

**Wrocław University of Science and Technology**  
**Faculty of Information and Communication Technology**

---

Field of study: **SZT**

Speciality: -

**MASTER THESIS**

**Estimation of FFR in coronary arteries with  
deep learning**

Patryk Rygiel

Supervisor  
**dr hab. inż. Maciej Zięba**

deep learning, geometric deep learning, point clouds, CAD, FFR, hemodynamics



## Abstract

Coronary artery disease (CAD) is one of the most common causes of death in the European Union and the USA. The crucial biomarker in its diagnosis is called Fractional Flow Reserve (FFR) and its in-vivo measurement is obtained via an invasive diagnostic technique in the form of coronary angiography. In order to address the invasive drawbacks associated with a procedure, a new approach virtual FFR (vFFR) measurement has emerged in recent years. This technique involves using computed tomography angiography (CTA) to obtain virtual measurements of FFR. By utilizing Computational Fluid Dynamics (CFD), vFFR estimates can be derived from CTA data, providing a promising in-silico alternative to traditional methods. However, the widespread adoption of vFFR from CTA as a diagnostic technique is hindered by two main challenges: time and computational requirements. In this work, we explore the usage of deep learning techniques as surrogate CFD engine models in the task of vFFR estimation in coronary arteries to drastically limit the required time and computational costs without a major drop in quality. We propose a novel approach to vFFR estimation by representing the input vessel geometry as a point cloud and utilizing the hybrid neural network that learns geometry representation based on both explicitly and implicitly given features. We evaluate the method from the clinical point of view and showcase that it can serve as a compelling replacement for commonly utilized CFD-based approaches.

## Streszczenie

Choroba niedokrwienienna serca (CAD) jest jedną z najczęstszych przyczyn zgonów w Unii Europejskiej i USA. Kluczowym biomarkierem w jej diagnozowaniu jest tzw. Wskaźnik Rezerwy Przepływu Wieńcowego (FFR), a jego pomiar in vivo uzyskuje się za pomocą inwazyjnej techniki diagnostycznej w postaci koronarografii. Aby poradzić sobie z wadami związanymi z inwazyjnym zabiegiem, w ostatnich latach pojawiło się nowe podejście, jakim jest pomiar wirtualnego FFR (vFFR). Ta technika polega na wykorzystaniu Komputerowej Mechaniki Płynów (CFD), w celu pozyskania wartości vFFR na podstawie danych z tomografii komputerowej (CTA), co stanowi obiecującą in-silico alternatywę dla tradycyjnych metod. Jednak powszechnie stosowanie vFFR z CTA jako techniki diagnostycznej napotyka dwa główne wyzwania: złożoność czasową i obliczeniową. W niniejszej pracy badamy zastosowanie technik głębokiego uczenia jako surogatu silnika CFD w zadaniu estymacji vFFR w tętnicach wieńcowych, aby znacznie ograniczyć wymagany czas i koszty obliczeniowe bez większego spadku w jakości. Proponujemy nowatorskie podejście do szacowania vFFR, polegające na reprezentowaniu geometrii naczynia wejściowego jako chmury punktów i wykorzystaniu hybrydowej sieci neuronowej, która uczy się reprezentacji geometrii na podstawie cech podanych wprost jak i wyuczonych nie wprost. Oceniamy tę metodę z punktu widzenia klinicznego i pokazujemy, że może ona stanowić przekonującą alternatywę dla powszechnie stosowanych podejść opartych na CFD.



# Contents

<b>List of figures</b>	<b>III</b>
<b>Introduction</b>	<b>1</b>
<b>Acknowledgments</b>	<b>3</b>
<b>1 Problem background</b>	<b>5</b>
1.1 Coronarography-based FFR estimation . . . . .	5
1.2 Virtual FFR estimation . . . . .	6
1.2.1 CFD-based vFFR estimation . . . . .	6
1.2.2 AI-based vFFR estimation . . . . .	8
1.3 Summary . . . . .	8
<b>2 Related work</b>	<b>9</b>
2.1 Learning on explicit hand-crafted features . . . . .	9
2.1.1 Siemens framework (Itu et al.) . . . . .	9
2.1.2 DEEPVESSEL-FFR (Wang et al.) . . . . .	11
2.1.3 Fossan et al. . . . .	11
2.1.4 Summary . . . . .	11
2.2 Learning on implicit features . . . . .	12
2.2.1 Li et al. . . . .	13
2.2.2 Suk et al. . . . .	13
2.2.3 Summary . . . . .	14
2.3 Summary . . . . .	14
<b>3 Learning on 3D shapes</b>	<b>15</b>
3.1 Representations of data in 3D . . . . .	15
3.1.1 Voxel grid . . . . .	15
3.1.2 Mesh . . . . .	16
3.1.3 Point cloud . . . . .	16
3.2 Geometric deep learning . . . . .	17
3.2.1 Symmetry . . . . .	17
3.2.2 Deformation stability . . . . .	19
3.2.3 Scale separation . . . . .	19
3.2.4 The Geometric Deep Learning blueprint . . . . .	20
3.3 Learning on 3D point clouds . . . . .	21
3.3.1 PointNet . . . . .	22
3.3.2 PointNet++ . . . . .	23
3.3.3 Domain-specific adaptations . . . . .	25
3.4 Summary . . . . .	26
<b>4 Proposed approach to the estimation of FFR in coronary arteries</b>	<b>27</b>
4.1 Proposed approach & workflow . . . . .	27
4.2 Data preparation . . . . .	28

4.2.1	Generation of synthetic coronary artery geometries . . . . .	28
4.2.2	CFD simulations . . . . .	29
4.3	Estimation of vFFR with point cloud based neural networks . . . . .	31
4.3.1	Point cloud construction . . . . .	31
4.3.2	Architectures . . . . .	32
4.3.3	Training & Inference setup . . . . .	33
4.3.4	Implementation . . . . .	34
4.4	Summary . . . . .	34
<b>5</b>	<b>Experiments &amp; results</b>	<b>35</b>
5.1	Metrics . . . . .	35
5.2	Quantitative results . . . . .	35
5.2.1	Evaluation under various blood flow characteristics . . . . .	35
5.2.2	Evaluation of stenosis grade impact . . . . .	37
5.2.3	Evaluation under clinical viability . . . . .	38
5.3	Qualitative results . . . . .	38
5.4	Problems & issues . . . . .	41
5.5	Future works . . . . .	41
5.6	Summary . . . . .	41
<b>6</b>	<b>Summary</b>	<b>43</b>
<b>Bibliography</b>		<b>48</b>
<b>A</b>	<b>Additional qualitative results</b>	<b>49</b>

# List of Figures

1.1 Coronarography procedure . . . . .	6
1.2 CFD-based vFFR estimation . . . . .	7
2.1 Workflow of Siemens framework . . . . .	10
2.2 Deep learning approach by Siemens . . . . .	10
2.3 DEEPVESSEL-FFR framework . . . . .	11
2.4 Deep learning approach by Fossan et al. . . . .	12
2.5 Point cloud based approach by Li et al. . . . .	13
2.6 Mesh-based DNN architecture by Suk et al. . . . .	14
3.1 Representations of data in 3D . . . . .	15
3.2 Principles of Geometric Deep Learning . . . . .	17
3.3 The Geometric Deep Learning blueprint . . . . .	20
3.4 PointNet architecture . . . . .	22
3.5 PointNet++ architecture . . . . .	24
3.6 Point grouping strategies . . . . .	25
4.1 Workflow diagram of the proposed approach . . . . .	27
4.2 Samples from the synthetic artery generator . . . . .	29
4.3 Placement of inlets and outlets for the CFD simulation . . . . .	30
4.4 CFD simulation for various inflow values . . . . .	30
4.5 Additional hand-crafted features . . . . .	31
4.6 Grouping methods for the vFFR estimation . . . . .	33
4.7 PointNet++ architecture details . . . . .	34
5.1 Evaluation of stenosis grade impact . . . . .	37
5.2 Qualitative results of vFFR estimation. . . . .	39
5.3 Issues with vFFR estimation. . . . .	40
A.1 Qualitative results of vFFR estimation for various inflow values . . . . .	49
A.2 Additional MAE results for MSG model. . . . .	50
A.3 Additional MAE results for EVG model. . . . .	50



# Introduction

The goal of this thesis is to design, develop and implement a novel approach to the estimation of virtual Fractional Flow Reserve (vFFR) in coronary arteries with the usage of deep learning methods. The scope of this work includes discussion and analysis of approaches to the task of vFFR estimation based on Computational Fluid Dynamics (CFD) and Artificial intelligence (AI). We primarily focus on AI-based methods which allow us to drastically limit the time-consumption of CFD methods while not suffering great loss in performance. We propose a novel approach to the vFFR estimations by leveraging the point cloud representation and designing an adequate learning framework. We pair implicit feature learning performed by a point cloud based neural network with hand-crafted features to construct a hybrid approach that achieves promising results from the clinical point of view.

This work consists of 5 chapters:

**Chapter 1** highlights the problem background by introducing how the *in-vivo* FFR measurement is performed and why there is a need for in-silico measurement. The concept of virtual FFR is introduced in the light of CFD and AI-based methods.

**Chapter 2** showcases related works which previously tackled the problem of vFFR estimation with AI. We group the methods under explicit and implicit feature learning categories and discuss its advantages and disadvantages.

**Chapter 3** introduces concepts of learning on 3D shapes. Common 3D data representations are discussed, and the blueprint of Geometric Deep Learning (GDL) is introduced. Through the lens of the GDL the architectures used for learning on point clouds, which are the main focus of this work, used in this work are described.

**Chapter 4** showcases the proposed approach to the estimation of vFFR based on point cloud representation. We discuss the construction of the synthetic dataset used in this work, by describing the process of vessel geometry generation and computation of CFD simulations which serve as the ground truth labels. Utilized deep learning architectures are described together with the training and inference pipelines.

**Chapter 5** showcases our experiments and results performed to evaluate the proposed approach. We report various scores that prove the robustness, generalization and clinical viability of the proposed approach. Finally, we also discuss the limitations and future works.



# Acknowledgments

This work has been performed together with the company Hemolens Diagnostics sp. z o.o.<sup>1</sup>.

I would like to thank my supervisor dr hab. inż. Maciej Zięba for the provided support and suggestions in the design of a proposed deep learning approach, dr. Tomasz Konopczyński for providing invaluable insights and guidance during the development of this work from both medical and technical points of view, and dr. Paweł Płuszka for helping with the intricacies of CFD and physics aspects of this work.

---

<sup>1</sup><https://hemolens.eu/>



# Problem background

In this chapter, we state and describe the problem of Fractional Flow Reserve (FFR) estimation by introducing the medical background. We discuss the *in-vivo* coronary angiography procedure used to obtain FFR and common *in-silico* replacement in the form of CFD simulation. At last, we point out the common issues with the aforementioned techniques and introduce the concept of AI-based CFD engine surrogates.

## 1.1 Coronarography-based FFR estimation

The *coronarography* is an invasive medical procedure performed on the patient's coronary arteries. It is performed by bringing a catheter to the patient's heart through the radial, femoral, or brachial artery, and injecting contrast dye into the coronary arteries [22]. To control the process the patient is under continuous supervision via coronary angiography imaging (see Fig. 1.1a). The main goal of performing the procedure is an assessment of abnormalities in the patient's coronary arteries. The procedure is often paired with the angioplasty treatment in the form of stent implantation in places where the vessel lumen is highly occluded by the atherosclerotic plaques or blood clots (the process of stent implanting is showcased in Fig. 1.1b). Such areas are called *stenoses* [24] and are the main reason for Coronary Artery Disease (CAD). In stenotic areas, the blood flow is reduced and the myocardium (heart muscle) is not sufficiently supplied with blood thus oxygen delivery is impeded causing myocardial ischemia. The vessel's cross-sectional area is radically reduced and the role of the stent is to ensure proper blood flow by keeping it from narrowing or closing again [27]. Accurate assessment of the stent placement is crucial to ensure the correct and safe treatment. The main biomarker, utilized to determine whether the stenosis impedes the blood flow, such the treatment is required, is called *Fractional Flow Reserve* (FFR).

The FFR measures pressure differences across coronary artery stenosis to compare the blood flow on either side of a blockage and determine how severe the impact of the stenosis on oxygen delivery to the myocardium is [59, 11]. It is defined as the ratio of maximal coronary blood flow in a diseased artery to maximal coronary blood flow in the same artery without stenosis. In reality, the ratio is computed as pressure after (distal to) stenosis relative to the pressure before (proximal to) the stenosis:

$$FFR = \frac{p_d}{p_a}, \quad (1.1)$$

where  $p_d$  is the pressure distal to the lesion, and  $p_a$  is the pressure proximal to the lesion [59]. The resulting value of FFR is an absolute value between 0 and 1, and when interpreting measurements, higher values indicate a non-significant stenosis, whereas lower values indicate a significant one [59, 23]. According to the medical literature and clinical trials [13, 1], the cut-off of 0.8 is considered to be a practical determinant (with greater than 90% accuracy [10]) of whether the stenosis is significant or not.

During the coronarography procedure, to correctly estimate FFR, the pressure measurement needs to be performed at a steady state (no pressure fluctuation with time) during maximal coronary blood flow (*hyperemia*). According to [10], hyperemia is "*a physiologic state where coronary microvascular resistance is minimized*" and it "*is typically achieved by administration of intravenous continuous infusion (140 mcg/kg/min) or intracoronary bolus (right coronary artery 50–100 mcg, left coronary artery 100–200 mcg) adenosine*".

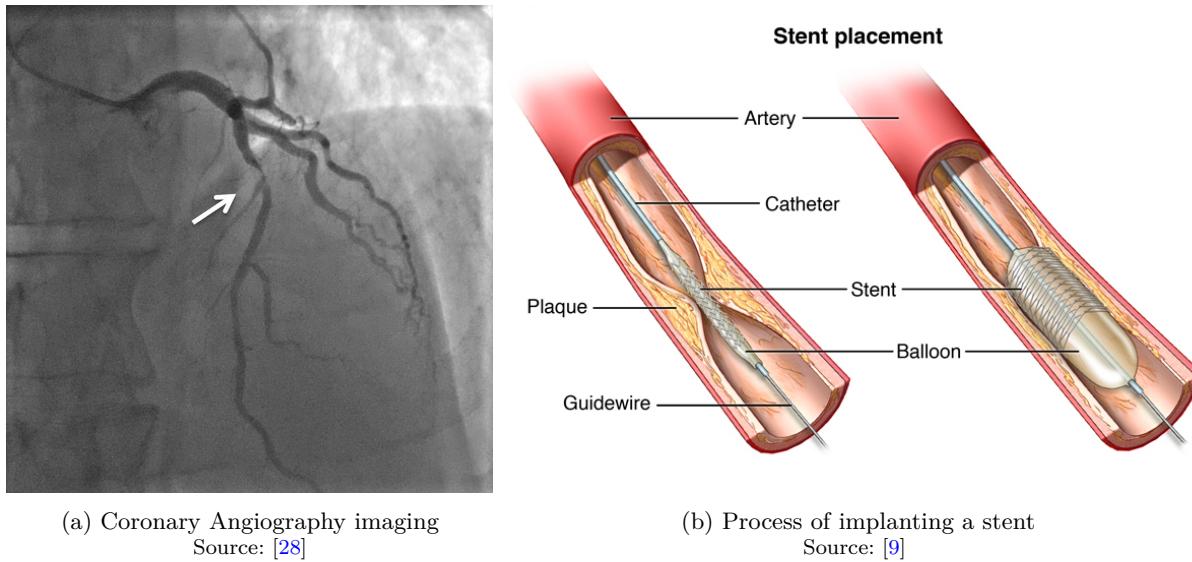


Figure 1.1: Coronarography procedure. The procedure is supervised via coronary angiography imaging and if required, the angioplasty treatment is performed by placing a stent in the stenotic region. The white arrow marks the severe stenosis seen on the coronary angiography imaging.

Even though the FFR is a minimally invasive diagnostic technique, it still poses risks for patients, especially ones with CAD. The risks include mainly radiation exposure in form of additional contrast injection (e.g. the aforementioned adenosine) and increased risk of coronary arterial dissection with catheter [15]. Additionally, since the qualification for the procedure is often based on the non-invasive medical imaging technique in the form of Computed Tomography Angiography (CTA), coronarography is rarely used as a prophylaxis but rather as a target form of treatment. Due to this fact, in recent years solutions estimating so-called virtual FFR (vFFR) based on CTA have been proposed as an alternative.

## 1.2 Virtual FFR estimation

The virtual FFR (vFFR) is an *in-silico* measurement of the FFR, that is most often based on CTA imaging and denoted in literature as  $\text{FFR}_{\text{CT}}$  [45]. It offers a compelling replacement for the classic FFR measurement by omitting its invasive aspect and allowing for wider screening of patients. The process of computing vFFR based on CTA requires a prior segmentation of coronary arteries. It can be done three-way: manually by the expert, semi-automatically with an algorithm of choice under expert continuous supervision or fully automatically (most often with the usage of AI-based solutions) with the latter potential refinement by the expert. The segmentation, obtained in either of the aforementioned ways, is a prerequisite for the *in-silico* estimation of hemodynamic features with the usage of Computer Fluid Dynamics (CFD) or AI.

### 1.2.1 CFD-based vFFR estimation

Estimation of vFFR with a CFD engine is the most common and recognized approach to non-invasive diagnosis. There are already solutions that have achieved FDA (U.S. Food and Drug Administration) clearance and can provide the product to hospitals and medical facilities [62]. The CFD-based approaches most often are derived from a common pipeline showcased in Fig. 1.2 which consists of 4 steps: (A) Image acquisition (CCTA), (B) Construction of anatomic 3D model, (C) Definition of physiology model and (D) Computation of coronary blood flow.

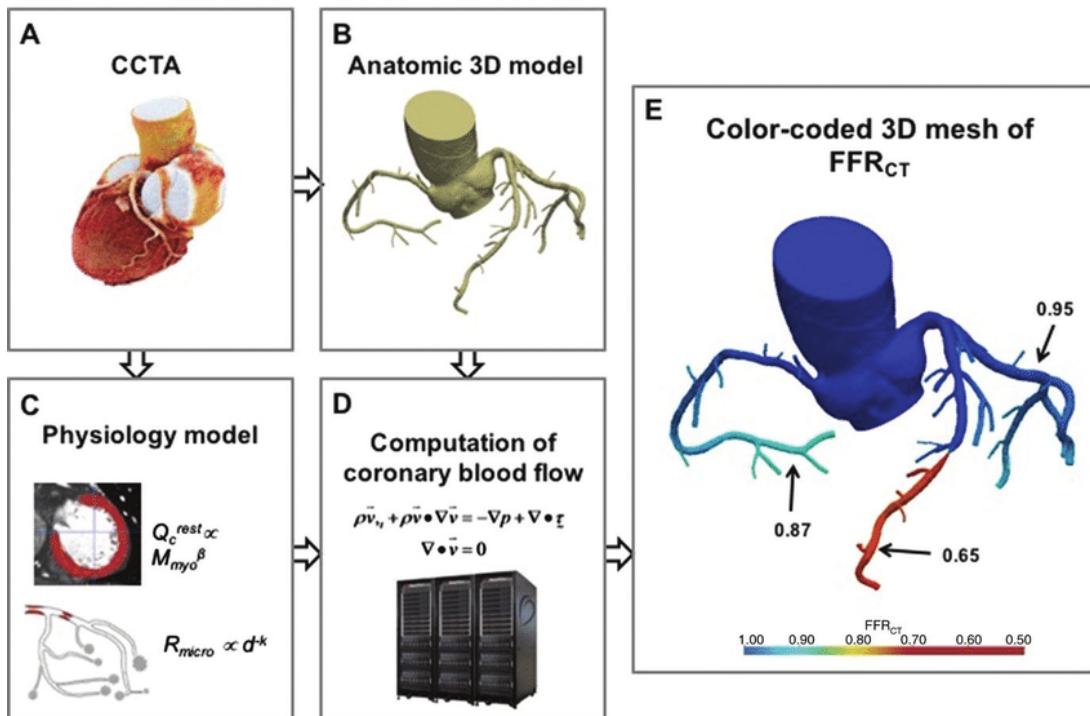


Figure 1.2: CFD-based vFFR estimation methods from CCTA are often derived from the common pipeline. Based on the CCTA image (A) the fine-grained anatomical 3D model (B) and physiology model defining boundary conditions (C), are constructed. Both of them serve as prerequisites of CFD numerical solver (D). The results are often visualised for the physician in the form of colour-coded 3D mesh (E).

Source: [54]

Steps A and B have been already described in general terms in section 1.2, however for the correct CFD simulation some further criteria must be met. The anatomic 3D model needs to be represented as a surface mesh of a fine-grained quality for the simulation to be stable [8]. Automatic generation of such mesh is not a trivial task and it often requires some postprocessing steps in the form of remeshing and manifold corrections [50]. Additionally, the level of mesh discretization, which can be understood as the maximal edge length, has a crucial influence on simulation quality and accuracy. One may consider utilizing a very fine discretization at all times to avoid the issue of stability, however, the aspect of time consumption becomes the major drawback. Thus the trade-off between time consumption and accuracy is an ever-looming issue which needs to be acknowledged and addressed in one way or the other to suit the task at hand.

In step C the physiology model is constructed based on the CCTA volume. The model consists of patient-specific boundary conditions (BC) estimated from rest state conditions including e.g. systolic and diastolic blood pressure, heart rate, and ventricular mass. To model the state of maximal hyperemia, induced by the administration of adenosine, the conditions are appropriately modified by modelling the decrease in microvascular resistance [54]. Moreover, the physiological model might be further enhanced towards specific patients by incorporating continuously recorded blood pressure waveforms obtained in a non-invasive measurement [31]. The correct definition of the physiology model is crucial for the CFD simulation to well reflect the real blood flow and be accurate.

The last step (D) is a computation of vFFR with a CFD simulation. It is based on solving the Navier-Stokes equations, the physical laws that govern fluid dynamics, to compute the blood flow and pressure fields in the coronary vasculature. However, because these equations are complicated nonlinear partial differential equations, numerical solving needs to be done in an iterative manner and thus is a computationally demanding task [54]. The most common approach derives vFFR from a 3D anatomical model

(obtained in step B), however computations for this approach, due to its complexity, need to be implemented off-site on supercomputers or cloud services [30, 56]. This model is called *full-model* and so-called *reduced-order* models have been proposed as alternative, less computationally demanding, approaches. These models are based on averaging Navier-Stokes equations over vessel-cross sections and generalizing coronary microvasculature resistance parameters [54, 53, 36]. This approach allows for on-site computation of vFFR in less than 1 hour [25], however, the accuracy in small, stenotic and bifurcating segments is decreased.

Although CFD-based vFFR estimation is a compelling *in-silico* alternative to invasive FFR measurement, its use as prophylaxis is still limited due to the high computational and time demands. With the recent rapid advances in machine learning (ML) and deep learning (DL), the interest in utilizing these techniques in the estimation of vFFR has been sparked. Due to its nature, deep learning allows to drastically limit the computational and time demand in inference settings which is the main drawback of CFD-based approaches.

### 1.2.2 AI-based vFFR estimation

The AI-based approaches utilize a similar pipeline as CFD ones (showcased in Fig. 1.2), by surrogating steps C and D with an ML model which estimates hemodynamic features directly. The ML models are most often trained in a supervised matter by considering CFD-based simulation results as the ground truth (GT), thus they can be only as good as the underlying simulation.

One may consider training the model on invasive FFR measurement, however, this approach is limited by the nature of the measurement which is done only locally on the suspected stenotic segment but not continuously along the whole vessel. There exists a locally-continuous technique of pressure measurement along the vessel called *intracoronary pressure pullback*. This technique is performed by bringing the measurement device in the form of a pressure wire to the lesion of interest and then performing a pullback manoeuvre whereby the pressure wire is pulled back at a fixed rate through the vessel towards the aorta, and the pressure along the way is recorded continuously [37, 35]. However, such a procedure is not commonly performed and only yields pressure values in the local segment of interest.

The main gain of utilizing the AI-based approaches over CFD ones lies in the computational and time demands in the inference setting. As mentioned in Section 1.2.1, the iterative solving of numerical equations can take up to hours for a single geometry. When using AI methods, the time and computational demands are shifted from the inference onto the training process, which is done only once in the non-production environment before deployment. The already trained model, depending on the exact architecture and approach, drastically reduces the time demand of the vFFR estimation to seconds and can often be run locally on-site. Due to this fact, AI-based approaches may serve as a compelling surrogate of the CFD engine and could bring vFFR estimation procedures to be a common prophylaxis non-invasive diagnostic tool.

## 1.3 Summary

In this chapter, we introduced and described the medical background of FFR estimation. We described in detail a process of obtaining an in-vivo FFR measurement and highlighted its issues, risks for the patient, and obstacles that stand in the way of utilizing it as a common diagnostic technique. Next, the concept of *in-silico* measurement in the form of virtual FFR estimation has been introduced as a compelling non-invasive alternative that allows for a broader patient screening. The most common technique based on the CFD engine was described and the main drawbacks in the form of time and computational demand were highlighted. As the solution, we introduced the approach of surrogating a CFD engine with the learned AI model. In the next chapter, we will delve deeper into AI-based methods and techniques by discussing the related works and the most commonly used approaches up to date.

# Related work

In this chapter, we will present and discuss the related works that tackle the problem of estimating hemodynamic features (not only in the form of vFFR) with AI. To solve this task a few different approaches have been proposed in recent years. The most basic ones perform extraction of hand-crafted features from the underlying geometry and use them as the geometry representation for the latter regressor of the hemodynamic feature of choice [26, 49, 57, 18]. In more advanced approaches, the geometry is encoded implicitly by the dedicated encoder working directly on the input mesh [32, 52, 51]. Such methods can be further enhanced by incorporating physics-inspired losses to form so-called *physics-informed neural networks* (PINNs), whereby the underlying differential equation is exploited to guide the network [44, 42, 43]. However, in this work, we will not delve into PINNs but rather only mention them for the sake of the potential future ways of research.

## 2.1 Learning on explicit hand-crafted features

The first models introduced to estimate vFFR via AI proposed to utilize hand-crafted features. As such features, we consider ones that need to be designed a priori by the domain expert and can be automatically extracted or computed based on the vessel geometry. When designing such features, expert knowledge is required since the features need to correlate with the estimated hemodynamic feature for the model to be even learned. This is not a trivial task, since not all correlating and important features are obvious, and can be missed. Since no other information besides these features is given to the model, its performance relies almost only on the design choice of the expert. On the other hand, including such features allows for greater explainability. The features are tangible for the expert and their influence on the result can be measured.

### 2.1.1 Siemens framework (Itu et al.)

We will start by introducing the work by Itu et al. [26], which together with Siemens, proposed the first-ever AI-based vFFR estimation framework. The steps of the proposed framework are showcased in Fig. 2.1. In the first step, the dataset of 12,000 coronary anatomies is generated to form a rich sample of the different morphologies of coronary blockage. The parameters of generated geometries are sampled in prespecified ranges based on the published medical literature [34]. One may raise a question of why the model is not trained on the dataset of real patients' anatomies. The authors claim, that it is known that DNNs require lots of data to be properly trained, and datasets of real geometries are often not large enough to be able to cover the vast variability between the patients. To prove this claim, the authors showcase that in their framework training on synthetic geometries does not hinder the performance on the real anatomies on which the model is validated.

The framework utilizes a supervised learning scheme and the ground truth labels are generated via CFD simulation (this notion was introduced in Section 1.2.1). The vessel geometry is split into segments that are considered stenotic and non-stenotic. From the stenotic segments, the hand-crafted features are extracted, and together with CFD ground truths, they form a training dataset for the ML model. In the inference setting (online section in Fig 2.1), the ML model trained on synthetic anatomies is utilized to infer vFFR for the patient-specific coronary anatomical models with latter feature extraction in the same

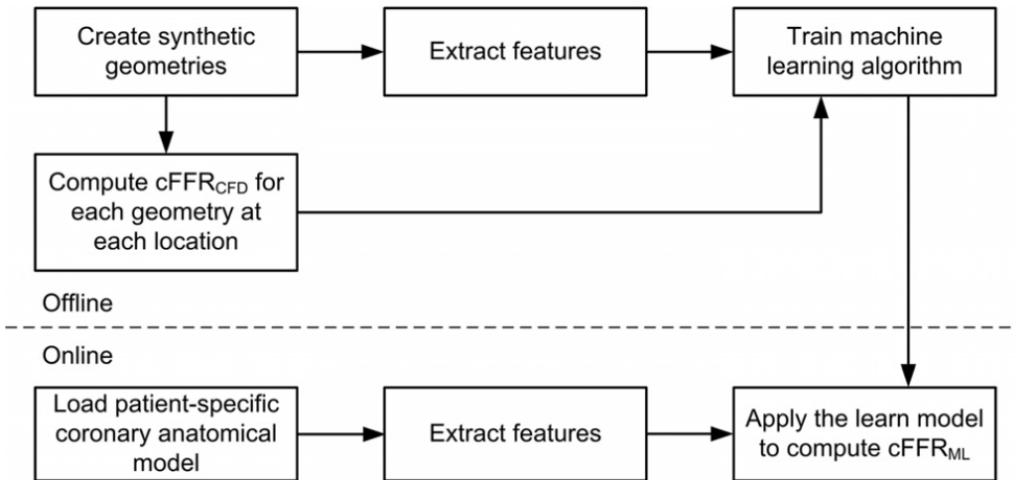


Figure 2.1: Workflow of Siemens framework. The model is trained on synthetic data offline and then deployed online to perform vFFR regression on real patients.

Source: [26]

manner as for the synthetic geometries.

As the ML model, the authors propose a simple multi-layer perception (MLP) built out of 4 hidden layers (see Fig. 2.2a). The model performs vFFR regression for the given stenotic segment based on a set of extracted features. The authors utilize a total of 28 hand-crafted features which encode local, upstream and downstream information (examples of local features are presented in Fig. 2.2b). The features, however, are not described in detail enough for the approach to be reproducible, which was also reported by Sklet et al. [49].

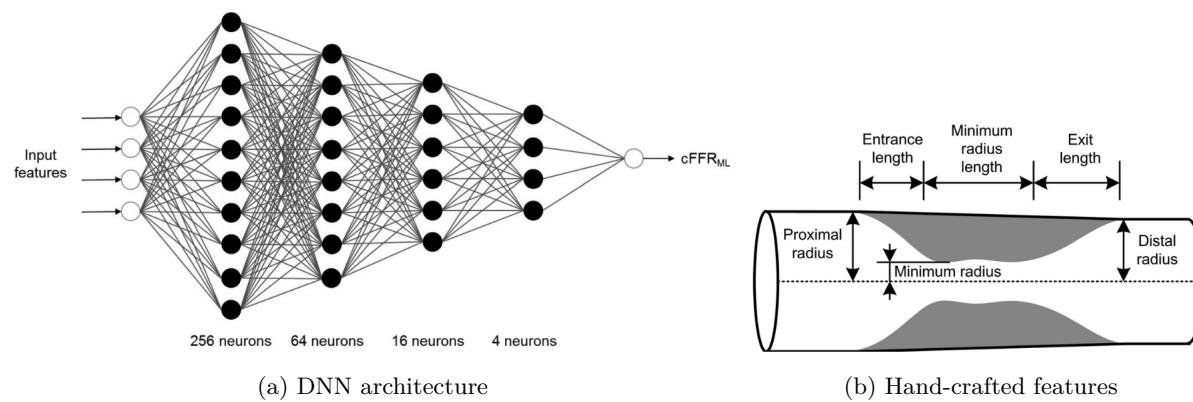


Figure 2.2: Deep learning approach by Siemens. The simple 4-layer MLP is utilized to regress local vFFR from hand-crafted features in the form of local and global vessel characteristics.

Source: [26]

The Siemens framework was validated on the database of 87 patient-specific anatomies resulting in a total of 125 stenotic segments. They report a vFFR correlation of 0.9994 between the CFD and AI approaches. However, for the invasive FFR, the reported correlation is only 0.729. As we mentioned in Section 1.2.1, the performance of the ML model can be only as good as the CFD ground truth. And since, the reduced-order CFD model utilized in this work, achieves a correlation of 0.725, we cannot expect the ML model to be substantially better.

### 2.1.2 DEEPVESSEL-FFR (Wang et al.)

The platform DEEPVESSEL-FFR proposed by Wang et al. [57] follows a similar approach as previously described Siemens framework (2.1.1) in utilizing hand-crafted features. However, instead of a simple MLP, the DNN model is arranged out of two components, a feature encoder in the form of a multilevel neural network (MLNN) and a bi-directional recursive neural network (BRNN) (see Fig. 2.3). The first phase encodes hand-crafted features describing vessel geometry in the form of lesion characteristics and proximal/distal markers subsequently defined for each lesion, into feature vectors. In the second phase, recurrent architecture is utilized to include in local vFFR regression upstream and downstream information in the form of surrounding centerline node embeddings. As for the Siemens approach (2.1.1), the utilized ground truths are obtained via classic CFD simulation.

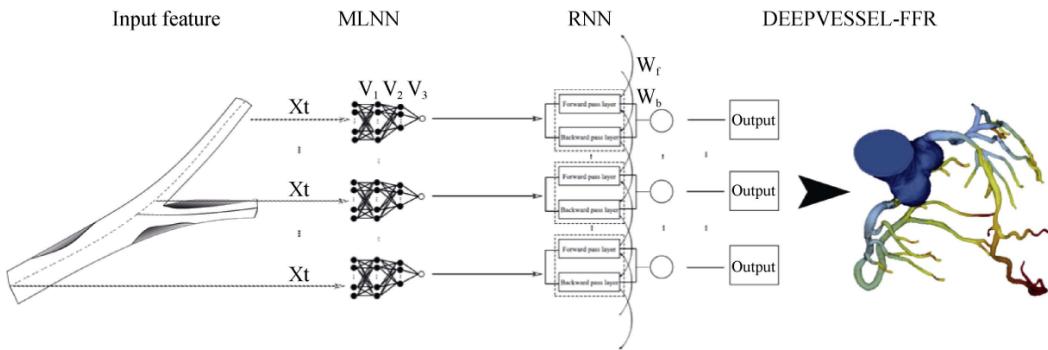


Figure 2.3: DEEPVESSEL-FFR framework. This approach utilizes two phases, a simple feature encoder, similar to Siemens framework (2.1.1), and a latter RNN for more robust vFFR regression that includes upstream and downstream vessel characteristics.

Source: [57]

The authors do not disclose the setup of the training database and only provide the information that the evaluation study has been performed on 68 patients. They report the correlation with invasive FFR measurement to be 0.686. However, as for the Siemens framework (2.1.1), the proposed approach shares the same issues regarding reproducibility, due to the lacking description of exactly used hand-crafted features.

### 2.1.3 Fossan et al.

The next approach that utilizes the hand-crafted features but also incorporates some physics-based knowledge was proposed by Fossan et al. [18]. In this work, the authors utilize a reduced-order physics model and the fully-connected neural network to regress pressure drops on the local vessel segment and based on the predicted results solve the exact vFFR. The local segment is represented by the set of hand-crafted features describing the vessel geometry (see Fig 2.4b) e.g. radii along the stenosis ( $r_d, r_p, r_s$ ), length of the stenosis ( $l$ ) and cross-sectional area information ( $A, d_{min}, d_{max}$ ). The features are further enhanced by incorporating the physics knowledge derived from the reduced-order physics model in the form of blood flow ( $Q$ ), pressure loss ( $\Delta P_{0D}$ ) and dynamic and flow separation changes/losses on upstream pressure ( $\Delta P_{sep}$ ).

The model was trained and validated on the dataset collected from 64 patients with stable CAD and ground truths were generated via the CFD simulation. The authors report an error standard deviation of 0.021 between CFD GT and their proposed approach.

### 2.1.4 Summary

To conclude, the models utilizing hand-crafted features were the first ones to be proposed to solve the problem of vFFR regression. They offer some form of explainability by utilizing explicit features and can

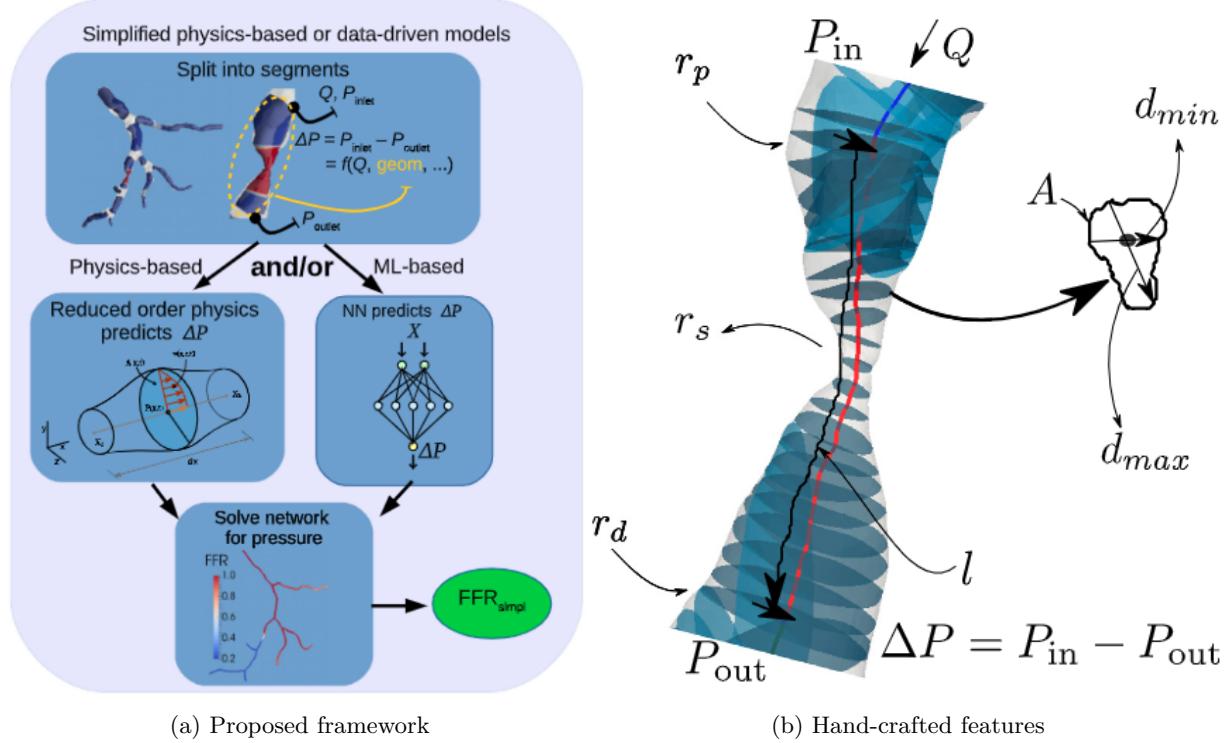


Figure 2.4: Deep learning approach by Fossan et al. In this approach, the pure MLP used in Siemens framework (2.1.1) is paired with a reduced-order physics model to incorporate physics-based knowledge into the training. The hand-crafted features used to encode the local segment are derived both from geometry: radii along the stenosis ( $r_p, r_s, r_d$ ), length of the stenosis ( $l$ ), cross-sectional area ( $A$ ), minimal cross-sectional diameter ( $d_{min}$ ), and maximal cross-sectional diameter ( $d_{max}$ ); and fluid motion: input pressure ( $P_{in}$ ), output pressure ( $P_{out}$ ), inflow ( $Q$ ), pressure loss ( $\Delta P_{0D}$ ) and flow separations changes on upstream pressure ( $\Delta P_{sep}$ ).

Source: [18]

be further enhanced by incorporating more geometry or physics-based features. However, since the features need to be defined explicitly by the expert for specific tasks, the models are not simply transferable to estimate different hemodynamic features and due to some important features being potentially missed in the design process, the DNN model might not be able to be trained properly. Moreover, the common lack of in-detail description of utilized hand-crafted features poses a major limitation in reproducing and validating the proposed approaches.

## 2.2 Learning on implicit features

The alternative approach to encoding geometry based on hand-crafted features is to train the model to perform implicit feature extraction from the raw geometry. These approaches utilize DNNs that are tailored towards the processing of 3D structures such as point clouds or meshes, which will be described in further detail in Chapter 3. By encoding raw geometry, the requirement of defining a set of carefully designed hand-crafted features is no longer needed, and thus the models are not limited by the encoding scheme proposed by the expert. The embeddings of the geometries are learnt implicitly for the task at hand and thus these architectures can be utilized to estimate a variety of hemodynamic features. In this approach, however, we lose some form of explainability by sacrificing known features for the implicit representation, which does not provide much interpretability.

### 2.2.1 Li et al.

In Li et al [32], the authors propose to represent input geometry as the 3D point cloud to estimate velocity and pressure fields in the coronary arteries and aorta before and after bypass surgery. The authors propose a novel PointNet-based [40] architecture (showcased in Fig. 2.5a) built out of two separate encoding branches. The input point cloud is split into *model point cloud* which contains boundary points of the geometry, and the *query point cloud* which contains points in the vessel lumen. According to the authors, the split was performed in this manner to encode local and global features separately to allow for better spatial relationship encoding. From both point clouds the same amount of input points are sampled and obtained encodings are stitched together to form a global representation. The obtained representation is further decoded with fully-connected layers to yield the final hemodynamic feature. When we described the learning on hand-crafted features, the issue of being unable to utilize the network for various hemodynamic features, due to different requirements for input features, has been highlighted. However, when we deal with implicit feature learning, this is no longer an issue, since the network can learn relevant features for the task on its own. This case is proved by Li et al. [32] by showcasing that the same architecture can be utilized with the same success for both 3D velocity and pressure field estimations (qualitative example of obtained pressure field is showcased in Fig. 2.5b).

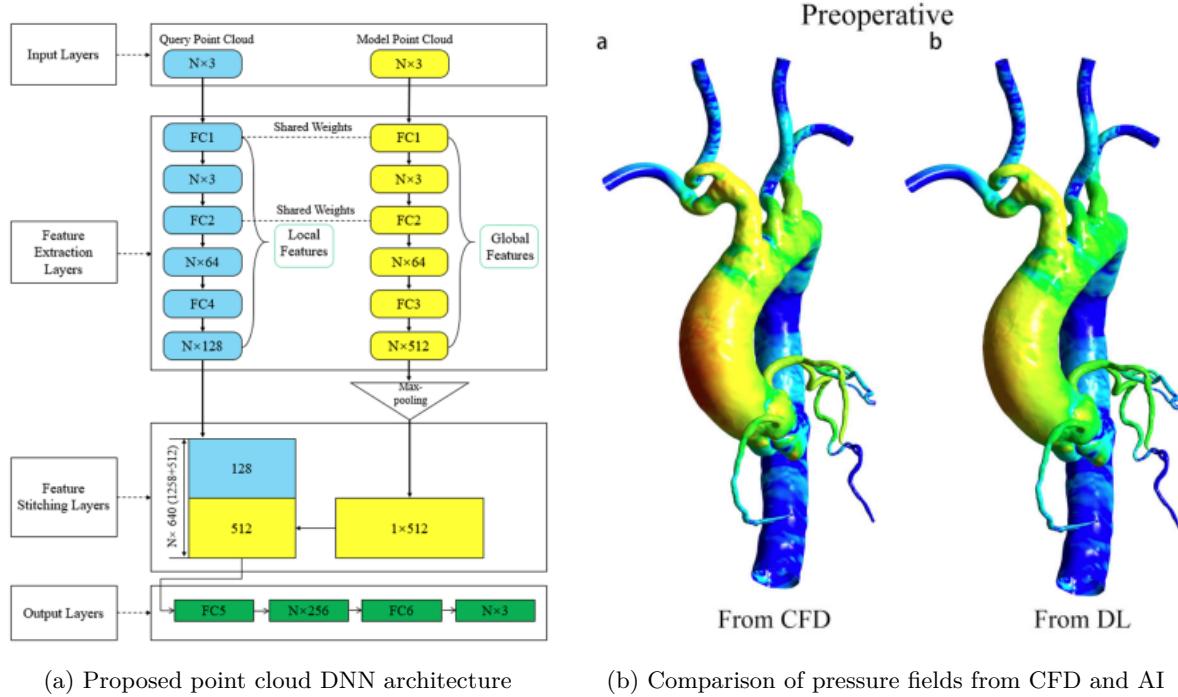


Figure 2.5: Point cloud based approach by Li et al.

Source: [32]

The study has been performed on 110 real patient data and the ground truths have been obtained via CFD simulation. They report the correlation of vFFR with CFD-obtained FFR to be 0.9580 and highlight that the performance must be further evaluated due to the lack of clinical data in the form of invasive FFR to compare with.

### 2.2.2 Suk et al.

The works by Suk et al. [52, 51] delve into utilizing mesh-based neural networks for the hemodynamics estimation in the form of Wall Shear Stress (WSS) and velocity field. The input geometry is represented as the surface or volumetric mesh and then processed with UNet-like [46] architecture tailored towards

processing raw meshes (see Fig. 2.6). The nodes of the input mesh, are further enhanced beyond spatial coordinates features only, to include simple hand-crafted features in the form of geodesic distance from the vessel inlet. This feature is easily computed and serves as the prior which informs the network of fluid flow direction. As for Li et al. [32], the advantage of the proposed approach lies in the notion that the architecture can be utilized for various hemodynamic feature estimations.

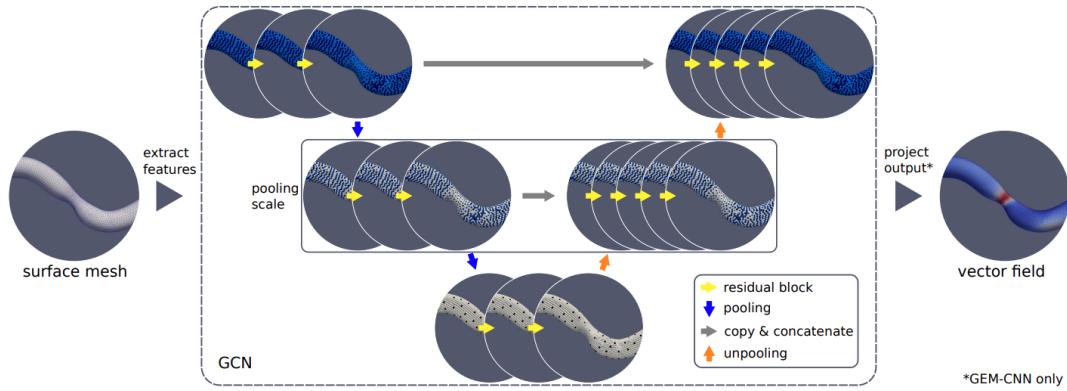


Figure 2.6: Mesh-based DNN architecture by Suk et al.

Source: [52]

The approach is trained and evaluated on synthetically generated datasets containing simple geometries with stenoses and bifurcating geometries, both containing 2000 unique geometries. Again, as for previously discussed approaches, the ground truths are obtained via CFD simulation. The authors report the x180 speedup by utilising the AI approach in comparison to CFD one and plan on further extending their work in the future to include real geometries.

### 2.2.3 Summary

To sum up, learning on features that are implicitly extracted by the model to the task at hand, is a compelling alternative to approaches utilising hand-crafted features. The implicit models are far more generic and can be utilized to estimate various hemodynamic features, whereas hand-crafted ones are limited towards the specific hemodynamic feature and cannot be extended to other problems. The main drawbacks lie in decreased explainability and higher computational demands since more advanced architectures, tailored towards the processing of raw 3D data, must be utilized. The hand-crafted representation takes the form of a low-dimensional numerical feature vector which can be sufficiently processed with simple MLP which has low computational overload. However, with recent advances in deep learning for 3D shapes, efficient architectures have been developed to process raw 3D point clouds and meshes, making implicit learning a viable choice.

## 2.3 Summary

In this chapter, we reviewed the most relevant works that tackle the problem of vFFR and other hemodynamic feature estimations. We grouped the methods into the explicit and implicit feature learning categories and highlighted their advantages and disadvantages. We discussed the proposed workflows of learning on synthetic and real data and noted that all the methods utilise CFD simulation to generate ground truth labels for the supervised task of vFFR estimation. In the next chapter, we will discuss in detail learning on 3D shapes (implicit feature learning).

# Learning on 3D shapes

In this work, we focus mainly on implicit feature learning (see Section 2.2) on the raw representations of 3D objects in the form of point clouds. Thus, in this chapter, we will describe the concepts behind learning on 3D shapes. We will start by introducing various 3D data representations and discussing their strengths and weaknesses. Next, the blueprint of the so-called geometric deep learning (GDL) [7] will be discussed as the set of concepts and guidelines for constructing architectures towards learning robust representations of 3D shapes. Through the lens of GDL, we will introduce specific architectures for learning on point clouds and meshes that are utilized in this work.

## 3.1 Representations of data in 3D

The 3D objects in computer vision can be represented in various ways, depending, among others, on the image acquisition device or specific use case of the task at hand. When defining the problem to be solved, we must also consider which representation is most suited to it. In this work, we will look at the three most common representations of data in 3D: voxel grids, point clouds and meshes (see Fig. 3.1).

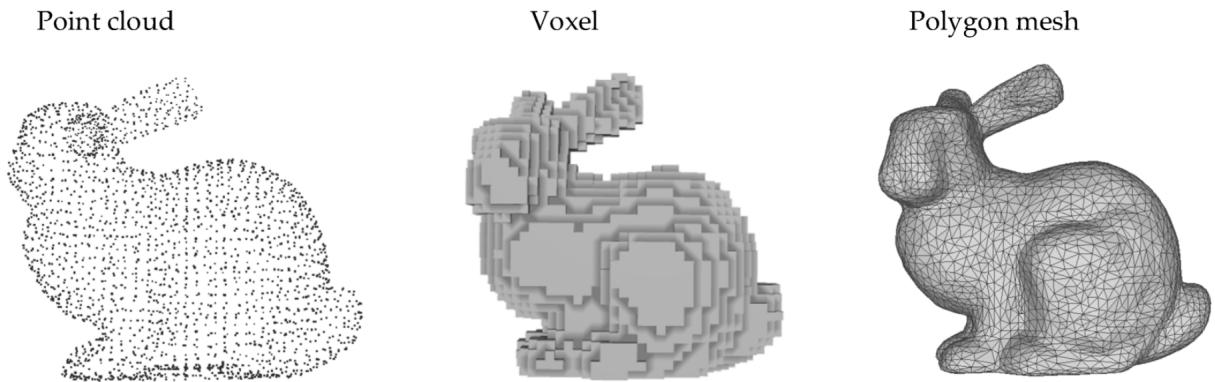


Figure 3.1: Representations of data in 3D. Three-dimensional model of the bunny represented in the 3 most common representations (from the left): point cloud, voxel grid and polygonal mesh.

Source: [20]

### 3.1.1 Voxel grid

The *voxel grid* is a 3D extension of a 2D *pixel grid*, which is the most common technique of representing images. The pixel grid is a 2D regular grid built out of square (isotropic) or rectangular (anisotropic) elements called pixels. The images may come in the form of single or multi-channel grids. In single-channel grids, each pixel takes a single value, and in multi-channel a tuple of values e.g. RGB, where each pixel is represented with the 3 element tuple encoding its colour. As mentioned earlier, the voxel grid is a natural extension of the pixel grid into 3D, where the smallest element called *voxel* can be either a cube or cuboid.

When dealing with 3D objects, we often represent them as binary masks with the voxel grid. The object is placed inside the cuboid bounding box, which is the regular grid of specified voxel density. The voxels belonging to the foreground (object) take a value of 1 and to the background of 0. In this manner, we can think of a voxel grid as a construction of an object out of small regular cuboids.

The main issue of the voxel grid representation of 3D objects lies in the resolution. Since the grid needs to be regular the regions with the various amount of detail are represented with the same amount of elements. Thus, when we want to increase the resolution of a certain part of the object, the resolution of the whole grid needs to be increased, resulting in high memory costs (see bunny model labelled as "voxel" in Fig. 3.1 - the resolution of a surface near the ears is low, the model is "blocky", however for a belly part it is high enough). Moreover, 3D objects are often a 2D surface embedded in a 3D space and thus the representation of space inside and outside of the object is not required. The voxel grids are not efficient in this manner, due to the aforementioned constant resolution across the whole grid. For example, the background of the object is modelled with a large amount of 0-value voxels which do not provide any information. For these reasons, we rarely utilized this data representation when dealing with 3D objects, and rather opt to utilize more sparse representations that allow varying resolution across the domain.

### 3.1.2 Mesh

The *mesh* is a graph-based representation of a 3D surface (surface mesh) or volume (volumetric mesh). We construct the mesh out of nodes, which represent the points embedded in the 3D space, and edges, which represent the connectivity between the nodes to form a 2D surface built out of closed polygons, called faces. Each node is represented as a tuple of 3D spatial coordinates  $x$ ,  $y$  and  $z$  in the specified metric space (most commonly we utilize a Euclidean metric space). The tuples can be further expanded to hold more values per point e.g. colour, normal vector etc.

Due to its graph nature, the mesh can represent the underlying object surface with varying resolution. The more detailed areas are densely sampled with nodes, thus conveying more detail, and more simple areas can be sampled sparsely since there is not much detail to convey (see bunny model labelled "polygon mesh" in Fig. 3.1). Moreover, when we deal with surface meshes, the interior and exterior of the object is not modelled at all, making it far more memory-efficient in comparison to the voxel grid. These properties make a mesh, to be an efficient, detailed and sparse representation of a 3D object.

However, there are some technical caveats and requirements that must be met when dealing with meshes. We often require a mesh to be a *manifold*, meaning that it is watertight (the inside and outside are clearly defined and not connected), each edge has exactly two adjacent faces and there are no intersections between the faces. Keeping these properties is not trivial when some processing techniques are utilized. Often, fixing techniques in the form of remeshing [50] or cleaning of degenerated faces, nodes or edges, are required. So, even though, the mesh is the far superior technique for representing 3D objects, its irregular nature makes it far more tricky to process than the voxel grid.

### 3.1.3 Point cloud

The *point cloud* is a lightweight 3D representation that in comparison with the mesh, consists of nodes only without edges. We can think of a point cloud, as an unordered set of points from some metric space (most commonly a Euclidean one). As for a mesh representation, nodes can be represented as  $n$ -dimensional tuples consisting of 3D spatial coordinates and other features of choice.

Compared with a mesh representation, a point cloud conveys less information due to the lack of connectivity between the points and explicitly defined surface. To correctly represent the object's surface, more dense point sampling is required than in the case of mesh, where nodes can be further away from each other since the surface is given via connected edges (see bunny labelled "point cloud" in Fig. 3.1 - without dense sampling in non-detailed areas the surface given implicitly is ambiguous). However, we shouldn't consider points in a point cloud representation totally isolated. The relations between the points are given

through the metric space and even mesh with the correct surface can be reconstructed from it via some standard techniques such as ball-pivoting algorithm [6] or Poisson surface reconstruction [29]. Moreover, due to their simple construction, processing point clouds is much easier and more efficient than meshes, especially in the case of neural networks.

However, designing a neural network architecture suited to the processing of raw point clouds is not a trivial task. Thus, before introducing the specific deep learning approaches, we will delve into the *Geometric Deep Learning* [7] which sets up a set of general principles to follow when designing architectures to learning on high-dimensional data.

## 3.2 Geometric deep learning<sup>1</sup>

*Geometric Deep Learning (GDL)* is a set of general principles developed by Bronstein et al. [7] for designing neural networks that can learn robust and stable representations of high-dimensional data. The proposed hypothesis behind GDL states that “*learning generic functions in high dimensions is a cursed estimation problem, most tasks of interest are not generic, and come with essential pre-defined regularities arising from the underlying low-dimensionality and structure of the physical world.*” [7]. The exploitation of such regularities allows for remedying the curse of dimensionality when dealing with large and complex systems.

The principles behind representation learning GDL can be explained with three concepts: *symmetry*, *deformation stability* and *scale separation* (see Fig. 3.2). Each of these principles addresses different aspects of exploiting geometrical priors for representation learning, and together they built a blueprint of GDL, which is universal with respect to data domains.

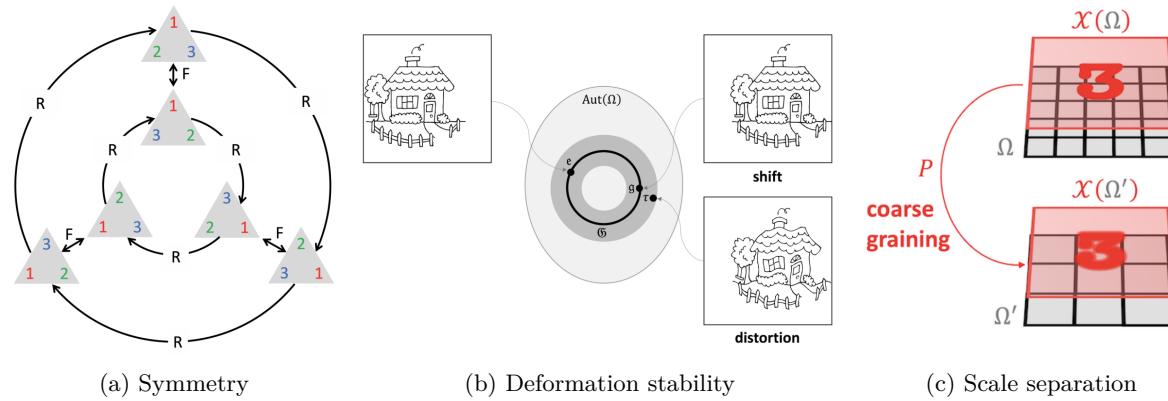


Figure 3.2: Principles of Geometric Deep Learning (GDL). The GDL combines invariance and equivariance to symmetries, stability to deformations of both signals and domain and separation of scale for local-to-global hierarchical learning, to propose a powerful generic blueprint of designing neural networks for learning robust and stable representation on various data domains.

Source: [7]

### 3.2.1 Symmetry

The *symmetry* of an object is a transformation that leaves its certain property or whole system unchanged [7] (the Fig. 3.2a showcases rotational symmetry, the triangle is still a triangle no matter the

<sup>1</sup>In this section, we will only introduce the basic concepts behind the GDL that are important for defining the notion of representation learning on point clouds. All the definitions and equations are based on the work by Bronstein et al. [7] and for more details the reader is referred to this work.

rotation). This concept is commonly exploited when dealing with the curse of dimensionality, to radically reduce the dimensionality of the data domain, since it imposes a powerful inductive bias. For example, when we tackle the classification task on images, the object category is unchanged (or *invariant*) by the translations. Thus, equipping the classification network with tools that enforce invariance under translation would allow to greatly simplify the problem. Practically, however, designing such mechanisms is not an easy task and often only some amount of invariance can be achieved. Coming back to the image classification example, a weaker form of invariance, namely *equivariance* is achieved by utilizing the convolutional layers. The image representation obtained after the convolutional layer is not the same when the object is shifted, however, the representation of the objects present in the image is the same, since the weights of the convolutional kernel are shared among the whole image. This is one of the reasons why the linear layers are not well-suited towards image analysis - they do not induce equivariance or any other form of weaker invariance, since the weights are not shared among the whole image.

Formally, the symmetries are defined by introducing *symmetry groups*, which are *algebraic groups*. The *algebraic group* is a set  $\mathfrak{B}$  with a binary operation  $\circ : \mathfrak{B} \times \mathfrak{B} \mapsto \mathfrak{B}$  (denoted also  $\mathfrak{g} \circ \mathfrak{h} = \mathfrak{gh}$ ) that satisfies four following axioms [7]:

1. **Associativity:**  $(\forall \mathfrak{g}, \mathfrak{h}, \mathfrak{l} \in \mathfrak{B}) (\mathfrak{gh})\mathfrak{l} = \mathfrak{g}(\mathfrak{hl})$
2. **Identity:**  $(\exists! \mathfrak{e} \in \mathfrak{B}) (\forall \mathfrak{g} \in \mathfrak{B}) \mathfrak{eg} = \mathfrak{ge} = \mathfrak{e}$
3. **Inverse:**  $(\forall \mathfrak{g} \in \mathfrak{B}) (\exists! \mathfrak{g}^{-1} \in \mathfrak{B}) \mathfrak{gg}^{-1} = \mathfrak{g}^{-1}\mathfrak{g} = \mathfrak{e}$
4. **Closure:**  $(\forall \mathfrak{g}, \mathfrak{h} \in \mathfrak{B}) \mathfrak{gh} \in \mathfrak{B}$

Having a notion of a symmetry group, the concept of a *group action* on the underlying data domain  $\Omega$  and its space of signals  $\mathcal{X}(\Omega)$  can be introduced.

A *group action* of  $\mathfrak{B}$  on a set  $\Omega$  is defined as a mapping  $(\mathfrak{g}, u) \mapsto \mathfrak{g}.u$ , where  $\mathfrak{g} \in \mathfrak{B}$  and  $u, \mathfrak{g}.u \in \Omega$ . Intuitively, we can think of group action as a process of applying some symmetry transformation  $\mathfrak{g}$  to the point  $u \in \Omega$  and obtaining another point on the data domain  $\Omega$ . Since the data domain  $\Omega$  is only an underlying data structure, we associate it with a space of signals  $\mathcal{X}(\Omega)$ , that defines the mapping between data points and exact values. For example, when considering a 2D RGB image, the data domain is a 2D pixel grid and the space of signals defines the mapping between pixels and exact RGB values. The group actions work automatically on the signal space  $\mathcal{X}(\Omega)$  as well:  $(\mathfrak{g}.x)(u) = x(\mathfrak{g}^{-1}u)$  [7]. Being equipped with symmetry groups and group actions, we can introduce invariant and equivariant functions which are fundamental building blocks of the GDL blueprint.

A function  $f : \mathcal{X}(\Omega) \mapsto \mathcal{Y}$  is  **$\mathfrak{B}$ -invariant** if  $(\forall \mathfrak{g} \in \mathfrak{B}, x \in \mathcal{X}(\Omega)) f(\rho(\mathfrak{g})x) = f(x)$ , i.e., its output is unaffected by the group action on the input. [7]

A function  $f : \mathcal{X}(\Omega) \mapsto \mathcal{X}(\Omega)$  is  **$\mathfrak{B}$ -equivariant** if  $(\forall \mathfrak{g} \in \mathfrak{B}) f(\rho(\mathfrak{g})x) = \rho(\mathfrak{g})f(x)$ , i.e., group action on the input affects the output in the same way. [7]

where  $\rho : \mathfrak{B} \mapsto \mathbb{R}^{n \times n}$  is a *group representation* - a mapping that assigns to each group element  $\mathfrak{g}$  a matrix of a size of the feature space on which the group acts.

Coming back to the example with image classification, we can now view the convolutional operator as a *shift-equivariant* function - the shifting of an image before the convolutional operation gives the same result as shifting it by the same amount after it. Moreover, when constructing a classification network, we would often utilize some form of global pooling operator to form a 1-dimensional representation vector. This global pooling operation, interpreted through the lens of symmetry groups, is a *shift-invariant* function - the pooling is done over the whole image thus it is unchanged by the shifting. Viewing the construction of commonly utilized layers in deep learning in a more abstract and generic way, as proposed in the GDL blueprint, allows us to see parallels between architectures utilized for various, at first glance completely different, data domains such as graphs, images or 3D shapes.

### 3.2.2 Deformation stability

The introduced notion of symmetry is a powerful mechanism when we know exactly which transformations are to be considered invariant or equivariant in a given setting. However, the data examples coming from the real world are often noisy and do not adhere to the strict symmetry definition, thus some weaker form of symmetry needs to be defined. This weaker notion is to be referred to as *deformation stability* and can be distinguished between two scenarios: stability to signal deformations and to domain deformations [7].

**Stability to signal deformations:** In this scenario we come from the a priori knowledge that the small deformations of the signal  $x$  should not influence the result of the given function  $f(x)$ . This idea looks similar to the symmetry problem, however, the deformations, which are defined as small diffeomorphisms  $\tau \in \text{Diff}(\Omega)$  [7], do not form the group due to not adhering to the closure axiom (see Section 3.2.1). The composition of multiple small deformations creates a large deformation, thus the closure can be achieved only for the set of all possible deformations which is in contradiction with our main goal.

To tackle this problem, the proposition is to introduce some form of a measure  $c(\tau)$  which quantifies how far the given  $\tau \in \text{Diff}(\Omega)$  is from the given symmetry group  $\mathfrak{B} \subset \text{Diff}(\Omega)$ . Obviously  $(\forall \tau \in \mathfrak{B}) c(\tau) = 0$ , so the definition of equivariant and invariant functions can be naturally replaced with a weaker notion of *deformation stability*, given by the following equation [7]:

$$||f(\rho(\tau)x) - f(x)|| \leq Cc(\tau)||x||, \forall x \in \mathcal{X}(\Omega) \quad (3.1)$$

where  $C$  is some constant independent of the signal  $x$  that quantifies how much deformation we consider to be stable. A function  $f$  that satisfies the above equation is to be referred to as *geometrically stable*.

This idea is showcased in Fig. 3.2b with the example of a translation group. The black ring is a translation symmetry group  $\mathfrak{B}$  and elements  $\mathbf{e}, \mathbf{g}$  belong to this group. The grey ring showcases the set of geometrically stable transformations which are no longer elements of the symmetry group  $\mathfrak{B}$ , however, are to be considered valid since they are quantified by the given measure  $c(\tau)$  to be close enough to the elements of the symmetry group. This way, the stability to small deformations of the signal can be achieved and incorporated into the neural network design.

**Stability to domain deformations:** In the second scenario, we are dealing with the deformations of the geometric domain  $\Omega$  rather than the signal. For 2D or 3D images we often talk only about the signal deformations, since the underlying grid is not deformed, however, when dealing with 3D objects such as meshes or point clouds, which we can categorize as forms of spatial graphs, the domain itself is deformed - the coordinates and connectivities between the points can change.

Formally, for the space of all possible domains  $\mathcal{D}$ , an appropriate distance metric  $d(\Omega, \tilde{\Omega})$ , where  $\Omega, \tilde{\Omega} \in \mathcal{D}$  and  $d(\Omega, \tilde{\Omega}) = 0$  if  $\Omega$  and  $\tilde{\Omega}$  are equivalent, can be defined. Again, as for the signal deformation, the measure  $d(\Omega, \tilde{\Omega})$  quantifies how large the deformation between the source and obtained domain is, and the notion of stability can be represented with the similarly constructed equation [7]:

$$||f(x, \Omega) - f(\tilde{x}, \tilde{\Omega})|| \leq Cd_{\mathcal{D}}(\Omega, \tilde{\Omega})||x||, \forall \Omega, \tilde{\Omega} \in \mathcal{D}, x \in \mathcal{X}(\Omega) \quad (3.2)$$

where  $f : \mathcal{X}(\mathcal{D}) \mapsto \mathcal{Y}$  and  $\mathcal{X}(\mathcal{D}) = \{(\mathcal{X}(\Omega), \Omega) : \Omega \in \mathcal{D}\}$  is the ensemble of possible input signals defined over a varying domain. Again, a function  $f$  that satisfies the above equation is considered to be stable to domain deformations.

The introduction of the notion of deformation stability allows expanding the concept of symmetry to be more flexible when it comes to the data we observe in a real setting.

### 3.2.3 Scale separation

The symmetry allows us to reduce the data domain by grouping the various data representations that are to be considered invariant or equivariant, and deformation stability strengthens these priors by incorpo-

rating invariance to small signal and domain noise. However, although these concepts address the curse of dimensionality, the problem is not yet overcome, since with the domain growth the number of possible functions that satisfy the aforementioned limitations is still too large. The key point lies in the incorporation of a multiscale analysis structure that allows building the representation from local to global features.

This concept is called *scale separation* and it is based on the notion of multiscale coarsening of the data domain  $\Omega$  into a hierarchy  $\Omega_1, \dots, \Omega_J$ . The coarsening operation assimilates nearby points  $u, u' \in \Omega$  together, based on the given metric over the data domain. To process points assimilated in this manner, the so-called *locally-stable* function is utilized. The global target function  $f$ , which forms the data representation, often depends on long-range interactions between the features over the whole domain. By utilizing *locally-stable* functions, the interactions between the features can be separated along the scales, and the final global stable representation can be obtained via local-to-global feature analysis and propagation along the coarse scales [7].

This idea is often incorporated in the design of neural networks in the form of local pooling operators. Fig. 3.2c showcases the classic local pooling operator utilized in convolutional neural networks. Four adjacent pixels are assimilated together to form one output pixel with the combined signal. For the images, the representation learning is done hierarchically by utilizing multiple blocks of local feature extraction layers in the form of convolutions and local pooling operators. For other data domains, the same hierarchical notion can be utilized with exact layers being tailored towards the given data domain.

### 3.2.4 The Geometric Deep Learning blueprint

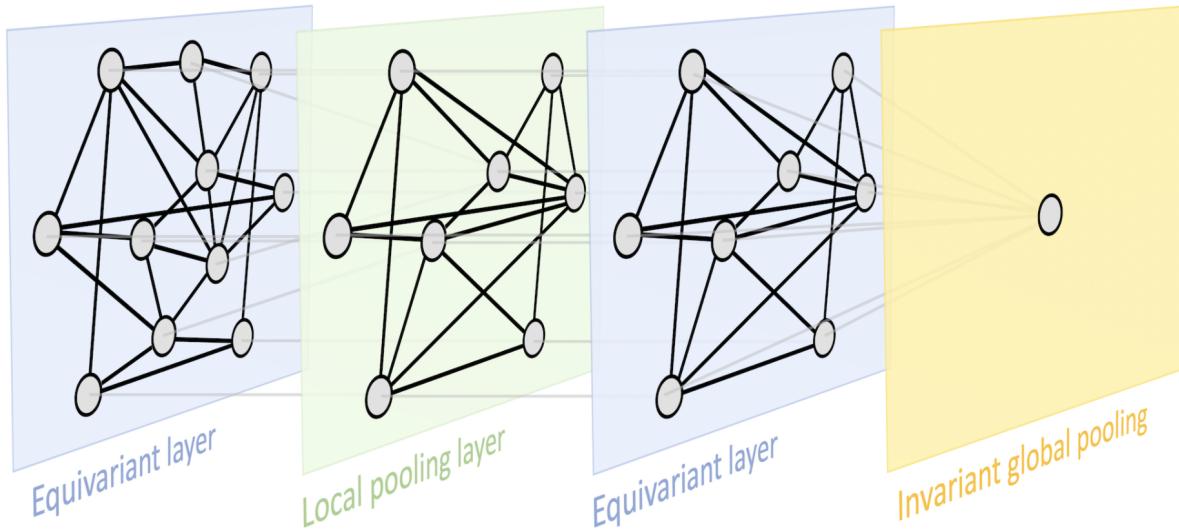


Figure 3.3: Example of the Geometric Deep Learning blueprint applied to learning graph representation. The representation is learned hierarchically with the blocks built out of equivariant and local pooling layers. The final representation is obtained via invariant global pooling.

Source: [7]

Equipped with the principles of symmetry, geometric stability and scale separation, we can move on to introducing a universal blueprint, proposed by Bronstein et al. [7], for learning stable representations of high-dimensional data. The blueprint does not impose any form of specific deep learning architecture, but rather a set of concepts that must be met. The principles provide three key building blocks that allow incorporating symmetry invariance, stability to deformations and robust function approximation in the construction of neural networks:

1. *Linear  $\mathfrak{B}$ -equivariant layer*:  $B : \mathcal{X}(\Omega, \mathcal{C}) \mapsto \mathcal{X}(\Omega', \mathcal{C}')$  satisfying  $B(\mathbf{g}.x) = \mathbf{g}.B(x)$  for all  $\mathbf{g} \in \mathfrak{B}$  and

$$x \in \mathcal{X}(\Omega, \mathcal{C}).$$

2. *Local pooling (coarsening)*:  $P : \mathcal{X}(\Omega, \mathcal{C}) \mapsto \mathcal{X}(\Omega', \mathcal{C})$ , such that  $\Omega' \subseteq \Omega$ .
3.  *$\mathfrak{B}$ -invariant layer*:  $A : \mathcal{X}(\Omega, \mathcal{C}) \mapsto \mathcal{Y}$  satisfying  $A(\mathfrak{g}.x) = A(x)$  for all  $\mathfrak{g} \in \mathfrak{B}$  and  $x \in \mathcal{X}(\Omega, \mathcal{C})$ .

where  $\Omega$  and  $\Omega'$  are domains and  $\mathfrak{B}$  a symmetry group over  $\Omega$ .

With the usage of the proposed blocks, one may construct a  $\mathfrak{B}$ -invariant functions that can learn data representation. The choice of the symmetry group is dependent purely on the data domain we act on and its properties which we want to enforce. For example, when dealing with images and utilizing Convolutional Neural Networks (CNNs), the data domain  $\Omega$  is a regular grid and the symmetry group  $\mathfrak{B}$  is translation. The definition of classic convolution can be further expanded to the concept of *group convolution* [12], which simply put, is a convolution that is invariant or equivariant to the chosen symmetry group. For example, when we are dealing with images for which we require the invariance to rotations we can utilize Roto-Translation CNNs [5], which take a group  $SE(2)$  (special Euclidean group [58]) as a symmetry group. On the other hand, Fig. 3.3 showcases the building blocks of a Graph Neural Network (GNN). In the case of GNNs, we require equivariance to the permutations of the nodes, and thus the symmetry group we deal with is a permutation group  $\Sigma_n$  [60]. The local pooling layer can be realised in many forms, for example with edge pooling [14], which downsamples the graph by removing the nodes with edge contraction operation.

Equipped with the GDL blueprint, we can now move on to introducing the exact architectures utilized for learning on point clouds, which are the main focus of this work, and discussing how they adhere to and realize the proposed principles.

### 3.3 Learning on 3D point clouds

We have already described the point cloud data structure (see Section 3.1.3) and introduced the GDL blueprint (see Section 3.2), hence we can move on to introducing how exactly the learning on 3D point clouds is realised in practice. To do it, we must first acknowledge the main properties of a point cloud which are of great importance when talking about representation learning [40]:

**Unordered (symmetry 3.2.1):** A point cloud is a set of vectors describing the features of individual points. Due to this fact, the ordering of the points has no impact on a point cloud itself, hence the representation should be invariant to it as well - the role of a symmetry group is fulfilled by the permutation group  $\Sigma_n$ . Thus when constructing the neural networks for consuming point clouds, we should seek to compose them out of permutation-invariant layers (3.2.4).

**Interaction among points (scale separation 3.2.3):** The points are not isolated, since they come from the metric space (most commonly a Euclidean one), hence the relationships between them should be acknowledged. The structure formed by the set of points is meaningful for the analysis of the shape, and both local and global interactions are to be considered when constructing the representation. We can clearly see the notion of scale separation in this reasoning and thus we would require the local-to-global feature analysis to be present in the neural network design.

**Invariance under transformation (symmetry 3.2.1 & deformation stability 3.2.2):** Point clouds are considered to be geometric objects placed in some sort of metric space, as we mentioned previously. In such spaces, the objects can undergo various types of transformations or deformations, and for some of them, we would like the learned representation to be invariant. For example, in the tasks of classification or segmentation, the learned representations should not be influenced by the affine transformations of the input. Moreover, we could also consider some sort of small noise applied to the points, namely domain deformation, to be non-influential in representation construction.

With key properties of point clouds, but also challenges in the light of network construction, highlighted, we can now move on to introducing the pioneering work in a raw point cloud analysis in the form of PointNet [40] and its successor PointNet++ [41]. Later on, we would also mention some domain-specific adaptations of these general architectures, which utilize prior knowledge about the point cloud structure to build a better more robust representation [48].

### 3.3.1 PointNet

The PointNet architecture takes as an input a raw point cloud in the form of a set of  $n$  points of  $k$  features each (at least 3 since we need to encode 3D spatial coordinates). The exact architecture is showcased in Fig. 3.4 in the two configurations: for the classification and segmentation task. The classification network is built out of the PointNet backbone and classic linear classification head. Meanwhile, the segmentation configuration utilizes an encoder-decoder scheme, with the same PointNet backbone serving the encoder role, and a simple fully-connected network as a decoder. The PointNet backbone is composed of three main components.

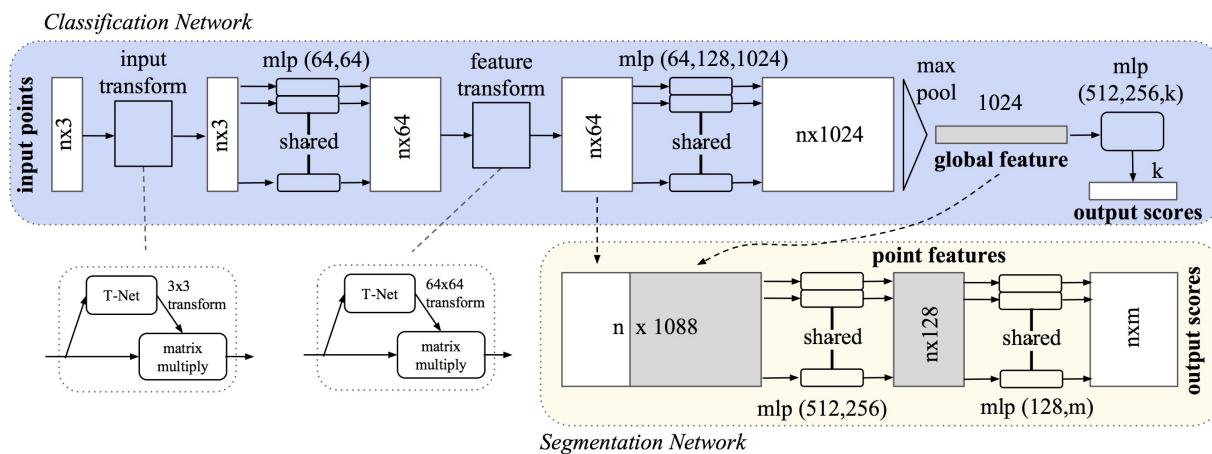


Figure 3.4: PointNet architecture.

Source: [40]

**Shared MLP:** The input point cloud takes a shape  $n \times k$ , as described previously. The classic MLP approach would be to flatten the 2-dimensional input into the 1-dimensional vector of length  $nk$ . That, however, is neither a robust nor efficient approach. First of all, the encoder is fixed to the arbitrarily pre-specified number of points and point clouds consisting of fewer or more points are unconsumable without some form of downsampling or upsampling preprocessing. Secondly, the notion of permutation-invariance is not met, since each vector element is processed with a unique independent neuron, thus the representation is dependent on the point order. Moreover, the number of parameters in the MLP with  $nk$  input neurons grows very fast with the size of a point cloud. To address these issues, authors propose a shared MLP layer, which can be simply defined as a *single-point encoder*. The shared MLP takes  $k$  input features (features of the single points), and processes them with linear layers, to obtain a single-point representation. Each point is encoded independently with the same shared MLP, hence the name *shared*, which refers to weight sharing between the point encoders. Through the lens of the GDL, the shared MLP is a *permutation-equivariant* layer, since the permutation of the points at the input is reflected in the obtained output representation of the points - the representations of points are ordered in the same way as input points. As we see in Fig. 3.4, the shared MLP layer is the main building block of the PointNet with increasing size of representation with the network depth.

**T-Net:** The second component proposed in the PointNet architecture is a T-Net (transformation net). It serves the purpose of regressing a transformation matrix that is later on applied to the current point features. The main idea behind introducing this component is to weakly enforce an invariance under

transformations/deformations. Since, as we mentioned previously (see Section 3.3), the input point cloud may come in various orientations or be slightly deformed, we would like to have a mechanism to remedy this problem. The T-Net is supposed to learn a transformation matrix which mitigates certain transformations implicitly considered to be invariant by the network, and bring all point clouds into some form of a canonical orientation that simplifies and stabilizes the representation learning. In practice, the T-Net is realised in the form of a small PointNet backbone with a transformation matrix regression head.

**Global feature pooling:** The point features encoded with the blocks of shared MLPs and T-Nets are at the end run through the global feature pooling layer to obtain a final 1-dimensional embedding of a given point cloud. The point cloud before the pooling procedure is of size  $n \times z$ , where  $z$  is the size of a per-point feature vector produced by the preceding shared MLP layer. The global pooling output vector is then of size  $z$  since the pooling is done along the point dimension, and as the pooling function, the *max-pooling* has been proposed by the authors. By performing the pooling with the *symmetric function* (one whose output is invariant under all permutations of the arguments [61]) in the form of max-pooling, the obtained point cloud representation is invariant to the permutation of the points at the input. When considering the GDL blueprint, the global feature pooling is a permutation-invariant layer.

As we can see, the PointNet architecture follows the principles of representation learning proposed by the GDL blueprint and addresses the notion of points being unordered in the point cloud, by utilizing permutation invariant and equivariant layers. It also attempts to solve the problem of invariance under transformation, by introducing implicit transformation matrix learning with T-Net. However, one major aspect of both the GDL and properties of the point cloud is not addressed in any way in the PointNet, namely scale separation / interaction among the points. The architecture does not propose any form of local-to-global feature analysis and only settles on learning single-point encodings and then performing a global aggregation. This is a major drawback, which limits the ability of a network to exploit local interactions and structure topology to enhance representation learning. This aspect of PointNet is to be addressed in the following work, namely PointNet++, which we will introduce in the following section.

### 3.3.2 PointNet++

The PointNet++ takes the raw point cloud of size  $n \times k$  on the input in the same way as PointNet. The architecture adopts the common encoder-decoder scheme (see Fig. 3.5). The encoder branch is built out of *set-abstraction* (SA) blocks and for the classification task the decoder is formed out of one PointNet layer which yields a global representation vector, and the fully-connected classification head. In the segmentation task, however, the decoder is more complex, and it is built out of *feature propagation* (FP) blocks and skip connections between same-level encoder and decoder blocks. Both SA and FP blocks, realize a missing component of PointNet in the form of exploiting local point interactions.

**SA-block:** The block is built out of the three steps: *sampling*, *grouping* and *encoding* (with PointNet).

1. *Sampling:* In this step, the subset of an input point cloud  $\mathcal{P}$  is sampled to form the representative point cloud  $\mathcal{R} \subset \mathcal{P}$ . Such a subset could be sampled randomly, however, the more densely packed areas would be over-represented this way. To avoid this issue, and better catch the global point cloud structure, the authors proposed to utilize a *Farthest Point Sampling* (FPS) algorithm. The FPS starts by initializing an empty set of representative points and adds a first point at random. Next, the point farthest (in the sense of chosen distance metric) from the whole set is found and added to the set. This process is done iteratively until the representative set is of the desired size.
2. *Grouping:* Upon having a representative point cloud  $\mathcal{R}$ , for each  $r \in \mathcal{R}$  the *grouping* step is performed independently to find the set of neighbouring points  $\mathcal{N}_r \subset \mathcal{P}$ . The aim of the grouping procedure is to extract local neighbourhoods of all  $r \in \mathcal{R}$ , thus it is based on the distance metric in the form of Euclidean distance. For each  $r \in \mathcal{R}$  a distance to each  $p \in \mathcal{P}$  is computed and the points are sorted based on it. The neighbourhoods then can be chosen in various ways, but the most common techniques, which were also considered by the authors, are *ball query* and *k-nearest-neighbours (KNN)*. The ball query defines a grouping threshold  $t$  and considers all points within the shorter

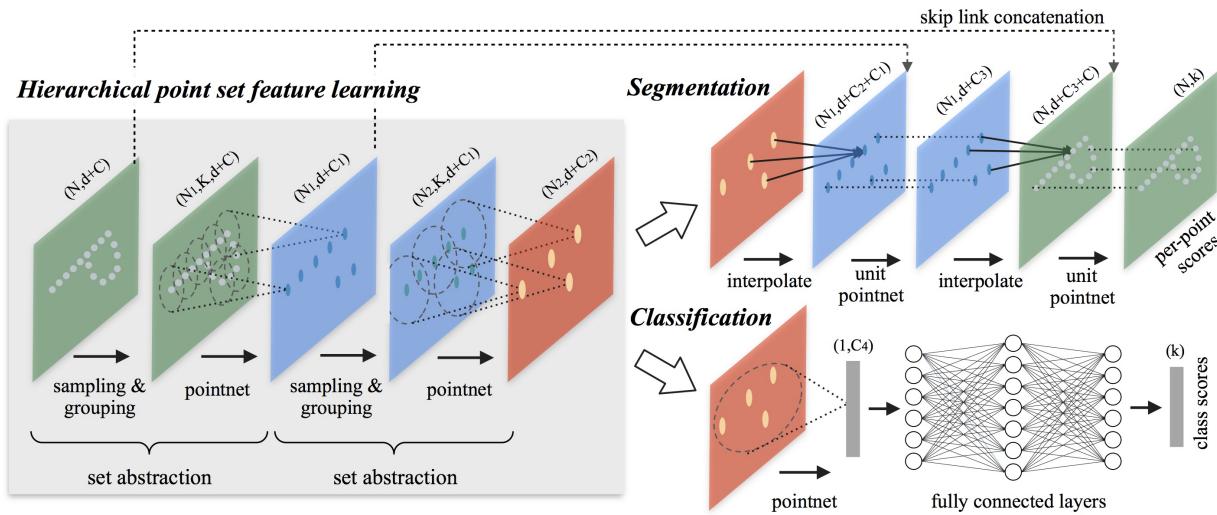


Figure 3.5: PointNet++ architecture.

Source: [41]

distance to be neighbours. The  $k$ -nearest-neighbours, on the other hand, takes simply  $k$  points with the shortest distance to the root point. The authors decided to utilize the ball query method, and argue that it is preferable due to being isotropic in regards to metric space, and thus the neighbourhoods are of the same size in the sense of distance metric.

3. *Encoding:* The neighbourhoods  $\mathcal{N}_r$  obtained in the aforementioned matter are later processed with PointNet to obtain neighbourhood embeddings, which become features of representative points.

The grouping procedure described above is referred to by the authors as *single-scale grouping (SSG)* and they propose also another scheme in the form of *multi-scale grouping (MSG)*. In the case of MSG, not one, but multiple thresholds  $t_1, t_2, \dots, t_k$  are considered, and for each independently, an SSG and encoding with PointNet are performed. The final features of representative points are a concatenation of encodings of multiple scales. This scheme is more robust since it allows the network to view a local neighbourhood in different scales and thus have a better view of the point cloud structure.

The final encoder is built out of multiple SA blocks, which perform both point cloud downsampling and hierarchical feature aggregation from local to global due to the increasing grouping threshold  $t$  of the ball query method. This very process, adopted by PointNet++, is the embodiment of the notion of scale separation proposed by the GDL.

**FP-block:** The block is built out of two steps: *interpolation* and *decoding* (with unit PointNet).

1. *Interpolation:* In this step, the features from the representative point cloud  $\mathcal{R}$  coming from the previous block are interpolated onto the point cloud from before the sampling stage in the encoder block on the same level. The interpolation is done with the KNN and the parameter  $k$  is set to 3 - each point takes features from the 3 closest points from  $R$ .
2. *Decoding:* The point cloud with interpolated features is at last processed with the unit PointNet which is the classic PointNet without the global pooling layer at the end. Thus the size of the point cloud is not changed and only single-point encodings are performed in this step.

As for the encoder branch, the decoder one is also built out of multiple blocks - in this case, the FP blocks. The number of blocks in each branch is the same, and they are connected together with skip connections to pass the information from different hierarchies.

As we can see, PointNet++ is a much more complex architecture than PointNet. The introduction of local-to-global analysis in the form of hierarchical neighborhood aggregation fulfills the role of taking into account interactions between the points in the representation learning and incorporates the notion of the scale separation proposed by the GDL. Although PointNet++ is a fine generic architecture for learning on point clouds, in some cases we may have prior knowledge about its structure which could be exploited in the network design. In the next section, we will delve into a few methods that propose such schemes.

### 3.3.3 Domain-specific adaptations

In some cases, we possess some prior knowledge about some characteristics of the topology of the point clouds that we are working on. By adopting the classic PointNet++ architecture we do not really utilize this information due to its generic approach. Since, in this work, we deal with vasculature trees in the form of coronary arteries, we would take a look at two domain-specific adaptations of the PointNet++ tailored toward the task of vessel labeling. Vessel labeling is the task of assigning a proper vessel segment label to each point in the vascular tree, so we can think of it as a part segmentation task. The proposed adaptations modify only the grouping procedure in PointNet++ to aggregate the features in a more robust way in light of the available prior knowledge about the data and the task.

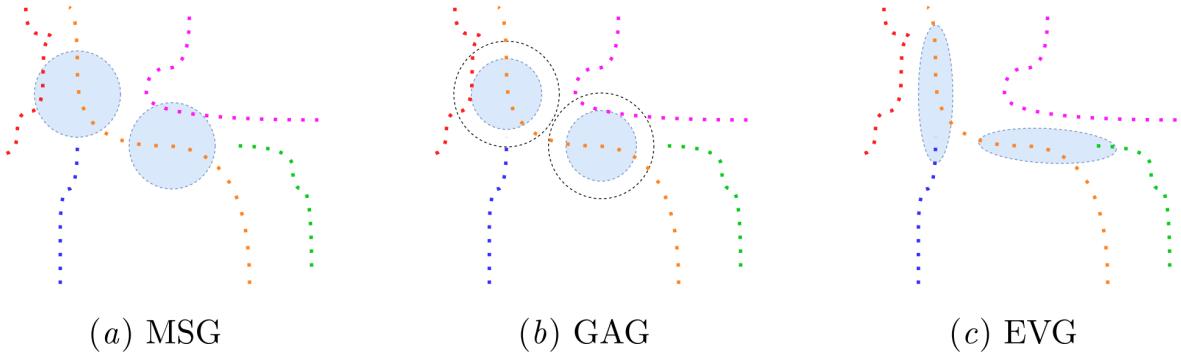


Figure 3.6: Point grouping strategies. MSG strategy is the most generic one and does not include any information about the underlying topology of the point cloud. In the GAG scheme, the connectivity between the components is taken into account. The EVG strategy utilizes the underlying topology of the point cloud by performing directed grouping and exploiting the prior knowledge about the vascular tree structure.

Source: [48]

*Geometry-aware grouping (GAG):* This grouping scheme has been proposed by He et al. [21] to put more priority on grouping together the points that belong to the same connected component. The grouping is performed with the ball query as in the MSG scheme, but the distance to the points that belong to the same connected component as the root point is scaled by the parameter  $\lambda$  set to 0.25 by the authors. Intuitively, this grouping procedure can be interpreted as grouping with two balls queries of different sizes, whereby the smaller one groups all the points and the bigger one only the points from the same component - this idea is portrayed in Fig. 3.6. This approach exploits prior knowledge in the form of a hypothesis that the connected components are also different segments.

*Eigenvector grouping (EVG):* This grouping scheme, proposed by Rygiel et al. [48], attempts to exploit the tubular structure of the vessels by performing the grouping along the vector estimating the vessel direction. The directional vector of the vessel is computed by extracting the top eigenvector of the local root point neighbourhood, hence the method name. The top eigenvector defines the direction of the most variance in the local point cloud, thus when paired with the tubular nature of the point clouds, allows to perform point grouping along the vessel. Fig. 3.6 showcases how the EVG grouping areas stretch along the vessels' directions, allowing for discrimination between disconnected vessel parts belonging to the same



segment, which may happen due to low quality of prior segmentation, and other segments.

The discussed domain-specific adaptations of PointNet++ showcase how the prior domain knowledge can be incorporated towards the design of the neural network. Even though these approaches are not specifically designed for the segmentation task, they are still of interest and are worth exploring in the task of hemodynamics estimation which we undertake in this work. We will discuss more in detail the applicability of these methods to the task at hand in Chapter 4.

### 3.4 Summary

In this chapter, we introduced various 3D data representations and discussed the principles of learning robust and stable representation by presenting the Geometric Deep Learning (GDL) blueprint. Through the lens of the GDL, we introduced how the learning on point clouds does look like, by presenting PointNet and PointNet++ architecture. At last, the domain-specific adaptions of generic PointNet++ have been showcased for the learning on vascular trees, which are of interest in this work. In the next chapter, we will move on to presenting the proposed approach to the estimation of FFR in coronary arteries, which is based upon the architectures and principles discussed in this chapter.

# Proposed approach to the estimation of FFR in coronary arteries

In this chapter, we will introduce the approach to the estimation of FFR in coronary arteries that is proposed in this work. We will showcase the whole workflow, by describing the process of data preparation and DNN design. At last, the training and inference setups will be discussed together with the details and specifics of the implementation.

## 4.1 Proposed approach & workflow

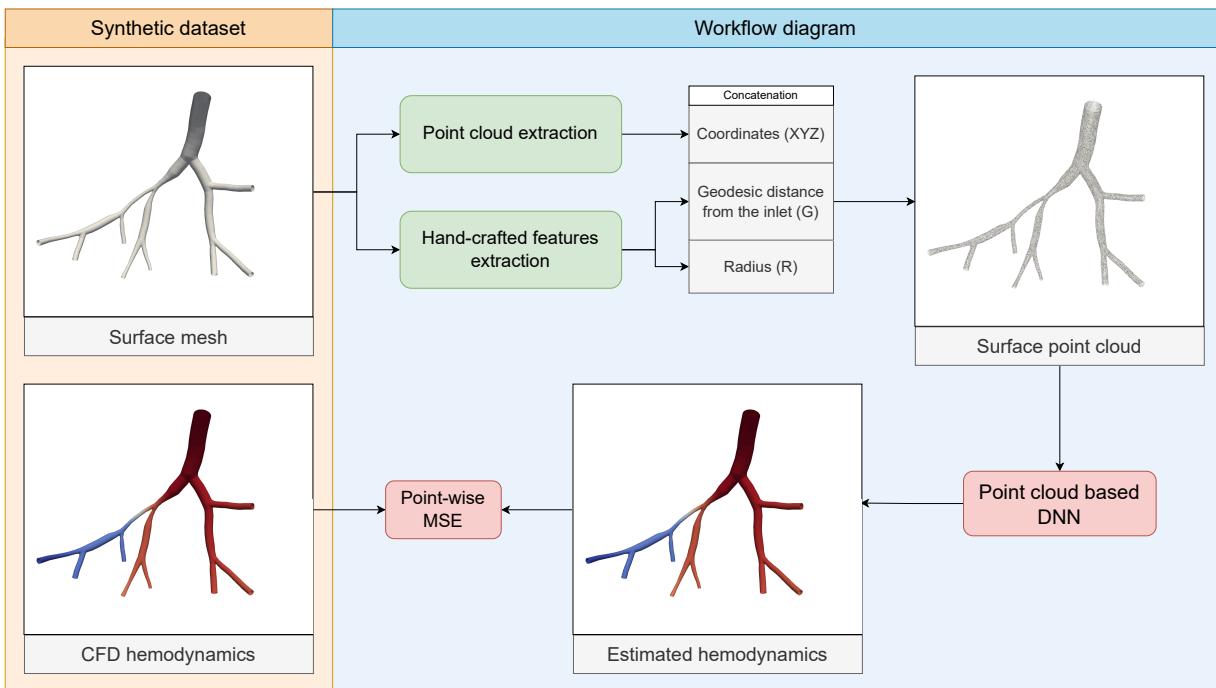


Figure 4.1: Workflow diagram of the proposed approach.

As described in Chapter 2, there are two common approaches towards learning on geometries for the task of hemodynamics estimation: explicit and implicit feature learning. In this work, we propose estimating FFR in coronary arteries, mainly focusing on implicit feature learning by representing the input vessel geometry as a 3D point cloud. However, we also propose to compute some generic hand-crafted features to further enhance the features of the input points. We argue, that the composition of both implicit and explicit feature learning is the most beneficial approach when dealing with the estimation of hemodynamic features. The usage of implicit learning allows a DNN to encode the input geometry with respect to the hemodynamic feature at hand. Moreover, consuming raw point clouds allows for including fine-grained

resolution of the surface in the learning procedure. Such details are almost impossible to be captured with any of the hand-crafted features. By utilizing hand-crafted features, we equip the network with some additional information that is not trivially extracted from the geometry only. The design of such features is based on prior knowledge regarding fluid dynamics and the geometry features that do correlate with it. The chosen features will be described in detail in Section 4.3.1.

Fig. 4.1 showcases the workflow diagram of the proposed approach. We follow the works by Itu et al. [26] and Suk et al. [52, 51] and construct the dataset of synthetic coronary arteries to both perform the training and evaluation, due to the lack of a representative and large enough database of real-patient geometries. The geometries of coronary arteries are represented in the form of a surface mesh and for each, the vFFR GT is computed via CFD simulation. The input mesh is decomposed into the point cloud and hand-crafted features which are later concatenated to form the input features for the points. The point cloud constructed in this matter is then processed with a point cloud based DNN and as the learning criterion, a simple mean squared error (MSE) between the estimated hemodynamics and CFD-based ones is utilized. In this work, the estimated hemodynamic is vFFR, however, we do not directly estimate it with the DNN. Instead, we estimate the so-called *pressure drops*, which inform how much the pressure of the fluid drops along the vessel. Since the FFR is a ratio of pressures, it can be directly computed by prescribing the patient-specific pressure at the vessel inlet and computing the following pressures with the usage of pressure drops (this process will be further discussed in Section 4.2.2). This is a high-level description of our approach. In the next sections, we will describe each workflow step in detail.

Most importantly our proposed approach allows us to drastically limit the time required for vFFR estimation in comparison to CFD simulation. The estimation of vFFR with CFD engine (one utilized in this work, more information about will be provided in Section 4.2.2) takes up to **2h** for single geometry, meanwhile, our proposed approach requires only around **15s** for single geometry.

## 4.2 Data preparation

As we briefly mentioned previously (see Section 4.1), there aren't any available large enough databases of real-patient geometries of coronary arteries. This fact renders the training on only real-patient unfeasible and thus the incorporation of synthetic data is required. For this work, we generated a dataset of 1700 synthetic coronary artery geometries and labels in the form of pressure drop maps obtained via CFD simulations.

### 4.2.1 Generation of synthetic coronary artery geometries

In this work, we utilize an undisclosed in-house synthetic coronary artery generator developed by Patryk Rygiel at Hemolens Diagnostics sp. z o.o.<sup>1</sup>. We cannot share much detail about the method, due to the information being company know-how. However, we can disclose that the method is derived from the work by Itu et al. [26], and discuss the set of generic steps necessary to obtain biologically proper synthetic coronary artery models.

The generation of synthetic vessels can be split into two general steps: *vessel centerline* and *vessel surface* generation.

**Vessel centerline generation:** The centerline is a weighted shortest path traced between two extremal points [55] - in the case of the vessels as the points we consider a vessel inlet and outlets. The path needs to be set within the vessel, and at each point be at approximately equal distance from all the surface points in the perpendicular plane to the centerline at this point. Intuitively, we think of a centerline as a line that describes the vessel skeleton which is represented as a graph. In the most common cases, the centerline is extracted from the vessel geometry, as the process of vessel analysis. However, during the generation

<sup>1</sup><https://hemolens.eu/>

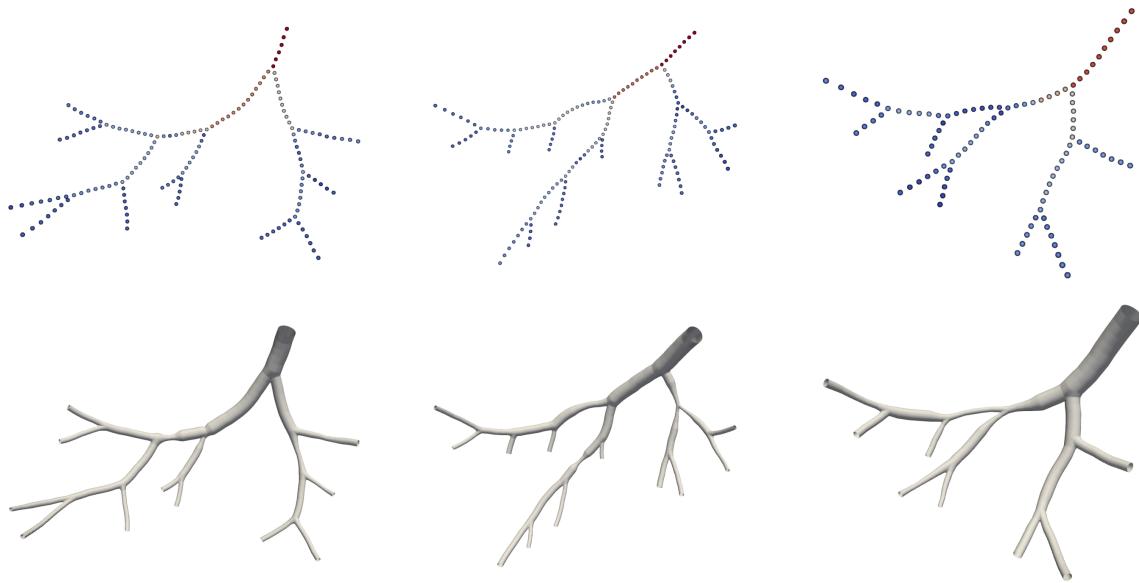


Figure 4.2: Samples from the synthetic artery generator. The first row showcases generated centerline graph with colour describing the radii at the nodes (from red (large) to blue (small)). In the second row, the surface meshes created based on the centerlines are presented.

process, we, first of all, generate a biologically proper centerline and later generate a vessel surface around it. For the synthetic vessel to be topologically correct and biologically relevant, the centerline is generated based on the anatomical atlases and studies [26, 34] which provide intervals of various coronary artery geometrical features e.g. radii, tapering or bifurcation angles. Since the atlases provide only the details about the healthy coronary trees, we must further enhance the synthetic data with the pathological cases. Such cases are constructed by generating stenotic segments along the centerlines according to the Coronary Artery Disease - Reporting and Data System (CAD-RADS) [38] stenoses percentage intervals. The first row of Fig. 4.2 showcases examples of generated centerlines, where the colours of nodes denote the radius of the vessel (from the largest (red) to the smallest (blue)).

**Vessel surface generation:** In the second step, based on the generated centerline, a 3D surface mesh is created. The construction of the surface mesh can be easily done with the usage of tools available in the Vascular Modelling Toolkit (VMTK) library [3], which is the common choice when dealing with the generation of synthetic vasculature. The second row of Fig. 4.2 showcases examples of surface meshes constructed based on the centerlines (row above) generated in the previous step.

Each synthetic vessel in the dataset was generated in the described two-step process with the topology, geometrical features and stenotic areas sampled randomly based on the provided value ranges. The generation of one sample took approximately 2.30min.

#### 4.2.2 CFD simulations

As we already mentioned in Sections 1.2.1, 2.1 and 2.2, the most common way of obtaining the ground truth labels for hemodynamics estimation is to run a CFD simulation. We take the same approach in this work and utilize a CFD engine by Kosior et al. [31] to run *stationary* blood flow simulations. By the *stationary* we consider the simulations when the velocity, pressure and fluid properties remain constant at any point in the flow field [2] - basically, they do not change in time. In this work, we treat the CFD engine as a black box and do not delve into the specifics of the algorithm, for the details the reader is referred to the cited patent [31].

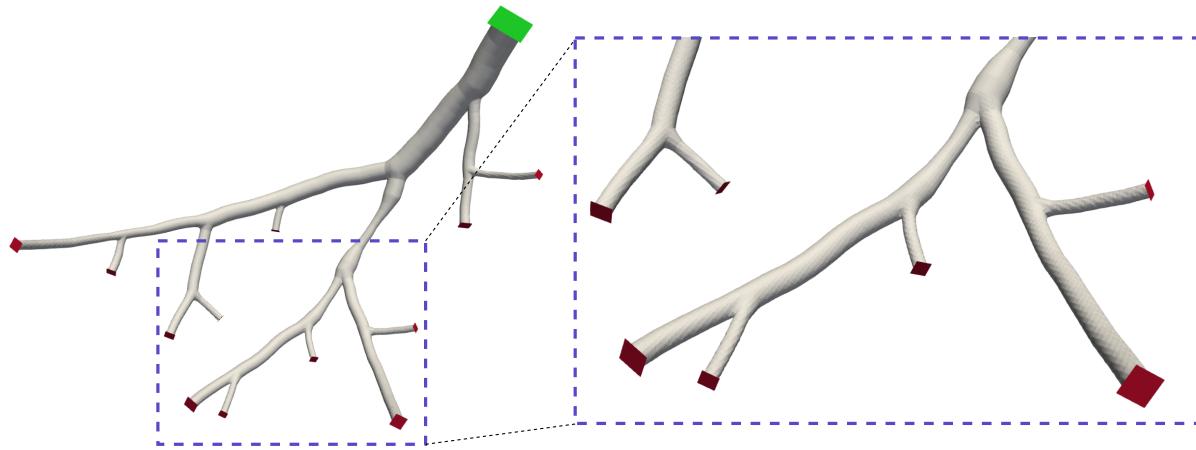


Figure 4.3: Placement of inlets and outlets for the CFD simulation. The green plane denotes the inlet through which the fluid comes into the system. The red planes denote the outlets through which the fluid leaves the system.

The input to the CFD engine is a vessel geometry represented as a surface mesh, inlet and outlet planes which describe respectively the place where the fluid comes into the system and where it leaves it (see Fig. 4.3), and a single number representing the prescribed flow of the blood at the vessel inlet given in ml/s (referred to as inflow). The inflow value, simply means how much blood is coming into the system, and since the simulation is stationary this value is constant across the time. For each vessel, we run three individual CFD simulations with inflow values from a biologically relevant range: 3, 5 and 7 ml/s, to simulate a patient under rest, mild exercise and high-intensity exercise conditions [33]. One simulation takes approximately 2 hours on the off-the-shelf processor with 16 processes.

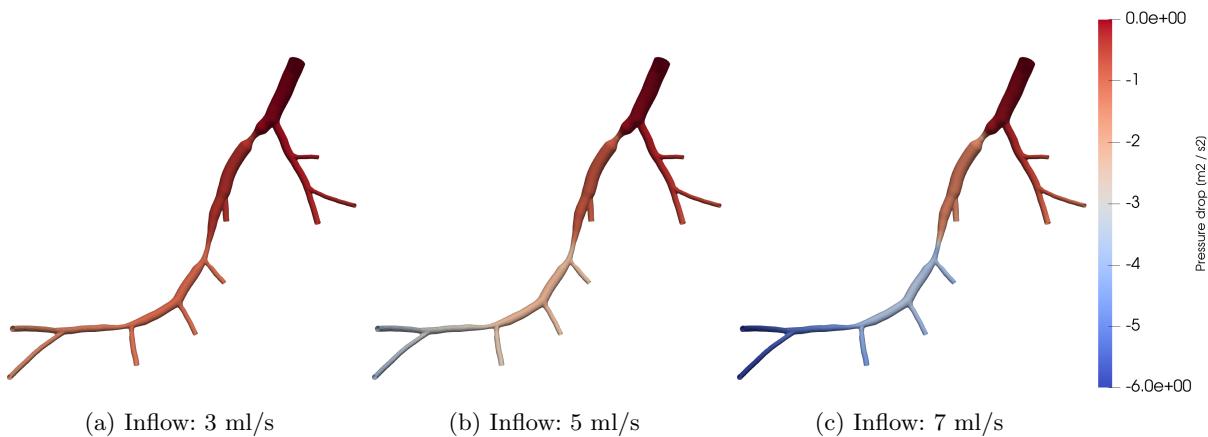


Figure 4.4: CFD simulation for various inflow values. The larger the amount of blood coming through the inlet, the larger drops of the pressure at the stenotic segments. Intuitively, there is a cap on the amount of blood that can go through the stenotic segment in a set amount of time. Thus, the more blood there is in the system the more of it gathers before the stenosis, hence the higher pressure is induced and the drop after the stenosis is larger.

As we already mentioned in Section 4.1, as the labels for a deep learning method, we extract *pressure drops* for each surface point from the CFD simulation. A *pressure drop* informs how much of the pressure of the fluid is lost as it travels along the vessel. The value of the pressure drop at the vessel inlet is 0 and the higher the drop the more negative it gets (Fig. 4.4) showcases how the values of pressure drops

are distributed along the vessel geometry and how they are impacted by the stenotic segments for various inflow values). To calculate the vFFR based on the pressure drops, the patient-specific pressure at the inlet needs to be prescribed. The value of the pressure for each point is computed by summing the inlet pressure with the point's pressure drop. Then the target vFFR at the given point is simply a ratio of the pressure at this point to the inlet pressure. The AI-based vFFR estimation is computed in the same way but based on the pressure drops obtained from an AI model.

## 4.3 Estimation of vFFR with point cloud based neural networks

In this work, as already mentioned in Section 4.1, we propose to utilize a hybrid approach to learning both on implicit and explicit features. The explicit features are provided as the input point features and the implicit ones are extracted by the point cloud based neural network. The process of point cloud construction and specifics of both utilized deep learning architectures and training regimes are provided in the following sections.

### 4.3.1 Point cloud construction

As highlighted in Fig. 4.1, the construction of the point cloud is done based on the input mesh representing the vessel surface. The mesh is constructed out of nodes with 3D coordinates and edges which convey the connectivity of the surface. In the *point cloud extraction* step (see Fig. 4.1) the edges of the mesh are stripped and only the nodes with 3D coordinates (denoted XYZ) are kept. In the *hand-crafted features extraction* step (see Fig. 4.1), the two features are extracted from the surface mesh: *geodesic distance from the inlet* (denoted G) and *radius* (denoted R).

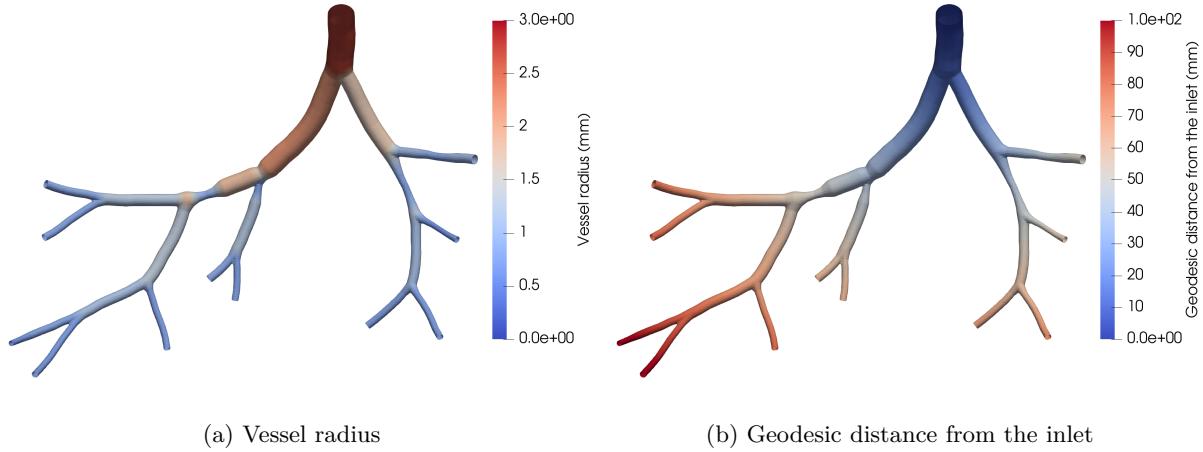


Figure 4.5: Additional hand-crafted features. (a) The vessel radius feature informs how wide is the vessel at the given point. (b) The geodesic distance from the inlet feature informs how far along the vessel surface the given point is from the inlet plane.

*Geodesic distance from the inlet:* In the coronary tree, the direction of the blood flow is set - the fluid moves from the inlet to the outlets. This information is crucial when estimating the hemodynamic features since the behaviour of the system is totally different when the fluid comes from a different side. Thus, to incorporate the information about the blood flow direction, in the form of a hand-crafted feature, we equip each surface point with its geodesic distance along the surface mesh from the inlet plane. The geodesic distance along the surface mesh is simply the shortest path between the given pair of points in the graph representing the surface mesh. This feature allows us to determine how far the given point is from the vessel inlet, and it also is continuous, thus even in the local vessel segment, the direction of the blood flow

is provided. Fig. 4.5b showcases an example of the coronary tree with the colours denoting values of the geodesic feature given in mm.

*Radius:* The second feature that we provide to the network explicitly is the vessel radius. For each surface point, its shortest Euclidean distance to the centerline is computed and denoted as the radius feature. This feature simply informs how wide the vessel is at the given point. One could argue that such a feature should be easily learnt implicitly by the neural network given the vessel geometry. Although, obviously, such a feature is learnt in some form implicitly, the fine-grained precision of it might be lacking. When incorporating radius explicitly as the input feature, we provide the network with an exact value of the narrowing which allows for a better estimation of the pressure drops in these key points. Fig. 4.5a showcases an example of the coronary tree with the colours denoting values of the radius features given in mm.

When extracted, both coordinates (XYZ) and hand-crafted features (R, G), are concatenated together to form the 5-element tuple (X, Y, Z, R, G) representing a single point. The point cloud constructed in such a manner is then provided as input to the point cloud based DNN.

### 4.3.2 Architectures

In this work, as the point cloud neural network, we utilize PointNet++ architecture (for the detailed description please refer to Section 3.3). We tested two different configurations of the network, namely usage of standard grouping technique multi-scale grouping (MSG) and eigenvector grouping (EVG), described in Section 3.3.3. The geometry-aware grouping (GAG) which has also been described in Section 3.3.3 is not applicable to the problem at hand. It is mainly tailored towards the analysis of point clouds with disconnected components. Meanwhile, in our task of hemodynamics estimation, the vessel is a single non-disconnected component, which renders GAG to be non-different from MSG. The difference between MSG and EVG in the process of hierarchical point cloud encoding is showcased in Fig 4.6. When utilizing Euclidean distance to group (MSG), in a single neighbourhood multiple branches might be present. Meanwhile, with EVG the neighbourhood mostly is stretched along a single branch.

For both grouping methods, we employ the same general architecture setup showcased in Fig. 4.7. The network is built out of 7 SA blocks which hierarchically perform feature extraction and downsampling for the point cloud to be finally encoded as a 1D feature vector of size 256. The decoder branch is built out of 7 FP blocks, and topped with two shared MLP layers to output the desired number of features per point  $F_{out}$ . In our case,  $F_{out} = 1$  since we only estimate scalar pressure drop values, and  $F_{in} = 5$ , due to the usage of 3D coordinates (XYZ) and two hand-crafted features (R, G) described in detail in Section 4.3.1. The information extracted from the following SA blocks is passed to decoder ones to perform feature interpolation and upsampling. For MSG and EVG configuration, only the grouping procedures in SA blocks are changed.

The EVG configuration utilizes MSG grouping in the first block and only EVG in the following ones. The neighbourhoods in the first block are small and do not span the vessel along its direction that much, thus there's no need for directional grouping. For both MSG and EVG configurations, we employ the multi-scale grouping scheme (see Section 3.3.2) with 2 scales and denote hyperparameters for these scales with respective subscripts 1 or 2. The MSG method is configured with a single hyperparameter, namely grouping radius  $r$ . The EVG one takes additional 2 parameters:  $k$  - number of nearest neighbours defining the prior neighbourhood to compute the eigenvector of, and  $L$  - a length of the grouping vector. For the EVG method, we set the following parameters to be constant across the encoder blocks:  $L_1 = 0.001$ ,  $L_2 = 0.002$ ,  $k_1 = 64$  and  $k_2 = 64$ . The radii  $r_1, r_2$  change between the blocks and are the same for both MSG and EVG - the exact values are provided in Table 4.1.

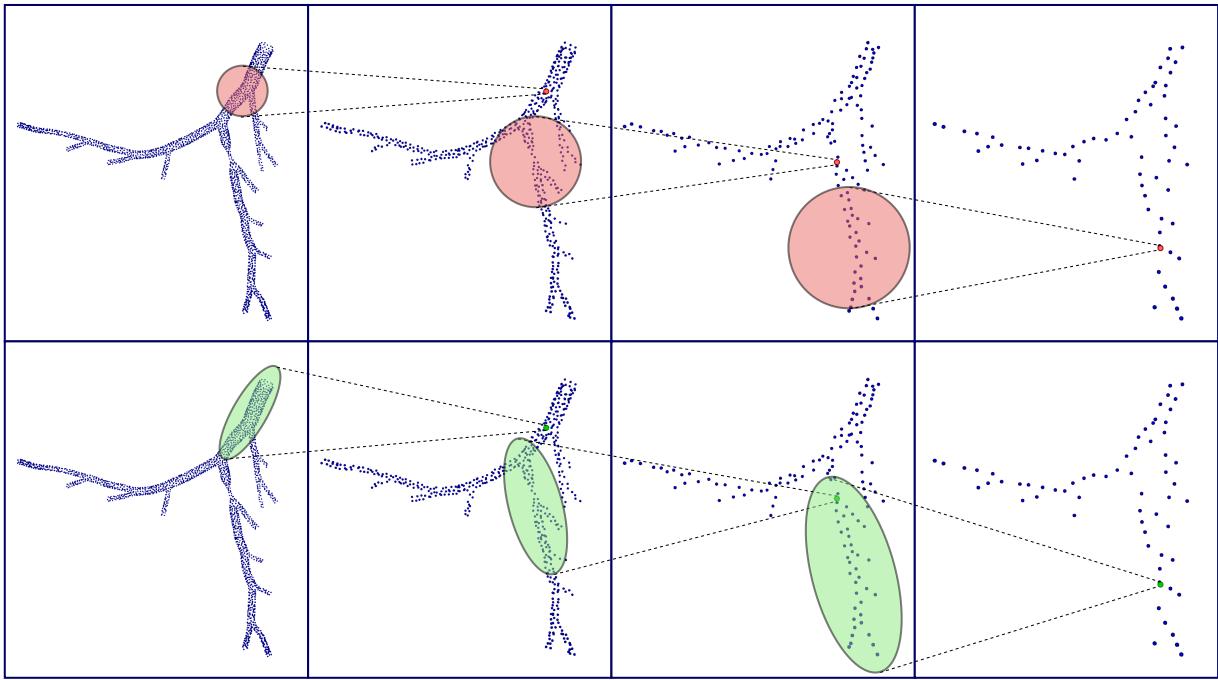


Figure 4.6: Grouping methods for the vFFR estimation. The figure showcases the process of hierarchical point cloud encoding in PointNet++. In the first row, the MSG method is utilized - the points are grouped based on the Euclidean distance. In the second row, the EVG method is utilized - the points are grouped along locally dominant eigenvectors, which estimate the local vessel direction.

Table 4.1: Values of the grouping radii in the MSG and EVG PointNet++ configurations. SA@n denotes the following SA-blocks in the encoder branch, and the *ALL* marker denotes that all the remaining points are grouped together.

Radius	Encoder						
	SA@1	SA@2	SA@3	SA@4	SA@5	SA@6	SA@7
$r_1$	0.04	0.04	0.08	0.16	0.24	0.6	<i>ALL</i>
$r_2$	0.08	0.08	0.16	0.24	0.32	0.8	<i>ALL</i>

### 4.3.3 Training & Inference setup

We split the generated synthetic dataset of 1700 samples into 3 subsets train, validation and test with 1500, 100, and 100 samples respectively. We utilize the same split in each experiment.

**Preprocessing:** During the preprocessing stage, before either the training or inference pipeline, the point cloud and hand-crafted features are extracted based on the provided surface mesh. The obtained point cloud  $\mathcal{P}$  is centred around the origin of Euclidean space and resized to fit the cube centred around the origin as well, with the edge of length 1:  $\mathcal{P}_{\text{resized}} \in [-0.5, 0.5]^3$  (values for the radii in Table 4.1 are provided in this resized metric space). For the sake of time efficiency, the preprocessed data is cached on the drive.

**Training setup:** During training, in each iteration, the input point cloud is subsampled to have exactly 20,000 points. The resolution of the original mesh is about 200 – 400k nodes, which is far too memory and time-exhausting. Subsampling to about 5 – 10% of the original resolution is a good tradeoff between the efficiency and quality of the results. As the loss function, we utilize a mean squared error

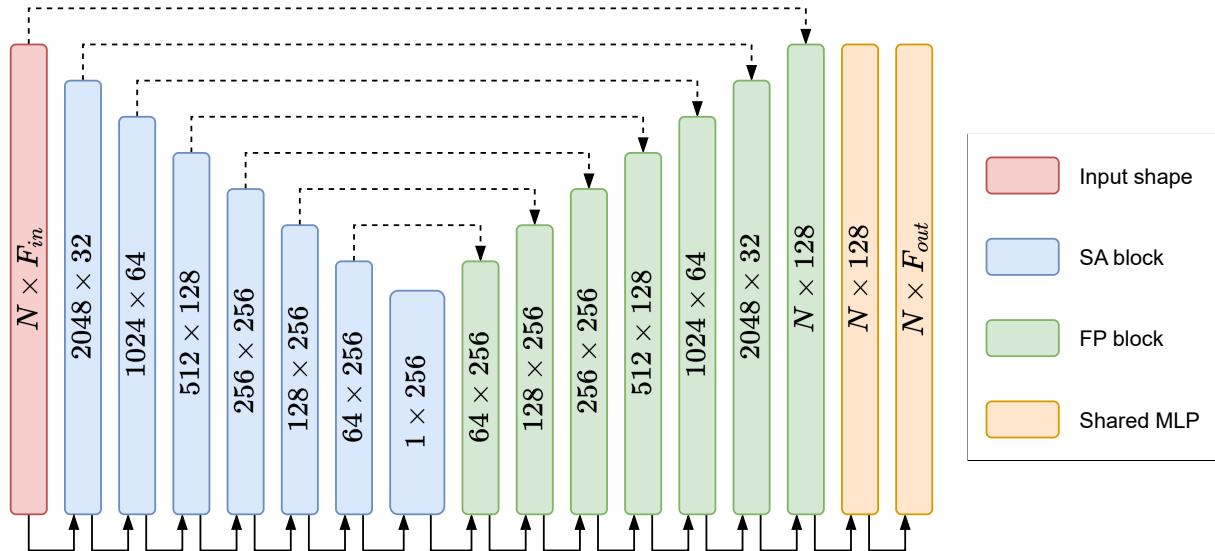


Figure 4.7: PointNet++ architecture details. We employ the general architecture which consists of 7 pairs of SA and FP blocks, and a regression head built out of 2 shared MLP layers.

(MSE) which is calculated point-wise between pressure drop labels and estimation from the model. The models are trained for 500 epochs with the batch size of 8 and for the optimizer we employ Adam with a constant learning rate of 0.001. For augmentations, we utilize random rotations of any angle in all planes.

**Inference setup:** For the inference to be performed on the whole point cloud, not only a subsampled one, as done in the training pipeline, we adopt the scheme proposed by Balsiger et al. [4]. The input point cloud is split randomly into chunks of 20,000 points, and for each one, a different inference is run. For the results to be more comprehensive, this procedure is performed 10 times and final per-point estimations are an average along all the runs.

#### 4.3.4 Implementation

The models are implemented in Python with the usage of PyTorch [39], PyTorch Geometric (PyG) [17] and pytorch-lightning [16] libraries. The CUDA kernel supporting the EVG grouping scheme was developed to be compatible with the PyG library and it is provided in the attached CD ??.

### 4.4 Summary

In this chapter, we introduced and described in detail our proposed approach to the estimation of vFFR in the coronary arteries. We discussed the reasoning behind combining explicit and implicit feature learning to perform hybrid and robust surface encoding. The process of the synthetic dataset generation was described, and the methods of obtaining the geometry and CFD labels were discussed. Finally, the employed architecture configurations of PointNet++ were introduced with the explicitly stated hyperparameters. At last, we described the training and inference pipelines and mentioned the implementation details. In the next chapter, we will describe the performed evaluations, of the proposed approach, together with the results and the discussion of limitations and future works.

# Experiments & results

In this chapter, we describe the performed evaluations of the proposed approach and report obtained results. The experimental settings are tailored towards evaluating the generalization of the method for the various blood flow characteristics and stenosis grades. Moreover, since the method is to be used in the clinical setting to qualify the patient for the invasive treatment, we showcase its clinical viability by performing classification under the FFR clinical threshold of 0.8. At last, we showcase the qualitative comparisons between the AI and CFD-based simulation results and discuss common issues and limitations.

## 5.1 Metrics

The problem of vFFR estimation is a per-point regression task, thus to evaluate the models' performance we utilize three common regression metrics:

$$\text{Mean Absolute Error (MAE)} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5.1)$$

$$\text{Normalized Mean Absolute Error (NMAE)} = \frac{1}{ny_{max}} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5.2)$$

$$R^2 \text{ score (R2)} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i \quad (5.3)$$

where  $y$  are ground truth labels,  $\hat{y}$  network predictions. The  $y_{max}$  denotes a maximum value label over the whole test dataset, which serves as a normalization term.

In this work, we also utilize well-known classification metrics, to perform the evaluation of the clinical viability of the proposed approach: accuracy, precision, recall and F1 score.

## 5.2 Quantitative results

We perform a wide series of quantitative evaluations to test the robustness, generalization and clinical applicability of the proposed approach. As mentioned in Section 4.2.2, we generated a 3 set of ground truth labels for various values of blood inflow from the biologically relevant range: 3, 5 and 7 ml/s. The evaluation of network performance on these various settings of boundary conditions is relevant in the light of real-life applications. All experiments provided in the following sections are evaluated on the same test set of 100 geometries and for all 3 inflow settings. The experiments' results are reported for PointNet++ in two configurations, namely MSG and EVG, described previously in detail in Section 4.3.3.

### 5.2.1 Evaluation under various blood flow characteristics

First of all, we perform the evaluation of the estimation of the pressure drops for various inflow values. The results are provided in Table 5.1 and we report results for mean, median and 75th percentile for both NMAE and MAE metrics. As we can see, the results between the MSG and EVG methods for all inflow

$Q_{in}$  values are quite similar across the board. The percentile error reported by NMAE is almost identical across the  $Q_{in}$  values, meaning that the proposed approach has good generalization capabilities and can perform on the same level for datasets of different boundary conditions of blood flow simulation. Of course, when looking at MAE, the error rises, however since this is the absolute error and values of pressure drops are much larger for the higher inflow (see Fig. 4.4 for the explanation of this phenomena), such behaviour is expected. Thus we consider the NMAE metric to be a more comprehensive one for the problem at hand.

Table 5.1: Evaluation of pressure drops estimation. We report mean, median and 75th percentile of NMAE and MAE metrics for inflow  $Q_{in}$  values of 3, 5 and 7 ml/s. We evaluate two PointNet++ configurations, namely MSG and EVG, and denote them accordingly. The NMAE metric is a percentage-based metric and MAE is provided in kinematic pressure units  $m^2/s^2$ .

Model	$Q_{in}$ (ml/s)	NMAE (%)			MAE ( $m^2/s^2$ )		
		mean	median	75th	mean	median	75th
MSG	3	2.09	0.79	<b>1.44</b>	0.29	0.11	<b>0.20</b>
EVG	3	<b>1.85</b>	<b>0.74</b>	1.62	<b>0.26</b>	<b>0.10</b>	0.23
MSG	5	<b>1.40</b>	<b>0.60</b>	<b>1.54</b>	<b>0.45</b>	<b>0.19</b>	<b>0.49</b>
EVG	5	1.83	0.66	1.59	0.58	0.21	0.51
MSG	7	<b>1.63</b>	<b>0.50</b>	<b>1.55</b>	<b>0.90</b>	<b>0.28</b>	<b>0.86</b>
EVG	7	1.80	0.66	1.58	1.00	0.36	0.88

Table 5.2: Evaluation of vFFr estimation. We report mean, median and 75th percentile of MAE metric for inflow  $Q_{in}$  values of 3, 5 and 7 ml/s, and input pressure  $p_{in}$  of 80, 100 and 120 mmHg. We evaluate two PointNet++ configurations, namely MSG and EVG, and denote them accordingly.

Model	$Q_{in}$ (ml/s)	MAE (10e2)								
		$p_{in} = 80$ mmHg			$p_{in} = 100$ mmHg			$p_{in} = 120$ mmHg		
		mean	median	75th	mean	median	75th	mean	median	75th
MSG	3	2.95	1.11	<b>2.03</b>	2.27	0.85	<b>1.56</b>	1.97	0.74	<b>1.35</b>
EVG	3	<b>2.61</b>	<b>1.04</b>	2.28	<b>2.01</b>	<b>0.80</b>	1.75	<b>1.74</b>	<b>0.69</b>	1.52
MSG	5	<b>4.45</b>	<b>1.91</b>	<b>4.89</b>	<b>3.42</b>	<b>1.47</b>	<b>3.76</b>	<b>2.97</b>	<b>1.27</b>	<b>3.26</b>
EVG	5	5.85	2.11	5.08	4.50	1.62	3.91	3.90	1.41	3.39
MSG	7	<b>9.04</b>	<b>2.76</b>	<b>8.61</b>	<b>6.95</b>	<b>2.12</b>	<b>6.62</b>	<b>6.03</b>	<b>1.84</b>	<b>5.74</b>
EVG	7	10.02	3.65	8.78	7.71	2.81	6.75	6.68	2.43	5.85

The evaluation of clinical applicability is hard to be done based on the estimation of pressure drops since they serve only as an intermediate step in the calculation of relevant biomarker FFR. Thus in Table 5.2, we report the quantitative results obtained for the vFFR estimation for various inflow and input pressure values prescribed for vFFR calculation (see Section 4.2.2 for the in detail description of obtaining vFFR from pressure drops). The vFFR takes values from between 0 and 1, and we report the MAE metric and all the values in the table are provided in the scientific notation of 10e2 for the readability sake. We see that the results between the MSG and EVG configurations do not vary much, however, for larger inflows the MSG seems to behave a bit better. Across the board, the mean MAE is larger than the median and the 75th percentile. This is due to the fact, that there are a few very pathological cases in the test dataset, for which the pressure drops are very large, and the estimations of vFFR have quite a large error - we will

discuss this issue more in detail in Section 5.4). However, across the board, the absolute error of estimated vFFR is kept under 0.1 and for more than half of the dataset, the error is almost negligible - below 0.03. Obviously, for larger input pressures the error gets smaller since the pressure drops are relatively smaller. These are promising results which convey that the proposed approach can be easily trained for different boundary conditions of underlying fluid simulation without a noticeable drop in percentage performance.

### 5.2.2 Evaluation of stenosis grade impact

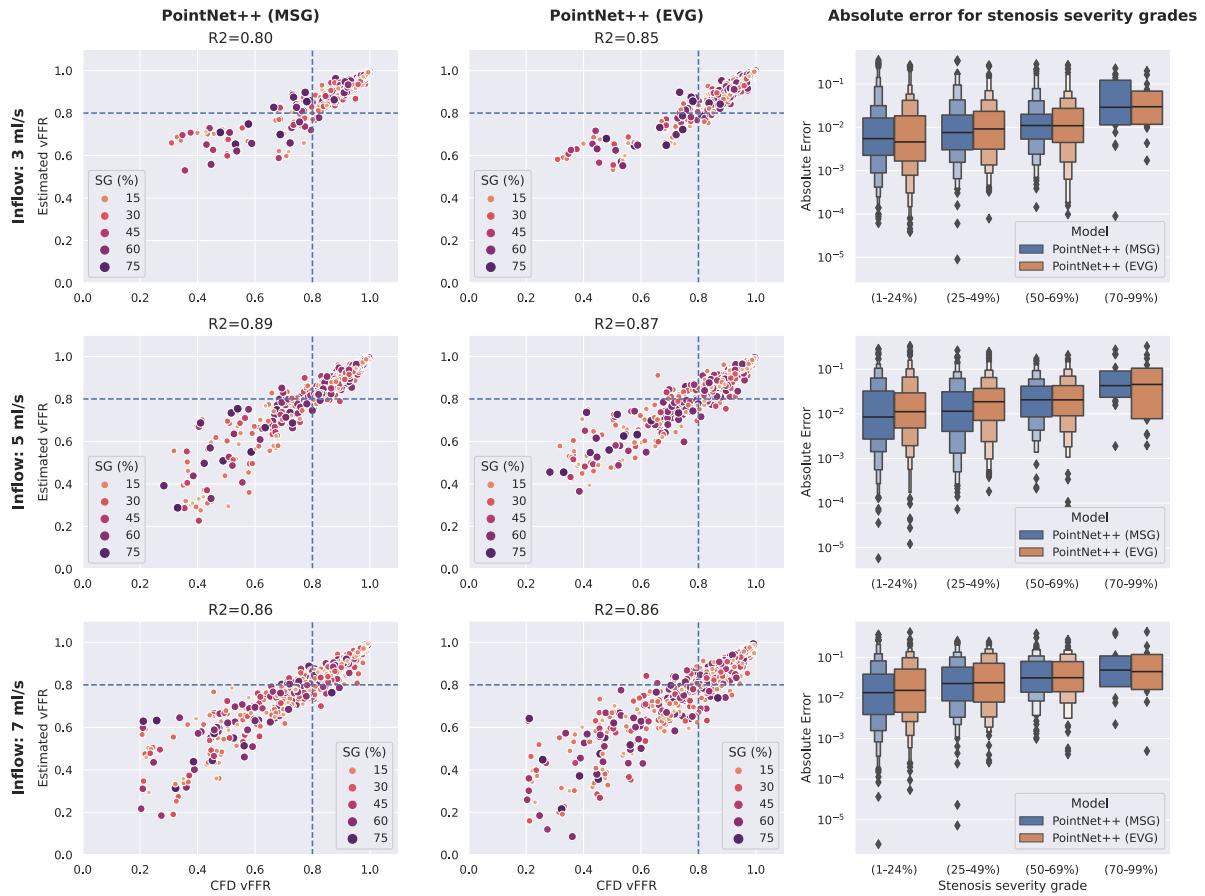


Figure 5.1: Evaluation of stenosis grade (SG) impact. The scatter plots showcase the correlation between AI and CFD-based vFFR estimations. The blue dashed line denotes the clinical FFR threshold of 0.8. The boxplots showcase the absolute estimation errors grouped based on the CAD-RADS scale of stenoses' severity grades.

In the second experiment scenario, we want to evaluate the method's correlation with the CFD-based FFR and its performance with respect to the stenosis grade. We extract the stenotic segments from the test set (about 700 samples) and label them with their percentage severity grade (SG). This time we report results only for the average input pressure  $p_{in}$  of 100 mmHg. In Fig. 5.1, we showcase the correlation plots between AI and CFD-based vFFR estimations for MSG and EVG model configurations and various inflow values. The stenoses' severity grades (SG) are marked with markers of varying size and colour, and the clinically accepted FFR threshold of 0.8 is marked with the blue dashed line. Again, both model configurations seem to behave quite similarly, and the R2 scores are kept above 0.8 across the board. For the inflow of 5 ml/s we report the highest R2 scores of 0.89 (MSG) and 0.87 (EVG), which showcases that the method has very high correlations with CFD-obtained vFFR. Moreover, as we can see the correlation in the decision margin around the FFR clinical threshold of 0.8 is of good quality (more on the clinical

viability will be discussed in Section 5.2.3). The errors on the stenoses well outside the decision range are much less impactful, since whether the patient's FFR is estimated as 0.3 or 0.5, the clinical decision regarding the treatment is the same. In the third column, we showcase the boxplots of the absolute errors with respect to stenoses' severity grades. The severity grades are grouped based on the CAD-RADS scale [38]. We report, as expected, that the errors grow larger with the stenoses severity grade. For all the grades, models and inflow configurations, the error is kept under 0.1 despite a few outliers. For smaller severity grades, the mean error is somewhere around 0.01 – 0.02. These results prove that the method is robust towards the estimation of the impact of various stenoses' grades without much loss in performance.

### 5.2.3 Evaluation under clinical viability

In this experiment, we evaluate the clinical viability of the proposed approach. Following the well-known Latin phrase "*Primum non nocere*" (eng. first, do no harm), which serves as the fundamental principle in healthcare, we need to evaluate whether the proposed approach is safe for clinical use, or at least does not cause any unnecessary risk for the patient.

To do that, we report the classification results under the clinically accepted FFR threshold of 0.8, of whether to perform the treatment or not. The results for various inflow and model configurations are showcased in Table 5.3. As we may see, the results are quite high across the board, not dropping below 90%. The precision scores seem to be a bit higher for the EVG method, with all the other metrics being in favour of MSG. For higher-intensity exercise conditions, reflected with higher inflow values, we see a drop in the performance of the proposed approach. These are more extreme blood flow characteristics than usual, however, when dealing with patients with suspected CAD, such conditions are not that uncommon. Although this aspect does not limit the clinical viability of the approach that much, it should be reported and known to the medical specialist utilizing the system for the treatment decision process.

In this work, we utilize only synthetic data for both training and testing of the proposed approach. Obviously, for proper and safe usage in the clinical routine, the approach would need to be trained and evaluated on real patient geometries.

Table 5.3: Evaluation under clinical viability. We perform the classification of whether to perform the treatment or not, by thresholding the vFFR based on the clinically accepted threshold of 0.8.

	$Q_{in}$ (ml/s)	Accuracy (%)	F1-score (%)	Precision (%)	Recall (%)
<b>MSG</b>	3	<b>95.66</b>	<b>97.57</b>	95.93	<b>99.27</b>
	3	95.02	97.19	<b>96.42</b>	97.96
<b>EVG</b>	5	<b>94.11</b>	<b>96.12</b>	<b>95.45</b>	<b>96.80</b>
	5	92.64	95.15	94.56	95.74
<b>MSG</b>	7	<b>92.37</b>	<b>93.91</b>	91.83	<b>96.07</b>
	7	91.10	92.63	<b>93.84</b>	91.46

## 5.3 Qualitative results

In Fig. 5.2 we showcase a few qualitative comparisons between ground truth labels obtained from CFD and from our proposed approach (in both PointNet++ configurations, namely MSG and EVG) and plot both vFFR predictions and MAE. All the rows utilize individual legend scales for better visualization of the intricacies of the predicted vFFR.

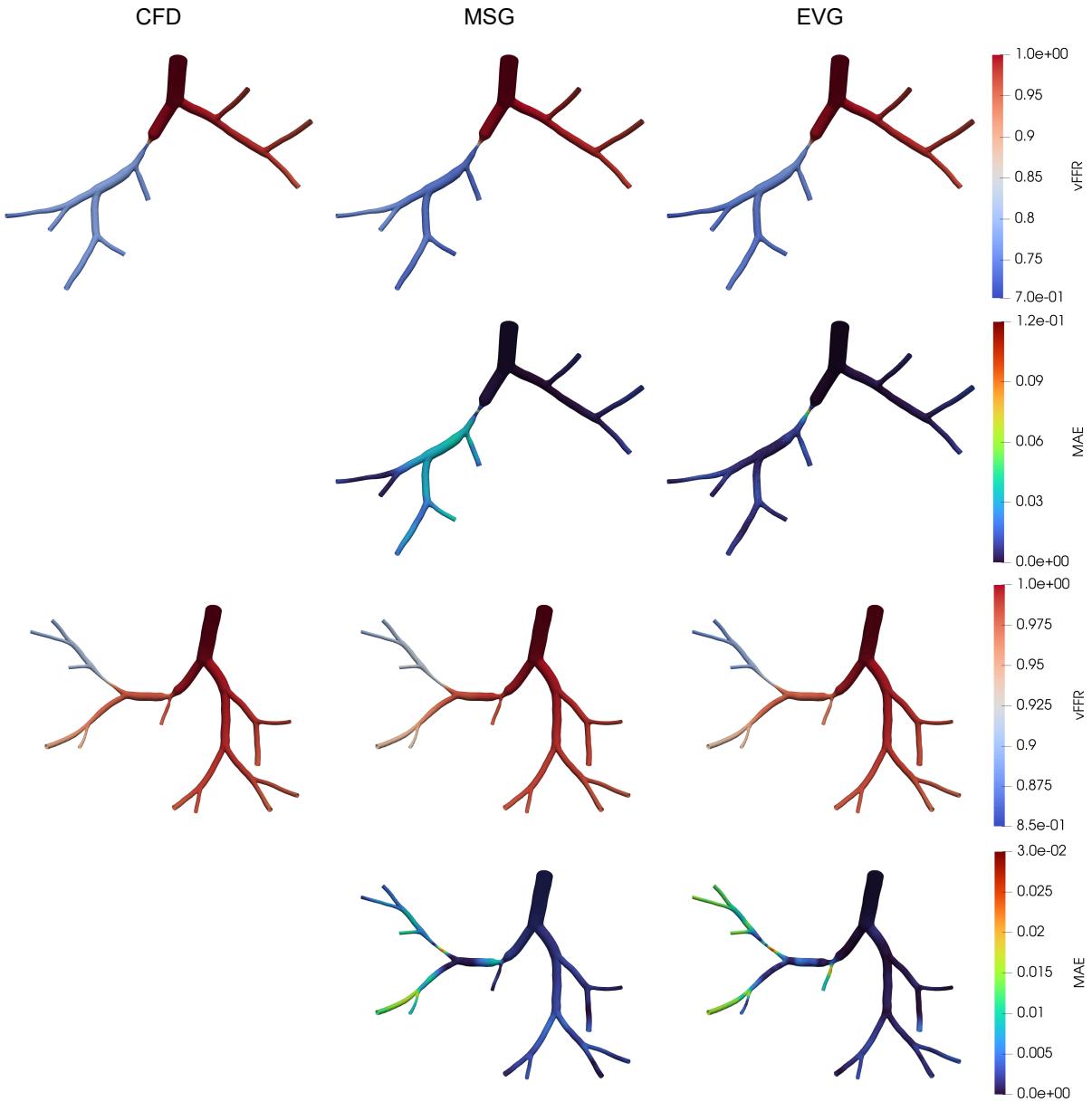


Figure 5.2: Qualitative results of vFFR estimation for  $Q_{in} = 3 \text{ ml/s}$  and  $p_{in} = 100 \text{ mmHg}$ . We visualize vessel geometries with estimations of vFFR and MAE for both tested PointNet++ configurations - MSG and EVG. For each row, the individual legend scale has been chosen to better visualize the intricacies between the methods.

For the first sample (rows 1 and 2) we see a very early, large 60% narrowing of the vessel lumen on the leftmost branch. This stenosis is an important one from a clinical point of view since it is close to the decision threshold of 0.8. The CFD-obtained vFFR reports vFFR of 0.75, and both MSG and EVG configurations yield similar results. For EVG and error is almost negligible, as seen on the MAE plot, and for MSG the error is in the range of 0.03. But most importantly, for both methods, the predicted vFFR is under 0.8, which is crucial in a clinical setting. The largest error is reported in the very middle of the stenosis, however, FFR is never estimated in such places but rather behind the stenosis where the blood dynamics are more stable.

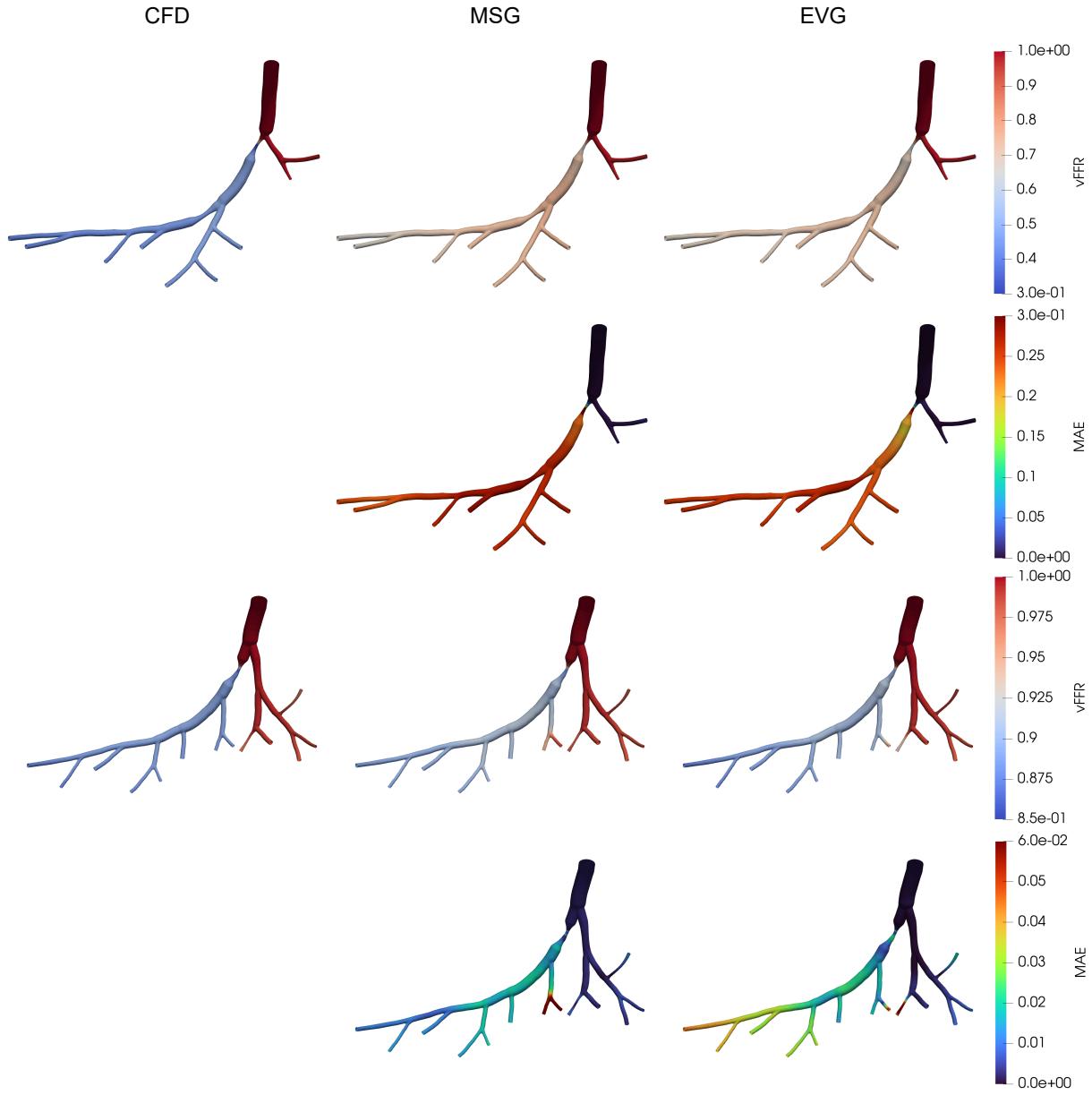


Figure 5.3: Issues with vFFR estimation for  $Q_{in} = 3 \text{ ml/s}$  and  $p_{in} = 100 \text{ mmHg}$ . We visualize vessel geometries with estimations of vFFR and MAE for both tested PointNet++ configurations - MSG and EVG. For each row, the individual legend scale has been chosen to visualize the intricacies between the methods better. The first sample showcases the problem with the underestimation of the vFFR after large stenosis. The second sample showcases the problem of "spilling" estimations of vFFR between two close branches.

The second sample (rows 3 and 4) has 2 medium-size narrowings on the leftmost branch, the first one of size 30%, and the second one of size 50%. In comparison to the first sample we discussed, the stenoses present here are not causing vFFR to drop below 0.8 - it is kept quite high almost above 0.9 for the whole artery. We see that EVG is a bit better at estimating the first narrowing, however, it lacks a bit of precision on the small branches for which the reported error is about 0.01. Both methods, however, correctly estimate the importance of the narrowings of the blood dynamics. For both showcased examples we see that the error on non-stenotic segments is almost negligible, meaning that the network is robust

enough to correctly grasp the notion of fluid dynamics in standard vessel geometries.

## 5.4 Problems & issues

The proposed approach achieves promising results in all evaluation scenarios. It generalizes well for various blood flow characteristics, achieves good correlation with CFD-estimated vFFR and is clinically viable under the assumption that the synthetic data can be replaced with real geometries without a large loss in estimation quality. However, there are some common problems and issues that can be observed in a few samples - these problems are showcased in Fig. 5.3.

One common issue, when estimating very large narrowings ( $> 70\%$ ) is underestimation - the network cannot correctly assess the impact of the stenosis. The example showcasing this problem is plotted in the first and second row of Fig. 5.3. The network spots the stenosis and considers it significant since the estimated vFFR is clearly below 0.8. However, it considers vFFR to be higher after the stenosis by almost 0.2 than it is in reality.

The second problem appears when there are two very close, in the Euclidean sense, branches in the vessel tree. In such cases, the estimated vFFR from one branch may "spill" onto another, causing the vFFR to rise, which is biologically impossible and simply incorrect. The example showcasing this issue is plotted in the third and fourth row of Fig. 5.3. A little branch in the middle of the tree for both MSG and EVG has increased vFFR more than the prior vessel segment. This is caused by the close presence of another branch on the right side with drastically different vFFR. The network has problems with such areas because points are considered neighbours based on the Euclidean distance and not proximity along the manifold, and the features are hierarchically aggregated in this manner.

## 5.5 Future works

This work serves as a proof-of-concept of utilizing point cloud based approaches to vFFR estimation. We plan on developing this approach further and thus outline below a few ideas that are to be addressed in the future:

- Collection of representative real-patient database to perform training and validation on.
- Incorporating information about the boundary conditions of the underlying CFD simulation into the neural network, to be able to train one model which can be parametrized in the inference setting.
- Enhancing the proposed approach with PINNs (physics-informed neural networks) to introduce a physics prior that can help with the problems mentioned in Section 5.4.

## 5.6 Summary

In this chapter, we reported the results of the proposed model evaluation for various experiment scenarios. We showcased that the method can be easily trained for the various underlying CFD simulations without loss of performance and that it achieves promising quantitative results. We evaluated the impact of the stenosis severity grade on the model prediction and calculated the correlation with CFD simulation. To prove the clinical viability of the model, the classification of whether to perform the treatment or not based on its predictions has been performed. We showcase some qualitative examples by plotting geometries with vFFR and highlight issues and problems with the estimations of vFFR with the proposed approach. At last we touched on future works, and describe what might be next steps of developing and further enhancing the proposed approach in this work.



# Summary

To conclude, in this work we propose a novel approach to estimating vFFR in coronary arteries with the usage of point cloud representation. We propose a hybrid approach of utilizing both implicit and explicit feature learning to encode the vessel geometry in a robust manner. To demonstrate the applicability of the proposed approach in the clinical practice we construct and run a series of evaluation scenarios. We prove that the proposed approach is a compelling replacement for commonly utilized CFD-based methods in this task, due to the achieved high correlation of vFFR and drastically decreased inference time from hours to seconds. This study serves as a promising proof-of-concept which we plan on developing further on.

During my master's studies, my main areas of research have been applications of deep learning in cardiovascular imaging and in the analysis of 3D shapes, in particular 3D point clouds. My work "Coronary Ostia Localization Using Residual U-Net with Heatmap Matching and 3D DSNT" [19] dealing with the localization of landmarks of interest in 3D medical images in form of CCTA, was published at Machine Learning in Medical Imaging (MLMI) 2022 Workshop held with MICCAI 2022. This master thesis utilized parts of research previously conducted during my bachelor thesis titled "Artefact removal from 3D coronary artery segmentation from CT via point cloud neural network." which was concluded with a paper "Eigenvector Grouping for Point Cloud Vessel Labeling" [48] at Geometric Deep Learning in Medical Imaging (GeoMedIA) 2022 Workshop endorsed by MICCAI 2022. A part of this work with a follow-up proposing centerline-guided surface embedding [47] was published at MICCAI 2023 conference.



# Bibliography

- [1] S. Achenbach, T. Rudolph, R. Johannes, H. Eggebrecht, G. Richardt, T. Schmitz, N. Werner, F. Boenner, and H. Möllmann. Performing and interpreting fractional flow reserve measurements in clinical practice: An expert consensus document. *Interv Cardiol*, 12(2):97–109, Sept. 2017.
- [2] AdSimuTec GmbH. (in)stationary flow, 2023. [Online; accessed 02-June-2023].
- [3] L. Antiga, M. Piccinelli, L. Botti, B. Ene-Iordache, A. Remuzzi, and D. A. Steinman. An image-based modeling framework for patient-specific computational hemodynamics. *Med Biol Eng Comput*, 46(11):1097–1112, Nov. 2008.
- [4] F. Balsiger, Y. Soom, O. Scheidegger, and M. Reyes. Learning shape representation on sparse point clouds for volumetric image segmentation. *CoRR*, abs/1906.02281, 2019.
- [5] E. J. Bekkers, M. W. Lafarge, M. Veta, K. A. J. Eppenhof, J. P. W. Pluim, and R. Duits. Roto-translation covariant convolutional networks for medical image analysis. *CoRR*, abs/1804.03393, 2018.
- [6] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.
- [7] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021.
- [8] Cadence — System analysis. The importance of meshing in cfd and structural fea, 2023. [Online; accessed 25-April-2023].
- [9] Cardiac Institute of the Palm Beaches, PA. Coronary angioplasty & stent placement, 2023. [Online; accessed 06-April-2023].
- [10] M. Chowdhury and E. A. Osborn. Physiological assessment of coronary lesions in 2020. *Curr Treat Options Cardiovasc Med*, 22(1):2, Jan. 2020.
- [11] Cleveland Clinic. Fractional flow reserve, 2023. [Online; accessed 06-April-2023].
- [12] T. S. Cohen and M. Welling. Group equivariant convolutional networks. *CoRR*, abs/1602.07576, 2016.
- [13] D. Corcoran, B. Hennigan, and C. Berry. Fractional flow reserve: a clinical perspective. *The international journal of cardiovascular imaging*, 33:961–974, 2017.
- [14] F. Diehl. Edge contraction pooling for graph neural networks. *CoRR*, abs/1905.10990, 2019.
- [15] Eiman Jahangir et al. Fractional flow reserve (ffr) measurement periprocedural care, 2023. [Online; accessed 14-April-2023].
- [16] W. Falcon and The PyTorch Lightning team. PyTorch Lightning, Mar. 2019.
- [17] M. Fey and J. E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.



- [18] F. E. Fossan, L. O. Müller, J. Sturdy, A. T. Bråten, A. Jørgensen, R. Wiseth, and L. R. Hellevik. Machine learning augmented reduced-order models for ffr-prediction. *Computer Methods in Applied Mechanics and Engineering*, 384:113892, 2021.
- [19] M. Gajowczyk, P. Rygiel, P. Grodek, A. Korbecki, M. Sobański, P. Podgórski, and T. Konopczyński. *Coronary Ostia Localization Using Residual U-Net with Heatmap Matching and 3D DSNT*, pages 318–327. 12 2022.
- [20] M. Gao, N. Ruan, J. Shi, and W. Zhou. Deep neural network for 3d shape classification based on mesh feature. *Sensors*, 22(18), 2022.
- [21] J. He, C. Pan, C. Yang, M. Zhang, Y. Wang, X. Zhou, and Y. Yu. *Learning Hybrid Representations for Automatic 3D Vessel Centerline Extraction*, pages 24–34. 10 2020.
- [22] Heart and Vascular Centre, Semmelweis University, Faculty of Medicine. Coronarography, 2023. [Online; accessed 06-April-2023].
- [23] D. Hwang, J. M. Lee, and B.-K. Koo. Physiologic assessment of coronary artery disease: Focus on fractional flow reserve. *Korean J Radiol*, 17(3):307–320, Apr. 2016.
- [24] intra: Image Guided Healthcare. Coronary artery stenosis, 2023. [Online; accessed 06-April-2023].
- [25] L. Itu, P. Sharma, A. Kamen, C. Suciu, and D. Comaniciu. Graphics processing unit accelerated one-dimensional blood flow computation in the human arterial tree. *International journal for numerical methods in biomedical engineering*, 29(12):1428–1455, 2013.
- [26] L. M. Itu, S. Rapaka, T. Passerini, B. Georgescu, C. Schwemmer, M. Schoebinger, T. Flohr, P. Sharma, and D. Comaniciu. A machine learning approach for computation of fractional flow reserve from coronary computed tomography. *Journal of applied physiology (Bethesda, Md. : 1985)*, 121:jap.00752.2015, 04 2016.
- [27] Johns Hopkins Medicine, Heart. Angioplasty and stent placement for the heart, 2023. [Online; accessed 06-April-2023].
- [28] M. Kamali-Sadeghian, P. Bot, R. Tukkie, H. Wellens, and D. Doorn. A peek behind the curtain. *Netherlands Heart Journal*, 26, 08 2018.
- [29] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP ’06, page 61–70, Goslar, DEU, 2006. Eurographics Association.
- [30] H. J. Kim, I. Vignon-Clementel, J. Coogan, C. Figueroa, K. Jansen, and C. Taylor. Patient-specific modeling of blood flow and pressure in human coronary arteries. *Annals of biomedical engineering*, 38:3195–3209, 2010.
- [31] A. Kosior, K. Mirota, and W. Tarnawski. Patient-specific modeling of hemodynamic parameters in coronary arteries, June 20 2019. US Patent App. 16/217,328.
- [32] G. Li, H. Wang, M. Zhang, S. Tupin, A. Qiao, Y. Liu, M. Ohta, and H. Anzai. Prediction of 3d cardiovascular hemodynamics before and after coronary artery bypass surgery via deep learning. *Communications Biology*, 4, 01 2021.
- [33] Z. Małota, W. Sadowski, K. Pieszko, R. Zimoląg, F. Czeała, R. Malinowska, and J. Hiczkiewicz. The comparative method based on coronary computed tomography angiography for assessing the hemodynamic significance of coronary artery stenosis. *Cardiovascular Engineering and Technology*, pages 1–16, 2023.
- [34] P. Medrano-Gracia, J. Ormiston, M. Webster, S. Beier, C. Ellis, C. Wang, O. Smedby, A. Young, and B. Cowan. A study of coronary bifurcation shape in a normal population. *Journal of Cardiovascular Translational Research*, 10:1–9, 02 2017.

- [35] B. N. Modi, O. M. Demir, H. Rahman, M. Ryan, S. A. Sherif, H. Ellis, A. Colombo, and D. Perera. Clinical utility of novel fractional flow reserve pullback for individual lesion contribution in serial disease. *Journal of Invasive Cardiology*, 2021.
- [36] P. D. Morris, D. Ryan, A. C. Morton, R. Lycett, P. V. Lawford, D. R. Hose, and J. P. Gunn. Virtual fractional flow reserve from coronary angiography: modeling the significance of coronary lesions: results from the virtu-1 (virtual fractional flow reserve from coronary angiography) study. *JACC: Cardiovascular Interventions*, 6(2):149–157, 2013.
- [37] D. Munhoz, J. Sonck, and C. Collet. Pullback pressure gradient: Discriminating focal and diffuse coronary artery disease using coronary physiology. *Cardiac Interventions Today*, 2021.
- [38] W. Y. Nikolaev A, Feger J. Coronary artery disease - reporting and data system. *Reference article, Radiopaedia.org (Accessed on 14 Feb 2023)*.
- [39] 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, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [40] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016.
- [41] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.
- [42] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.
- [43] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566*, 2017.
- [44] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- [45] P. Rajiah, K. W. Cummings, E. Williamson, and P. M. Young. CT fractional flow reserve: A practical guide to application, interpretation, and problem solving. *Radiographics*, 42(2):340–358, Feb. 2022.
- [46] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.
- [47] P. Rygiel, P. Płuszka, M. Zieba, and T. Konopczyński. Centerlinepointnet++: A new point cloud based architecture for coronary artery pressure drop and vffr estimation. In H. Greenspan, A. Madabhushi, P. Mousavi, S. Salcudean, J. Duncan, T. Syeda-Mahmood, and R. Taylor, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2023*, pages 781–790, Cham, 2023. Springer Nature Switzerland.
- [48] P. Rygiel, M. Zieba, and T. Konopczynski. Eigenvector grouping for point cloud vessel labeling. In E. Bekkers, J. M. Wolterink, and A. Aviles-Rivero, editors, *Proceedings of the First International Workshop on Geometric Deep Learning in Medical Image Analysis*, volume 194 of *Proceedings of Machine Learning Research*, pages 72–84. PMLR, 18 Nov 2022.
- [49] V. Sklet. Exploring the capabilities of machine learning (ml) for 1d blood flow: Application to coronary flow. *Master's Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2018.*, 2018.
- [50] Stanford Geometry. Remeshing i, 2023. [Online; accessed 24-April-2023].

- [51] J. Suk, C. Brune, and J. M. Wolterink. Se(3) symmetry lets graph neural networks learn arterial velocity estimation from small datasets, 2023.
- [52] J. Suk, P. Haan, P. Lippe, C. Brune, and J. Wolterink. Mesh convolutional neural networks for wall shear stress estimation in 3d artery models, 09 2021.
- [53] C. A. Taylor, T. A. Fonte, and J. K. Min. Computational fluid dynamics applied to cardiac computed tomography for noninvasive quantification of fractional flow reserve: scientific basis. *Journal of the American College of Cardiology*, 61(22):2233–2241, 2013.
- [54] C. Tesche, C. N. De Cecco, M. Albrecht, T. Duguay, R. Bayer, S. Litwin, D. Steinberg, and U. Schoepf. Coronary ct angiography-derived fractional flow reserve. *Radiology*, 285:17–33, 10 2017.
- [55] The Vascular Modelling Toolkit (VMTK) documentation. Computing centerlines, 2023. [Online; accessed 02-June-2023].
- [56] I. E. Vignon-Clementel, C. Figueroa, K. Jansen, and C. Taylor. Outflow boundary conditions for 3d simulations of non-periodic blood flow and pressure fields in deformable arteries. *Computer methods in biomechanics and biomedical engineering*, 13(5):625–640, 2010.
- [57] Z. Wang, Y. jie Zhou, Y. xin Zhao, D. mei Shi, Y. yang Liu, W. Liu, X. Liu, and Y. ping Li. Diagnostic accuracy of a deep learning approach to calculate ffr from coronary ct angiography. *Journal of Geriatric Cardiology : JGC*, 16:42 – 48, 2019.
- [58] Wikipedia contributors. Euclidean group, 2023. [Online; accessed 22-May-2023].
- [59] Wikipedia contributors. Fractional flow reserve, 2023. [Online; accessed 06-April-2023].
- [60] Wikipedia contributors. Permutation group, 2023. [Online; accessed 23-May-2023].
- [61] Wikipedia contributors. Symmetric function, 2023. [Online; accessed 24-May-2023].
- [62] L. Zhong, J.-M. Zhang, B. Su, R. S. Tan, J. Allen, and G. Kassab. Application of patient-specific computational fluid dynamics in coronary and intra-cardiac flow simulations: Challenges and opportunities. *Frontiers in Physiology*, 9, 06 2018.

# Additional qualitative results

In this supplementary, we include additional qualitative results of vFFR estimation.

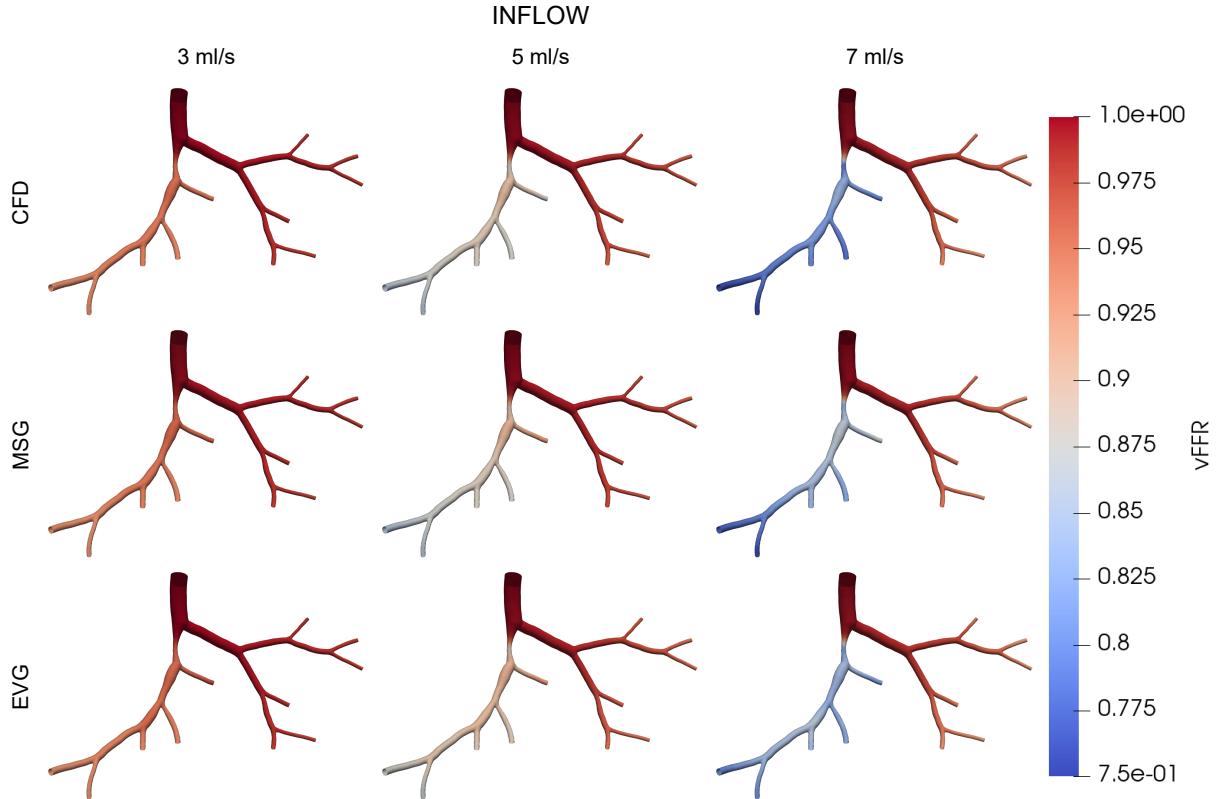


Figure A.1: Qualitative results of vFFR estimation for various inflow values. We can see that the proposed approach can be easily adapted for various sets of boundary conditions e.g. inflow, of underlying CFD simulation. The absolute errors do become larger, when inflow gets larger, however, the percentage errors stay the same - see Table. 5.2.

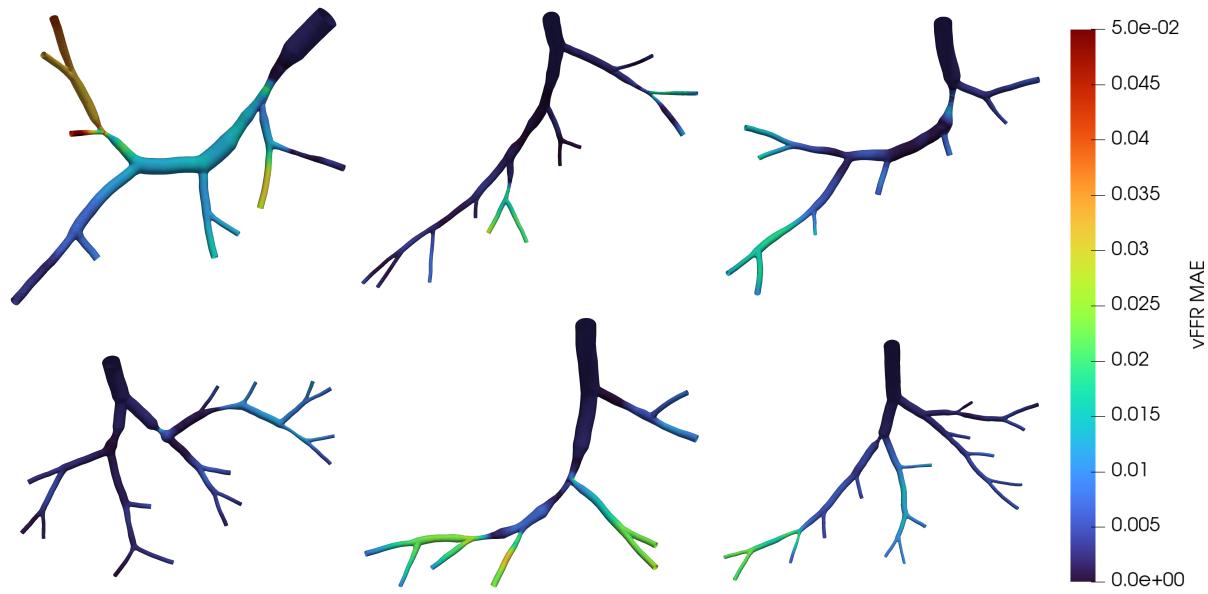


Figure A.2: Additional MAE results for MSG model. We see that the highest errors appear in the stenotic areas or after them. The errors tend to be propagated after under or over-estimated narrowing impact.

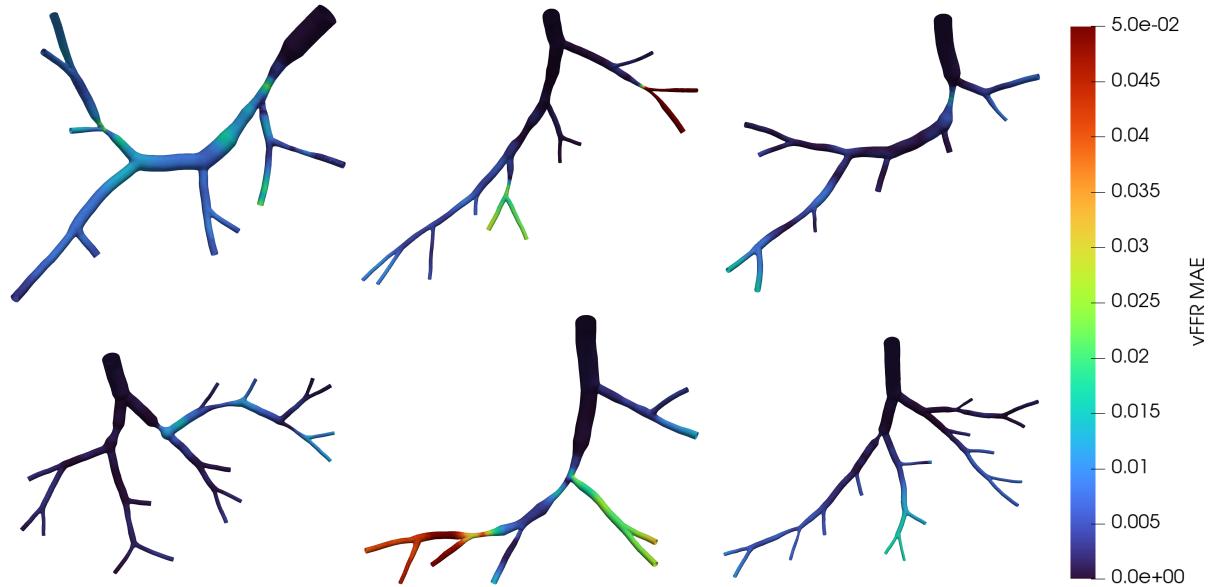


Figure A.3: Additional MAE results for EVG model. We report the same issues for EVG as well - there are no significant differences between both methods.