V2PNet: Voxel-to-Point Feature Propagation and Fusion That Improves Feature Representation for Point Cloud Registration

Han Hu, *Member, IEEE*, Yongkuo Hou, Yulin Ding, Guoqiang Pan, Min Chen, and Xuming Ge

***Abstract*—Point-based and voxel-based methods can learn the local features of point clouds. However, although point-based methods are geometrically precise, the discrete nature of point clouds negatively affects feature learning performance. Moreover, although voxel-based methods can exploit the learning power of convolutional neural networks, their resolution and detail extraction may be inadequate. Therefore, in this study, point-based and voxel-based methods are combined to enhance localization precision and matching distinctiveness. The core procedure is embodied in V2PNet, an innovative fused neural network that we design to perform voxel-to-pixel propagation and fusion, which seamlessly integrates the two encoder–decoder branches. Experiments are conducted on indoor and outdoor benchmark datasets with different platforms and sensors, i.e., the 3DMatch and Karlsruhe Institute of Technology and Toyota Technological Institute datasets, with the registration recall of 89.4% and the success rate of 99.86%, respectively. Qualitative and quantitative evaluations demonstrate that V2PNet has shown improvements in semantic awareness, geometric structure discernment, and other performance metrics.**

***Index Terms*—3-D feature point, 3-D metric learning, point cloud registration, sparse convolution.**

I. Introduction

POINT cloud registration aims to obtain the complete 3-D information of an entire scene [1] and is a prerequisite in real-world applications, such as 3-D reconstruction [2] and architectural inspection [3]. Registration is typically realized by detecting the salient keypoints in a scene and then enriching these keypoints with distinctive descriptors (or features) [4]. The positions of the detected keypoints must be accurately defined to correctly estimate the optimal transformation matrix for a pair of point clouds [5]. Moreover, a descriptor must be able to unambiguously capture semantic information during feature matching. However, in urban and indoor environments consisting of human-made objects, these two requirements are contradictory: High-precision positioning requires local information, whereas contextual information is needed to distinguish repeating patterns in human-made environments and thus capture semantic information [6]. Therefore, the detection of 3-D features remains challenging.

Feature extraction methods based on deep learning have been effective in image matching [7] and point cloud registration applications [8]. However, unlike 2-D images, point clouds are 3-D and are characterized by disorder, inhomogeneous spatial densities, and irregularities in locations and neighborhoods, which limit point cloud-based feature learning [9]. Moreover, although deep convolutional neural networks (CNNs) are commonly used for image processing, they cannot be directly applied to point clouds [10]. At present, the following two point cloud-based learning strategies are used: 1) the point-based learning method, which is derived from PointNet [11] and 2) the voxel-based learning method, which extends CNNs from 2-D images to 3-D space [12]. Despite advancements in these two methods, each has limitations that must be addressed to achieve high performance, as follows.

1) *Feature learning capability of point-based methods:* In point-based methods, the neighborhood for a point is searched using k-nearest neighbor (KNN) or ball query methods. The features are projected to latent space through a simple multilayer perceptron (MLP) and then aggregated (typically through max-pooling). While point-based methods are advantageous in their ability to learn features directly from point clouds and maintain precise position information, their weak feature learning capability is a major limitation due to the irregularity of point clouds. Moreover, obtaining contextual information through order-invariant pooling can pose difficulties in encoding background information. Advanced networks such as kernel point convolution (KPConv) [13] and PointConv [14] use an analog convolution strategy, but the irregularity of point clouds negatively affects their feature learning performance.
2) *Accuracy of voxel-based methods:* The limitations of point clouds can be overcome by clustering unordered point clouds into regularly distributed voxels, which can be

Manuscript received 22 March 2023; revised 2 May 2023; accepted 17 May 2023. Date of publication 22 May 2023; date of current version 12 June 2023. This work was supported in part by the National Key Research and Development Program of China under Project 2022YFF0904400, in part by the National Natural Science Foundation of China under Projects 42230102, 42071355, and 41871291, and in part by the Sichuan Science and Technology Fund for Distinguished Young Scholars under Project 22JCQN0110. *(Corresponding author: Yulin Ding.)*

Han Hu, Yongkuo Hou, Yulin Ding, Min Chen, and Xuming Ge are with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, Sichuan 611756, China (e-mail: han.hu@swjtu.edu.cn; houyongkuo@my.swjtu.edu.cn; dingyulin@swjtu.edu.cn; minchen@home.swjtu.edu.cn; xuming.ge@swjtu.edu.cn).

Guoqiang Pan is with the Equipment Project Management Center, Chinese People's Armed Police Force, Beijing 100161, China (e-mail: 281517817@qq.com).

Codes are made public at https://github.com/houyongkuo/V2PNet.

Digital Object Identifier 10.1109/JSTARS.2023.3278830

processed using 3-D CNNs [12]. Moreover, although point clouds typically occupy only the 2-D manifold surface of 3-D objects rather than their entire solid geometries, sparse 3-D convolution [15] can be applied to process voxels, as CNNs can effectively aggregate and propagate contextual information and thus exhibit strong feature-learning capabilities. Voxels exhibit significant structural features in contrast to the discrete distribution of point clouds, and their regular receptive field of convolution allows for advantageous feature learning. However, voxelization can be interpreted as downsampling, which may lead to a loss of geometric accuracy and surface information. Moreover, voxel resolution sensitivity limits its capability to meet the high-precision requirements of subsequent registration tasks.

To address the limitations of point-based and voxel-based methods, we design V2PNet, an innovative fused neural network in which voxel features are propagated to point features. The resulting point-based neural network preserves accurate point-location information, allowing descriptors to identify subtle differences. Sparse convolution is used, as it does not require complex neighborhood search operations, such as KNN, and can effectively aggregate contextual information to increase convolution efficiency and learning capability. Our method introduces a novel point-voxel feature description for registration tasks, which balances the geometric fidelity of point clouds and the sparsity of voxels. This sets it apart from previous methods and has resulted in significant improvements over them. Specifically, V2PNet first records a map between voxel-to-point indices. Next, it feeds forward the point clouds and voxels into the following two branches: 1) a point-based architecture modeled on KPConv [13] and 2) a voxel-based UNet modeled on MinkowskiNet [16], Next, after acquiring the point-wise and voxel-wise values, it propagates the voxel features to the original points and fuses them through direct concatenation. Finally, it uses an MLP to map the dimension to the desired magnitude.

The key contribution of this study is the innovative introduction of a novel fusion strategy that combines point-based and voxel-based feature descriptions for point cloud registration tasks. This approach overcomes the limitations of single strategies by combining the advantages of preserving surface information in point-based methods and the regular receptive field of voxel-based methods. Specifically, our V2PNet network propagates voxel features to point features while retaining accurate point-location information, resulting in improved contextual information, increased convolution efficiency, and learning capability. Our method is fundamentally different from previous approaches that solely use either point-based or voxel-based architectures without a fusion strategy. To the best of our knowledge, this is the first time that point and voxel features have been fused for feature extraction and description in point cloud registration tasks. The rest of this article is organized as follows. Section II presents a brief review of related studies. Section III describes our method. Section IV describes the experiments conducted using the indoor-scene 3DMatch and outdoor-scene Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) datasets to evaluate the performance of our method. Finally, Section V concludes this article.

II. Related Works

This section discusses the existing 3-D feature learning and extraction methods, focusing on the following neural networks used in our method: 1) point-based neural networks; 2) 3-D convolutional networks; and 3) point-voxel hybrid neural networks.

A. Point-Based Neural Networks

Point-based 3-D neural networks provide intuitive and flexible representations for learning 3-D geometric features. These networks use raw point clouds as input, without converting them to other formats. Key 3-D neural networks include PointNet [11] and PointNet++ [17], which have a simple, efficient, and powerful point-based 3-D feature encoder–decoder structure consisting of an MLP and max pooling layers, and can be applied to most point cloud processing applications. Several registration methods have been developed based on these networks. For example, point pair feature network (PPFNet) [18] is a permutation-invariant network inspired by PointNet that realizes 3-D point cloud registration by using complete sparsity on the original point cloud. However, a satisfactory feature-matching rate is difficult to obtain with PPFNet as it is not fully rotation-invariant [19]. PointNet Lucas and Kanade [20] treats PointNet as a learnable "imaging" function for learning its representations and uses the lucas & kanade (LK) algorithms that explicitly encode the registration process to obtain a result. However, the registration results are not robust to noise and are limited to synthetic objects. Point cloud registration network [21] is robust to noise but mainly targets simple artificial objects. 3DFeat-Net [22] applies a weakly supervised network based on PointNet to jointly learn 3-D point cloud keypoint detectors and descriptors but cannot effectively use contextual semantic information [23].

KPConv [13] is a novel point-convolution method that assigns a weight matrix to a kernel point and defines a space. D3Feat [24] aims to detect the location of keypoints based on the KPConv architecture. This framework extracts the features of each point and fuses the detector and descriptor networks. In this manner, D3Feat overcomes the problem of low robustness exhibited by keypoint detectors in 3DFeat-Net, which focuses on learning only feature descriptors.

B. 3-D CNNs

Existing 3-D CNNs are based on traditional dense convolution, and several recent examples apply sparse convolution. In the first reported application of a 3-D CNN to voxel grids, [12] classified objects in light detection and ranging (LiDAR) point clouds by combining supervised and unsupervised training techniques. Huang and You [25] used a 3-D CNN based on voxelization to design effective algorithms for labeling complex point-cloud data. Methods that use rectangular grids and dense representations have also gradually emerged, in which an empty area is represented by zero or a signed distance function [26]. 3DSmoothNet [19] introduces the concept of smoothed density value voxelization and can fully convolute standard deep-learning library layers, thereby decreasing the sparsity of an input voxel grid and saving network capacity for learning highly descriptive features. Additionally, several 3-D CNNs are based on density maps. PointConv [14] uses the inverse density scale to reweigh the continuous function learned by an MLP. However, when 3-D CNNs are directly applied to dense voxel grids derived from initially sparse point clouds, large memory costs are incurred for encoding empty space. Moreover, computational complexity grows cubically with respect to voxel grid resolution, although fine details are required only on object surfaces.

In the field of sparse 3-D convolution, 3DMatch [27] is a self-supervised feature learning method that randomly selects keypoints, aggregates keypoints and their surrounding points into voxels, transforms them into a truncated distance function, and then feeds them into a Siamese network for training. The dataset provided by 3DMatch is widely used in point-cloud registration domains. MinkowskiEngine [16] is a powerful tool for sparse convolution that has promoted the development of voxel-based feature extraction methods. Its sparse convolution strategy was used to establish the first point-cloud feature extraction network that uses full convolution, fully convolutional geometric features (FCGF) [28], which has been employed to develop several point-cloud registration methods, such as DeepGlobalRegistration [29], 3D_multiview_reg [30], and you only hypothesize once [31]. SpinNet [32] uses spherical voxelization to enhance rotational invariance.

C. Point-Voxel Hybrid Neural Networks

To overcome the limitations of the above-mentioned neural networks, many neural networks that fuse point-based and voxel-based features have been developed in recent years. These methods can be divided into the following two categories: 1) point-wise-level methods [33] and 2) voxel-level methods [34]. Privacy-preserving and verifiable CNN [10] combines the advantages of point-based neural networks (i.e., low memory consumption) and voxel-based 3-D convolution networks (i.e., high regularity and locality). In addition, given the hardware and efficiency requirements of single and hybrid models, researchers have combined point-based and voxel-based networks for feature learning. For example, hybrid voxel-point representation [34] effectively and efficiently integrates both types of networks into a single 3-D representation. A memory module is used to alleviate the computational cost and obtain a hybrid 3-D representation in the form of a pseudoimage that allows efficient localization of 3-D objects in a single stage. PrimitiveNet [35] combines locally supervised encouraged learning of the embeddings and discriminators describing local surface properties with point-voxel neural networks for scene segmentation. Point-voxel transformer [36] fuses two types of features with a transformer to overcome the problem of point-based transformers spending considerable time constructing irregular data.

However, despite the potential of point-voxel hybrid neural networks for point cloud scene segmentation and object detection [37], they have not been applied to point cloud registration.

III. Methodology

A. Overview and Problem Setup

1) Process Flow: As shown in Fig. 1, the inputs in the training stage of V2PNet include two point-aligned point clouds and matching indices based on a ball-query with a search radius determined by the average point distance. Next, each individual set of point clouds is fed into the dual branches of V2PNet, and the local geometric representations of the points are embedded into the latent feature space (a 32-D space in this study) of the point-based and voxel-based branches. An MLP is then used to concatenate and downscale the two feature representations to obtain point-wise 32-D feature descriptors, in addition to the corresponding saliency scores. Finally, a match index drives learning losses to back-propagate through the two networks and satisfy the requirements for repetitive detection scores and distinctive feature descriptors. In the testing stage, only one point cloud is used as the input, and V2PNet outputs both the 32-D features and corresponding scores for each point. Nonlocal minimum suppression is used to enforce the even distribution of the keypoints, and a desired number of keypoints is specified based on the descending order of the scores.

2) Problem Setup: The inputs for V2PNet are two aligned point clouds $(\mathcal{P}, \mathcal{P}')$ and corresponding point-matching indices $E \in \mathbb{Z}^{2\times|I|}$. Each point cloud is represented by N point coordinates $p \in \mathbb{R}^{N\times 3}$ and corresponding original features $x \in \mathbb{R}^{N\times C_I}$. The original features may be colors from red, green, blue and depth (RGB-D) scans $(C_I = 3)$ or intensities $(C_I = 1)$ from LiDAR point clouds. A unity placeholder may be used to focus on the coordinates of points, i.e., $x = \mathbb{I}^N$. For the training matches E, the dimension $E[0]$ is indexed into the source point clouds $\mathcal{P}$, and the other dimension $E[1]$ is indexed into the target point clouds $\mathcal{P}'$. In addition, a voxel size δ must be determined, based on the density of the point clouds. Several key operators of V2PNet are described in the following sections.

Voxelizer: $V(\lfloor p/\delta \rfloor, x) \to v \in \mathbb{Z}^{M\times 3}, x_v \in \mathbb{R}^{M\times C}, m \in \mathbb{Z}^N$ The voxelizer $V(\cdot)$ transforms the quantized coordinates $[p/\delta] \in \mathbb{Z}^{N\times 3}$ ($\lfloor\cdot\rfloor$ denotes the floor operation to integer) and the corresponding features x to M voxels v $(M \leq N)$ and the corresponding voxel features x_v. For voxels with multiple points, the average or maximum values of the features are used. The mapping between the consecutive voxel indices $[0, M - 1]$ and corresponding quantized coordinates v is managed by vector m (Section III-C). The mapping is used to convert projected voxel features back to the original points.

Dual-Branch Networks: $\Theta_K(p, x) = x_K \in \mathbb{R}^{N\times C}$ and $\Theta_V(v, x) = x_V \in \mathbb{R}^{M\in C}$. We introduce the following two branches to learn the local descriptors: 1) the point-based network Θ_K and 2) the voxel-based network Θ_V. Both networks use UNet [38] architecture that is ported to the representations of point clouds and sparse voxels. The two networks encode the spatial characteristics and other input

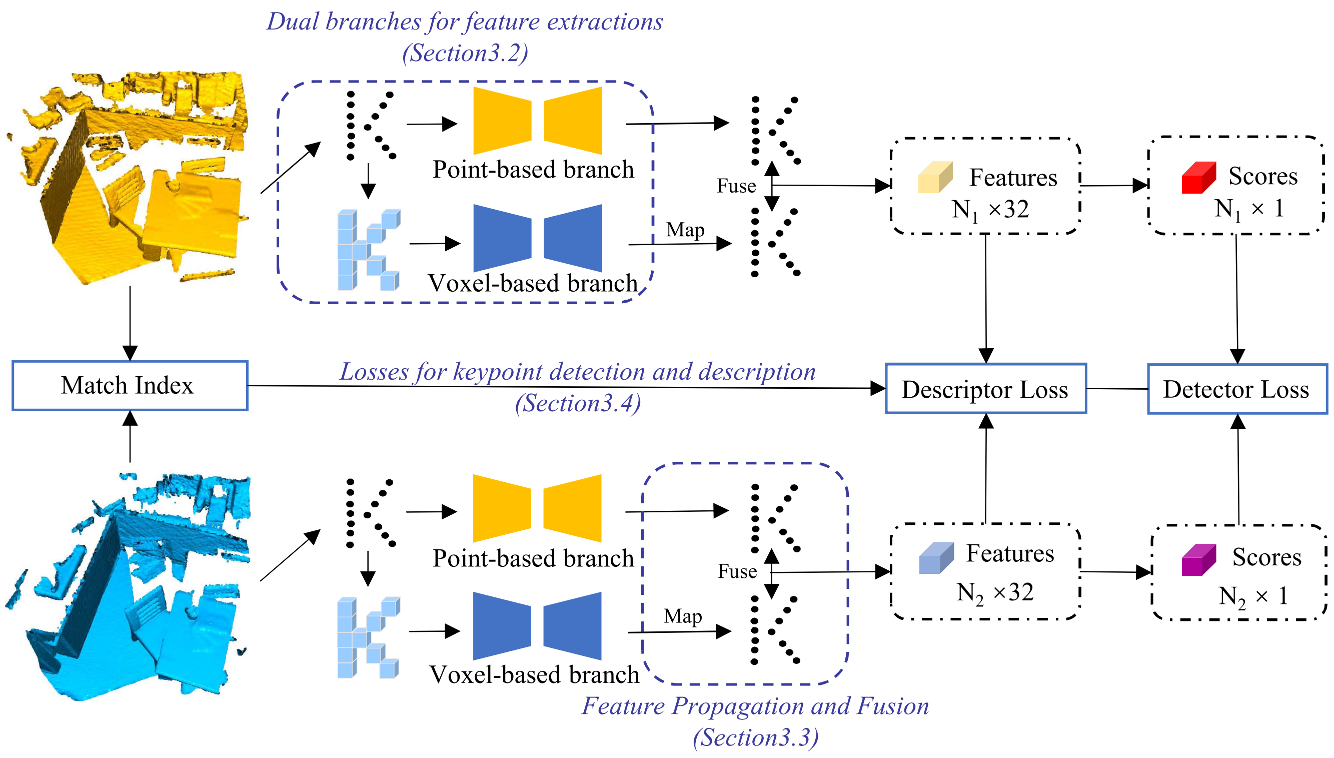


Fig. 1. Architecture of V2PNet. The raw point cloud imported into the network is divided into two branches. First, the point-based feature extraction branch performs dense convolution to extract point-level features. Then, the voxel-based feature extraction branch applies sparse convolution to extract the voxel features and map them to points. Finally, the fused features extracted from the two branches are used to calculate the saliency score and losses.

features into feature spaces x_K and x_V, as discussed in Section III-B.

Detector and Extractor Head: $\Theta_H(x) = f \in \mathbb{R}^{N\times C}, s \in \mathbb{R}^N$. A simple MLP layer converts the fused features of each point into the local descriptors f for the feature extractor s and the corresponding saliency scores. We design two sets of loss functions to enforce the distinctiveness of the local descriptors and the repeatedness of detectors, as described in Section III-D.

B. Dual Branches for Feature Extraction With Point-Based and Voxel-Based Networks

1) Dense Convolution in the Point-Based Branch: Many encoder–decoder architectures based on PointNet++ have been devised for point-based networks. We use KPConv [13] as the backbone for point-based dense feature extraction. This method uses kernel points for images, with these points being analogous to the gridded kernels of a CNN. Because the relative coordinate offsets of N_r neighbor points and K kernel positions do not coincide, linear interpolation is performed to distribute the features of each point to the kernel positions. For completeness, we introduce the convolution operation of KPConv. First, the relative offsets Δp of the neighboring coordinates for each point are gathered. Then, the K kernel weights are distributed to each point using linear interpolation, as follows:

$$K\left(\Delta p\right) = \sum_{k<K} h\left(\Delta p, p_k\right) W_k \tag{1}$$

where $h(\cdot)$ is a linear correlation between the kernel point coordinates and offsets; and p_k and W_k are the position and learned weights for the kth kernel, respectively.

Because the density of the point clouds is not homogeneous, each point has a different number of neighbors [39]. Thus, as the performance of the original implementation of KPConv is affected by varying point density, we replace the direct summation with a density normalization term during the aggregation of neighboring points, as shown in

$$x' = \frac{1}{N_r}\sum_{i<N_r} K_i x_i. \tag{2}$$

In addition, the ball-query is used to collect N_r neighboring points to address the nonuniform sampling problem. Then, multiple KPConv layers are stacked, as shown in Fig. 2, with the points downsampled with doubled voxel sizes δ^l at each consecutive layer l. In addition, the correlation distances and the radius of the ball query are defined by δ^l and $2.5\delta^l$, respectively. Three layers are used instead of four layers, as this enables the focus to be on the local geometric characteristics, which is important for feature matching.

2) Sparse Convolution in the Voxel-Based Branch: In the voxel branch, the point clouds are sampled into a lattice grid in 3-D space and inherently ordered, and integer indices are used to explicitly define the neighborhood as a sparse octree. Therefore, a 3-D sparse CNN and corresponding spatial pooling operators are directly used. Specifically, MinkowskiNet [16], an auto-differential library supporting sparse tensors, is used

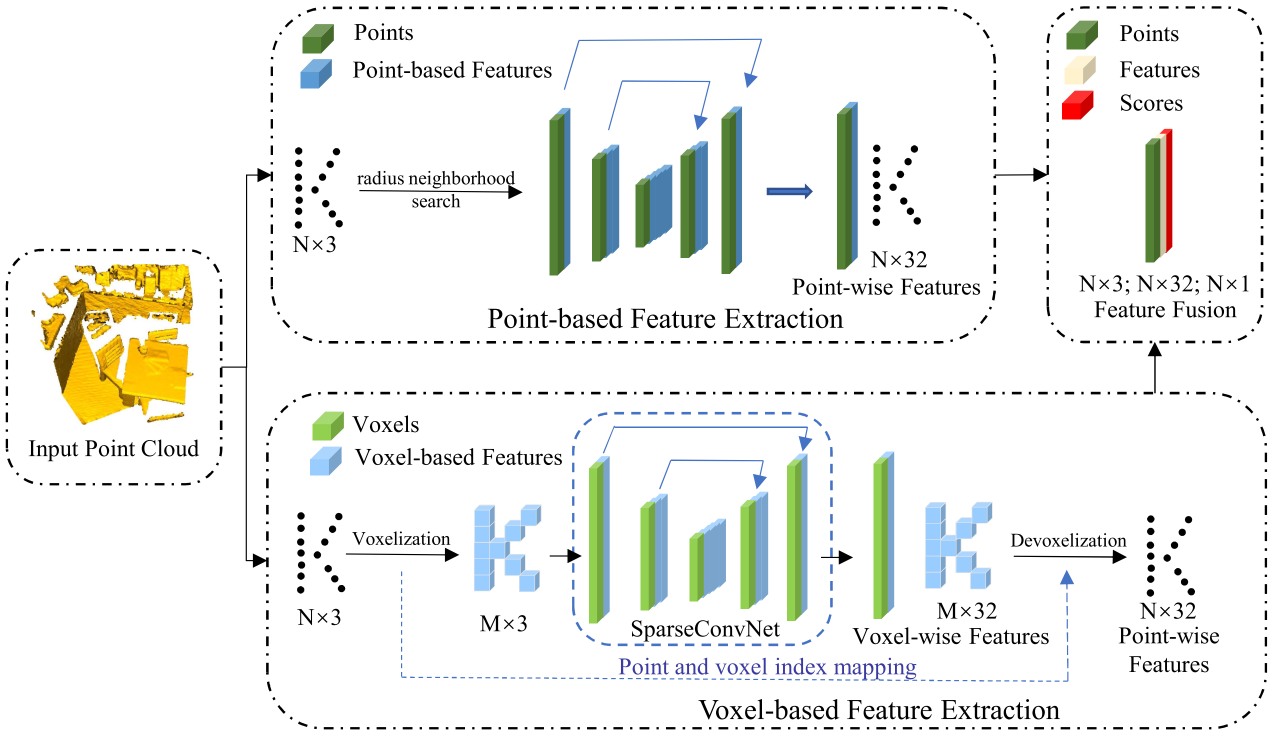


Fig. 2. V2PNet framework for feature propagation and fusion. The raw point cloud imported into the network is divided into two branches. First, the point-based feature extraction branch performs a radius neighborhood search and dense convolution to extract the point-wise features. Next, the voxel-based branch voxelizes and extracts the voxel-wise features by sparse convolution and then performs devoxelization mapping to obtain the point-wise voxel features. Finally, the fused features are extracted from the two branches.

to implement a 3-D sparse CNN. MinkowskiNet supports many standard layers analogous to those used in 2-D image processing, including convolution, transpose convolution, pooling, and unpooling layers. Most of the modular layers are built using atomic operators and a broadcasting mechanism for sparse tensors. The 3-D sparse CNN differs from a dense convolution network in the following two respects: 1) an empty voxel does not contribute to a convolution in a local neighborhood and 2) even if the sparse tensor has occupied neighbors, it only has a feature at a certain position if the corresponding coordinates exist in the coordinate set $v \in \mathbb{Z}^{M\times 3}$, i.e., empty voxels are regarded as featureless during sparse convolution in a local neighborhood. Therefore, 3-D sparse convolution does not change voxel sparsity.

The sparse tensor is the input of MinkowskiNet. We use the voxelizer $V(\lfloor p/\delta \rfloor, x)$ from the quantized coordinates $\lfloor p/\delta \rfloor \in \mathbb{Z}^{N\times 3}$ and input features x into the voxel coordinates $v \in \mathbb{Z}^{M\times 3}$ and voxel features $x_v \in \mathbb{R}^{M\times C}$. The sparse tensor is built using the coordinate tensor v and feature tensor x_v (v and x_v are classical tensors, i.e., 2-D matrices), with additional coordinate management by MinkowskiNet for efficient neighborhood indexing and mapping in the sparse convolution.

A UNet-like architecture [40] is used for the encoder–decoder network to learn the voxel features in the latent space. Specifically, residual skip connections are used, as employed in ResUNet [41]. As shown in Fig. 2, ResUNet involves three layers. Thus, the voxel sizes and channels of features are doubled between two consecutive layers. Finally, a 32-D voxel-level feature tensor is obtained, which is subsequently devoxelized and mapped to a 32-D point-level feature tensor, as described in the next subsection.

Algorithm 1: Voxelizer.

```
**Inputs**: quantized octree coordinates $c = \lfloor p/\delta \rfloor \in \mathbb{Z}^{N\times 3}$
and point-based attributes $x \in \mathbb{R}^{N\times C}$
1: $h \in \mathbb{Z}^N \leftarrow hash(c)$
2: $h' \in \mathbb{Z}^N, i' \in \mathbb{Z}^N \leftarrow sort(h)$
3: $i'' \in \mathbb{Z}^M, m \leftarrow unique(h', i')$ ;
     $v \leftarrow c(i'')$ ; $m \in [0,\ M-1]$
4: $x_v \in \mathbb{R}^{M\times C} \leftarrow ScatteredReduce(x, m)$
**return** $v, x_v, m$
```

C. Voxel-to-Point Propagation and Feature Fusion

The performance of the existing point-based methods that use K-dimensional (KD)-tree to search for neighborhoods in continuous space is affected by uneven point-cloud distributions. Thus, our method converts discrete points to voxel representations to explicitly access the neighborhood and thereby achieve efficient convolution. However, we need to obtain the mapping between N points and M voxels, as point- and voxel-based attributes are stored as 2-D tensors with leading sizes of N and M, respectively. The voxelizer $V(\lfloor p/\delta \rfloor, x)$ produces the mapping $m \in \mathbb{Z}^N$ from the quantized coordinates $\lfloor p/\delta \rfloor \in \mathbb{Z}^{N\times 3}$ to the voxelized indices $[0, M-1]$. The generation procedure of the voxelizer is presented in Algorithm 4.

We use MinkowskiNet to implement sparse convolution. The inputs to generate the sparse tensor require both the quantized coordinates of the voxels v and the corresponding features x_v. A hash function is used to transform each quantized 3-D coordinate

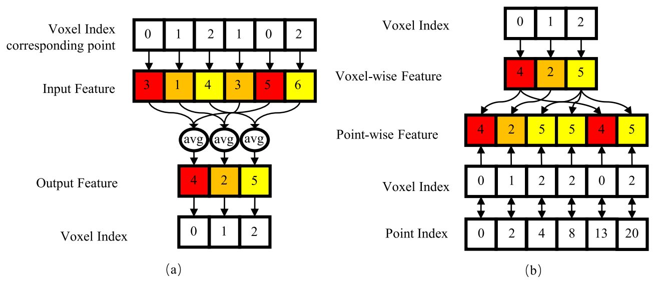


Fig. 3. Relationships of indices between points and voxels. (a) Scattered reduction to map features from points to voxels. (b) Index gathering to map features from voxels to points.

into a unique hash key for efficient neighbor indexing in graphics processing unit (GPU)-based computing devices. We use the same hash function in MinkowskiNet to generate the hash keys h for the input quantized coordinates $c = \lfloor p/\delta \rfloor$. Next, the hash keys are sorted into h' for further processing, and the sorted indices i' are returned. Because several points may exist in the same voxel, the total number of occupied voxels M must be smaller than the number of points N. A unique operation is applied to select a single point inside each voxel as i'', and the reverse indices are returned to maintain the voxel-to-point mapping m. The corresponding quantized coordinates of the voxels v are directly retrieved. Finally, a scattered reduction [42] is used to average the features inside a given voxel, as shown in Fig. 3(a).

After convolution through each of the two branches, the point-based and voxel-based features are extracted. The hash table corresponding to the indexes and features is derived through simple devoxelization, as shown in Fig. 3(b). The features of adjacent voxels can be accumulated into point features using linear interpolation [42]. However, given the small size of the voxels, we apply direct nearest sampling.

Finally, we fuse the two features extracted from the two branches. Feature fusion necessitates the quantitative unification of the two features, as this deals with the inconsistency of their magnitudes. First, we concatenate the two 32-D features and reduce the dimensionality by using the MLP to obtain 32-D fused features, and then perform an L2-normalized operation on the obtained feature descriptors. This process is illustrated in Fig. 2.

D. Losses for Keypoint Detection and Description

1) Saliency Scores for Keypoint Detection: Most of the existing learning-based registration methods randomly select keypoints, which means it is difficult to ensure repeatability and reliability. Thus, we use a strategy based on D3Feat [24], i.e., we jointly perform keypoint detection and descriptor extraction. We calculate the local saliency score $s \in \mathbb{R}^N$ for each point based on the fused features $f \in \mathbb{R}^{N\times C}$ obtained in the network. With a slight abuse of notation, we use f_i^j to denote the feature value of the ith point and jth channel.

A representative keypoint detector for images is based on the measurement of self-similarity [43], with the similarity calculated using low-level features, such as intensity values. According to the D3Feat strategy, a keypoint: 1) must use the most prominent channel and 2) must select the most salient point in a local region of the ith point, N_i, as mathematically represented by

$$k = \underset{t}{argmax} f_i^t \; and \, i = \underset{t}{argmax} f_t^k. \tag{3}$$

Because the argmax operator is not differentiable, the differentiable *softmax* approximation is used, as follows:

$$s_i = \max_k \log\left(1 + \exp\left(f_i^k - \frac{1}{|N_i|}\sum_{t\in N_i} f_t^k\right)\right)\frac{f_i^k}{\max\limits_t f_i^t}. \tag{4}$$

The first part (a logarithmic function) uses the the density invariant version of *softmax* to compare the saliency value of point i against its neighborhood N_i. The second part (a ratio) approximately evaluates the most prominent channel

$$th_1 = mean(S) + \frac{1}{2}(\max(S) - mean(S))$$
$$th_2 = mean(S) \tag{5}$$

where $S(s_i)$ denotes the set of saliency scores.

Then, inspired by the local distinctness and global rarity [44] of each point, the values higher than threshold th_1 or lower than threshold th_2 considered to have global rarity and squared as the final scores for better detection of keypoints. The specified number of keypoints is filtered by sorting them in descending order. A uniform distribution of keypoints can be obtained by applying a nonmaximum suppression postprocessing step.

2) Loss Functions: Defining the loss function is vital for 3-D metric learning. In most approaches for joint keypoint detection and descriptor extraction [45], loss functions consist of the following two parts: 1) descriptor loss and 2) detector loss. However, to obtain better optimization and faster convergence, we use circle loss [46] for descriptor learning and self-supervised loss [24] for keypoint saliency detection learning. For completeness, we briefly introduce these methods below.

First, we define the intraclass and interclass similarities for a feature correspondence pair (f_i, f_j'), where i and j are the indices for the source and target point clouds $(\mathcal{P}, \mathcal{P}')$, respectively. The learning process aims to decrease the distances between positive matches $(i,j) \in E$, i.e., the intraclass similarity d^+, and increase the distances between negative matches $(i,j) \notin E$, i.e., the interclass similarity d^-. The intraclass and interclass similarities are defined as follows:

$$d_i^+ = \left\|f_i - f_j'\right\|_2, \forall(i,j) \in E$$
$$d_i^- = \min_j(\left\|f_i - f_j'\right\|_2), s.t. \left\|p_i - p_j\right\|_2 > r, \forall(i,j) \notin E \tag{6}$$

where r is a safe margin applied to prune possible false-alarm matches.

Next, the detector and descriptor losses are established based on the similarity measurements, as follows:

$$\mathcal{L}_{det} = \frac{1}{|E|}\sum_{i,j}\left(d_i^+ - d_i^-\right)\left(s_i + s_j'\right)$$
$$\mathcal{L}_{des} = \log\left[1 + \sum_{i,j}\exp\left(\gamma\left(\omega_j^- d_j^- - \omega_i^+ d_i^+\right)\right)\right] \tag{7}$$

where γ is a scale factor, and ω denotes the nonnegative weight parameters that are used to control the gradient size of the interclass and intraclass similarities. The hyperparameters γ and ω are set to be similar to those in as previous work [46]. The final loss for the optimizer is the sum of the two loss functions, i.e.,

$$\mathcal{L} = \mathcal{L}_{\text{det}} + \mathcal{L}_{\text{des}}. \tag{8}$$

IV. Experimental Analyses

The feasibility and accuracy of our method are evaluated over the following two datasets: 1) The first dataset is the 3DMatch dataset [27], which was derived via indoor RGB-D image reconstruction and consists of 62 real-world indoor scenes (e.g., of kitchens, study rooms, and hotel rooms). Each scene contains a number of overlapped fragments, and the data provider supplies ground truth rotation and translation parameters for evaluating performance. 2) The second dataset is the KITTI dataset [47], [48], which consists of 20 outdoor real-world mobile measurement point-cloud scenes and is currently the largest international dataset for computer vision algorithm evaluation in autonomous driving scenarios. Thus, the two datasets consist of different scenarios and capture point clouds with different modalities, e.g., RGB-D and LiDAR.

Although recent transformer-based methods, such as GeoTransformer [49], have demonstrated promising results, they often have high GPU memory requirements ranging from 80-110 G, which can be challenging for individual researchers or smaller research institutions to employ. Conversely, our method requires only approximately 11 G of GPU memory, making it executable on consumer hardware and providing a more practical alternative for researchers with limited resources.

A. Evaluation Metrics

When using the 3DMatch dataset, we employ the following two standard metrics to evaluate the feature descriptors extracted by our model in the registration scenario: 1) feature match recall (FMR) [18] and 2) registration recall (RR) [50]. FMR indicates the percentage of fragment pairs with highly credible recovery poses, and RR indicates the proportion of overlapping segments that are correctly recovered during reconstruction. A robust local registration algorithm such as random sample consensus (RANSAC) is used to estimate the rigid transformation between point clouds. Subsequently, the root-mean-square error (RMSE) of the ground truth correspondence under the estimated transformation is calculated. To ensure a fair comparison, we perform 50 000 iterations (as used in the other frameworks) to evaluate the transformation results.

The key application for point matching is the registration of different point clouds. We use the following metrics that are commonly applied to evaluate point-cloud registration [22]: 1) relative translation error (RTE) and 2) relative rotation error (RRE). That is, RTE and RRE are the registration errors of the features used for RANSAC. In the registration evaluation over the KITTI dataset, results with RTEs less than 2 m and RREs less than 5° are considered accurate.

TABLE I
Evaluation Results Obtained Using the 3DMatch Dataset With Different Keypoint Settings

Year	#Keypoints	5000	2500	1000	500	250
	Feature Match Recall (%)					
2019	3DSmoothNet [19]	94.7	94.2	92.6	90.1	82.9
2021	SpinNet [32]	97.6	97.2	96.8	95.5	94.3
2021	Predator [52]	96.6	96.6	96.5	96.3	96.5
2022	CAFeat3D [53]	96.1	96.0	95.7	94.7	93.8
2022	GeoTransfomer [49]	**98.1**	**98.1**	**98.1**	**98.1**	**97.8**
-	V2PNet *(ours)*	96.9	96.8	96.7	95.9	95.0
	Registration Recall (%)					
2019	3DSmoothNet [19]	80.3	77.5	73.4	64.8	50.9
2021	SpinNet [32]	88.6	86.6	85.5	83.5	70.2
2021	Predator [52]	89.0	89.9	90.6	88.5	86.6
2022	GeoTransfomer [49]	**92.2**	**92.0**	**91.6**	**91.5**	**91.1**
2022	CAFeat3D [53]	-	-	85.7	83.8	82.5
-	V2PNet*(ours)*	89.4	88.0	88.9	85.5	80.0

The bold values indicates the best performance.

B. Evaluation on the Indoor-Scene 3DMatch Dataset

Dataset: We use the same strategy as is used in the official protocol [27] to divide 3DMatch into training and test datasets. The test dataset contains eight scenes with partially overlapped point-cloud fragments and corresponding transformation parameters available for evaluating the metrics. The training dataset consists of the point cloud pairs with overlaps greater than 30%.

Experimental implementation: The input point-cloud file for training is obtained by downsampling with a 3-cm voxel using Open3D [51]. Thus, by using nearest neighbor searching, pairs of points with Euclidean distances of less than 3 cm are identified, and these are used as the set of correspondence points. We perform data augmentation for each point cloud fragment by adding Gaussian noise with a standard deviation of 0.005, performing random scaling in the range of $[0.8,\ 1.2]$, and introducing a random rotation angle of $[0,\ 2\pi]$ around an arbitrary axis. We randomly select 64 batches at a time to calculate the loss. Our network is implemented in PyTorch and converges at approximately 70 epochs.

Quantitative Evaluations: Testing for each scene fragment is performed using 5000, 2500, 1000, 500, and 250 keypoints. Because fewer than 2500 keypoints have nonnegative saliency scores, we randomly select 5000 and 2500 keypoints. Additionally, we select keypoints according to the descending ranks of the saliency scores. Table I summarizes the FMRs and RRs for our method and for state-of-the-art methods.

The indoor-scene dataset suffers from limited diversity and realism due to the small number of scenes and object types. Although state-of-the-art methods have achieved high performance, our approach remains competitive in certain aspects. The rotation invariance of our method is also verified, as shown in Table II.

We conducted ablation experiments on an indoor-scene dataset to compare the performance of our point-voxel feature fusion strategy with single strategies, where the voxel-only strategy used FCGF [28] and the point-only strategy used D3Feat [24]. Table III presents the results, which demonstrate the superior effectiveness of our V2PNet over existing methods, particularly with low numbers of keypoints.

TABLE II
EVALUATION OF THE ROTATION INVARIANCES (FMR = FEATURE MATCH RECALL, AND STD = THE STANDARD DEVIATION OF FMR)

	Origin		Rotated		
	FMR (%)	STD	FMR (%)	STD	Feat. dim.
3DMatch [27]	59.6	8.8	1.1	1.2	512
3DSmoothNet [19]	94.7	2.7	94.9	2.5	32
FCGF [28]	95.2	2.9	95.3	3.3	32
D3Feat [24]	95.8	2.9	95.5	3.5	32
SpinNet [32]	**97.6**	1.9	**97.5**	**1.9**	32
V2PNet(*ours*)	96.9	**1.7**	96.0	2.0	32

The bold values indicates the best performance.

TABLE III
ABLATION STUDY ON FUSED FEATURES USING THE 3DMATCH DATASET WITH DIFFERENT KEYPOINT SETTINGS

Methods	**5000**	**2500**	**1000**	**500**	**250**
Feature Match Recall (%)					
Voxel-only	95.2	95.5	94.6	93.0	89.9
Point-only	95.8	95.6	94.6	94.3	93.3
Point-voxel	**96.9**	**96.8**	**96.7**	**95.9**	**95.0**
Registration Recall (%)					
Voxel-only	87.3	85.8	85.8	81.0	73.0
Point-only	82.2	84.4	84.9	82.5	79.3
Point-voxel	**89.4**	**88.0**	**88.9**	**85.5**	**80.0**

The bold values indicates the best performance.

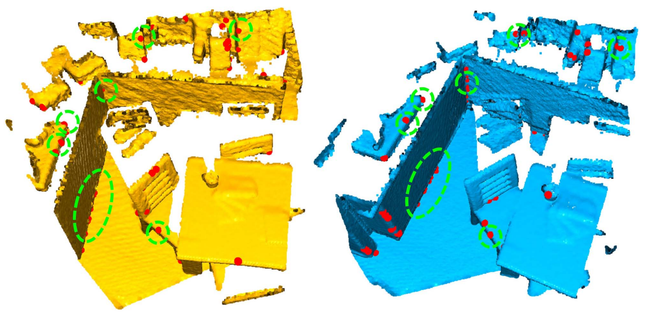


Fig. 4. Visualization of the top-50 detected keypoints (in terms of saliency score ranking) for the 3DMatch dataset.

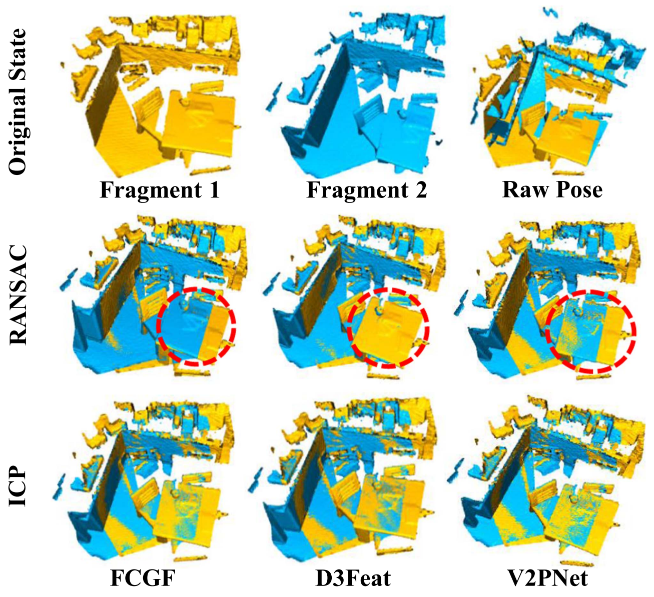


Fig. 5. Registration results on 3DMatch.

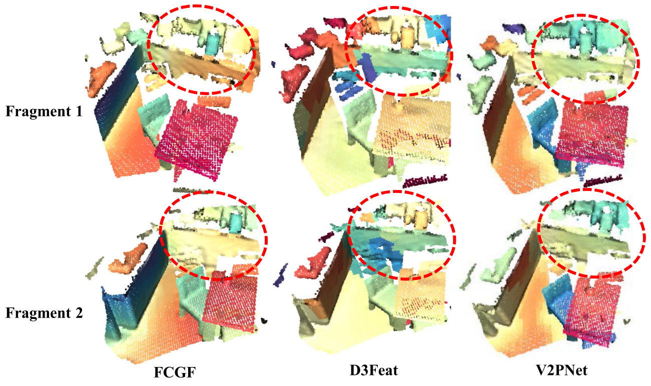


Fig. 6. Color-coded features using t-SNE overlaid on selected fragment pairs on the 3DMatch dataset.

Qualitative Evaluations: We select a random pair of scans to qualitatively evaluate the detector and descriptor performances. Only the top 50 keypoints from the source and target scans by the detector are plotted in Fig. 4. Even when few keypoints are used, many of them coincide in the two datasets. We visualize the original raw scans for the descriptors. Additionally, the results for the cases with RANSAC-based coarse registration and iterative closest point-based fine registration are shown in Fig. 5. With only coarse registration, FCGF and D3Feat demonstrate clear layering effects, which indicates there is a systematic error in their transformations. In contrast, V2PNet demonstrates less prominent layering effects.

Visual Inspection: We train the comparative models and retrieve the feature descriptors for their visualizations. For the sake of comparison, we adopt the same 32-D point-cloud feature strategy as the point-based and voxel-based approach for convenience. The visualization effect is shown in Fig. 6, which is derived using the t-distributed stochastic neighbor embedding (t-SNE) visualization method [54]. The feature descriptors are projected to low-dimensional space and encoded as colored spectra. Compared with the descriptors of the existing frameworks, the descriptor of V2PNet exhibits superior semantic awareness for the same object, in addition to higher distinctiveness for different objects in a scene.

C. Evaluation on the Outdoor-Scene KITTI Dataset

Dataset and experimental implementation: Compared with the indoor scenes in 3DMatch, outdoor scenes have a larger number of points and thus use different thresholds for certain preprocessing steps. Therefore, the KITTI dataset is downsampled in Open3D using a 30-cm voxel [51]. To enhance the robustness of the result, a random Gaussian noise of 0.01 is introduced during training. Because outdoor LiDAR scenarios feature absolute object metrics, scale augmentation is not performed. Finally, 1024 nodes are extracted in each batch for backward propagation of the network. The other steps are identical to those used for training with the indoor dataset.

TABLE IV
SUCCESS RATES (%) OBTAINED USING THE KITTI DATASET WITH VARIOUS NUMBERS OF KEYPOINTS

Methods	All	5000	2500	1000	500	250
SpinNet [32]	99.10	98.74	96.40	90.99	71.53	45.05
Predator [52]	99.64	99.64	99.64	99.64	99.46	99.48
GeoTransfomer [49]	99.82	**99.82**	99.82	99.82	**99.82**	**99.82**
V2PNet(*ours*)	**99.86**	99.76	**99.84**	**99.87**	99.74	99.29

The bold values indicates the best performance.

TABLE V
ABLATION STUDY ON THE REGISTRATION SUCCESS (%) RATE OF FUSED FEATURES USING THE KITTI DATASET WITH VARIOUS NUMBERS OF KEYPOINTS

Methods	All	5000	2500	1000	500	250
Voxel-only	96.57	96.39	96.03	93.69	90.09	68.62
Point-only	99.81	**99.81**	99.81	99.81	**99.81**	**99.63**
Point-voxel	**99.86**	99.76	**99.84**	**99.87**	99.74	99.29

The bold values indicates the best performance.

TABLE VI
QUANTITATIVE COMPARISONS OF REGISTRATION ERRORS OBTAINED USING THE KITTI DATASET

	RTE (*cm*)		RRE (○)	
	AVG	STD	AVG	STD
3DFeat-Net [22]	25.9	26.2	0.57	0.46
FCGF [28]	9.52	1.30	0.30	0.28
SpinNet [32]	9.88	0.50	0.47	0.09
Predator [52]	7.74	0.27	0.28	0.27
D3Feat [24]	6.90	0.30	0.24	0.06
GeoTransfomer [49]	7.35	**0.06**	0.27	0.25
V2PNet(*ours*)	**4.48**	0.16	**0.15**	**0.03**

RTE and RRE are the relative translation and rotation errors, respectively, with lower values preferred.
The bold values indicate the best performance.

Quantitative Evaluations: Because each scan of LiDAR data captures only a small fraction of large outdoor scenes, different timestamps must be registered to obtain a comprehensive representation of scenes. To this end, the evaluation metrics for rigid transformation are used, e.g., the RTEs and RREs obtained by comparing different scans. To compute the transformation parameters, we use RANSAC with 50 000 iterations. Table IV lists the success rates of the registration with various keypoint settings.

We performed further ablation experiments on an outdoor-scene dataset to compare our fusion strategy with single strategies, namely the voxel-only strategy using FCGF [28] and the point-only strategy using D3Feat [24]. Table V demonstrates the effectiveness of our point-voxel feature strategy, with V2PNet exhibiting significant superiority over the existing methods. The results of our method are comparable with those of the existing methods. Nevertheless, in terms of the quantitative registration errors, V2PNet outperforms its counterparts by a large margin, as shown in Table VI.

As for the outdoor KITTI dataset, it is a sparse point cloud dataset acquired by Velodyne-64 3D LiDAR scanners. Notably, the point clouds are gravity-aligned, and our method does not involve alignment with a reference axis, similar to 3DFeat-Net [22]. While our V2PNet achieves slightly lower performance than state-of-the-art methods in the indoor 3DMatch scene, well-aligned point cloud datasets tend to promote descriptor learning [32], and we defer further exploration of these aspects to future research.

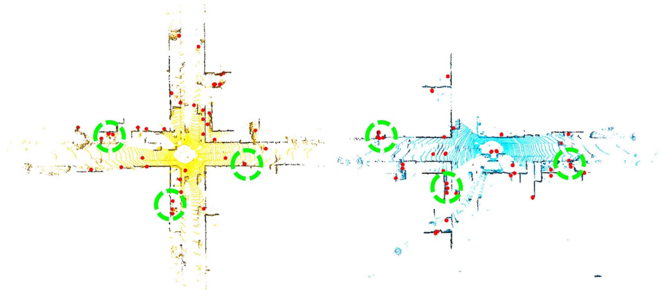


Fig. 7. Visualization of the top-50 detected keypoints (in terms of saliency score ranking) obtained using the KITTI dataset.

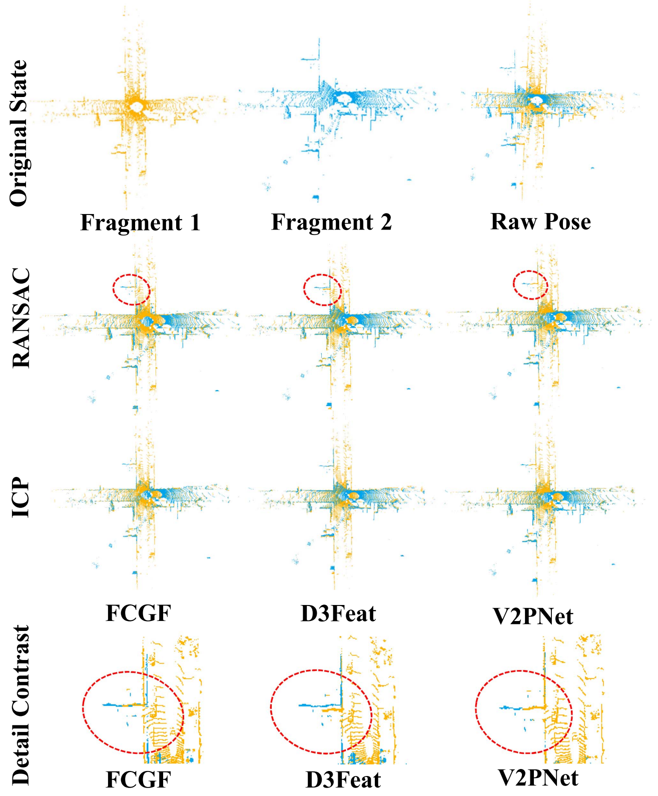


Fig. 8. Registration results obtained using KITTI.

Qualitative Evaluations: We arbitrarily select a set of scans to evaluate the detector and descriptor performances. Fig. 7 shows the top-50 keypoints in the source and target scans for the detector. Although only a few keypoints are used, they accumulate in certain regions, which contributes to the later alignment. The coarse and fine alignment results are shown in Fig. 8. In the coarse alignment scenario, FCGF and D3Feat exhibit obvious misalignment in several regions, which makes it difficult to suppress systematic errors. In contrast, when V2PNet is used, the errors are effectively suppressed in the coarse matching scenario, and more accurate matching results are obtained. Consequently,

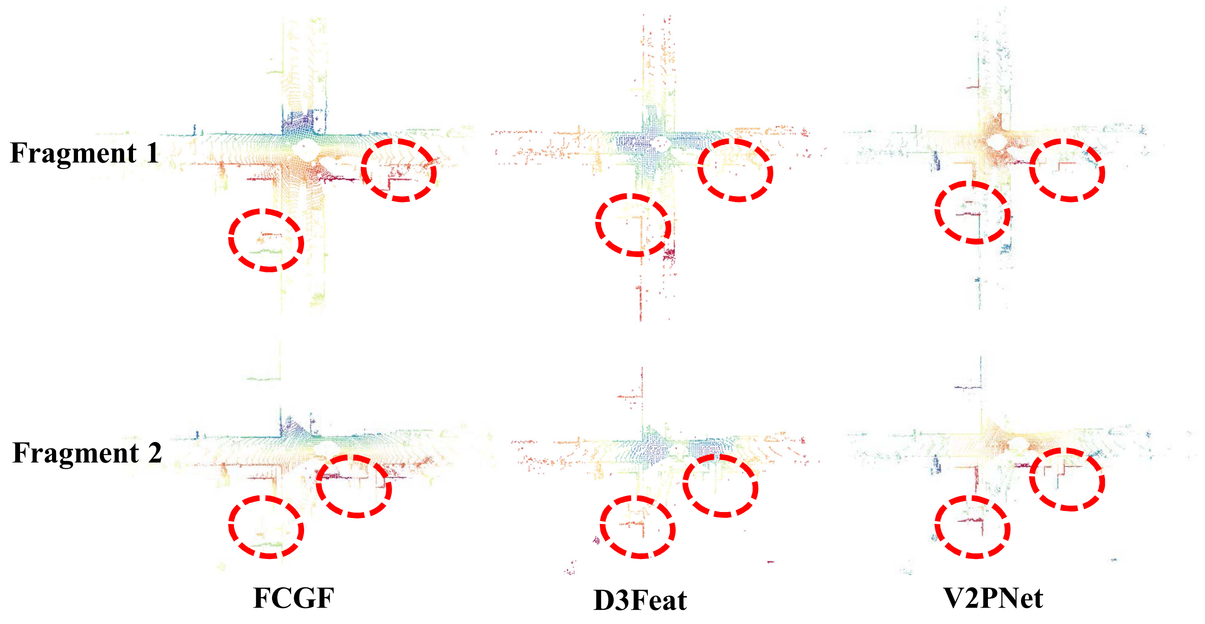


Fig. 9. Color-coded features using t-SNE overlaid on selected fragment pairs in the KITTI dataset.

the overall process of V2PNet consumes less time than the overall processes of FCGF and D3Feat.

Visual Inspect: The features of the KITTI dataset are visualized as shown in Fig. 9. Specifically, the 32-D point cloud features learned by FCGF, D3Feat, and V2PNet are generated using the t-SNE visualization method. In several locations and compared with the descriptors of FCGF and D3Feat, the descriptors of V2PNet are more discriminative, with distinct geometric structures.

V. Conclusion

We design V2PNet, a novel descriptor extraction network that fuses point-based features and voxel-based features for point cloud registration. The dual-branch architecture of V2PNet incorporates the accuracy achievable with the point-based strategy and the continuous representation achievable with the voxel-based strategy. Extensive experiments on an indoor-scene dataset and an outdoor-scene dataset (the 3DMatch and KITTI datasets, respectively) demonstrate the effectiveness of our method. In particular, our method outperforms a single-feature network and a descriptor extraction method in terms of most quantitative evaluation metrics. Limitations of our method are that fusion is performed only in the last step, and no communication occurs between the two branches in the encoder–decoder steps. Future work could explore more interaction paths by interleaving the propagation mechanism into each layer of the two networks.

References

[1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[2] S. Wang et al., "Robust 3D reconstruction of building surfaces from point clouds based on structural and closed constraints," *ISPRS J. Photogramm. Remote Sens.*, vol. 170, pp. 29–44, Dec. 2020.

[3] Y. Zhao, H. Zhao, M. Radanovic, and K. Khoshelham, "A unified framework for automated registration of point clouds, mesh surfaces and 3D models by using planar surfaces," *Photogrammetric Rec.*, vol. 37, no. 180, pp. 366–384, 2022.

[4] R. Hänsch, T. Weber, and O. Hellwich, "Comparison of 3D interest point detectors and descriptors for point cloud fusion," *ISPRS J. Photogramm. Remote Sens.*, vol. 2, no. 3, pp. 57–64, 2014.

[5] X. Ge and B. Wu, "Contextual global registration of point clouds in urban scenes," *Photogramm. Eng. Remote Sens.*, vol. 85, no. 8, pp. 559–571, 2019.

[6] W. Zhou, X. Cao, X. Zhang, X. Hao, D. Wang, and Y. He, "Multi voxel-point neurons convolution (MVPConv) for fast and accurate 3D deep learning," 2021, *arXiv:2104.14834*.

[7] L. Li, L. Han, M. Ding, H. Cao, and H. Hu, "A deep learning semantic template matching framework for remote sensing image registration," *ISPRS J. Photogramm. Remote Sens.*, vol. 181, pp. 205–217, Nov. 2021.

[8] Z. Zhang, G. Chen, X. Wang, and M. Shu, "DDRNet: Fast point cloud registration network for large-scale scenes," *ISPRS J. Photogramm. Remote Sens.*, vol. 175, pp. 184–198, May 2021.

[9] A. Wichmann, A. Agoub, V. Schmidt, and M. Kada, "RoofN3D: A database for 3D building reconstruction with deep learning," *Photogramm. Eng. Remote Sens.*, vol. 85, no. 6, pp. 435–443, Jun. 2019.

[10] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel CNN for efficient 3D deep learning," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 965–975, 2019.

[11] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.

[12] D. Prokhorov, "A convolutional learning system for object classification in 3-D lidar data," *IEEE Trans. Neural Netw.*, vol. 21, no. 5, pp. 858–863, May 2010.

[13] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6410–6419.

[14] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9613–9622.

[15] B. Graham, M. Engelcke, and L. V. D. Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9224–9232.

[16] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convNets: Minkowski convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3070–3079.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 5105–5114, 2017.

[18] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3D point matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 195–205.

[19] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3D point cloud matching with smoothed densities," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5540–5549.

[20] Y. Aoki, H. Goforth, R.A. Srivatsan, and S. Lucey, "PointNetLK: Robust & efficient point cloud registration using PointNet," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7156–7165.

[21] V. Sarode et al., "Pcrnet: Point cloud registration network using PointNet encoding," 2019, *arXiv:1908.07906*.

[22] Z. J. Yew and G. H. Lee, "3dfeat-Net: Weakly supervised local 3D features for point cloud registration," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 607–623.

[23] S. Liu et al., "Rethinking of learning-based 3D keypoints detection for large-scale point clouds registration," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 112, 2022, Art. no. 102944.

[24] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6358–6366.

[25] J. Huang and S. You, "Point cloud labeling using 3D convolutional neural network," in *Proc. Int. Conf. Pattern Recognit.*, 2016, pp. 2670–2675.

[26] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.

[27] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dMatch: Learning local geometric descriptors from RGB-D reconstructions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 199–208.

[28] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8957–8965.

[29] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2511–2520.

[30] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal, "Learning multiview 3D point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1756–1766.

[31] H. Wang, Y. Liu, Z. Dong, and W. Wang, "You only hypothesize once: Point cloud registration with rotation-equivariant descriptors," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 1630–1641.

[32] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "SpinNet: Learning a general surface descriptor for 3D point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11748–11757.

[33] S. Shi et al., "PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection," *Int. J. Comput. Vis.*, vol. 131, pp. 1–21, 2022.

[34] J. Noh, S. Lee, and B. Ham, "HVPR: Hybrid voxel-point representation for single-stage 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14600–14609.

[35] J. Huang, Y. Zhang, and M. Sun, "PrimitiveNet: Primitive instance segmentation with local primitive embedding under adversarial metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15343–15353.

[36] C. Zhang, H. Wan, X. Shen, and Z. Wu, "PVT: Point-voxel transformer for point cloud learning," *Int. J. Intell. Syst.*, vol. 37, no. 12, pp. 11985–12008, 2022.

[37] Z. Wang et al., "3D MSSD: A multilayer spatial structure 3D object detection network for mobile lidar point clouds," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 102, 2021, Art. no. 102406.

[38] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention* (ser. Lecture Notes in Computer Science), N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham:Springer, 2015, pp. 234–241.

[39] G. Qian et al., "Pointnext: Revisiting PointNet with improved training and scaling strategies," 2022, *arXiv:2206.04670*.

[40] A. Akhtar, W. Gao, X. Zhang, L. Li, Z. Li, and S. Liu, "Point cloud geometry prediction across spatial scale using deep learning," in *Proc. IEEE Int. Conf. Vis. Commun. Image Process.*, 2020, pp. 70–73.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[42] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," 2019, *arXiv:1903.02428*.

[43] J. Revaud et al., "R2d2: Repeatable and reliable detector and descriptor," 2019, *arXiv:1906.06195*.

[44] X. Ding, W. Lin, Z. Chen, and X. Zhang, "Point cloud saliency detection by local and global feature fusion," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5379–5393, Nov. 2019.

[45] Y. Wang, B. Yang, Y. Chen, F. Liang, and Z. Dong, "JoKDNet: A joint keypoint detection and description network for large-scale outdoor TLS point clouds registration," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 104, Dec. 2021, Art. no. 102534.

[46] Y. Sun et al., "Circle loss: A unified perspective of pair similarity optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6397–6406.

[47] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[48] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Rob. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.

[49] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11133–11142.

[50] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5556–5565.

[51] Q. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3D data processing," 2018, *arXiv:1801.09847*.

[52] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "PREDATOR: Registration of 3D point clouds with low overlap," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4265–4274.

[53] S. Guo, Y. Fu, Z. Qian, Z. Rong, and Y. Wu, "Cross-attention-based feature extraction network for 3D point cloud registration," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2022, pp. 1–6.

[54] J. Wang et al., "Learning fine-grained image similarity with deep ranking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1386–1393.

Han Hu (Member, IEEE) received the B.S. degree in photogrammetry and remote sensing from the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China, and the Ph.D. degree in photogrammetry and remote sensing from the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, in 2010 and 2015, respectively.

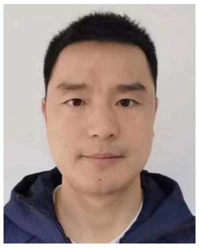
From 2015 to 2019, he was a Postdoctoral Fellow with the Hong Kong Polytechnic University. He is currently a Professor with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, China. His research interests include photogrammetry, point clouds processing, and 3-D modeling.

Yongkuo Hou received the B.E. degree in geomatics engineering from the College of Geodesy and Geomatics, Shandong University of Science and Technology, Qingdao, China, in 2019. He is currently working toward the M.E. degree in geomatics engineering with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, China.

His research interests include photogrammetry, 3-D reconstruction, and point cloud processing.

Yulin Ding received the B.S. degree in geographic information system from the School of Resource and Environmental Sciences, Wuhan University, Wuhan, China, and the Ph.D. degree in photogrammetry and remote sensing in the State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing, Wuhan University, Wuhan, China, in 2009 and 2014, respectively.

From 2014 to 2016, she was a Postdoctoral Fellow with The Chinese University of Hong Kong. She is currently an Associate Professor with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, China. Her research interests include virtual geographic environments, 3-D GIS, and big data.



Guoqiang Pan received the B.S. degree in geographic information system from the School of Resource and Environmental Sciences, Wuhan University, Wuhan, China.

He is with the Equipment Project Management Center, Chinese People’s Armed Police Force, Beijing, China. His research interests include 3-D GIS, photogrammetry, and 3-D modeling.

Min Chen received the B.S. and Ph.D. degrees in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2009 and 2014, respectively.

From 2016 to 2018, he was a Postdoctoral Fellow with Purdue University. He is currently an Assistant Professor with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, China. His research interests include feature detection, image matching, and 3-D reconstruction.

Xuming Ge received the Ph.D. degree in geodesy from Technical University of Munich, Munich, Germany, in 2016.

From 2017 to 2019, he was a Postdoctoral Fellow with the Hong Kong Polytechnic University, Hong Kong. He is currently an Assistant Professor with the Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu, China. His research interests include photogrammetry, 3-D reconstruction, point cloud processing, and sensor calibration.