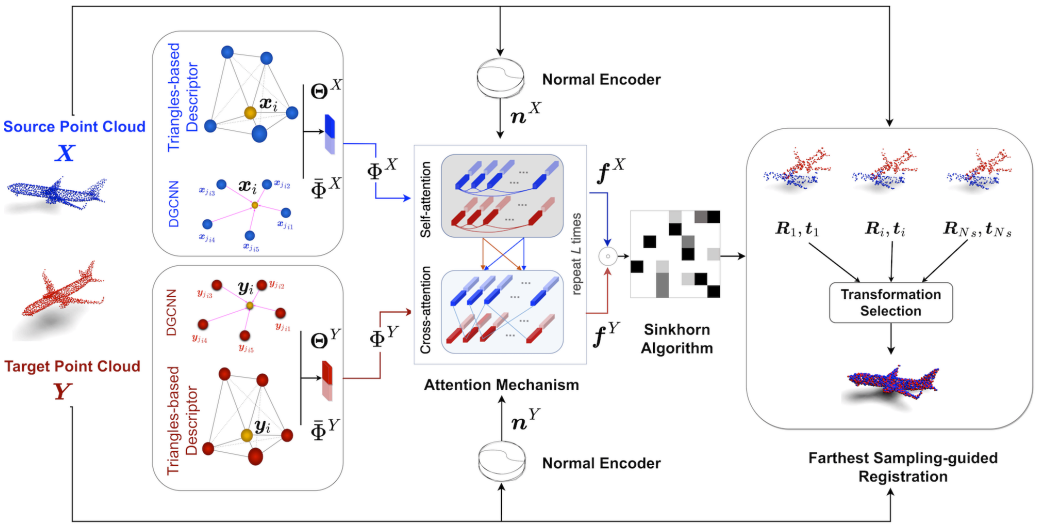
RoCNet++: Triangle-based descriptor for accurate and robust point cloud registration

Karim Slimani, Catherine Achard, Brahim Tamadazte

Graphical Abstract

RoCNet++: Triangle-based descriptor for accurate and robust point cloud registration

Karim Slimani, Catherine Achard, Brahim Tamadazte



Highlights

RoCNet++: Triangle-based descriptor for accurate and robust point cloud registration

Karim Slimani, Catherine Achard, Brahim Tamadazte

- Point cloud registration method based on a new descriptor encoding triangles
- Experimental validation on three well-established point clouds datasets: ModelNet40, KITTI and 3DMatch
- Point cloud registration using triangles demonstrates efficiency in terms of accuracy and robustness

RoCNet++: Triangle-based descriptor for accurate and robust point cloud registration

Karim Slimani[a], Catherine Achard[a], Brahim Tamadazte[a]

[a]*ISIR, Sorbonne Université, CNRS UMR 7222, INSERM U1150, 4 place Jussieue, Paris, 75005, , France*

Abstract

This paper introduces RoCNet++, a point cloud registration method with two main contributions, one concerning the design of a robust descriptor and another concerning the estimation of the rigid transformation. First, to robustly capture the local geometric properties of the surface, i.e., each point is characterised by all the triangles formed by itself and its nearest neighbours in the 3D point cloud. The idea is to assist the learning of the descriptor by introducing *a priori* information about interesting geometric properties such as the invariance of triangle angles under rigid transformations. This local triangle-based descriptor is integrated into the recently developed RoCNet architecture for estimating the correspondences between source and target point clouds. We then introduce the Farthest Sampling-guided Registration (FSR), which relies on successive farthest point samplings to estimate the global rigid transformation between 3D point clouds. The new proposed architecture RoCNet++ has been evaluated in different configurations: clean, noisy and partial data on both synthetic and real databases such as ModelNet40, KITTI, and 3DMatch. RoCNet++ shows improved performances on

Email address: {karim.slimani@isir.upmc.fr} (Karim Slimani)

these benchmark datasets in favourable and unfavourable conditions. Furthermore, both the local triangle-based descriptor and the Farthest Sampling-guided Registration (FSR) can be used in other registration algorithms.
Keywords: Point Cloud Learning, Registration, Geometric Descriptor, Attention Mechanism, Pose Estimation

1. Introduction

3D point cloud registration is a fundamental step in many robotics and computer vision tasks with applications in autonomous driving, visual servoing, augmented reality and medical imaging.

Given two overlapping sets of unordered points, point cloud registration aims to compute the rigid transformation that projects them onto each other. This inevitably involves estimating the point-wise correspondences between the two sets. Different methods attempt to solve this challenging step either incrementally by minimising a predefined metric, as in standard algorithms such as ICP [3] and RANSAC [5], or by estimating point cloud descriptors followed by nearest neighbour search in the descriptor space, as in the feature learning methods PRNET [28] and SpinNet [1]. This type of method aims to provide rigid transformation invariant features while dealing with the disorder aspect of the point cloud and taking advantage of the local surface properties. For example, PointNet [16] projects the input data into a learned canonical space and uses a symmetric function to keep the output invariant to the order of the input points. In the same spirit, the widely used DGCNN [29] proposes to capture local properties between points by introducing successive convolutional layers. They take as input graphs whose

nodes are the K Nearest Neighbours (KNN) of each point. These graphs are dynamically updated in each layer by searching for the new KNN in the current feature space instead of the Cartesian coordinate space.

To learn the information on both local and global scales, most of the recent methods feed the extracted features into a transformer module as in DCP [27] and VRNet [37]. Different models build on the classical transformers, adding richer geometric information such as the pairwise distances and triplet angles encoded by GeoTransformer [18] and the surface normal variations in RoCNet [23].

The features from the source point cloud are projected onto the target point cloud to create a similarity matrix used in a matching module that finds correspondences between the two point clouds. Sinkhorn algorithm [22] or dual *softmax* [13] are often employed for this. Once correspondences have been established, the transformation is usually computed through methods like Singular Values Decomposition (SVD) as in MDGAT [21] or RANSAC as in Predator [7]. Other methods predict point weights from the similarity matrix and estimate the transformation using a soft SVD as done in [27] or in SACF [30], where a skip-attention decoder is added to filter out false matches.

In this paper, we introduce RoCNet++, an improvement on RoCNet [23], with two main contributions. The first one is a local triangle-based feature extractor that leverages the geometric properties of the point cloud, particularly the angle invariance of triangles under rigid transformation. This involves aggregating the three angles of the triangles formed by each point and its K nearest neighbours into a vector introduced into a simple Feed-

Forward Neural Network (FFNN) to provide a highly discriminative representation. This vector, combined with several high-level features well-known in the literature, systematically improves their performances.

The second main contribution is the Farthest Sampling-guided Registration module, which estimates multiple rigid transformations given a set of matched points by constructing multiple smaller subsets of points using a sampling based on farthest point exploration. A rigid transformation is then computed on each subset and the best one is selected according to the number of inliers provided. Both contributions are evaluated on the point cloud registration problem using a well-established dataset, i.e., the ModelNet40 [28], by integrating it into RoCNet [23], as shown in (Fig. 1). We also propose to evaluate them on high-resolution point clouds by adopting the coarse-to-fine matching strategy proposed by [18] on KITTI [6] and 3DMatch [35] datasets.

The various upgrades proposed showed significant impact making the architecture more accurate in nearly all the performed experiments, scoring impressive matching results on the clean data of ModelNet40 with over 99% in three metrics: precision, accuracy and recall.

2. Related Work

2.1. 3D Point Cloud Descriptors

Hand-crafted 3D descriptors aim to capture local point cloud information by incorporating various geometric characteristics such as distances, angles, surface normals, and curvature. Authors in TPSH [14] have demonstrated strong performance in terms of repeatability and robustness to noise by employing trigonometric projections to encode both spatial and geometric

feature statistics in histograms while maintaining high efficiency compared to state-of-the-art methods. Learning-based methods emerged and became widely used for different 3D analysis tasks. For instance, PointNet++ [17] that improves PointNet [16] architecture, proposes a set of abstraction modules that capture multiple scales information by uniformly down-sampling the input to obtain a set of centroids, grouping the points using the nearest neighbours of each centroid and predicting a features vector based on a PointNet layer. LGR-Net [39] introduced a novel attention module that allows combining local geometric attributes with the global point cloud topology by weighting the contribution of the two to consider the local region shape and thus ensures rotation-invariant features. Recent geometric registration methods for both deformable scenes [13] and rigid scenes [18] adopted Kernel Point Convolution (KP-Conv) [24] as a backbone taking advantage of its efficiency on large-scale data and its ability to simultaneously extract features and down-sample the input point cloud.

RepSurf [19] is a recent method proposing two variants of surface representation for point clouds using the second derivative from the Taylor series to describe the local geometry of each point. The first version utilizes features such as the normal vector, surface position, and normalized coordinates in triangles formed by points and their nearest neighbours. The second version proposes to learn a projection of the geometric features thanks to a Multi-Layers Perceptron (MLP) followed by a pooling operation. The method outperforms most of the state-of-the-art algorithms in segmentation, classification and 3D detection but exhibits limited robustness with noisy and disordered data. Triangle-NET [32] introduces various triangle-based func-

tions to embed local structures by measuring edge lengths, interior angles, and surface normals at each vertex. These geometric features are transformed into a latent representation using graph aggregation. The method showed interesting robustness to noise and scale-invariance properties.

2.2. Registration

2.2.1. Iterative Methods

As most of the iterative algorithms for point cloud registration, Iterative Closest Point (ICP) [3] contains two main stages: point-wise matching and transformation estimation. ICP iteratively refines the estimated pose by alternately assigning points from the source cloud to their nearest neighbours in the target cloud and computing transformations using an SVD [3]. Recent variants of ICP, such as Go-ICP [33] and Fast ICP [36], have introduced techniques to enhance convergence speed and robustness to outliers while drawing inspiration from the original ICP. The main drawback of iterative methods is their sensitivity to initialization as they may converge to local minima. To handle this limitation, authors in [12] have proposed to fuse learned features and geometric features that are updated by repositioning the input point clouds by a transformation estimated at each iteration. RANSAC [5] is another commonly used iterative algorithm. It estimates the transformation by randomly selecting subsets of correspondences at each iteration, transforming the source cloud, and identifying inliers within a specified distance threshold. The global transformation is determined using the subset with the highest number of inliers. The authors of WSDesc [11] employ differentiable voxelization to capture the local geometry. A voxel grid encodes in each voxel the probability of containing at least one point. Its orientation is

defined by a local reference frame as its scale is learned by the network.

2.2.2. Learning-based Methods

Many registration methods are now based on learning. For example, R-PointHop [9] extracts point descriptor invariant under rigid transformation using different neighbourhood sizes, thanks to a local reference frame (LRF) defined for the point by its nearest neighbours. The authors of WSDesc [11] employ differentiable voxelization to capture the local geometry. A voxel grid encodes in each voxel the probability of containing at least one point. Its orientation is defined by the LRF and its scale is learned by the network. The method shows very interesting results on geometric registration benchmarks, however, the point-to-voxel operation may lead to significant computation costs for dense point clouds. DCP [27] and RTE [38] propose to use features which are extracted with DGCNN [29] and enhanced using attention mechanisms. A *softmax* applied to the dot product of these features leads to the matching that is introduced in a differentiable soft SVD [15] to predict the transformation. More recently, VRNet [37] which is built in the same spirit, uses a correction-walk module to construct Rectified virtual Corresponding Points (RCPs) having the shape of the source and the same pose as the target. Although it demonstrated impressive results on ModelNet40, experiments showed the method encountered more difficulties in very low overlap and symmetric cases. Early 2023, RoCNet [23] introduces a surface normal variations encoding transformer based on sinusoidal functions [25]. This encoding is fed to a transformer which also takes as input point-wise features extracted with DGCNN. The matching is then solved as an optimal transport problem using Sinkhorn [22] allowing RANSAC to estimate the

transformation in a limited number of iterations. PRNet [28] learns jointly keypoints detection, representation and matching and has been specifically designed to manage occlusion partial-to-partial point cloud registration. Geo-Transformer [18] detects superpoints and encodes their geometric structure using pair-wise distances and triplet-wise angles. The matching is performed by first searching the correspondence of super points before matching finer-resolution dense points. The rigid transformation is then estimated in a local-to-global fashion.

3. Proposed Method

The registration method described in this article introduces two major improvements to the RoCNet [23] architecture, the first by adding a descriptor based on triangles invariance properties and the second one on the estimation of the rigid transformation between the point clouds using Farthest Sampling-Guided registration. An overview of RoCNet++ is depicted in Figure 1.

3.1. Background: the RoCNet Algorithm

The prior method RoCNet [23] relies on the following three main modules. First, given two point clouds to register $\boldsymbol{X} \subset \mathbb{R}^{3\times M}$ and $\boldsymbol{Y} \subset \mathbb{R}^{3\times N}$, the method utilizes the graph convolution based DGCNN [29] to extract high dimensional features $\boldsymbol{\Phi}^X \subset \mathbb{R}^{d\times M}$ and $\boldsymbol{\Phi}^Y \subset \mathbb{R}^{d\times N}$ for each point from the two point clouds.

Second, a transformer is used to update features by applying successive self-attention and cross-attention mechanisms thanks to the classical equa-

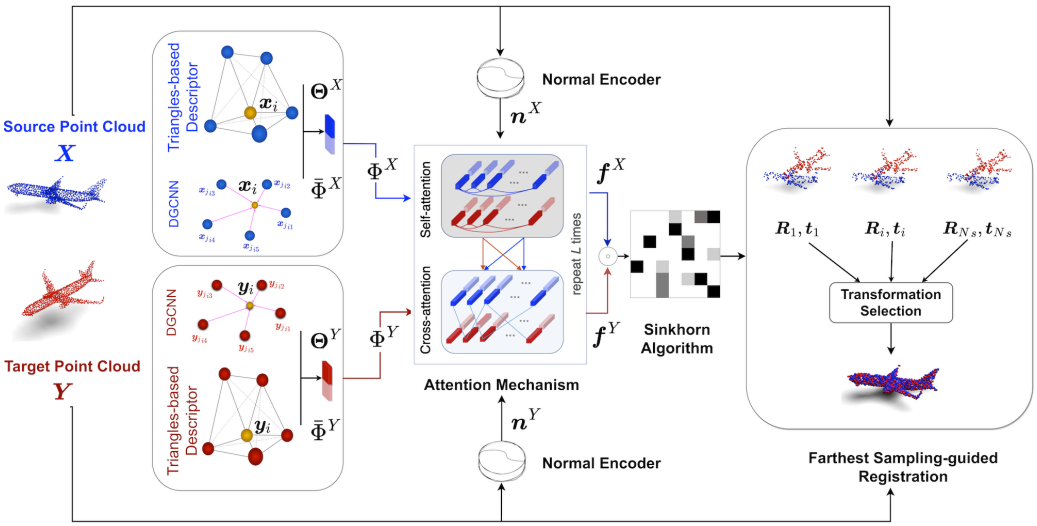


Figure 1: Overview of the RoCNet++ pipeline.

tions, respectively:

$$\mathbf{\Phi}_i^X = \sum_{j=1} \alpha_{ij}(\mathbf{\Phi}_j^X \mathbf{W}^v) \tag{1}$$

$$\mathbf{\Phi}_i^X = \sum_{j=1} \alpha_{ij}(\mathbf{\Phi}_j^Y \mathbf{W}^v) \tag{2}$$

where $\mathbf{W}^v \in \mathbb{R}^{d \times d}$ is the learned projection matrix for values, $\mathbf{\Phi}_j^X$ and $\mathbf{\Phi}_j^Y$ represent the feature vectors of the j^{th} point from the point cloud X and Y in that order. In the case of self-attention, α_{ij} coefficients are computed according to 3:

$$\alpha_{ij} = \underset{j}{softmax}\left(\frac{(\mathbf{\Phi}_i^X \mathbf{W}_Q^s)(\mathbf{\Phi}_j^X \mathbf{W}_K^s + \boldsymbol{n}_{i,j}^X \mathbf{W}_R^s)^T}{\sqrt{d}}\right) \tag{3}$$

where $\mathbf{W}_Q^s$, $\mathbf{W}_K^s$ and $\mathbf{W}_R^s \in \mathbb{R}^{d \times d}$ are the projection parameters for queries, keys and normal-based embedding $\boldsymbol{n}_{i,j}^X$, d is the dimension of the features $\mathbf{\Phi}_i^X$. For cross-attention aggregation, α_{ij} are computed by:

$$\alpha_{ij} = \underset{j}{softmax}\left(\frac{(\mathbf{\Phi}_i^X \mathbf{W}_Q^c)(\mathbf{\Phi}_j^Y \mathbf{W}_K^c)^T}{\sqrt{d}}\right) \tag{4}$$

where $\mathbf{W}_Q^c$ and $\mathbf{W}_K^c$ represent the projection parameters for queries and keys. The self-attention mechanism uses a surface normals encoder inspired by the frequency embedding introduced in [25]. The main motivation of the authors is to design a robust descriptor that captures the connections between points while being invariant to 3D transformations. Every couple of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are linked with a vector $\boldsymbol{g}_{\boldsymbol{x}_i,\boldsymbol{x}_j}$ embedding the variation of their normal orientation $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ by sinusoidal functions whose frequency explicitly depends on the vector indices as shown in the following:

$$\begin{cases} \boldsymbol{g}^{2p}_{\boldsymbol{x}_i,\boldsymbol{x}_j} = \sin\left(\frac{\angle(\boldsymbol{n}_i,\boldsymbol{n}_j)}{\tau \times 10000^{2p/d}}\right) \\ \boldsymbol{g}^{2p+1}_{\boldsymbol{x}_i,\boldsymbol{x}_j} = \cos\left(\frac{\angle(\boldsymbol{n}_i,\boldsymbol{n}_j)}{\tau \times 10000^{2p/d}}\right) \end{cases} \tag{5}$$

where p corresponds to the current value index of $\boldsymbol{g}_{\boldsymbol{x}_i,\boldsymbol{x}_j}$, τ is a normalisation coefficient and d the dimension of the descriptor. A learned projection $\mathbf{W}_N^s \in \mathbb{R}^{d\times d}$ is applied to get a final embedding $\boldsymbol{n}^X_{i,j} = \boldsymbol{g}_{\boldsymbol{x}_i,\boldsymbol{x}_j}\mathbf{W}_N^s$.

Let us call $\boldsymbol{f}_i^X$ and $\boldsymbol{f}_j^Y$ the final output features for the points $\boldsymbol{x}_i \in \boldsymbol{X}$ and $\boldsymbol{y}_j \in \boldsymbol{Y}$, respectively. Their cross projection gives the similarity matrix $\mathbf{S} \in \mathbb{R}^{M\times N}$. As reported in [21], a matching module relying on the Sinkhorn algorithm [22] estimates point-wise correspondences by incrementally normalizing $\mathbf{S}$ thanks to the iterations detailed in (6) and (6)

$$\begin{aligned} \textit{Iteration } n: &\qquad \mathbf{S'}_{i,j} = \mathbf{S}_{i,j} - \log\textstyle\sum_j \exp(\mathbf{S}_{i,j}) \\ \textit{Iteration } n+1: &\qquad \mathbf{S}_{i,j} = \mathbf{S'}_{i,j} - \log\textstyle\sum_i \exp(\mathbf{S'}_{i,j}) \end{aligned} \tag{6}$$

The final matrix will be called $\tilde{\mathbf{S}}$. A mutual top score searching along the two dimensions of $\tilde{\mathbf{S}}$ is used to find the hard correspondences between points which are used as input for a RANSAC [5] that estimates the rigid transformation within 500 iterations.

3.2. Local Triangle-based Descriptor

The main idea developed in this work and illustrated in Figure 2, is to take advantage of the invariance of triangle angles when subjected to rigid transformations. Thus, whatever transformation is applied to a point cloud, the triangles formed by 3 points before and after the transformation will have exactly the same angles.

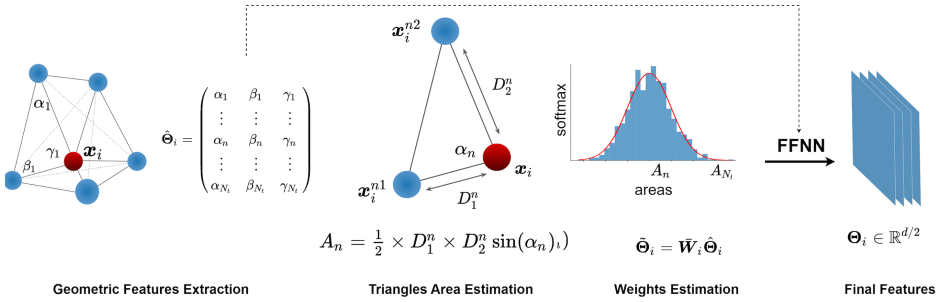


Figure 2: Triangle-based descriptor illustration for the point $\boldsymbol{x}_i$

Considering a point $\boldsymbol{x}_i$ and its K nearest neighbours, the set of triples defined from these $K+1$ points and containing $\boldsymbol{x}_i$ allows to construct N_t triangles that capture the local surface structure. A discriminative and rigid transformation invariant embedding of the point $\boldsymbol{x}_i$ is formed by the three interior angles of each triangle $\theta_n = (\alpha_n, \beta_n, \gamma_n)$ with $n \leq N_t$, thereby creating the feature vector $\hat{\boldsymbol{\Theta}}_i = \{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_n, ..., \boldsymbol{\theta}_{N_t}\} \in \mathbb{R}^{N_t \times 3}$.

If the interior angles of the triangles remain invariant under rigid transformations, they are also sensitive to noise and the smaller the triangles, the more sensitive they are. For each $\boldsymbol{x}_i$, we have therefore decided to weight the contribution of each triangle by its area using a *softmax*:

$$\boldsymbol{W}_i = \underset{n}{softmax}\left(\frac{1}{2}\left\|\boldsymbol{x}_i - \boldsymbol{x}_i^{n1}\right\|\left\|\boldsymbol{x}_i - \boldsymbol{x}_i^{n2}\right\|\sin\left(\alpha_n\right)\right)\ ,\ n \in [1, N_t] \qquad (7)$$

where $\boldsymbol{x}_i^{n1}$, $\boldsymbol{x}_i^{n2}$, $\boldsymbol{x}_i$ are the three vertices of the n^{th} triangle and α_n is the angle at vertex $\boldsymbol{x}_i$.

The updated feature vector $\tilde{\boldsymbol{\Theta}}_i \in \mathbb{R}^{N_t \times 3}$ is obtained with:

$$\tilde{\boldsymbol{\Theta}}_i = \bar{\boldsymbol{W}}_i \hat{\boldsymbol{\Theta}}_i \tag{8}$$

where $\bar{\boldsymbol{W}}_i \in \mathbb{R}^{N_t \times 3}$ is obtained by reshaping $\boldsymbol{W}_i \in \mathbb{R}^{N_t}$ in order to get the same weight for the three angles of the same triangle $\bar{\boldsymbol{W}}_i = [\boldsymbol{W}_i, \boldsymbol{W}_i, \boldsymbol{W}_i]$.

Finally, aiming for better robustness, we predict a high-dimensional descriptor $\boldsymbol{\Theta}_i \in \mathbb{R}^{d/2}$ by feeding $\tilde{\boldsymbol{\Theta}}_i$ to an FFNN which applies 1D convolutions, batch normalisation and *ReLU* function to the input.

We divide the dimension of the DGCNN [29] descriptor $\boldsymbol{\Phi}^X \subset \mathbb{R}^{d \times M}$ used in the baseline method RoCNet [23] by two ($\bar{\boldsymbol{\Phi}}^X \subset \mathbb{R}^{\frac{d}{2} \times M}$) and concatenate it with the proposed triangle based descriptor, such that the dimension of the whole descriptor remains d:

$$\Phi^X \longleftarrow [\bar{\Phi}^X, \boldsymbol{\Theta}^X] \tag{9}$$

$$\Phi^Y \longleftarrow [\bar{\Phi}^Y, \boldsymbol{\Theta}^Y] \tag{10}$$

Φ^X and Φ^Y are subsequently introduced in the self-attention and cross-attention mechanisms detailed in (1), (2), (3) and (4) as in the baseline method RoCNet.

3.3. Farthest Sampling-guided Registration (FSR)

The second contribution we propose takes place at the end of the process when the rigid transformation is estimated from the matching points.

Inspired by RANSAC [5] and the local-to-global registration proposed by [18], we design a Farthest Sampling-guided Registration (FSR) module

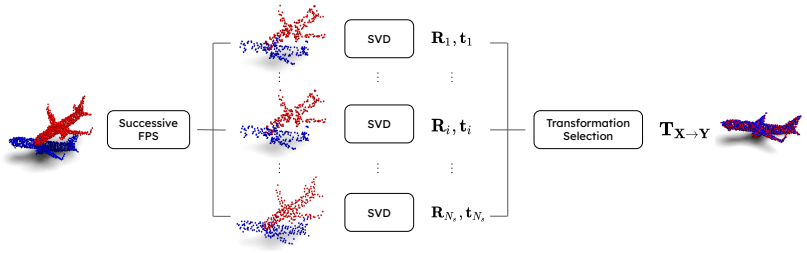


Figure 3: Farthest Sampling-guided Registration overview

that combines the advantages of both methods. Given the set of correspondences $\mathbf{C}_{\mathbf{X}\to\mathbf{Y}}$ between the two input point clouds, N_s subsets $\bar{\mathbf{C}}_{\mathbf{X}_i\to\mathbf{Y}_i}$, $i \leq N_s$ are created using the farthest point sampling algorithm proposed in [4], which preserves the structure of the point cloud in each subset by searching for the most dispersed points as possible, starting from a randomly chosen initial point. A first subset is sampled and the operation is repeated N_s times by removing the collected pairs from the original set of correspondences at each iteration to get N_s different sets of N_p points as far apart as possible. The cross-variance matrix $\mathbf{H}_i$ of each subset $\bar{\mathbf{C}}_{\mathbf{X}_i\to\mathbf{Y}_i}$ is then computed and decomposed to singular values with $\mathbf{H}_i = \mathbf{U}_i\mathbf{\Sigma}_i\mathbf{V}_i^T$ to obtain the rigid transformation as usually:

$$\mathbf{R}_i = \mathbf{V}_i\mathbf{U}_1^T \quad \text{and} \quad \mathbf{t}_i = -\mathbf{R}_i\mathbf{X}_i + \mathbf{Y}_i \tag{11}$$

where $\mathbf{X}_i$ and $\mathbf{Y}_i$ are input and target point cloud for the subset $\bar{\mathbf{C}}_{\mathbf{X}_i\to\mathbf{Y}_i}$, $\mathbf{R}_i$ and $\mathbf{t}_i$ are the rotation matrix and the translation vector from $\mathbf{X}_i$ to $\mathbf{Y}_i$. Using all subsets of matching points, different rotation matrices $\{\mathbf{R}_1, ..., \mathbf{R}_i, ..., \mathbf{R}_{N_s}\}$ and translation vectors $\{\mathbf{t}_1, ..., \mathbf{t}_i, ..., \mathbf{t}_{N_s}\}$ are obtained. As in [18], FSR chooses the final transformation as the one which provides the maximum

number of inliers from the initial set of correspondences $\mathbf{C}_{\mathbf{X}\to\mathbf{Y}}$:

$$\mathbf{R}_{\mathbf{X}\to\mathbf{Y}}, \mathbf{t}_{\mathbf{X}\to\mathbf{Y}} = \underset{i}{argmax} \sum_{n=1}^{|\mathbf{C}_{\mathbf{X}\to\mathbf{Y}}|} [\![\mathbf{R}_i \boldsymbol{x}_n + \mathbf{t}_i - \boldsymbol{y}_n < \tau]\!] \tag{12}$$

where $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ are a pair of matching points from $\mathbf{C}_{\mathbf{X}\to\mathbf{Y}}$, $\mathbf{R}_i$, $\mathbf{t}_i$ are the rotation and the translation estimated for the i^{th} subset of points, $|\mathbf{C}_{\mathbf{X}\to\mathbf{Y}}|$ is the number of matched points and τ is a threshold for determining whether a pair consisting of a transformed point from the source point cloud and its estimated corresponding point from the target point cloud is an inlier or not.

Thus, FSR works similarly to RANSAC, except that the subsets used are not selected randomly, but are chosen to have points that are as far apart as possible, allowing a gain in efficiency by estimating an accurate transformation in just a few iterations. The intuitive idea is that the further apart the points are, the more robust the rigid transformation is estimated.

4. Experiments

The proposed architecture RoCNet++, with the addition of triangle-based descriptor and FSR registration, is evaluated for the first time on ModelNet40 [31] dataset. Secondly, we evaluate the added value of the new architecture on the KITTI database [6] and 3DMatch indoor scenes [35]. Since the feature extraction by DGCNN and the normal encoding transformer is encountering an out-of-memory issue due to excessive GPU consumption, we adopt the coarse-to-fine matching strategy proposed in GeoTransformer [18] on which we incorporate our descriptor by dividing the output dimension of the KP-Conv backbone used in the original paper and concatenating its predicted feature with those extracted by our triangle-based descriptor.

Specifically, we begin by collecting a low-resolution point set from each input point cloud, resulting in sets of superpoints $\hat{\boldsymbol{X}} \subset \mathbb{R}^{3\times\hat{M}}$ and $\hat{\boldsymbol{Y}} \subset \mathbb{R}^{3\times\hat{N}}$. We also gather higher-resolution sets (half the input resolution) to create dense points, $\tilde{\boldsymbol{X}} \subset \mathbb{R}^{3\times\tilde{M}}$ and $\tilde{\boldsymbol{Y}} \subset \mathbb{R}^{3\times\tilde{N}}$. Next, we extract combined features from KP-Conv and the triangle-based descriptors for the superpoints before feeding them into the normal encoding attention mechanism. We denote the final features for the superpoints $\hat{\boldsymbol{x}}_i \in \hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{y}}_j \in \hat{\boldsymbol{Y}}$ as $\hat{\boldsymbol{f}_i^X}$ and $\hat{\boldsymbol{f}_j^Y}$, respectively. To establish correspondences between superpoints, we compute a score matrix $\hat{\mathbf{S}} \subset \mathbb{R}^{\hat{M}\times\hat{N}}$ using the formula $\hat{s}_{i,j} = \exp\Big(-|\hat{\boldsymbol{f}_i^X} - \hat{\boldsymbol{f}_j^Y}|2^2\Big)$. Finally, we construct the set of superpoint correspondences $\mathbf{C}_{\hat{\boldsymbol{X}}\to\hat{\boldsymbol{Y}}}$ by selecting the top-k scores from $\hat{\mathbf{S}}$. Once the superpoints paired, local patches $\boldsymbol{P}_i^X$ and $\boldsymbol{P}_j^Y$ are built around each $\hat{\mathbf{x}}_i \in \hat{\boldsymbol{X}}$ and $\hat{\mathbf{y}}_i \in \hat{\boldsymbol{Y}}$ by assigning each dense point from $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{Y}}$ to its nearest neighbor from $\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$ respectively. Inside each local patch, the dense point features $\tilde{\boldsymbol{f}_i^X}$ and $\tilde{\boldsymbol{f}_j^Y}$ are used to run an optimal transport algorithm as in (6) and get a partial assignment matrix $\tilde{\mathbf{S}}$. A mutual top selection on the obtained matrix gives us the dense correspondences $\mathbf{C}_{\tilde{\mathbf{X}}\to\tilde{\mathbf{Y}}}$. For a fair comparison, we repeat the same experiments as reported [23] and [18] on ModelNet40, KITTI [6] and 3DMatch [35] datasets. The performances are extracted from [37], [11] and [9] except Geo-Transformer [18] for which we use the official model implementation.

On ModelNet40, four measures will be presented as usually done in the related literature: the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE) for translation and rotation. We use the Registration Recall (RR), the Relative Rotation Error (RTE) and the Relative Translation Error (RTE) metrics on KITTI and 3DMatch as reported in [18].

4.1. Datasets

4.1.1. ModelNet40 [31]

The synthetic ModelNet40 benchmark contains man-made point clouds representing 40 different classes split into 9,843 point clouds for training and 2,468 point clouds for testing. As in [28], 1024 points are sampled on each object and then transformed by applying a random rotation between 0 and 45 degrees around each axe and translating it along each direction by a random distance between $[-0.5, 0.5]$ to create a target point cloud. A partial overlap is synthesized by sampling the 768 nearest neighbours of a random point in each point cloud. Finally, the 3D coordinates of each point are perturbed by adding a clipped Gaussian noise with a range of $[0.05, 0.05]$, a zero mean and a variance of $\sigma = 0.01$.

4.1.2. KITTI odometry dataset [6]

It is composed of 22 sequences of point clouds obtained with a Velodyne HDL64 LiDAR. Ground truth poses are provided using GPS/INS system for the sequences 00 to 10 followed by an ICP [3] refinement and consider point cloud pairs that are at least 10 meters away for testing and downsampling them with a voxel size of 30 cm to obtain approximately $20k$ points per point cloud. We follow [18] to use the first six sequences for training, the seventh and eighth for validation and the last three sequences for evaluation.

4.1.3. 3DMatch indoor scenes dataset [35]

To evaluate our method under challenging conditions with low-overlapping point clouds, we assess the matching and transformation estimation produced by RoCNet++ on the indoor scenes dataset 3DMatch. This dataset

comprises 62 scenes, with 46 used for training, 8 for validation, and 8 for testing purposes. We adopt the same down-sampling procedure as in prior works [7; 18], where each input point cloud is down-sampled using a voxel size of 2.5 *cm*.

4.2. Implementation Details

The method is implemented with PyTorch on a Nvidia Tesla V100-32G GPU. On ModelNet40, the network was trained with Adam optimizer [10] for 30 epochs on clean data and for 100 epochs on noisy data with a learning rate of 10^{-4}. The number of neighbours used for local triangle-based descriptor estimation is set to $K = 12$, leading to 66 triangles. The output dimension of DGCNN and the triangle-based descriptor is set to 64, producing a final concatenated descriptor with 128 dimensions. The farthest point sampling is operating $N_s = 5$ times during testing, with $N_p = 100$ points in each subset. The method is trained and supervised for feature extraction using the ground truth correspondences, while the transformation estimation bloc is parameter-free. For ModelNet40, the network is trained with a batch size of 4 examples using the gap loss [21] defined by:

$$L_{Gap} = \sum_{i=1}^{M} \log \left(\sum_{n=1}^{N+1} [\max((-\log \tilde{\mathbf{S}}_{i,\bar{i}} + \log \tilde{\mathbf{S}}_{i,n} + \alpha), 0)] + 1 \right) + \sum_{j=1}^{N} \log \left(\sum_{n=1}^{M+1} [\max((-\log \tilde{\mathbf{S}}_{j,\bar{j}} + \log \tilde{\mathbf{S}}_{n,j} + \alpha), 0)] + 1 \right) \quad (13)$$

where α is a positive scalar set to 0.5, $\tilde{\mathbf{S}}_{i,\bar{i}}$ and $\tilde{\mathbf{S}}_{j,\bar{j}}$ are the scores for the ground truth true matches of the points $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$, respectively.

In the context of KITII and 3DMatch, we trained the model for 100 and 30 epochs respectively with a batch size set to 1 using Adam optimizer [10]. We used the overlap-aware circle superpoint matching loss denoted as L_{op}, as reported in [18]. To elaborate, for each superpoint patch $\boldsymbol{P}_i^X$ originating from the source point cloud, we conducted a search to find its positive sample $\mathbf{Q}^+$

and negative sample $\mathbf{Q}^-$ within the target point cloud. This search involved considering all patches with an overlap greater than 10% with $\boldsymbol{P}_i^X$ for positive samples and all patches with no overlap for negative samples.

$$L_{op}^i = \frac{1}{|\mathbf{P}|} \sum_{i=1}^{|\mathbf{P}|} \log \left[1 + \sum_{\mathbf{p}_j^+ \in \mathbf{Q}^+} e^{\sqrt{(r_{i,j})}\theta_{i,j}^+ (d_{i,j} - \Delta^+)} \cdot \sum_{\mathbf{p}_k^- \in \mathbf{Q}^-} e^{\theta_{i,k}^- (\Delta^- - d_{k,i})} \right] \tag{14}$$

where $r_{i,j}$ and $d_{i,j}$ are the overlap ratio and the distance in the superpoint features space between $\hat{\boldsymbol{x}}_i$ and $\hat{\boldsymbol{y}}_j$ respectively, Δ^+ and Δ^- are positive and negative samples margins set to 0.1 and 1.4 respectively. Finally, $\theta_{i,j}^+ = \gamma(d_{j,i} - \Delta^+)$ and $\theta_{i,j}^- = \gamma(\Delta^- - d_{k,i})$ represent the weights for each sample.

In the case of the dense points matching loss, a set of ground truth superpoint correspondences $\mathbf{C}_{\hat{\boldsymbol{X}} \to \hat{\boldsymbol{Y}}}^{GT}$ is first picked. For each couple k of these correspondences, a subset of correspondences M_k is collected from the ground truth dense correspondences $\mathbf{C}_{\tilde{\boldsymbol{X}} \to \tilde{\boldsymbol{Y}}}^{GT}$ to compute the negative log-likelihood loss.

$$L_d^k = - \sum_{(x,y) \in M_k} \log \tilde{s}_{x,y}^k - \sum_{x \in I_i} \log \tilde{s}_{x,m_{k+1}}^k - \sum_{y \in J_k} \log \tilde{s}_{n_{k+1},y}^i \tag{15}$$

where I_i and J_i are the sets of dense points from the source and the target point clouds, respectively without any matching in the ground truth. The overall loss is the mean of the two previous losses. In the evaluation phase, FSR is performed for $N_s = 10$ iterations for both 250 and 500 correspondences, when $N_s = 20$ in all other cases. The sample size N_p is determined by dividing the number of correspondences by N_s. The refinement process from [2] is performed on the transformation estimated by FSR for 5 iterations on these two datasets.

4.3. Results on ModelNet40

4.3.1. ModelNet40 clean point clouds

Table 1 presents the registration performances on clean data from ModelNet40 with full and partial overlapping point clouds, respectively. RoCNet++ achieves the best performance with significant improvements in each one of the four metrics evaluating the rigid transformation, reducing the translation and rotation errors with a factor greater than 50% in most of the metrics compared to the second-best method. An example of performed registrations is depicted in Figure 4.

Table 1: Performances of RoCNet++ trained on all classes with clean point clouds on ModelNet40.

Method	full overlap				partial overlap			
	rotation		translation		rotation		translation	
	RMSE (↓)	MAE(↓)	RMSE(↓)	MAE(↓)	RMSE(↓)	MAE(↓)	RMSE(↓)	MAE(↓)
ICP [3]	12.28	4.613	0.04774	0.00228	33.683	25.045	0.293	0.2500
DCP-V2 [27]	1.090	0.752	0.00172	0.00117	6.709	4.448	0.027	0.0200
PRNET [28]	1.722	0.665	0.00637	0.00465	3.199	1.454	0.016	0.0100
R-PointHop [9]	0.340	0.240	0.00037	0.00029	1.660	0.350	0.014	0.0008
VRNet [37]	0.091	0.012	0.00029	0.00005	0.982	0.496	0.006	0.0039
WSDesc [11]	-	-	-	-	1.187	0.975	0.008	0.0070
GeoTransf [18]	0.232	0.075	0.00173	0.00063	0.327	0.118	0.002	0.0009
RoCNet [23]	0.082	0.011	0.00047	0.00008	0.412	0.133	0.002	0.0002
RoCNet++	**0.028**	**0.003**	**0.00022**	**0.00002**	**0.035**	**0.061**	**0.0003**	**0.00006**

4.3.2. ModelNet40 unseen categories

To access the generalisation performances of RoCNet++, errors on unseen categories are estimated and reported in Table 2. RoCNet++ outperforms other methods in translation estimation (equality with VRNet [37] in MAE

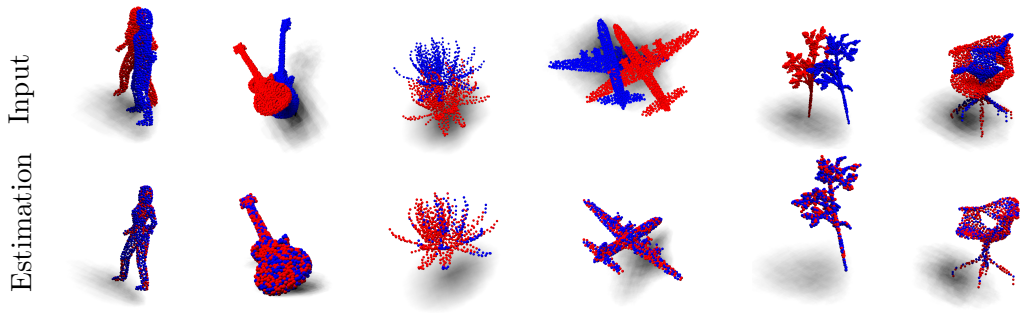


Figure 4: Illustration of some examples of performed registrations on ModelNet40 using RoCNet++ in case of clean data and full overlap *versus* the ground truth registrations.

and for the RMSE in rotations estimation. RoCNet [23] and VRNet [37] slightly overtake the proposed method in rotation for the MAE measure.

Table 2: Performances of RoCNet++ trained on half the classes with clean and full overlap point clouds on ModelNet40.

Method	RMSE(**R**)(↓)	MAE(**R**)(↓)	RMSE(**t**)(↓)	MAE(**t**)(↓)
ICP [3]	12.707	5.075	0.04853	0.00235
DCP-V2 [27]	3.256	2.102	0.00631	0.00462
PRNET [28]	3.060	1.326	0.01009	0.00759
R-PointHop [9]	0.340	0.250	0.00039	0.00030
VRNet [37]	0.209	0.028	0.00078	**0.00009**
GeoTransf [18]	0.253	0.067	0.00173	0.00100
RoCNet [23]	0.235	**0.026**	0.00180	0.00020
RoCNet++	**0.132**	0.031	**0.00023**	**0.00009**

4.3.3. ModelNet40 noisy data

To evaluate the robustness of the proposed method, we conducted two experiments on noisy point clouds. The results on full point clouds reported in Table 3 shows that RoCNet++ outperforms most of the recent state-of-the-art methods even in the presence of noise since the method is the best in

MAE($\mathbf{R}$) and MAE($\mathbf{t}$) while being the second best in the two other metrics where GeoTransformer [18] outperforms the other methods in rotation errors while R-PointHop [9] has the best performance in translation RMSE. Otherwise, the same Table shows that RoCNet++ is the best method in the case of noisy and partial overlapping input, thanks to its superiority in three over four metrics MAE($\mathbf{R}$), RMSE($\mathbf{t}$) and MAE($\mathbf{t}$) while being outperformed by only GeoTransformer [18] in RMSE($\mathbf{R}$).

Table 3: Performances of RoCNet++ trained on all classes of ModelNet40 with noisy data.

Method	full overlap				partial overlap			
	rotation		translation		rotation		translation	
	RMSE(↓)	MAE(↓)	RMSE(↓)	MAE(↓)	RMSE(↓)	MAE(↓)	RMSE(↓)	MAE(↓)
ICP [3]	11.971	4.497	0.04832	0.00433	33.067	25.564	0.294	0.250
DCP-V2 [27]	8.417	5.685	0.03183	0.02337	6.883	4.534	0.028	0.021
PRNET [28]	3.218	1.446	0.11178	0.00837	4.323	2.051	0.017	0.012
R-PointHop [9]	2.780	0.980	**0.00087**	0.00375	-	-	-	-
VRNet [37]	2.558	1.016	0.00570	0.00289	3.615	1.637	0.010	0.006
WSDesc [11]	-	-	-	-	3.500	0.759	0.006	0.004
GeoTransf [18]	**0.692**	0.267	0.00519	0.00200	**0.915**	0.386	0.007	0.003
RoCNet [23]	1.920	0.555	0.00260	0.00180	1.810	0.620	0.004	0.003
RoCNet++	1.004	**0.249**	0.00133	**0.00092**	1.278	**0.318**	**0.002**	**0.001**

4.4. KITTI Odometry Dataset

As discussed in Sec. 4, the large number of points contained in the KITTI dataset can lead to memory issues due to the combination of the Transformer and DGCNN. To tackle this problem, we adopted the approach presented in [18] using superpoints which consist of two steps. The first step is to match the superpoints extracted by KP-Conv. Secondly, dense correspondences are performed by creating local subsets of points for the

paired superpoints. This process involves collecting neighbouring points for each superpoint. To generate feature vectors for each point, we concatenate the proposed triangle descriptor with the KP-Conv features. In this scenario, we apply the Transformer with normal variation encoding as reported in [23] at the superpoints level.

Table 4: Registration performances on KITTI. The transformation is estimated using all the putative correspondences.

Method	RTE(cm)(↓)	RRE(deg)(↓)	RR(%)(↑)
Predator [7]	**6.8**	0.27	90.6
FMR [8]	∼66	1.49	90.6
SpinNet [1]	9.9	0.47	99.1
CoFiNet [34]	8.2	0.41	**99.8**
GeoTransformer [18]	**6.8**	<u>0.25</u>	**99.8**
Ours	<u>7.3</u>	**0.23**	**99.8**

Table 4 shows that our descriptor equals the state-of-the-art performance in Registration Recall (RR) and improves precision in Relative Rotation Error (RRE) while achieving the second-best performance in Relative Translation Error (RTE). In addition to this, we propose to compare the registration errors obtained using FSR, FGR [40], SVD, and RANSAC with different numbers of sampled correspondences by collecting 250, 500, 1000, 2500 and 5000 best similarity scores. The results reported in Table 5 show that FSR outperforms the other methods in rotation and translation, except in the case of 2500 samples where it ranks second (in translation) behind RANSAC. It can be highlighted that FSR surpasses RANSAC-50k, regardless of the number of samples. It can even outperform FGR in processing time (t) when using fewer than 1,000 samples. Some performed registrations are depicted

in Figure 5.

Table 5: Registration performances on KITTI using different pose estimators (RTE (cm), RRE (degrees), and t(ms).

#samples	5000			2500			1000			500			250		
Error	RTE(↓)	RRE(↓)	t(ms)	RTE(↓)	RRE(↓)	t(ms)	RTE(↓)	RRE(↓)	t(ms)	RTE(↓)	RRE(↓)	t(ms)	RTE(↓)	RRE(↓)	t(ms)
RSAC5k	8.3	0.31	95	8.3	0.30	88	8.4	0.30	50	8.8	0.33	39	8.8	0.33	24
RSAC10k	8.1	0.29	185	7.8	0.30	171	8.3	0.29	98	8.7	0.31	77	9.0	0.33	48
RSAC50k	7.6	**0.27**	906	7.8	**0.27**	838	8.1	0.30	548	8.6	0.31	548	8.9	0.33	234
FGR	9.4	0.38	56	9.4	0.38	55	8.4	0.35	53	8.8	0.36	52	9.4	0.40	51
SVD	13.1	0.39	11	12.7	0.38	11	11.1	0.36	11	11.1	0.38	11	11.4	0.42	11
FSR (ours)	**7.3**	**0.27**	171	**7.4**	0.28	143	**7.6**	**0.28**	50	**7.9**	**0.29**	37	**8.2**	**0.30**	27

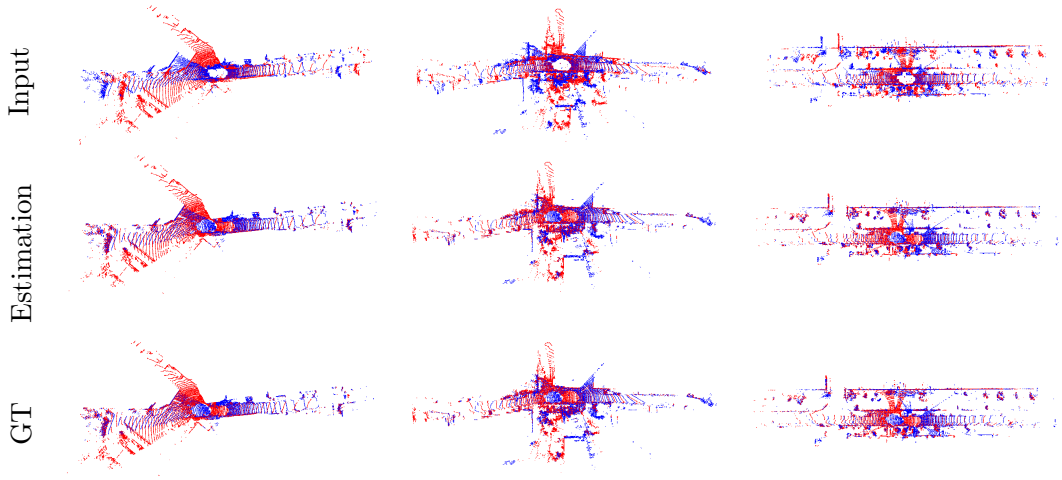


Figure 5: Illustration of some performed registrations on KITTI.

4.5. 3DMatch Indoor Scenes Dataset

We evaluate the performance of correspondence prediction using the same metrics employed in in [18] and [7], i.e., Inlier Ratio (IR), which calculates the ratio of matched points with a distance smaller than $\tau_1 = 10\ cm$ under the ground truth transformation; Feature Matching Recall (FMR), which represents the ratio of point cloud pairs with an IR above 5%. To assess the

precision of the estimated transformation, we use three metrics: Registration Recall (RR); Relative Translation Error (RTE); and Relative Rotation Error (RRE).

Table 6: Matching recall performances on 3DMatch *versus* number of correspondences.

# samples	5000	2500	1000	500	250
Feature Match Recall (%)(↑)					
SpinNet [1]	97.6	97.2	96.8	95.5	94.3
Predator [7]	96.6	96.6	96.5	96.3	96.5
YOHO [26]	**98.2**	97.6	97.5	97.7	96.0
CoFiNet [34]	<u>98.1</u>	**98.3**	<u>98.1</u>	**98.2**	**98.3**
GeoTransformer [18]	97.9	97.9	97.9	<u>97.9</u>	97.6
Ours	**98.2**	<u>98.2</u>	**98.2**	**98.2**	<u>98.0</u>

Table 6 displays the FMR results obtained using different sampled correspondence sizes, where we select the top-k scores from the complete set of paired points, similar to the approach in [18]. RoCNet++ surpasses Geotransformer [18] and is second among all the methods and all cases. It matches the best scores when using 5000 and 500 points while performing best with 1000 sampled points and second with 2500 and 250 points. This suggests that RoCNet++ excels in challenging cases, consistently achieving a minimum of 5% inliers across a broader range of test examples. Inlier Ratio results are presented in Table 7 demonstrating that RoCNet++ produces accurate correspondences, as it consistently ranks second, following Geotransformer [18], except for the sampling of 1000 points, where it achieves the highest scores.

For the registration results, Table 8 demonstrates the efficiency of FSR, which achieves the highest registration recall with fewer sampled correspondences (i.e., 250 and 500), while ranking as the second fastest method, just

behind SVD. However, for larger sets of correspondences (i.e., 1000, 2500, and 5000), FSR is outperformed only by RANSAC-50k, which is approximately four times slower than our method. Finally, Table 9 shows that in terms of RRE and RTE, RoCNet++ achieves results comparable to Geotransformer [18], which is the best method across all metrics. RoCNet++ is narrowly outperformed by CoFiNet [34] in RR.

Table 7: Inlier Ratio results on 3DMatch *versus* number of correspondences.

#samples	5000	2500	1000	500	250
Inlier Ratio (%)(↑)					
SpinNet [1]	47.5	44.7	39.4	33.9	27.6
Predator [7]	58.0	58.4	57.1	54.1	49.3
YOHO [26]	64.4	60.7	55.7	46.4	41.2
CoFiNet [34]	49.8	51.2	51.9	52.2	52.2
GeoTransformer [18]	**71.9**	**75.2**	76.0	**82.2**	**85.1**
Ours	68.2	68.2	**79.4**	77.9	80.8

Table 8: Comparison of different pose estimators with our pipeline on 3DMatch

# samples	5000		2500		1000		500		250	
	RR(↑)	t**(ms)**(↓)	RR(↑)	t**(ms)**(↓)	RR(↑)	t**(ms)**(↓)	RR(↑)	t**(ms)**(↓)	RR(↑)	t**(ms)**(↓)
SVD	77.6	7	77.6	7	77.6	7	79.1	5	79.2	5
FGR	85.1	107	84.5	101	85.3	101	86.4	096	85.9	089
RANSAC-50k	**90.2**	560	**90.4**	550	**90.2**	506	89.6	339	**88.9**	235
FSR (ours)	89.1	150	88.8	140	88.8	133	**89.8**	086	**88.9**	086

Table 9: Registration errors obtained on 3DMatch using all the correspondences. RRE is in degrees, RTE is in centimeters and RR is in %

Method	RR(↑)	RRE(↓)	RTE(↓)
Predator [7]	89.0	2.029	0.064
CoFiNet [34]	89.3	2.011	0.062
Geotransf. [18]	**91.5**	**1.625**	**0.053**
Ours	89.1	1.660	**0.053**

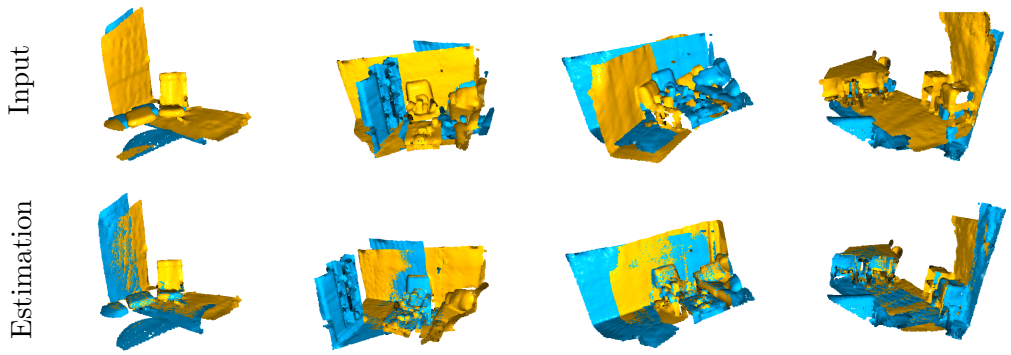


Figure 6: Illustration of some examples of performed registrations on 3DMatch

5. ABLATION STUDY

In this ablation study, we show the improvements provided by the introduction of the triangle-based descriptor, by the weighting of the triangles and, finally, by the FSR method introduced.

5.1. Triangle-based Descriptor Ablation

Table 10, presents the matching performance of RoCNet++ in comparison to another version where the descriptor is replaced by FPFH [20]. Additionally, we provide an analysis of RoCNet++ without the triangle-based descriptor and a version that combines FPFH and triangles. This analysis allows highlighting the impact of the triangles and its ease of incorporation into other descriptors. This analysis demonstrates that the use of triangles can clearly improve performances in both clean and noisy data. For example, the three metrics precision, accuracy and recall exceed the 99% in clean data.

Table 10: Matching performances on ModelNet40 with partial overlap (**P**: Precision, **A**: Accuracy, **R**: Recall)

Descriptor	clean data			noisy data		
	P(↑)	**A**(↑)	**R**(↑)	**P**(↑)	**A**(↑)	**R**(↑)
FPFH	92.1	84.6	88.3	72.1	71.0	71.2
FPFH + Triangles	<u>99.3</u>	<u>98.6</u>	<u>98.9</u>	82.6	81.8	81.5
DGCNN	98.0	97.2	97.6	<u>85.8</u>	<u>85.9</u>	<u>85.5</u>
DGCNN+Triangles (ours)	**99.9**	**99.8**	**99.8**	**89.4**	**89.4**	**89.2**

5.2. Triangle Weighting Ablation

For a better understanding of the motivation behind the use of different weights for every single triangle, we provide a comparison between RoC-Net++ and the modified version on which the $softmax$ is removed to let the same contribution for all triangles in the predicted descriptor. Table 11 highlights the important role played by the area weighting of the triangles in the presence of noise since the performance in all the matching metrics is improved. It is worth noting that in the case of clean data, the use of $softmax$ has no huge impact on the matching performances which are already very high.

Table 11: Matching performances on ModelNet40 with partial overlapping input, where RoCNet++$^{(\dagger)}$ is a version of the method without weighting of triangles.

Method	clean data			noisy data		
	P(↑)	**A**(↑)	**R**(↑)	**P**(↑)	**A**(↑)	**R**(↑)
RoCNet++$^{(\dagger)}$	**99.9**	99.7	99.7	86.0	85.9	85.5
RoCNet++	**99.9**	**99.8**	**99.8**	**89.4**	**89.4**	**89.2**

5.3. Influence of the Number of Triangles in the Descriptor

In this experiment, we study the influence of the number of K nearest neighbours used to construct the triangle-based descriptor. Let us notice that the number of neighbours imposes the number of triangles that is $K(K-1)/2$. The analysis demonstrates that the best model for matching challenges is obtained for $K = 12$. We therefore kept this configuration throughout our studies.

5.4. Farthest Sampling-guided Registration (FSR) ablation

Finally, to validate the FSR algorithm effectiveness, we compare the registration results obtained using RANSAC, SVD and FSR for the same trained model on clean and noisy point clouds. Table 12 shows that FSR guarantees the best compromise for both cases: while RANSAC provides the best results on clean data but is less efficient in the presence of noise, SVD behaves oppositely, being efficient on noisy data but having surprising limits on clean data. The proposed Farthest Sampling-guided Registration (FSR) allows good performances in both cases by providing at least the second-best score in each metric.

Table 12: Registration performances on ModelNet40 using RANSAC with 500 iterations, SVD and our FSR on partial overlapping data.

Method	clean data				noisy data			
	rotation		translation		rotation		translation	
	RMSE ($\mathbf{R}$)(↓)	MAE($\mathbf{R}$)(↓)	RMSE($\mathbf{t}$)(↓)	MAE($\mathbf{t}$)(↓)	RMSE($\mathbf{R}$)(↓)	MAE($\mathbf{R}$)(↓)	RMSE($\mathbf{t}$)(↓)	MAE($\mathbf{t}$)(↓)
RANSAC	**<0.001**	**<0.001**	**0.0001**	**0.00002**	1.513	0.633	0.004	0.003
SVD	3.068	0.100	0.001	0.0001	**1.268**	<u>0.349</u>	**0.002**	**0.001**
FSR (ours)	<u>0.035</u>	<u>0.061</u>	<u>0.0003</u>	<u>0.00006</u>	<u>1.278</u>	**0.318**	**0.002**	**0.001**

6. Conclusion and Future Work

This paper discussed a new method for point cloud registration. The proposed architecture is inspired by our recent RoCNet method, to which we have made two significant improvements. The first contribution concerns the setting up of a robust geometric descriptor encoding the 3D point as triangles formed by the K nearest neighbours. The extracted features correspond to the angles formed by each triangle, which are invariant to rigid transformations. These angles combined with the surface normals form the final descriptor. The second contribution of this paper refers to the estimation of the rigid transformation. To improve the accuracy of the estimated transformation, we introduced the Farthest Sampling-guided Registration (FSR) which allows estimating several rigid transformations for each subset of matched points obtained thanks to a furthest points sampling method. Then, the global transformation is selected to be the one that yields the highest number of inliers.

The developed method was assessed using three well-established datasets, i.e., ModelNet40 (synthetic objects), KITTI (odometry), and 3DMatch (indoor scenes). Different scenarios have been set up to evaluate the performance of the method in the context of the state of the art: clean or noisy data under different overlap ratios. For the ModelNet40 data, RoCNet++ outperformed all the state-of-the-art methods in the case of clean data and in most cases of unfavourable conditions (e.g. noisy data). Finally, our method also performed well for the KITTI dataset, which is known to be challenging, improving 2 out of 3 metrics.

Although the proposed method has proved to be efficient for clean data,

noisy data degrades performance significantly. It would be interesting to work on a new variant of the descriptor able to be more robust to noise. For instance, it would be interesting to replace the normals used in RocNet++ with EGI (Extended Gaussian Image), which have proved to be more robust to noise than simple surface normals. One other limitation is the complexity of handling large-scale databases such as 3DMatch due in particular to the use of transformers in the architecture. To remedy this, we proposed to adopt a coarse-to-fine matching strategy using a voxel-grid downsampling which may lead to the same computation cost problem depending on the size of the input point clouds. Hence, a significant reduction in the resolution of the point cloud, if required to address this issue, can lead to performance degradation as it may result in the loss of relevant local geometric information.

In future work, we plan to investigate new effective downsampling strategies and extend our approach to cross-modality (e.g., 2D-3D) registration and non-rigid correspondence estimation. Also, it would be interesting to consider the triangle-based descriptor in other applications such as segmentation and classification of points-clouds.

7. Acknowledgements

This work was supported by the French ANR program MARSurg (ANR-21-CE19-0026).

References

[1] Ao, S., Hu, Q., Yang, B., et al., 2021. Spinnet: Learning a general surface descriptor for 3d point cloud registration, in: Conf. Comput.

Vision Pattern Recognit., pp. 11753–11762.

[2] Bai, X., Luo, Z., Zhou, L., et al., 2021. Pointdsc: Robust point cloud registration using deep spatial consistency, in: EEE/CVF Conf. on Comput. Vision and Pattern Recognit., pp. 15859–15869.

[3] Besl, P.J., McKay, N.D., 1992. Method for registration of 3-d shapes, in: Sensor fusion IV: control paradigms and data structures, pp. 586–606.

[4] Eldar, Y., Lindenbaum, M., Porat, M., et al., 1997. The farthest point strategy for progressive image sampling. IEEE Trans. on Image Process. 6, 1305–1315.

[5] Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395.

[6] Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite, in: Conf. Comput. Vision Pattern Recognit., pp. 3354–3361.

[7] Huang, S., Gojcic, Z., Usvyatsov, M., et al., 2021. Predator: Registration of 3d point clouds with low overlap, in: IEEE/CVF Conf. Comput. Vision Pattern Recognit., pp. 4267–4276.

[8] Huang, X., Mei, G., Zhang, J., 2020. Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences, in: Conf. Comput. Vision Pattern Recognit., pp. 11366–11374.

[9] Kadam, P., Zhang, M., Liu, S., et al., 2022. R-pointhop: A green, accurate, and unsupervised point cloud registration method. IEEE Trans. on Ima. Process. 31, 2710–2725.

[10] Kinga, D., Adam, J.B., et al., 2015. A method for stochastic optimization, in: Int. Conf. on Learn. Repres., p. 6.

[11] Li, L., Fu, H., Ovsjanikov, M., 2022. Wsdesc: Weakly supervised 3d local descriptor learning for point cloud registration. IEEE Trans. Vis. Comput. Graph. .

[12] Li, Q., Wang, C., Wen, C., et al., 2023. Deepsir: Deep semantic iterative registration for lidar point clouds. Pattern Recognit. 137, 109306.

[13] Li, Y., Harada, T., 2022. Lepard: Learning partial point cloud matching in rigid and deformable scenes, in: Conf. Comput. Vision Pattern Recognit., pp. 5554–5564.

[14] Liu, X., Li, A., Sun, J., Lu, Z., 2023. Trigonometric projection statistics histograms for 3d local feature representation and shape description. Pattern Recogn. , 109727.

[15] Papadopoulo, T., Lourakis, M.I., 2000. Estimating the jacobian of the singular value decomposition: Theory and applications, in: Europ. Conf. on Comput. Vision, pp. 554–570.

[16] Qi, C.R., Su, H., et al., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Conf. Comput. Vision Pattern Recognit., pp. 652–660.

[17] Qi, C.R., Yi, L., Su, H., et al., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Adv. Neural. Inf. Process. Syst. 30.

[18] Qin, Z., Yu, H., Wang, C., et al., 2022. Geometric transformer for fast and robust point cloud registration, in: IEEE/CV Conf. Comput. Vision Pattern Recognit., pp. 11143–11152.

[19] Ran, H., Liu, J., Wang, C., 2022. Surface representation for point clouds, in: Conf. Comput. Vision Pattern Recognit., pp. 18942–18952.

[20] Rusu, R.B., Blodow, N., et al., 2009. Fast point feature histograms (fpfh) for 3d registration, in: IEEE Int. Conf. on Rob. and Auto., pp. 3212–3217.

[21] Shi, C., Chen, X., Huang, K., et al., 2021. Keypoint matching for point cloud registration using multiplex dynamic graph attention networks. IEEE Rob. and Auto. Let. 6, 8221–8228.

[22] Sinkhorn, R., Knopp, P., 1967. Concerning nonnegative matrices and doubly stochastic matrices. Pacific J. of Math. 21, 343–348.

[23] Slimani, K., Tamadazte, B., Achard, C., 2023. Rocnet: 3d robust registration of point-clouds using deep learning. arXiv preprint arXiv:2303.07963 .

[24] Thomas, H., Qi, C.R., Deschaud, J.E., et al., 2019. Kpconv: Flexible and deformable convolution for point clouds, in: IEEE/CVF Int. Conf. on Comput. vision, pp. 6411–6420.

[25] Vaswani, A., Shazeer, N., et al., 2017. Attention is all you need. Adv. Neural. Inf. Process. Syst. 30.

[26] Wang, H., Liu, Y., et al., 2022. You only hypothesize once: Point cloud registration with rotation-equivariant descriptors, in: ACM Int. Conf. on Multimedia, pp. 1630–1641.

[27] Wang, Y., Solomon, J.M., 2019a. Deep closest point: Learning representations for point cloud registration, in: IEEE Int. Conf. on Comput. Vision.

[28] Wang, Y., Solomon, J.M., 2019b. Prnet: Self-supervised learning for partial-to-partial registration. Adv. Neural. Inf. Process. Syst. 32.

[29] Wang, Y., Sun, Y., et al., 2019. Dynamic graph cnn for learning on point clouds. ACM Trans. on Grap. .

[30] Wu, Y., Hu, X., Zhang, Y., et al., 2023. Sacf-net: Skip-attention based correspondence filtering network for point cloud registration. IEEE Trans. Circuits Syst. Video Technol. .

[31] Wu, Z., Song, S., Khosla, A., et al., 2015. 3d shapenets: A deep representation for volumetric shapes, in: Conf. Comput. Vision Pattern Recognit., pp. 1912–1920.

[32] Xiao, C., Wachs, J., 2021. Triangle-net: Towards robustness in point cloud learning, in: IEEE/CVF Winter Conf. on Appl. of Comp. Vis., pp. 826–835.

[33] Yang, J., Li, H., Campbell, D., et al., 2015. Go-icp: A globally optimal solution to 3d icp point-set registration. EEE Trans. Pattern Anal. Mach. Intell. 38, 2241–2254.

[34] Yu, H., Li, F., Saleh, M., et al., 2021. Cofinet: Reliable coarse-to-fine correspondences for robust pointcloud registration. Adv. in Neur. Info. Proces. Sys. 34, 23872–23884.

[35] Zeng, A., Song, S., Nießner, M., et al., 2017. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions, in: Conf. Comput. Vision Pattern Recognit., pp. 1802–1811.

[36] Zhang, J., Yao, Y., Deng, B., 2021. Fast and robust iterative closest point. IEEE Trans. on Pattern Anal. and Mach. Intell. 44, 3450–3466.

[37] Zhang, Z., Sun, J., Dai, Y., et al., 2022a. Vrnet: Learning the rectified virtual corresponding points for 3d point cloud registration. IEEE Trans. on Cir. and Sys. for Video Tech. 32, 4997–5010.

[38] Zhang, Z., Sun, J., et al., 2022b. Self-supervised rigid transformation equivariance for accurate 3d point cloud registration. Pattern Recognit. 130, 108784.

[39] Zhao, C., Yang, J., et al., 2022. Rotation invariant point cloud analysis: Where local geometry meets global topology. Pattern Recognit. 127, 108626.

[40] Zhou, Q., Park, J., et al., 2016. Fast global registration, in: Eur. Conf. Comput. Vis., pp. 694–711.