Otto-von-Guericke University Magdeburg

Department of Computer Science
Chair for Visualization

Master Thesis

Deep Learning Based Co-Registration of Point Clouds

Author:
Vatsal Pandey
Matriculation Number: 231615

August 2, 2023

Advisers:

Supervisor
Dr.-Ing. Sylvia Saalfeld
Department of Computer Science
Otto-von-Guericke University
Universitätsplatz 2
39106 Magdeburg, Germany

Supervisor
Dr.-Ing. Marko Rak
Department of Computer Science
Otto-von-Guericke University
Universitätsplatz 2
39106 Magdeburg, Germany

Abstract

Medical image registration is a process that involves aligning and matching two or more medical images of the same patient or anatomical region. It aims to establish a spatial correspondence between images acquired from different imaging modalities or at different time points. These images can be represented as 2D or 3D images, CT and MRI scans, and point clouds or meshes. This thesis uses deep learning-based registration methods to focus on the point cloud registration of Time-of-Flight (TOF) and Phase-Contrast Magnetic Resonance Imaging (PC MRI) data. Registering these two imaging modalities is essential for aligning and combining their respective point cloud data, enabling a comprehensive analysis of vascular anomalies and blood flow dynamics.

The thesis demonstrates the effectiveness and suitability of the selected methods for TOF and PC MRI point cloud registration. The registration procedure could make spotting vascular disorders such as malformations, aneurysms, and stenoses easier. Traditional methods of point cloud registration face challenges in handling complex and high-dimensional data, which is the case in TOF and PC MRI data. To address this, the thesis explores and evaluates three methods that leverage deep learning techniques: ROPNet, DIP, and C2PNet. These methods utilize convolutional neural networks (CNNs) to extract features from the point cloud data and optimize the registration process.

Contents

1 Introduction

1.1 Motivation

This thesis focuses on registering point clouds obtained from TOF and PC MRI modalities. Combining these two modalities can enhance our understanding of anatomical structures and vascular anomalies in the brain. For example, according to [12], TOF MRI can show the shape and structure of blood vessels in the brain and help detect vessel narrowings. However, it cannot directly tell us if there is a lack of blood supply to a particular area (ischemia). The blood flow in the brain is an essential factor in detecting ischemic stroke [12]. In an ischemic stroke, insufficient blood flow to the brain can lead to cerebral infarction or symptoms of a lack of blood flow. Therefore, flow imaging techniques like PC MRI can help diagnose the risk of an upcoming stroke [12]. In such situations, TOF MRI can initially evaluate narrowings in the cerebral arteries. The second technique involves flow imaging using PC-MRI, which can measure blood flow in the cerebral arteries. Slow blood flow in these areas can be a significant risk factor for a stroke [12]. Therefore, combining the two medical modalities can achieve an imaging solution for stroke assessment. Therefore, combining information about the structure of blood vessels from TOF MRI and blood flows information from PC MRI could help us more accurately identify, locate, and describe vascular issues like abnormal blood vessel formations, aneurysms, and narrowed areas.

The motivation for using the deep learning-based method comes from its advantages compared to traditional methods. Firstly, deep learning-based approaches allow for end-to-end learning, meaning the entire registration process can be optimized through a single neural network. Traditional methods often involve multiple stages and manual parameter tun-

ing, which can be time-consuming [75] and less flexible. Deep learning models can automatically learn meaningful features from raw point cloud data without explicit feature engineering. This enables the extraction of complex and abstract representations that capture relevant information for registration, potentially leading to improved accuracy. Deep learning methods can learn to identify features invariant to noise and occlusions, making them more robust to these challenges than traditional methods [102]. Also, traditional methods like ICP [71] are prone to get stuck at local minima [87, 75].

1.2 Medical Image Registration

Medical image registration is a critical aspect of medical imaging that involves aligning and matching different medical images. These images can be represented as 3D point clouds, 3D meshes, computed tomography(CT) scans, MRI scans, and 2D data like X-rays and ultrasounds. It is vital in applications like image-guided surgery, radiation therapy, and disease progression monitoring [75]. Doctors can compare images by overlaying them that belong to the same patient taken at different times or by different methods. It helps them find any changes, gather essential details, and make sure they diagnose and plan treatment correctly [75].

If the registration involves aligning images taken from the same sensor at different points in time, it is referred to as unimodal registration. Multimodal registration refers to the images taken from different sensors and vary in shape and structure. Additional challenges arise regarding multimodal registration. Multimodal data can demonstrate diverse characteristics, such as different data structures, dimensions, densities, levels of noise, and types of geometric inaccuracies [75]. A registration framework to align medical images from different modalities is more challenging than registering images of the same modality. Figure 1.1 shows an example of multimodal registration using deep learning.

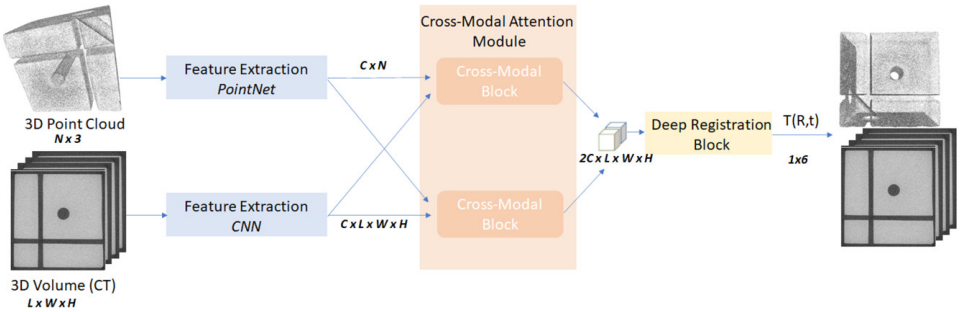


Figure 1.1: Multimodal Image Registration [76]

1.3 Medical Background

One of the critical applications of multimodal registration is the diagnosis and treatment of cerebral small vessel disease (CSVD). CSVD is a prevalent and significant neurological condition affecting the brain's tiny blood vessels. It is a collective term encompassing various pathological changes in these vessels, leading to impaired blood flow, ischemia, and damage to the brain tissue. CSVD is a common cause of stroke, cognitive impairment, and other neurological symptoms [14]. CSVD is believed to result from genetic, vascular risk, and lifestyle factors. Hypertension, diabetes, smoking, hyperlipidemia, and aging are common risk factors associated with the development and progression of CSVD [90]. These risk factors contribute to the structural and functional changes in the small blood vessels, including arteriosclerosis, lipo hyalinosis, endothelial dysfunction, and blood-brain barrier dysfunction [27, 28, 33]. According to epidemiological studies as given in [69], at least 700 million people worldwide suffer from various forms of CSVD, which are responsible for 25% of stroke cases, 45% of dementia cases, and 70% of cases of vascular dementia (VD) [65]. The primary factor behind aged people's neurological degeneration and cognitive decline is CSVD. CSVD, which accounts for 80% of the overall incidence, increases incidence as an age-related risk factor. CSVD is more prevalent in people over 60, and its prevalence is over 100% in people over 90 years of age [10]. MRI is the most effective imaging diagnostic technique for CSVD, and it plays a guiding role in this condition's early detection, diagnosis, and treatment [69]. In imaging the CSVD brain microstructure and micro vascularization, MRI has produced impressive results [64].

Anatomical three-dimensional brain images are created using MRI, a non-invasive imaging method. It is frequently used for illness detection, treatment monitoring, and diagnosis. It is based on a technique that induces and tracks alterations in the protons' rotational axes in the water that makes up biological tissues [60]. During an MRI scan, strong magnets in the machine create a magnetic field, and protons try to align with this magnetic field. A radiofrequency pulse is applied to the protons, causing them to spin out of alignment with the magnetic field. When the pulse is turned off, the protons align once again by returning to their original position and generating a signal that may be utilized to produce a picture. A patient is positioned inside a massive magnet for an MRI scan and is expected to maintain total stillness to guarantee quality images without blurring. In some circumstances, the patient could have intravenous injections of contrast agents, which often contain gadolinium [60]. A brighter picture is produced due to these agents' assistance in accelerating the realignment of protons with the magnetic field. The distinction between white and grey matter in the brain may be made with MRI, which can also be used to find tumors and aneurysms [60]. It is especially suitable for regular imaging requirements in the brain because it does not involve using X-rays or other types of radiation [60]. Due to this, MRI is the method of choice for treating and diagnosing brain-related diseases.

1.3.1 Time of Flight MRI

Time-of-flight magnetic resonance imaging (TOF MRI) is a widely used and reliable technique for imaging blood vessels. It takes advantage of the flow-related enhancement observed in incoming blood to generate highly contrasted images of the vascular system, all without the need for gadolinium-based contrast agents [48]. Figure 1.2 shows the circle of willis segmented from a TOF MRI scan.

Figure 1.2: Circle of Willis segmented from TOF MRI scan [81]

TOF MRI can achieve high spatial resolution, identifying and characterizing small vascular structures as seen in figure 1.2 and abnormalities. It is essential for detecting conditions such as aneurysms, arteriovenous malformations (AVMs) [26], and stenoses, which require detailed visualization for accurate diagnosis and treatment planning. TOF MRI primarily provides information about the presence and location of blood flow within vessels. However, it does not provide quantitative information about blood flow velocities. Additional techniques like phase-contrast MRI may be necessary to assess vascular function and flow dynamics comprehensively. Also, it is susceptible to flow-related artifacts, which can affect the quality and accuracy of the images. These artifacts can arise from turbulent or slow-flowing blood, patient motion, or technical factors [43, 108]. 3D TOF MRI primarily focuses on imaging arterial blood vessels and may have limitations in visualizing venous structures [108].

1.3.2 Phase Contrast MRI

Phase-contrast magnetic resonance imaging (PC MRI) utilizes the phase shift in moving blood after applying bipolar gradients. This technique produces angiographic images of blood vessels and enables the quantification of velocity and pressure gradients in stenotic lesions [48]. Figure 1.3 shows the circle of Willis segmented from a PC MRI scan. When compared

to the TOF MRI image, it can be seen that PC MRI has less resolution and only shows large blood vessels.

Figure 1.3: Circle of Willis segmented from PC-MRI [81]

Phase Contrast MRI allows for directly measuring and quantifying blood flow velocities and volumes in cerebral arteries [43]. It provides valuable information about the direction, speed, and blood flow patterns, which are crucial for assessing vascular function and identifying abnormalities such as stenoses [43]. PC MRI of the brain for diagnosing aneurysms in patients with bleeding and assessing arteriovenous malformations is better than TOF MRI [94].

1.4 Aim of this thesis

The primary objective of this thesis is to research and implement robust and efficient deep learning-based methods for the registration of point clouds acquired from TOF and PC MRI imaging techniques. The thesis aims to study the critical challenges in registering point clouds obtained from TOF and PC MRI modalities, the existing methods for point cloud registration, and how they perform in the context of TOF and PC MRI data.

1.5 Structure of this thesis

For ease of understanding, an overview of the thesis and contents of each section is described below.

- Chapter 2 provides a detailed explanation of point cloud registration and its different techniques. It also explains the basic terminology associated with it.
- Chapter 3 deals with related work for deep learning-based point cloud registration.
- Chapter 4 explains the different methods implemented and the related algorithms.
- Chapter 5 provides the results and comparison of different implemented methods.
- Chapter 6 summarizes the thesis and discusses future work possibilities.

2

Theoretical Framework

2.1 Mesh and Point Cloud

Meshes have various applications in various fields, including computer graphics, computational geometry, and medical imaging. In medical imaging, meshes serve as a valuable representation of anatomical structures, allowing for the visualization, analysis, and manipulation of complex geometric information [49]. A mesh is a collection of interconnected points, lines, and surfaces. The points, called vertices or nodes, represent specific positions in three-dimensional space. The lines, or edges, connect pairs of vertices and define the structural lines or curves of the mesh. The surfaces, or faces, are formed by connecting three or more vertices, shaping the outer surface of the mesh. Different meshes exist, categorized based on their connectivity and geometry, for example, triangular meshes, quadrilateral meshes, and tetrahedral meshes. A point cloud describes an object's shape using data points scattered in a two or three-dimensional space to create a dense representation. Each data point in the 3D cloud has XYZ coordinates, indicating its specific position along the axes. A point cloud can be generated from a mesh by only considering the vertices and removing the faces. In addition to spatial information, the points in the cloud and vertices of a mesh can also include additional data such as normals or colors. Normals indicate the orientation of each point's surface and are vital for shading, lighting, and surface analysis. Color information enhances the visual appearance and can be captured using RGB values or other color spaces. Some additional properties associated with point clouds and meshes are (i) **Density**: The number of points per unit area or volume. A higher density of points may result in a more detailed representation of the object. (ii) **Noise**: Noise refers to unwanted

variations, regions, or errors in the point cloud data. (iii) **Size**: It refers to the total number of points in the point cloud. Larger point clouds may offer more detailed representations but require additional computational resources for classification, segmentation, or registration tasks. (iv) **Data format**: Point clouds can be stored in various formats, such as LIDAR Data Exchange Format, PLY (Polygon File Format), or XYZ.

A point cloud can be denoted as $P = \{p_1, p_2, ..., p_n\}$ where $p \in \mathbb{R}^3$ if the point are in a 3D space. However, if additional properties like curvature, normals, and color information (RGB), the points can be represented by a more extensive vector [51]. For example, if normals of the points are available, each point will be represented by a 6D vector, i.e., $p \in \mathbb{R}^6$. The mesh representation is shown in figure 2.1 while the point cloud representation is shown in figure 2.2. There are noticeable variations between the two images when compared. Due to the faces in the mesh, the vessels are smooth and well-defined, resulting in an aesthetically appealing form. On the other hand, the point cloud lacks the mesh's smooth surfaces, and the vessels are shown in a more dispersed and disjointed manner.

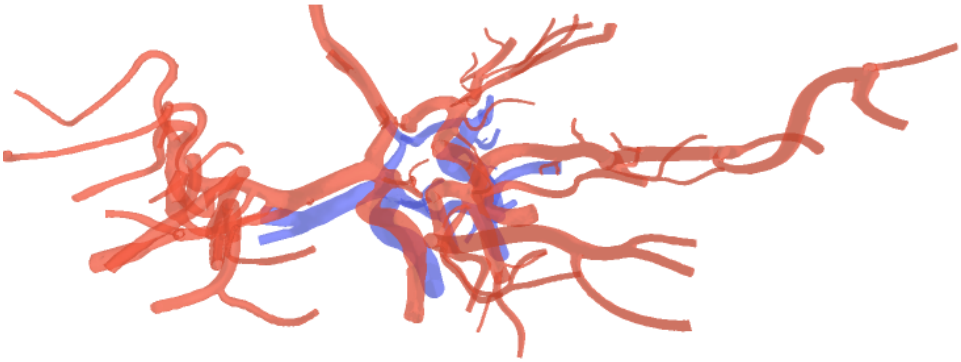


Figure 2.1: TOF(red) and PC(blue) mesh.

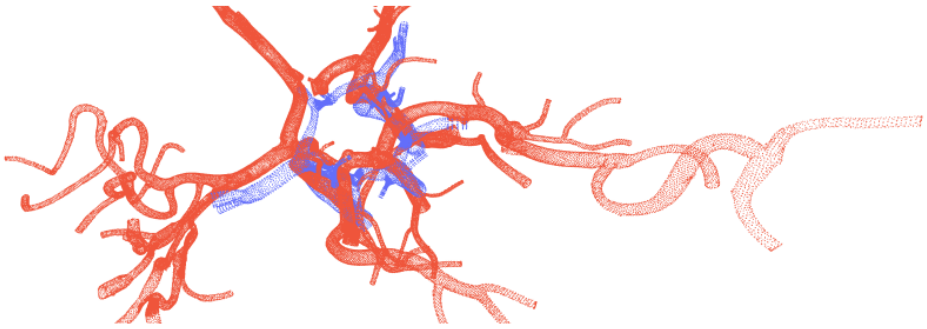


Figure 2.2: TOF(red) and PC(blue) point cloud.

2.2 Point Cloud Registration

Registration is the process of aligning point clouds of similar objects or multiple samples of the same object captured at different points in time, from different perspectives, or using different sensors in the same reference system. A registration technique involves three main components: the transformation that establishes the relationship between the datasets, the similarity metric that measures how similar the datasets are, and an optimization method that calculates the best transformation parameters based on the similarity metric [75]. A registration method aligns two datasets by finding the best transformation that minimizes the discrepancy between them, as determined by the similarity metric. During registration, two point clouds are utilized: the target point cloud and the source point cloud. The target point cloud is the reference point cloud that remains unchanged throughout the registration process. It is the fixed or desired position to which the source point cloud is aligned. On the other hand, the source point cloud transforms. Let us consider two point clouds, S (source) and T (target), with N_S and N_T points, respectively. The registration goal is finding a motion or pose that transforms S to align with T.

2.2.1 Rigid and Non-Rigid Registration

Based on the type of transformation, a point cloud registration algorithm can be classified as rigid or non-rigid. Rigid registration refers to align-

ing two or more point clouds by applying a rigid transformation that combines rotation and translation, and it preserves the distances, angles, and shapes between points [51]. Mathematically, the rigid transformation can be represented as $p' = R * p + T$ where:

- p is a point in the source point cloud,
- p' is the same point after alignment and the corresponding point in the target point,
- R is the matrix that describes the rotation,
- T is the translation vector that represents the displacement.

The goal is to find the optimal values for R and T that minimize the difference between the source and target point clouds by finding the transformations that minimize a specific objective function, for example, the sum of squared distances between the corresponding points. Given source and target point clouds, $S = \{s_1, s_2, ..., s_n\}$ and $T = \{t_1, t_2, ..., t_n\}$, the rigid registration problem can be formulated as an optimization problem [51]: (1) minimize $\sum \|(R * s_i + T) - t_i\|^2$, where s_i and t_i are the points in source and target cloud respectively; (2) subject to $R^{\mathcal{T}} * R = I$, where I is the identity matrix and $\mathcal{T}$ represents the transpose. The objective function $\sum \|(R * s_i + T) - t_i\|^2$ represents the sum of squared distances between the transformed source points and the corresponding target points. The condition $R^{\mathcal{T}} * R = I$ ensures that the transformation matrix is orthogonal and preserves the source point cloud's shape, resulting in a rigid registration.

Non-rigid or deformable registration refers to aligning two or more point clouds or images that can undergo local deformations or shape changes. While rigid Non-rigid or deformable registration refers to aligning two or more point clouds that can undergo local deformations or shape changes. While rigid registration only allows for transformations that preserve the overall shape of the source surface, non-rigid registration allows for more flexible transformations that can capture local deformations, such as reflections, rotations, scaling, and translations. However, these transformations are applied locally and not on the whole point cloud differentiating non-rigid with affine transformation. Non-rigid registration is utilized primarily for two reasons [51]: (1) to account for non-linearities and errors in

data that result in warping of rigid objects [9, 50]. These warps occur during the scanning phase when due to measurement inaccuracies, the point cloud undergoes slight misalignment and has irregular deformations and (2) to align and register deformable or moving scenes or objects that undergo shape changes over time. For example, as shown in figure 2.3, there are multiple scans of a dinosaur point cloud. Each scan differs in position and shape of the head and tail. These clouds cannot be directly registered using rigid transformation, and warping is required.

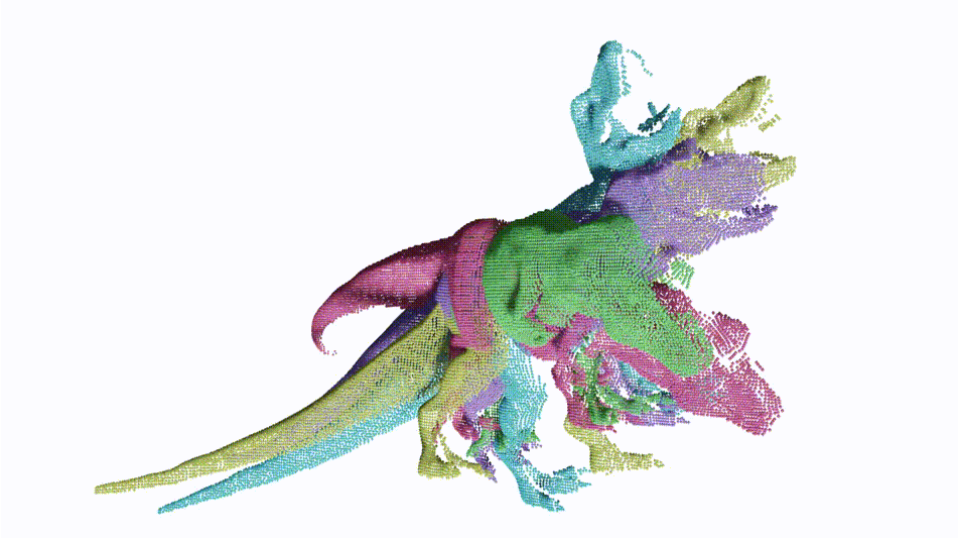


Figure 2.3: Example of point clouds that changes over time [53].

Non-rigid registration is a complex task with several challenges. Choosing a suitable representation for the deformation field that balances accuracy and computational efficiency is crucial [51]. Moreover, point clouds may contain noise and outliers and have limited overlap where only certain point cloud regions can be aligned. These factors can hinder the establishment of reliable correspondences between the point clouds for accurate alignment [51]. After the correspondences are estimated, the non-rigid registration process is simpler to perform [17, 11]. Additionally, in many practical applications, performing registration efficiently, preferably in real-time, is desirable [18].

2.2.2 Local and Global Registration

Point cloud registration algorithms can also be classified as global and local registration based on the transformation domain or region of interest. While both methods aim to achieve alignment, they differ in scope and the information they utilize.

Global registration [72, 15, 87, 1, 88, 98] aims to align the entire point clouds by considering their overall structure and shape and seeks to find a transformation that aligns them as a whole making these methods computationally expensive [51]. Global methods do not lead to accurate results [8]. If there are noise and outliers in the point clouds, the algorithm will also use these points for transformation estimation leading to misalignment. RANSAC (Random Sample Consensus) [24] is a widely used global registration method.

Local registration focuses on aligning a limited region or a subset of points within the point clouds and typically operate on smaller patches or neighborhoods. These methods can be stuck in local minima [8], meaning they may find a good alignment for the point clouds, but there may be better possible ones. A registration pipeline can use a global registration to provide an initial alignment estimate followed by a local registration for final transformation [8]. This initial estimate is then used as a starting point for a local optimization algorithm, which can then find a better alignment. Examples of local registration techniques include Iterative Closest Point (ICP) [5] algorithm, which iteratively estimates the transformation based on closest point correspondences between local patches, and variants such as Generalized-ICP (G-ICP) [78] that incorporate additional information such as surface normals or color.

2.3 Traditional Registration Methods

Traditional point cloud registration refers to the methods and techniques that do not use neural networks and typically rely on geometric or statistical algorithms to transform the point clouds.

2.3.1 Iterative Closest Point

The ICP algorithm [5] is an iterative method designed to achieve accurate, fast, and stable registration under optimal conditions. It can be seen as an expectation-maximization (EM) problem [51], where transformations are estimated using the correspondences between the point clouds and iteratively refines the registration until an error metric converges. However, it should be noted that ICP does not guarantee to reach a globally optimum solution [8]. ICP-based registration methods consist of two primary stages: estimating correspondences and transforming. Correspondences refer to pairs of points occupying the same position in an object or scene, each originating from a different point cloud.

Correspondence step: For a given iteration number or the current iteration given as k, this step aims to find the closest point $\hat{q}_i^{(k)}$ in the target point cloud (Q) for each point (p_i) in the source point cloud (P) based on the transformation defined by $R^{(k)}$ and $T^{(k)}$. It is done by calculating the squared distance between the transformed source point and each point in the target point cloud and selecting the point with minimum distance given by equation 2.1 [101].

$$\hat{q}_i^{(k)} = \underset{q \in Q}{\arg\min} \| R^{(k)} p_i + T^{(k)} - q \|^2 \tag{2.1}$$

Estimation step: This step minimizes the squared distance between the corresponding points. The updated transformation $R^{(k+1)}$ and $T^{(k+1)}$ is obtained by minimizing the objective function given in equation 2.2 [101].

$$(R^{(k+1)}, T^{(k+1)}) = \underset{R,T}{\arg\min} \sum_{i=1}^{M} \| R p_i + T - \hat{q}_i^{(k)} \|^2 + I_{SO(d)(R)} \tag{2.2}$$

In this equation, $Rp_i + T$ represents the transformed source point and $I_{SO(d)(R)}$ is a regularization term to enforce the constraint that the rotation matrix R must belong to the group of all rotation matrices in dimension d, known as the special orthogonal group SO(d), and M represents the total number of correspondences between the source and target points. The optimization is done through the singular value decomposition (SVD) method [2].

Besl et al. [5], and Arun et al. [3] introduced a point-to-point approach for point cloud registration. This strategy involves pairing the closest points

in both the point clouds with each other based on Euclidean distances. These pairs form the correspondences, as discussed in the correspondence step. The sum of Euclidean distances between the matched pairs is given in equation (2.3) [51]. The objective is to find the optimal values of the transformation parameters R and T that minimize equation (2.4) [51].

$$L(P,Q) = \sum_{i=1}^{n} \|p_i - q_i\|^2, \tag{2.3}$$

$$\hat{L}(R,T) = \underset{R,T}{\arg\min} \sum_{i=1}^{n} \|p_i - (Rq_i + T)\|^2 \tag{2.4}$$

The equations, $\|\cdot\|^2$ represent the Euclidean distance. The point pairs p_i and q_i represent the i-th correspondence between source and target point clouds P and Q, with N total pairs. The rotation matrix R belongs to the orthogonal group $SO(3)$, and the translation vector T belongs to $\mathbb{R}^3$. The source point cloud is updated using the estimated transformation, and the process proceeds to the subsequent computation. The ICP algorithm continues until the difference between the current error metric $\hat{L}$ and the following error metric $\hat{L}_{next}$ is smaller than a given threshold δ, expressed as $|\hat{L} - \hat{L}_{next}| < \delta$ [51]. The point-to-point metric in point cloud registration involves finding the closest point pair as a correspondence based on either the coordinate or feature distance. Various approaches have been proposed to enhance this concept and obtain improved correspondence. For instance, the ICP [5] algorithm utilizes the original point-to-point distance metric. EfficientVarICP [71] presents optimizations to speed up the ICP process. Another method called IMLP [6] incorporates the measurement noise into the transformation estimation to enhance the accuracy of ICP.

Chen et al. [13] and Bergevin et al. [4] presented a point-to-plane approach involving the nearest point search between the point and normal of the target and source point cloud. The goal is to minimize the sum of squared distances between each correspondence, considering the distance from a point to a plane, given in equation (2.5) [51]. As a result, it is less affected by outliers and noise and offers improved accuracy and faster convergence compared to the point-to-point ICP method [101].

$$\hat{L}(R,T) = \underset{R,T}{\arg\min} \sum_{i=1}^{n} \Big(\big(Rq_i + T - p_i\big) \cdot n_{p_i} \Big)^2 \tag{2.5}$$

To find an optimal transformation represented by the point-to-plane ICP algorithm involves transforming the point q_i using R and T and considering the normal vector n_{p_i} at p_i. The remaining variables remain the same as in equation (2.4). To solve equation 3, various linear optimization techniques can be applied, such as the SVD method [2], computation of orthonormal matrices [37], or unit quaternions [38]. Nonlinear solvers like the Levenberg-Marquardt method [25] can also be utilized. These methods demonstrate similar accuracy and stability when minimizing the sum of squared error metric [21].

2.3.2 RANSAC

RANSAC (Random Sample Consensus) [24] is a robust method to find correspondences between point clouds that are noisy or with outliers. It achieves this by randomly sampling subsets of points from each point cloud and fitting models to these subsets. The algorithm then selects the model with the most inliers (points that fit the model well) as the final model. The optimal number of iterations is crucial, as it directly impacts the likelihood of obtaining a good model. Too few iterations might not allow sufficient time to find a suitable model, while excessive iterations could lead to a wasteful search for inadequate models.

The complete algorithm is given in 1. In each algorithm iteration, a random selection of three points is made from the source cloud. The next step involves finding correspondences between source and target point clouds by calculating the features for each point and selecting the most similar point pairs using the estimated features. The FPFH (Fast Point Feature Histograms) [72] method is commonly used for this purpose due to its balance between descriptor uniqueness and computation time [35]. The third step is to estimate the transformation matrix using the three-point pairs. It is a well-known problem in linear algebra [36]. The source cloud is then transformed using the estimated matrix, and the algorithm checks how many points from the source cloud lie within a particular region around each target point [47]. The algorithm continues iterating until the ratio of inliers exceeds a specified threshold. RANSAC can provide an acceptable registration result in the presence of noise and outliers, which can be seen in the experiment section in this thesis. However, an opti-

mal solution is not guaranteed, and the algorithm can have high computational time.

Algorithm 1 RANSAC [47]

START:
  Set the maximum number of iterations.
  Set the threshold for the inliers ratio.
  Initialize the best transformation matrix; best inliers count to 0.
LOOP:
  Randomly select three points from the source point cloud.
  Find each selected point's nearest neighbor in the feature space.
  Estimate the transformation matrix based on the three pairs of points.
  Transform the source cloud using the estimated matrix.
  Count the number of inliers by comparing the transformed points to the corresponding points in the target cloud.
  Calculate the inliers ratio as the number of correspondences divided by the total number of points.
  if inliers ratio > threshold **then**
    Update the best transformation matrix and best inliers count.
  end if
  if the number of iterations reaches the maximum number **then**
    go to **STOP**.
  end if
STOP:
  Output the best transformation matrix and the corresponding inliers count.

2.3.3 Coherent Point Drift

Myronenko et al. developed a probabilistic method called Coherent Point Drift (CPD) [63] for registering point clouds. CPD can be used to register both rigid and non-rigid point clouds. CPD aims to find the transformation that best aligns two point sets by fitting a Gaussian Mixture Model (GMM) to the data from each point cloud. A GMM is a statistical model that can represent a probability distribution. The GMM for the first point cloud is fitted to the data from the second point cloud. The movement of the GMM centroids is then enforced as a coherent group. This means that the GMM centroids are not allowed to move independently of each other.

In rigid registration, the coherence constraint is a technique used to enforce consistency in the alignment of data points by reparameterizing the

locations of the GMM centroids using rigid parameters [63]. It ensures that the centroids move consistently with the rigid transformation applied to the point clouds. This leads to a closed-form solution for the maximization step of the expectation-maximization (EM) algorithm [63]. The EM algorithm is used to optimize the GMM parameters iteratively. The closed-form solution simplifies the maximization step, making it computationally efficient and applicable in any dimension [63]. In non-rigid registration, the coherence constraint is imposed by a constraint that encourages smoothness and coherence in the deformations. It is done by using variational calculus [63].

CPD assumes that the PDF can be modeled using a GMM. A GMM is a weighted sum of Gaussian distributions, representing a potential correspondence between points in X and Y. The GMM is parameterized by the mean and covariance of each Gaussian component and the mixing coefficients. The GMM probability function is given in equation 2.3.3 [63].

$$p(x) = \sum_{m=1}^{M+1} P(m) \frac{1}{(2\pi\sigma^2)^{D/2}} exp^{-\frac{\|x-y_m\|^2}{2\sigma^2}}$$

Here D is the dimension of point cloud, N, M is the number of points, σ^2 is isotropic covariance and $P(m) = \frac{1}{M}$ for all GMM components. The points in Y are considered GMM centroids, and X are the data points generated by the GMM [33]. Adding weights following a uniform distribution i.e. $w, 0 \leq w \leq 1$, equation 2.3.3 can be modified as [63]:

$$p(x) = w\frac{1}{N} + (1-w) \sum_{m=1}^{M} \frac{1}{M} \frac{1}{(2\pi\sigma^2)^{D/2}} exp^{-\frac{\|x-y_m\|^2}{2\sigma^2}}$$

The complete algorithms from [63] are given in algorithm 2 for rigid CPD, algorithm 3 for affine CPD, and algorithm 4 for non-rigid CPD.

Algorithm 2 Rigid CPD [63]

Initialization:

Set the initial values for the rotation matrix R=I, translation vector t=0, scale factor s=1, and weight parameter 0 ≤w≤ 1.

Compute the initial value for the noise variance σ^2 as:

$$\sigma^2 = \frac{1}{DNM} \sum_{n=1}^{N} \sum_{m=1}^{M} \|x_n - y_m\|^2 \tag{2.6}$$

EM Optimization: Repeat the following steps until convergence.

E-step:

Compute the posterior probability matrix P, which represents the probability of correspondence between points in the source and target point sets as:

$$p_{mn} = \frac{exp^{-\frac{1}{2\sigma^2}\left\|x_n-(sR_{y_m}+t)\right\|^2}}{\sum_{k=1}^{M} exp^{-\frac{1}{2\sigma^2}\left\|x_n-(sR_{y_k}+t)\right\|^2} + (2\pi\sigma^2)^{D/2}\frac{w}{1-w}\frac{M}{N}} \tag{2.7}$$

M-step: Solve for the optimal values of R, s, t, and σ^2

Compute the centered point sets $\hat{X}$ and $\hat{Y}$ by subtracting the mean vectors μ_x and μ_y, respectively, from the original point sets X and Y, where $N_p = 1^T P1$, $\mu_x = \frac{1}{N_p} X^T P^T 1$ and $\mu_y + \frac{1}{N_p} Y^T P1$

Compute the matrix $A = \hat{X}^T P^T \hat{Y}$.

Perform singular value decomposition (SVD) on A to obtain U, S, and V.

Compute the rotation matrix R as UCV^T, where C is a diagonal matrix with elements $(1, ..., 1, det(UV^T))$.

Compute the scale factor s as the trace of $A^T R$ divided by the trace of $\hat{Y}^T d(P1)\hat{Y}$, where $d(P1)$ is a diagonal matrix with the sum of each row of P as its elements.

Compute the translation vector t as $\mu_x - sR\mu_y$.

Compute the noise variance σ^2 as $1/(N_P D) \cdot (tr(\hat{X}^T d(P^T 1)\hat{X}) - s \cdot tr(A^T R))$, where N is the number of points, D is the dimensionality, and $d(P^T 1)$ is a diagonal matrix with the sum of each column of P as its elements.

Compute the aligned point set $T(Y)$ by applying the transformation (scale, rotation, and translation) to the target point set $Y : T(Y) = sYR^T + t^T$.

The probability of correspondence between each point x_n in the source point set and its corresponding point y_m in the aligned target point set $T(Y)$ is given by the corresponding element in the posterior probability matrix P.

Algorithm 3 Affine CPD [63]

Initialization:

Set the initial values for the affine transformation matrix B=I, translation vector t=0, and weight parameter $0 \leq w \leq 1$.

Compute the initial value for the noise variance σ^2 as:

$$\sigma^2 = \frac{1}{DNM} \sum_{n=1}^{N} \sum_{m=1}^{M} \|x_n - y_m\|^2 \tag{2.8}$$

EM Optimization: Repeat the following steps until convergence.

E-step:

Compute the posterior probability matrix P, which represents the probability of correspondence between points in the source and target point sets as:

$$p_{mn} = \frac{exp^{-\frac{1}{2\sigma^2}\left\|x_n-(B_{y_m}+t)\right\|^2}}{\sum_{k=1}^{M} exp^{-\frac{1}{2\sigma^2}\left\|x_n-(B_{y_k}+t)\right\|^2} + (2\pi\sigma^2)^{D/2}\frac{w}{1-w}\frac{M}{N}} \tag{2.9}$$

M-step: Solve for the optimal values of B, t, and σ^2

Compute the centered point sets $\hat{X}$ and $\hat{Y}$ by subtracting the mean vectors μ_x and μ_y, respectively, from the original point sets X and Y, where $N_p = 1^T P 1$, $\mu_x = \frac{1}{N_p} X^T P^T 1$ and $\mu_y + \frac{1}{N_p} Y^T P 1$

Compute the matrix B as $(\hat{X}^T P^T \hat{Y})(\hat{Y}^T d(P1)\hat{Y})^{-1}$, where $d(P1)$ is a diagonal matrix with the sum of each row of P as its elements.

Compute the translation vector t as $\mu_x - sB\mu_y$.

Compute the noise variance σ^2 as $1/(N_P D) \cdot (tr(\hat{X}^T d(P^T 1)\hat{X}) - tr(\hat{X}^T P^T \hat{Y} B^T))$.

Compute the aligned point set $T(Y)$ by applying the transformation to the target point set $Y : T(Y) = YB^T + 1t^T$.

The probability of correspondence between each point x_n in the source point set and its corresponding point y_m in the aligned target point set $T(Y)$ is given by the corresponding element in the posterior probability matrix P.

Algorithm 4 Non-Rigid CPD [63]

Initialization:
Set the initial weight matrix W to all zeros.
Compute the initial value for the noise variance.
Initialize the weight parameter w ($0 \leq w \leq 1$), the regularization parameter $\beta > 0$, and the regularization parameter $\lambda > 0$. σ^2 as:

$$\sigma^2 = \frac{1}{DNM} \sum_{n=1}^{N} \sum_{m=1}^{M} \|x_n - y_m\|^2 \tag{2.10}$$

Construct the matrix G, where each element g_{ij} represents the influence of point y_i on point y_j based on their Euclidean distance. $g_{ij} = exp^{\frac{-1}{2\beta^2}\|y_i - y_j\|^2}$
EM Optimization: Repeat the following steps until convergence.
E-step:
Compute the posterior probability matrix P, which represents the probability of correspondence between points in the source and target point sets.
For each point x_n in the source point set and each point y_m in the target point set, calculate the probability p_{mn} as:

$$p_{mn} = \frac{exp^{-\frac{1}{2\sigma^2}\|x_n - (y_m + G(m,\cdot)W)\|^2}}{\sum_{k=1}^{M} exp^{-\frac{1}{2\sigma^2}\|x_n - (y_m + G(m,\cdot)W)\|^2} + (2\pi\sigma^2)^{D/2}\frac{w}{1-w}\frac{M}{N}} \tag{2.11}$$

M-step:
Construct the matrix $A = G + \lambda\sigma^2 d(P^T 1)^{-1}$
Solve the linear system $A \cdot W = d(P1)^{-1}PX - Y$, where PX represents the matrix multiplication of P and X.
Compute the matrix $N_P = 1^T P1$, where T is the transpose operator given as $T = Y + GW$
Compute σ^2 as $\frac{1}{N_P D}(tr(X^T d(P^T 1)X) - 2tr((PX)^T T) + tr(T^T d(P1)T))$.
The aligned point set $T(Y,W)$ is obtained by adding the weighted matrix multiplication of G and W to the target point set $Y : T = Y + GW$.
The probability of correspondence between each point x_n in the source point set and its corresponding point y_m in the aligned target point set $T(Y)$ is given by the corresponding element in the posterior probability matrix P.

2.4 Learning Based Registration Methods

Learning-based point cloud registration methods can be categorized into feature learning methods and end-to-end learning-based methods.

Feature learning methods employ deep neural networks to learn robust feature correspondences. These methods streamline the registration process by incorporating a one-step estimation, such as RANSAC(2.3.2), to finalize the transformation matrix without requiring iterative steps. When the quality of learned features is good, accurate correspondences are estimated, improving the registration speed and accuracy, as demonstrated in 5.2. Developing sophisticated network architectures or loss functions in this category aims to estimate robust correspondences based on learned distinctive features. There are two main advantages to these methods. Firstly, deep learning-based point features enable more accurate and reliable correspondence searching [42]. Secondly, accurate correspondences contribute to achieving precise registration results using a simple RANSAC method [42]. However, there are certain limitations associated with these methods. Firstly, they require many training samples [42]. Secondly, the registration performance may significantly deteriorate in unknown samples with a substantial difference from the training data [42]. Thirdly, these methods employ a separate training process of a network to extract features, which focuses on point-to-point matching rather than registration [42].

End-to-end learning-based methods perform registration by employing neural networks that optimize the transformation estimation. Unlike feature-learning methods focusing on learning point features, these methods integrate the transformation estimation within the neural network optimization process. The neural network directly outputs the transformation matrix. Therefore, these methods convert the registration process to a regression problem [42]. For example, point features are learned in [97], and the transformation parameters are estimated by regression using the features [42]. [86] formulates the correlation using the shape descriptors obtained by the network between the source and target point sets and predicts the transformation based on the defined correlation [42]. Auto-encoder registration networks [22] and keypoint detection methods with simultaneous pose estimation are also explored in this category [57]. The advantage of end-to-end learning methods is that the neural network is specifically designed and optimized for the registration task, leveraging the strengths of both conventional mathematical theories and deep neural networks [42]. Also, these methods consider the local struc-

ture information crucial for accurate registration [42]. However, there are limitations to the current methods. Regression-based methods treat transformation parameter estimation as a black box, lacking interpretability [42]. Additionally, the distance metric used in these methods is measured in Euclidean space, which is sensitive to noise and density variations [42]. Secondly, the feature-learning registration methods consider local structure information, which is essential for point cloud registration [42].

2.4.1 Major Architectures

In this section, the focus will be on describing and discussing several state-of-the-art neural networks that are used for extracting features from a point cloud.

2.4.1.1 PointNet and PointNet++

PointNet [67] is a deep learning architecture that has been successfully applied to various computer vision tasks involving 3D point cloud data, including point cloud registration (PCR). The traditional methods for point cloud registration often rely on handcrafted features and iterative optimization algorithms. However, PointNet offers an alternative approach by leveraging deep learning techniques to directly learn feature representations from raw point cloud data and estimate transformations end-to-end. It takes a 3D point cloud as input, where each point has its XYZ coordinates and potentially additional attributes such as color or surface normals. The input point cloud can have varying numbers of points, making PointNet flexible for handling point clouds of different sizes. The core of PointNet is a shared MLP network that processes each point individually. The MLP consists of multiple fully connected layers, which can capture local features by learning non-linear mappings from the input point attributes. The shared weights across all points enable the network to capture permutation invariance, meaning the network's predictions are not affected by the order of the input points.

PointNet uses a max pooling function to create a global feature vector from a point cloud. Max pooling takes the maximum value of the features from each point in the point cloud and combines them into a single vector. This vector represents the overall structure and characteristics of the point

cloud. In PCR, it is crucial to consider the local geometric transformations of points. To address this, PointNet introduces a feature transformation network. This network learns a linear transformation for each local point feature, aligning the features to the global coordinate system. This alignment improves the network's ability to capture rotation and translation invariance, making it more robust to geometric transformations in the input point cloud. The output of PointNet is a set of features representing the input point cloud. These features can be further processed to estimate the transformation parameters that align the source point cloud with the target point cloud. While PointNet is primarily designed for 3D classification and segmentation tasks [23], it offers valuable insights and inspiration for the registration task. The descriptors utilized in segmentation and object classification tasks can be used as features to establish correspondences in registration. PointNet is trained in an end-to-end manner using supervised learning. The training data consists of pairs of aligned point clouds, where the ground truth transformations between them are known. The model is trained to minimize a loss function that measures the difference between predicted and ground truth transformations. Gradient descent optimization is used to update the network parameters iteratively, improving the model's ability to estimate accurate transformations.

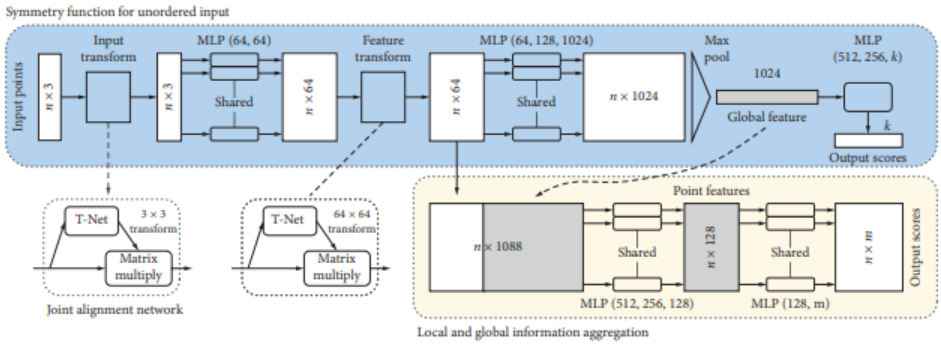


Figure 2.4: PointNet Architecture [51]

PointNet++ [68] is an extension of the original PointNet framework developed to directly process raw point cloud data. It introduces learning features by considering the local neighborhood of individual points by leveraging metric space distance [51]. Unlike PointNet, which focuses on learning the local features of the points and aggregating those features into a

global feature vector of the point cloud, PointNet++ employs a hierarchical neural network architecture. The hierarchical structure of PointNet++ comprises several levels of set abstraction. Each level has three main components: a sampling layer, a grouping layer, and a PointNet layer [51]. The sampling layer selects a subset of points from the input data. This helps to reduce the computational complexity and enables efficient processing. On the other hand, the grouping layer constructs local region sets around each point by grouping nearby points. This allows for capturing local patterns and relationships within the point cloud. The PointNet layer, a fundamental building block of PointNet++, aims to encode the local region patterns and produce feature vectors. It inputs the grouped points and applies operations to extract meaningful features. PointNet++ can capture fine-grained details and local structures in the point cloud by encoding the local region patterns. The architecture of PointNet++ is illustrated in Figure 2.5. This architecture's input consists of N points with d-dimensional coordinates and C-dimensional features. The number of centroid points' neighborhoods, denoted as K, plays a crucial role in determining the local context captured by the network.

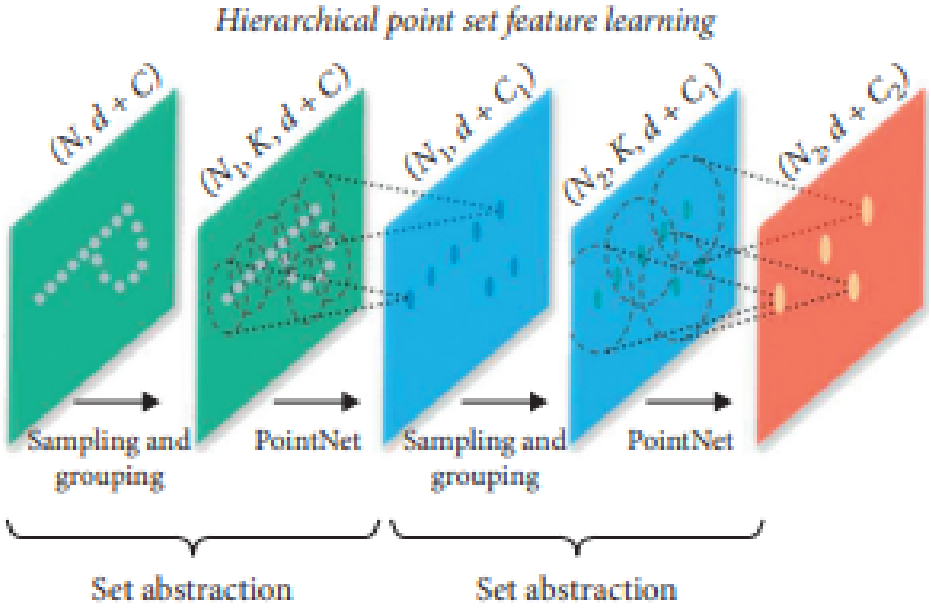


Figure 2.5: PointNet++ Architecture [51]

2.4.1.2 DGCNN

DGCNN (Dynamic Graph CNN) [89] is a deep learning architecture designed for point cloud analysis tasks. It was proposed as an extension of the traditional Convolutional Neural Network (CNN) to address the unique challenges of irregular and unstructured point cloud data. DGCNN incorporates dynamic graph structures and edge convolutions to capture spatial relationships and local patterns within the point cloud. DGCNN leverages the concept of graph-based representations, where the point cloud is treated as a graph, and the relationships between points are modeled using edges. DGCNN begins with a PointNet layer, which extracts local features from individual points. The PointNet layer processes each point independently, capturing local information through shared MLP networks. This allows PointNet to encode each point's spatial and attribute-based information individually. The next step is constructing a dynamic graph representing the relationships between points. It is crucial for capturing the point cloud's spatial dependencies and connectivity patterns. DGCNN achieves this by employing a k-nearest neighbors (k-NN) algorithm to find the local neighbors of each point. These neighbors are nodes in the graph, and edges are established between them based on proximity. With the dynamic graph in place, DGCNN performs edge convolutions to capture spatial features and propagate information between neighboring points. Edge convolutions operate on the graph structure and are designed to leverage both local and global information. Each edge convolution layer aggregates information from neighboring points by considering their features and edge connections. This allows the network to capture hierarchical patterns and relationships within the point cloud. It employs graph pooling, which reduces the size of the graph while preserving important features. Graph pooling operates by selecting representative points based on their features and connectivity, effectively downsampling the point cloud. This downsampling process allows the network to capture global context while reducing computational complexity.

After several iterations of edge convolutions and graph pooling, DGCNN concludes with fully connected layers for higher-level feature learning and task-specific predictions. The fully connected layers aggregate information from the entire point cloud and produce the final output, which can be used for various applications such as classification, segmentation, or

object detection. DGCNN combines the strengths of graph-based representations and convolutional neural networks to analyze point cloud data effectively. It can capture local structures, spatial relationships, and hierarchical patterns within the point cloud by incorporating dynamic graphs and edge convolutions.

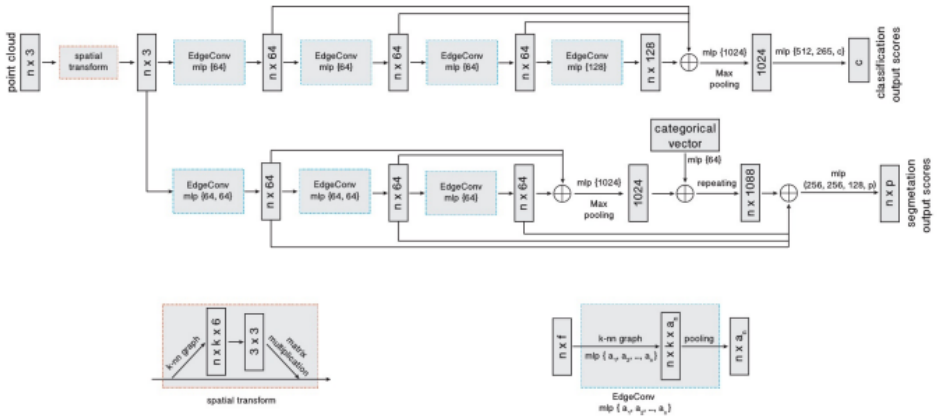


Figure 2.6: DGCNN Architecture [89]

The architecture of DGCNN is given in figure 2.6 where the top branch is used for classification, and the bottom branch is used for segmentation. The DGCNN classification model takes n points as input and computes a feature set for each point using the EdgeConv layer. An edge feature set contains information about the relationships between a point and its neighboring points. After computing the edge feature sets for all points, the model aggregates the features within each set to generate EdgeConv responses for the corresponding points. This aggregation process combines the local information from neighboring points to enhance the feature representation of each point. In the final EdgeConv layer, the output features from all points are globally aggregated. This means the model considers all points together to create a $1D$ global descriptor. This global descriptor summarizes the collective information from the entire point cloud, effectively encoding the overall structure and characteristics of the data. The $1D$ global descriptor is then used as the input for a classification layer.

The segmentation model combines the 1D global descriptor with the EdgeConv outputs using concatenation operation $\bigoplus$ [89]. The EdgeConv outputs serve as local descriptors, providing information about each point's

immediate neighborhood [89]. Once the descriptors are combined, the model generates per-point classification scores for p semantic labels. These scores represent the likelihood or probability of each point belonging to a specific semantic label. The point cloud transform block aligns an input set of points to a standard or canonical space. This alignment is achieved by using a 3×3 matrix. To estimate this matrix, a tensor is created by combining the coordinates of each point in the point cloud with the coordinate differences between the point and its k neighboring points [89]. The EdgeConv block takes an input tensor with a shape of $n \times f$, where n is the number of points and f is the number of features associated with each point. The block then uses an MLP to calculate edge features for each point. The number of neurons in the MLP is given as $\{a_1, a_2, ..., a_n\}$, which means there are n layers, and each layer has a different number of neurons. After computing the edge features for each point, the block performs pooling among neighboring edge features. It aggregates the edge features of neighboring points to generate a tensor with a shape of $n \times a_n$.

2.5 Evaluation Metrics

Several evaluation metrics and distance measures are commonly used for point cloud registration.

Chamfer Distance: Chamfer distance measures the average distance between two point clouds. It quantifies how well the points in one point cloud align with those in the other. The Chamfer distance can be calculated using the following equation [92]:

$$d_{CD}(S,T) = \frac{1}{|S|}\sum_{x\in S}\min_{y\in T}\|x-y\|_2 + \frac{1}{|T|}\sum_{y\in T}\min_{x\in S}\|y-x\|_2 \tag{2.12}$$

where S and T are the point clouds, $|S|$ and $|T|$ represent the number of points in S and T respectively, x and y are points in S and T, and $\|\cdot\|$ denotes the Euclidean distance between two points. The equation computes the average of the minimum distances from each point in S to the nearest point in T and vice versa. The lower the Chamfer distance, the better the alignment between the point clouds.

Hausdorff Distance: Hausdorff distance measures the maximum distance between two point clouds. It captures the largest separation between cor-

responding points in the two point clouds. The Hausdorff distance can be calculated using the following equation:

$$H(S,T) = \max\big(h(S,T), h(T,S)\big) \tag{2.13}$$

where $h(S,T)$ represents the directed Hausdorff distance from point cloud S to T and is defined as [74]:

$$h(S,T) = \max_{s\in S} \min_{t\in T} \|s-t\| \tag{2.14}$$

Here, s and t are points in S and T, $\|\cdot\|$ denotes the Euclidean distance, and the equation computes the maximum of the minimum distances from each point in S to the nearest point in T.

Feature-match recall: The feature-match recall evaluates the proportion of pairs of fragments that can accurately recover the pose with a high confidence level. It can be expressed mathematically as follows [102]:

$$R_{fa} = \frac{1}{M}\sum_{s=1}^{M} 1\Big(\Big[\frac{1}{|\Omega_s|}\sum_{(i,j)\in\Omega_s} 1\Big(\|T^* x_i - y_j\| < \tau_1\Big)\Big] > \tau_2\Big) \tag{2.15}$$

In evaluating feature match recall, a scenario is considered where a total of M fragment pairs exist. Each fragment pair, denoted by s has a corresponding set of correspondences called Ω_s. These correspondences contain the 3D coordinates of matching points in the first and second fragments, represented as x and y, respectively. The ground-truth pose between these fragments is $T^* \in SE_3$. To assess the performance of the feature-matching algorithm, two thresholds are introduced. The first threshold, τ_1, determines whether a correspondence is considered an inlier by comparing its distance. Correspondences with a distance below τ_1 are classified as inliers, indicating a successful match. The second threshold, τ_2, is the recall threshold, which establishes the minimum fraction of inlier correspondences required for a fragment pair to be deemed as having a high-confidence pose recovery. The feature match recall can be evaluated by comparing the correspondences Ω_s with the ground-truth pose T^*. The percentage of fragment pairs is calculated where the number of inlier correspondences exceeds the recall threshold τ_2. This measurement provides insight into the algorithm's ability to accurately recover the pose by determining the proportion of fragment pairs with an adequate number of inlier correspondences above the recall threshold [102].

Registration Recall: Registration recall evaluates the performance of a matching algorithm by considering a collection of overlapping fragments

and comparing them to the ground truth pose. It aims to determine how accurately the algorithm can recover the overlapping fragments. To calculate registration recall, an error metric is utilized. This error metric compares the estimated fragments $\{i, j\}$ obtained from the matching algorithm to the corresponding pose estimation $\hat{T}_{i,j}$. Based on this comparison, a true positive is defined as [102]:

$$E = \sqrt{\frac{1}{\Omega^*} \sum_{(x^*, y^*) \in \Omega^*} \| \hat{T}_{i,j} x^* - y^* \|^2} < \tau_3 \tag{2.16}$$

The registration recall evaluates the correctness of fragment pairs by comparing them to the corresponding ground-truth pairs. The ground-truth pairs are denoted by Ω^*, and their 3D coordinates are represented by x^* and y^*. The threshold τ_3 determines if a fragment pair is considered correct when they overlap. The recall metric is significant because it improves precision by implementing more effective pruning techniques. This observation has been highlighted in various studies [32, 73].

MRAE: Mean relative angular error measures the average difference between the estimated and ground truth poses across multiple instance. Each pose estimation instance calculates the relative rotation error to compute the mean relative angular error. The angular difference's magnitude is measured by taking the absolute value of the relative rotation error for each instance. The mean relative angular error is obtained by averaging all the absolute relative rotation errors. This metric assesses the average angular deviation between the estimated and ground truth poses, giving insight into the accuracy of the pose estimation algorithm. Mathematically, the mean relative angular error (MRAE) can be expressed as [102]:

$$MRAE = mean\Big(\cos^{-1}\Big(\frac{trace(R_{pre}^{-1} R_{gt}) - 1}{2}\Big)\Big) \tag{2.17}$$

where R_{pre} is the estimated rotation matrix and R_{gt} is the ground truth rotation matrix. A lower mean relative angular error value indicates a more accurate pose estimation algorithm, which signifies that the estimated rotations closely match the actual rotations. On the other hand, a higher value suggests more significant angular deviations and poorer performance in estimating the poses.

MRTE: The Mean Relative Translation Error is a metric used to evaluate the accuracy of estimated translations in point cloud registration. It measures

the average relative difference between the estimated and ground truth translations. The formula for calculating MRTE is as follows [102]:

$$MRTE = mean\Big(\| t_{pre} - t_{gt} \|_2\Big) \tag{2.18}$$

where t_{pre} represents the predicted translation vector and t_{gt} represents the ground truth translation vector.

In addition to the overall evaluation of registration accuracy, other metrics can be used to assess registration quality. These metrics include the mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). Ideally, in a perfect alignment, all these error metrics should be zero, indicating a perfect match between the ground truth and predicted values. However, in practical scenarios, it is rare to achieve zero error due to various factors such as noise, outliers, and limitations of the registration algorithm.

3
Related Work

In recent years, there has been significant research in point cloud registration, focusing on leveraging deep learning techniques to improve alignment accuracy and robustness. This section overviews several notable papers in point cloud registration, highlighting their critical contributions and methodologies.

PointNetLK [1] is a deep neural network framework that combines PointNet with the Lucas-Kanade (LK) algorithm [58]. It utilizes PointNet to generate K-dimensional vector descriptors and estimates the current transformation, $\triangle G$, through an exponential map in the LK layer [51]. This allows for alignment estimation without the need for computationally expensive point correspondences. The final transformation, G_{est}, is obtained by aggregating all the incremental estimates computed during the iterative loop [51]. Evaluation on the ModelNet40 [93] dataset shows that PointNetLK exhibits greater robustness to noisy data and is less sensitive to the initial position compared to the ICP algorithm [51].

Deep Closest Point (DCP) [87] was introduced as an alternative to address the limitations of the ICP algorithm. DCP comprises three major components: a point cloud embedding network, an attention-based module for probabilistic soft-matching, and an SVD layer for rigid transformation estimation [51]. Initially, the source and target point clouds are embedded in high-dimensional feature vectors using PointNet or DGCNN. An attention module is used to extract co-contextual information from the feature vectors, and a pointer generation module is employed to establish soft correspondences between the point clouds [87]. The transformation between the point clouds is calculated using a differentiable SVD layer [87]. The DCP-v2 variant incorporates an attention module, unlike DCP-

v1. Comparatively, DGCNN-based DCP performs better in terms of accuracy on the ModelNet40 dataset than PointNet-based DCP [87]. However, DCP alone may not achieve sufficient accuracy, and further refinement can be achieved by combining it with the ICP algorithm, leveraging DCP's improved initialization for convergence to the global optimum [51].

Robust Point Matching Network (RPM-Net) [98] is a rigid registration method less sensitive to initialization. RPM-Net consists of three main components: (1) Computation of hybrid features by combining spatial coordinates and geometric properties using PointNet [51]; (2) Estimating the correspondences using a differentiable Sinkhorm layer [80] and annealing algorithm [51]; (3) Estimating transformation using SVD [51]. To train the network, l^1 distance between predicted and ground truth transformation is used as a loss function.

Deep Global Registration (DGR) [15] is a registration method consisting of three key modules. The first module is a 6D convolutional network that estimates the accuracy of correspondences between pairwise point clouds. The second module utilizes a weighted Procrustes solver to perform a rigid registration based on the estimated correspondences. The final module is an optimization module that further refines the registration by minimizing a robust loss function. DGR employs fully convolutional geometric features (FCGFs) [16] to extract features from the point clouds and uses a convolutional network to identify correct correspondences whose output is the same regardless of the position of points. The Procrustes method [31] is then employed to minimize the weighted mean squared error and obtain the rotation and translation parameters. A fine-tuning module is proposed to enhance registration accuracy by minimizing the robust loss function and achieving smooth optimization in the registration space.

Partial Registration Network (PRNet) [88] addresses the challenges of partial correspondence problems by leveraging self-supervised learning to learn geometric priors. It consists of two main components: an appropriate geometric representation and a keypoint detector that identifies common points and establishes correspondences. Similar to the iterative nature of the ICP algorithm, PRNet is designed to perform coarse-to-fine refinement. Initially, PRNet detects critical points in both points clouds S and T and then predicts a mapping and rigid transformation (m_p) from S to T based on these critical points. The obtained transformation is ap-

plied to S, which is iteratively repeated with the updated transformation as input. Notably, to achieve sharper and approximately differentiable results compared to traditional methods, PRNet incorporates an actor-critic approach using Gumbel-Softmax [44] to sample a matching matrix for predicting the mapping of critical points from S to T [51].

PCRNet [77] is a novel framework for point cloud registration that also leverages PointNet features like PointNetLK to estimate the transformation between point clouds. It employs a Siamese architecture consisting of five MLPs, similar to PointNet. These MLPs encode the point cloud into global feature vectors that capture geometric and orientation information. Unlike PointNetLK, which relies on traditional algorithms for transformation calculation, PCRNet utilizes fully connected (FC) layers to estimate the transformation in a single forward pass. The input point clouds are processed by the MLPs with 64, 64, 64, 128, 1024 filters in each 2D convolutional layer, and the resulting global features are concatenated and fed into the FC layers with 1024, 1024, 512, 512, 256, 7 filters in each layer to obtain the translation and rotation values. Experimental results on the ModelNet40 [93] dataset demonstrate that PCRNet achieves similar accuracy to Go-ICP [95] but with significantly faster computation [51].

The Non-Rigid Point Set Registration Networks [86] consists of three key components. The first component focuses on learning a shape descriptor tensor, a structured grid that allows for feature learning and correlation analysis of diverse and irregularly-structured data. The second component is dedicated to learning a shape correlation tensor, which measures the correlation between two shape descriptor tensors of the point sets being registered [86]. This tensor is calculated by performing point-wise operations on the shape descriptor tensors obtained in the previous component. The third component involves learning the transformation parameters by establishing a mapping function between the space of the shape correlation tensor and the transformation parameters. PRNet utilizes a Convolutional Neural Network (CNN) as a regression model to achieve this. The CNN approximates the mapping function and learns the desired transformation parameters required for the registration process. The primary objective of the PRNet framework is to achieve statistical alignment between the source and target point cloud sets. This is accomplished by optimizing the geometric transformation by utilizing learned parameters.

CorrNet3D [100] is a deep learning framework that can learn the dense correspondence between 3D shapes without any annotated data. It first learns point-wise features from a pair of raw point clouds. Then, it uses a correspondence indicator to generate a correspondence matrix representing point-to-point correspondences. Finally, a symmetric deformer is used to transform the permuted point clouds, which helps learn the unsupervised correspondences. CorrNet3D differs from previous works because it focuses on deformation-like reconstruction to drive the correspondence matrix instead of relying on ground-truth correspondences or functional maps. CorrNet3D can also be used as a supervised model by removing the deformation module and using ground-truth correspondences to supervise the learning of the correspondence matrix.

The Deep Graph Matching Based Dense Correspondence Learning Between Non-Rigid Point Clouds (CorrDGM) [59] is designed to learn dense correspondences between non-rigid point clouds using deep graph matching techniques. It exploits the capabilities of deep learning and graph matching to establish meaningful correspondences between points in different non-rigid shapes. Initially, the point clouds are represented as graphs, with each point serving as a node and the edges representing pairwise relationships or similarities. These graph representations are then processed by a deep neural network that learns to encode the graph structure and extract informative features from the nodes. The core step involves graph matching, where the algorithm identifies correspondences between nodes in the two graphs, effectively mapping points from one shape to their corresponding points in the other. It is accomplished by optimizing an objective function that quantifies the similarity or agreement between the graphs. To optimize the objective function, the algorithm utilizes graph-matching algorithms that consider both the graph structure and the node features. It aims to find an optimal matching that maximizes the similarity between the graphs. During training, the parameters of the deep neural network and the graph-matching algorithm are learned using labeled or unlabeled data. Labeled data consists of pairs of point clouds with known correspondences, while unlabeled data consists of point clouds without pre-defined correspondences. The algorithm establishes dense correspondences between non-rigid point clouds through data-driven learning, enabling shape alignment, reconstruction,

and other applications that rely on point-wise correspondence information.

Various deep learning-based algorithms exist for point cloud registration, and the ones mentioned above are just a few examples. It is essential to highlight that deep learning in point cloud registration continually evolves, and new methods are being developed regularly. Researchers are constantly exploring innovative approaches to improve point cloud registration accuracy, robustness, and efficiency using deep learning techniques. Therefore, the list of algorithms mentioned earlier is incomplete, and more advanced and sophisticated methods are likely available beyond those discussed.

4 Methods

This chapter provides a detailed explanation of the methods employed in the experiments. Three distinct approaches for point cloud registration are explored, each offering unique characteristics and advantages. These three approaches were selected based on their focus on utilizing overlapping points for transformation estimation rather than considering the entire point cloud for feature computation.

The first method is based on ROPNet [107], an end-to-end learning-based approach for rigid registration. ROPNet has demonstrated superiority over established algorithms such as DCP [87], DeepGMR [99], and RPMNet [98], making it a compelling choice for the experiments. The second method relies on DIP [66], which adopts a feature-learning approach to point cloud registration. DIP learns distinctive features for individual points and utilizes RANSAC for final transformation estimation. DIP's flexibility is one advantage that does not require prior voxelization. It also allows for a comparison of the extracted features with FPFH [72]. The third method is based on C2P-net [54]. Initially, C2P-net performs rigid registration using GCNet [106], a method demonstrating state-of-the-art performance on the 3DMatch dataset. Subsequently, C2P-net refines the registration using Neural Deformation Pyramid (NDP) [53]. NDP is particularly suitable for the data used in this thesis since no ground truth is available for the warping function, and it does not require pretraining.

4.1 Dataset

The data acquisition information provided further is taken from [29]. The PC MRI data was collected from a healthy volunteer using a 7 T whole-

body MRI system by Siemens Healthineers using a 32-channel head coil manufactured by Nova Medical. The acquisition sequence involved radio-frequency (RF)-spoiled gradient echo with quantitative flow encoding in all three spatial dimensions [61, 62]. Electrocardiogram (ECG) gating information was obtained from an acoustic cardiac gating device by MRI.Tools GmbH Berlin. The acquired data comprises 17-time steps, with each step containing three velocity maps representing the x-, y-, and z-velocity components, along with a structural magnitude image. The voxel size of the acquired images is isotropic at 0.64mm, and the temporal resolution is 54.4 ms. The velocity encoding parameter (VENC), which determines the maximum velocity uniquely encoded in the phase, was set to 0.9m/s. The signal-to-noise ratio (SNR) was measured to be approximately 55. Postprocessing of the raw data was conducted using MeVisLab 2.3.1 and the automated tool described in Bock et al. [7]. Furthermore, a high-resolution TOF MRI was carried out on the same healthy volunteer with a voxel size of 0.32mm.

Initially only one pair of TOF and PC MRI mesh is available. These meshes are obtained by segementation of TOF and PC MRI images as described in [82]. To train the models large number of samples are required. To do this various augmentations and data processing is performed to generated the required dataset for training and testing of the methods.

4.1.1 Normalization

The PC mesh consists of 23,985 vertices, while the TOF mesh contains 122,935 vertices, resulting in a significant difference in size between the two meshes. However, processing the meshes while maintaining their original size is computationally infeasible. Sampling a random number of points from the meshes leads to the loss of shape and vessel structure, as observed through various experiments and visual comparisons with the original vertex count.

To address this, the TOF mesh is first scaled to fit into a unit cube. The scaling factor is determined as 1/max(maximum bound – minimum bound), where the maximum bound and minimum bound are vectors of size three with the maximum and minimum values across each dimension, respectively. The PC mesh is then fitted into the same cube using the scaling

factor determined for the TOF mesh. Rescaling the PC mesh with the scaling factor used for the TOF mesh is necessary because when the PC mesh is fit into the cube, its vessel volume is larger than that of the TOF mesh. This is due to the fact that more points need to be fit inside a unit cube for the PC mesh compared to the TOF mesh. After scaling the meshes, 20,000 points are uniformly sampled from the PC mesh, and 50,000 points are sampled from the TOF mesh to generate the point clouds using Open3d [105]. These point clouds are then voxelized with a voxel size of 0.01. A global registration is performed using RANSAC provided by Open3d to bring the point clouds into a common reference frame. Optionally, the TOF point cloud can be automatically cropped by determining the dimensions of a cube that fits the PC mesh. Only the points within this cube are retained in the TOF point cloud, ensuring that it contains points within the range of the PC cloud.

Finally, approximately 3,000 points are sampled from the PC point cloud, and 4,000 points are sampled from the TOF point cloud using Poisson disk sampling provided by Open3d. The reason for sampling points is primarily due to hardware limitations, as more points would ideally be used to avoid information loss. After completing these steps, the meshes are transformed into pairs of point clouds, which can be further utilized for analysis and experimentation.

4.1.2 Augmentation

Using the samples obtained in the previous steps, 200 augmented point cloud pairs are generated. These point clouds are generated by randomly rotating the original point clouds within the $[-90, 90]$ degrees range. The rotation process involves generating a random angle for each axis using the Numpy library. These angles are then used to create a rotation matrix for each axis. The final rotation matrix is obtained by taking the dot product of the rotation matrices for the y and z-axes with the rotation matrix for the x-axis. An example of a rotation matrix is given below, where θ represents a random axis angle [91]:

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

After generating the initial 200 samples, an additional 200 samples are generated by translating them within the $[-1.5, 1.5]$ range. This translation is performed by generating a random vector of size 3 using Numpy and adding it to all the points in the point clouds row by row. Finally, all the generated samples undergo jittering with a sigma value of 0.001 and a clip value of 0.005. An array with the same shape as the input point cloud is created to achieve this. The array is initialized with random values drawn from a normal distribution with a mean of 0 and a standard deviation of sigma. The clipping value is then applied to the generated array, limiting the values within the range of -clip to clip. This ensures that the jittered points stay within the original. The generated array is then added element-wise to the original point cloud, resulting in the final jittered point clouds.

4.1.3 Quality Check

This quality check ensures that the augmentations applied to the point clouds do not alter their geometric and semantic meaning. To assess the quality of the generated samples, the Chamfer and Hausdorff distances are calculated for each sample compared to the original point clouds. Furthermore, PointNet, a classification network, is utilized to evaluate the quality of the generated samples. The PointNet network is trained on a dataset comprising PC and TOF point clouds generated from the samples mentioned earlier. The training is performed using samples where the average Chamfer and Hausdorff distances are below the average of all the samples. The PointNet network is based on the PointNet classification example provided by the Keras library [45]. After training, the trained PointNet network predicts whether the remaining point clouds belong to PC or TOF. Any samples incorrectly predicted by the network or with a confidence value below 80% are removed from the dataset and not used for the subsequent registration process. This ensures that only high-quality and reliable samples are considered for registration.

4.1.4 Registration

The registration methods used in this work require ground truth transformation matrix to train the networks. Therefore, the generated point clouds are registered using RANSAC+ICP provided by Open3d [105] and

then ground truth information like transformation matrix, set of correspondences and inlier ratio is extracted. The TOF and PC clouds are randomly paired and registered. A value of 0.07 was selected after experimenting with different values as distance threshold criteria for RANSAC and ICP. Other registration techniques like rigid CPD, affine CPD and non-rgid CPD provided by probreg [84] library is also performed to have four different versions of the dataset. But only the dataset registered using RANSAC+ICP is used by the registration methods as the quality of registration of other methods were not satisfactory. Finally, there are 517 pairs of registered point clouds with ground truth transformations.

4.2 End-to-End Learning-based Rigid Registration

ROPNet [107] is a feature-learning-based method that focuses explicitly on situations where the points overlap. The model extracts essential information from these overlapping points to improve the accuracy of the alignment. It does this by learning discriminative features or unique characteristics that help distinguish or identify the points. These representative points capture the essential characteristics of the point clouds and enable accurate registration. The transformation is achieved using a subset of overlapping points between the two point clouds. ROPNet has a couple of modules that help in the registration process. The first module, the context-guided (CG) module, predicts a score that tells us how much the points overlap. It is done by looking at the features of the points and how they interact with the overall features of the entire point cloud. This helps determine the level of overlap between the points. Another module called the Transformer-based feature matching removal (TFMR) module is used to enhance the identification of representative points, and it enriches the features of the points and estimates a similarity matrix based on these features. This matrix helps identify the representative overlapping points more accurately. Non-representative points are then removed from one of the point clouds, transforming the registration task from partial-to-partial (where only some points overlap) to partial-to-complete (where all points are aligned).

The model uses SVD to estimate the transformation $T \in SE(3)$ that describes the alignment between the point clouds by combining the transfor-

mations from the CG and TFMR modules. Figure 4.1 illustrates the step-by-step workflow of the ROPNet model.

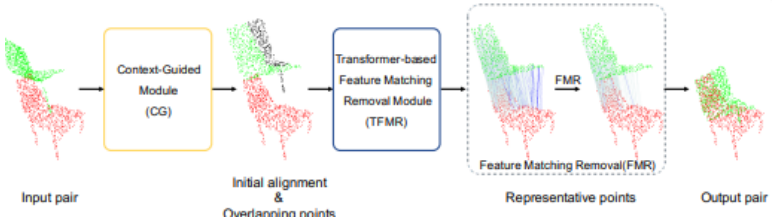


Figure 4.1: ROPNet Pipeline [107]

4.2.1 Context-guided (CG) module

The CG module uses a PointNet network as an encoder that extracts global features from the point clouds and predicts a 7D vector representing the transformation [107]. The first four values represent the rotation, and the last three represent translation. When the initial rotation is small, an initial transformation is used to enhance point feature learning, and this helps improve the accuracy of correspondence, which is crucial for successful point cloud registration [39]. The equation defines the initial alignment step [107]:

$$v = h_\phi\big(cat(\max(f_\theta(X)), \max(f_\theta(Y)))\big) \tag{4.1}$$

where $f_\theta(\cdot)$ is the encoder network that results in high dimensional features $F_X \in \mathbb{R}^{N\times C}$ and $F_Y \in \mathbb{R}^{M\times C}$ for the PC point cloud X and TOF point cloud Y. $\max(\cdot)$ is the max-pooling operation performed on the feature vectors, and cat is the concatenation of features from both clouds. $h_\phi(\cdot)$ is the decoder network made of a simple MLP. The transformed PC point cloud X' is obtained by multiplying the transpose of the rotation vector with the original PC point cloud and adding the transpose of the translation vector to each point. In this work, the encoder consists of five 1D convolution layers with [192,192,192,384,1536] filters in each layer respectively, while the original paper has filter sizes of [64,64,64,128,512]. Each filter uses a kernel size of 1, stride equal to 1, and padding equal to 0. A group norm layer and a ReLU activation function follow each convolutional layer. The MLP for transformation prediction contains fully connected layers with [1536,1536,768,7] filters, while the original works have [512,512,256,7] filters. Each fully connected layer has a ReLU activation

function. This work adds a dropout layer to each fully connected layer with a probability of 0.2 to reduce overfitting.

To predict overlapping points, utilizing the overlap information from the PC and TOF point clouds is crucial [107]. In order to achieve this, an information interaction module is used that operates on the global features extracted during the initialization step. The objective is to solve the overlap prediction problem, which predicts whether a point is an overlapping point by performing a binary classification. The overlap score can be computed using the following equations [107]:

$$O_X = g_\psi\big(cat(F_X, r(F_X^g), r(F_Y^g), r(F_X^g - F_Y^g))\big), \tag{4.2}$$

$$O_Y = g_\psi\big(cat(F_Y, r(F_Y^g), r(F_X^g), r(F_Y^g - F_X^g))\big) \tag{4.3}$$

Here, $r(\cdot)$ represents repetition of concatenated global features to match the size of features from the encoder. The overlap decoder g_ψ, which is also a PointNet architecture, is applied to the concatenated features. A softmax function is then applied to obtain the point overlap scores O_X and O_Y for the PC and TOF point clouds. It is worth noting that the parameters of g_ψ are shared between the input point clouds. The overlap decoder network has the architecture of the encoder with [1536,1536,768,2] filters, while the original work contained [512,512,256,2] filters.

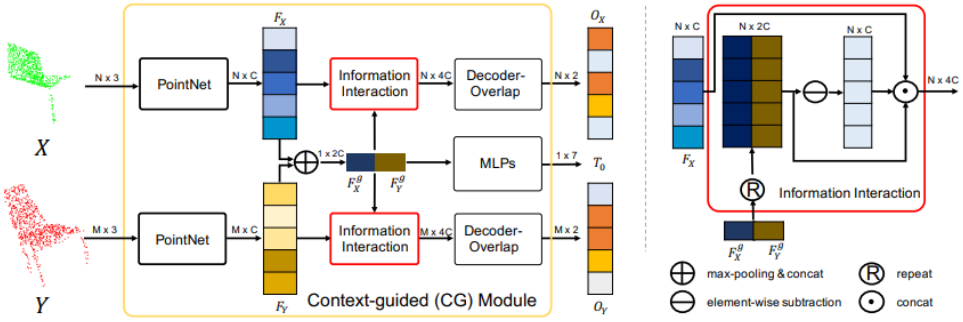


Figure 4.2: Architecture of CG module (left) and Information Interaction module (right) [107].

In this work, all the networks mentioned earlier are three times larger than the original. It is because the data used in this work is more complex and has more number of point. Therefore, a more extensive network with more learnable parameters was required for efficient learning.

4.2.2 Transformer-based Feature Matching Removal

The TFMR module removes non-representative points from the PC point cloud and its architecture is given in figure 4.3. To enhance the point features, a feature augmentation step is performed. The k-nearest neighbor points M_k within a specified radius are identified for each point x in the transformed PC point cloud X'. After experimenting with different radius values, the best results were achieved using a value of 0.15. These neighbors' spatial coordinates and point pair features [20] PPF_i for each m_i in M_k are concatenated to create augmented feature vectors $F^a_{m_i} \in \mathbb{R}^{10}$. Max-pooling is applied to these augmented features $F^a_{m_i}$ to obtain the most significant features $F^p_x \in \mathbb{R}^{C_p}$ for each point. It is done by a series of 1-dimensional convolutional layers represented by the function μ_θ. The max operation is employed to select the most significant features. The network contains three convolutional layers with [256,512,192] filters compared to [128,128,192] filters in the original work. Each convolution layer contains a group norm and ReLU activation function.

The Transformer module incorporates point features by considering all input points, which has advantages over local features [52, 55, 68, 85, 89, 103, 107]. It uses a self-attention mechanism to generate point features based on query, key, and value matrices represented by Q_i, K_i, and V_i, respectively. The self-attention weights s_i are computed using the softmax function applied to the matrix multiplication of the query Q_i and the transpose of key matrices, i.e., K_i^T. The feature matrix is then updated by multiplying the self-attention weights with the value matrix giving $F^{i+1}_{X'}$. The Transformer module consists of five self-attention and Feature Forward Neural Network (FFNN) groups in this work. The resulting feature matrices for X' and Y are denoted as $F^t_{X'}$ and F^t_Y respectively, with dimensions $\mathbb{R}^{N\times 4C_p}$ and $\mathbb{R}^{M\times 4C_p}$ respectively.

In the feature matching removal step, a similarity matrix H is computed by multiplying the feature matrices of the transformed PC point cloud and the TOF point cloud, i.e., $F^t_{X'}$ and ${F^t_Y}^T$. Non-overlapping points are removed from the transformed PC point cloud to ensure that the remaining points X'_{o1} satisfy $\sum_j^M H_{o1,ij} = 1$ for all i. This is necessary as there could be points in the transformed PC point cloud that do not have corresponding

points in the TOF point cloud. By performing this step, partial-to-partial registration can be effectively addressed [107].

A two-step process is followed to obtain representative overlapping points. Firstly, top-N1 (996 in this work) overlapping points $X'_{o1} \in \mathbb{R}^{N_1 \times 3}$. are selected based on the scores from the CG module. Secondly, feature matching is performed to refine the set of representative points by selecting points with the most defining features. The final set of representative points X'_{o2} is obtained. To calculate X'_{o2}, the indices of the points in X'_{o1} with the highest similarity scores based on the probability proportion $prob$ are identified. The probability used in this work is 0.6 during training and 0.4 during testing. Using these indices, the similarity matrix H_{o2} is computed by multiplying the transposed feature matrix $F^t_{X'_{o2}}$ with the feature matrix F^t_Y for the representative overlapping points. This transformation enables the transition from partial-to-partial registration to partial-to-complete registration [107].

For each point, $x_{o2,i}$ in the final set of representative points X'_{o2}, a corresponding point y_i in the TOF point cloud is selected having top-k maximum similarity scores. The top three points are selected during the training phase, while only the top one is selected during testing. The weight matrix w_{ij} for each $x_{o2,i}$ concerning these corresponding points is computed using these indices. The coordinates of the corresponding point are obtained by summing the weighted coordinates of the points in the TOF point cloud, where the weights are determined based on the similarity scores.

Each point pair is defined as $\left(x_{o2,i}, y_i\right)$ with a weight $O_{X'_{o2},i}$ represents the set of correspondences, and weighted SVD is used to estimate the transformation. The final transformation is obtained by combining the initial transformation R_0 with the estimated transformation R_1, and the translation is computed as the sum of the initial translation t_0 multiplied by the estimated rotation and the estimated translation t_1.

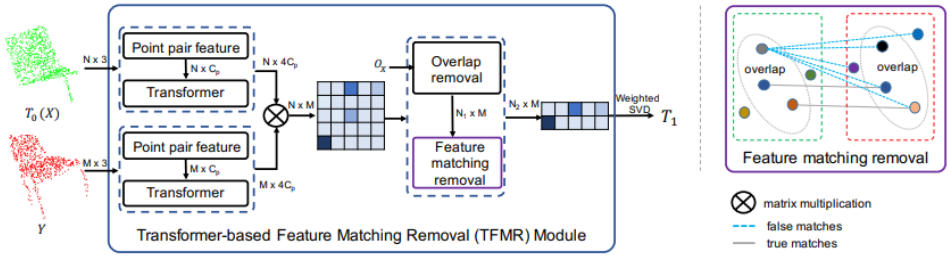


Figure 4.3: TFMR Module (left) and Feature Matching Removal (right) [107].

4.2.3 Loss function

The loss function measures the difference between the predicted transformed PC point cloud and the actual transformed PC point cloud. For the prediction of overlap scores a cross-entropy loss is used. To improve the training process, an additional loss is included by the authors that consider the initial transformation parameters R_0 and t_0:

- The loss for the transformed PC point cloud is calculated by taking the absolute differences between X transformed by R^T and X transformed by $\hat{R}^T$ and adding it to the absolute differences between t and $\hat{t}$.

- The auxiliary loss for the initial transformation is computed similarly, comparing X transformed by R_0^T and X transformed by $\hat{R}^T$ for rotation, and comparing t_0 and $\hat{t}$ for translation.

An overlap loss, L_{ol}, which evaluates the agreement between the predicted overlap scores O_X and O_Y and the actual scores $\hat{O}_X$ and $\hat{O}_Y$ is also defined. The ground truth overlap score is determined using the mapping Φ_X obtained from equation 4.4 [107]. This mapping finds the nearest neighbor in Y for each point X_i, and if the correspondence falls below a certain threshold d, which is 0.15 in this work, the score is set to 1; otherwise, it is set to 0.

$$\Phi_X = \left\{X_i | NN\left(\hat{R} \cdot X_i + \hat{t}, Y\right) < d, \forall i\right\} \tag{4.4}$$

The overlap loss is computed by taking the element-wise product of $\left(\hat{O}_X\right)_{ij}$ and $\log\left(O_X\right)_{ij}$, summed over all i and j, multiplied by $1/2N$; and by taking

the element-wise product of $\left(\hat{O}_Y\right)_{ij}$ and $\log\left(O_Y\right)_{ij}$, summed over all i and j, multiplied by $1/2M$ [107].

$$L_{ol} = \frac{1}{2N}\sum_i\sum_j\left(\hat{O}_X\right)_{ij}\cdot\log\left(O_X\right)_{ij} + \frac{1}{2M}\sum_i\sum_j\left(\hat{O}_Y\right)_{ij}\cdot\log\left(O_Y\right)_{ij} \tag{4.5}$$

4.3 Feature-learning based Rigid Registration

This approach uses the feature-learning method: Distinctive 3D local deep descriptors (DIP) [66]. DIP uses a PointNet-based network in a Siamese configuration [19] to learn compact feature descriptors for each point trained using canonicalized patches. Each point in the patches contains its Local Reference Frame (LRF) [30]. An affine transformation is incorporated in the estimation of LRF to improve the canonicalization process by minimizing the Euclidean distance via Chamfer loss [104]. However, DIP differs with [19] in three ways:

- LRFs are used to canonicalize patches.
- Descriptors specifically encode local information, making them more robust against clutter, occlusions, and missing regions.
- Using a hardest contrastive loss allows for mining quadruplets [16], improving metric learning.

The advantage of DIP is that it directly processes points without additional features or preliminary voxelizations of the point clouds [66]. The pipeline is shown in figure 4.4. Each branch of the approach involves the following steps: (i) Extracting a patch for each point and computing the corresponding LRF based on the points within the patch; (ii) Applying a rigid transformation to the patch using the LRF and randomly sampling points from the transformed patch; (iii) Normalizing and expressing the coordinates of the sampled points relative to the patch center so that the distance between any sampled point and the patch center is equal to 1; and (iv) Using the normalized points as inputs for the network to learn the descriptors. Chamfer loss [104] is used on the output of TNet and the hardest contrastive loss [16] on the network output.

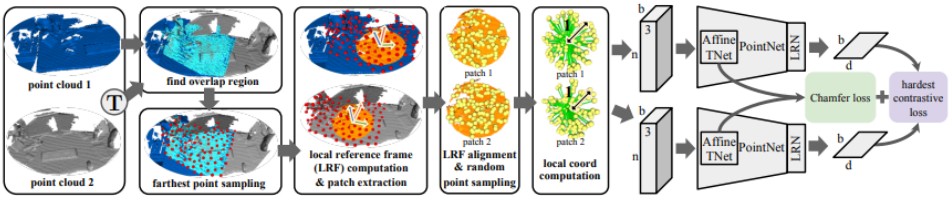


Figure 4.4: DIP Pipeline [66].

4.3.1 Patch Extraction

Firstly the ground-truth transformation is applied on the PC and TOF point clouds, and correspondences within a distance threshold of 0.03 are estimated using ICP. The value of 0.03 was selected as it resulted in the highest inlier ratio in the ICP registration process. These correspondences represent the overlap regions O and O' between the clouds. The points in the overlap region are used as possible anchor points for the Hardest-contrastive loss given in equation 4.8. Selecting anchor points is essential to minimize the loss effectively [66]. Random sampling [30, 16] is commonly used, but it can lead to spatial proximity between anchors and negative examples. To address this issue, negatives within a specific radius from an anchor can be excluded by computing the Euclidean distance between all the anchor points and determining whether to penalize this distance in the embedding space. If the points within this radius are used for training, feature for regions with similar geometric structures will be learned, leading to poor training [66]. To avoid distance computation among all the anchors, Farthest Point Sampling (FPS) [68] is utilized. FPS selects b points from O that have the largest distances among themselves. These points are then matched with their nearest neighbors in O' to construct the minibatch for the hardest-negative mining process. After experimenting with values like 256, 512, and 1024, b was set to 256 as other values did not significantly improve loss values and only resulted in longer training time.

Using FPS, a point c is selected from O and its nearest neighbor c' from O'. Sets Y and Y' are constructed from the local patch $\mathcal{X}$, consisting of points y that satisfy the condition $\|y - c\|_2 \leq \tau_r$. Although the authors used the radius value of $voxelsize \times \sqrt{3}$, a value of $0.15 \times \sqrt{3}$ is used in this work after experimenting with different thresholds. The points within each set

are used to compute its own LRF [30, 96]. "The LRF is independently constructed by determining three orthogonal axes. The z-axis is the normal of the local surface defined by the points in Y and similarly for the points in Y'. The x-axis is computed as a weighted sum of vectors obtained by projecting the vectors between c and the points in Y (excluding c) onto the plane orthogonal to the z-axis. The y-axis is obtained as the cross-product between the z-axis and the x-axis" [66]. A random sampling of n points is performed from Y and Y' to regularize training and enhance generalization. In this work, n is set to 256. The coordinates of these points are then recalculated relative to their patch center, and the radius of the encompassing sphere is normalized. This normalization is achieved by dividing each point $\hat{y}$ in $Q(Y)$ by τ_r, where $\hat{y} = (y-c)/\tau_r$ for y in Y. The resulting set is denoted as $Q(Y)$ with a size of $|Q(Y)| = n$. Finally, the LRF L is applied to $Q(Y)$, rotating the points with respect to their respective LRFs. The same operations are also performed for $Q(Y')$ [66].

4.3.2 Network Architecture

DIP is based on the PointNet architecture, with specific modifications to the Transformation Network, Bottleneck, and Local Response Normalization layer. As shown in Figure 4.5, the architecture aims to generate a d-dimensional descriptor f with a unitary length for an input patch $\mathcal{X}$.

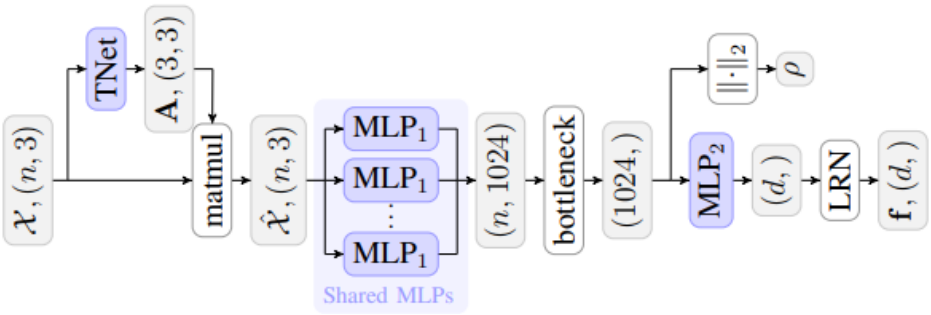


Figure 4.5: Pointnet Architecture for DIP[66].

The initial part of the architecture is the Transformation Network (TNet), responsible for predicting an affine transformation $A \in \mathbb{R}^{3\times3}$. This transformation is applied to each point x in the patch $\mathcal{X}$. To encourage the TNet to resemble an orthogonal matrix and promote a rigid transformation, a reg-

ularization term $l_{reg} = \|I - AA^T\|_F^2$ is included. The TNet architecture follows a similar structure to the PointNet network, with the final MLP layer generating nine values that are reshaped to form the matrix A. The PointNet feature extraction network consists of three 1D convolutional layers with a filter size of [256,256,512], followed by three fully connected layers with [256,256,32] filters. This work adds a dropout layer to each layer with a probability of 0.1 in the convolutional layers and 0.3 in fully connected layers. The transformation network follows the same architecture as the feature extraction network. The original work included [256,512,1024] filters in the convolutional layers and [512,256,32] in the fully connected layers. The dropout was only used in the first fully connected layer.

The Bottleneck layer is designed as a symmetric function that produces the same output regardless of the order of its input elements [66]. It performs max pooling defined as $\max : \mathbb{R}^{n\times m} \rightarrow \mathbb{R}^m$ on the output of the previous layer, resulting in a global signature $\gamma = (g_1, g_2, ..., g_m)$ for the input patch such that:

$$\gamma = \max_{\mathcal{X}}\Big(\Phi_\Theta\big(\mathcal{X}\big)\Big) \tag{4.6}$$

Here m represents the number of channels in the output from the layer preceding the bottleneck, g_i denotes the i^{th} element of γ, and Φ_{Θ_j} corresponds to an intermediate network output at the j^{th} layer. It has been observed that the global signature γ in DIP can be used to assess the amount of information provided by the network [66]. Additionally, a function $\alpha = \arg\max_{\mathcal{X}}\Big(\Phi_{\Theta_j}(\mathcal{X})\Big)$ is defined, which allows us to retrieve the indices corresponding to the γ values. In other words, α gives us the positions of the maximum values in γ, with the mapping $\arg\max : \mathbb{R}^{n\times m} \rightarrow \big[1, n\big]^m$.

The values of γ, α, γ', and α' are calculated for the patches from PC and TOF point clouds. The values of α and α' remain the same irrespective of the order of points in the patches. These values can be interpreted as corresponding points between both patches. The reliability of these correspondences can be assessed by examining the maximum values of γ and γ'. The norm of γ, denoted as $\|\gamma\|_2$, is an effective measure of reliability. It allows us to identify and potentially discard patches extracted from flat surfaces or regions with low information content. To select robust descriptors that provide reliable correspondences, a threshold condition is set as $\rho = \|\gamma\|_2 > \tau_\rho$, where τ_ρ is a threshold value [66].

Finding a single threshold value τ_ρ that works well for different input data and network architectures can be challenging [66]. To address this challenge, an alternative approach is proposed. It involves inferring τ_ρ based on the distribution of ρ values. Using the cumulative density function, τ_ρ can be determined as the p_ρ^{th} percentile. p_ρ is calculated as the area under the probability density function $f(\rho)$ up to the threshold τ_ρ, expressed as a percentage of 100. This approach ensures that τ_ρ is chosen so that the accumulated area to the left of τ_ρ in the probability density function corresponds to $p_\rho/100$. A series of MLP layers in the MLP_2 block process these global signature values. The Local Response Normalization (LRN) layer ensures that the descriptors have a unitary length. It performs L_2 normalization on the output of the final MLP layer, which has a dimensionality of d [66].

4.3.3 Loss functions

The Chamfer loss is utilized to minimize the distance between each point x in the patch $\mathcal{X}$ and its closest neighbor x' in the patch $\mathcal{X}'$. The Chamfer loss is computed as the sum of the minimum distances between points in $\mathcal{X}$ and $\mathcal{X}'$, as shown in equation 4.7 [66].

$$l_c(\mathcal{X}) = \frac{1}{2n}\Big(\sum_{x\in\mathcal{X}} \min_{x'\in\mathcal{X}'} \|Ax - A'x'\|_2 + \sum_{x'\in\mathcal{X}'} \min_{x\in\mathcal{X}} \|Ax - A'x'\|_2\Big) \tag{4.7}$$

To facilitate metric learning, the Hardest-contrastive loss with negative mining is used. It is a technique used to compute distances for matching the descriptors. For each pair of anchors (f, f'), the hardest-negatives (f_-, f'_-) are identified [66]. The loss is defined as a combination of three terms: the positive loss, which encourages a small distance between positive pairs; the hardest-negative loss, which penalizes distances to the hardest negatives; and the hardest-negative loss for the second descriptor, which penalizes distances to the hardest negatives of the second descrip-

tor. The full equation for the hardest-contrastive loss is given in Equation 4.8 [66].

$$\begin{aligned} l_h = \frac{1}{b} \sum_{(f,f') \in C_+} \Big(& \frac{1}{|C_+|} \big[d(f,f') - m_+ \big]_+^2 \\ & + \frac{1}{2|C_-|} \big[m_- - \min_{\tilde{f} \in C_-} d(f,\tilde{f}) \big]_+^2 \\ & + \frac{1}{2|C_-|} \big[m_- - \min_{\tilde{f} \in C_-} d(f',\tilde{f}) \big]_+^2 \Big) \end{aligned} \tag{4.8}$$

Here C_+ represents the set of anchor pairs, and C_- represents the set of descriptors used for negative mining within a minibatch. The values m_+ and m_- correspond to the margins for positive and negative pairs, respectively. The function $[\cdot]_+$ extracts the positive part from its argument [66].

4.4 Non-Rigid Registration

The non-rigid registration approach is inspired by C2P-net [54] which utilizes a two step pipeline. Intially rigid registration is performed by extraction features using GCNet [106] and correspondences are estimated using one step RANSAC. Then these correspondences are used to perform non-rigid registration using NDP [53].

4.4.1 Rigid Registration and Correspondence Estimation

This method, called GCNet [106], is a feature-based learning approach that leverages the proportion of correspondences to extract features from point clouds. The pipeline of GCNet is illustrated in Figure 4.6.

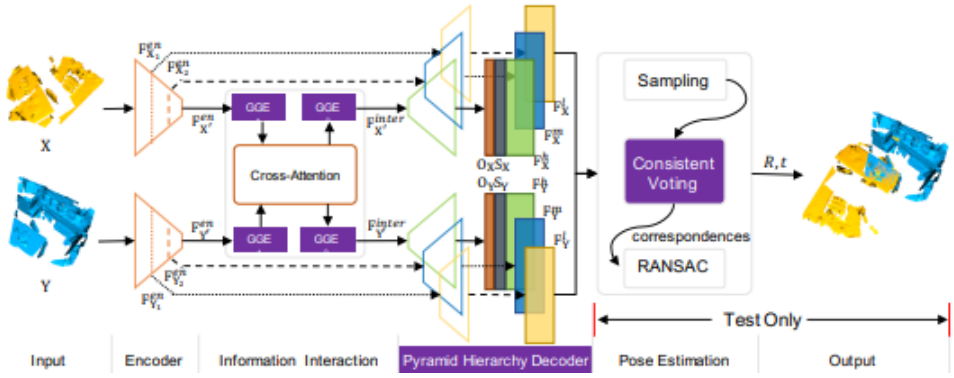


Figure 4.6: Pipeline for GCNet[106].

The input point clouds are processed through a shared encoder network based on KP-FCNN [85], which produces a feature vector for each KPConv block. The encoder module consists of L layers, resulting in multiple point clouds of different resolutions $\{X_1, ..., X_L\}$ and their corresponding feature vectors $\{F_X^{en}, ..., F_{X_L}^{en}\}$. In this thesis the encoder consists of 4 layers in each block. The first layer contains a *simple* block and a *resnet* block and the other three layers contains 2 *resnet* block each. A *simple* block consists of a KPConv block followed by batch normalization and LeakyReLU activation function. The *resnet* block consists of a KPConv block, a batch normalization block, a single layer MLP with LeakyReLU activation function, a max pooling block, another MLP and finally LeakyReLU activation function. KPConv uses a points within a specified radius to extract local features and in this thesis the radius is set to 0.06 after experimenting with several values. The features from the encoder is fed into an information interaction module. The information interaction module includes two shared geometry-guided encoding (GGE) modules and a cross-attention module. The GGE module takes as input the L^{th} downsampled point cloud from the encoder (X'), its feature vector ($F_{X'}$), and the original point cloud and generates geometry-enhanced features ($F_{X'}^{gge}$). The architecture of the GGE module is shown in Figure 4.7.

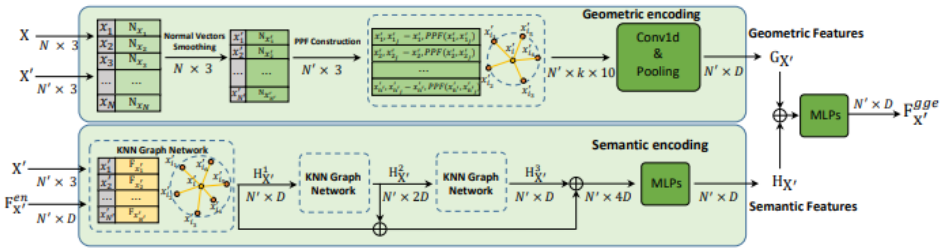


Figure 4.7: GGE Module[106].

In the GGE module, the geometry of the normals of points in X' is first corrected through normal vector smoothing, which involves averaging the normals of corresponding points in the original cloud. Then, geometric encoding based on PPF (Point Pair Features) [20] is performed, where points within a radius r^G are utilized, and a pointwise max pooling operation is applied. Using the features from the encoder and X', a dense graph is constructed to enhance the semantic features using a graph neural network. Finally, a D-dimensional feature $F_{X'}^{gge}$ is generated by fusing the geometric and semantic features through a multi-layer perceptron (MLP). The radius is set to 0.32 in this thesis. The PPF block contains three 2d convolutional layer with (128,256,128) filters and each layer uses a ReLU activation function.

The cross-attention module calculates the information interaction feature using X' and $F_{X'}^{gge}$. The features from the GGE module are transformed into query (Q), key (K), and value (V) representations using trainable weight matrices W_i^Q, W_i^K, and W_i^V, respectively. This transformation is performed independently for each head of the multi-head attention mechanism. The attention mechanism computes each head as $\text{softmax}\left(\frac{Q_i \cdot K_i^T}{\sqrt{D_i}}\right) \cdot V_i$. The heads are then concatenated and passed through an MLP to obtain cross-attention features. These features and the original features X' are used as input to the second GGE module, resulting in the final feature $F_{X'}^{inter}$. The same procedure is applied to the second point cloud. The cross-attention contains four 1d convolutional layer followed by a MLP. Each convolutional layer contains 256 filters. The MLP contains a 1d convolutional layer followed by an instance norm layer, a ReLU activation function and a 1d convolutional laeyr at the end.

The decoder generates three-level C-dimensional features by utilizing the features from the previous steps. Finally, overlap scores and saliency scores are calculated, which are used for sampling the overlapping points. A certain number of points (768 in this work) are sampled based on the scores, and correspondences are estimated using consistent voting. The voting mechanism utilizes the feature vectors to perform nearest-neighbor searches and predict correspondences. For a point in the first point cloud the mechanism finds its nearest neighbor in the second point cloud's feature space for each level. Starting from the first layer if nearest neighbors from two consecutive layers are matched, it is considered to be a correspondence. The transformation is then estimated using the RANSAC algorithm.

The overall loss function is computed as the sum of the feature losses, overlap loss, and saliency loss [106]:

$$\mathcal{L} = \mathcal{L}^h(F) + \mathcal{L}^m(F) + \mathcal{L}^l(F) + \mathcal{L}(O) + \mathcal{L}(S) \tag{4.9}$$

The feature loss is calculated using the circle loss formulation for different feature representations F_X^h, F_X^m, F_X^l, F_Y^h, F_Y^m, and F_Y^l. The loss is computed based on a randomly sampled set of correspondences $C = \{(x_i, y_{\sigma(i)}) | i = 1, 2, ..., S\}$, where S is the number of correspondences. For example, the loss $\mathcal{L}_h^X(F)$ for F_X^h is defined as [106]:

$$\begin{aligned}\mathcal{L}_X^h(F) = \frac{1}{S}\sum_{i=1}^{S} log[1 + \sum_{y_j \epsilon \mathcal{P}_i} exp\big(\gamma \alpha_{ij}^p (D_{ij}^h - \triangle p)\big) \\ \cdot \sum_{y_k \epsilon \mathcal{N}_i} exp\big(\gamma \alpha_{ik}^n (\triangle n - D_{ik}^h)\big)]\end{aligned} \tag{4.10}$$

where $\mathcal{P}_i$ represents the set of corresponding points $y_{\sigma(i)}$, N_i denotes the set of points y such that $(x, y) \epsilon C$ and $x \neq x_i, D_{ij}^h$ represent the Euclidean distance between $F_{x_i}^h$ and $F_{y_j}^h$, $\triangle p$ and $\triangle n$ are positive and negative margins, $\alpha_{ij}^p = [D_{ij}^h - \triangle p)]_+$, $\alpha_{ik}^n = \triangle n - [D_{ik}^h]_+$, and γ is a scale factor. Similar formulations are used for $\mathcal{L}_Y^h(F)$, $\mathcal{L}^m(F)$, and $\mathcal{L}^l(F)$ [106].

The overlap and saliency scores O_X and S_X are treated as binary classification tasks, and binary cross-entropy loss is used. The losses for overlap and saliency are defined as [106]:

$$\begin{aligned}\mathcal{L}_X(O) &= -\sum_{x_i \epsilon X} w_i^O [\overline{O}_{x_i} log O_{x_i} + (1 - \overline{O}_{x_i}) log(1 - O_{x_i})] \\ \mathcal{L}_X(S) &= -\sum_{x_i \epsilon X} w_i^S [\overline{S}_{x_i} log S_{x_i} + (1 - \overline{S}_{x_i}) log(1 - S_{x_i})]\end{aligned} \tag{4.11}$$

where w_i^O and w_i^S are weighting factors for category balance, $\overline{O}_{x_i}$ and $\overline{S}_{x_i}$ are the ground truth binary overlap and saliency scores for point x_i, respectively. $\overline{O}_{x_i}$ is set to 1 if the distance between the transformed point and its nearest neighbor is below a threshold. $\overline{S}_{x_i}$ is set to 1 if the ground truth candidate point is matched based on feature matching. $\mathcal{L}Y(O)$ and $\mathcal{L}Y(S)$ are defined similarly [106].

4.4.2 Neural Deformation Pyramid

After rigid registration, non-rigid refinement is performed using the Neural Deformation Pyramid (NDP) [53] method. The NDP utilizes a pyramid structure where each level has a MLP that operates on sinusoidally encoded points. The architecture of the NDP is depicted in Figure 4.8.

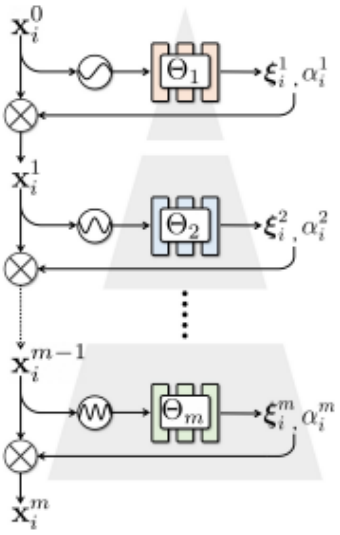


Figure 4.8: NDP Architecture [53].

The NDP decomposes the estimation parameter ξ into a sequence of sub-transformations $\{\xi_i^1, \xi_i^2, ..., \xi_i^m\}$. These sub-transformations are easy to estimate and can be combined to obtain the final transformation. At each level of the pyramid, the NDP employs a continuous function Γ and an MLP Θ. The function Γ performs positional encoding by mapping the inputs from the previous level in 3D space to a 6D sinusoidal encoding. It is given by:

$$\big(\sin(2^{k+k_0} x_i^{k-1}), \cos(2^{k+k_0} x_i^{k-1})\big) \tag{4.12}$$

where k represents the pyramid level and k_0 controls the frequency of the sinusoidal encoding. A hierarchical rigid-to-nonrigid motion decomposition is achieved by gradually increasing the frequency with each level. Lower frequencies correspond to relatively rigid motion, while higher frequencies represent highly non-rigid motion. The MLP Θ takes the encoded coordinates as input and estimates the transformation increments for the current pyramid level. The output of the MLP represents the confidence in the accuracy of the motion estimates at that level. Using the estimated transformation parameters ξ_i^k and the confidence values α_i^k, the transformed coordinates at the k^{th} level are computed by applying a warp function W to the previous level's coordinates x_i^{k-1}. The degree of deviation from the previous level is controlled by α_i^k, which determines the level-wise deformability. In this thesis three level pyramid is used and each layer contains a MLP with a linear layer of 128 filters with ReLU activation function. k_0 is set to -8

Regularization terms are applied to the confidence values α_i^k to promote as-rigid-as-possible movement. These regularization terms help maintain the rigidity of the motion during the registration process. The deformability regularization term E_{reg}^k is formulated as a negative log-likelihood computed over the set S^k points in the point cloud. The term aims to minimize the negative log-likelihood of the confidence scores not being zero, encouraging a deformability score of zero and preserving the point cloud's original geometry.

The NDP uses Chamfer and Correspondence loss as cost functions at each pyramid level. The correspondence loss measures the discrepancy between the matched points in the point clouds. Finally, the total loss E_{total}^k at each level is computed as a weighted combination of the Chamfer loss E_{cd}^K, Correspondence loss E_{cor}^K, and deformability regularization loss E_{reg}^K. The correspondence loss is given as:

$$E_{corr}^k = \frac{1}{|M|} \sum_{(u,v)\in M} \rho\left(x_u^k - y_v\right) \tag{4.13}$$

Here M is the set of correspondences and (u, v) are the matched indices. Finally total loss is computed as:

$$E_{total}^k = \lambda_{cd} E_{cd}^K + \lambda_{cor} E_{cor}^K + \lambda_{reg} E_{reg}^K \tag{4.14}$$

where λ_{cd},λ_{corr} and λ_{reg} are the weighting factors.

5

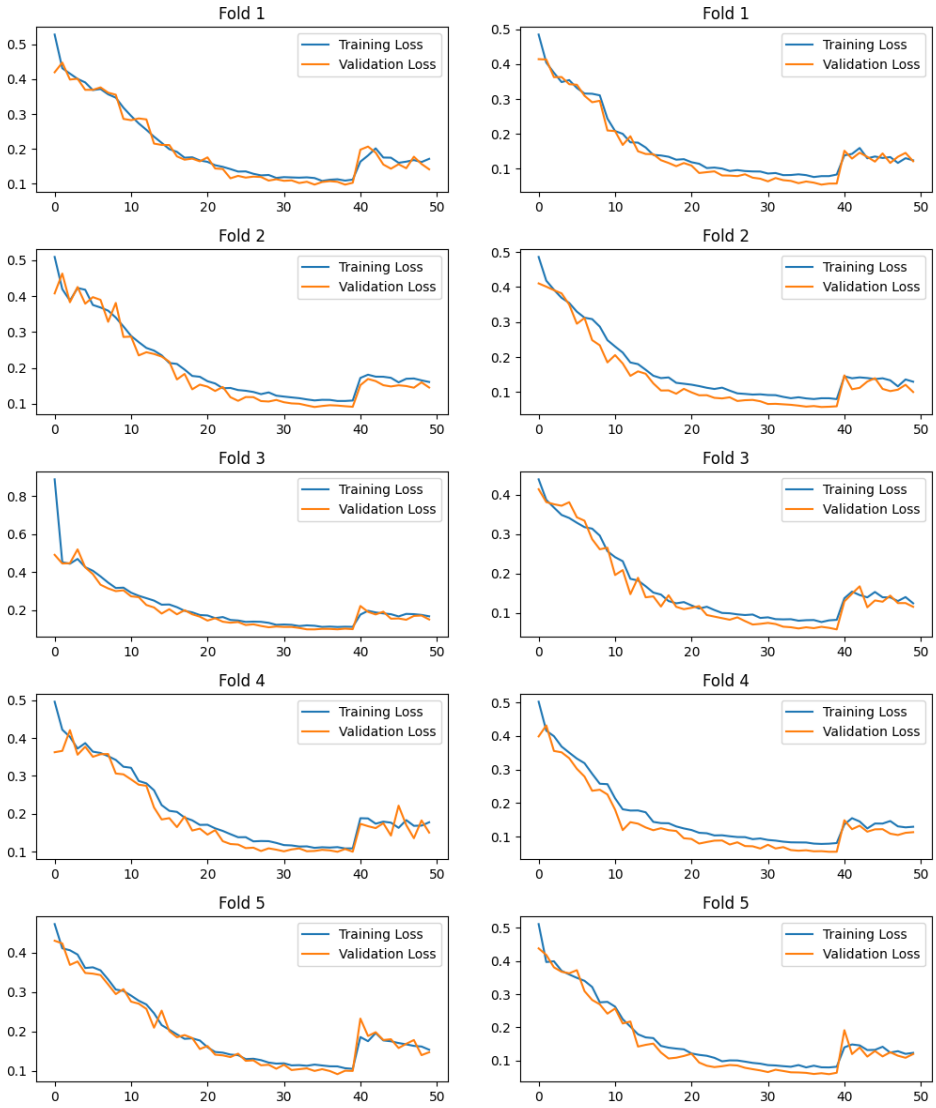
Experiments and Evaluation

All the experiment methods were implemented in PyTorch 2.0.0, RANSAC, and ICP from Open3d and NVIDIA GeForce with 11 Gb memory used for computation.

5.1 ROPNet

After experimenting with different learning rate values, the model is trained using five-fold cross-validation due to a lack of data with an initial learning rate of 0.0001 and Adam [46] optimizer for 50 epochs. Figure 5.1 shows each fold's training and validation loss. The average of best training loss across all folds is 0.108 (original) and 0.078 (cropped), while the average of best validation loss is 0.095 (original) and 0.056 (cropped). The best model across all folds is selected to be evaluated on the test set, and the average loss is 0.113 for the original and 0.091 for the cropped version of the data. The result of the original and cropped versions is similar because the network uses overlapping points for training.

Other metrics like Chamfer distance, mean square error, Hausdorff distance, and mean absolute error before and after the registration are used to evaluate the registration quality. This value is also compared with traditional CPD as a baseline, as shown in table 5.1 and table 5.2. Since the ground truth transformation is obtained using RANSAC, it is not a baseline. From the comparison, it can be inferred that ROPNet registration quality is similar to CPD. Still, ROPNet takes an average of 1.03 seconds to register a sample, whereas CPD takes 12.40 seconds.



(a) Original (b) Cropped

Figure 5.1: ROPNet Cross Validation Results

	Before	ROPNet	Rigid CPD
Mean Correspondences	61	**2045**	2000
Mean Square Error	0.472	**0.019**	0.031
Mean Absolute Error	0.585	**0.092**	0.143
Mean Hausdorff Distance	0.605	**0.031**	0.062
Mean Chamfer Distance	1.003	0.102	**0.098**

Table 5.1: ROPNet Registration Result (Original)

	Before	ROPNet	Rigid CPD
Mean Correspondences	31	2046	**2048**
Mean Square Error	0.352	0.001	0.001
Mean Absolute Error	0.495	0.027	**0.022**
Mean Hausdorff Distance	0.633	**0.026**	0.062
Mean Chamfer Distance	0.930	**0.036**	0.038

Table 5.2: ROPNet Registration Result (Cropped)

A sample before and after registration of original and cropped clouds is shown below.

(a) Before (b) After

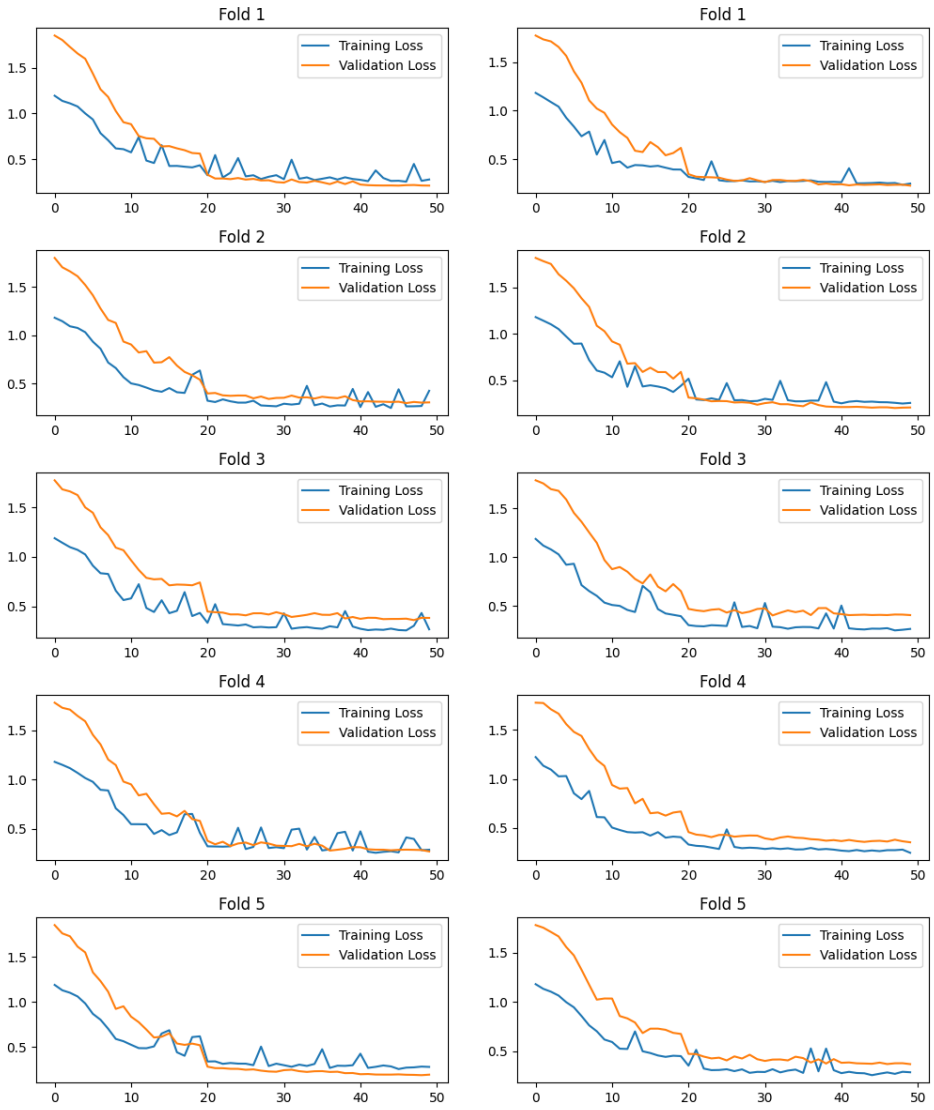
Figure 5.2: ROPNet Registration Result (Original)

(a) Before (b) After

Figure 5.3: ROPNet Registration Result (Cropped)

5.2 DIP

During training, a 5-fold cross-validation strategy is employed, utilizing the Stochastic Gradient Descent (SGD) optimizer [70] with a learning rate of 0.009, momentum of 0.9, and weight decay of 0.01. The scheduler is Cosine Annealing with Warm Restarts [56]. The total number of epochs for training is set to 50.



(a) Original (b) Cropped

Figure 5.4: DIP Cross Validation Results

Figure 5.4 illustrates each fold's training and validation loss. The average of best training loss across all folds is 0.253 (original) and 0.246 (cropped), while validation loss is 0.264 (original) and 0.310 (cropped). Here is little difference between the original and cropped version's result because the network uses the correspondences for patch extraction and training. The best model across all folds is selected to extract the descriptors from patches of the test set. After obtaining descriptors for each point, RANSAC

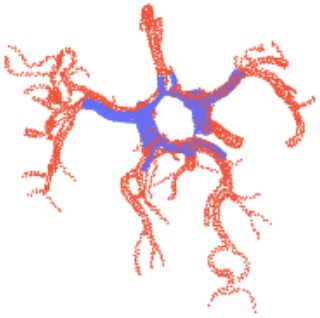
	Before	**DIP**	**FPFH**
Mean Correspondences	206	**2864**	2282
Mean Square Error	1.566	**0.019**	0.022
Mean Absolute Error	1.038	**0.092**	0.106
Mean Hausdorff Distance	1.036	**0.023**	0.080
Mean Chamfer Distance	1.875	**0.100**	0.126

Table 5.3: DIP Registration Result (Original)

	Before	DIP	FPFH
Mean Correspondences	254	**2845**	2742
Mean Square Error	1.696	**0.001**	0.002
Mean Absolute Error	1.028	**0.027**	0.032
Mean Hausdorff Distance	1.154	**0.023**	0.040
Mean Chamfer Distance	1.988	**0.033**	0.043

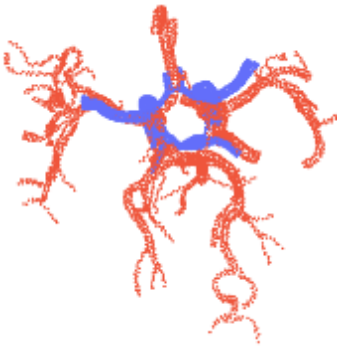
Table 5.4: DIP Registration Result (Cropped)

is used to perform registration. Ideally, if the descriptors are good enough, only 1 iteration of RANSAC should register the point clouds successfully. To compare the quality of DIP with FPFH, commonly used with RANSAC, registration is performed for different numbers of iterations for a single sample as shown in figure 5.5. The figure shows that using DIP descriptors, RANSAC can register the point clouds in 1 iteration, while FPFH takes 50000 iterations for perfect alignment. Therefore, DIP is better than FPFH as a feature for point clouds. The below tables show the registration result on the test set of original and cropped point clouds for 50000 iterations. DIP outperforms FPFH in both cases.

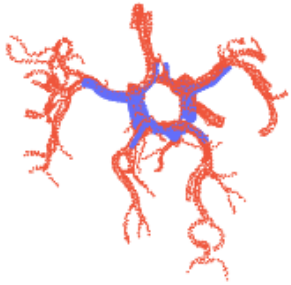


(a) DIP(Iter: 1).

(b) FPFH(Iter: 1).



(c) FPFH(Iter: 10000).

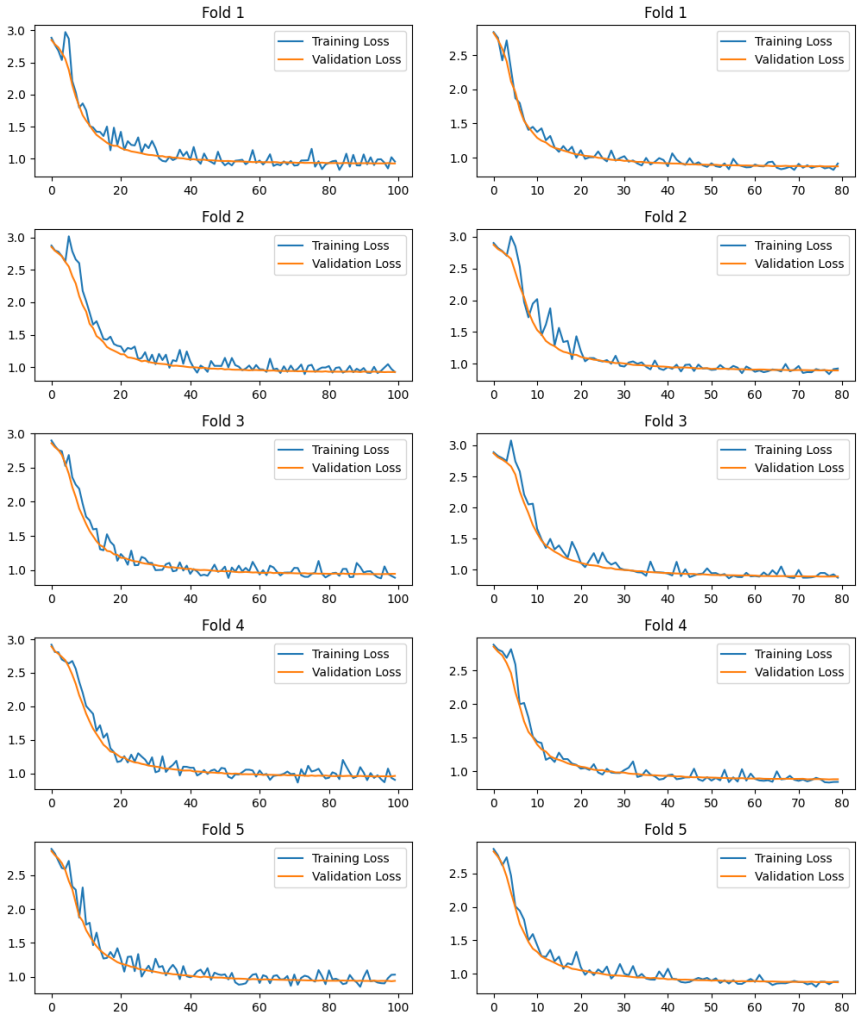


(d) FPFH(Iter: 50000).

Figure 5.5: DIP vs FPFH Registration using RANSAC

5.3 Non Rigid Registration

GCNet is trained using 5-fold cross-validation for 100 epochs per fold to perform rigid registration. After experimenting with various values, the initial learning rate is 0.05 with SGD optimizer and exponential learning rate scheduler. Figure 5.6 shows the training and validation losses for the original and cropped data, respectively. The average of best training loss across all folds is 0.863 (original) and 0.822 (cropped), while the validation loss is 0.935 (original) and 0.870 (cropped). RANSAC was used to register



(a) Original (b) Cropped

Figure 5.6: GCNet Cross Validation Result.

after extracting features for individual points using GCNet. However, as shown in Figure 5.7, the registration results were inconsistent when using a single RANSAC iteration. Therefore, more iterations were used to improve the registration results.

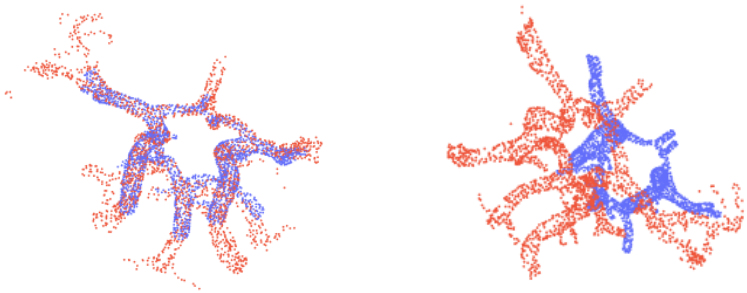


Figure 5.7: RANSAC Result using GCNet (Cropped, Iter: 1).

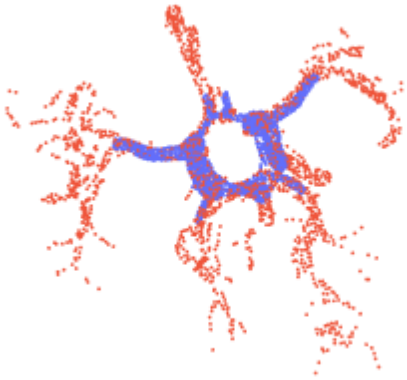


Figure 5.8: RANSAC Result using GCNet (Original, Iter: 1).

Table 5.5 and 5.6 shows the registration result for 10000 iterations of RANSAC on the test set. The search radius of 0.03 is used for correspondence estimation as this value gave the best ICP registration result in the data preparation step. Rigid CPD was used as a baseline for comparison, and experimental results show that GCNet was marginally better in the case of cropped data. When using the original data, rigid CPD cannot perform registration properly, as shown in figure 5.9. A comparison

	Before	GCNet	Rigid CPD
Mean Correspondences	69	**1484**	717
Mean Square Error	0.065	7.66e-05	**7.52e-05**
Mean Absolute Error	0.039	**0.0010**	0.0012
Mean Hausdorff Distance	0.044	**0.0009**	0.004
Mean Chamfer Distance	0.0767	**0.0013**	0.0022

Table 5.5: GCNet Registration Result (Cropped)

	Before	GCNet	Rigid CPD
Mean Correspondences	40	895	**1093**
Mean Square Error	0.058	**0.0007**	0.009
Mean Absolute Error	0.038	**0.003**	0.073
Mean Hausdorff Distance	0.038	**0.001**	0.129
Mean Chamfer Distance	0.069	**0.003**	0.123

Table 5.6: GCNet Registration Result (Original)

with FPFH must not be made, as DIP was shown to perform better in the previous section, and GCNet will be compared to DIP in the next section. After performing the rigid registration, correspondences are estimated and used to refine the registration using NDP. The tables 5.7 and 5.8 below show the registration results of NDP and GCNet. As can be seen, the MSE of GCNet is already very close to 0, indicating that the registration is nearly perfectly aligned. Therefore, there is no room for improvement using NDP. The NDP registration result of a sample is shown in figure 5.10.

	GCNet	NDP	Non Rigid CPD
Mean Correspondences	**1484**	1417	894
Mean Square Error	7.66e-05	**7.08e-05**	0.018
Mean Absolute Error	0.0010	0.0010	0.114
Mean Hausdorff Distance	**0.0009**	0.0011	0.073
Mean Chamfer Distance	**0.0013**	0.0014	0.149

Table 5.7: NDP Registration Result (Cropped).

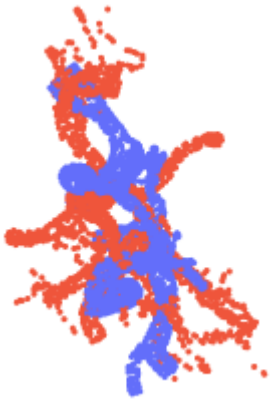
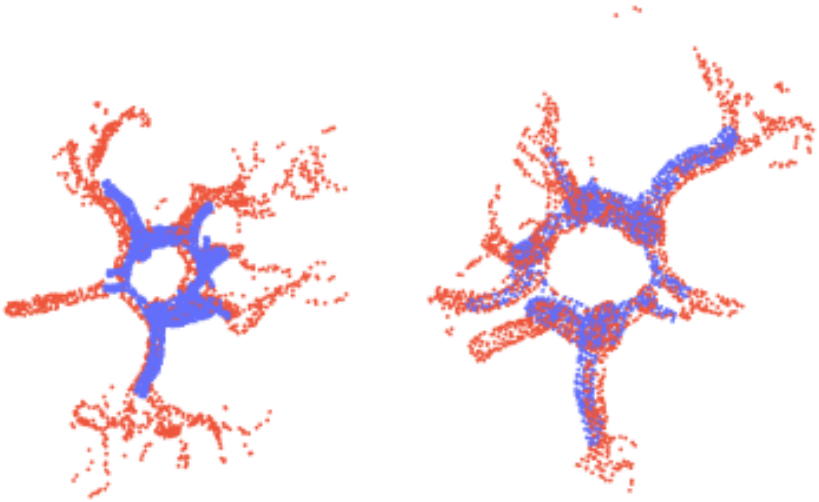


Figure 5.9: Registration Result of Rigid CPD.

	GCNet	NDP	Non Rigid CPD
Mean Correspondences	**895**	293	848
Mean Square Error	0.0007	**0.0005**	0.045
Mean Absolute Error	0.003	0.003	0.177
Mean Hausdorff Distance	**0.001**	0.002	0.088
Mean Chamfer Distance	**0.003**	0.004	0.214

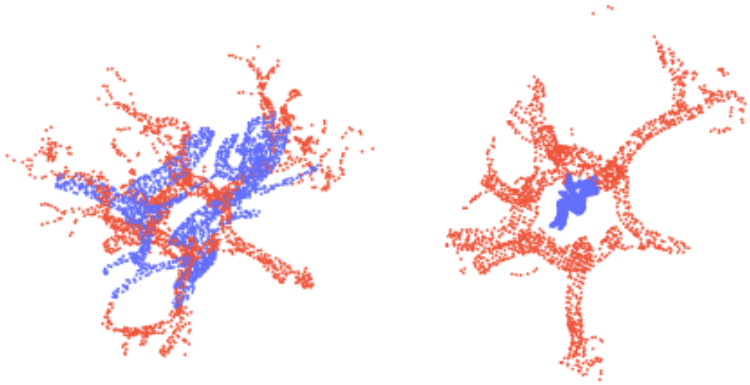
Table 5.8: NDP Registration Result (Original).



(a) Original (b) Cropped

Figure 5.10: NDP Result.

The registration result of nonrigid CPD is poor. After visualizing the results, it was inferred that the model is squeezing the PC point cloud along one dimension, converting it to almost 2D data. The result is shown in the figure.



(a) Original (b) Cropped

Figure 5.11: Non-Rigid CPD Result.

5.4 Comparison and Additional Results

The table below displays the registration results obtained using various methods in this thesis. All three methods demonstrate similar performance. The estimated correspondences are less in GCNet when compared to other methods. Still, the evaluation metrics show better results, as seen in table 5.9 and 5.10. ROPNet has advantages over DIP and GCNet in certain aspects, as it does not rely on RANSAC for transformation estimation and can be trained end-to-end. However, it requires the voxelization of point clouds. In this work, the voxel size of the TOF and PC point cloud is the same after the data preparation step, but it is not the case in reality, and in such cases, ROPNet might fail. On the other hand, DIP benefits from not requiring the voxelization of point clouds, with the only parameter to be adjusted is the radius for estimating LRF. Also, DIP uses patches of size 256 from point clouds to extract descriptors and estimate transformation, allowing point clouds with many points. GCNet's high-quality results rely on multiple radius values used throughout the algorithm, which need

thorough tuning for effective learning. For instance, the overlap radius in the preprocessing step determines the correspondences between source and target point clouds, with larger values leading to false matches and smaller values resulting in fewer correspondences. The deform radius in each network layer establishes neighborhood limits to remove unwanted points. The convolutional radius in the encoder blocks extracts features for individual points by only considering neighbors within the mentioned radius. Adjusting these values affects the network's ability to learn local or global features and impacts computational efficiency. Lastly, another radius serves as a distance threshold to identify point matches.

	ROPNet	DIP	GCNet	NDP
Mean Correspondences	2045	**2864**	895	293
Mean Square Error	0.019	0.019	**0.0007**	0.0005
Mean Absolute Error	0.092	0.092	**0.003**	0.003
Mean Hausdorff Distance	0.031	0.023	**0.001**	0.002
Mean Chamfer Distance	0.102	0.100	**0.003**	0.004

Table 5.9: All Registration Result (Original).

	ROPNet	DIP	GCNet	NDP
Mean Correspondences	2046	**2845**	1484	1417
Mean Square Error	0.001	0.001	7.66e-05	**7.08e-05**
Mean Absolute Error	0.027	0.027	0.001	0.001
Mean Hausdorff Distance	0.026	0.023	**0.0009**	0.0011
Mean Chamfer Distance	0.036	0.033	**0.0013**	0.0014

Table 5.10: All Registration Result (Cropped).

Learning3D [79] is a Python library that provides various classification, segmentation, and registration algorithms for point clouds, and it was used to test DCP and RPMNet. The experiments were done to estimate their performance, and no extensive tuning was done to adapt these algorithms to work on TOF and PC point clouds. The training and test loss for DCP is given in figure 5.12. In this experiment, the network size is three times the network provided in Learning3D. The registration result is not provided as the network did not learn.

Figure 5.12: DCP Loss.

Next RPMNet was experimented with, and the model performed poorly, as seen in figure 5.13.

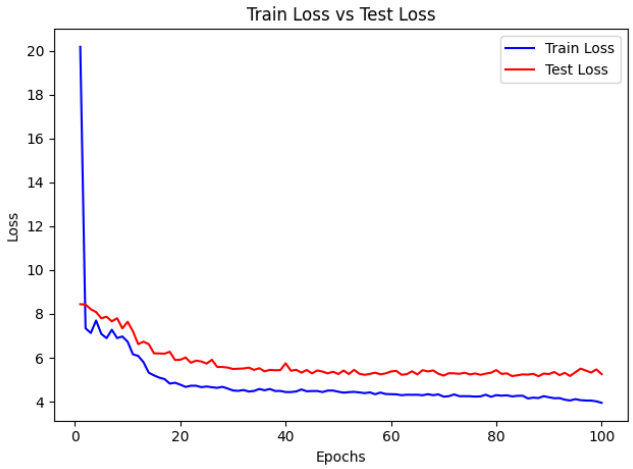


Figure 5.13: RPMNet Loss.

Another method experimented with was FMR [40], as the authors claimed it works for multimodal data. The loss is shown in figure 5.14. Even though the loss is low during training, the model cannot register the point clouds properly, as shown in figure 5.15.

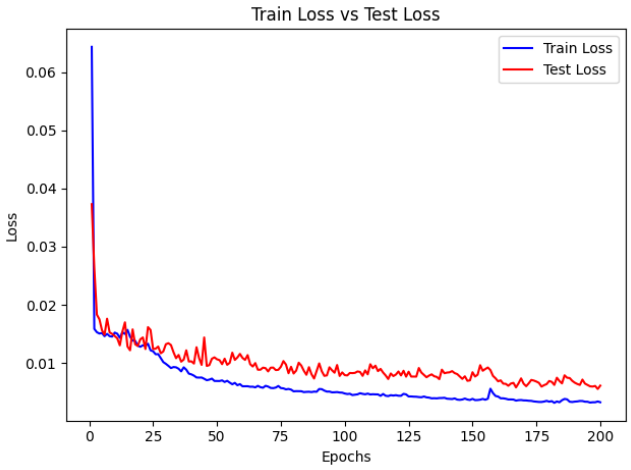
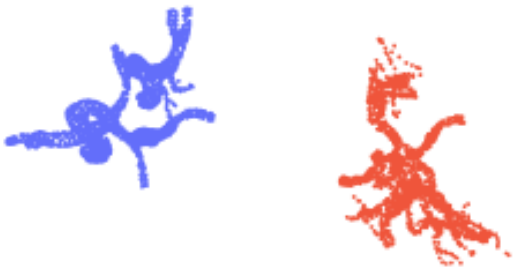


Figure 5.14: FMR Loss.



(a) Before

(b) After

Figure 5.15: FMR Result.

6

Conclusion and Future Work

This thesis presented an overview of deep learning-based point cloud registration and its use in registering 3D TOF and PC MRI data. MRI has shown promising results in detecting CSVD, and registration of the two modalities can help notice vascular anomalies in the brain. This work aims to research state-of-art methods available for point cloud registration and evaluate them on the MRI data. The first approach in the thesis performs rigid registration using ROPNet, which only considers the overlapping region for learning. The second method used is a feature-learning-based method based on DIP, which provides local descriptors of each point that can be used to perform registration using RANSAC. The third method is inspired by C2PNet and uses GCNet for performing rigid registration and then using NDP to perform non-rigid registration. NDP can also be combined with any other method that results in correspondence estimation. The motivation behind using GCNet is to compare another feature-learning-based method with DIP. However, using GCNet provided good registration results and left no room for improvement for NDP to improve results using non-rigid registration. Many registration methods are available, but most of them have yet to be tested on multimodal data, including those used in this thesis.

There are a few challenges and limitations in this thesis. The first is the need for more computational resources. The original TOF point cloud contains 122935 points with a voxel size of 0.3 mm, and the PC point cloud has 23985 points with a voxel size of 0.6 mm. It is not feasible to process all the points for registration; therefore, sampling strategies were used to reduce the number of points. The original TOF and PC point clouds were of different voxel sizes, but all the methods applied in this work use the same voxel size for feature extraction. Only DIP does not use the voxel size of the

point clouds, but it uses the same radius value for both the point clouds to estimate LRF. Therefore, the points clouds were voxel downsampled with size 0.01 for efficient feature matching, another limitation of this work. An extension of this work could be evaluating the methods with the original voxel size. A limitation of this work is not estimating the metric values like Chamfer and Hausdorff distance in mm. The works [81, 83] are related to registration of TOF and PC MRI and the Hausdorff distance is provided in mm and the results of this thesis could not be compared due to the limitation. Another challenge of multimodal registration is the presence of outliers and partially overlapping regions [41]. In the TOF point cloud, many vessels and regions are not present in the PC point cloud. These extra points are the outliers that could hinder the registration process. The implemented methods were prone to outliers and partial overlap as they only used overlapping points or correspondences to estimate the transformation. This is proven by the fact that the registration result of these methods on cropped and original data is similar.

One more probable extension of this work could be using DGCNN instead of PointNet as the encoder of the ROPNet architecture. DGCNN is known to capture more local geometric information than PointNet [87] and might improve the feature learning process. Most of the point cloud registration use a PointNet based encoder to extract features. MeshCNN [34] has better mesh segmentation results than PointNet and could be used as an encoder for point cloud registration

In conclusion, the implemented methods worked well on scaled and normalized TOF and PC point clouds if the loss of information is ignored. Rigid registration showed promising results, but due to a lack of ground truth deformation data, the quality of non-rigid registration could not be evaluated effectively. If more computational resources are available, these methods could be tried on unscaled data having differences between the voxel sizes of TOF and PC point clouds.

A

Bibliography

[1] [AOKI et al. 2019] Y. Aoki, H. Goforth, R. A. Srivatsan und S. Lucey. **PointNetLK: Robust & Efficient Point Cloud Registration using PointNet**. 2019.

[2] [ARUN et al. 1987] K. S. Arun, T. S. Huang und S. D. Blostein. **Least-Squares Fitting of Two 3-D Point Sets**. IEEE Trans. Pattern Anal. Mach. Intell., Vol. 9(5):698–700, 1987.

[3] [ARUN et al. 1987] K. S. Arun, T. S. Huang und S. D. Blostein. **Least-Squares Fitting of Two 3-D Point Sets**. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-9:698–700, 1987.

[4] [BERGEVIN et al. 1996] R. Bergevin, M. Soucy, H. Qagnon und D. Laurendeau. **Towards a general multi-view registration technique**. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18:540–547, 1996.

[5] [BESL und MCKAY 1992] P. J. Besl und N. D. McKay. **A Method for Registration of 3-D Shapes**. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14:239–256, 1992.

[6] [BILLINGS et al. 2015] S. D. Billings, E. M. Boctor und R. H. Taylor. **Iterative Most-Likely Point Registration (IMLP): A Robust Algorithm for Computing Optimal Shape Alignment**. PLoS ONE, Vol. 10, 2015.

[7] [BOCK et al. 2007] J. Bock, B. W. Kreher, J. Hennig und M. Markl. **Optimized pre-processing of time-resolved 2 D and 3 D Phase Contrast MRI data**. 2007.

[8] [BRIGHTMAN et al. 2023] N. Brightman, L. Fan und Y. Zhao. **Point cloud registration: a mini-review of current state, challenging issues and future directions**. AIMS Geosciences, Vol. 9(1):68–85, 2023.

[9] [BROWN und RUSINKIEWICZ 2007] B. J. Brown und S. Rusinkiewicz. **Global non-rigid alignment of 3-D scans**. pp. 21–es, 2007.

[10] [CANNISTRARO et al. 2019] R. J. Cannistraro, M. Badi, B. H. Eidelman, D. W. Dickson, E. H. Middlebrooks und J. F. Meschia. **CNS small vessel disease: a clinical review**. Neurology, Vol. 92(24):1146–1156, 2019.

[11] [CHANG et al. 2020] S. Chang, C. Ahn, M. Lee und S. Oh. **Graph-matching-based correspondence search for nonrigid point cloud registration**. Computer vision and image understanding, Vol. 192:102899, 2020.

[12] [CHEN et al. 2019] C. Y. Chen, C. W. Li, H. K. F. Mak, M. F. Lin und W. P. Chan. **Combined native magnetic resonance angiography, flow-quantifying, and perfusion-imaging for impending second-stroke assessment**. Quantitative Imaging in Medicine and Surgery, Vol. 9:521, 2019.

[13] [CHEN und MEDIONI 1991] Y. Chen und G. Medioni. **Object modeling by registration of multiple range images**. Proceedings - IEEE International Conference on Robotics and Automation, Vol. 3:2724–2729, 1991.

[14] [CHOJDAK-ŁUKASIEWICZ et al. 2021] J. Chojdak-Łukasiewicz, E. Dziadkowiak, A. Zimny und B. Paradowski. **Cerebral small vessel disease: A review**. Advances in clinical and experimental medicine : official organ Wroclaw Medical University, Vol. 30:349–356, 2021.

[15] [CHOY et al. 2020] C. Choy, W. Dong und V. Koltun. **Deep Global Registration**. 2020.

[16] [CHOY et al. 2019] C. Choy, J. Park und V. Koltun. **Fully Convolutional Geometric Features**. 2019.

[17] [CHUI und RANGARAJAN 2003] H. Chui und A. Rangarajan. **A new point matching algorithm for non-rigid registration**. Computer Vision and Image Understanding, Vol. 89(2-3):114–141, 2003.

[18] [DENG et al. 2022] B. Deng, Y. Yao, R. M. Dyke und J. Zhang. **A Survey of Non-Rigid 3D Registration**. Number 2 STAR-State of The Art Report, Vol. 41, 2022.

[19] [DENG et al. 2018] H. Deng, T. Birdal und S. Ilic. **Ppfnet: Global context aware local features for robust 3d point matching**. pp. 195–205, 2018.

[20] [DROST et al. 2010] B. Drost, M. Ulrich, N. Navab und S. Ilic. **Model Globally, Match Locally: Efficient and Robust 3D Object Recognition**. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 998–1005, 2010.

[21] [EGGERT et al. 1997] D. W. Eggert, A. Lorusso und R. B. Fisher. **Estimating 3-D rigid body transformations: a comparison of four major algorithms**. Machine Vision and Applications, Vol. 9:272–290, 1997.

[22] [ELBAZ et al. 2017] G. Elbaz, T. Avraham und A. Fischer. **3D point cloud registration for localization using a deep neural network auto-encoder**. pp. 4631–4640, 2017.

[23] [FANG et al. 2015] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu und E. Wong. **3d deep shape descriptor**. pp. 2319–2328, 2015.

[24] [FISCHLER und BOLLES 1981] M. A. Fischler und R. C. Bolles. **Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography**. Commun. ACM, Vol. 24:381–395, 1981.

[25] [FITZGIBBON 2002] A. Fitzgibbon. **Robust Registration of 2D and 3D Point Sets**. Image and Vision Computing, Vol. 21:1145–1153, 2002.

[26] [FLEETWOOD und STEINBERG 2002] I. G. Fleetwood und G. K. Steinberg. **Arteriovenous malformations**. The Lancet, Vol. 359(9309):863–873, 2002.

[27] [FORSBERG et al. 2018] K. M. Forsberg, Y. Zhang, J. Reiners, M. Ander, A. Niedermayer, L. Fang, H. Neugebauer, J. Kassubek, I. Katona, J. Weis et al. **Endothelial damage, vascular bagging and remodeling of the microvascular bed in human microangiopathy with deep white matter lesions**. Acta neuropathologica communications, Vol. 6:128, 2018.

[28] [FREEZE et al. 2018] W. M. Freeze, H. I. Jacobs, F. H. Schreuder, R. J. V. Oostenbrugge, W. H. Backes, F. R. Verhey und C. J. Klijn. **Blood-brain barrier dysfunction in small vessel disease related intracerebral hemorrhage**. Frontiers in Neurology, Vol. 9:926, 2018.

[29] [GAIDZIK et al. 2021] F. Gaidzik, S. Pathiraja, S. Saalfeld, D. Stucht, O. Speck, D. Thévenin und G. Janiga. **Hemodynamic data assimilation in a subject-specific circle of willis geometry**. Clinical Neuroradiology, Vol. 31:643–651, 2021.

[30] [GOJCIC et al. 2019] Z. Gojcic, C. Zhou, J. D. Wegner und A. Wieser. **The perfect match: 3d point cloud matching with smoothed densities**. pp. 5545–5554, 2019.

[31] [GOWER 1975] J. C. Gower. **Generalized procrustes analysis**. Psychometrika, Vol. 40:33–51, 1975.

[32] [GROSS et al. 2019] J. Groß, A. Ošep und B. Leibe. **Alignnet-3d: Fast point cloud registration of partially observed objects**. pp. 623–632, 2019.

[33] [HAINSWORTH et al. 2015] A. H. Hainsworth, A. T. Oommen und L. R. Bridges. **Endothelial Cells and Human Cerebral Small Vessel Disease**. Brain Pathology, Vol. 25:44, 2015.

[34] [HANOCKA et al. 2019] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman und D. Cohen-Or. **MeshCNN**. ACM Transactions on Graphics, Vol. 38(4):1–12, 2019.

[35] [HÄNSCH et al. 2014] R. Hänsch, T. Weber und O. Hellwich. **Comparison of 3D interest point detectors and descriptors for point cloud fusion**. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 2(3):57, 2014.

[36] [HARTLEY und ZISSERMAN 2003] R. Hartley und A. Zisserman. **Multiple view geometry in computer vision**. Cambridge university press, 2003.

[37] [HORN et al. 1988] B. Horn, H. Hilden und S. Negahdaripour. **Closed-Form Solution of Absolute Orientation using Orthonormal Matrices**. Journal of the Optical Society of America A, Vol. 5:1127–1135, 1988.

[38] [HORN 1987] B. K. P. Horn. **Closed-form solution of absolute orientation using unit quaternions**. J. Opt. Soc. Am. A, Vol. 4(4):629–642, 1987.

[39] [HUANG et al. 2021] S. Huang, Z. Gojcic, M. Usvyatsov und K. S. Andreas Wieser. **PREDATOR: Registration of 3D Point Clouds with Low Overlap**. 2021.

[40] [HUANG et al.] **Feature-metric Registration: A Fast Semi-supervised Approach for Robust Point Cloud Registration without Correspondences**.

[41] [HUANG et al. 2023] X. Huang, G. Mei und J. Zhang. **Cross-source point cloud registration: Challenges, progress and prospects**. Neurocomputing, p. 126383, 2023.

[42] [HUANG et al. 2021] X. Huang, G. Mei, J. Zhang und R. Abbas. **A comprehensive survey on point cloud registration**. arXiv preprint arXiv:2103.02690, 2021.

[43] [IKAWA et al. 1994] F. Ikawa, M. Sumida, T. Uozumi, S. Kuwabara, K. Kiya, K. Kurisu, K. Arita und H. Satoh. **Comparison of three-dimensional phase-contrast magnetic resonance angiography with three-dimensional time-of-flight magnetic resonance angiography in cerebral aneurysms**. Surgical Neurology, Vol. 42:287–292, 1994.

[44] [JANG et al. 2016] E. Jang, S. Gu und B. Poole. **Categorical reparameterization with gumbel-softmax**. arXiv preprint arXiv:1611.01144, 2016.

[45] [KERAS] **Keras implementation of PointNet**. `https://keras.io/examples/vision/pointnet/`. Accessed on [03-15-2023].

[46] [KINGMA und BA 2014] D. P. Kingma und J. Ba. **Adam: A method for stochastic optimization**. arXiv preprint arXiv:1412.6980, 2014.

[47] [KOGUCIUK 2017] D. Koguciuk. **Parallel RANSAC for Point Cloud Registration**. Foundations of Computing and Decision Sciences, Vol. 42(3):203–217, 2017.

[48] [KUO et al. 2019] A. H. Kuo, P. Nagpal, B. B. Ghoshhajra und S. S. Hedgire. **Vascular magnetic resonance angiography techniques**. Cardiovascular Diagnosis and Therapy, Vol. 9:S28, 2019.

[49] [LEVINE et al. 2012] J. A. Levine, R. R. Paulsen und Y. Zhang. **Mesh Processing in Medical-Image Analysis—a Tutorial**. IEEE Computer Graphics and Applications, Vol. 32(5):22–28, 2012.

[50] [LI et al. 2008] H. Li, R. W. Sumner und M. Pauly. **Global correspondence optimization for non-rigid registration of depth scans**. Vol. 27(5):1421–1430, 2008.

[51] [LI et al. 2021] L. Li, R. Wang und X. Zhang. **A tutorial review on point cloud registrations: principle, classification, comparison, and technology challenges**. Mathematical Problems in Engineering, Vol. 2021:1–32, 2021.

[52] [LI et al. 2018] Y. Li, R. Bu, M. Sun, W. Wu, X. Di und B. Chen. **PointCNN: Convolution On $\mathcal{X}$-Transformed Points**. 2018.

[53] [LI und HARADA 2022] Y. Li und T. Harada. **Non-rigid Point Cloud Registration with Neural Deformation Pyramid**. 2022.

[54] [LIU et al. 2023] P. Liu, J. Golde, J. Morgenstern, S. Bodenstedt, C. Li, Y. Hu, Z. Chen, E. Koch, M. Neudert und S. Speidel. **Non-rigid point cloud registration for middle ear diagnostics with endoscopic optical coherence tomography**. International Journal of Computer Assisted Radiology and Surgery, pp. 1–7, 2023.

[55] [LIU et al. 2019] Y. Liu, B. Fan, S. Xiang und C. Pan. **Relation-Shape Convolutional Neural Network for Point Cloud Analysis**. 2019.

[56] [LOSHCHILOV und HUTTER 2016] I. Loshchilov und F. Hutter. **SGDR: Stochastic Gradient Descent with Warm Restarts**. arXiv preprint arXiv:1608.03983, 2016.

[57] [LU et al. 2019] W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan und S. Song. **Deepvcp: An end-to-end deep neural network for point cloud registration**. pp. 12–21, 2019.

[58] [LUCAS und KANADE 1981] B. D. Lucas und T. Kanade. **An Iterative Image Registration Technique with an Application to Stereo Vision**. 1981.

[59] [LUO et al. 2022] J. Luo, M. Yuan, K. Fu, M. Wang und C. Zhang. **Deep Graph Matching Based Dense Correspondence Learning Between Non-Rigid Point Clouds**. IEEE Robotics and Automation Letters, Vol. 7:5842–5849, 2022.

[60] [Magnetic Resonance Imaging] **Magnetic Resonance Imaging (MRI)**. `https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri`. Accessed: 2023-07-13.

[61] [MARKL et al. 2012] M. Markl, A. Frydrychowicz, S. Kozerke, M. Hope und O. Wieben. **4D flow MRI**. Journal of Magnetic Resonance Imaging, Vol. 36(5):1015–1036, 2012.

[62] [MARKL et al. 2007] M. Markl, A. Harloff, T. A. Bley, M. Zaitsev, B. Jung, E. Weigang, M. Langer, J. Hennig und A. Frydrychowicz. **Time-resolved 3D MR velocity mapping at 3T: improved navigator-gated assessment of vascular anatomy and blood flow**. Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine, Vol. 25(4):824–831, 2007.

[63] [MYRONENKO und SONG 2010] A. Myronenko und X. Song. **Point set registration: Coherent point drift**. IEEE transactions on pattern analysis and machine intelligence, Vol. 32(12):2262–2275, 2010.

[64] [PARK et al. 2018] C.-A. Park, C.-K. Kang, Y.-B. Kim und Z.-H. Cho. **Advances in MR angiography with 7T MRI: From microvascular imaging to functional angiography**. Neuroimage, Vol. 168:269–278, 2018.

[65] [PETTY et al. 2000] G. W. Petty, R. D. Brown Jr, J. P. Whisnant, J. D. Sicks, W. M. O'Fallon und D. O. Wiebers. **Ischemic stroke subtypes:**

a population-based study of functional outcome, survival, and recurrence. Stroke, Vol. 31(5):1062–1068, 2000.

[66] [POIESI und BOSCAINI 2020] F. Poiesi und D. Boscaini. **Distinctive 3D local deep descriptors**. 2020.

[67] [QI et al. 2017] C. R. Qi, H. Su, K. Mo und L. J. Guibas. **Pointnet: Deep learning on point sets for 3d classification and segmentation**. pp. 652–660, 2017.

[68] [QI et al. 2017] C. R. Qi, L. Yi, H. Su und L. J. Guibas. **Pointnet++: Deep hierarchical feature learning on point sets in a metric space**. Advances in neural information processing systems, Vol. 30, 2017.

[69] [REN et al. 2022] B. Ren, L. Tan, Y. Song, D. Li, B. Xue, X. Lai und Y. Gao. **Cerebral small vessel disease: neuroimaging features, biochemical markers, influencing factors, pathological mechanism and treatment**. Frontiers in Neurology, Vol. 13:843953, 2022.

[70] [ROBBINS und MONRO 1951] H. Robbins und S. Monro. **A stochastic approximation method**. The Annals of Mathematical Statistics, Vol. 22(3):400–407, 1951.

[71] [RUSINKIEWICZ und LEVOY 2001] S. Rusinkiewicz und M. Levoy. **Efficient variants of the ICP algorithm**. Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM, pp. 145–152, 2001.

[72] [RUSU et al. 2009] R. B. Rusu, N. Blodow und M. Beetz. **Fast point feature histograms (FPFH) for 3D registration**. pp. 3212–3217, 2009.

[73] [RUSU et al. 2008] R. B. Rusu, Z. C. Marton, N. Blodow und M. Beetz. **Persistent point feature histograms for 3D point clouds**. pp. 119–128, 2008.

[74] [RYU und ICHIRO KAMATA 2021] J. Ryu und S. ichiro Kamata. **An efficient computational algorithm for Hausdorff distance based on points-ruling-out and systematic random sampling**. Pattern Recognition, Vol. 114:107857, 2021.

[75] [SAITI und THEOHARIS 2020] E. Saiti und T. Theoharis. **An application independent review of multimodal 3D registration methods**. Computers & Graphics, Vol. 91:153–178, 2020.

[76] [SAITI und THEOHARIS 2022] E. Saiti und T. Theoharis. **Multimodal registration across 3D point clouds and CT-volumes**. Computers & Graphics, Vol. 106:259–266, 2022.

[77] [SARODE et al. 2019] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey und H. Choset. **PCRNet: Point Cloud Registration Network using PointNet Encoding**. 2019.

[78] [SEGAL et al. 2009] A. V. Segal, D. Hähnel und S. Thrun. **Generalized-ICP**. 2009.

[79] [SINGH] **Learning3D: A Modern Library for Deep Learning on 3D Point Clouds**.

[80] [SINKHORN 1964] R. Sinkhorn. **A relationship between arbitrary positive matrices and doubly stochastic matrices**. The annals of mathematical statistics, Vol. 35(2):876–879, 1964.

[81] [SPITZ 2020] L. Spitz. **Multi-Modal Co-Registration of High-Resolution 7T MRI Vessel Data**. Masterthesis, Dept. of Computer Science, 2020.

[82] [SPITZ et al. 2022] L. Spitz, M. Allgaier, A. Mpotsaris, D. Behme, B. Preim und S. Saalfeld. **Segmentation of Circle of Willis from 7T TOF-MRI data and immersive exploration using VR**. Current Directions in Biomedical Engineering, Vol. 8(1):129–132, 2022.

[83] [SPITZ et al. 2023] L. Spitz, F. Gaidzik, D. Stucht, H. Mattern, B. Preim und S. Saalfeld. **A hybrid hierarchical strategy for registration of 7T TOF-MRI to 7T PC-MRI intracranial vessel data**. International Journal of Computer-Assisted Radiology and Surgery, Vol. 18(5):837–844, 2023.

[84] [TANAKA et al.] **Probreg: Probablistic Point Cloud Registration Library. 2020**.

[85] [THOMAS et al. 2019] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette und L. J. Guibas. **KPConv: Flexible and Deformable Convolution for Point Clouds**. 2019.

[86] [WANG et al. 2019] L. Wang, J. Chen, X. Li und Y. Fang. **Non-Rigid Point Set Registration Networks**. 2019.

[87] [WANG und SOLOMON 2019] Y. Wang und J. M. Solomon. **Deep closest point: Learning representations for point cloud registration**. pp. 3523–3532, 2019.

[88] [WANG und SOLOMON 2019] Y. Wang und J. M. Solomon. **Prnet: Self-supervised learning for partial-to-partial registration**. Advances in neural information processing systems, Vol. 32, 2019.

[89] [WANG et al. 2018] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein und J. M. Solomon. **Dynamic Graph CNN for Learning on Point Clouds**. ACM Transactions on Graphics, Vol. 38:13, 2018.

[90] [WANG et al. 2021] Z. Wang, Q. Chen, J. Chen, N. Yang und K. Zheng. **Risk factors of cerebral small vessel disease: A systematic review and meta-analysis**. Medicine, Vol. 100:E28229, 2021.

[91] [WIKIPEDIA CONTRIBUTORS] **Rotation matrix — Wikipedia, The Free Encyclopedia**. [Online; accessed 29-July-2023].

[92] [WU et al. 2021] T. Wu, L. Pan, J. Zhang, T. Wang, Z. Liu und D. Lin. **Density-aware Chamfer Distance as a Comprehensive Metric for Point Cloud Completion**. 2021.

[93] [WU et al. 2015] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang und J. Xiao. **3D ShapeNets: A Deep Representation for Volumetric Shapes**. 2015.

[94] [WYMER et al. 2020] D. T. Wymer, K. P. Patel, W. F. Burke und V. K. Bhatia. **Phase-contrast MRI: Physics, techniques, and clinical applications**. Radiographics, Vol. 40:122–140, 2020.

[95] [YANG et al. 2016] J. Yang, H. Li, D. Campbell und Y. Jia. **Go-ICP: A Globally Optimal Solution to 3D ICP Point-Set Registration**. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 38(11):2241–2254, 2016.

[96] [YANG et al. 2017] J. Yang, Q. Zhang, Y. Xiao und Z. Cao. **TOLDI: An effective and robust approach for 3D local shape description**. Pattern Recognition, Vol. 65:175–187, 2017.

[97] [YANG et al. 2019] Z. Yang, J. Z. Pan, L. Luo, X. Zhou, K. Grauman und Q. Huang. **Extreme relative pose estimation for rgb-d scans via scene completion**. pp. 4531–4540, 2019.

[98] [YEW und LEE 2020] Z. J. Yew und G. H. Lee. **Rpm-net: Robust point matching using learned features**. pp. 11824–11833, 2020.

[99] [YUAN et al. 2020] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox und J. Kautz. **Deepgmr: Learning latent gaussian mixture models for registration**. pp. 733–750, 2020.

[100] [ZENG et al. 2021] Y. Zeng, Y. Qian, Z. Zhu, J. Hou, H. Yuan und Y. He. **CorrNet3D: Unsupervised End-to-end Learning of Dense Correspondence for 3D Point Clouds**. 2021.

[101] [ZHANG et al. 2021] J. Zhang, Y. Yao und B. Deng. **Fast and robust iterative closest point**. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 44(7):3450–3466, 2021.

[102] [ZHANG et al. 2020] Z. Zhang, Y. Dai und J. Sun. **Deep learning based point cloud registration: an overview**. Virtual Reality & Intelligent Hardware, Vol. 2(3):222–246, 2020. 3D Visual Processing and Reconstruction Special Issue.

[103] [ZHAO et al. 2019] H. Zhao, L. Jiang, C.-W. Fu und J. Jia. **PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing**. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5560–5568, 2019.

[104] [ZHAO et al. 2019] Y. Zhao, T. Birdal, H. Deng und F. Tombari. **3D Point Capsule Networks**. 2019.

[105] [ZHOU et al. 2018] Q.-Y. Zhou, J. Park und V. Koltun. **Open3D: A modern library for 3D data processing**. arXiv preprint arXiv:1801.09847, 2018.

[106] [ZHU et al. 2022] L. Zhu, H. Guan, C. Lin und R. Han. **Leveraging Inlier Correspondences Proportion for Point Cloud Registration**. 2022.

[107] [ZHU et al. 2021] L. Zhu, D. Liu, C. Lin, R. Yan, F. Gómez-Fernández, N. Yang und Z. Feng. **Point Cloud Registration using Representative Overlapping Points**. 2021.

[108] [ÖZKAN ÖZSARLAK et al. 2004] Özkan Özsarlak, J. W. V. Goethem, M. Maes und P. M. Parizel. **MR angiography of the intracranial vessels: Technical aspects and clinical applications**. Neuroradiology, Vol. 46:955–972, 2004.

Declaration of Academic Integrity

I hereby declare that I have written the present work myself and did not use any sources or tools other than the ones indicated.

Datum: .....

(Signature)