Ph.D. Dissertation

Enhancing 3D Point Cloud Analysis using Aligned Kernels

정렬 커널 기반 포인트 클라우드 분석 개선

2023 년 8 월

Department of Computer Science and Engineering
The Graduate School
Seoul National University

Seunghwan Jung

Enhancing 3D Point Cloud Analysis using Aligned Kernels

지도교수 신 영 길

이 논문을 공학박사 학위논문으로 제출함

2023 년 5 월

서울대학교 대학원

컴퓨터공학부

정 승 환

정승환의 공학박사 학위논문을 인준함

2023 년 6 월

위 원 장 염 헌 영 (인)

부위원장 신 영 길 (인)

위 원 엄 현 상 (인)

위 원 오 경 수 (인)

위 원 정 민 영 (인)

Abstract

Point cloud analysis is becoming a popular research area due to the growth in the capability of 3D sensors, which can capture detailed 3D geometric information. One of the fundamental tasks in point cloud analysis is to extract salient geometric information from unstructured and irregularly distributed point sets for use in various applications. While many works based on deep learning have been developed to process 3D point clouds, a common drawback is the lack of consideration for the rotation invariance property, resulting in poor generalization performance under rotational variations of the point cloud.

In this dissertation, a new method for analyzing point clouds in a rotation-invariant manner is presented to address this problem. The main idea is to place aligned and structured additional points (i.e., kernels) around each point, which are used to extract local geometric information from unstructured point sets. By aligning the kernel points based on the point distribution, the proposed method can extract consistent local geometric information under rotational variations. To improve scale-robustness, the optimal kernel size is determined through analysis of various sizes of kernels. For the registration task, differences between the extracted information (i.e., descriptors) are estimated to select and use only discriminative descriptors.

The proposed method was tested on benchmark and real-world datasets to evaluate its performance in registration, classification, and segmentation of 3D point clouds. The results showed that the proposed method outper-

formed state-of-the-art methods in the registration task and also performed better in the classification and part-segmentation tasks under random rotation environments.

Keywords: Point cloud classification, point cloud registration, point cloud segmentation, rotation-invariant point descriptor

Student Number: 2020-35803

Contents

Chapter 1 Introduction **1**

1.1 Background and motivation 1

1.2 Problem statement 3

1.3 Main contributions 5

1.4 Contents and organization 7

Chapter 2 Preliminary **9**

2.1 Point cloud 9

2.2 Local reference frame 13

Chapter 3 Related Works **15**

3.1 Overview 15

3.2 Hand-crafted method 15

3.2.1 Overview 15

3.2.2 LRF based method 16

3.2.3 Non-LRF based method 20

3.3 Deep learning based rotation-variant method 22

3.3.1 Overview 22

3.3.2 Volumetric based method 22
3.3.3 Multi-view based method 25
3.3.4 Point based method 26
3.4 Deep learning based rotation-invariant method 31
3.4.1 Overview 31
3.4.2 LRF based alignment 31
3.4.3 Rotation-invariant feature extraction 33

Chapter 4 Aligned Kernel based Rotation-Invariant Method 37
4.1 Overview 37
4.2 Descriptor generation 39
4.2.1 Kernel alignment 40
4.2.2 Rotation robust feature projection 42
4.2.3 Circular convolution 44
4.2.4 Global context aggregation 47
4.2.5 Scale adaptation module 50
4.3 CNN encoding 51
4.4 Experimental results 54
4.4.1 Overview 54
4.4.2 Data configurations 54
4.4.3 Evaluation metric 55
4.4.4 Quantitative analysis 57
4.4.5 Parameter and ablation study 70
4.5 Discussion 77

Chapter 5 Applying Aligned Kernels for Dense Point Cloud Registration in Real-World Scenarios 79

5.1 Overview 79
5.2 Descriptor generation 82
5.2.1 Local patch alignment 83
5.2.2 Rotation-invariant feature extraction and encoding . 83
5.3 Salient descriptor selection 85
5.4 Experimental results 87
5.4.1 Overview 87
5.4.2 Data configurations 87
5.4.3 Evaluation metric 90
5.4.4 Quantitative analysis 90
5.4.5 Test on indoor 3DMatch dataset 90
5.4.6 Test on outdoor ETH dataset 97
5.4.7 Ablation study 104
5.5 Discussion 107

Chapter 6 Conculsion and Future Works **110**

Bibliography **112**

초록 **127**

List of Figures

Figure 1.1 Point cloud analysis using techniques such as classification, segmentation, and registration enables diverse applications such as autonomous driving, 3D reconstruction, and augmented reality. 2

Figure 1.2 For given naive rotation-invariant features to represent the neighboring point x from the point p and m (i.e., $f_{p,m}(x)$), there are multiple corresponding x locations like a green circle. 5

Figure 2.1 Two N point sets, which are sorted without specific order, represent the same point cloud. 10

Figure 2.2 (a) Input point cloud and interest region (dotted box). (b) local reference frame (LRF) of the region. (c) Aligned region based on the LRF. (d) Rotational and normal sign variations due to the incorrect LRF. 13

Figure 3.1 (a) Two kinds of descriptors: signatures and histograms; (b) SHOT descriptor [1]. 16

Figure 3.2 (a) RGB-D image; (b) Range image of the interest point; (c) NARF descriptor [2]. 18
Figure 3.3 (a) PFH descriptor [3] and (b) FPFH descriptor [4]. 19
Figure 3.4 (a) Geometric relationship between the keypoint p and its neighbor point x; (b) SPIN descriptors (spin-images) [5] for three keypoints. 21
Figure 3.5 Taxonomy of deep learning based approaches. . . . 23
Figure 3.6 (a) Typical 2D image convolution and (b) 3D grid convolution of VoxNet [6]. 24
Figure 3.7 Network architecture of MVCNN [7] for 3D shape recognition. 25
Figure 3.8 Network architecture of PointNet [8]. N and D denote the number of input points and output dimension. 27
Figure 3.9 Graph construction and encoding operation (EdgeConv) of DGCNN [9]. The output of EdgeConv is calculated by aggregating the edge features associated with all the edges emanating from each connected vertex. 29
Figure 3.10 (a) Typical volumetric-based approaches. (b) Kernels of PointwiseCNN [10]. For each point, nearest neighbors are queried and binned into kernel cells before encoding. 30
Figure 3.11 Voxelization process of 3DsmoothNet [11] using the LRF. 31
Figure 3.12 Local coordinates (θ_k, ϕ_k, ρ_k) for the camera viewpoint c_k in the LRF of p. 32

Figure 3.13 Network architecture of REQNN [12]. Both input point clouds and intermediate-layer features are represented by quaternion features. The quaternion feature $g(R \cdot x \cdot \bar{R})$ generated from the point cloud rotated by a specific angle is equivalent to the rotated quaternion feature $R \cdot g(x) \cdot \bar{R}$ generated from the original point cloud. 34

Figure 3.14 Global and local rotation-invariant representations of RIF [13]. Two kinds of additional reference points (m_i and s_i) are used to represent the neighbors of p_i. 35

Figure 4.1 Overview of the proposed architecture. First, features are extracted using multiple kernel sizes. Subsequently, scale analysis is employed based on the interpolation between the kernel sizes. Finally, the descriptor is encoded using the adjusted scale for the downstream tasks. 39

Figure 4.2 (a) Kernel points are aligned using the LRF of the target point; (b) Neighbors are selected for each kernel point; (c) Weighted-average location is estimated based on the distance from the kernel point to the neighbors; (d) For each kernel, the relative location of the averaged neighbor is estimated using distances and angles; (e) After convolution, kernel features are aggregated by summation and maximum value selection to represent the interest point. 41

Figure 4.3 Visualization of normal vectors. The signs for each point on a planar surface are not determined uniquely. 42
Figure 4.4 Green regions of (a), (b), (c), and (d) illustrate the possible locations of the neighbor point of the interest point when (f1), (f1, f2), (f1, f2, f3), and (f1, f2, f3, f4) are given, respectively. 44
Figure 4.5 (a) Channel-wise convolution and (b) circular convolution. The red transparent region indicates the receptive field. S indicates the kernel size of the convolution. 45
Figure 4.6 Same point cloud, different normal vector signs. Regardless of the normal vector sign, the kernel point located at "1" in (a) ("4" in (b)) is processed together with the kernel point located at "2" in (a) ("3" in (b)). In addition, the kernel weight applied to the kernel "1" in (a) and kernel "4" in (b) is the same by using the symmetric convolution method. 46
Figure 4.7 Left: two descriptors obtained from repeated area have same values because local descriptors only have the limited range of geometric information around the interest points. Right: two global descriptors obtained by aggregating all the descriptors have different values. Gray circles indicate the range of encoded geometric information of descriptors. By merging local descriptors and global descriptors, two descriptors have different values. 47

Figure 4.8 Colors of point cloud indicate global context weights of the interest point. Red indicates a weight value of one and blue indicates a weight value of zero. . . . 48
Figure 4.9 Multiple features are extracted using multiple kernel sizes (α and β). By applying simple convolution operations to these features, an interpolation weight W between the two kernel sizes is estimated. The final kernel size is then determined by applying the interpolation weight to the original kernel sizes. . . 49
Figure 4.10 Network architectures for (a) registration, (b) classification, and (c) part segmentation. 52
Figure 4.11 Left: source (blue) and target (red) point clouds, Middle: registration results of deep closest point method [14]. Right: registration results of the proposed method. Green indicates the transformed source point clouds. 58
Figure 4.12 Left: source points; Right: part-segmentation results 65
Figure 4.13 Comparison of methods 72

Figure 5.1 Overview of the descriptor generation. (a) Given an interest point, the LRF is first estimated from its neighbor points (patch). The local patch is then aligned using the LRF. (b-c) Rotation-invariant features are computed for each kernel. (d) The extracted kernel features are encoded using the symmetric circular convolution in a rotation-invariant manner. Consequently, the descriptor for the local patch is generated. N is the number of interest points, and D is the dimension of the descriptor. (e) The network architecture encodes extracted features. . . 82

Figure 5.2 Salient descriptor selection process using dissimilarities. (a) Given point clouds, descriptors are generated. (b) For each point cloud, pairwise dissimilarities are estimated using the descriptors. (c) For each point (that is, each row of the dissimilarity matrix), dissimilarity intensity is estimated by calculating L2-norm for each row vector. (d) Salient descriptors, which have high-intensity values (that is, strong dissimilarities between points), are selected for RANSAC-based registration. 85

Figure 5.3 (a) Visualization of fragments and a reconstruction of an apartment from SUN3D [15]. (b) Visualization of several RGB-D reconstructions used to train the network. 87

Figure 5.4 (a, d) Environment Topologies of gazebo and wood. (b, e) Contextual Photographs of gazebo and wood. (c, f) Point cloud views gazebo and wood. 89
Figure 5.5 Registration results of the proposed method on the indoor 3DMatch dataset. 92
Figure 5.6 Registration results of the proposed method on the indoor 3DMatch dataset. 93
Figure 5.7 Registration results of the proposed method on the indoor 3DMatch dataset. 94
Figure 5.8 Registration results of the proposed method on the indoor 3DMatch dataset. 95
Figure 5.9 Registration results of the proposed method on the outdoor ETH dataset. 99
Figure 5.10 Registration results of the proposed method on the outdoor ETH dataset. 100
Figure 5.11 Registration results of DIP [16], LMVD [17], SpinNet [18], and the proposed method on the outdoor ETH dataset. The dotted box indicates the characteristic overlap area. The arrow indicates the distance between the source and target point clouds. . 101
Figure 5.12 Registration results of DIP [16], LMVD [17], SpinNet [18], and the proposed method on the outdoor ETH dataset. The dotted box indicates the characteristic overlap area. The arrow indicates the distance between the source and target point clouds. . 102

Figure 5.13 Registration results of the proposed method on the indoor 3DMatch dataset. (a) All points are colored yellow (source) and blue (target). (b) Only salient points obtained by using the selection method are colored yellow and blue. 105

Figure 5.14 Registration results of the proposed method on the outdoor ETH dataset with different τ_S of Section 5.3 (that is, τ_S percentage of similar points are excluded). The horizontal axis and vertical axis indicate τ_S and FMR score. 108

List of Tables

Table 4.1 Registration results with randomly rotated same point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-). C indicates the number of used classes for training. 57

Table 4.2 Registration results with randomly rotated noisy point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-). 59

Table 4.3 Registration results with randomly rotated and partially sampled point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-). . . 60

Table 4.4 ModelNet40 classification results (i.e. OA) and ShapeNet-Part segmentation results (i.e. mIoU). OA, mIoU, and mcIoU indicate the overall accuracy, instance average intersection over union, and class average intersection over union, respectively. RI indicates whether the method is the rotation-robust method or not. . . 63
Table 4.5 Results of ModelNet40 classification. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively. 64
Table 4.6 Results of ScanObjectNN classification. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively. 66
Table 4.7 Results of ShapeNetPart segmentation. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively. 67
Table 4.8 Convolution operation study for ModelNet40 global registration task. 71
Table 4.9 The number of nearest neighbors study for ModelNet40 global registration task. 72
Table 4.10 Global context study for ModelNet40 global registration task. 74
Table 4.11 Global context study with the different number of used points. 74

Table 4.12 ModelNet40 global registration results with different scales. (#) is the experiments in which networks are trained with the original scale of point clouds and tested with (#) scale of point clouds. Values indicate the mean and standard deviation of the results. . . . 75

Table 4.13 Model statistics on ModelNet40 dataset. The models were experimented with a GTX 1080. 77

Table 5.1 Evaluation results for the indoor 3DMatch dataset. The evaluation metrics are Feature-matching recall (FMR) and standard deviation of FMR (std). H and D Type indicate handcrafted and deep learning-based methods, respectively. Feat dim is the descriptor dimension. Saliency Selection indicates whether the method uses salient and reliable descriptor selection. R-I (Rotation-Invariance) indicates whether the method uses the LRF or rotation-invariant features. Results are divided according to the Type (H or D) and R-I values. 91

Table 5.2 Feature-matching recall (FMR) scores on the outdoor ETH dataset. The network is trained on the 3DMatch dataset and tested on the ETH dataset. H and D Type indicate handcrafted and deep learning-based methods, respectively. Saliency Selection indicates whether the method uses salient and reliable descriptor selection. R-I (Rotation-Invariance) indicates whether the method uses the LRF or rotation-invariant features. Results are divided according to the Type (H or D) and R-I values. 98

Table 5.3 Feature-matching recall (FMR) scores of the ablation study of the salient descriptor selection method on the 3DMatch dataset. 104

Table 5.4 Feature-matching recall (FMR) scores of the ablation study of the salient descriptor selection method on the ETH dataset. 104

Table 5.5 Feature-matching recall (FMR) scores of the ablation study on the 3DMatch dataset with the different number of sampled points. 107

Chapter 1

Introduction

1.1 Background and motivation

Over the past few years, there has been rapid development in 3D acquisition technologies, making various 3D sensors, including LiDAR sensors in autonomous vehicles, RGB-D cameras in Kinect, and 3D scanners for reconstruction tasks, more accessible and affordable. These sensors provide vast amounts of 3D data, which offers an opportunity to gain a better understanding of the surrounding environment by providing rich geometric and shape information. This information can be applied to various fields such as robotics [19–21], augmented reality [22, 23], and autonomous vehicles [24, 25] (Fig. 1.1).

There are four types of formats used to represent 3D data, namely depth images, point clouds, meshes, and volumetric grids. Among these formats, point cloud representation is the closest 3D representation to raw sensor

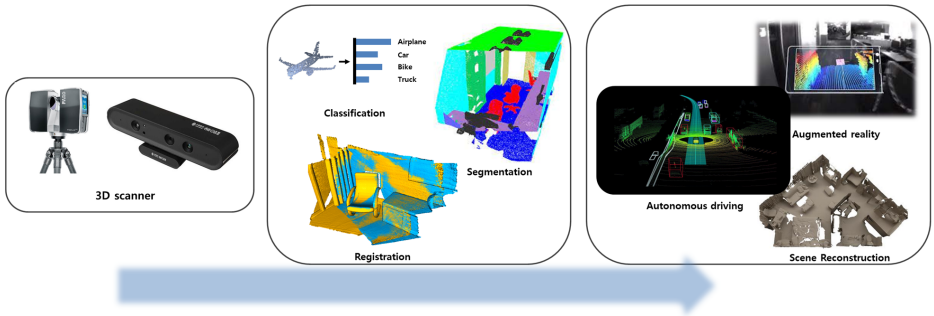

Figure 1.1: Point cloud analysis using techniques such as classification, segmentation, and registration enables diverse applications such as autonomous driving, 3D reconstruction, and augmented reality.

data, as it represents the object's surface as a collection of 3D coordinate points. This allows it to preserve the original geometric information in 3D space without any discretization. As a result, it is the most commonly used format for scene understanding related applications. Deep learning-based approaches for point cloud applications have recently been widely researched for many fields, including point cloud registration, model segmentation, and classification, and have outperformed earlier works, which were based on hand-crafted feature extraction approaches [1, 4].

Point cloud classification is a fundamental technique for understanding the collected scene data, including autonomous driving and scene perception in robotics. It is also a crucial step for further point cloud processing. **Point cloud segmentation** process partitions points based on their similarity, and the resulting point sets should be meaningful. The segmentation results can be used for further scene analysis, such as locating and recognizing objects, and classification. The objective of **point cloud reg-**

istration is to estimate a transformation between two point clouds, which plays a critical role in various computer vision applications. Real-world point cloud data generated by 3D sensors represents only partial geometric information since the sensors capture scenes with a limited view range. Therefore, the registration task is necessary to generate a complete 3D scene from the partial scenes. As point cloud analysis is essential for many computer vision applications, there is a growing need to develop effective and robust analysis methods to make them more efficient and reliable for use in various industries.

1.2 Problem statement

In point cloud processing, a descriptor is a set of compact values that represent the local geometric information of a 3D point cloud. While there are many methods for representing point clouds, simplistic representations such as object color or size may not capture the intricate details required for complex tasks like object classification or registration. To successfully tackle these challenges, it is crucial to extract general and robust local geometric patterns from point clouds.

Early approaches to point cloud processing relied on hand-crafted feature extraction to compute low-level features for capturing local geometric information [1,4]. However, these methods have limitations when processing noisy or occluded point cloud data [26], leading to inferior performance compared to recent deep learning-based approaches. In contrast, deep learning-based approaches compute features from a point cloud and then encode the features, leading to groundbreaking results in several vision tasks compared

to hand-crafted approaches. Nonetheless, achieving consistent performance with real-world point clouds can be challenging due to the misalignment of point clouds caused by rotation and translation.

To address this challenge, it is essential to achieve both rotation- and translation-invariance properties when obtaining descriptors from point clouds. These properties are critical for many applications, such as object recognition, segmentation, and registration, where point clouds may be captured from different angles or locations. Translation-invariance means that the method produces the same output regardless of the location of the point cloud in space. Most deep learning-based approaches typically use features that are invariant to translation, such as the relative coordinates of each point with respect to the point cloud center. Rotation-invariance means that the method produces the same output regardless of the rotational alignment of the point clouds being analyzed. Compared to the translation-invariance, achieving rotation-invariance is challenging. There are three approaches to achieve it, but each approach has a problem. One approach involves training a network using random rotation augmentation, but this approach can decrease network performance due to insufficient capacity to learn all randomly rotated inputs. Another approach is to use rotation-invariant features (e.g., relative distances and angles between points) as network inputs instead of using Cartesian coordinates [12,13,27,28]. However, representing geometric relationships between points uniquely with rotation-invariant features is difficult compared to non-rotation-invariant features. When using naive rotation-invariant features, there can be multiple possible point cloud shapes corresponding to the given features (Fig. 1.2), which can result in worse network performance compared to rotation-variant networks. A third

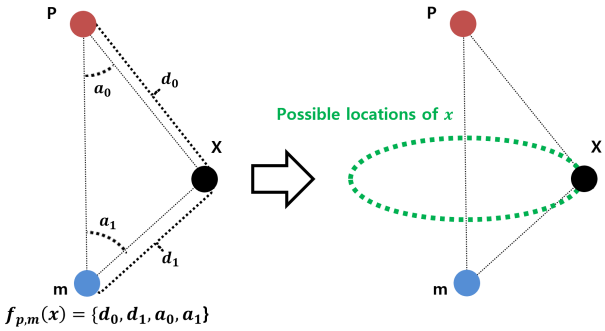

Figure 1.2: For given naive rotation-invariant features to represent the neighboring point x from the point p and m (i.e., $f_{p,m}(x)$), there are multiple corresponding x locations like a green circle.

approach to achieve rotation-invariance is to align point clouds based on point distributions to have the same orientation, but accurate alignment can be challenging due to the presence of noise and occlusions in the point clouds.

Despite these challenges, developing approaches that can achieve both rotation- and translation-invariance is critical for successful processing of real-world point clouds, opening up new opportunities for point cloud applications.

1.3 Main contributions

In this dissertation, a method is proposed for generating descriptors that perform accurate classification, segmentation, and registration tasks under rotation variations. To extract structured geometric information from an irregular and unstructured point cloud, aligned and structured addi-

tional points (i.e., kernels) are employed for each point while generating the descriptor. In the descriptor generation process, the method estimates point relationships (i.e., features) and encodes the features using the aligned kernel points to improve rotation-robustness. The kernel-based descriptor showed high benchmark performances with or without rotational variations. Besides, an additional scale analysis network and aggregation method are successfully embedded in the method to improve scale-robustness and descriptor performance, respectively. Secondly, a salient descriptor selection method is proposed for the registration task. Real-world large-scale registration tasks, where it is hard to find correspondences between source and target point clouds, are successfully performed by excluding non-salient descriptors from repeated and monotonous areas. A brief overview of the algorithms and achievements is described in the following paragraphs.

Generating rotation-invariant descriptors is a crucial and challenging task to achieve high performance in classification, segmentation, and registration. To overcome the challenges of ambiguous rotation-invariant features and unstable alignment, the proposed descriptor generation method uses an aligned kernel structure that places structured additional points around each point to capture its surrounding geometric information. The method uses these kernel points as references to accurately represent relationships between points with rotation-invariant features. It also addresses the issue of unstable alignment by utilizing a cylindrical kernel structure. Additionally, a scale analysis network is used to determine the optimal kernel size for improving scale-robustness. To enhance descriptor performance, a weighting-based descriptor aggregation method is employed to aggregate all descriptor values according to point distances. The proposed method

is evaluated against state-of-the-art methods, and the results show its superior performance. Parameter and ablation studies are also conducted to validate the proposed methods.

For the registration task with real-world large-scale point clouds, a dissimilarity-based salient descriptor selection method is proposed to exclude non-salient descriptors obtained from monotonous and repeating areas. Firstly, a fixed number of points (i.e., anchors) and their neighbor points (i.e., patches of the anchors) are sampled from all the points. Secondly, a descriptor is generated for each sampled patch using the aforementioned rotation-robust descriptor generation method. Finally, a dissimilarity-based salient descriptor selection method is applied to the generated descriptors for accurate registration. The proposed method analyzes dissimilarities between descriptors and excludes descriptors that are similar to the other descriptors (i.e., non-salient descriptors). Indoor and outdoor real-world point clouds are used for a parameter study and quantitative evaluation. The performance of the proposed method is evaluated by comparing it with other state-of-the-art methods, and the results show that the proposed method significantly improves the overall performance by excluding non-salient descriptors.

1.4 Contents and organization

The remainder of this dissertation is organized as follows. First, point cloud definition and properties are introduced. Then, related works are explored in chapter 3. In chapter 3, several hand-crafted and deep learning based methods for point cloud descriptors are illustrated. The proposed descrip-

tor method and several point cloud analysis tasks based on the method are described in chapters 4. The proposed method for the point cloud registration is described in chapters 5. Each chapter comprises an overview, detail methodology, corresponding experimental results, and discussion. The conclusion and future works are presented in chapter 6.

Chapter 2

Preliminary

2.1 Point cloud

Point clouds are widely used in various applications such as computer graphics, robotics, and computer vision. They are typically represented as a collection of points in a space. However, point clouds have specific characteristics and properties that make them different from other data types. In this section, the point cloud definitions and notations are introduced.

Point cloud definition

A point cloud is a collection of points in space. In this dissertation, a point cloud P in the 3D Euclidean space $\mathbb{R}^3$ is defined as:

$$P = \{x_i \in \mathbb{R}^3\}_{i<N} \tag{2.1}$$

Where N is the number of points in the cloud. The points in a point cloud are typically represented in 3D space, but they can also be represented in

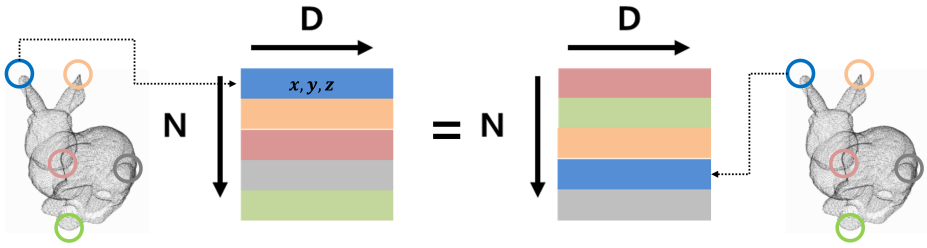

Figure 2.1: Two N point sets, which are sorted without specific order, represent the same point cloud.

lower or higher dimensional spaces. In addition to point coordinates, point clouds can contain more information, such as colors or any other types of features. Point clouds with additional features are much more descriptive than point clouds with only point coordinates, making them more suitable for deep learning algorithms.

Point cloud properties

Point clouds have specific properties that make them different from other data types. When designing point cloud processing algorithms, it is crucial to take these properties into account.

Unordered: Point clouds are unordered point sets, meaning that every operation on point clouds should be invariant to any permutation of points (Fig. 2.1). This property is essential because using non-symmetric functions to each point would lead to bad results.

Unstructured: Point clouds are unstructured point sets, meaning that there is no inherent connectivity or organization between points. This property makes it difficult to process and analyze the point cloud, as there is

no inherent structure to rely on.

Continuity: Point clouds are continuous data, which means that the point coordinates are not a countable set of locations. This property makes point clouds different from 2D image grids and makes it challenging to adapt 2D image convolution methods to point clouds.

Large data size: Point clouds can be very large, and this property makes it challenging to store and process the data. Limited storage capacity and slow processing speeds can make it difficult to work with large datasets.

Irregular distribution: The point density can be non-uniform, and the distribution can vary depending on the shape and reflectivity of the objects being scanned. This property makes it challenging to accurately analyze the data using algorithms that rely on the density or proximity of points.

When designing point cloud processing algorithms, it is crucial to take into account these properties. Having a clear understanding of these properties is essential to ensure that point cloud data is accurately processed and analyzed.

Point cloud descriptor

Point cloud descriptors are feature vectors that summarize the geometry and structure of a point cloud. In this dissertation, a set of descriptors D for a point cloud is defined as:

$$D = \{d_i \in \mathbb{R}^M\}_{i<N} \tag{2.2}$$

Where M is the dimension of the features.

These descriptors are essential for tasks such as classification, segmentation, and registration of point clouds. The primary goal of point cloud

descriptors is to encode the relevant information of the point cloud in a compact and invariant way. Invariant descriptors are essential for various applications as they can robustly represent the geometry and structure of a point cloud, even when it is subjected to variations in its orientation and position.

Translation invariance is a crucial property of point cloud descriptors, as it ensures that the same descriptor is obtained for two point clouds that are identical, except for their position in space. Since point clouds can be shifted or moved, their positions in space can vary, and this can affect their descriptors. Therefore, having translation-invariant descriptors is important to accurately represent the geometry of the point cloud regardless of its position. Rotation invariance is another important property of point cloud descriptors, which ensures that the same descriptor is obtained for a point cloud regardless of its orientation. Point clouds can be captured from different viewpoints, and their orientation can vary. Therefore, having rotation-invariant descriptors is essential to represent the geometry and structure of the point cloud independently of its orientation.

To process real-world data, it is important to have invariant descriptors that can represent the essential information of point clouds while being insensitive to variations in rotation and translation. These descriptors enable the processing of unaligned data and are essential for tasks such as classification, segmentation, and registration of point clouds in various applications.

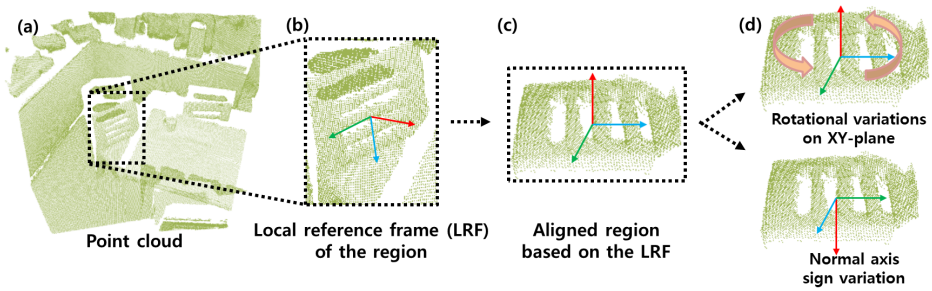

Figure 2.2: (a) Input point cloud and interest region (dotted box). (b) local reference frame (LRF) of the region. (c) Aligned region based on the LRF. (d) Rotational and normal sign variations due to the incorrect LRF.

2.2 Local reference frame

In point cloud processing, a local reference frame (LRF) refers to a canonical coordinate system that is defined at a specific point (i.e., keypoint or interest point) in the point cloud data. This coordinate system is used to describe the orientation of points relative to the specific point at which the LRF is defined. LRFs are commonly used in point cloud registration, which involves aligning two or more point clouds with different orientations and positions to form a coherent model (Fig. 2.2). By defining LRFs at specific points in each point cloud, it becomes possible to transform the point clouds to a common coordinate system and align them.

One common method for obtaining an LRF is to compute the principal component analysis (PCA) of the points in the local neighborhood of the specific point. To compute the PCA, the first step is to identify the k nearest neighbors of the specific point. Then, the covariance matrix of these

neighbors is calculated, which contains information about the distribution of the points in the neighborhood. The eigenvectors of the covariance matrix represent the principal axes of the point cloud in the neighborhood, and the eigenvalues correspond to the variances along each of these axes. The eigenvector with the smallest eigenvalue corresponds to the normal vector of the LRF, while the other two eigenvectors can be used to define the other two orthogonal axes of the LRF. The origin of the LRF is located at the specific point. Once the LRF is established, each point in the point cloud can be represented as a vector relative to the origin of the LRF, simplifying geometric operations and enabling accurate registration and analysis of the point cloud data. The problem is that the alignment may not be accurate due to variations caused by occlusion or noise in the point cloud. When estimating the LRF using point distributions, there may be rotational variations in the XY-plane and changes in the normal axis sign (Fig. 2.2 (d)).

Chapter 3

Related Works

3.1 Overview

In this chapter, a literature review on point cloud processing is presented. The chapter covers two main subjects: 1) Hand-crafted approaches for point cloud processing, and 2) deep learning-based approaches for point cloud processing. In particular, deep learning-based approaches are extensively reviewed. The following section describes common limitations of current algorithms and the corresponding motivations.

3.2 Hand-crafted method

3.2.1 Overview

Before the advancement of deep learning, a 3D feature descriptor was developed based on handcrafted methods. Owing to the specific properties

of 3D point clouds, such as unstructured and unordered, handcrafted approaches are important in 3D point cloud processing. These approaches mostly extract features from aligned volumes (using the normal or LRF) or directly extract rotation-invariant features to create a rotation-robust descriptor. The approaches can be classified into two main categories: LRF based descriptors and non-LRF based descriptors.

3.2.2 LRF based method

These approaches first compute a LRF and compute the spatial distribution of the local neighbors or geometric relationships between points according to the LRF.

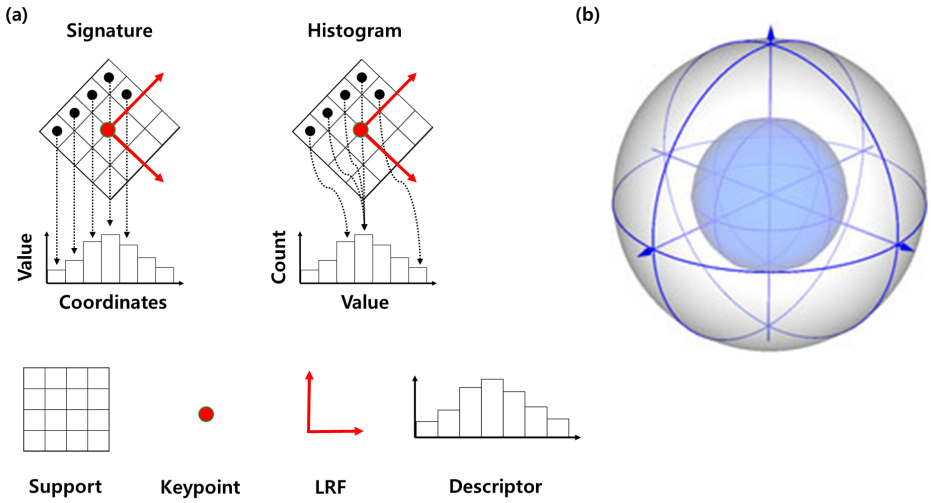

Figure 3.1: (a) Two kinds of descriptors: signatures and histograms; (b) SHOT descriptor [1].

Signature of Histogram of Orientation (SHOT) [1] is a combination of

two approaches: Signatures and Histograms (Fig. 3.1(a)). Signatures describe the neighborhood of a given keypoint by estimating a LRF and encoding geometric values computed for each point of a subset of the support based on the local coordinates. These Signatures are highly descriptive by using the spatially well-localized information, but vulnerable to small perturbations in the local coordinates. On the other hand, Histograms describe the 3D surface information by encoding quantities of local geometric characteristics (e.g. point coordinates, curvatures, normal angles) into histograms. Histograms are less descriptive when compared to Signatures, but improve robustness by compressing point information into histogram bins. By using the combination of Signatures and Histograms, SHOT aims at a more favorable balance between robustness and descriptive power. It encodes the surface information within a spherical support structure with 32 bins (8 divisions along the azimuth, 2 along the elevation, and 2 along the radius) (Fig. 3.1(b)). For every bin, values of a one-dimensional local histogram are computed, such as the angle between the normal of the keypoint and the current point within the bin. When the computations have been completed, all local histograms are merged to build a final descriptor.

3D Shape Context (3DSC) [29] creates a spherical support structure centered at the keypoint. The North Pole of the sphere is placed to match with the normal of the keypoint. Then, the sphere is divided in bins (divisions are equally spaced along the azimuth and elevation, divisions are logarithmically spaced along the radius so that they are smaller towards the center), and the number of neighboring points within the bin is accumulated for each bin. However, this still remains one degree of freedom because the method uses only the normal vector. To overcome this am-

biguous problem, the sphere is rotated around the normal vector N times. As a result of the repeated calculations for each rotation, a total of N descriptors are built for the keypoint. Unique Shape Context (USC) [30] is an improvement of the 3DSC descriptor by defining a LRF to provide a unique orientation for each point. This method estimates the LRF same way as the SHOT method. First, a covariance matrix of a keypoint and its neighboring points within the spherical support region is computed. The unit vectors of LRF are estimated using the Eigen Vector Decomposition of the covariance matrix. The eigenvectors corresponding to the maximum and minimum eigenvalues are used as unit vectors. The third unit vector is computed by the cross product of the two unit vectors. By using the LRF, the method improves the accuracy of the descriptor and also reduces descriptor size by removing the unnecessary multiple descriptors.

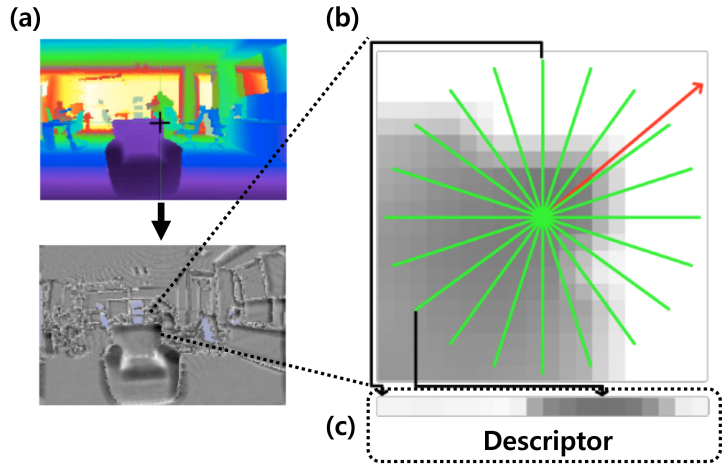

Figure 3.2: (a) RGB-D image; (b) Range image of the interest point; (c) NARF descriptor [2].

Normal aligned radial feature descriptor (NARF) [2] takes range images as inputs instead of a point cloud. The method first projects a point cloud into an image with depth values (RGB-D image), and then finds interest points, which show significant changes of the surface in their local neighborhood. Subsequently, they computed the NARF descriptor by building a normal aligned range value patch around the interest point and projecting a star pattern onto this patch to extract values for the final descriptor. This method also achieves the rotation invariance by shifting the NARF descriptor according to a unique orientation value extracted from the original descriptor.

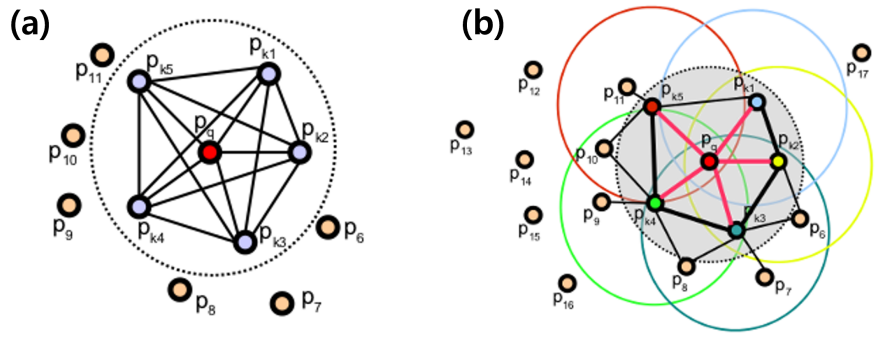

Figure 3.3: (a) PFH descriptor [3] and (b) FPFH descriptor [4].

Point Feature Histogram (PFH) [3] captures geometric information around each point by analyzing the relationships between points. The algorithm first pairs the keypoint with its neighbors and pairs the neighbors with themselves (Fig. 3.3(a)). Subsequently, for each pair, a Darboux frame is

computed from their normals as follows:

$$u = n_s,\ v = u \times \frac{p_t - p_s}{||p_t - p_s||_2},\ w = u \times v, \tag{3.1}$$

where p_s and p_t are points of the pair, and n_s is the normal vector of p_s. Then, using this frame, the difference between the normals and the Euclidean distance between the points are computed. Finally, the PFH descriptor is created by binning these computed angular and distance features into histogram bins. However, the theoretical computational complexity for a point cloud of n points with k neighbors is $O(nk^2)$, and the computational complexity makes PFH inappropriate for real-time or near real-time applications. Therefore, Fast Point Feature Histogram (FPFH) [4] is proposed to resolve the computation complexity problem. They first computed angular features between only keypoint and its neighbors using the same way as PFH and bin into histograms (Fig. 3.3(b)). In the second step, the final histogram (FPFH) is calculated as the weighted-average of the computed histograms based on the distance between keypoint and its neighbors. The FPFH descriptor can reduce the computational complexity to $O(nk)$ by considering only the direct connections between the keypoint and its neighbors, while still having effective power like PFH.

3.2.3 Non-LRF based method

Unlike the LRF based approaches, non-LRF based approaches builds the geometric relationships between each point and its neighbors to describe the local surface information without certain LRF.

Spin Image [5] was originally designed to describe mesh surfaces, but can be used for point clouds. They used an aligned cylinder using the nor-

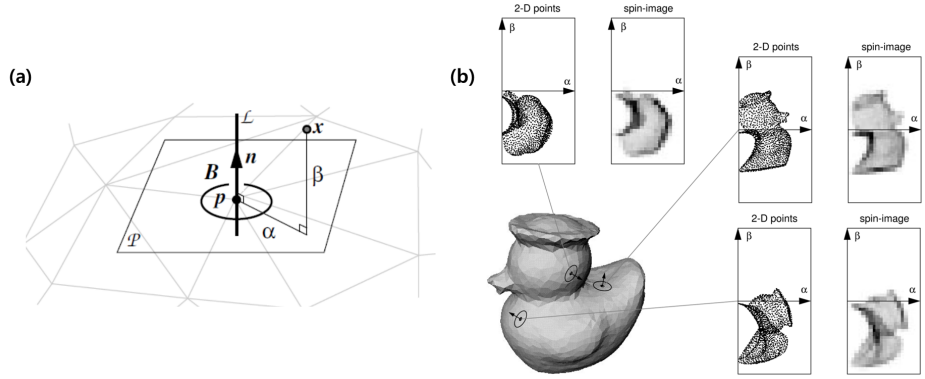

Figure 3.4: (a) Geometric relationship between the keypoint p and its neighbor point x; (b) SPIN descriptors (spin-images) [5] for three keypoints.

mal vector as the support structure. This cylinder is divided in bins along the radial and vertical values (Fig. 3.4(a)), and the final descriptor of the keypoint is established by counting the number of neighbor points lying inside each discrete 2D bin. The descriptor is displayed as a grayscale image, where dark areas correspond to bins with higher density.

Radius based Surface Descriptor (RSD) [31] describes the surface by computing an approximated radius of the fitting curve between a keypoint and its neighbor points. They assumed that the keypoint and its neighbors lie on the sphere surface and find the sphere by using the distance between points and angle between their normal vectors. In the case of an ideal plane, the estimated radius value will be infinite with all neighbors, and in the case of a corner, the radius value changes similarly as for spheres. From all the neighbor points of the keypoint, they computed radius values, and only the maximum and minimum radius values are saved to the descriptor of the

keypoint. The advantage is that this method is simple but effective.

3.3 Deep learning based rotation-variant method

3.3.1 Overview

The rapid developments of deep learning technology also bring developments in researching 3D point cloud descriptors. However, it is challenging to apply deep learning architectures to 3D point cloud straightforwardly because of the specific properties of 3D point clouds. Therefore, the representation format of 3D point clouds to be fed into deep learning pipeline becomes an important issue for many research works. One common approach is to build structured representations, which deep learning architectures can process, from the point clouds. The deep learning based descriptor approaches can be classified into three categories: volumetric based descriptors, multi-view based descriptors, and point based descriptors. The point based descriptors can be classified into three categories: pointwise MLP based descriptors, graph based descriptors, and point convolution based descriptors (Fig. 3.5).

3.3.2 Volumetric based method

In the case a 2D image, a convolution method uses a kernel to extract certain features from an input image by sliding the kernel across the image and multiplying that with the input. However, typical convolutional methods require highly regular input data formats. Therefore, the volumetric based methods voxelize an unstructured point cloud into a regular format like 3D occupancy grid, which the standard convolutional methods can

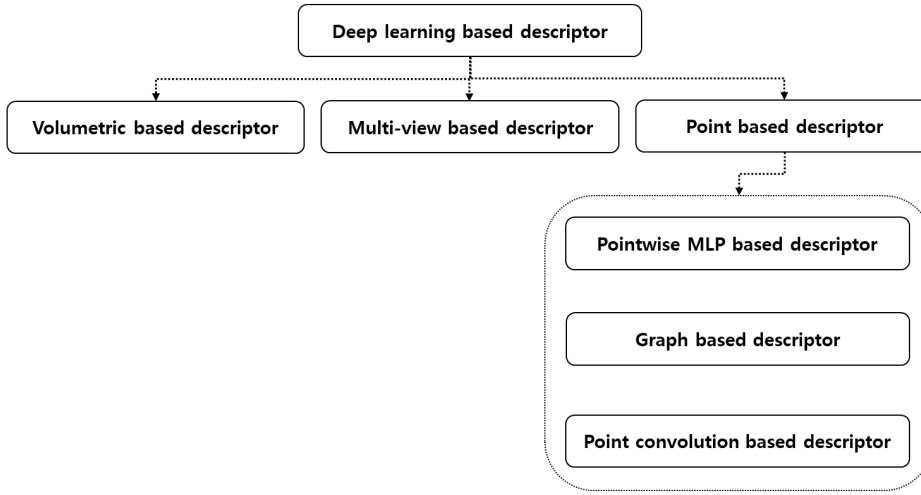

Figure 3.5: Taxonomy of deep learning based approaches.

process [6, 32] (Fig. 3.6). Moreover, by doing so, these methods represent unordered and irregular point clouds with simple and efficient structure, which makes the manipulation and storage easier.

VoxNet [6] is a pioneering work towards deep learning on 3D point cloud. They converted an input point cloud to an occupancy grid, and applied 3D convolutional methods on the occupancy grid for object recognition. The success of VoxNet boosts the development of deep learning on 3D point cloud. However, the quantification of the floating-point data results in an approximation, such that the input data intrinsically contains discretized artifacts. Moreover, because of the sparsity property of 3D point cloud, the conversion into the grid leads to a memory wastage problem due to existing of unoccupied space. Because the voxelization process consumes memory severely, these kinds of methods typically convert an input point

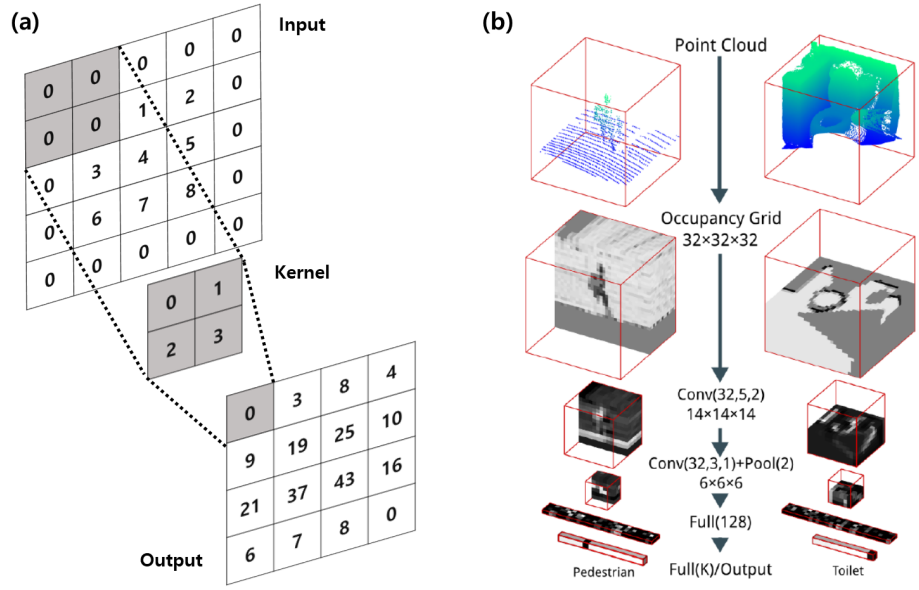

Figure 3.6: (a) Typical 2D image convolution and (b) 3D grid convolution of VoxNet [6].

cloud into a coarse grid of volumetric representation. Otherwise, the point cloud sparsity results in many zero multiplication operations for 3D convolution, and the additional third dimension makes the convolution process even more computationally expensive when compared to 2D image convolutions. To address the problem, certain methods represent point cloud data by optimizing the memory consumption or optimize the convolution process to ignore the unoccupied grids. In the case of optimizing the memory consumption, OctNet [33] divides the space by employing a set of unbalanced octrees based on density, and FCGF [34–36] uses a sparse tensor that only saves the nonempty space coordinates and features. In the case

of optimizing the convolutional process, Vote3Deep [37] constructs efficient convolutional layers to apply the convolution only at each non-zero location by using flipped the convolutional weights. However, it is still challenging to deal with a large scale point cloud data.

3.3.3 Multi-view based method

Multi-view based approaches are one of the simplest ways to analyze a 3D point cloud by using a set of 2D images rendered with different virtual cameras. The basic idea of these approaches is to generate 2D images of a 3D point cloud from different camera views and establish descriptors from the 2D view images with 2D CNNs.

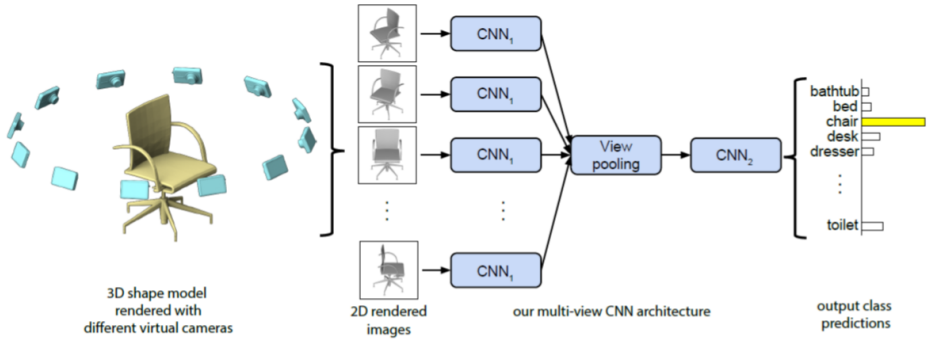

Figure 3.7: Network architecture of MVCNN [7] for 3D shape recognition.

MVCNN [7] is the earliest multi-view based method for 3D shape recognition. To generate rendered 2D views of 3D models, the method uses the Phong reflection model [38], and then the convolution network is trained to produce a descriptor for each 2D views. To produce a single and compact 3D shape descriptor, they aggregated the multiple descriptors at a

view-pooling layer and processed the aggregated descriptor with the second convolution network (Fig. 3.7). The method has made the milestone for 3D shape recognition, and many methods have been developed to improve the view aggregation method. In the case of MVCNN, all view descriptors are treated equally and aggregated using the view-pooling operation. This naive procedure smooths out the subtle local patterns and constrains the descriptor performances. To address the problems, Fuseption-ResNet (FRN) [39] uses the Inception-style architecture [40,41] as a shortcut branch at the view-pooling layer. This shortcut branch reinforces the view-pooling layer by using the feature maps obtained from the Inception-style networks. GVCNN [42] employs a grouping module to group the views according to their contents, and the group level descriptors are built using the group information. Subsequently, all group level descriptors are weighted ensembled to generate the final shape level descriptor. View-GCN [43] builds a graph with view descriptors and hierarchically aggregates view descriptors on the graph to generate the final descriptor. These methods improved performances on both 3D shape classification and retrieval tasks when compared to MVCNN, but there is an additional view-related issue in the multi-view based approach. It is a trade-off between performance and time efficiency. If the number of views decreases, the network performance drops sharply, while too many views increase time consumption. The optimal selection on the number of views needs to be determined based on the tasks.

3.3.4 Point based method

The aforementioned approaches, which convert point clouds into regular representations, have unsatisfactory performances because of the informa-

tion loss and quantitation artifacts. Therefore, recent works shift attention toward point-based approaches, which process raw 3D point cloud directly without data conversions, such as the voxelization [8, 44, 45] The point-based approaches can be classified into three categories: pointwise MLP based descriptors, graph based descriptors, and point convolution based descriptors.

Pointwise MLP based method

The basic idea of these approaches is to process raw 3D point cloud directly, but deep learning on 3D point clouds faces a significant challenge. The problem is that point clouds are an unordered set of points, unlike pixel arrays in 2D images or 3D volumetric grids. This means that even if point clouds, which represent the same 3D shape but are sorted in different data feeding order, are given as inputs, the network outputs of the inputs should be the same. This requires a network, which uses a 3D model with N input points, to be invariant to any permutations of the input points in data feeding order.

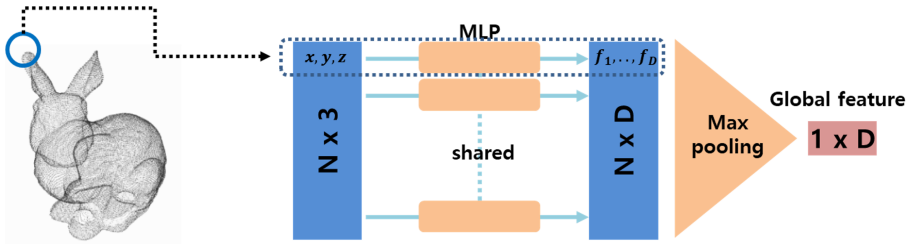

Figure 3.8: Network architecture of PointNet [8]. N and D denote the number of input points and output dimension.

PointNet [8] is a pioneering work, which deals with irregular and unordered point cloud straightforwardly. This method encodes each point individually using a shared multi-layer perceptron (MLP) and selects the maximum values among all the encoded features to generate a single global feature through global max pooling (Fig. 3.8). Because the shared MLP and global max pooling are symmetric functions, the method achieves permutation invariance. However, since the features are encoded independently for each point, the local spatial relationships between points are not captured, leading to limited network performance. To capture local geometric structure, some works [46–48] extend PointNet [8] by adopting hierarchical architectures that group each point's local neighbors and apply PointNet [8] to each group. Due to its simplicity, several works have adopted PointNet [8, 46] in the feature extraction and learning [49–51].

Graph based method

The success of pointwise MLP based approaches has been extended to various other approaches, such as graph based approaches. These approaches construct a graph from an input point cloud by using each point as a graph vertex and generating graph edges based on relationships between points. Then, the constructed graph is encoded to capture local spatial relationships among points.

DGCNN [9] is a pioneering work in the graph-based approaches. It constructs a graph using k-nearest neighbor (k-NN) strategy and encodes each edge of the graph using MLP independently (Fig. 3.9). The encoded edges are then aggregated to the connected points, and the graph is updated using the aggregated point features. LDGCNN [52] removes the transfor-

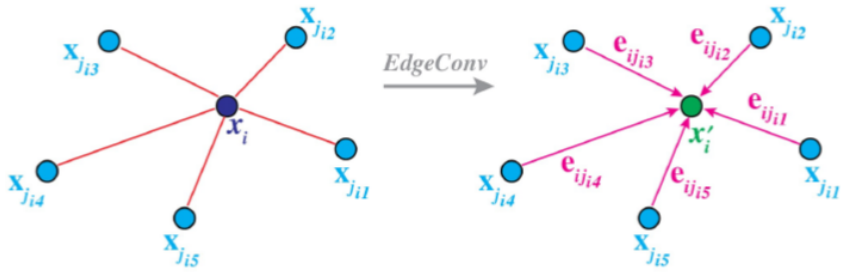

Figure 3.9: Graph construction and encoding operation (EdgeConv) of DGCNN [9]. The output of EdgeConv is calculated by aggregating the edge features associated with all the edges emanating from each connected vertex.

mation network of DGCNN, which is used to align an input point cloud, to reduce the network size and links the hierarchical features from different dynamic graphs by adding shortcuts between different layers to improve the performance. Similarly, the work [53] encodes multi-level graph features obtained from different sizes of graphs. DenseGCN [54] uses Inception architectures [55, 56] in feature extraction to obtain multi-scale features.

These approaches improve performance by constructing graph structures, but their performance is more sensitive to changes in point distribution than point-based approaches since they use neighboring points' coordinates directly in the graph construction.

Point convolution based method

Similar to image or volumetric-based approaches, point convolution-based approaches utilize convolutional kernels. To overcome the limitations of

volumetric-based approaches, which require a large amount of memory and have a high ratio of unoccupied grids, point convolution-based approaches define convolutional kernels on a 3D continuous space. This allows for the efficient processing of convolution operators.

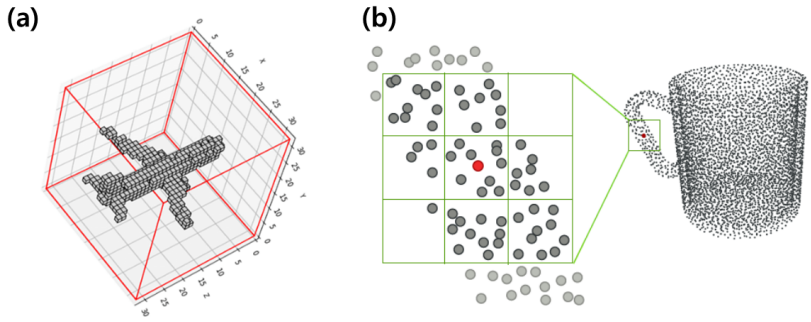

Figure 3.10: (a) Typical volumetric-based approaches. (b) Kernels of PointwiseCNN [10]. For each point, nearest neighbors are queried and binned into kernel cells before encoding.

PointwiseCNN [10] locates a kernel at each point of a point cloud instead of projecting a point cloud to 3D volume. For each point, nearest neighbors are queried and binned into kernel cells, and then the kernel cells are encoded like typical volumetric-based approaches (Fig. 3.10). This method reduces the amount of memory by using the pointwise kernel but lacks flexibility like volumetric-based approaches because the number of kernel cells is constrained. Some works define kernels as polynomial functions applied on the k-nearest neighbors with different weights depending on their distances [57, 58]. PCNN [59] and KPConv [60] use additional points, which have learnable weights. To adapt kernel points to local geometry, KPConv

learns the kernel point positions, maximizing the number of active kernel points. These approaches improve efficiency when compared to volumetric-based methods, but the choice of kernel shape can significantly affect the results. Therefore, determining the optimal kernel shape is necessary based on the tasks at hand.

3.4 Deep learning based rotation-invariant method

3.4.1 Overview

The aforementioned deep learning-based approaches, which are rotation-variant descriptors, show remarkable results when aligned point clouds are used as inputs. However, their performance is significantly reduced when randomly rotated point clouds are fed into the networks. Recent studies have attempted to build descriptors with rotation invariance. There are two types of approaches to achieving rotation-invariance: 1) Aligning input point clouds before feature extraction or 2) utilizing rotation-invariant feature extraction methods, such as relative distance or angle between points.

3.4.2 LRF based alignment

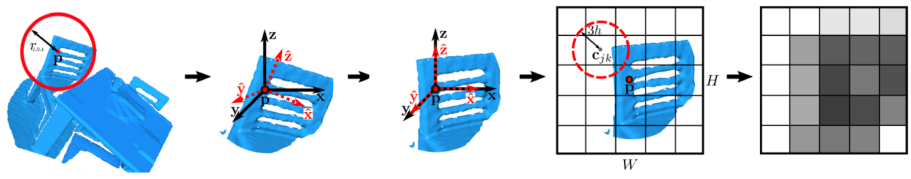

Figure 3.11: Voxelization process of 3DsmoothNet [11] using the LRF.

The aforementioned volumetric based methods simply convert points

as occupancy grids, so the obtained grids are not invariant to rotational variations. To address the problem, 3DsmoothNet [11] calculates a LRF for each interest point and transforms interest points and their neighbor points (i.e., patch) using the LRFs before the voxelization (Fig. 3.11). Similary, for each keypoint, GCAConv [61] establishes a LRF and transforms all the other points according to the keypoint's LRFs. Subsequently, it divides the transformed points into eight volumetric bins and encodes the barycenters of the bins, but estimating the eight barycenters from all the other points smooths out local subtle patterns. To supplement the incorrect LRF problem due to the rotational variance on the XY-plane, SpinNet [18] aligns a point patch using only the z-axis of the LRF and converts the point patch to the cylindrical volume, which can deal with the XY-plane rotational variance. However, since these methods build the volume for each point patch, it requires a lot of computational memory to build descriptors.

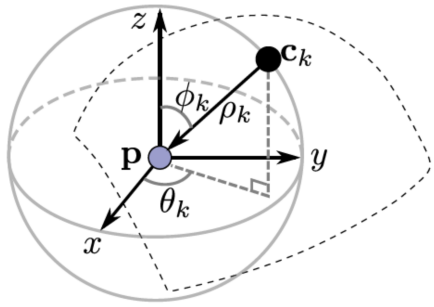

Figure 3.12: Local coordinates (θ_k, ϕ_k, ρ_k) for the camera viewpoint c_k in the LRF of p.

Similar to the volumetric based methods, multi-view based methods

also use LRFs to build rotation-invariant descriptors. LMVD [17] builds rotation-invariant descriptors by aligning cameras instead of patches. For a given interest point p, LMVD computes the LRF by using the normal of the interest point and a constant upright vector u: the z-axis is collinear with the normal of p, the x-axis is obtained by taking the cross product of u and the z-axis, and the y-axis is obtained by taking the cross product of the z-axis and x-axis. Then, the method places camera viewpoints in the LRF of the interest point (Fig. 3.12). However, because u is the constant vector, the x-axis and y-axis of the LRF can be changed under point cloud rotations.

In the case of the point based methods, DIP [16] and GeDi [62] align patches and then build descriptors by applying PointNet [8] and PointNet++ [46] methods to the aligned patches. Different from the aforementioned methods, to mitigate the incorrect LRF problems (i.e., XY-plane rotational variation), DIP and GeDi align patches using the additional transformation networks in addition to the LRF. However, these methods require training the additional transformation networks with many convolution channels.

3.4.3 Rotation-invariant feature extraction

REQNN [12] converts both the input point clouds and intermediate-layer features into quaternion features, which possess the property of rotation equivariance. This means that the feature generated from a point cloud rotated by a specific angle is equivalent to the rotated feature generated from the original point cloud (Fig. 3.13). To enable further tasks such as classification, the quaternion features are transformed into real-valued vec-

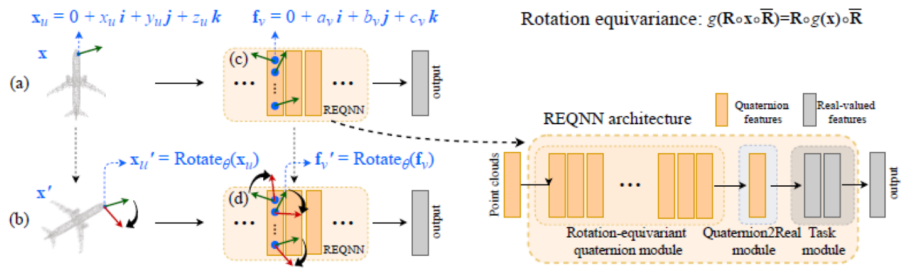

Figure 3.13: Network architecture of REQNN [12]. Both input point clouds and intermediate-layer features are represented by quaternion features. The quaternion feature $g(R \cdot x \cdot \bar{R})$ generated from the point cloud rotated by a specific angle is equivalent to the rotated quaternion feature $R \cdot g(x) \cdot \bar{R}$ generated from the original point cloud.

tors by computing the square of the norm of each quaternion element, which have rotation-invariant property. However, REQNN encodes each point independently for rotation invariance, which constrains the network performance. SFCNN [63] maps encoded 3D points, which are obtained using PointNet [8], onto a discretized sphere graph. Since the encoded points vary with rotations, SFCNN rotates them to a constant vector to ensure rotation-robustness. However, this approach is similar to checking only the distance from the origin to each point, resulting in the loss of spatial information. RIConv [27] encodes rotation-invariant features, such as relative angles and distances between points and their neighbors, using a shared multi-layer perceptron (MLP) and divides points into bins based on distances from the object center. The method subsequently encodes the ordered bins using 1D convolution to address the point ordering issue.

ClusterNet [28] uses hierarchical clustering to explore the geometric structure of the point cloud and applies a MLP to each point and cluster. The previously employed rotation-invariant features of RIConv [27] and ClusterNet [28] are insufficient to represent the relative positions of neighbors from each point completely. To supplement the point relationship represen-

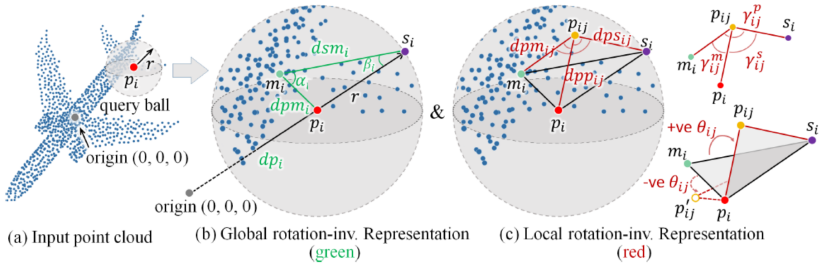

Figure 3.14: Global and local rotation-invariant representations of RIF [13]. Two kinds of additional reference points (m_i and s_i) are used to represent the neighbors of p_i.

tations, RIF [13] represents neighbors within the query ball of the interest point by using rotation-invariant features and additional reference points (centroid m_i of neighbors and intersection s_i between the query ball and line extended from origin in Fig. 3.14). However, the reference points have a chance to be changed depending on object shape variation, and it may result in insufficient consistency of the descriptors between similar object parts. Moreover, the used encoding method (i.e, MLP) simply encodes each point feature without considering other points [13,27,28] to avoid processing in non-rotation-invariant order, and it constrains the network performance. RIConv++ [64] estimates relationships between the interest point and its neighbor points and additionally estimates relationships between neigh-

bor points to supplement inaccurate point relationship representations. By using additional representations, it can improve the point relationship representations, but if one of the neighbor point is changed or occluded, it can affect the other adjacent neighbor's features.

Chapter 4

Aligned Kernel based Rotation-Invariant Method

4.1 Overview

Point cloud analysis such as classification and segmentation is typically performed using descriptors that describe point cloud geometric shapes. Thus, generating good descriptors through geometric feature extraction and extracted feature encoding is an important task for accurate point cloud analysis. However, it is challenging to develop consistently good descriptors for real-world point clouds obtained from 3D sensors, because real-world point clouds are not aligned (i.e., rotated and translated) unlike the high-quality human-made point clouds. Therefore, to make descriptors applicable to general datasets, it is the major requirement to make rotation-robust descriptors.

In this chapter, a rotation-robust descriptor generation method is pre-

sented. The rotation-robustness is achieved by using both a LRF-based kernel alignment and rotation-invariant features in the descriptor generation process. Additionally, to enhance both the performance and scale-robustness of the descriptor, a global context aggregation method and a scale adaptation module are applied.

The primary objective of the proposed method is to enhance rotation-robustness for practical applicability in scene understanding tasks such as classification, segmentation, and registration. With rotated and translated point clouds, achieving consistent results is challenging, making it necessary to improve robustness performance. In this dissertation, a descriptor generation method that shows a great robustness performance is proposed. The proposed method shows that the rotation-invariant feature extraction and the aligned kernels are the most important factor for improving rotation-robustness. Without the rotation-robustness, the network has to learn all the varied feature patterns, resulting in reduced performance under rotational variations. Moreover, because the generated descriptors only contain local geometric information around each point, an additional network, which aggregates geometric information from all the descriptors, is proposed to increase the range of geometric information of each descriptor. In addition to the rotation-robustness, an additional network, which finds the optimal value of the input parameter affected to scale variations, is proposed to make the network robust to scale variations. Extensive studies will be presented regarding the performance of rotation-robustness by comparing the proposed method with several state-of-the-art methods and self-ablations. The overall architecture is described in the following sections.

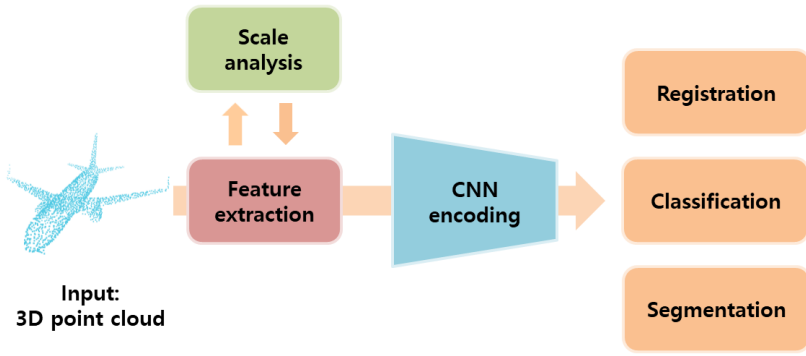

Figure 4.1: Overview of the proposed architecture. First, features are extracted using multiple kernel sizes. Subsequently, scale analysis is employed based on the interpolation between the kernel sizes. Finally, the descriptor is encoded using the adjusted scale for the downstream tasks.

4.2 Descriptor generation

Figure 4.1 illustrates an overview of the proposed descriptor-generation framework. The basic idea is to place additional aligned and structured points (i.e., kernel points) around each point and extract rotation-invariant point relationships (i.e. rotation-invariant features) using the aligned kernel points (Feature extraction in Fig. 4.1). The extracted rotation-invariant features are then encoded in a rotation-invariant manner (CNN encoding in Fig. 4.1) to generate per-point descriptors for one of the three tasks (Registration, Classification, and Segmentation in Fig. 4.1). Additionally, to supplement the local geometric information of each descriptor, all descriptor information aggregation (i.e., global context aggregation) is performed in the encoding procedure (CNN encoding in Fig. 4.1). Moreover, to enhance

scale-robustness, a scale analysis is performed to determine the optimal kernel size according to the point cloud size (Scale analysis in Fig. 4.1).

This chapter is organized as follows. Section 4.2.1 presents the kernel placement and alignment. The rotation-invariant feature extraction is described in Section 4.2.2, and encoding the extracted features is presented in Section 4.2.3. The global context aggregation is discussed in Section 4.2.4. Section 4.2.5 presents the scale analysis, which enhances scale-robustness. Various convolutional neural network (CNN) encoder architectures are presented in Section 4.3, which are used for the downstream tasks.

4.2.1 Kernel alignment

Compared to rotation-variant features (i.e., point coordinates), representing point relationships accurately with rotation-invariant features (i.e., relative distances) is difficult due to the reduced amount of information. Using incomplete rotation-invariant features may decrease overall network performance because too many geometric shapes can be mapped to those features (Fig. 1.2). To resolve the problem, additional cylinder-shaped kernel points, which are aligned using a local reference axis (i.e., normal vector), are used to assist point relationship representations (Fig. 4.2(a)). The normal vector is computed based on the 3DsmoothNet method [11]. Given a point cloud $\mathcal{P} = \{x_i \in \mathbb{R}^3\}_{i<N}$, a neighboring point set $\mathrm{P}_{i,r} = \{x_j : ||x_j - x_i||_2 \leq r\}$, where r is the radius, is extracted for each point x_i. For the normal vector of x_i, a covariance matrix of x_i is first computed using the neighbor points $\mathrm{P}_{i,r}$:

$$\Sigma_i = \frac{1}{|\mathrm{P}_{i,r}|} \sum_{|x_j \in \mathrm{P}_{i,r}} (x_j - x_i)(x_j - x_i)^T. \tag{4.1}$$

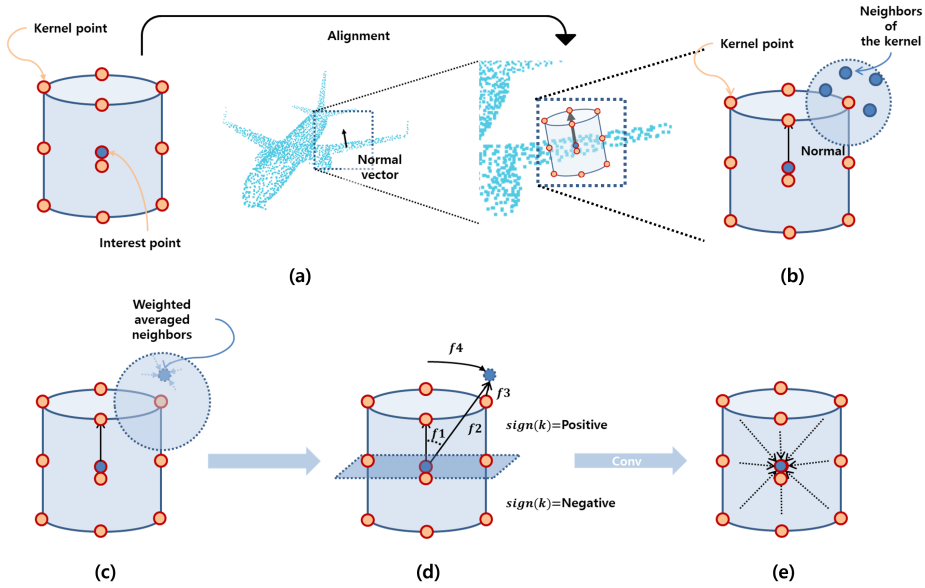

Figure 4.2: (a) Kernel points are aligned using the LRF of the target point; (b) Neighbors are selected for each kernel point; (c) Weighted-average location is estimated based on the distance from the kernel point to the neighbors; (d) For each kernel, the relative location of the averaged neighbor is estimated using distances and angles; (e) After convolution, kernel features are aggregated by summation and maximum value selection to represent the interest point.

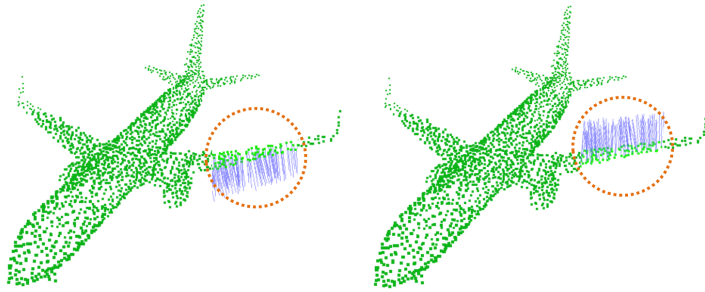

Figure 4.3: Visualization of normal vectors. The signs for each point on a planar surface are not determined uniquely.

Then, the eigenvector corresponding to the smallest eigenvalue of the matrix is used as the normal vector. From the normal vector, the rotation matrix R is estimated, and the kernel is aligned using R.

The estimated normal vector may have problems, such as sign variation problems (Fig. 4.3), but the cylindrical shape, of which the cylinder column is aligned to the normal vector, makes the kernel robust to the XY-plane rotational variations and invariant to the normal axis sign variation.

4.2.2 Rotation robust feature projection

Once all the kernels are aligned for each point, the next step is to represent rotation-invariant point relationships (i.e., rotation-invariant features) using the kernels. First, the averaged location of the neighbor points is calculated based on their distance from the kernel point x_i^k:

$$\hat{x}_i^k = \sum_{x_j \in \mathrm{P}_n} \frac{w_j x_j}{\sum w_j} \text{ where } w_j = exp(\frac{-(x_j - x_i^k)^2}{d^2}), \tag{4.2}$$

where $\hat{x}_i^k \in \mathbb{R}^{N\times 3}$ is the weighted-average location for the k-th kernel point x_i^k, and d indicates the cylinder radius. For the weighting term w_j, the Gaussian function is used to reduce the influence of outliers in (4.2).

For rotation-invariant representations, four types of features are estimated. First, the angle between two vectors, one is from the center point of the kernel to the weighted-average point and the other is the normal vector, is estimated (the angle f1 in Fig. 4.2(d)). However, because the normal vector has a normal vector sign variation problem (Fig. 4.3), a negative sign is multiplied with the normal vector if the kernel is located below the tangent plane, as shown below:

$$f1_i^k = v_i \cdot \frac{\hat{x}_i^k - x_i}{||\hat{x}_i^k - x_i||} \cdot sign(k), \tag{4.3}$$

where v_i indicates the normal vector of x_i and $sign(k)$ returns a negative sign if the k-th kernel point is located below the tangent plane. This term determines the angle value, regardless of the normal sign. Next, the distance from the kernel center x_i to the averaged neighbor point $\hat{x}_i^k$ and distance from the kernel point x_i^k to the averaged neighbor point $\hat{x}_i^k$ are estimated (distances f2 and f3 in Fig. 4.2(d)):

$$f2_i^k = ||\frac{\hat{x}_i^k - x_i}{d}|| \tag{4.4}$$

$$f3_i^k = ||\frac{\hat{x}_i^k - x_i^k}{d}|| \tag{4.5}$$

Finally, to provide direction to the closest adjacent kernel points, the distance ratio from two adjacent kernel points to the averaged point is estimated (ratio f4 in Fig. 4.2(d)):

$$f4_i^k = \frac{||\hat{x}_i^k - x_i^{k+1}||}{||\hat{x}_i^k - x_i^{k+1}|| + ||\hat{x}_i^k - x_i^{k-1}||} \tag{4.6}$$

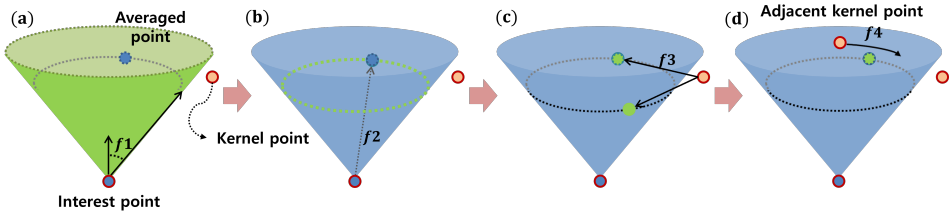

Figure 4.4: Green regions of (a), (b), (c), and (d) illustrate the possible locations of the neighbor point of the interest point when (f1), (f1, f2), (f1, f2, f3), and (f1, f2, f3, f4) are given, respectively.

where x_i^{k+1} and x_i^{k-1} are adjacent kernel points of x_i^k. The relative location of the averaged neighbor points can be successfully represented based on the presented angle- and distance-based descriptions. Figure 4.4 illustrates the possible locations of the neighbor point of the interest point (i.e., x_i), represented by using the proposed rotation-invariant features.

4.2.3 Circular convolution

The kernels are not aligned to the unique LRFs, so the kernel direction may change on the XY-plane according to the distribution of points, but the adjacent kernel points within the cylinder layer are invariant to rotation. Therefore, a naive $1 \times 1 \times 1$ channel-wise convolution method (4.7), which encodes each kernel point independently (Fig. 4.5(a)), can be extended to (4.8):

$$x_i = \sum_{\hat{x}_i^k} f(g(\hat{x}_i^k)), \tag{4.7}$$

$$x_i = \sum_{\hat{x}_i^k} f(g(\hat{x}_i^{c(k,-1)}), g(\hat{x}_i^k), g(\hat{x}_i^{c(k,+1)})), \tag{4.8}$$

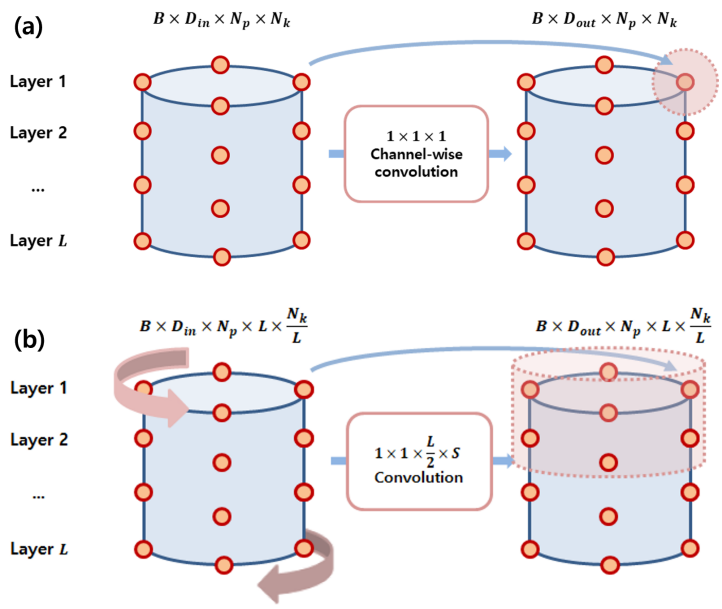

Figure 4.5: (a) Channel-wise convolution and (b) circular convolution. The red transparent region indicates the receptive field. S indicates the kernel size of the convolution.

where $g(\cdot)$ and $f(\cdot)$ are the feature extraction function and convolution function, respectively. $c(k, +1)$ and $c(k, -1)$ indicate the clockwise and counterclockwise adjacent kernel points of the k-th kernel point in the cylinder layer. Subsequently, to avoid the sign problem, kernel points are divided into two groups: the collection of kernel points above the tangent plane and the collection of kernel points below the tangent plane. If the k-th kernel point belongs to the first group, the kernel point for $c(k, +1)$ is selected in a clockwise order. Otherwise, the kernel point for $c(k, +1)$ is selected in a counterclockwise order (red arrows in Fig. 4.5(b)). By dividing ker-

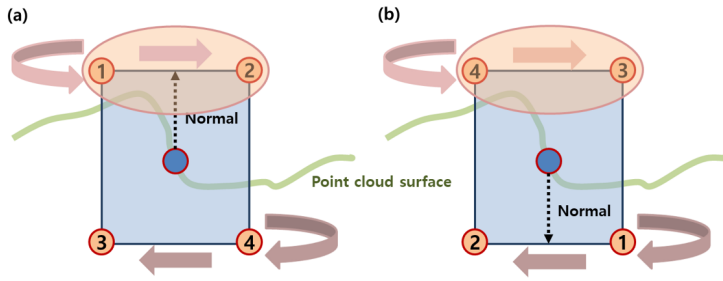

Figure 4.6: Same point cloud, different normal vector signs. Regardless of the normal vector sign, the kernel point located at "1" in (a) ("4" in (b)) is processed together with the kernel point located at "2" in (a) ("3" in (b)). In addition, the kernel weight applied to the kernel "1" in (a) and kernel "4" in (b) is the same by using the symmetric convolution method.

nel points into two groups and applying symmetric convolution, the deep learning result remains the same even if the cylindrical kernel flips due to changes in the normal vector signs (Fig. 4.6). This is because the adjacent kernels chosen for each kernel do not change.

In addition, if the layers belong to the same group, multiple layers are encoded together (Fig. 4.5(b) right), i.e., (4.8) is extended to

$$x_i = \sum_{\hat{x}_i^k} f(g(\hat{x}_i^k), g(\hat{x}_i^j)_{j \in adj(k)}), \tag{4.9}$$

where $adj(k)$ indicates a set of adjacent kernel points of the k-th kernel point in the same group. A circular padding convolution operation was used to implement (4.9). Figure 4.5 illustrates the convolution process using the kernels, and figure 4.6 illustrates an example of the process with the same point cloud but different normal vector signs. After convolution, the

encoded features, which are stored in the kernel points, are aggregated by the summation and maximum value selection.

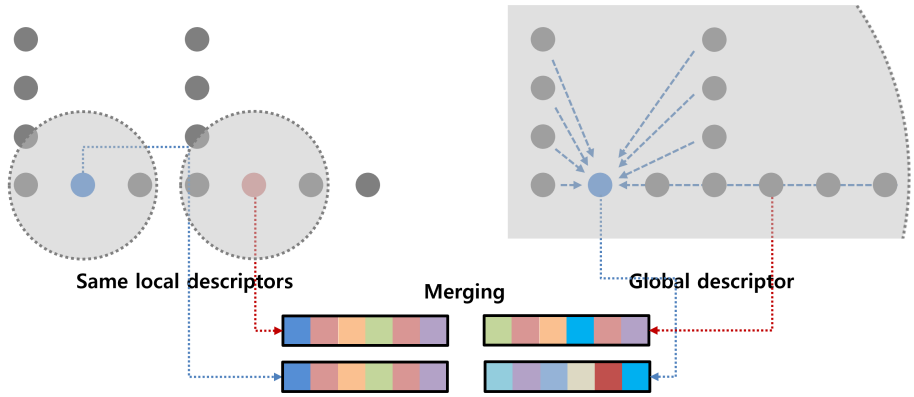

Figure 4.7: Left: two descriptors obtained from repeated area have same values because local descriptors only have the limited range of geometric information around the interest points. Right: two global descriptors obtained by aggregating all the descriptors have different values. Gray circles indicate the range of encoded geometric information of descriptors. By merging local descriptors and global descriptors, two descriptors have different values.

4.2.4 Global context aggregation

Typically, a descriptor for a point (i.e., local descriptor) only contain geometric information around the point, not all point cloud information. The problem is that local descriptors of monotonous and repeating areas have same or similar values because of the limited geometric information range (Fig. 4.7), and these similar descriptors can hinder further analyses, such

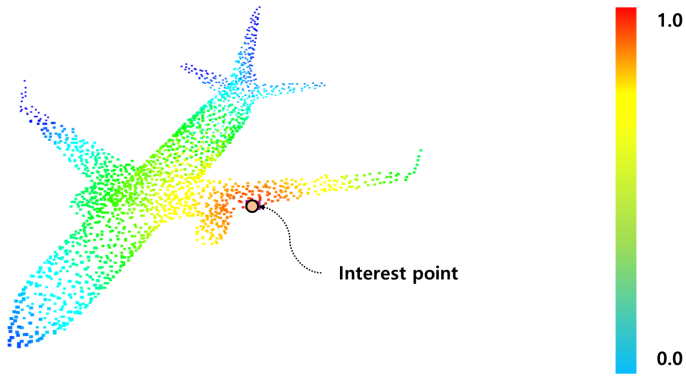

Figure 4.8: Colors of point cloud indicate global context weights of the interest point. Red indicates a weight value of one and blue indicates a weight value of zero.

as registration. To make distinguishable descriptors, the global context (i.e. global descriptor) is estimated from all the local descriptors. However, simply aggregating all the local descriptor values using max-pooling or summation makes only a single global context, which is not useful for improving descriptor distinctness. Therefore, rather than estimating a single global context, the adaptive global contexts for each point are estimated by using distance-based weights.

To estimate the global context for the i-th point (i.e., interest point), weights w_{ij} are calculated based on the Gaussian distance between the i-th and j-th points (weights for an interest point in Fig. 4.8). Subsequently, the averaged local descriptor value g_i is estimated using the weights w_{ij} for all j.

$$g_i = \sum \frac{w_{ij} f_j}{\sum w_{ij}} \text{ where } w_{ij} = exp(\frac{-(x_j - x_i)^2}{d^2}), \tag{4.10}$$

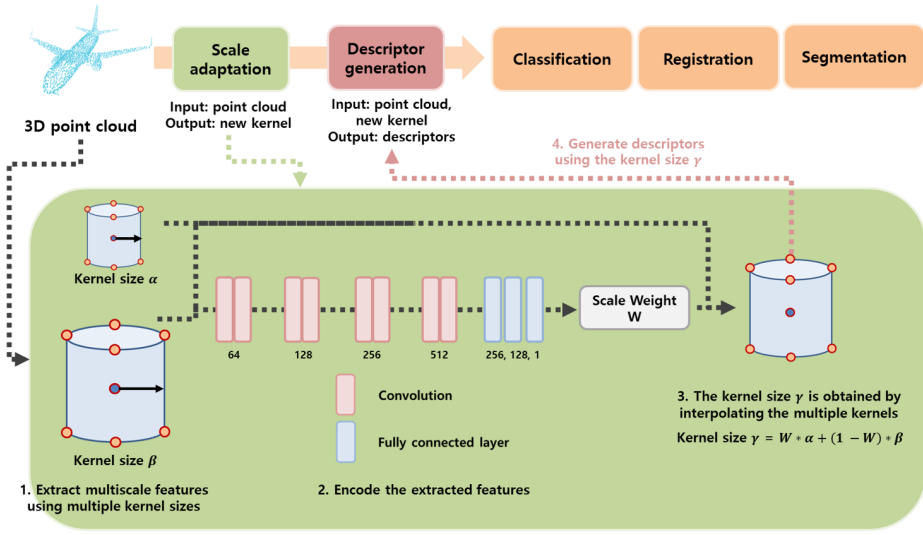

Figure 4.9: Multiple features are extracted using multiple kernel sizes (α and β). By applying simple convolution operations to these features, an interpolation weight W between the two kernel sizes is estimated. The final kernel size is then determined by applying the interpolation weight to the original kernel sizes.

where f_j indicates the descriptor value of the j-th point, and d is the kernel size. This way, points that are close to the interest point have a higher weight than those that are far away, and their descriptor values are more influential in calculating the global context. Once the global contexts are estimated for each point, the global contexts are concatenated to each local descriptors. Finally, a simple $1 \times 1 \times 1$ channel-wise convolution operation is performed on concatenated descriptors (equation 4.7 in Section 4.2.3) to refine the descriptors for downstream tasks.

4.2.5 Scale adaptation module

The point cloud size can vary for the same object, depending on the viewpoint of capture. Similar to the rotational and translational variations, scale variations also increase the burden of the deep learning network by increasing the input patterns. To resolve the scale variation problem, one common approach is to normalize point cloud coordinates to be within a range of -1.0 and 1.0. However, If an input point cloud has some noises or has an unusual shape when compared to the training data, the normalization process might fail to resolve the scale problem. To mitigate the scale problem, the proposed scale adaptation module is performed after the point cloud normalization.

In the proposed method, the kernel size (d in (4.2)) is the key parameter affected by scale variation. This implies that the features extracted, which describe the nearby geometric structure, can vary depending on the used kernel size. If a kernel size that is too large is used, the extracted features may contain overly approximate geometric information or miss important details. Conversely, if a kernel size that is too small is used, the features extracted may only contain a limited range of regional information, which could be insufficient for effectively analyzing the shape of the point cloud. Therefore, the task of the scale adaptation module is to analyze multiple kernel sizes to identify the optimal one, thereby modifying the kernel size instead of altering the point cloud scale.

First, multiple features using multiple sizes of the kernels (α and β in Fig. 4.9) are extracted. Subsequently, the multiscale features are concatenated, simple $1 \times 1 \times 1$ channel-wise convolution operations and fully

connected layer operations are employed for the scale analysis (Convolution and Fully connected layer in Fig. 4.9). By extracting and analyzing multi-scale features, the scale adaptation module is able to estimate the optimal kernel size. As a result, the convolution's output is the interpolation weight (Scale weight W in Fig. 4.9), between the smallest and largest kernel sizes used. Finally, the kernel, which is derived from interpolating among multiple kernels, is used to generate the proposed descriptor for the classification, registration, and segmentation tasks. The weights of this module are concurrently learned during the training of the registration and classification networks.

4.3 CNN encoding

The designed CNN encoders are illustrated in Fig. 4.10. There are five modules: feature extraction module (Section 4.2.2), three kinds of encoding related modules (circular convolution (Section 4.2.3), global context aggregation (Section 4.2.4), and MLP), and registration related module (SVD).

The feature extraction modules are initially used, and then the circular convolution modules are applied to generate descriptors from the extracted features. The generated multiscale descriptors are concatenated using a shortcut connection (side arrows in Fig. 4.10). Subsequently, global contexts are estimated from the concatenated descriptors to improve the descriptor distinctness.

For the registration task, the singular value decomposition (SVD) module [14] is used to estimate the rotation matrix R and translation vector t

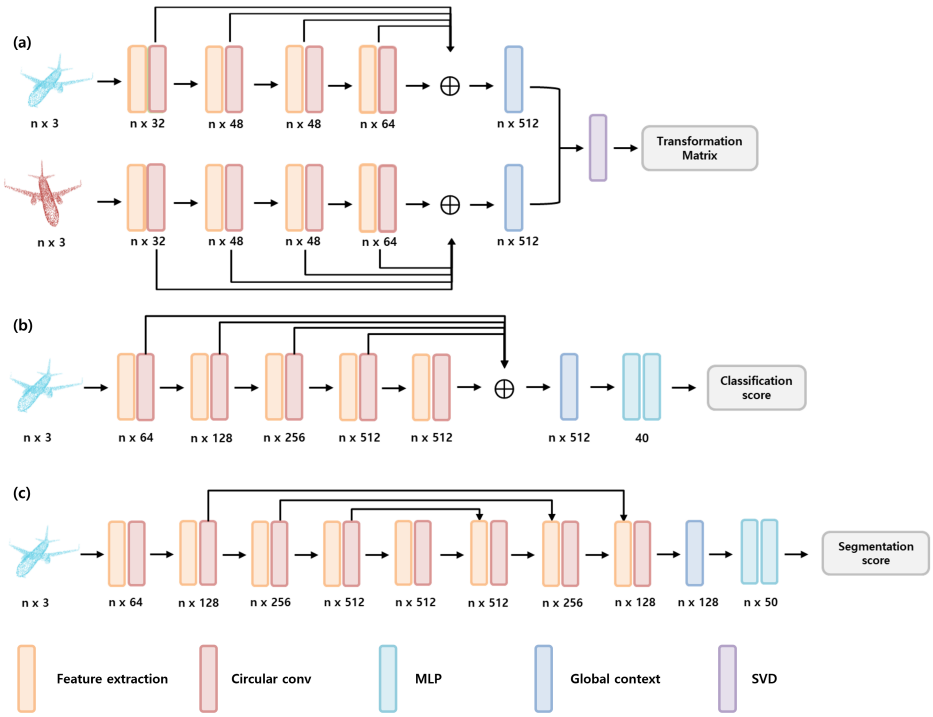

Figure 4.10: Network architectures for (a) registration, (b) classification, and (c) part segmentation.

from the two descriptor sets. For given two descriptor sets $\mathcal{D}_P \in \mathbb{R}^{N\times 512}$ and $\mathcal{D}_Q \in \mathbb{R}^{N\times 512}$, the module first estimates $\hat{\mathcal{D}_P}$ from $\mathcal{D}_Q$ based on the similarity matrix $Mat_{P,Q} = \mathcal{D}_P \times {\mathcal{D}_Q}^T$:

$$\hat{\mathcal{D}_P} = softmax(Mat_{P,Q}) \times \mathcal{D}_Q, \tag{4.11}$$

where $softmax(*)$ is the row-wise normalization function. Then, using $\mathcal{D}_P$ and $\hat{\mathcal{D}_P}$, the module estimates the covariance matrix H:

$$H = (\mathcal{D}_P - centroid_{\mathcal{D}_P})(\hat{\mathcal{D}_P} - centroid_{\hat{\mathcal{D}_P}})^T, \tag{4.12}$$

where $centroid_*$ is the center location of $*$. Finally, by applying the singular value decomposition to H (i.e., $[U, S, V] = SVD(H)$), R and t are estimated:

$$R = VU^T, \tag{4.13}$$

$$T = centroid_{\hat{\mathcal{D}_P}} - R \times centroid_{\mathcal{D}_P}. \tag{4.14}$$

The registration network is trained by minimizing the following loss function as DCP [14]:

$$Loss_{regi} = ||R^T R_{gt} - I||^2 + ||T - T_{gt}||^2, \tag{4.15}$$

where R_{gt} and t_{gt} are the ground-truth rotation matrix and translation vector.

For the classification tasks, the single descriptors are estimated by averaging all the descriptors (i.e., $N \times 512$ to 512) before applying the multi-layered perceptron (MLP) modules to predict the class for the input point cloud, not for each point. Because the number of class is 40, the output is the vector of length 40. Each value represents the probability for each class, and the class with the highest probability is the predicted class for the input point cloud. The classification network is trained by minimizing a cross-entropy loss:

$$Loss_{cls} = -\sum_{i}^{40} g_i log(p_i), \tag{4.16}$$

where p_i and g_i are the i-th values of the output vector and ground-truth vector, respectively.

Similarly, for the segmentation task, MLP modules are applied to estimate per-point segmentation scores, and the segmentation network is trained by minimizing a point-wise cross-entropy loss:

$$Loss_{seg} = -\sum_{j}^{N}\sum_{i}^{50} g_i log(p_i^j), \tag{4.17}$$

where p_i^j is the i-th values of the output vector of the j-th point.

For the training, Adam optimizer [65] is applied. The learning rate is set to 0.001 and decayed by multiplying 0.7 for every 20 epoch. The network is trained for 250 epochs using Intel i7-7700 CPU with 3.60 GHz processors, 16 GM memory, and NVIDIA Geforce 1080 (12 GB) GPU machine. The PyTorch framework is used for the implementation of the network.

4.4 Experimental results

4.4.1 Overview

The aim of the experimental section is to evaluate the proposed method's strengths and weaknesses in three tasks: registration, classification, and segmentation. The following sections present the configuration of prepared data and evaluation metrics, followed by an overall comparison with state-of-the-art methods. A rich ablation study of the proposed method is then presented.

4.4.2 Data configurations

In total, three kinds of tasks were implemented: registration, classification, and segmentation. For the registration task, ModelNet40 [66] database was used. ModelNet40 contains 12,311 meshed computer-aided design models

from 40 classes. For each model, 1,024 points were used for training and testing. For the classification task, ScanObjectNN [67] database was used in addition to ModelNet40 to demonstrate the adaptability on the real-world dataset. ScanObjectNN contains 2,902 real-world models from 15 classes, and each model is randomly translated, rotated, scaled, and partially occluded. For each model, 2,048 points were used for training and testing. For the segmentation task, ShapeNetPart [68] database was used. ShapeNetPart contains 16,681 models from 16 classes. Each point is annotated using part labels. For each model, 2,048 points were used for training and testing.

In the experiments, the rotation-robustness and generalization ability of the proposed network were evaluated by comparing with the state-of-the-art descriptor methods. The state-of-the-art descriptor methods, PointNet [8], DGCNN [9], KPConv [60], RIF [13], and the proposed method were used for the performance evaluation.

4.4.3 Evaluation metric

The registration results were measured using mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) between ground-truth values and predicted values (rotation (R-) and translation (T-)):

$$R\text{-}MSE = \sum_{i}^{3} (euler(R_{pred}) - euler(R_{gt}))_i^2, \tag{4.18}$$

$$R\text{-}RMSE = \sqrt{\sum_{i}^{3} (euler(R_{pred}) - euler(R_{gt}))_i^2}, \tag{4.19}$$

$$R\text{-}MAE = \sum_{i}^{3} |euler(R_{pred}) - euler(R_{gt})|_i, \tag{4.20}$$

$$T\text{-}MSE = \sum_{i}^{3} (T_{pred} - T_{gt})_i^2, \tag{4.21}$$

$$T\text{-}RMSE = \sqrt{\sum_{i}^{3} (T_{pred} - T_{gt})_i^2}, \tag{4.22}$$

$$T\text{-}MAE = \sum_{i}^{3} |T_{pred} - T_{gt}|_i, \tag{4.23}$$

where R and T represent the rotation matrix and translation vector, respectively. $*_{pred}$ and $*_{gt}$ indicate predicted and ground-truth transformation information, respectively. The function $euler(*)$ returns Euler angles for a given rotation matrix.

For classification, results were measured using the classification overall accuracy (OA):

$$OA = \frac{N_C}{N_T}, \tag{4.24}$$

where N_C and N_T represent the number of correct predictions and total number of predictions, respectively.

For the segmentation, the results were measured using the mean class IoU (mcIoU) per class:

$$mcIoU = \frac{1}{C} \sum_{j}^{C} \frac{1}{|C_j|} \sum_{i \in C_j} \frac{|L_{pred}^i \cap L_{gt}^i|}{|L_{pred}^i \cup L_{gt}^i|} \tag{4.25}$$

where L_{pred}^i and L_{gt}^i represent predicted and ground-truth label sets of the i-th point cloud, respectively. N, C, and C_j represent the number of point clouds, the number of classes, and the point clouds of the j-th class, respectively.

4.4.4 Quantitative analysis

Method	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE	C
ICP	892.601	29.876	23.626	0.086	0.293	0.251	-
Go-ICP [69]	192.258	13.865	2.914	0.000	0.022	0.006	-
FGR [70]	97.002	9.848	1.445	0.000	0.013	0.002	-
PointNetLK [71]	306.323	17.502	5.280	0.000	0.028	0.007	20
PointNetLK [71]	227.870	15.095	4.225	0.000	0.022	0.005	40
DCP [14]	9.923	3.150	2.007	0.000	0.005	0.003	20
DCP [14]	1.307	1.143	0.770	0.000	0.001	0.001	40
Proposed	**0.017**	**0.130**	**0.064**	**0.000**	**0.000**	**0.000**	20

Table 4.1: Registration results with randomly rotated same point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-). C indicates the number of used classes for training.

Registration

For registration, the proposed method was analyzed using randomly rotated point clouds from ModelNet40 [66]. Three types of experiments were conducted: 1) randomly rotated point clouds (Table 4.1), 2) randomly rotated noisy point clouds (Table 4.2), and 3) randomly rotated and partially sampled point clouds (Table 4.3).

As listed in Table 4.1, the proposed descriptor method significantly reduced the registration errors when compared to the other methods. Even

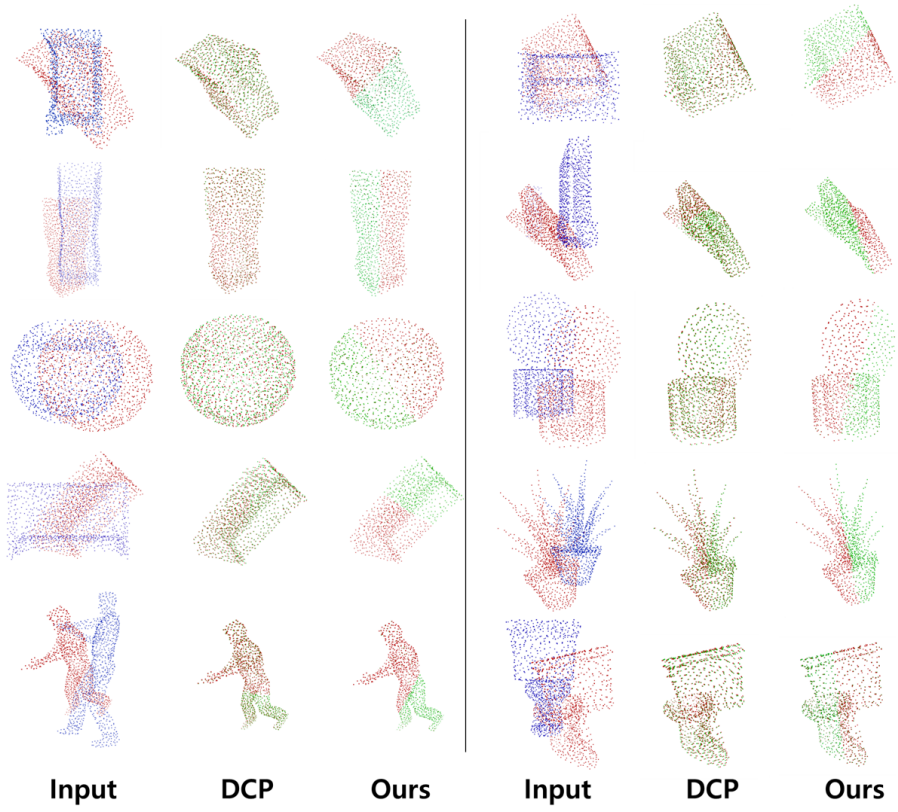

Figure 4.11: Left: source (blue) and target (red) point clouds, Middle: registration results of deep closest point method [14]. Right: registration results of the proposed method. Green indicates the transformed source point clouds.

Method	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
ICP	882.564	29.707	23.557	0.084	0.290	0.249
Go-ICP [69]	131.182	11.453	2.534	0.000	0.023	0.004
FGR [70]	607.694	24.651	10.055	0.011	0.108	0.027
PointNetLK [71]	256.155	16.004	4.595	0.000	0.021	0.005
DCP [14]	1.169	1.081	0.737	0.000	0.001	0.001
Proposed	**0.137**	**0.370**	**0.275**	**0.000**	**0.000**	**0.000**

Table 4.2: Registration results with randomly rotated noisy point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-).

when compared to the results trained with all classes (C in Table 4.1), the proposed method showed the lowest errors in all cases while trained with only 20 kinds of classes. The compared methods can be classified into two main categories: non-deep learning based methods [69, 70] and deep learning based methods [14, 71]. Deep learning based methods typically showed better performance than non-deep learning based methods because ICP and its variants (Go-ICP, FGR) often stalled in suboptimal local minima due to the problem's non-convexity [9]. PointNetLK [71] encoded a point cloud to generate a single descriptor using PointNet [8] and estimated a rigid transformation using descriptors of two point clouds. Because the feature extraction method used in PointNet [8] was non-structured, non-translation-invariant, and non-rotation-invariant, the registration results were even worse than some non-deep learning based methods [69, 70]. This

Method	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
ICP	297.080	17.236	8.610	0.007	0.082	0.043
Go-ICP [69]	184.199	13.572	3.416	0.002	0.045	0.015
FGR [70]	40.832	6.390	1.240	0.001	0.038	0.008
PointNetLK [71]	334.67	18.294	9.730	0.008	0.092	0.053
DCP [14]	45.617	6.754	4.366	0.004	0.061	0.040
PRNet [72]	7.355	2.712	1.372	0.000	0.017	0.012
FMR [73]	25.412	5.041	2.304	0.001	0.038	0.016
IDAM [74]	46.950	6.852	1.761	0.003	0.054	0.014
DeepGMR [75]	356.832	18.890	9.322	0.008	0.087	0.056
OMNet [76]	4.322	2.079	0.619	0.000	0.018	0.008
SpinNet [18]	1.355	1.164	0.902	0.000	0.013	0.011
Proposed	**0.741**	**0.861**	**0.440**	**0.000**	**0.008**	**0.004**

Table 4.3: Registration results with randomly rotated and partially sampled point clouds of ModelNet40. The evaluation metrics are mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE) for rotation (R-) and translation (T-).

indicates that the two descriptors generated from randomly rotated point clouds contained significantly different values. DCP [14] generated descriptors using the graph based structured and translation-invariant method (DGCNN [9]) in the feature extraction procedure, reducing registration errors compared to PointNetLK [71]. The feature extraction method used in DCP implies the translation-invariant property but does not imply the rotation-invariant property. This leads to differences in descriptor values according to rotations, resulting in relatively large errors when compared

to the proposed method. The registration results of DCP and the proposed method are visualized in Fig. 4.11. The results of the DCP method [14] showed a small error between the two point clouds. Conversely, the results of the proposed method showed superior matching performance. These results indicate that the proposed descriptor matches the feature-based correspondences between the source and target points in a superior manner compared to DCP [14]. Furthermore, as listed in Table 4.2, the proposed method showed the lowest errors in the registration task with noisy point clouds generated using Gaussian noise. These results indicate that the proposed method can be applied to noisy point clouds, like real-world datasets.

In addition to the registration with fully overlapped point clouds (Table 4.1 and Table 4.2), registration with partially overlapped point clouds was conducted. Inspired by PRNet [72], for given point clouds X and Y, partial scans of X and Y were generated by randomly placing a point in space and computing its 768 nearest neighbors in X and Y, respectively. The overall architecture is similar to the registration architecture, but a rigid transformation was computed using RANSAC [77] instead of the SVD module. Table 4.3 lists the partial registration results. As shown in Table 4.3, the registration errors of the proposed method were lower than all the other methods. PRNet [72] is the extended version of DCP [14] designed to improve registration performance by using an iterative and differentiable (i.e., trainable) point-to-point matching module based on Gumbel-Softmax. By using the improved matching module, PRNet showed reduced registration errors, but still had relatively large errors compared to the rotation-invariant descriptor-based methods (SpinNet [18] and the proposed method). Similarly, PointNet [8] based methods (DeepGMR [75] and OMNet [76]), which

focus on establishing point-to-point correspondences rather than the feature extraction, also showed lower performances. These results demonstrate the importance of the rotation-invariance in the registration task. SpinNet [18] generates rotation-invariant descriptors by building spherical volume grids on aligned points using LRFs. Owing to the rotation-invariance, the method could generate consistent descriptor values from corresponding areas regardless of rotations, and SpinNet showed lower errors compared to the non-rotation-invariant methods. However, there are drawbacks to the method: 1) SpinNet generates descriptors using the volume grids, which consume a significant amount of computational memory. 2) The method aligns points using LRF, but does not consider the sign problem. On the contrary, the proposed method consumes relatively a small amount of computational memory when compared to SpinNet and addresses the sign problem by using the symmetric circular convolution method. As a result, the proposed method significantly reduced registration errors with a simple architecture and less computation memory.

Classification and Segmentation

The proposed method was evaluated using the ModelNet40 [66] and ScanObjectNN [67] databases for classification, and the ShapeNetPart [68] database for part-segmentation. Three kinds of experiments were conducted in the classification and segmentation: 1) training and testing under non-rotational environments (NR/NR); 2) training under azimuthal rotations (around the gravity axis) and testing under arbitrary rotations (ZR/AR); and 3) training and testing under arbitrary rotations (AR/AR). The proposed method was compared with both non-rotation-invariant and rotation-invariant meth-

Method	OA	mIoU	mcIoU	RI
PointNet [8]	89.2	83.7	80.3	X
PointNet++ [46]	90.7	85.1	81.9	X
PointCNN [78]	92.2	86.1	84.6	X
DGCNN [9]	92.2	85.2	85.0	X
KPConv [60]	92.9	86.2	**85.1**	X
ShellNet [79]	93.1	-	-	X
PointTransformer [80]	**93.7**	**86.6**	83.7	X
RIConv [27]	86.5	80.3	75.3	O
Clusternet [28]	87.1	-	-	O
REQNN [12]	83.0	-	-	O
PRIN [81]	-	71.1	67.6	O
TVCG2021 [13]	89.4	82.5	79.4	O
RIConv++ [64]	91.3	-	-	O
Our method	**93.2**	**85.9**	**83.3**	O

Table 4.4: ModelNet40 classification results (i.e. OA) and ShapeNetPart segmentation results (i.e. mIoU). OA, mIoU, and mcIoU indicate the overall accuracy, instance average intersection over union, and class average intersection over union, respectively. RI indicates whether the method is the rotation-robust method or not.

Method	OA (ZR/AR)	OA (AR/AR)	RI
PointNet [8]	16.4	75.5	X
PointNet++ [46]	28.6	85.0	X
PointCNN [78]	41.2	84.5	X
DGCNN [9]	20.6	81.1	X
ShellNet [79]	19.9	**87.8**	X
KPConv [60]	**47.8**	**87.8**	X
PointTransformer [80]	17.2	82.2	X
RIConv [27]	86.5	86.5	O
ClusterNet [28]	87.1	87.1	O
REQNN [12]	83.0	-	O
GCAConv [61]	89.1	89.2	O
RIF [13]	89.4	89.3	O
RIConv++ [64]	**91.3**	91.3	O
Proposed	89.0	**91.6**	O

Table 4.5: Results of ModelNet40 classification. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively.

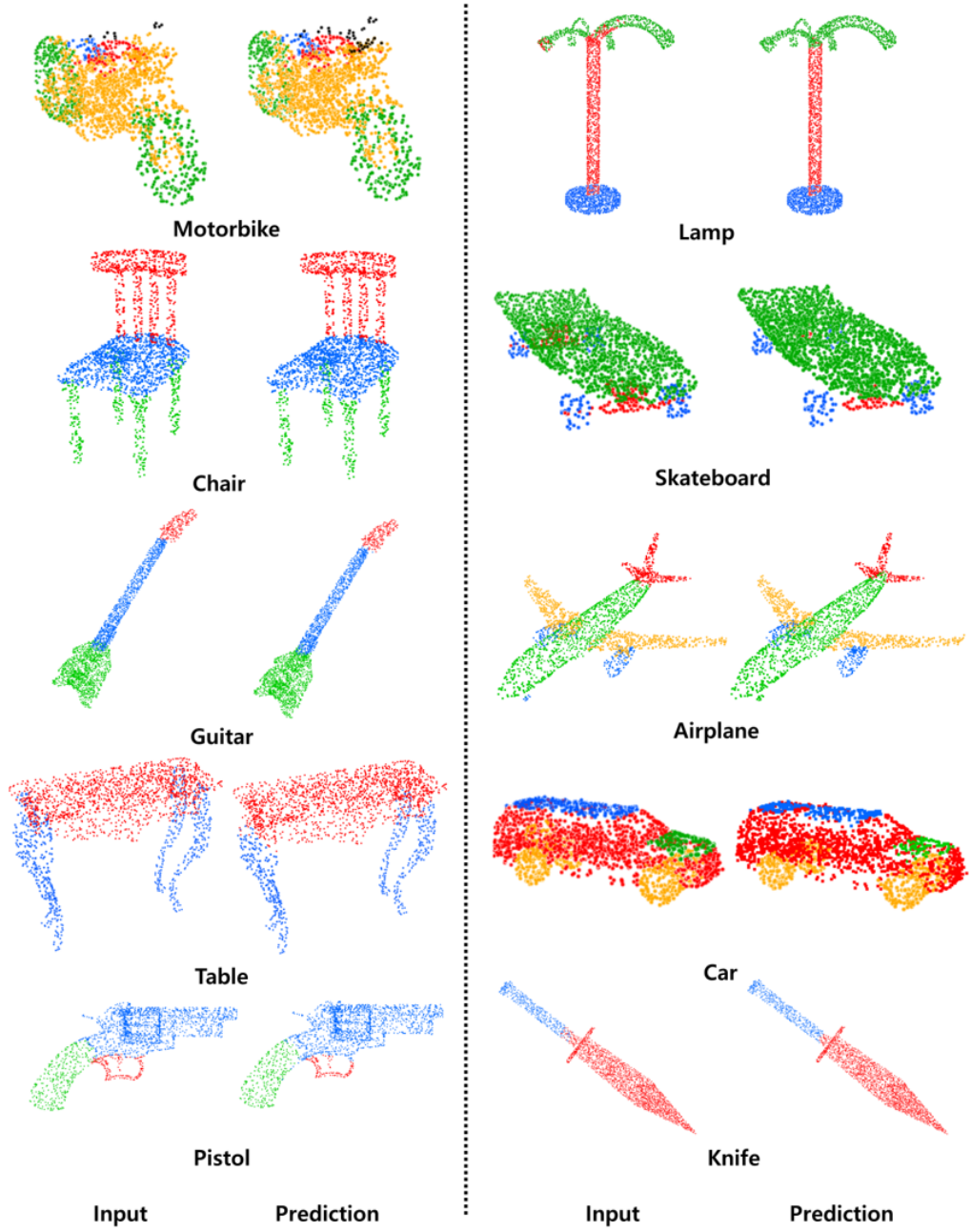

Figure 4.12: Left: source points; Right: part-segmentation results

Method	OA (ZR/AR)	OA (AR/AR)	RI
PointNet [8]	17.1	42.3	X
PointNet++ [46]	15.8	60.1	X
PointCNN [78]	14.9	51.8	X
DGCNN [9]	16.1	**63.4**	X
PointTransformer [80]	**22.1**	43.1	X
RIConv [27]	68.3	68.3	O
GCAConv [61]	69.8	70.0	O
RIConv++ [64]	**80.3**	80.3	O
Proposed	79.4	**80.8**	O

Table 4.6: Results of ScanObjectNN classification. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively.

ods in the classification and part-segmentation tasks.

Typically, rotation-invariant methods (RI is O in Table 4.4) showed inferior performance compared to non-rotation-invariant methods (RI is X in Table 4.4) under non-rotation environments (NR/NR). There are two main reasons for this: Firstly, it is difficult to represent the accurate relationship between points with rotation-invariant features. If the rotation-invariant features (such as relative distance or angle) cannot accurately represent point relationships, points of different topological areas can be represented with the same rotation-invariant features. This means that descriptors generated from different topological areas can have similar values, resulting

Method	mcIoU (ZR/AR)	mcIoU (AR/AR)	RI
PointNet [8]	37.8	74.4	X
PointNet++ [46]	**48.3**	**76.7**	X
PointCNN [78]	34.7	71.4	X
DGCNN [9]	37.4	73.3	X
ShellNet [79]	47.2	77.1	X
KPConv [60]	46.3	75.8	X
RIConv [27]	75.3	75.3	O
PRIN [81]	64.6	67.6	O
GCAConv [61]	77.3	77.1	O
RIF [13]	79.2	79.4	O
RIConv++ [13]	**80.5**	80.5	O
Proposed	77.2	**80.6**	O

Table 4.7: Results of ShapeNetPart segmentation. (-/-) indicates the training and test environments. NR, ZR, and AR indicate no rotation (NR), azimuthal rotations (ZR), and arbitrary rotations (AR), respectively.

in inferior performance compared to non-rotation-invariant methods. Secondly, convolution with more than one point in rotation-invariant order is a challenging problem. Some previous works, such as RIConv [27], Clusternet [28], RIF [13]), encoded each point independently using MLPs to avoid encoding in non-rotation-invariant order. However, the independent point encoding methods cannot provide local geometric information around each point.

To overcome these limitations, the proposed method used the aligned kernels as additional reference points during the feature extraction process to extract accurate rotation-invariant features. Moreover, adjacent kernels were encoded simultaneously using the circular convolution method to provide local geometric information around each kernel, instead of encoding each point (or kernel) independently. As demonstrated in Table 4.4, the proposed descriptor outperforms rotation-invariant methods and shows comparable performance to state-of-the-art non-rotation-invariant methods. Although the proposed method has addressed the drawbacks of rotation-invariant methods, the performance was similar to that of non-rotation-invariant methods. This is because rotation-invariant features are less intuitive compared to non-rotation-invariant features such as coordinates. Non-rotation-invariant features encompass both relative positions among points and their positions within the overall space, while rotation-invariant features only contain information about the relative positions among points. Therefore, surpassing the performance of methods using non-rotation-invariant features poses a challenging problem. However, rotation-invariance is a desired feature for real-world applications. Therefore, it is significant that the proposed method achieves superior accuracy among rotation-invariant methods.

To demonstrate the proposed method's rotation robustness, the experiments with rotated point clouds were conducted in the classification and segmentation tasks. The network was trained with azimuthal rotations (around the gravity axis) (ZR) and arbitrary rotations (AR) and tested with arbitrary rotations (ZR/AR, AR/AR) (Tables 4.5, 4.6, and 4.7). (-/-) indicates which rotational metric was used for training/testing, re-

spectively. The rotation-invariant methods showed consistent performances under rotation environments (ZR/AR, AR/AR) unlike the non-rotation-invariant methods. The non-rotation-invariant methods [9, 79] used point coordinate-based features, which varied according to rotational variations, increasing the burden of the deep learning network by increasing the input patterns. In contrast, the rotation-invariant methods showed consistent performances under rotations by using the rotation-invariant features. However, some methods [27, 28] showed inferior performances than non-rotation-invariant methods [60, 79]. It indicates that inaccurate or ambiguous rotation-invariant features have limitations in representing local geometric structure, and naive convolution methods (MLPs) constrain the network learning. To address this limitation, RIF [13] used the additional reference points (such as the centroid of neighbors and intersection between the query ball and line extended from the origin), but these reference points have a chance to be changed depending on object shape variation, and it may result in insufficient consistency of the descriptors between similar object parts. Similarly, RIConv++ [64] represented accurate point relationships by additionally using distances and angles between neighbors, but one neighbor point variation can affect the other neighbor features. On the contrary, the proposed method resolved the problems by using the kernels as the reference points to represent point relationships accurately and by using the circular convolution method, which encodes the adjacent kernels at once to provide local geometric structure around each kernel.

In Tables 4.5 and 4.7, the proposed method showed comparable performances to the rotation-invariant methods in the (ZR/AR) case. One issue is that the kernel directions may change on the XY-plane based on

the distribution of points. This property increased the burden of the deep learning network by increasing the input patterns and affected the method's performances under the (ZR/AR) case. However, in the cases where training and testing were conducted under arbitrary rotations (AR/AR), the proposed method outperformed both rotation-invariant and non-rotation-invariant methods because the network learned the input patterns that were increased by the kernel direction variations. Figure 4.12 illustrates the sample outputs of the proposed method.

4.4.5 Parameter and ablation study

To verify the effect of the proposed method, several parameter and ablation studies were conducted on the registration task by varying the following parameters: 1) convolution operation, 2) the number of nearest neighbors for each kernel, 3) global context aggregation usage, 4) scale adaptation module usage, and 5) network statistics.

Convolution operation

The proposed method employed two types of operations: the $1 \times 1 \times 1$ channel-wise convolution and the circular convolution. The $1 \times 1 \times 1$ channel-wise convolution operation independently encodes each kernel, thereby limiting its ability to capture local geometric information. In contrast, the circular convolution operation collectively encodes adjacent kernels in a rotation-invariant manner, thus aiding in the capture of local geometric information. As shown in Table 4.8, compared to the network utilizing the $1 \times 1 \times 1$ channel-wise convolution, the network with the circular convolution significantly enhanced performance in terms of both ro-

tation and translation. By encoding adjacent kernels together, the method was able to analyze the surrounding landscapes encapsulated in each kernel. This resulted in improved learning of the structural information between the points. These results demonstrate that circular convolution operations successfully capture geometric features through rotation-invariant multiple kernel encoding. Figure 4.13 illustrates the registration results according to the kernel alignment and convolution methods. As depicted, the network with aligned kernel-based circular convolution demonstrated superior registration results.

Conv method	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
channel-wise	0.040420	0.201046	0.105576	0.000000	0.000149	0.000094
circular conv	**0.017159**	**0.130991**	**0.064475**	**0.000000**	**0.000048**	**0.000027**

Table 4.8: Convolution operation study for ModelNet40 global registration task.

The number of nearest neighbors for each kernel

The performance of the method was further analyzed by varying the number of neighbors for the kernels. As described in Section 4.2.2, the method extracted k-nearest neighbors for each kernel. It then estimated the weighted average location of those neighbors. This location was used as a representative point to capture the local geometry where the neighbors are located. Because the distance-based weights were used for each neighbor, closer neighbors had more impact, while the effect of faraway neighbors was reduced. Experiments were conducted with different numbers of neighbors,

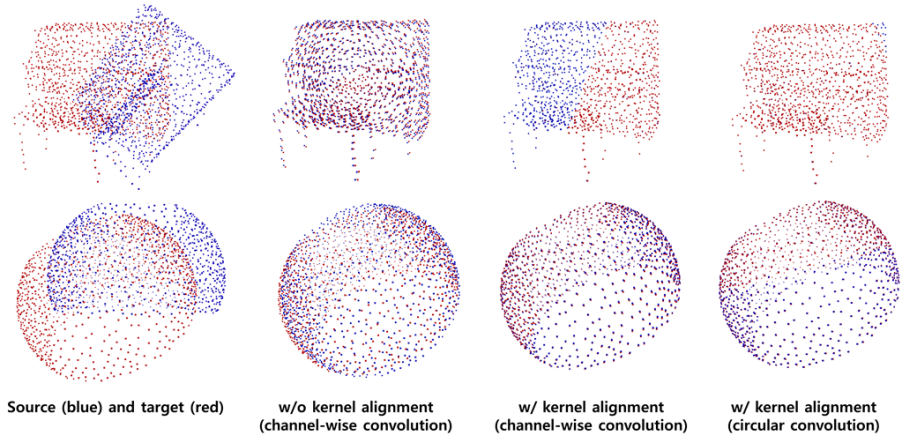

Figure 4.13: Comparison of methods

and the results are listed in Table 4.9. It was observed that the performance of the proposed method is not significantly affected by variations in the number of neighbors. This robustness can be attributed to the approach of estimating the weighted average location, which approximates neighbors, thereby reducing sensitivity to changes in their distribution. This approximation technique, similar to the strategy used by volumetric methods for approximating point clouds, contributes to making the proposed method less sensitive to distribution changes.

KNN	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
2	**0.014517**	**0.120486**	**0.065029**	**0.000000**	**0.000037**	**0.000025**
10	0.017159	0.130991	0.064475	0.000000	0.000048	0.000027

Table 4.9: The number of nearest neighbors study for ModelNet40 global registration task.

Global context aggregation

As the generated descriptors only contain local geometric information around each point, similar descriptor values from monotonous and repeating areas may hinder computation of a rigid transformation in the SVD module. To address this issue, all descriptor values were aggregated based on distances to allow for each descriptor to have a distinct value. To evaluate the effect of the global context, registration was conducted both with and without it (O and X in Table 4.10). The global context yielded improved performance compared to the method without a global context. By aggregating all the local descriptor information, the method can encompass information about the entire point cloud. This approach provides distinctiveness to local descriptors generated from repetitive and similar landscapes and effectively resolves the issue of resemblance among descriptors generated from different locations but similar terrains. These results highlight this method's capability to address the problem of descriptor similarity across diverse locations with similar terrains.

Moreover, to check the efficiency of the method, two global contexts were created, one using descriptors of all points (with 1024 points in Table 4.11), and another using descriptors of uniformly sampled points (with 128 points in Table 4.11). The global contexts generated from both sampled points and all points showed comparable performance. This implies that an effective global context can be generated with a smaller number of points. Because information was gathered from uniformly sampled points, it was possible to accumulate information about the entire point cloud. This is why there was no significant difference in the results when using all points.

Global information	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
X	0.017159	0.130991	0.064475	0.000000	0.000048	0.000027
O	**0.008142**	**0.091766**	**0.046526**	**0.000000**	**0.000047**	**0.000027**

Table 4.10: Global context study for ModelNet40 global registration task.

Point num	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
w/ 128 points	0.007513	0.086675	0.041369	0.000000	0.000047	0.000026
w/ 1024 points	0.008142	0.091766	0.046526	0.000000	0.000047	0.000027

Table 4.11: Global context study with the different number of used points.

Scale adaptation module

In descriptor generation, the extracted neighbors, estimated feature values, and distance-based weights for neighbors depend on the kernel size. As this kernel size is influenced by scale variations, the scale adaptation module was introduced to automatically determine the appropriate kernel size under various scales, as described in Section 4.2.5. To evaluate the scale-robustness of the module, experiments were conducted with and without the scale adaptation module. The network was trained using multiple kernel sizes and tested with different scales (0.50, 1.00, 1.50) of point clouds to demonstrate scale robustness. Table 4.12 presents the results of the different scale tests for each model, with the mean and standard deviations provided. By analyzing features of various scales, the module was able to identify an optimal kernel size that corresponded to informative features for accurately representing the terrain. With this approach, the proposed network demonstrated stable results when using the scale adaptation mod-

Scale adaptation (test scale)	R-MSE	R-RMSE	R-MAE	T-MSE	T-RMSE	T-MAE
X (0.50)	0.084	0.290	0.195	0.007	0.086	0.075
X (1.00)	0.018	0.137	0.085	0.000	0.000	0.000
X (1.50)	0.116	0.341	0.213	0.007	0.086	0.075
X (Total)	0.072 ± 0.040	0.256 ± 0.086	0.164 ± 0.056	0.004 ± 0.003	0.057 ± 0.040	0.049 ± 0.035
O (0.50)	0.017	0.130	0.085	0.000	0.006	0.003
O (1.00)	0.018	0.136	0.085	0.000	0.006	0.003
O (1.50)	0.016	0.127	0.083	0.000	0.005	0.003
O (Total)	0.017 ± 0.000	0.131 ± 0.003	0.084 ± 0.001	0.000 ± 0.000	0.005 ± 0.000	0.003 ± 0.000

Table 4.12: ModelNet40 global registration results with different scales. (#) is the experiments in which networks are trained with the original scale of point clouds and tested with (#) scale of point clouds. Values indicate the mean and standard deviation of the results.

ule. These results indicate that the module determined the optimal kernel size for capturing geometric information, leading to improved performance in global registration.

Network statistics

The space and time complexities of a deep learning network can be estimated using the number of parameters in the network and the number of floating-point operations per sample (FLOPs). Space complexity is typically measured by the number of parameters in the network, and the more complex the network, the more training data it can memorize. FLOPs describe the number of operations required to run a single instance of a given network, with a higher number of FLOPs indicating longer inference time.

Table 4.13 lists the space (number of parameters in the network) and time (FLOPs per sample) complexities of networks for registration on Mod-

elNet40, as well as the processing time for one batch. The complexities of the proposed method were compared to those of PointNet [8] and DGCNN [9]. The results showed that PointNet [8] used the fewest number of parameters and FLOPs and was also the fastest in processing. As PointNet [8] simply processed each point independently, it is the most straightforward and fastest method among the point-based approaches. However, because it can't effectively learn structural geometric information, its performance is the most limited, despite its fast speed. DGCNN [9] found neighbors for each point to build the graph and processed each edge of the graph, resulting in a higher amount of computation than PointNet [8]. In the case of DGCNN, 2D convolution was used to process neighbors for each point, unlike PointNet, which used 1D convolution. This made DGCNN slower than PointNet. Similarly to DGCNN, the proposed method built structures (i.e., kernels) for each point, but it additionally required kernel alignment and rotation-invariant feature extraction for each kernel point. Additionally, it processed adjacent kernel points together when each kernel point was processed, unlike DGCNN, which processed each edge independently. All of these factors affected the complexities of the proposed method and inference speed. Although the proposed method is relatively slower than PointNet and DGCNN, these efforts to achieve rotation-invariant properties are essential for applicability to real-world datasets. Due to its robust rotation-invariant properties, the proposed method can enhance performance even under challenging conditions, such as those presented by real-world datasets (Table 4.6). Rather than reducing speed, it is crucial to perform each task accurately.

Method	PointNet	DGCNN	Proposed (w/o global context)	Proposed (w/ global context)
Params	5,343,049	5,568,905	5,957,545	6,514,857
FLOPs	10,912,399,360	13,223,591,936	15,134,056,448	16,211,992,576
Batch/s	10	3.9	3.3	3.2

Table 4.13: Model statistics on ModelNet40 dataset. The models were experimented with a GTX 1080.

4.5 Discussion

Building rotation- and scale-robust descriptors is essential to achieve consistent and accurate results under various environmental conditions as real-world datasets are not always aligned. While recent studies have made progress in developing rotation-robust descriptors through LRF based alignment or rotation-invariant features, there are still challenges to overcome, such as incorrect LRFs, ambiguous rotation-invariant features, and difficulty in encoding features in a rotation-invariant manner.

In this dissertation, a new rotation-robust descriptor generation method is proposed to address these challenges. The proposed method extracted accurate rotation-invariant features and encoded them in a rotation-invariant and sign-independent manner using the aligned cylindrical kernel. The results of various tasks, such as registration, classification, and part segmentation, demonstrated that the proposed method outperformed previous approaches significantly under rotation variations. The registration task showed a significant reduction in rotation and translation errors, indicating that the proposed descriptors captured salient and corresponding geometric information between two transformed point clouds successfully. The classi-

fication and segmentation results showed that the proposed method is not only suitable for transformed point cloud tasks but also for general purposes. Overall, the proposed method provides a promising solution to the challenge of building rotation- and scale-robust descriptors in point cloud analysis.

Chapter 5

Applying Aligned Kernels for Dense Point Cloud Registration in Real-World Scenarios

5.1 Overview

With the development of 3D acquisition technologies, many researchers have generated various real-world 3D scene models for scene-understanding tasks using 3D-sensing devices [82, 83]. The major challenge in scene modeling is the matching task (that is, registration) caused by the limited view range for each partial scene. The difficulty of registration arises from establishing correspondences based on local descriptor comparisons. In contrast, similar descriptor values are prone to be obtained in many regions, irrespective of their saliency. Moreover, inconsistent descriptor values are obtained from arbitrary rotations; this hinders the establishment of correct correspondence between points. Thus, generating rotation-robust descrip-

tors and selecting salient descriptors is key for accurate registration.

Recently, deep learning-based registration approaches, which typically sample interest points and their neighbors (that is, patches) and build a local descriptor for each patch, have been widely researched, but it is still challenging to build rotation-robust and salient descriptors that contain distinctive information differentiated by geometric terrain. There are two reasons for this: 1) It is hard to achieve rotation-invariant property. Because real-world point clouds obtained from 3D sensors are not aligned (that is, rotated and translated), it is an essential to obtain consistent descriptor values from corresponding rotated regions. To achieve rotation invariance, some methods [11] align the patches using the LRF and build volume-based descriptors from aligned patches, but the estimated LRF is not unique and can be varied. To mitigate the problem of incorrectly estimated LRF, DIP [16], and GeDI [62] use additional networks to align the patches and build descriptors from the patches; however, it still requires training the additional transformation networks, which indicates that it is difficult to obtain rotation-invariance descriptors. 2) Descriptors extracted from the monotonous and repeating areas hinder the registration task. To build point correspondences, it is necessary to identify the most similar local descriptor for each point; however, descriptors from monotonous or repeated regions hinder the procedure. Therefore, featureless descriptors must be excluded to build correspondences using only salient descriptors. Some methods [16, 84] extract salient descriptors by selecting descriptors with high-intensity feature values [16] or using a key-point detection module to find the most suitable points. However, both selection methods are problematic. In the case of [16], there is no guarantee that descriptors from monotonous and

repeating areas have low-intensity feature values, and the selection method of [84] cannot be used for patch-based descriptor approaches because the method requires all the points at once.

In this dissertation, to overcome the limitations, a point cloud registration method with rotation-invariant and dissimilarity-based salient descriptors is proposed. The proposed method employs the aligned cylindrical kernel-based convolution (Chapter 4) to build an adequate rotation-invariant descriptor and propose a dissimilarity-based salient descriptor selection method to exclude descriptors extracted from monotonous and repeating areas.

The major advantage of the proposed method is that the generated rotation-invariant descriptors can deal with incorrect LRF problems (that is, XY-plane rotational variation and normal-axis sign variation) by extracting and encoding cylindrical kernel features in a rotation-invariant manner without an additional transformation network [16, 62]. In addition, the proposed method employs a salient descriptor selection method that compares dissimilarities between descriptors. This enables exclusion of descriptors from monotonous and repeating areas and selection of only the most salient descriptors for the registration process. This study suggests a simple method for building and extracting effective descriptors from 3D point clouds without additional training networks for adaptability. The significance of this study is that the proposed method demonstrated superior generalization performance without adapting training steps. This chapter is organized as follows. Based on the aligned local patches, feature extraction and encoding processes to generate descriptors are presented in Section 5.2. In Section 5.3, salient descriptor selection based on dissimilarities is

introduced.

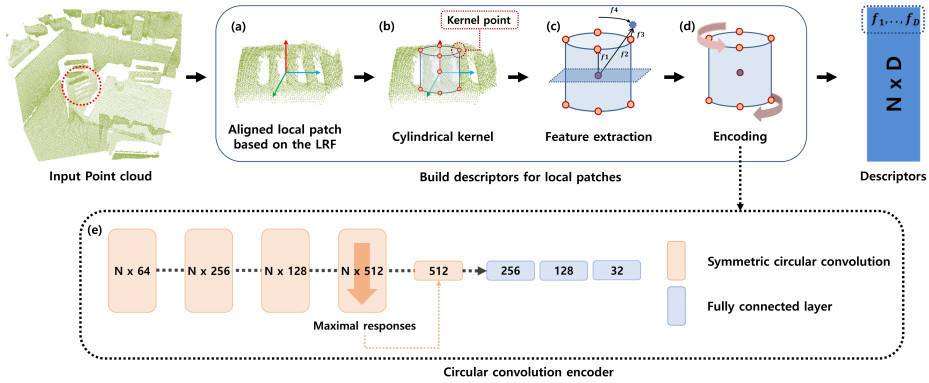

Figure 5.1: Overview of the descriptor generation. (a) Given an interest point, the LRF is first estimated from its neighbor points (patch). The local patch is then aligned using the LRF. (b-c) Rotation-invariant features are computed for each kernel. (d) The extracted kernel features are encoded using the symmetric circular convolution in a rotation-invariant manner. Consequently, the descriptor for the local patch is generated. N is the number of interest points, and D is the dimension of the descriptor. (e) The network architecture encodes extracted features.

5.2 Descriptor generation

The basic concept of the descriptor generation method is to gather rotation-invariant geometric information from cylindrically located kernels to create a rotation-robust descriptor. Figure 5.1 presents an overview of the descriptor generation framework. Given the interest points of the input point clouds, the LRFs are computed from its neighbor points (i.e., patch), and

the local patches are aligned using the LRFs (Fig. 5.1 (a)). Subsequently, a cylindrical kernel is placed on the aligned patches (Fig. 5.1 (b)), and the rotation-invariant features are then extracted for each kernel point (Fig. 5.1 (c)). Each patch's point descriptor is generated by encoding the extracted features (Fig. 5.1 (d)).

5.2.1 Local patch alignment

Given a point cloud $\mathcal{P} = \{x_i \in \mathbb{R}^3\}_{i<N}$, the interest points for registration are selected using Farthest Point Sampling [46]. For each interest point x, the neighborhood $\mathrm{P}_n = \{x_i : ||x_i - x||_2 \leq r\}$ (i.e., patch), defined as the set of neighboring points within a certain radius r, is selected. Based on the points within the patch P_n, the LRF is estimated using the 3Dsmooth-Net method [11]. This method computes three orthogonal vectors for the LRF: the eigenvector corresponding to the smallest eigenvalue of the patch covariance matrix, weighted sum of vectors from the interest point to the neighboring points, and orthogonal vector of the previous two vectors. From the three orthogonal vectors, the rotation matrix R is estimated, and then the patch P_n is aligned using R. Specifically, the aligned patch P_r is obtained by applying the rotation matrix to the points in P_n and subtracting the rotated interest point x, i.e., $\mathrm{P}_r = \mathrm{R}\mathrm{P}_n - \mathrm{R}_x$.

5.2.2 Rotation-invariant feature extraction and encoding

Once the patch is aligned, rotation-invariant features are extracted to capture geometric information based on the feature extraction method introduced in chapter 4. The basic idea is to place a cylindrical kernel on each point and extract geometric information (i.e., features) for each point of the

kernel (Fig. 5.1 (b-c)). Once the features are extracted, the kernel features are encoded using the circular convolution method introduced in chapter 4 to efficiently generate rotation-invariant descriptors (Fig. 5.1 (d)).

The network architecture used to encode the features is illustrated in Fig. 5.1 (e). Four subsequent circular convolutions are applied to the extracted features (from $N \times 64$ to $N \times 512$ in Fig. 5.1 (e)). To generate a single descriptor from the N encoded features, the features are compressed by computing the maximal responses along the columns (from $N \times 512$ to 512). Subsequently, additional fully connected layers are applied to generate the final descriptor (from 256 to 32 in Fig. 5.1 (e)).

The network parameters are learned using the hardest-contrastive loss [36]. This loss function first finds the hardest negatives, which are irrelevant but similar descriptors, for a given positive pair, which are corresponding descriptors. Because the positive pair indicates two corresponding points in the matching procedure, the descriptors of the positive pair should be close to each other. Conversely, the irrelevant descriptors (i.e., negatives) should be far from the positive descriptors. To minimize the distance between corresponding descriptors while maximizing irrelevant descriptors, the method computes the pairwise loss for the quadruplet consisting of the positive pair and the hardest negatives. The loss function is defined as follows:

$$\begin{aligned} L = \sum_{i,j \in P} \{ & (D(d_i, d_j) - m_p))^2 / |P| \\ & + \lambda (m_n - \min_{k \in N} D(d_i, d_k)))^2 / |P_i| \\ & + \lambda (m_n - \min_{k \in N} D(d_j, d_k)))^2 / |P_j| \}, \end{aligned} \tag{5.1}$$

where λ is the weight for negative losses (i.e., the second and third terms), and m_* are constant margin values. Here, P is a set of positive pairs, and

N is a set of randomly sampled points. Then, $\min_{k \in N} D(d_*, d_k)$ indicates the distance between the closest irrelevant descriptor and the $*$-th positive descriptor.

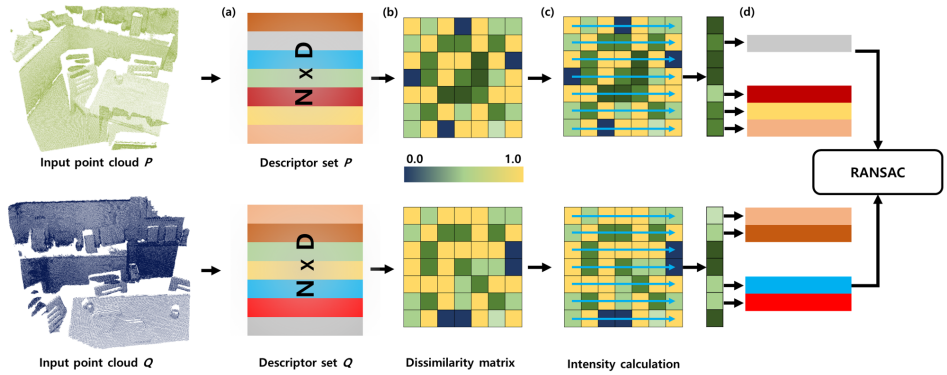

Figure 5.2: Salient descriptor selection process using dissimilarities. (a) Given point clouds, descriptors are generated. (b) For each point cloud, pairwise dissimilarities are estimated using the descriptors. (c) For each point (that is, each row of the dissimilarity matrix), dissimilarity intensity is estimated by calculating L2-norm for each row vector. (d) Salient descriptors, which have high-intensity values (that is, strong dissimilarities between points), are selected for RANSAC-based registration.

5.3 Salient descriptor selection

The interest points are uniformly selected using Farthest Point Sampling [46], so some points may be extracted from monotonous and repeating areas. These areas may result in descriptors that are too similar to one another, making it challenging to distinguish them; thus, dissimilarities

between descriptors are analyzed, and salient descriptors are extracted by selecting descriptors that are not similar to others.

Given two descriptor sets $\mathcal{P}$ and $\mathcal{Q}$ obtained from two point clouds $\mathcal{P}$ and $\mathcal{Q}$ (Fig. 5.2 (a)), a dissimilarity matrix is computed for each set using cosine dissimilarity (Fig. 5.2 (b)).

$$D^*_{i,j} = 1 - \frac{d^*_i \cdot d^*_j}{||d^*_i||||d^*_j||}, \tag{5.2}$$

where d^*_i and d^*_j are the i-th and j-th descriptors of point cloud $*$. Based on the dissimilarity matrix, the dissimilarity intensity for each point is estimated using the L2-norm (Fig. 5.2 (c)).

$$D^*_i = (\sum_j^{N_*} |D^*_{i,j}|^2)^{\frac{1}{2}}, \tag{5.3}$$

where D^*_i is the L2-norm value obtained using the i-th row of $D^*_{i,j}$, and N_* is the number of descriptors of $*$. A higher D^*_i value indicates that the ith descriptor differs from many other descriptors. Then, descriptors, whose dissimilarity intensities are less than the bottom τ_S percent dissimilarity intensity, are excluded. However, the repeated area ratio varied from point cloud to point cloud. Therefore, multiple τ_S values are set and a transformation matrix is estimated for each τ_S value. Then, a transformation matrix with the highest inlier correspondence ratio owing to RANSAC is used. A registration task can be performed using salient descriptors by excluding similar descriptors. The advantage of this selection method is that there is no need to train an additional network to select the salient descriptors.

5.4 Experimental results

5.4.1 Overview

The purpose of the experimental section is to highlight the strengths and weaknesses of the proposed method in registering real-world dense point clouds. This section begins by presenting the configuration of the prepared data and evaluation metrics, followed by a comparison with state-of-the-art methods, and a detailed ablation study of the proposed method.

5.4.2 Data configurations

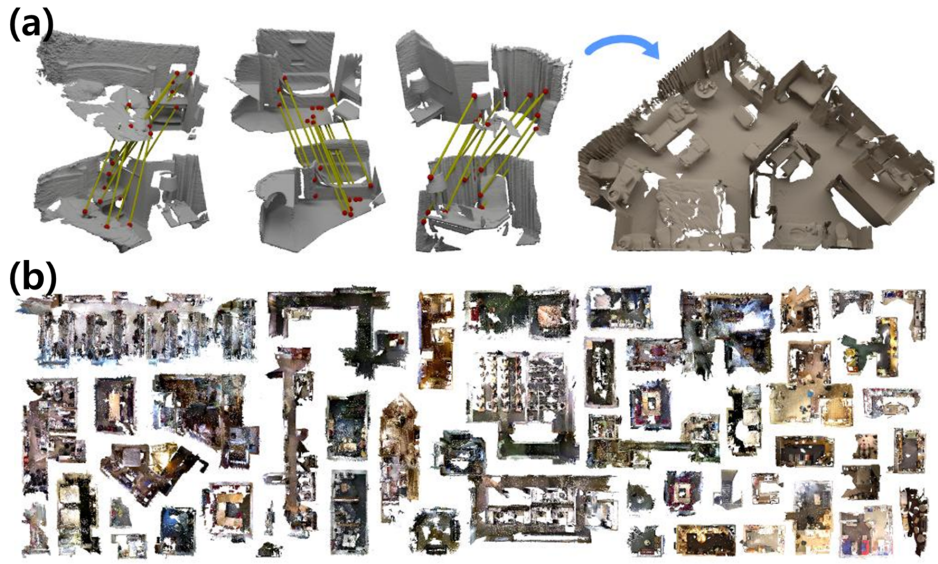

Figure 5.3: (a) Visualization of fragments and a reconstruction of an apartment from SUN3D [15]. (b) Visualization of several RGB-D reconstructions used to train the network.

Indoor 3DMatch dataset

The 3DMatch [82] contains over 200K RGB-D images of 62 different indoor scenes collected from analysis-by-synthesis [85], 7-Scenes [86], SUN3D [15], RGB-D Scenes v.2 [87], and Halber *et al.* [88] (Fig. 5.3). The dataset consists of partially overlapped point cloud fragments of indoor bedrooms, offices, living rooms, tabletops, and restrooms. Among the 62 indoor scenes, 54 were used for training, and eight were used for testing.

Outdoor ETH dataset

The ETH dataset was scanned using an RGB-D scanner, while the 3DMatch dataset was scanned using a laser scanner. It consists of partially overlapped point cloud sequences from four outdoor scenes: Gazebo-Summer, Gazebo-Winter, Wood-Summer, and Wood-Autumn (Fig. 5.4). The "Gazebo" scenes were recorded in a park during summer and winter and mainly consist of grasses and sparse trees with a small paved road. The primary structure in the scenes is a gazebo made of rock walls and a wooden ceiling covered with vines and trees. People usually walk on the path and take a break under the gazebo. The primary motivation for this dataset was to test registration algorithms against semi-structured environments. The "Wood" scenes were recorded during summer and autumn in an environment consisting of vegetation, with a small paved road crossing the wood. The primary motivation for this dataset is to evaluate the robustness of registration algorithms to unstructured environments. To evaluate the generalization ability of the method, it was trained on the indoor 3DMatch dataset and tested on the outdoor ETH dataset.

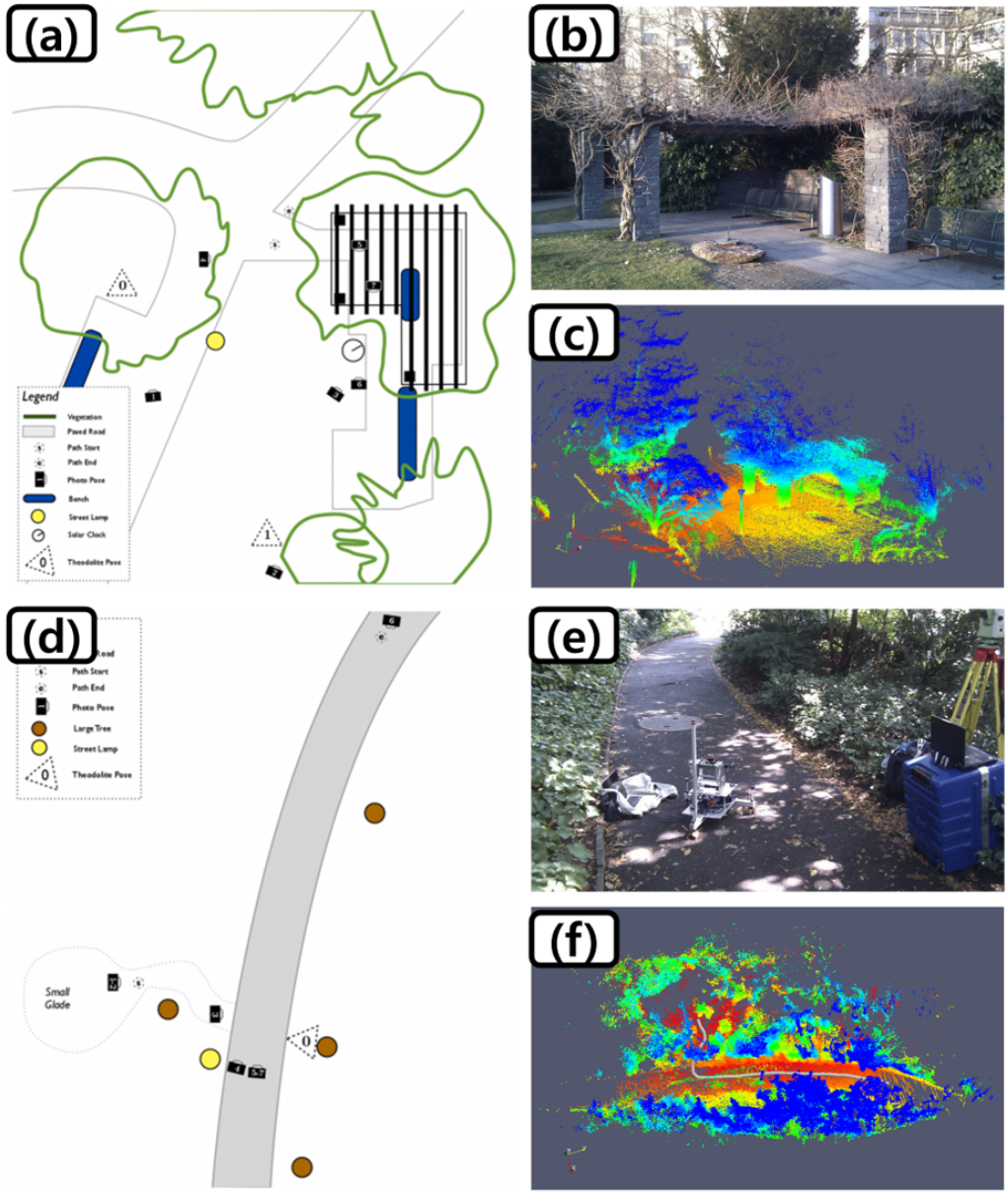

Figure 5.4: (a, d) Environment Topologies of gazebo and wood. (b, e) Contextual Photographs of gazebo and wood. (c, f) Point cloud views gazebo and wood.

5.4.3 Evaluation metric

The quality of the descriptors was evaluated using the feature-matching recall (FMR) metric [89, 90], which can evaluate descriptor quality without applying RANSAC [77]. The basic idea is to compute the recall by averaging the number of matched points across the datasets, given by:

$$FMR = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}([\frac{1}{|\Omega|}\sum_{i,j\in\Omega}\mathbb{I}((x_i - Ty_j) < \tau_1)] > \tau_2), \tag{5.4}$$

where N is the number of ground-truth corresponding point pairs, x_i and y_j are the corresponding point pairs found by applying the nearest-neighbor search in the descriptor space, and Ω, T, and $\mathbb{I}$ are the corresponding set, ground-truth transformation, and indicator functions, respectively. The Euclidean distance threshold τ_1 is set to 10cm to determine whether the matching pair is correct, and the correctly corresponding point pair ratio threshold τ_2 is set to 0.05. In theory, RANSAC finds at least three correct corresponding points with 99.9% confidence using no more than 55,258 iterations [16, 90].

5.4.4 Quantitative analysis

5.4.5 Test on indoor 3DMatch dataset

The proposed method was trained and tested on the indoor 3DMatch dataset. For each fragment, 5000 interest points were sampled using the Farthest Point Sampling method [46]. To demonstrate the rotation robustness, the proposed method was compared with state-of-the-art rotation-invariant methods (LRF-based LMVD [17], SpinNet [18], and GeDI [62], which uses an additional transformation network) on the rotated 3DMatch

Method	Type	3DMatch FMR	3DMatchR FMR	Feat dim	Saliency Selection	R-I
SpinImage [5]	H	0.633 ± 0.106	0.639 ± 0.098	153	X	X
FPFH [4]	H	0.754 ± 0.071	0.767 ± 0.075	33	X	O
USC [30]	H	0.868 ±.052	0.877 ± 0.053	1980	X	O
SHOT [1]	H	0.875 ± 0.034	0.875 ± 0.036	352	X	O
3DMatch [82]	D	0.596 ± 0.073	0.011 ± 0.010	512	X	X
FCGF [36]	D	0.952 ± 0.029	0.953 ± 0.033	32	X	X
D3Feat-rand [84]	D	0.953 ± 0.027	0.952 ± 0.032	32	X	X
D3Feat-pred [84]	D	0.958 ± 0.029	0.955 ± 0.035	32	O	X
CGF [91]	D	0.478 ± 0.094	0.499 ± 0.092	32	X	O
PPF-FoldNet [90]	D	0.718 ± 0.105	0.731 ± 0.104	512	X	O
Equivariant3D [92]	D	0.942 ± 0.040	0.939 ± 0.048	512	X	O
PerfectMatch [11]	D	0.947 ± 0.027	0.949 ± 0.025	32	X	O
DIP [16]	D	0.948 ± 0.046	0.946 ± 0.046	32	O	O
LMVD [17]	D	0.975 ± 0.028	0.969 ± -	32	X	O
SpinNet [18]	D	0.976 ± 0.019	0.975 ± 0.019	32	X	O
GeDi [62]	D	**0.979** ± 0.022	**0.976** ± 0.027	32	X	O
Proposed	D	0.976 ± 0.020	**0.976** ± 0.020	32	O	O
Proposed + TNet [16]	D	0.976 ± 0.022	**0.976** ± 0.022	56	O	O

Table 5.1: Evaluation results for the indoor 3DMatch dataset. The evaluation metrics are Feature-matching recall (FMR) and standard deviation of FMR (std). H and D Type indicate handcrafted and deep learning-based methods, respectively. Feat dim is the descriptor dimension. Saliency Selection indicates whether the method uses salient and reliable descriptor selection. R-I (Rotation-Invariance) indicates whether the method uses the LRF or rotation-invariant features. Results are divided according to the Type (H or D) and R-I values.

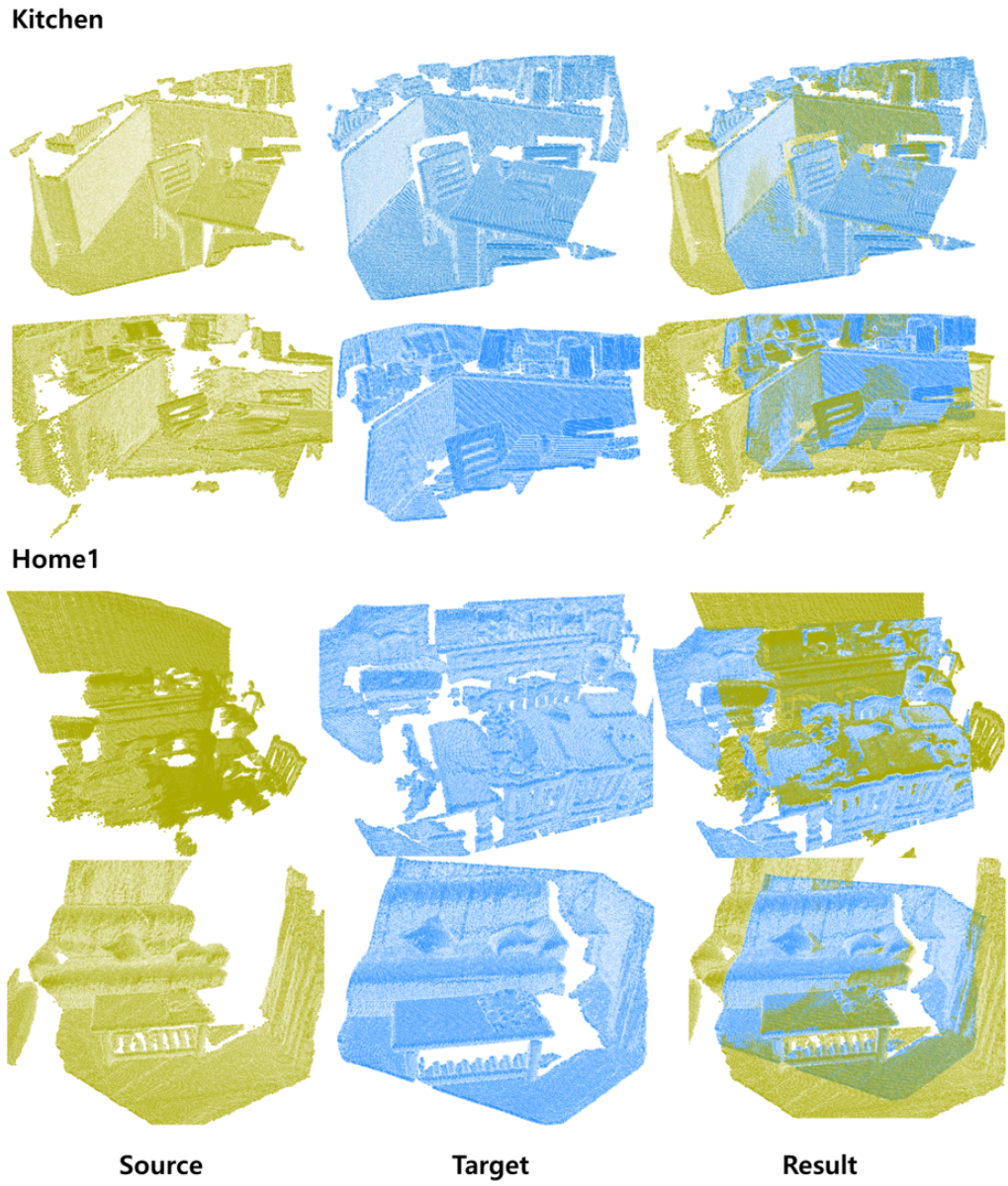

Figure 5.5: Registration results of the proposed method on the indoor 3DMatch dataset.

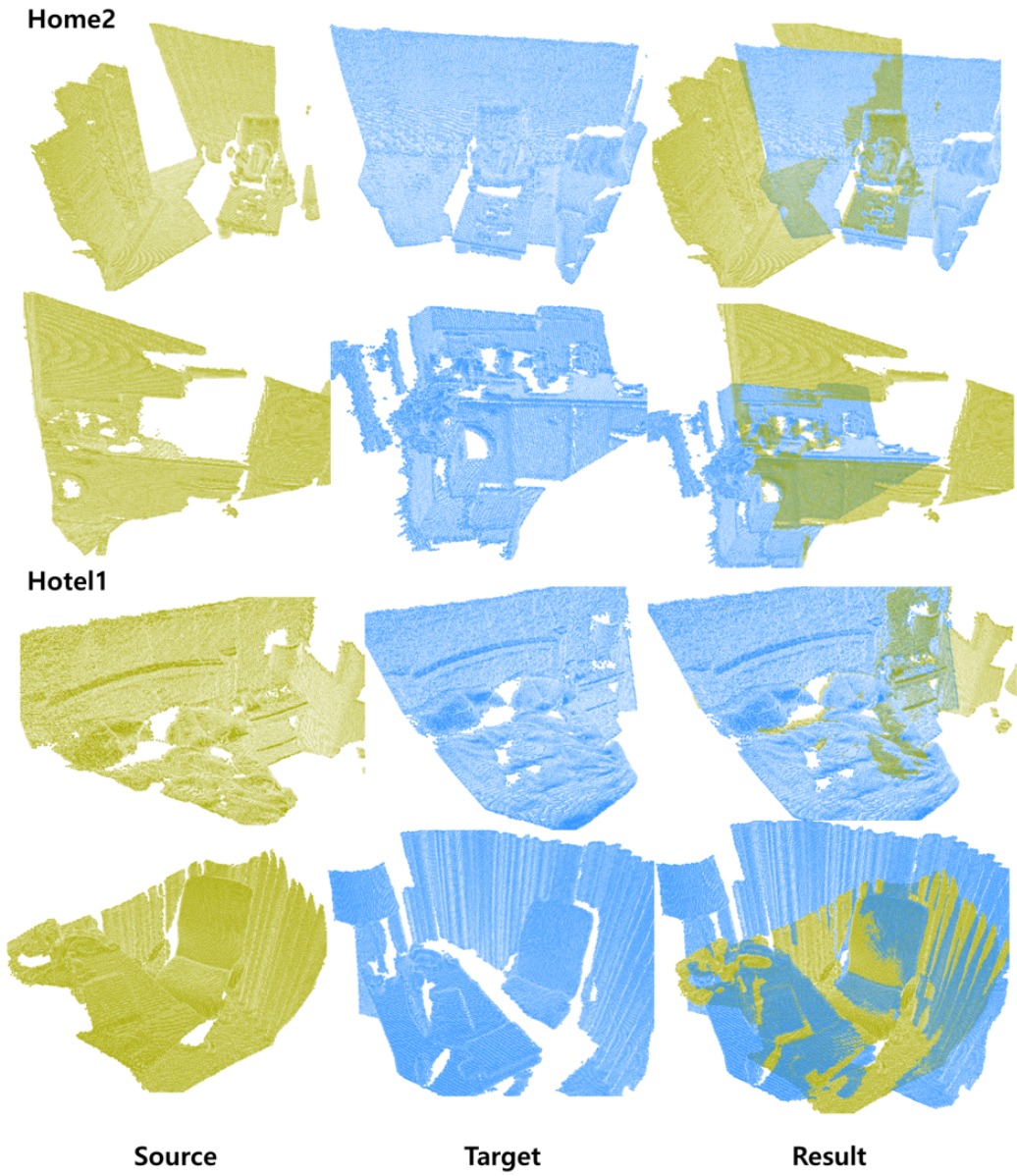

Figure 5.6: Registration results of the proposed method on the indoor 3DMatch dataset.

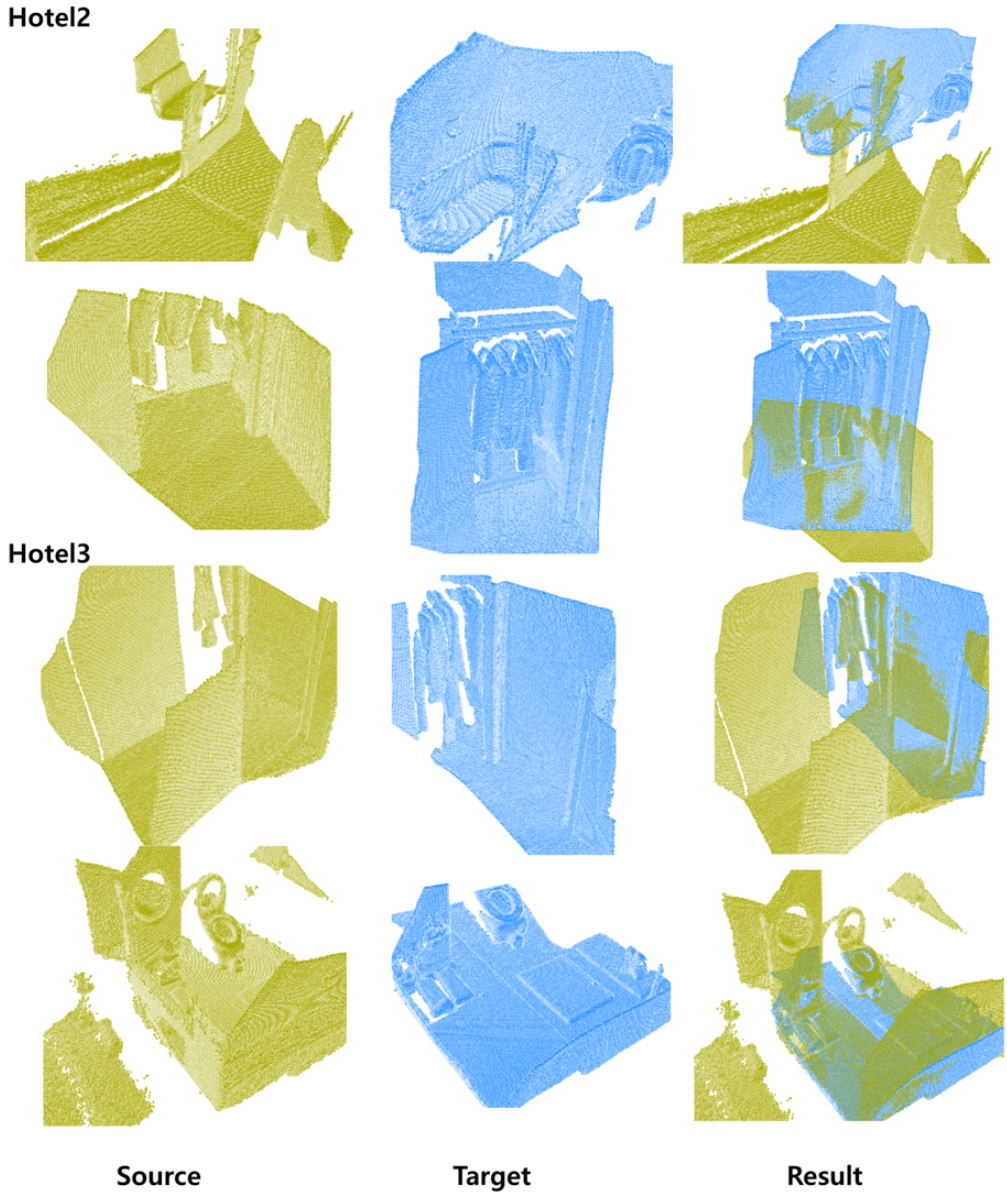

Figure 5.7: Registration results of the proposed method on the indoor 3DMatch dataset.

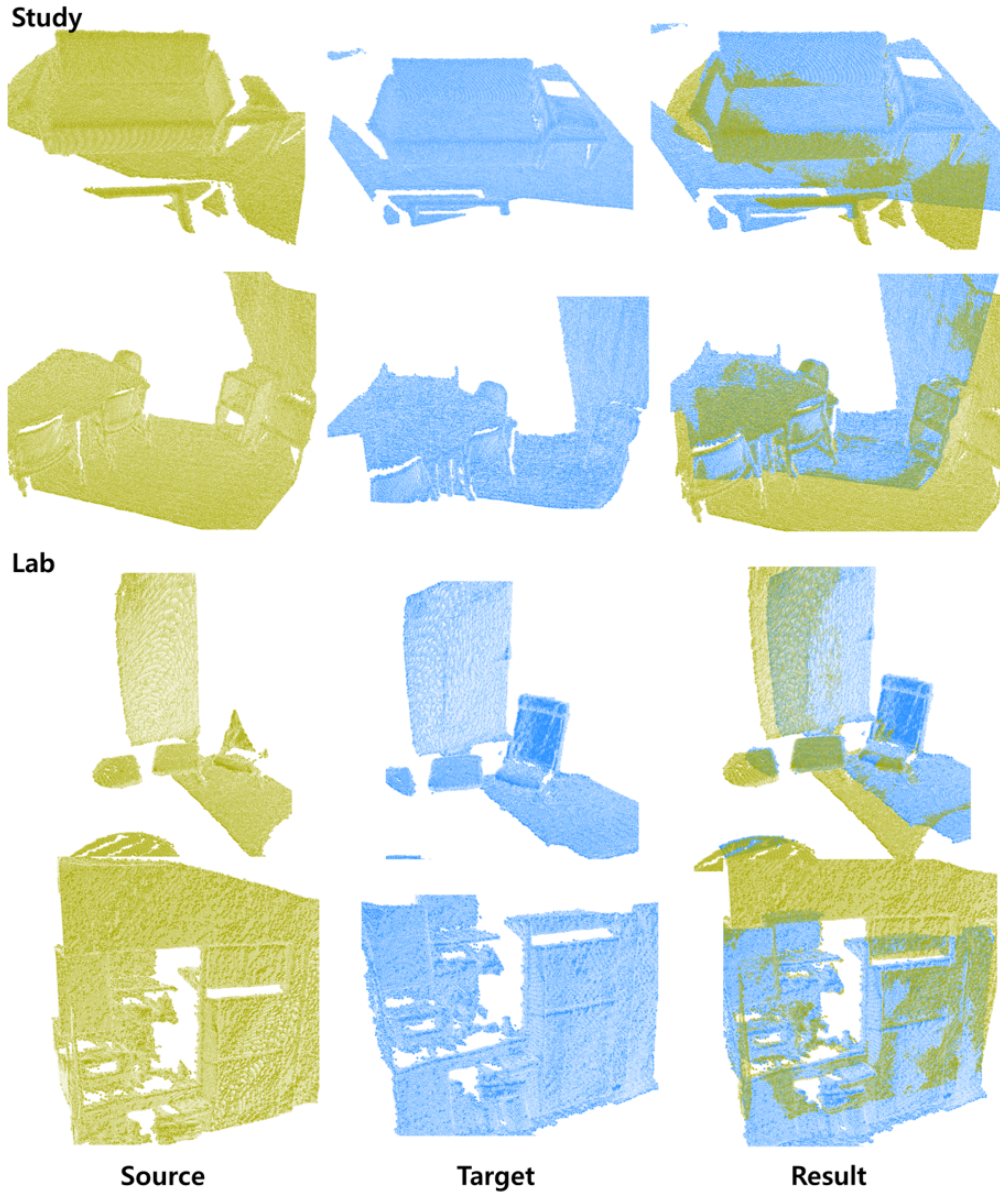

Figure 5.8: Registration results of the proposed method on the indoor 3DMatch dataset.

dataset by applying arbitrary rotations to point clouds (3DMatchR in Table 5.1). Additionally, to demonstrate the effectiveness of the descriptor selection method, the proposed method was compared with key-point selection methods (DIP [16] and D3Feat [84]).

The registration results of the proposed method for the 3DMatch dataset are shown in Figures 5.5, 5.6, 5.7, and 5.8. Table 5.1 presents the second-highest FMR scores of the proposed method on the 3DMatch dataset without rotation (3DMatch) and the highest FMR scores with rotations (3DMatchR). Under rotations, all rotation-invariant methods show consistent performance (labeled O in the R-I column of Table 5.1). Some rotation-variant methods use random rotation data augmentation [36, 84] to achieve rotation invariance. As the terrain patterns of the training and test datasets are similar, the rotation-variant methods with simple data augmentation show consistent performance by learning rotated input patterns. Thanks to the aligned cylindrical kernel-based convolution method, the proposed method achieves rotation invariance and outperforms other methods in both experiments. However, GeDI [62] shows the highest FMR scores in both experiments despite using a simple feature extraction method. The main difference between this and other rotation-invariant methods is that GeDI [62] uses a transformation network to mitigate incorrect LRFs. This means that incorrect LRFs exist because of the occluded and noisy point clouds. Thus, incorrectly estimated LRFs result in limited descriptor performance because they increase the burden of the deep learning network by increasing the input patterns from incorrectly aligned patches. To improve the overall performance, non-salient descriptors were excluded by analyzing dissimilarities between points, and only selected salient descriptors were

used, resulting in comparable performance compared to additional transformation network-based approaches. Moreover, to evaluate the effectiveness of mitigating incorrect LRFs, an additional experiment was conducted using an additional transformation network [16] (Proposed + TNet [16] in Table 5.1). However, due to the large size of TNet, the aligned points using TNet were encoded independently. The encoded point coordinate features were then concatenated with the proposed descriptors, but this concatenation had limited influence on correcting the point correspondences. Therefore, the performances of the proposed method with and without TNet were not significantly different.

5.4.6 Test on outdoor ETH dataset

The proposed method was trained on the indoor 3DMatch dataset and then tested on the outdoor ETH dataset to evaluate its generalization ability. The ETH dataset contains more complex outdoor scenes than the indoor 3DMatch dataset and has a lower point cloud resolution. As a result, the differences between the two datasets make it challenging to achieve good generalization performance for descriptors.

Table 5.2 lists the FMR scores of the outdoor ETH dataset. Rotation-variant methods, such as FCGF [36] and D3Feat [84], show significant performance decrease on the ETH dataset, whereas rotation-invariant methods (LMVD [17], SpinNet [18], and GeDI [62]) demonstrate superior performance. The low generalization performance of rotation-variant methods is attributed to their rotational variance properties. In addition to the LRF, DIP [16], and GeDI [62] used additional transformation networks to mitigate the eventual noisy canonicalization of input points. Despite using sim-

Method	Type	Gazebo Summer	Gazebo Winter	Wood Autumn	Wood Summer	Average	Saliency Selection	R-I
FPFH [4]	H	0.386	0.142	0.148	0.208	0.221	X	O
SHOT [1]	H	0.739	0.457	0.609	0.640	0.611	X	O
3DMatch [82]	D	0.228	0.083	0.139	0.224	0.169	X	X
FCGF [36]	D	0.228	0.100	0.148	0.168	0.161	X	X
D3Feat-rand [84]	D	0.457	0.239	0.130	0.224	0.262	X	X
D3Feat-pred [84]	D	0.859	0.630	0.496	0.480	0.616	O	X
CGF [91]	D	0.375	0.138	0.104	0.192	0.202	X	O
PerfectMatch [11]	D	0.913	0.841	0.678	0.728	0.790	X	O
LMVD [17]	D	0.853	0.720	0.840	0.783	0.799	X	O
DIP [16]	D	0.908	0.886	0.965	0.952	0.928	O	O
SpinNet [18]	D	0.929	0.917	0.922	0.944	0.928	X	O
GeDI [62]	D	0.989	0.965	0.974	**1.000**	0.982	X	O
Proposed	D	**0.995**	0.938	0.965	0.928	0.956	O	O
Proposed + TNet [16]	D	0.984	**0.997**	**1.000**	**1.000**	**0.995**	O	O

Table 5.2: Feature-matching recall (FMR) scores on the outdoor ETH dataset. The network is trained on the 3DMatch dataset and tested on the ETH dataset. H and D Type indicate handcrafted and deep learning-based methods, respectively. Saliency Selection indicates whether the method uses salient and reliable descriptor selection. R-I (Rotation-Invariance) indicates whether the method uses the LRF or rotation-invariant features. Results are divided according to the Type (H or D) and R-I values.

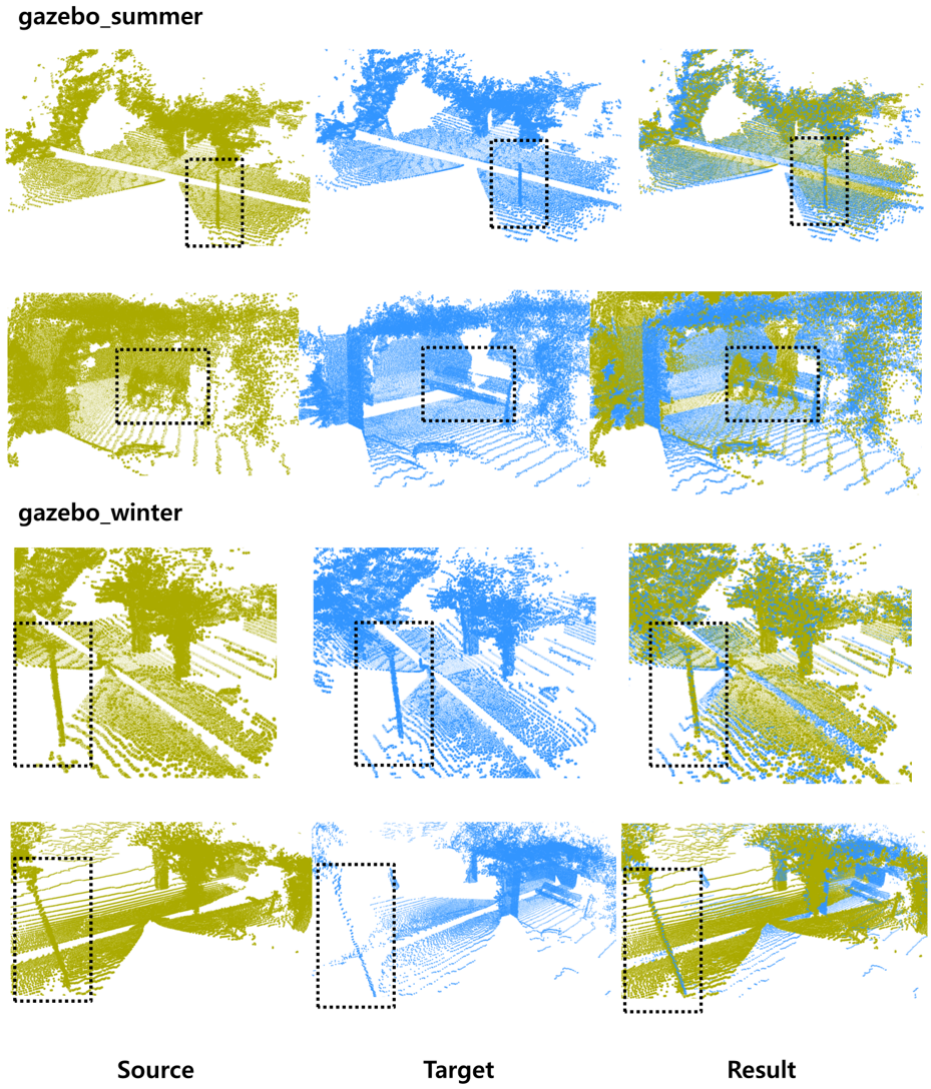

Figure 5.9: Registration results of the proposed method on the outdoor ETH dataset.

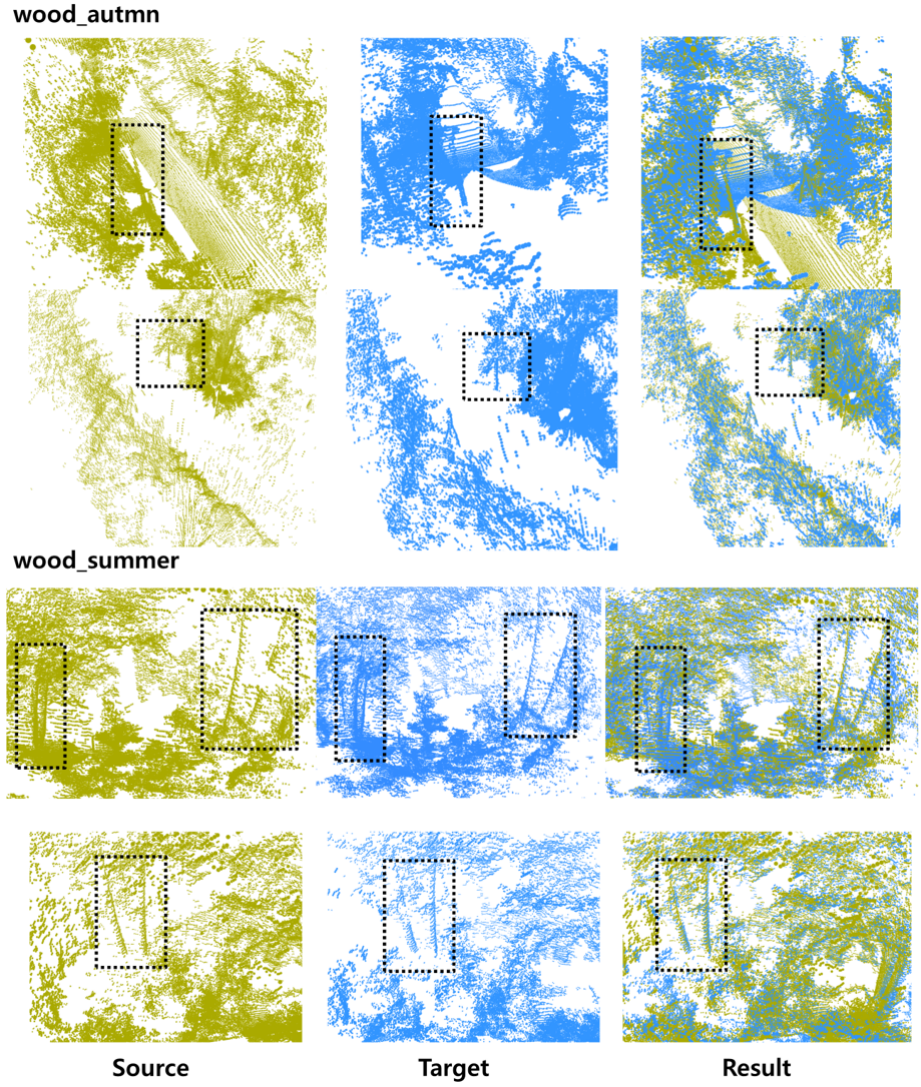

Figure 5.10: Registration results of the proposed method on the outdoor ETH dataset.

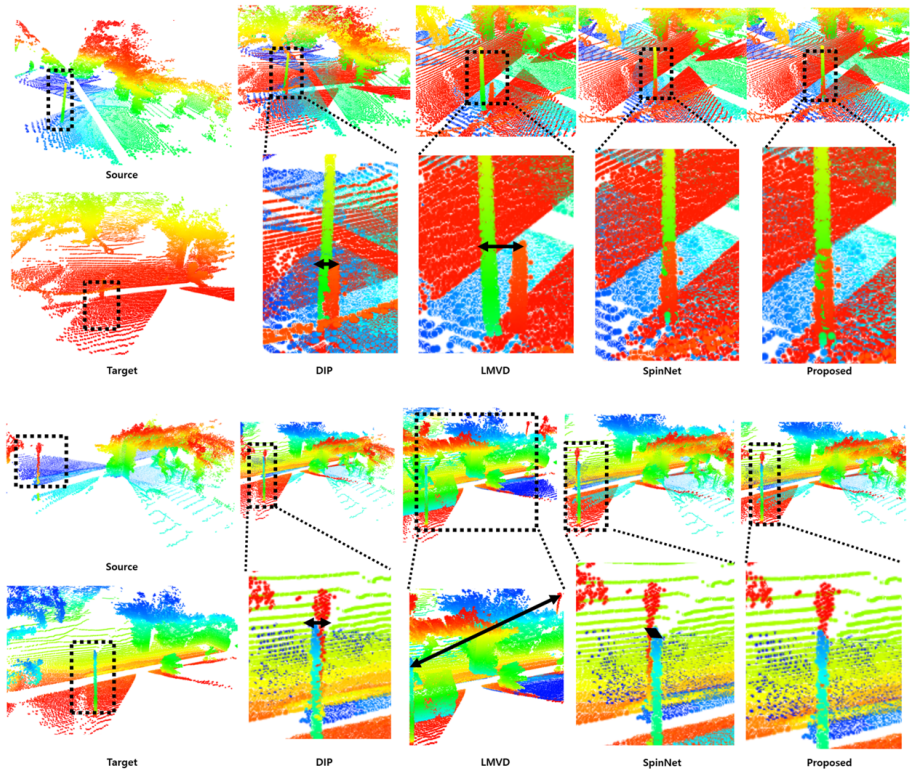

Figure 5.11: Registration results of DIP [16], LMVD [17], SpinNet [18], and the proposed method on the outdoor ETH dataset. The dotted box indicates the characteristic overlap area. The arrow indicates the distance between the source and target point clouds.

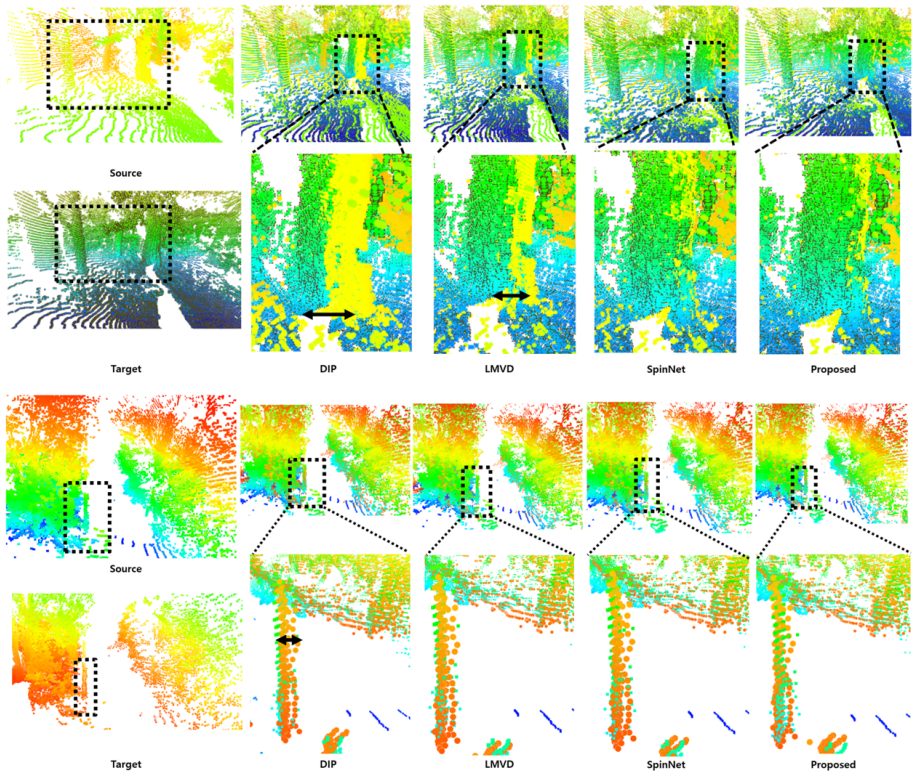

Figure 5.12: Registration results of DIP [16], LMVD [17], SpinNet [18], and the proposed method on the outdoor ETH dataset. The dotted box indicates the characteristic overlap area. The arrow indicates the distance between the source and target point clouds.

ple feature extraction methods (PointNet [8] and PointNet++ [46]), these two methods show superior performance thanks to the additional transformation networks.

As shown in Table 5.2, the proposed method achieves the highest FMR score for the Gazebo-Summer scenario, but for all other scenarios, except for Gazebo-Summer, GeDI [62] outperforms the proposed method. Because the ETH point clouds are complex, occluded, and represented with low resolutions, obtaining correct LRFs is challenging compared to the 3DMatch point clouds. To mitigate incorrect LRFs, DIP [16] and GeDI [62] use additional transformation networks, resulting in significantly better performance compared to other rotation-invariant methods in these experiments. An additional experiment was conducted to investigate cases where patches have incorrect LRFs by using an additional transformation network [16] (Proposed + TNet [16] in Table 5.2). The results show that the proposed method with the additional transformation network [16] achieved the highest FMR scores in the Gazebo-Winter, Wood-Autumn, and Wood-Summer scenarios, indicating that incorrect LRFs affect the descriptor generation process more than the 3DMatch datasets. Nevertheless, the proposed method shows comparable performance with transformation-based methods such as GeDI [62] and outperforms other methods when the transformation network is used. These results demonstrate the method's superior generalization ability on datasets by leveraging its rotation-invariance property and salient descriptor selection. Figures 5.9 and 5.10 illustrate the registration results of the proposed method for the ETH dataset, and figures 5.11 and 5.12 compare the registration results with those of DIP [16], LMVD [17], SpinNet [18], and the proposed method. The results of the

DIP [16], LMVD [17], and SpinNet [18] show relatively large errors between the two point clouds compared to the proposed method (dotted boxes in Figs. 5.11 and 5.12).

Method	Kitchen	Home1	Home2	Hotel1	Hotel2	Hotel3	Study	MIT Lab	Avg
w/o Salient selection	**0.992**	0.981	0.923	0.996	0.971	**1.000**	0.952	0.896	0.964
w/ Salient selection	**0.992**	**0.994**	**0.928**	**0.996**	**0.971**	**1.000**	**0.965**	**0.961**	**0.976**

Table 5.3: Feature-matching recall (FMR) scores of the ablation study of the salient descriptor selection method on the 3DMatch dataset.

Method	Gazebo Summer	Gazebo Winter	Wood Autumn	Wood Summer	Avg
w/o Salient selection	0.962	0.910	0.939	**0.928**	0.935
w/ Salient selection	**0.995**	**0.938**	**0.965**	**0.928**	**0.956**

Table 5.4: Feature-matching recall (FMR) scores of the ablation study of the salient descriptor selection method on the ETH dataset.

5.4.7 Ablation study

To evaluate the effectiveness of the proposed method, ablative experiments were conducted. First, the network was trained on the 3DMatch dataset and evaluated with and without the dissimilarity-based salient descriptor selection method on both the 3DMatch and ETH datasets. Tables 5.3 and 5.4 list the FMR scores for the experiments. As a baseline, descriptors without the salient descriptor selection method were also evaluated (w/o salient selection in Tables 5.3 and 5.4). The results demonstrate that descriptors with the selection method outperform those without the selection

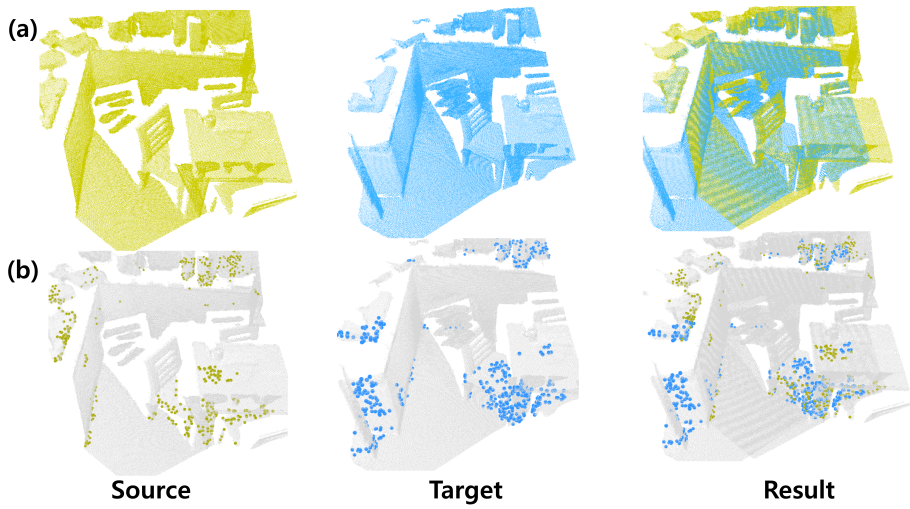

Figure 5.13: Registration results of the proposed method on the indoor 3DMatch dataset. (a) All points are colored yellow (source) and blue (target). (b) Only salient points obtained by using the selection method are colored yellow and blue.

method for most scenes, indicating the effectiveness of the proposed selection method. Salient points extracted using the selection method are shown in Figure 5.13. The figure demonstrates that the proposed dissimilarity criterion (5.3) effectively excludes repeated and monotonous points, such as floor points, while preserving overlapped points around complex terrain shapes, such as chairs and tables.

Second, the proposed method was evaluated on the 3DMatch with different numbers of sampled points. As shown in Table 5.5, the descriptor generated by the proposed method exhibited consistent FMR scores for different numbers of sampled points. GeDI [62] demonstrated the best per-

formance in most of the experiments. The robustness of a different number of points can be mainly attributed to the superior descriptor performance based on improved rotation-invariance owing to the additional transformation network. However, it requires training the network with many convolution channels. Instead of using the transformation network, the proposed method improved the overall performance by excluding non-salient descriptors and achieved relatively higher performance compared with the state-of-the-art methods. In addition, the method shows superior performance compared with D3Feat-pred [84], which uses a key-point detector. These results demonstrate that combining the rotation-invariant descriptor and the selection method improves the robustness for various points differently from GeDI [62].

Third, the proposed method was evaluated on the outdoor ETH dataset by varying the τ_S value in Section 5.3. The FMR scores according to different τ_S values are illustrated in Figure 5.14. A higher τ_S value indicates that more non-salient descriptors are excluded. As shown in the graph, the FMR scores increased as the τ_S value increased, but the scores started to decrease when the τ_S value approached approximately 80%. Because the two point clouds are partially overlapped, salient descriptors exist in non-overlapped and overlapped areas. Thus, excluding descriptors with a strict τ_S value (i.e., a high value) can remove corresponding salient descriptors, leading to a high ratio of non-corresponding descriptors. Hence, it is crucial to determine the optimal τ_S value to maximize the ratio of corresponding salient descriptors.

Method	5000	2500	1000	500	250	Avg	Saliency Selection
PerfectMatch [11]	0.947	0.942	0.926	0.901	0.829	0.909	X
FCGF [36]	0.952	0.955	0.946	0.930	0.899	0.936	X
D3Feat-rand [84]	0.953	0.951	0.942	0.936	0.908	0.938	X
D3Feat-pred [84]	0.958	0.956	0.946	0.943	0.933	0.947	O
SpinNet [18]	0.976	0.975	0.973	0.963	0.943	0.966	X
GeDI [62]	**0.979**	**0.977**	**0.976**	0.972	**0.973**	**0.975**	X
Proposed	0.976	0.975	**0.976**	**0.974**	0.967	0.974	O

Table 5.5: Feature-matching recall (FMR) scores of the ablation study on the 3DMatch dataset with the different number of sampled points.

5.5 Discussion

The task of registration requires the extraction of 3D descriptors from each point cloud to align them accurately. However, building robust and discriminative descriptors for this task remains challenging. In this dissertation, a new method is proposed to build robust and discriminative 3D local descriptors that achieve rotation invariance and exclude non-salient descriptors. The method uses aligned cylindrical-shape kernels and symmetric circular convolution to build rotation-robust descriptors, even in unstable LRF problems. Moreover, it excludes similar descriptors by comparing their dissimilarities and selecting discriminative descriptors, which may hinder registration tasks.

The proposed method was experimented on indoor and outdoor datasets to evaluate its performance. The method shows comparable performance to

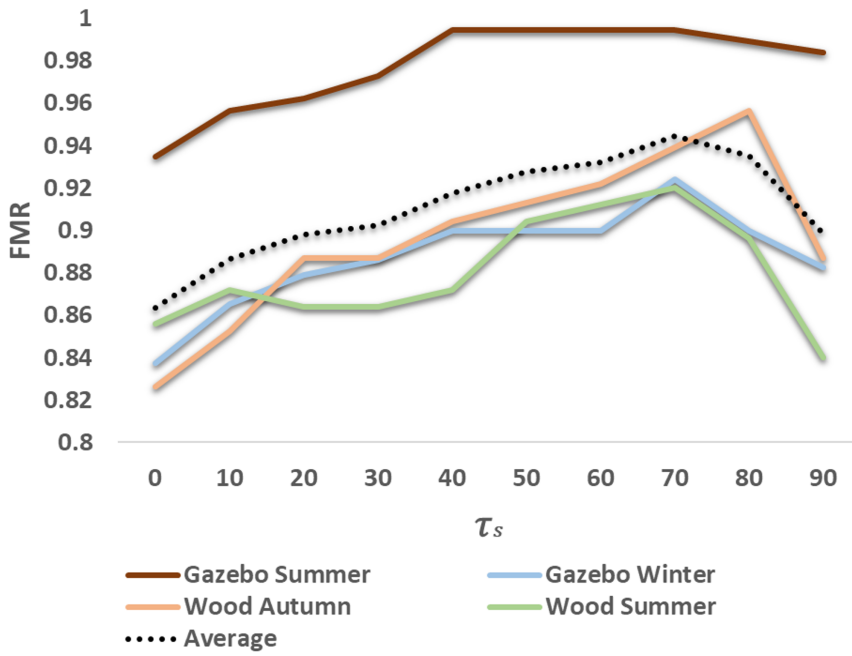

Figure 5.14: Registration results of the proposed method on the outdoor ETH dataset with different τ_S of Section 5.3 (that is, τ_S percentage of similar points are excluded). The horizontal axis and vertical axis indicate τ_S and FMR score.

state-of-the-art methods, even under rotational variances, owing to rotation-invariant features and convolution. These features and convolution reduce the number of input patterns that the network must learn and allow the network to handle a new type of point cloud. However, in most experiments, the transformation network-based methods demonstrated the best performance. This indicates that incorrect LRF estimation significantly affects de-

scriptor performance, and resolving this problem for LRF-based local patch alignment is still necessary. Nevertheless, the proposed method achieved comparable performance to the transformation network-based methods by using a simple descriptor method, which mitigates incorrect LRF problems. Experiments across different domains demonstrate a superior generalization ability due to the combination of the rotation-invariant descriptor and selection method. Additionally, ablation studies on the selection method and the number of sampled points demonstrate that the proposed selection method successfully selects salient descriptors by excluding similar descriptors.

Chapter 6

Conculsion and Future Works

The task of encoding rotation- and scale-robust features is crucial for point cloud representation. The success of various downstream tasks depends on the robustness of these parameters. In this dissertation, an aligned kernel based feature representation is proposed to resolve these limitations. The proposed methods, including kernel alignment, rotation-invariant feature extraction, and kernel-conscious convolution, show superior performance when compared to previous approaches. These findings suggest that the proposed methods improve the stability and the feature representation of the descriptor. Additionally, the proposed scale adaptation and global aggregation methods capture the optimum scale parameter and global geometric features for each local descriptor, respectively. Furthermore, the salient descriptor selection method is successfully employed in the registration task by analyzing the differences between points. This study proposes a simple method for extracting effective descriptors from 3D point clouds

without additional training networks for adaptability.

In future research, it will be essential to develop density-robust descriptors, given that the proposed method is based on point-based approaches. These approaches process each point directly and aggregate geometric information from all the processed points. As a result, variations in density can lead to variations in the aggregated geometric information, potentially impacting the performance of the analysis. Additionally, the proposed method's performance can be further investigated under more challenging environmental conditions, such as non-rigid deformations. These deformations are commonly encountered in various fields like medical imaging, 3D modeling, animation, and robotics. For instance, in medical imaging, handling non-rigid deformations efficiently could lead to more accurate diagnoses and treatment plans. In the case of robotics and autonomous vehicles, robust registration even under constant environmental changes could enhance navigation and perception capabilities. Therefore, these investigations can provide valuable insights into the potential applications and limitations of the proposed method.

Overall, the proposed method's effectiveness in generating rotation- and scale-robust descriptors for point clouds shows promise for future research and practical applications in various fields, including robotics, augmented reality, and autonomous vehicles.

Bibliography

[1] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, 08 2014.

[2] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "Narf: 3d range image features for object recognition," in *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 44, 2010, p. 2.

[3] R. Rusu, N. Blodow, Z. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," 09 2008, pp. 3384–3391.

[4] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[5] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 433–449, 1999.

[6] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 922–928.

[7] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 945–953.

[8] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[9] A. V. Phan, M. L. Nguyen, Y. L. H. Nguyen, and L. T. Bui, "Dgcnn: A convolutional neural network over large-scale labeled graphs," *Neural Networks*, vol. 108, pp. 533 – 543, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608018302636

[10] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 984–993.

[11] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5540–5549.

[12] W. Shen, B. Zhang, S. Huang, Z. Wei, and Q. Zhang, "3d-rotation-equivariant quaternion neural networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer, 2020, pp. 531–547.

[13] X. Li, R. Li, G. Chen, C.-W. Fu, D. Cohen-Or, and P. Heng, "A rotation-invariant framework for deep point cloud analysis," *IEEE transactions on visualization and computer graphics*, vol. PP, 2021.

[14] Y. Wang and J. Solomon, "Deep closest point: Learning representations for point cloud registration," 10 2019, pp. 3522–3531.

[15] J. Xiao, A. Owens, and A. Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 1625–1632.

[16] F. Poiesi and D. Boscaini, "Distinctive 3d local deep descriptors," in *2020 25th International conference on pattern recognition (ICPR)*. IEEE, 2021, pp. 5720–5727.

[17] L. Li, S. Zhu, H. Fu, P. Tan, and C.-L. Tai, "End-to-end learning local multi-view descriptors for 3d point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1919–1928.

[18] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "Spinnet: Learning a general surface descriptor for 3d point cloud registration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 753–11 762.

[19] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3357–3364.

[20] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3d point cloud based object maps for household environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.

[21] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[22] N. Hagbi, O. Bergig, J. El-Sana, and M. Billinghurst, "Shape recognition and pose estimation for mobile augmented reality," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 10, pp. 1369–1379, 2010.

[23] A. P. Placitelli and L. Gallo, "Low-cost augmented reality systems via 3d point cloud sensors," in *2011 Seventh International Conference on Signal Image Technology & Internet-Based Systems*. IEEE, 2011, pp. 188–192.

[24] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 641–656.

[25] T. Pylvanainen, K. Roimela, R. Vedantham, J. Itaranta, and R. Grzeszczuk, "Automatic alignment and multi-view segmentation

of street view data using 3d shape priors," in *Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, vol. 737, 2010, pp. 738–739.

[26] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and J. Zhang, "Performance evaluation of 3d local feature descriptors," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 178–194.

[27] Z. Zhang, B.-S. Hua, D. W. Rosen, and S.-K. Yeung, "Rotation invariant convolutions for 3d point clouds deep learning," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 204–213.

[28] C. Chen, G. Li, R. Xu, T. Chen, M. Wang, and L. Lin, "Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4994–5002.

[29] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," vol. 3, 05 2004, pp. 224–237.

[30] F. Tombari, S. Salti, and L. Di Stefano, "Unique shape context for 3d data description," 01 2010.

[31] Z.-C. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3d modelling of novel objects from a single view," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3700–3705.

[32] C. Ruizhongtai Qi, H. Su, M. NieBner, A. Dai, M. Yan, and L. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," 06 2016, pp. 5648–5656.

[33] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6620–6629.

[34] B. Graham, "Spatially-sparse convolutional neural networks," 2014.

[35] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," 06 2019, pp. 3070–3079.

[36] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 8957–8965.

[37] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1355–1361.

[38] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.

[39] L. Zhou, S. Zhu, Z. Luo, T. Shen, R. Zhang, M. Zhen, T. Fang, and L. Quan, "Learning and matching multi-view descriptors for registra-

tion of point clouds," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 505–522.

[40] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[42] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao, "Gvcnn: Group-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 264–272.

[43] X. Wei, R. Yu, and J. Sun, "View-gcn: View-based graph convolutional network for 3d shape analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1850–1859.

[44] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "Pointweb: Enhancing local neighborhood features for point cloud processing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5565–5573.

[45] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-

scale point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 108–11 117.

[46] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5099–5108. [Online]. Available: http://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature-learning-on-point-sets-in-a-metric-space

[47] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-net: Point cloud upsampling network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2790–2799.

[48] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 8778–8785.

[49] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 770–779.

[50] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "Std: Sparse-to-dense 3d object detector for point cloud," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1951–1960.

[51] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 918–927.

[52] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked dynamic graph cnn: Learning on point cloud via linking hierarchical features," *arXiv preprint arXiv:1904.10014*, 2019.

[53] K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8160–8171.

[54] G. Qian, A. Abualshour, G. Li, A. Thabet, and B. Ghanem, "Pu-gcn: Point cloud upsampling using graph convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 683–11 692.

[55] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[56] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolu-

tions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[57] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," 03 2018.

[58] F. Groh, P. Wieschollek, and H. Lensch, *Flex-Convolution: Million-Scale Point-Cloud Learning Beyond Grid-Worlds*, 05 2019, pp. 105–122.

[59] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3197517.3201301

[60] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 6410–6419.

[61] Z. Zhang, B.-S. Hua, W. Chen, Y. Tian, and S.-K. Yeung, "Global context aware convolutions for 3d point cloud understanding," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 210–219.

[62] F. Poiesi and D. Boscaini, "Learning general and distinctive 3d local deep descriptors for point cloud registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[63] Y. Rao, J. Lu, and J. Zhou, "Spherical fractal convolutional neural networks for point cloud recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 452–460.

[64] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "Riconv++: Effective rotation invariant convolutions for 3d point clouds deep learning," *International Journal of Computer Vision*, vol. 130, no. 5, pp. 1228–1243, 2022.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[66] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1912–1920.

[67] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1588–1597.

[68] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *ACM Transactions on Graphics (TOG)*, vol. 35, pp. 1 – 12, 2016.

[69] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-icp: A globally optimal solution to 3d icp point-set registration," *IEEE Transactions on Pat-*

tern Analysis and Machine Intelligence, vol. 38, no. 11, pp. 2241–2254, 2016.

[70] Q.-Y. Zhou, J. Park, and V. Koltun, *Fast Global Registration*, 2016, pp. 766–782. [Online]. Available: https://app.dimensions.ai/details/publication/pub.1034541096

[71] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust efficient point cloud registration using pointnet," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7156–7165.

[72] Y. Wang and J. Solomon, "Prnet: Self-supervised learning for partial-to-partial registration," in *NeurIPS*, 2019.

[73] X. Huang, G. Mei, and J. Zhang, "Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 366–11 374.

[74] J. Li, C. Zhang, Z. Xu, H. Zhou, and C. Zhang, "Iterative distance-aware similarity matrix convolution with mutual-supervised point elimination for efficient point cloud registration," in *European conference on computer vision*. Springer, 2020, pp. 378–394.

[75] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," in *European conference on computer vision*. Springer, 2020, pp. 733–750.

[76] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng, "Omnet: Learning overlapping mask for partial-to-partial point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3132–3141.

[77] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[78] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *NeurIPS*, 2018.

[79] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1607–1616.

[80] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 259–16 268.

[81] Y. You, Y. Lou, Q. Liu, Y.-W. Tai, L. Ma, C. Lu, and W. Wang, "Pointwise rotation-invariant network with adaptive sampling and 3d spherical voxel convolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 717–12 724.

[82] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning the matching of local 3d geometry in range scans," in *CVPR*, vol. 1, no. 2, 2017, p. 4.

[83] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms," *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.

[84] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai, "D3feat: Joint learning of dense detection and description of 3d local features," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6359–6367.

[85] J. Valentin, A. Dai, M. Nießner, P. Kohli, P. Torr, S. Izadi, and C. Keskin, "Learning to navigate the energy landscape," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 323–332.

[86] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2930–2937.

[87] K. Lai, L. Bo, and D. Fox, "Unsupervised feature learning for 3d scene labeling," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 3050–3057.

[88] M. Halber and T. Funkhouser, "Structured global registration of rgb-d scans in indoor environments. 2 (3), 9 (2016)," *arXiv preprint arXiv:1607.08539*.

[89] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 195–205.

[90] H. Deng, T. Birdal, and S. Ilic, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," in *ECCV*, 2018.

[91] M. Khoury, Q.-Y. Zhou, and V. Koltun, "Learning compact geometric features," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 153–161.

[92] R. Spezialetti, S. Salti, and L. D. Stefano, "Learning an effective equivariant 3d descriptor without supervision," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6401–6410.

초록

최근 3차원 센서의 발전으로 지형 정보를 담은 포인트 클라우드 데이터가 활용 가능해져, 포인트 클라우드 분석 기술은 로봇 공학, 자율 주행 및 증강/혼합 현실 등 다양한 분야에서 필수적인 연구 분야가 되고 있다. 포인트 클라우드 객체 간 대응 관계를 찾거나 포인트 클라우드 지형 정보 분석 등 다양한 응용을 위해서는 포인트 클라우드로부터 핵심적인 정보를 추출하는 것이 필수적이다. 최근 인공지능 기반의 3차원 포인트 클라우드 분석 기술들이 많이 연구되고 있지만, 회전 불변성 속성을 고려하지 않아 주어진 포인트 클라우드가 임의로 회전되어 있는 경우에는 출력값이 일정하지 않아 일반화 성능이 좋지 않다는 단점이 있다.

본 논문에서는 이러한 문제를 해결하기 위해 포인트 클라우드의 회전에 강건한 디스크립터 생성 방법을 제시한다. 제안된 방법의 주요 아이디어는 포인트 클라우드의 각 점마다 주변 점들의 분포를 분석하고, 그 분포에 따라 구조화된 추가 점 (커널)들을 정렬하여 배치하는 것이다. 실린더 모양의 커널을 사용하여 정렬이 잘못되는 경우를 보완하고, 포인트 클라우드의 점들 간 관계성을 표현하기 위한 특징값을 추출 시, 정렬된 커널 구조의 점들을 같이 사용함으로써 회전에도 일관적 (회전 불변)이면서 정확하게 관계를 표현하는 특징값들을 추출한다. 또한, 추출한 특징값들을 회전 불변 방식으로 인공지능 방법으로 처리한다. 회전 불변성 외에도 포인트 클라우드 크기 변화에 대한 견고성을 향상시키기 위해 포인트 클라우드를 분석하여 최적의커널 크기를 분석 및 사용한다. 생성한 디스크립터는 유사도를 비교하여 특징 있는 디스크립터만 선택하여 분석 작업에 사용한다.

제안하는 디스크립터 방법은 3차원 포인트 클라우드에서 분류, 분할, 정합

성능을 평가하기 위해 생성된 벤치마크 데이터 세트에서 실험하였으며, 실제 응용 가능성을 평가하기 위해 3차원 센서로 촬영하여 생성된 실내 및 실외 데이터 세트에서도 실험하였다. 실험 결과, 제안하는 방법은 등록 작업에서 최신 방법에 비해 우수한 성능을 보였고, 무작위 회전 환경에서의 분류 및 부분 분할 작업에서도 우수한 성능을 보였다.

주요어: 포인트 클라우드 분류, 분할, 정합, 회전 강건 디스크립터
학번: 2020-35803