager l'exploration des segments dont la direction s'approche de l'ostium.

Finalement, pour ce qui est du GCN siamois, il serait important de générer plus de segmentations d'artères et de paires positives de bifurcations. Cela permettrait de mieux entraîner et évaluer le GCN siamois. D'ailleurs, il pourrait être utile d'implémenter l'opération de "EdgeConv". La librairie PyTorch geometric offre déjà une implémentation de celle-ci, mais elle n'utilise malheureusement que les vecteurs de caractéristiques des nœuds. Il faudrait donc la

modifier pour utiliser aussi les caractéristiques des liens comme il se doit.

RÉFÉRENCES

[1] H. DeSena, B. Landis, R. Moore, D. Spar, W. Whiteside et B. Blaxall, “Future Diagnostic Strategies—Pediatric,” dans *Cardioskeletal Myopathies in Children and Young Adults*. Elsevier, 2017, p. 361–379. [En ligne]. Disponible : https://linkinghub.elsevier.com/retrieve/pii/B9780128000403000182

[2] M. Plourde, “Reconstruction 3D+T des artères coronaires à partir d’une séquence rotationnelle angiographique,” Thèse de doctorat, École des Technologies Supérieure, Montréal, Canada, mars 2013.

[3] M. Unberath, O. Taubmann, A. Aichert, S. Achenbach et A. Maier, “Prior-Free Respiratory Motion Estimation in Rotational Angiography,” *IEEE Transactions on Medical Imaging*, vol. 37, n°. 9, p. 1999–2009, sept. 2018. [En ligne]. Disponible : https://ieeexplore.ieee.org/document/8291503/

[4] K. Ananthasubramaniam, N. Sareen et G. Rudin, “Advances in Cardiac Computed Tomography,” dans *Computed Tomography - Advanced Applications*, A. M. Halefoglu, édit. InTech, août 2017. [En ligne]. Disponible : http://www.intechopen.com/books/computed-tomography-advanced-applications/advances-in-cardiac-computed-tomography

[5] J. A. Garcia, “Three-Dimensional Imaging for Coronary Interventions,” p. 6, 2013. [En ligne]. Disponible : https://citoday.com/articles/2013-jan-feb/three-dimensional-imaging-for-coronary-interventions

[6] S. Çimen, A. Gooya, M. Grass et A. F. Frangi, “Reconstruction of coronary arteries from X-ray angiography : A review,” *Medical Image Analysis*, vol. 32, p. 46–68, août 2016. [En ligne]. Disponible : https://linkinghub.elsevier.com/retrieve/pii/S1361841516000220

[7] S. Habert, “Reconstruction 3D des artères coronaires à partir d’images angiographiques pour la visualisation en ligne des anévrismes,” Thèse de doctorat, Polytechnique Montréal, Montréal, Canada, févr. 2013.

[8] S. Mahmoodzadeh, M. Moazenzadeh, H. Rashidinejad et M. Sheikhvatan, “Diagnostic performance of electrocardiography in the assessment of significant coronary artery disease and its anatomical size in comparison with coronary angiography,” *Journal of Research in Medical Sciences : The Official Journal of Isfahan University of Medical Sciences*, vol. 16, n°. 6, p. 750–755, juin 2011. [En ligne]. Disponible : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3214392/

[9] A. R. Pérez-Riera, R. Barbosa-Barros, R. Daminello-Raimundo et L. C. de Abreu, “Main artifacts in electrocardiography,” *Annals of Noninvasive Electrocardiology : The Official Journal of the International Society for Holter and Noninvasive Electrocardiology, Inc*, vol. 23, n°. 2, sept. 2017. [En ligne]. Disponible : https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6931710/

[10] A. Whitehead, R. Laganiere et P. Bose, “Temporal Synchronization of Video Sequences in Theory and in Practice,” dans *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION’05) - Volume 1*, vol. 2, janv. 2005, p. 132–137.

[11] Rao, Gritai, Shah et Syeda-Mahmood, “View-invariant alignment and matching of video sequences,” dans *Proceedings Ninth IEEE International Conference on Computer Vision*, oct. 2003, p. 939–945 vol.2.

[12] S. Song, C. Du, X. Liu, Y. Huang, H. Song, Y. Jiang, D. Ai, A. F. Frangi, Y. Wang et J. Yang, “Deep motion tracking from multiview angiographic image sequences for synchronization of cardiac phases,” *Physics in Medicine and Biology*, vol. 64, n°. 2, p. 025018, 2019.

[13] H. Sundar, A. Khamene, L. Yatziv et C. Xu, “Automatic Image-Based Cardiac and Respiratory Cycle Synchronization and Gating of Image Sequences,” dans *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009*, ser. Lecture Notes in Computer Science, G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble et C. Taylor, édit. Berlin, Heidelberg : Springer, 2009, p. 381–388.

[14] M. Panayiotou, A. P. King, R. J. Housden, Y. Ma, M. Cooklin, M. O. Neill, J. Gill, C. A. Rinaldi et K. S. Rhode, “A statistical method for retrospective cardiac and respiratory motion gating of interventional cardiac x-ray images,” *Medical Physics*, vol. 41, n°. 7, p. 071901, 2014, _eprint : https ://aapm.onlinelibrary.wiley.com/doi/pdf/10.1118/1.4881140. [En ligne]. Disponible : https://aapm.onlinelibrary.wiley.com/doi/abs/10.1118/1.4881140

[15] P. Wieschollek, I. Freeman et H. P. A. Lensch, “Learning Robust Video Synchronization without Annotations,” *arXiv :1610.05985 [cs]*, sept. 2017, arXiv : 1610.05985. [En ligne]. Disponible : http://arxiv.org/abs/1610.05985

[16] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet et A. Zisserman, “Temporal Cycle-Consistency Learning,” *arXiv :1904.07846 [cs]*, avr. 2019, arXiv : 1904.07846. [En ligne]. Disponible : http://arxiv.org/abs/1904.07846

[17] C. Ciusdel, A. Turcea, A. Puiu, L. Itu, L. Calmac, E. Weiss, C. Margineanu, E. Badila, M. Berger, T. Redel, T. Passerini, M. Gulsun et P. Sharma, “Deep Neural Networks for ECG-free Cardiac Phase and End-Diastolic Frame Detection on Coronary

Angiographies," *arXiv :1811.02797 [cs]*, nov. 2018, arXiv : 1811.02797. [En ligne]. Disponible : http://arxiv.org/abs/1811.02797

[18] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, n°. 2, p. 91–110, nov. 2004. [En ligne]. Disponible : https://doi.org/10.1023/B:VISI.0000029664.99615.94

[19] H. Bay, T. Tuytelaars et L. Van Gool, "SURF : Speeded Up Robust Features," dans *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof et A. Pinz, édit. Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, vol. 3951, p. 404–417, series Title : Lecture Notes in Computer Science. [En ligne]. Disponible : http://link.springer.com/10.1007/11744023_32

[20] E. Tola, P. Fua et V. Lepetit, "A fast local descriptor for dense matching," dans *In : IEEE Conf. Comput. Vision Pattern Recog*, 2008, p. 1–8.

[21] D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, M. Calonder, V. Lepetit, C. Strecha et P. Fua, "BRIEF : Binary Robust Independent Elementary Features," dans *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos et N. Paragios, édit. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, vol. 6314, p. 778–792, series Title : Lecture Notes in Computer Science. [En ligne]. Disponible : http://link.springer.com/10.1007/978-3-642-15561-1_56

[22] J. Liu et G. Zeng, "Low-Dimensional Local Self-Similarity Descriptor for Image Matching," dans *Advances in Automation and Robotics, Vol. 2*, ser. Lecture Notes in Electrical Engineering, G. Lee, édit. Berlin, Heidelberg : Springer, 2012, p. 291–300.

[23] J. Revaud, P. Weinzaepfel, Z. Harchaoui et C. Schmid, "DeepMatching : Hierarchical Deformable Dense Matching," *arXiv :1506.07656 [cs]*, oct. 2015, arXiv : 1506.07656. [En ligne]. Disponible : http://arxiv.org/abs/1506.07656

[24] C. B. Choy, J. Gwak, S. Savarese et M. Chandraker, "Universal Correspondence Network," *arXiv :1606.03558 [cs]*, oct. 2016, arXiv : 1606.03558. [En ligne]. Disponible : http://arxiv.org/abs/1606.03558

[25] G. Long, L. Kneip, J. M. Alvarez et H. Li, "Learning Image Matching by Simply Watching Video," *arXiv :1603.06041 [cs]*, mars 2016, arXiv : 1603.06041. [En ligne]. Disponible : http://arxiv.org/abs/1603.06041

[26] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla et J. Sivic, "Neighbourhood Consensus Networks," *arXiv :1810.10510 [cs]*, nov. 2018, arXiv : 1810.10510. [En ligne]. Disponible : http://arxiv.org/abs/1810.10510

[27] M. A. Fischler et R. C. Bolles, “Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, n°. 6, p. 381–395, juin 1981. [En ligne]. Disponible : https://doi.org/10.1145/358669.358692

[28] A. Krizhevsky, I. Sutskever et G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” dans *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou et K. Q. Weinberger, édit. Curran Associates, Inc., 2012, p. 1097–1105. [En ligne]. Disponible : http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[29] Y. Lecun, L. Bottou, Y. Bengio et P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, n°. 11, p. 2278–2324, nov. 1998. [En ligne]. Disponible : http://ieeexplore.ieee.org/document/726791/

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai et S. Chintala, “PyTorch : An Imperative Style, High-Performance Deep Learning Library,” dans *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox et R. Garnett, édit. Curran Associates, Inc., 2019, p. 8026–8037. [En ligne]. Disponible : http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[31] K. He, X. Zhang, S. Ren et J. Sun, “Deep Residual Learning for Image Recognition,” *arXiv :1512.03385 [cs]*, déc. 2015, arXiv : 1512.03385. [En ligne]. Disponible : http://arxiv.org/abs/1512.03385

[32] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto et H. Adam, “MobileNets : Efficient Convolutional Neural Networks for Mobile Vision Applications,” *arXiv :1704.04861 [cs]*, avr. 2017, arXiv : 1704.04861. [En ligne]. Disponible : http://arxiv.org/abs/1704.04861

[33] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng et T. Darrell, “DeCAF : A Deep Convolutional Activation Feature for Generic Visual Recognition,” *arXiv :1310.1531 [cs]*, oct. 2013, arXiv : 1310.1531. [En ligne]. Disponible : http://arxiv.org/abs/1310.1531

[34] Y. Bengio, P. Simard et P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Transactions on Neural Networks*, vol. 5, n°. 2, p. 157–166, mars 1994. [En ligne]. Disponible : https://ieeexplore.ieee.org/document/279181/

[35] M. Defferrard, X. Bresson et P. Vandergheynst, “Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering,” *arXiv :1606.09375 [cs, stat]*, févr. 2017, arXiv : 1606.09375. [En ligne]. Disponible : http://arxiv.org/abs/1606.09375

[36] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang et P. S. Yu, “A Comprehensive Survey on Graph Neural Networks,” *IEEE Transactions on Neural Networks and Learning Systems*, p. 1–21, 2020, arXiv : 1901.00596. [En ligne]. Disponible : http://arxiv.org/abs/1901.00596

[37] M. Niepert, M. Ahmed et K. Kutzkov, “Learning Convolutional Neural Networks for Graphs,” *arXiv :1605.05273 [cs, stat]*, juin 2016, arXiv : 1605.05273. [En ligne]. Disponible : http://arxiv.org/abs/1605.05273

[38] J. Atwood et D. Towsley, “Diffusion-Convolutional Neural Networks,” *arXiv :1511.02136 [cs]*, juill. 2016, arXiv : 1511.02136. [En ligne]. Disponible : http://arxiv.org/abs/1511.02136

[39] C. R. Qi, H. Su, K. Mo et L. J. Guibas, “PointNet : Deep Learning on Point Sets for 3D Classification and Segmentation,” *arXiv :1612.00593 [cs]*, avr. 2017, arXiv : 1612.00593. [En ligne]. Disponible : http://arxiv.org/abs/1612.00593

[40] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein et J. M. Solomon, “Dynamic Graph CNN for Learning on Point Clouds,” *arXiv :1801.07829 [cs]*, juin 2019, arXiv : 1801.07829. [En ligne]. Disponible : http://arxiv.org/abs/1801.07829

[41] S. Y. Shin, S. Lee, I. D. Yun et K. M. Lee, “Deep Vessel Segmentation By Learning Graphical Connectivity,” *arXiv :1806.02279 [cs]*, juin 2018, arXiv : 1806.02279. [En ligne]. Disponible : http://arxiv.org/abs/1806.02279

[42] H. T. Trung, N. T. Toan, T. V. Vinh, H. T. Dat, D. C. Thang, N. Q. V. Hung et A. Sattar, “A comparative study on network alignment techniques,” *Expert Systems with Applications*, vol. 140, p. 112883, févr. 2020. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0957417419305937

[43] O. Ronneberger, P. Fischer et T. Brox, “U-Net : Convolutional Networks for Biomedical Image Segmentation,” *arXiv :1505.04597 [cs]*, mai 2015, arXiv : 1505.04597. [En ligne]. Disponible : http://arxiv.org/abs/1505.04597

[44] F. Girard, C. Kavalec et F. Cheriet, “Joint segmentation and classification of retinal arteries/veins from fundus images,” *Artificial Intelligence in Medicine*, vol. 94, p. 96–109, mars 2019, arXiv : 1903.01330. [En ligne]. Disponible : http://arxiv.org/abs/1903.01330

[45] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers et L. Van Gool, “One-Shot Video Object Segmentation,” *arXiv :1611.05198 [cs]*, avr. 2017, arXiv : 1611.05198. [En ligne]. Disponible : http://arxiv.org/abs/1611.05198

[46] Z. Lai et W. Xie, “Self-supervised Learning for Video Correspondence Flow,” *arXiv :1905.00875 [cs]*, juill. 2019, arXiv : 1905.00875. [En ligne]. Disponible : http://arxiv.org/abs/1905.00875

[47] S. Xia, H. Zhu, X. Liu, M. Gong, X. Huang, L. Xu, H. Zhang et J. Guo, “Vessel Segmentation of X-Ray Coronary Angiographic Image Sequence,” *IEEE Transactions on Biomedical Engineering*, vol. 67, n°. 5, p. 1338–1348, mai 2020, conference Name : IEEE Transactions on Biomedical Engineering.

[48] M. Benovoy, “Automatic spatiotemporal analysis of cardiac image series,” Thèse de doctorat, Polytechnique Montréal, 2016. [En ligne]. Disponible : https://publications.polymtl.ca/2208/1/2016_MitchelBenovoy.pdf

[49] P. Sasha, G. Borgefors et G. Sanniti di Baja, “A survey on skeletonization algorithms and their applications,” *Pattern Recognition Letters*, vol. 76, avr. 2015. [En ligne]. Disponible : https://reader.elsevier.com/reader/sd/pii/S0167865515001233

[50] T. Y. Zhang et C. Y. Suen, “A fast parallel algorithm for thinning digital patterns,” *Communications of the ACM*, vol. 27, n°. 3, p. 236–239, mars 1984. [En ligne]. Disponible : https://doi.org/10.1145/357994.358023

[51] T. C. Lee, R. L. Kashyap et C. N. Chu, “Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms,” *CVGIP : Graphical Models and Image Processing*, vol. 56, n°. 6, p. 462–478, nov. 1994. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S104996528471042X

[52] G. Hu et Z.-N. Li, “An x-crossing preserving skeletonization algorithm,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 07, n°. 05, p. 1031–1053, oct. 1993, publisher : World Scientific Publishing Co. [En ligne]. Disponible : https://www.worldscientific.com/doi/abs/10.1142/S0218001493000522

[53] M. E. Martínez-Pérez, A. D. Hughes, A. V. Stanton, S. A. Thom, N. Chapman, A. A. Bharath et K. H. Parker, “Geometrical and Morphological Analysis of Vascular Branches from Fundus Retinal Images,” dans *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2000*, ser. Lecture Notes in Computer Science, S. L. Delp, A. M. DiGoia et B. Jaramaz, édit. Berlin, Heidelberg : Springer, 2000, p. 756–765.