

WSDesc: Weakly Supervised 3D Local Descriptor Learning for Point Cloud Registration

Lei Li, Hongbo Fu, and Maks Ovsjanikov

Abstract—In this work, we present a novel method called WSDesc to learn 3D local descriptors in a weakly supervised manner for robust point cloud registration. Our work builds upon recent 3D CNN-based descriptor extractors, which leverage a voxel-based representation to parameterize local geometry of 3D points. Instead of using a predefined fixed-size local support in voxelization, we propose to *learn* the optimal support in a data-driven manner. To this end, we design a novel differentiable voxelization layer that can back-propagate the gradient to the support size optimization. To train the extracted descriptors, we propose a novel registration loss based on the deviation from rigidity of 3D transformations, and the loss is weakly supervised by the prior knowledge that the input point clouds have partial overlap, without requiring ground-truth alignment information. Through extensive experiments, we show that our learned descriptors yield superior performance on existing geometric registration benchmarks.

Index Terms—Point cloud, 3D local descriptor, geometric registration, differentiable voxelization, 3D CNN, weak supervision.

1 INTRODUCTION

ENCODING 3D local geometry into descriptors has been an essential ingredient in many computer graphics and vision problems, such as recognition [1], retrieval [2], segmentation [3], registration [4], [5], etc. In this work, we are interested in developing 3D local descriptors for robust point cloud registration (Fig. 1). Matching 3D geometry of scans of real-world scenes is a challenging task due to the presence of noise and partiality in the input data. To address such issues, learning-based descriptors have received significant attention in recent years, demonstrating superior performance over hand-crafted ones [6].

To capture local geometric structures, an important step for learning-based descriptors [7], [8], [9], [10] is to extract local neighborhood support of 3D points with some predefined size. The local neighborhoods have a variety of representations, such as the point-based [8], voxel grids [9], histograms [11], or point pair features [12], which are amenable to feature learning with networks. In all of these approaches, the support size is crucial in determining the amount of local geometry information captured in the learned descriptors. Normally, this parameter is set empirically and is not involved in network training, which may keep the network from extracting more informative descriptors. Moreover, simply using a large support may lead to descriptors that are too global and thus specific to a shape or a scene, whereas using a small support may lead to loss of robustness and informativeness, as discussed in [6], [13]. Thus, finding an optimal support is not an effortless task.

To train local descriptors, prior works have leveraged contrastive learning, such as with the triplet [9] or the N-tuple loss [12], to optimize the descriptor

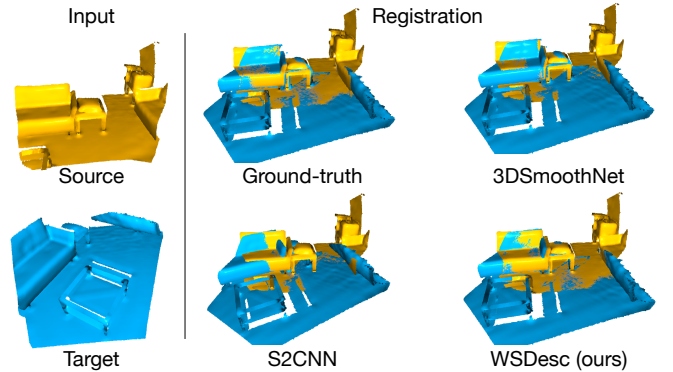


Fig. 1: Registration of two point cloud fragments (with a partial overlap ratio of 51%) using learned descriptors. 3DSmoothNet [9] is trained *with* ground-truth alignment information, while S2CNN [14] and our WSDesc do not require such information for training.

similarity. Researchers have also investigated end-to-end registration-based training by applying the extracted descriptors to in-network alignment estimation for pairwise point clouds [15], [16]. In the aforementioned works [9], [12], [15], [16], the training typically requires ground-truth alignment information for supervision. While the ground-truth can potentially be derived from existing 3D reconstruction pipelines [17], it can be prone to errors, and typically requires careful manual verification. Recently there is a growing body of literature advocating descriptor learning without supervision [14], [18], [19], [20], [21], which can sidestep the issues that arise in ground-truth labeling and potentially benefit from broader untapped 3D data.

In this paper, we propose a weakly supervised 3D local descriptor learning method (WSDesc for short), endowed with differentiable voxelization for a learnable local support and a novel registration loss for training without ground-truth alignment information.

• L. Li and M. Ovsjanikov are with LIX, École Polytechnique, IP Paris, France. E-mail: {lli,maks}@lix.polytechnique.fr

• H. Fu is with the School of Creative Media, City University of Hong Kong, Hong Kong. E-mail: hongbofu@cityu.edu.hk

(Corresponding author: Hongbo Fu.)

Specifically, given a pair of 3D point clouds as input, to extract point descriptors, we build upon 3DSmoothNet [9], a 3D CNN-based architecture using a voxel-based representation for 3D local geometry. That work adopts a predefined local support for input voxelization. In contrast, we enable the network to learn the support size in a data-driven manner. In order to back-propagate the gradient to the support size optimization, we propose a differentiable voxelization layer to bridge the gap between the point clouds and their local voxel-based representations.

Next, to train the descriptor extraction network, we introduce a powerful registration loss based on deviation from rigidity of the alignment of the point cloud pair. Prior works [15], [16] normally formulate a registration-based loss by evaluating the difference between the ground-truth and a 3D transformation computed by matching the descriptors of the input point clouds and applying the differentiable singular value decomposition (SVD). The use of the SVD ensures the rigidity of the estimated 3D transformation. Differently, we propose to relax the alignment to be an *affine* transformation, which can be solved in a linear system of equations without additional constraints. Inspired by recent non-rigid correspondence techniques [22], in our proposed registration loss, we enforce the rigidity of the computed affine transformation by promoting its structural properties such as orthogonality and cycle consistency.

Our registration loss is *weakly supervised* in the sense that we only expect the input point clouds to have partial overlap, which is to ensure the *presence* of some underlying rigid alignment between the point clouds. Notably, our registration loss does not require the knowledge of ground-truth alignment information, compared to the above SVD-based works, thus giving rise to a simpler training procedure.

Our main contributions are summarized as follows: (1) We propose a differentiable voxelization layer, enabling in-network conversion from point clouds to a voxel-based representation and allowing data-driven local support optimization; (2) We propose a weakly supervised registration loss based on the deviation from rigidity of 3D transformations between point clouds, effectively guiding the descriptor similarity optimization; (3) Our method shows superior performance on existing geometric registration benchmarks. Our code will be made publicly available¹.

2 RELATED WORK

In this section, we briefly discuss relevant works on both hand-crafted and learned 3D local descriptors as well as learning geometric registration.

Hand-crafted 3D Local Descriptors. A considerable amount of literature has investigated hand-crafted 3D local descriptors. A histogram-based representation is widely used to parameterize local geometry. Generally, the statistical information of points collected by the histograms can be categorized into spatial distributions and geometric attributes [6]. The former is adopted by descriptors like Spin Image [23], 3D Shape Context [24], and USC [25]; while the latter is adopted by descriptors like PFH [26], FPFH [27],

and SHOT [28], [29]. We refer the reader to a comprehensive survey by Guo et al. [6] on the hand-crafted descriptors.

Learned 3D Local Descriptors. With the recent development of deep neural networks, significant research attention has focused on a data-driven approach to encode 3D local geometry into descriptors. Existing works on learned descriptors generally differ in the choices of input parameterizations and network backbones. To parameterize 3D local geometry, researchers have explored many representations such as voxel grids [7], [9], [10], spherical signals [14], multi-view images [13], [30], radial histograms [11], and point pair features [12], [18], [19]. To extract descriptors, various network architectures have been leveraged, such as 3D CNNs [7], [9], Spherical CNNs [31], 2D CNNs [13], [30], MLPs [11], PointNet [12], [18], [32], [33], [34], sparse convolutions [35], [36], [37], and kernel point convolutions [38], [39], [40]. In this work, we base our descriptor extractor on 3DSmoothNet [9], which uses a voxel-based representation and 3D CNNs. Differently from that work, we propose a novel differentiable voxelization layer to enable descriptor extraction with a learnable local support instead of a predefined one, allowing the network to capture more representative local geometry in the descriptors.

Contrastive learning is normally adopted to optimize descriptor similarity. For example, the triplet loss [41], [42] used in 3DSmoothNet or the N-tuple [12], [43] used in PPFNet. To avoid the issues of ground-truth labeling, existing literature has further investigated unsupervised descriptor learning typically by taking auto-encoders [44] with a reconstruction loss [14], [18], [19]. Closely related to the unsupervised approaches, our work proposes a novel registration loss, based on deviation from rigidity, to train the descriptor extractor without requiring ground-truth alignment information.

Learning Geometric Registration. To learn local features well suited for registration, a myriad of studies have incorporated a differentiable registration layer into their networks, such as [15], [16], [45], [46], [47], [48], [49], [50], [51], [52], among many others. Works such as [51], [53], [54], [55] further examine feature learning of putative correspondences for outlier removal in pairwise matching. Training the registration-based networks is normally done by minimizing the errors between the ground-truth and 3D transformations estimated by the networks. Differently, our work zooms in informative local descriptor extraction and investigates a powerful registration-based training loss that penalizes deviation from rigidity for the estimated transformations, without requiring the knowledge of the ground-truth.

3 METHOD

As illustrated in Fig. 2, WSDesc takes a pair of 3D point clouds \mathcal{P} and \mathcal{Q} as input. For a point $\mathbf{p}_i \in \mathcal{P}$, to learn a robust local descriptor, our method has two stages: descriptor extraction and registration-based training. In what follows, we first briefly discuss the pipeline of WSDesc and then present the details in the subsequent subsections.

To extract descriptors, we transform the local 3D geometry into a voxel-based representation [7], [9] for the following reasons. First, as an analogy to 2D images, the

1. <https://github.com/craigleili/WSDesc>

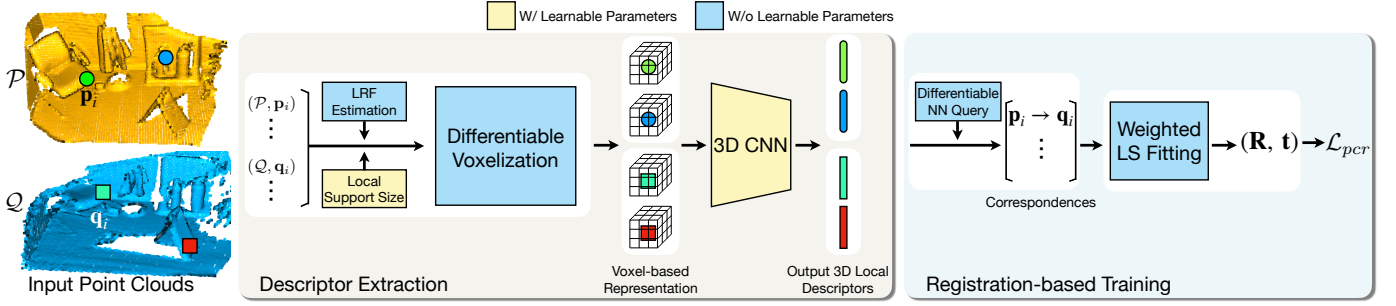


Fig. 2: Overview of our fully differentiable learning pipeline WSDesc. In the descriptor extraction stage, the local geometry of a point \mathbf{p}_i is converted to a voxel-based representation for the 3D CNN-based descriptor extractor. Each voxel grid is transformed by a local reference frame (LRF) for rotation invariance and fitted to a learned local support size. The online point-to-voxel conversion is enabled by the differentiable voxelization layer. In the registration-based training stage, the 3D local descriptors are used for building putative correspondences between point clouds \mathcal{P} and \mathcal{Q} and estimating an affine transformation (\mathbf{R}, \mathbf{t}) by weighted least-squares fitting for our registration loss \mathcal{L}_{pcr} .

voxel-based representation is structured and works well with the off-the-shelf convolutional networks. Second, as we show below, this allows us to optimize the local support size by directly making the voxel grid size a learnable parameter. Third, unlike other local representations (e.g., point pair features [12] and multi-view images [13]) requiring additional information such as normals or colors, the voxel-based representation only depends on point coordinates, and is less sensitive to point density changes [9]. Finally, using a local voxel-based representation can help to avoid overfitting to the global data modality (scene types) and thus offer better generalization ability than dense feature extractors (e.g., KPConv [38], PointConv [56], and SparseConv [35]), as observed in existing works [10], [33], [38].

Our descriptor extraction network is built upon 3DSmoothNet [9], which considers a fixed-size local neighborhood of \mathbf{p}_i in voxelization. We instead enable the network to learn the support size in a data-driven manner. However, the conversion from point clouds to a voxel-based representation is a discrete operation lacking gradient definition. To back-propagate the gradient to a learnable local support, we develop a differentiable voxelization layer based on probabilistic aggregation. After the differentiable voxelization, we use a 3D CNN to extract robust features from the resulting voxel-based representation.

To train the descriptor extractor, we propose a weakly supervised registration loss that enforces rigidity in the pairwise alignment of point clouds with partial overlap. To formulate the registration loss, we first use the descriptors to build putative correspondences between point clouds. Next, inspired by [22], we propose to relax the alignment to be an affine transformation, which can be computed simply through weighted least-squares fitting. This is different from existing end-to-end registration works [15], [16] that directly solve for a rigid transformation via the SVD. Finally, we define our registration loss by promoting structural properties of the computed transformation, including orthogonality and cycle consistency.

3.1 Descriptor Extraction

To extract a robust descriptor for the point \mathbf{p}_i , we first convert its local geometry in \mathcal{P} to a function defined over

a voxel grid \mathcal{V}_i (Fig. 3). Suppose that \mathcal{V}_i has a resolution of $h \times h \times h$, where h is an integer hyperparameter. Prior to voxelization, we need to determine the orientation and the size of the local support of \mathbf{p}_i . The orientation is for ensuring rotation invariance, and the support size determines the amount of local geometry information captured in the final descriptor.

For the orientation, as discussed in [9], it is nontrivial to perform its regression as an integral part of neural networks [12], [57], [58]. Instead, we use the approach based on explicit local reference frame (LRF) estimation [59] in 3DSmoothNet. Specifically, a local patch $\mathcal{S}_i \subset \mathcal{P}$ centered at \mathbf{p}_i is extracted (Fig. 3). The local patch is defined as $\mathcal{S}_i = \{\mathbf{p}_j \mid \|\mathbf{p}_j - \mathbf{p}_i\|_2 \leq r^{\text{LRF}}\}$, where r^{LRF} is a predefined radius. We follow [9] to compute the LRF with resolved sign ambiguity based on eigendecomposition of the covariance matrix of the points in \mathcal{S}_i . We stack the axes of the resulting LRF as column vectors in a matrix $\mathbf{T}_i^{\text{LRF}} \in \mathbb{R}^{3 \times 3}$.

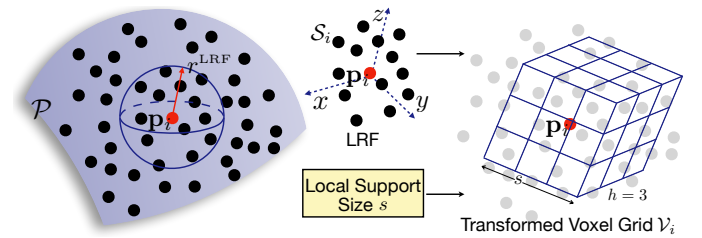


Fig. 3: Illustration of transforming the local geometry of a point \mathbf{p}_i to a voxel-based representation \mathcal{V}_i . The resolution of \mathcal{V}_i is h^3 , and the voxel size is s/h .

For the support size, 3DSmoothNet empirically uses a fixed setting $s = \frac{2r^{\text{LRF}}}{\sqrt{3}}$ as the grid size of \mathcal{V}_i for enclosing \mathcal{S}_i . In contrast, we integrate the voxelization into the network and enable the support size s to be a learnable parameter during training. In this way, the network can gain better flexibility to capture representative local geometry not circumscribed by the predefined \mathcal{S}_i , thus boosting the geometric informativeness in the learned descriptors (Sec. 4.1).

Next, as shown in Fig. 3 (right), we anchor the center of the voxel grid \mathcal{V}_i at \mathbf{p}_i , and rotate \mathcal{V}_i by $\mathbf{T}_i^{\text{LRF}}$ for alignment

with the LRF. The grid size of \mathcal{V}_i is set to the learnable parameter s .

Differentiable Voxelization. The conventional voxelization is non-differentiable and thus cannot back-propagate gradient w.r.t. training losses to the local support size optimization. To address this, we propose point-to-voxel conversion within the network and design a differentiable voxelization layer leveraging probabilistic aggregation [13], [60].

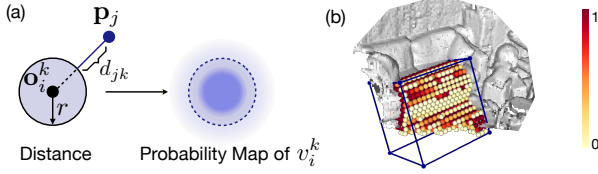


Fig. 4: Illustration of the differentiable voxelization layer. (a) Probability map computation for voxel $v_i^k \in \mathcal{V}_i$. (b) Voxelization example; for clarity, only voxels with values ≥ 0.01 are visualized.

For the k -th voxel v_i^k in the transformed \mathcal{V}_i , we compute its value in a probabilistic manner. To simplify computation, we consider each voxel as a sphere with a radius of $r = \frac{s}{2h}$ [61]. We use p_{jk} to denote the probability of some point $p_j \in \mathcal{P}$ contained in the voxel v_i^k (Fig. 4-a):

$$p_{jk} = \text{sigmoid}(\delta_{jk} \cdot \frac{d_{jk}^2}{\sigma}), \quad d_{jk} = \|\mathbf{p}_j - \mathbf{o}_i^k\|_2 - r, \quad (1)$$

where $\mathbf{o}_i^k \in \mathbb{R}^3$ denotes the voxel center, δ_{jk} is a sign indicator $\delta_{jk} = -\text{sign}(d_{jk})$, and σ controls the sharpness of the probability distribution. We then propose to aggregate the voxel value of v_i^k as:

$$v_i^k = 1 - \prod_{p_j \in \mathcal{P}} (1 - p_{jk}). \quad (2)$$

In Eq. (2), the voxel value can be viewed as the probability of having at least one point contained in the voxel v_i^k . The resulting voxel values are a continuous function of the point cloud coordinates and of the learnable support size s , and thus Eq. (2) is fully differentiable. We also note that Eq. (2) is permutation invariant to the input points and less sensitive to point density changes, thus accommodating an arbitrary number of points. In Fig. 4-b, we visualize a voxelization example, showing that probabilistic aggregation can approximate the voxelization procedure to capture the structure of local geometry.

3D CNN. After converting the local geometry of \mathbf{p}_i to a signal on the voxel grid \mathcal{V}_i , we use the 3D CNN-based architecture from [9] to compute a descriptor $\mathbf{f}_i \in \mathbb{R}^n$. Let $\mathbf{f}_i = \mathcal{G}_\Phi(\mathcal{V}_i)$, where Φ denotes the learnable parameters of the 3D CNN \mathcal{G} . Fig. 5 presents the details of \mathcal{G} . Specifically, the network \mathcal{G} is comprised of six convolutional layers, which progressively down-sample the input \mathcal{V}_i . Each convolutional layer is followed by a normalization layer [62] and a ReLU activation layer. The output of the last convolutional layer is flattened and fed to a fully connected layer followed by ℓ^2 normalization, resulting a unit-length n -dimensional local descriptor \mathbf{f}_i .

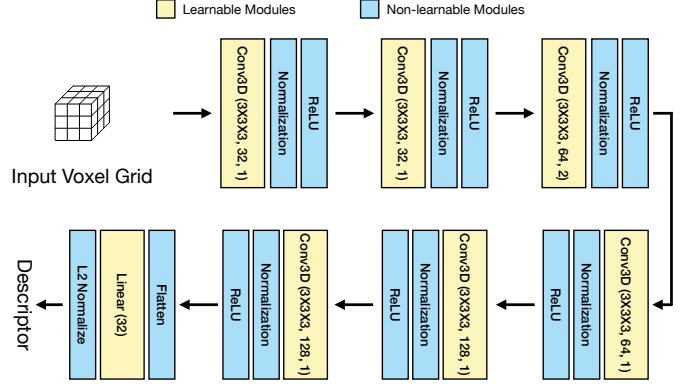


Fig. 5: The 3D CNN-based descriptor extractor. The numbers in the parentheses of each 3D convolution layer represent kernel size, number of filters, and stride, respectively. The linear layer outputs n -dimensional descriptors.

3.2 Registration-based Training

In this section, we propose a novel registration loss for optimizing descriptor similarity across point clouds. To formulate the loss, we assume the input point clouds \mathcal{P} and \mathcal{Q} to have partial overlap. Specifically, partial overlap means the existence of corresponding points between \mathcal{P} and \mathcal{Q} , and the overlap ratio [17] is normally defined as the number of corresponding points over $\min\{|\mathcal{P}|, |\mathcal{Q}|\}$. Having partial overlap is a very weak supervision for ensuring the *presence* of some underlying rigid alignment. Note that this weak supervision can be easily satisfied in existing 3D point cloud datasets [7], such as by leveraging temporally adjacent point clouds due to the continuity of the scanning process in practice or by employing a RANSAC-based overlap detection procedure used in the reconstruction pipelines [63], [64]. We stress that our loss calculation *does not* require the knowledge of the ground-truth transformation between \mathcal{P} and \mathcal{Q} .

Putative Correspondences. We construct a set of putative correspondences between \mathcal{P} and \mathcal{Q} by a nearest neighbor (NN) search between their extracted local descriptors. Note that the hard assignments of the NN search are not differentiable. Thus we adopt a differentiable NN search based on the softmax operation, which has been used in the prior works [49], [65]. We use \mathbf{C} to denote the resulting set of correspondences $\{c_i = (\mathbf{p}_i \in \mathcal{P}, \mathbf{q}_i \in \mathcal{Q})\}$, where for a point \mathbf{p}_i its closest neighbor in the descriptor space is the point \mathbf{q}_i .

Transformation Estimation. Given the constructed \mathbf{C} , we solve for a 3D transformation aligning \mathcal{P} to \mathcal{Q} and then formulate our registration loss by promoting structural properties of the resulting transformation. We denote the transformation as (\mathbf{R}, \mathbf{t}) , where \mathbf{R} is a 3×3 affine matrix and \mathbf{t} is a 3D vector. We stack the points $\{\mathbf{p}_i\}$ in \mathbf{C} as columns of a matrix $\mathbf{P}_c \in \mathbb{R}^{3 \times |\mathbf{C}|}$, and \mathbf{Q}_c is a similar matrix stacking the points $\{\mathbf{q}_i\}$ in \mathbf{C} . To estimate (\mathbf{R}, \mathbf{t}) , we minimize the following weighted quadratic error:

$$\mathbf{R}, \mathbf{t} = \arg \min_{\mathbf{R}, \mathbf{t}} \|(\mathbf{R}\mathbf{P}_c + \mathbf{t}\mathbf{1} - \mathbf{Q}_c)\mathbf{W}\|^2, \quad (3)$$

where $\mathbf{1}$ is a row vector filled with 1, and the ma-

trix \mathbf{W} denotes the correspondence confidence: $\mathbf{W} = \text{diag}(w_1, \dots, w_i, \dots, w_{|\mathbf{C}|})$ computed as follows.

For the i -th correspondence, we define its confidence as

$$w_i = w_i^f \cdot w_i^{sm}, \quad (4)$$

where w_i^f is the similarity of the two points in the descriptor space, and w_i^{sm} is the compatibility with other correspondences in the 3D space. Specifically, given the i -th correspondence $c_i = (\mathbf{p}_i, \mathbf{q}_i)$ with local descriptors $\mathbf{f}_{\mathbf{p}_i}$ and $\mathbf{f}_{\mathbf{q}_i}$, the first term w_i^f is computed by the softmax function:

$$w_i^f = \frac{\exp(-\|\mathbf{f}_{\mathbf{p}_i} - \mathbf{f}_{\mathbf{q}_i}\|_2)}{\sum_{\mathbf{q}_j \in \mathcal{Q}} \exp(-\|\mathbf{f}_{\mathbf{p}_i} - \mathbf{f}_{\mathbf{q}_j}\|_2)}. \quad (5)$$

For the second term w_i^{sm} , we follow [54] and leverage the well-known spectral matching technique [66], which finds consistent correspondences based on their isometric compatibility. Spectral matching does not require ground-truth labels and is fully differentiable. Details of this technique can be found in the supplementary material and in [66], [67].

We note that while there exist recent deep outlier filtering works [51], [53], [55] using neural networks to regress correspondence confidence, we adopt Eq. (4) for the following reasons. First, those deep outlier filtering methods require to be fully supervised for the regression, and thus cannot be adapted straightforwardly in our work, which strives to use very weak supervision for local descriptor learning. Second, the networks used in the above works typically have a myriad of trainable parameters, which can significantly increase the complexity of our network. In contrast, the above computation for \mathbf{W} is non-parametric and fully differentiable, allowing gradient back-propagation to the descriptor extractor. At test time, following 3DSmoothNet, we combine our extracted local descriptors with RANSAC [68] for robust point cloud registration.

Registration Loss. We formulate our registration loss based on the transformation estimation by Eq. (3). Prior registration-based works [15], [16], [48], [49] employ the SVD to solve a similar optimization problem, ensuring that (\mathbf{R}, \mathbf{t}) is a rigid transformation that can be compared with the ground-truth during training. Differently, we propose to relax (\mathbf{R}, \mathbf{t}) to be an affine transformation and leverage its *deviation* from rigidity as the driving force for local descriptor training, as done in the context of non-rigid shape matching [22]. Specifically, with the affine relaxation, we can solve Eq. (3) in a least-squares sense as follows:

$$[\mathbf{R} \ \mathbf{t}] = \mathbf{Q}_c \mathbf{W} (\bar{\mathbf{P}}_c \mathbf{W})^+, \quad (6)$$

where $\bar{\mathbf{P}}_c \in \mathbb{R}^{4 \times |\mathbf{C}|}$ is \mathbf{P}_c in homogeneous coordinates, and $(\cdot)^+$ is the Moore–Penrose inverse, which is differentiable. The inverse transformation $(\mathbf{R}', \mathbf{t}')$ aligning \mathcal{Q} to \mathcal{P} is computed similarly.

We denote our registration loss as \mathcal{L}_{pcr} , consisting of the following terms:

$$\mathcal{L}_o = (\|\mathbf{R}^\top \mathbf{R} - \mathbf{I}\|_1 + \|\mathbf{R}'^\top \mathbf{R}' - \mathbf{I}\|_1)/2, \quad (7)$$

$$\mathcal{L}_c = \|\mathbf{R}\mathbf{R}' - \mathbf{I}\|_1 + \|\mathbf{R}\mathbf{t}' + \mathbf{t}\|_1, \quad (8)$$

$$\mathcal{L}_{pcr} = \lambda_o \mathcal{L}_o + \lambda_c \mathcal{L}_c, \quad (9)$$

where λ_o and λ_c are the weights for the loss terms, and \mathbf{I} is an identity matrix. The term \mathcal{L}_o regularizes the orthogonal-

ity of \mathbf{R} and \mathbf{R}' . The term \mathcal{L}_c enforces the cycle consistency by requiring

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}' & \mathbf{t}' \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}\mathbf{R}' & \mathbf{R}\mathbf{t}' + \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \mathbf{I}. \quad (10)$$

Discussion. We briefly discuss the intuition behind \mathcal{L}_{pcr} for providing training signals without requiring ground-truth alignment information. If \mathbf{C} is a set of inlier correspondences, the underlying rigid transformation can always be recovered by Eq. (6). However, outlier correspondences may exist in \mathbf{C} during the local descriptor training, making the transformation by Eq. (6) deviate from rigidity. Minimizing this deviation in \mathcal{L}_{pcr} will translate to a tendency of promoting weights for the inliers and lowering weights for the outliers (Eq. (3)). The gradient w.r.t. \mathcal{L}_{pcr} can be back-propagated through the differentiable correspondence matching layer to the descriptor extractor, thus enabling the similarity optimization between local descriptors.

3.3 Implementation

We implemented our method with PyTorch [69]. Following 3DSmoothNet [9], we set $r^{\text{LRF}} = 0.3$, the voxel grid resolution $h = 16$, and the descriptor dimension $n = 32$. In the differentiable voxelization, we set $\sigma = 10^{-3}$. For the term weights in \mathcal{L}_{pcr} , we use $\lambda_o = 1$ and $\lambda_c = 1$. We adopt Adam [70] with a learning rate of 10^{-3} as the network optimizer. We use two point cloud datasets, 3DMatch [7] and ModelNet40 [71]. During training on the 3DMatch dataset, we sample 512 keypoints in each point cloud with farthest point sampling for sparse descriptor extraction and matching. The network is trained for 16K steps. For the ModelNet40 dataset, we sample 128 keypoints in each point cloud during training. The network is trained for 20K steps.

4 EXPERIMENTS

We evaluate the performance of our proposed WSDesc on existing point cloud registration benchmarks including 3DMatch (Sec. 4.1), ModelNet40 (Sec. 4.3), and ETH (Sec. 4.4). The 3DMatch dataset [7] consists of point clouds of indoor scene scans. The ModelNet40 dataset [71] consists of object-centric point clouds generated from CAD models. The ETH dataset [72] consists of outdoor scene scans and is used for descriptor generalization test. We present the ablation study in Sec. 4.2.

4.1 3DMatch Dataset

The 3DMatch dataset is widely adopted for evaluating the local descriptor performance on geometric registration. In total, there are 62 indoor scenes: 54 of them for training and validation, and 8 of them for testing. In the test set, the number of points per point cloud is $\sim 13\text{K}$ on average, and each point cloud has 5K randomly sampled keypoints for sparse descriptor extraction and matching.

Evaluation Metrics. Following [9], [35], we compute the inlier ratio (IR), feature-match recall (FMR), and registration recall (RR) on 3DMatch. Consider a set of testing point cloud pairs $\Lambda = \{(\mathcal{P}, \mathcal{Q})\}$ with \mathcal{P} and \mathcal{Q} having at least 30% overlap. For each point cloud pair, a set of putative

correspondences Ω between keypoints is built by finding mutually closest neighbors in the descriptor space:

$$\Omega = \{(\mathbf{p}_i \in \mathcal{P}, \mathbf{q}_i \in \mathcal{Q}) | \mathcal{D}(\mathbf{p}_i) = \text{NN}(\mathcal{D}(\mathbf{q}_i), \mathcal{D}(\mathcal{P})) \wedge \mathcal{D}(\mathbf{q}_i) = \text{NN}(\mathcal{D}(\mathbf{p}_i), \mathcal{D}(\mathcal{Q}))\}, \quad (11)$$

where \mathcal{D} denotes a specific descriptor extractor, and $\text{NN}(\cdot, \cdot)$ is the nearest neighbor query based on the ℓ^2 distance.

IR measures the fraction of correct correspondences between \mathcal{P} and \mathcal{Q} as follows:

$$\text{IR} = \frac{1}{|\Omega|} \sum_{(\mathbf{p}_i, \mathbf{q}_i) \in \Omega} \mathbb{I}[\|\mathcal{T}^*(\mathbf{p}_i) - \mathbf{q}_i\|_2 < \tau_1], \quad (12)$$

where $\mathcal{T}^*(\cdot)$ denotes the ground-truth transformation, $\tau_1 = 0.1$ m is the inlier distance threshold, and $\mathbb{I}[\cdot]$ is the Iverson bracket.

FMR measures the fraction of point cloud pairs in Λ , for which a RANSAC-based [68] registration pipeline can recover the transformations with high confidence [12]:

$$\text{FMR} = \frac{1}{|\Lambda|} \sum_{(\mathcal{P}, \mathcal{Q}) \in \Lambda} \mathbb{I}[\text{IR} > \tau_2], \quad (13)$$

where τ_2 is the inlier ratio threshold in the range of $[0.05, 0.2]$.

RR examines the performance of local descriptors in an actual reconstruction system. For a pair of point clouds \mathcal{P} and \mathcal{Q} , let $\hat{\mathcal{T}}(\cdot)$ denote the estimated transformation by a registration pipeline. Suppose Ω^* is the set of ground-truth correspondences, and their root-mean-square error is computed as follows:

$$\text{RMSE} = \sqrt{\frac{1}{|\Omega^*|} \sum_{(\mathbf{p}_i, \mathbf{q}_i) \in \Omega^*} \|\hat{\mathcal{T}}(\mathbf{p}_i) - \mathbf{q}_i\|_2^2}. \quad (14)$$

RR is computed as the fraction of point cloud pairs in Λ with $\text{RMSE} < 0.2$ m.

Comparisons. We compare our descriptor with hand-crafted descriptors and existing descriptor learning methods that do not rely on ground-truth transformations. The former includes FPFH [27] and SHOT [29], which have been implemented in the PCL library [73] and have descriptor dimensions of 33 and 352, respectively. The latter includes PPF-FoldNet [18], CapsuleNet [19], and S2CNN [14]. Their implementations are based on publicly available codebases²³⁴, and they all have a descriptor dimension of 512. The high dimensionality makes NN search computationally inefficient. Thus, for a direct comparison with the state-of-the-art S2CNN, we also implemented an S2CNN variant that outputs 32-dimensional descriptors.

Table 1 (bottom) shows the IR comparison for the methods w/o GT. It can be observed that our descriptor obtains the highest IR (42.3%). WSDesc outperforms S2CNN (32) by 13.2 percentage points and S2CNN (512) by 7.8 percentage points, indicating the better quality of correspondences built by our method. For the FMR comparison, in the case of $\tau_2 = 0.05$, our descriptor achieves an FMR of 95.1%, slightly better than S2CNN (512). Yet $\tau_2 = 0.05$ is a relatively easy threshold [9], and the descriptor performance tends to be

TABLE 1: Performance (%) on the 3DMatch dataset w.r.t. inlier ratio (IR), feature-match recall (FMR), and registration recall (RR). The best and the second best results among the methods w/o GT are highlighted. (GT – ground-truth labels; Dim. – descriptor dimension.)

τ_2	GT	Dim.	IR \uparrow	FMR \uparrow		RR \uparrow
				0.05	0.2	
3DMatch	w/	512	8.3	57.3	7.7	51.9
CGF	w/	32	10.1	60.6	12.3	51.3
PPFNet	w/	64	-	62.3	-	71.0
3DSmoothNet	w/	32	36.0	95.0	72.9	78.8
FCGF	w/	32	-	95.2	67.4	82.0
D3Feat	w/	32	40.7	95.8	75.8	82.2
LMVD	w/	32	46.1	97.5	86.9	81.3
SpinNet	w/	32	-	97.6	85.7	-
WSDesc (BH)	w/	32	54.3	96.7	88.2	81.4
FPFH	w/o	33	9.3	59.6	10.1	54.8
SHOT	w/o	352	14.9	73.3	26.9	59.4
PPF-FoldNet	w/o	512	20.9	83.8	41.0	69.0
CapsuleNet	w/o	512	16.9	82.5	31.3	67.6
S2CNN	w/o	512	34.5	94.6	70.3	78.4
S2CNN	w/o	32	29.1	92.4	59.9	73.9
WSDesc	w/o	32	42.3	95.1	77.7	80.0

saturated. In the harder case of $\tau_2 = 0.2$, our descriptor obtains 77.7%, significantly outperforming S2CNN (32) by 17.8 percentage points and S2CNN (512) by 7.4 percentage points. Further, Fig. 6 plots the FMR performance w.r.t. different τ_2 values, and our descriptor shows lower sensitivity to the inlier ratio τ_2 , which can be ascribed to the superior IR performance. For the RR metric, our descriptor also achieves the best performance (80.0%) and outperforms S2CNN (32) by 6.1 percentage points.

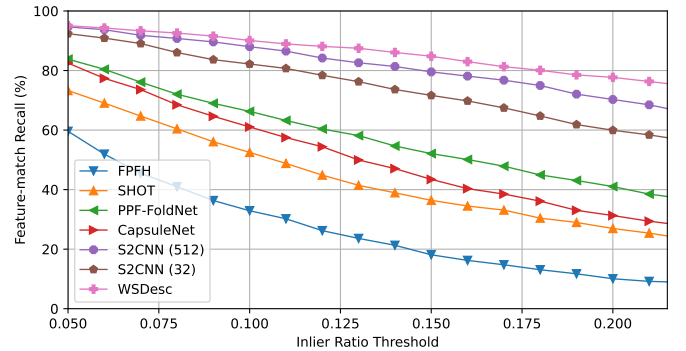


Fig. 6: Feature-match recall of the methods w/o GT on 3DMatch, with a varying inlier ratio threshold τ_2 .

In Table 1 (top) we include the performance of supervised descriptor learning methods, including 3DMatch [7], CGF [11], PPFNet [12], 3DSmoothNet [9], FCGF [35], D3Feat [38], LMVD [13], and SpinNet [10]. Interestingly, our WSDesc achieves even better performance than the supervised 3DSmoothNet, which can be ascribed mainly to our learnable support that allows capturing the local context in an appropriate size. To validate this, WSDesc (BH) in Table 1 is the oracle performance of our method, that is, we use the same supervised batch-hard loss as 3DSmoothNet for training instead of \mathcal{L}_{per} . It can be found that using the

2. <https://github.com/XuyangBai/PPF-FoldNet>

3. <https://github.com/yongheng1991/3D-point-capsule-networks>

4. <https://github.com/jonas-koehler/s2cnn>

learnable support significantly improves the performance of 3DSmoothNet, making it comparable to the state-of-the-arts [10], [13], [38].

Rotation Invariance. To test the rotation invariance, following [18], we used the rotated 3DMatch dataset, where each point cloud is rotated with randomly sampled axes and angles in $[0, 2\pi]$. Table 2 reports the performance of the compared methods. Our descriptor has the best IR (39.8%), FMR (74.3% at $\tau_2 = 0.2$), and RR (78.5%) scores.

TABLE 2: Performance (%) on the rotated 3DMatch dataset. $\tau_2 = 0.2$ for FMR.

	Dim.	IR \uparrow	FMR \uparrow	RR \uparrow
FPFH	33	9.3	10.0	55.3
SHOT	352	14.9	26.9	61.6
PPF-FoldNet	512	21.0	41.6	68.7
CapsuleNet	512	16.8	31.9	68.0
S2CNN	512	<u>34.6</u>	<u>70.5</u>	<u>78.3</u>
S2CNN	32	29.2	59.5	75.2
WSDesc	32	39.8	74.3	78.5

Qualitative Visualization. Fig. 7 (top) visualizes challenging point cloud registration examples with a large portion of flat surfaces, where our descriptor demonstrates better robustness. Fig. 8 shows a failure case for our local descriptor, which we ascribe partly to the difficulty of extracting discriminative local descriptors for the thin structures of the two point clouds with the low-resolution ($h = 16$) voxel grids. More qualitative registration examples can be found in the supplementary material.

Computation Time. Table 3 shows the running time comparisons. The results were collected on a desktop computer with an Intel Core i7 @ 3.6GHz and an NVIDIA GTX 1080Ti GPU. Note that for the input preparation, PPF-FoldNet and CapsuleNet compute point pair features; S2CNN performs spherical representation conversion; and WSDesc estimates LRFs. Overall, our method has the best running time, while S2CNN is computationally much slower.

In Table 4, we further collect the running time of RANSAC-based registration [64] with different descriptors. *NN Query* denotes the search time for building a putative correspondence set (Eq. (11)) with an efficient proximity search structure, such as a k -d tree. S2CNN (32) and WSDesc have better efficiency due to the lower descriptor dimensionality. WSDesc has the best RANSAC time owing to its high IR performance.

TABLE 3: Average running time (ms) of local descriptor extraction on the 3DMatch dataset.

	Dim.	Input Prep.	Inference	Total
PPF-FoldNet	512	4.01	0.44	4.45
CapsuleNet	512	3.94	7.19	11.13
S2CNN	512	6.44	26.56	33.00
S2CNN	32	6.44	26.58	33.02
WSDesc	32	0.37	3.00	3.37

4.2 Ablation Study

To get a better understanding of the contribution of the network components used by WSDesc, we perform an ex-

TABLE 4: Average running time (ms) of pairwise registration on the 3DMatch dataset.

	Dim.	NN Query	RANSAC
PPF-FoldNet	512	9795.76	5.55
CapsuleNet	512	3652.68	6.78
S2CNN	512	13926.57	2.95
S2CNN	32	657.77	3.77
WSDesc	32	979.08	2.84

tensive ablation study on the 3DMatch dataset.

TABLE 5: Ablation studies on 3DMatch. $\tau_2 = 0.2$ for FMR.

	IR \uparrow	FMR \uparrow	RR \uparrow
WSDesc	42.3	77.7	80.0
w/o LLS	25.3	51.6	72.8
w/ LP	38.9	72.9	78.0
w/ GP	40.0	74.2	78.7
w/o LRF	21.6	39.5	50.8
w/o w_i	6.4	5.4	31.2
w/o w_i^f	8.0	6.4	45.6
w/o w_i^{sm}	32.2	63.3	74.3
w/o \mathcal{L}_o	39.4	75.3	77.5
w/o \mathcal{L}_c	39.5	75.7	78.0

Learnable Local Support. First, to validate the effectiveness of using a learnable local support (Sec. 3.1), we test a variant of WSDesc using the predefined fixed-size local support from 3DSmoothNet. We keep the loss \mathcal{L}_{per} and other settings unchanged. In Table 5, the performance of this variant is listed as *w/o LLS*, which is worse than our full model. This indicates that making the local support size optimizable enables the network to extract more informative descriptors.

Next, we discuss alternative designs for the learnable local support. In WSDesc, we propose to treat the local support size s as a universal learnable parameter. One might consider employing a sub-network to estimate the support size. We implemented two variants for such a design. One variant is based on a local sub-network that estimates a local support size s_i individually for each point $\mathbf{p}_i \in \mathcal{P}$. Specifically, to regress s_i , we feed the local patch \mathcal{S}_i of \mathbf{p}_i , cropped with the predefined radius r^{LRF} , to a mini-PointNet [12], [34]. Thus the local support size s_i is dependent on the local geometry of \mathbf{p}_i , but the mini-PointNet may need to hallucinate the support size due to the missing of points outside \mathcal{S}_i . We list this variant as *w/ LP* in Table 5.

The other variant is based on a global PointNet [34] for estimating a local support size s_i individually for each \mathbf{p}_i . Specifically, we use the semantic segmentation architecture of PointNet [34] for the estimation. We feed the point cloud \mathcal{P} to the PointNet backbone, which regresses a scalar value s_i for each point \mathbf{p}_i in a single forward pass. In this case, the local support size s_i is coupled with the global scene structure (i.e., \mathcal{P}) and the local geometry of \mathbf{p}_i . We list this variant as *w/ GP* in Table 5.

It is observed that both the *w/ LP* and *w/ GP* variants perform worse than WSDesc, which uses a universal learn-

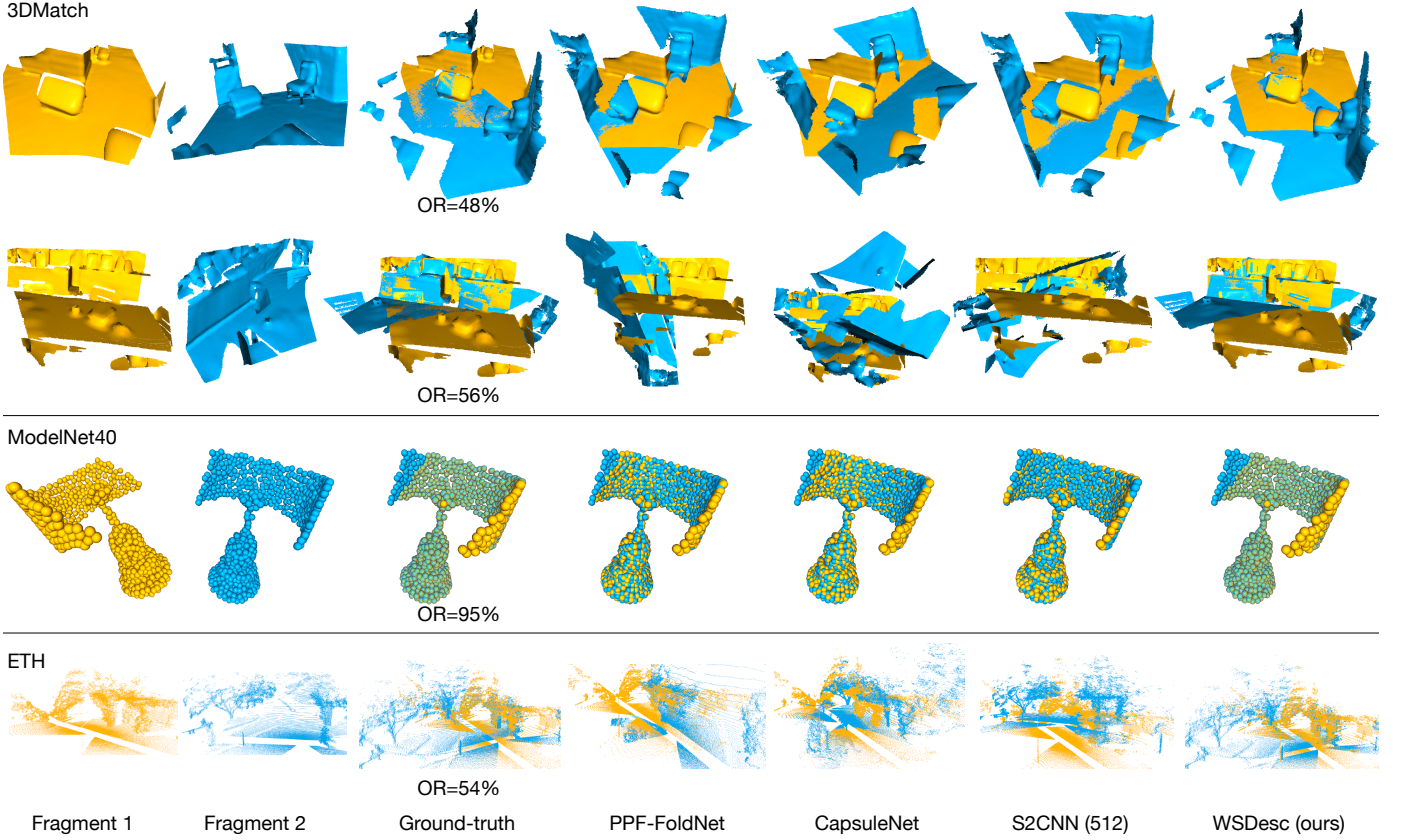


Fig. 7: Qualitative examples of point cloud registration by RANSAC with different descriptors on the 3DMatch, ModelNet40, and ETH datasets. The percentages denote the overlap ratio (OR) between fragments 1 and 2.

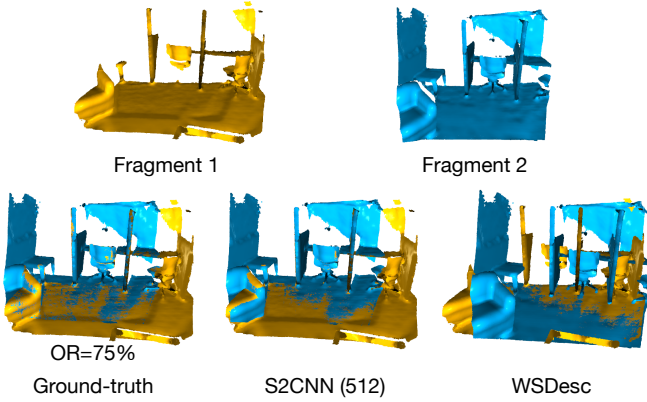


Fig. 8: A failure case of point cloud registration with our local descriptor.

able local support size (converged to 1.004m after training). Interestingly, we found that the sizes s_i learned by w/LP are on average 1.719m ($\pm 5.0 \times 10^{-6}$), and the ones learned by w/GP are on average 1.701m ($\pm 2.1 \times 10^{-2}$). The small variance phenomenon of the two variants further confirms our choice of using a universal learnable local support size. This is possibly because robust descriptor matching typically requires optimized yet consistent support sizes for rigidly aligning point clouds, especially in the presence of significant partiality and noise. Although the experimental

results indicate that an exhaustive parameter search seems possible, we stress that it is computationally expensive and needs to be repeated on each used point cloud dataset. Instead, our proposal offers higher efficiency for learning informative local descriptors in an end-to-end manner.

Local Reference Frame. In Sec. 3.1, we follow 3DSmoothNet to use the LRF estimation for ensuring rotation invariance in the descriptors. We test a variant of WSDesc without the LRF estimation and report its performance as *w/o LRF* in Table 5. Unsurprisingly, without LRF, the local descriptors have reduced descriptiveness, leading to worse geometric registration performance. Our finding echoes a similar observation of the LRF contribution in the 3DSmoothNet work. During training, due to the lack of rotation invariance in the descriptors, the putative correspondences across point clouds can be spurious in the estimation of affine transformations between point clouds, making \mathcal{L}_{per} ineffective to provide training signals.

Transformation Estimation. To estimate the affine transformation, we adopt a weighted quadratic formulation in Eq. (3), where the weight w_i is comprised of two terms w_i^f and w_i^{sm} (Eq. (4)). We test a variant of WSDesc without the weights (i.e., by setting $w_i = 1$), and we show its performance as *w/o w_i* in Table 5. Further, we remove one of the weight terms from w_i and list the performance of these two variants as *w/o w_i^f* and *w/o w_i^{sm}* in Table 5, respectively. We observe that the network *w/o w_i* fails on 3DMatch, and so does *w/o w_i^f* . This is likely because without

the weight terms, especially, w_i^f computed directly from the extracted descriptors, the gradients w.r.t. \mathcal{L}_{pcr} can only pass through positions and the differentiable NN search, making it less effective to flow into the descriptor extractor for optimization [51]. As also shown by w/o w_i^{sm} , adding w_i^f to the network results in reasonable performance on 3DMatch. Nevertheless, incorporating spectral matching indeed boosts the learning of descriptors and thus the registration performance.

Registration Loss. Our training loss \mathcal{L}_{pcr} penalizes deviation from rigidity for the estimated affine transformations without requiring the ground-truth. \mathcal{L}_{pcr} is comprised of the orthogonality loss \mathcal{L}_o and the cycle consistency loss \mathcal{L}_c . We study the contribution of the two loss terms by removing one of them during training and keeping other settings unchanged. In the bottom of Table 5, we show the results of the two experiments as w/o \mathcal{L}_o and w/o \mathcal{L}_c , respectively. Note that there is noticeable performance degradation when using each loss term alone, and combining both loss terms produces the best results.

4.3 ModelNet40 Dataset

We also perform comparisons with existing learning-based registration methods on the ModelNet40 benchmark introduced by [15]. The dataset has 40 man-made object categories. There are 9,843 point clouds for training and 2,468 point clouds for testing. To generate point cloud pairs, a new point cloud is obtained by transforming each testing point cloud with a random rigid transformation. The rotation angle along each axis is sampled in the range of $[0^\circ, 45^\circ]$, and the 3D translation offset is sampled in $[-0.5, 0.5]$. To synthesize partial overlapping for a point cloud pair, 768 nearest neighbors of a randomly placed point in 3D space are collected in each point cloud.

Metrics. Given rotations and translations estimated by a specific registration method, we follow [15] to compare them with the ground-truth by measuring root-mean-square error (RMSE) and coefficient of determination (R^2). The rotation errors are computed with the Euler angle representation in degrees.

Comparisons. In Table 6 (bottom), we evaluate existing axiomatic registration methods, including ICP [74], Go-ICP [75], and FGR [63], and the RANSAC-based registration methods (w/o GT) previously tested on 3DMatch, including PPF-FoldNet, CapsuleNet, S2CNN, and our WSDesc. Besides, two recent unsupervised learning-based registration methods, ARL [76] and RMA-Net [21], are considered⁵⁶. However, the released codebase for RMA-Net failed to produce reasonable results on the ModelNet40 benchmark, and thus we include only the comparison with ARL. In Table 6 (top), we also show the performance of learning-based registration methods *with* supervision, including PointNetLK [45], DCP [16], PRNet [15], DeepGMR [77], RPM-Net [48], Predator [39], and OMNet [78]. In Table 6 (bottom), our WSDesc achieves the best performance across all of the computed metrics, compared to the other methods w/o GT, and is even on a par with the state-of-the-art RPM-Net, which is a supervised method highly specialized for

object-centric datasets. Fig. 7 (middle) shows qualitative registration examples, where our descriptor leads to more accurate alignment. We further perform comparisons on the noisy ModelNet40 benchmark by [15]. Table 7 reports the registration results, and our WSDesc shows better robustness to noise.

TABLE 6: Testing on unseen point clouds of the ModelNet40 dataset w.r.t. rotation (**R**) and translation (**t**) estimations. The best and the second best results among the methods w/o GT are highlighted.

	GT	RMSE(R)↓	R^2 (R)↑	RMSE(t)↓	R^2 (t)↑
PointNetLK	w/	16.735	-0.654	0.045	0.975
DCP-v2	w/	6.709	0.732	0.027	0.991
PRNet	w/	3.199	0.939	0.016	0.997
DeepGMR	w/	19.156	-1.164	0.037	0.983
RPM-Net	w/	1.290	0.990	0.005	1.000
Predator	w/	1.875	0.979	0.017	0.997
OMNet	w/	4.280	0.891	0.019	0.996
ICP	w/o	33.683	-5.696	0.293	-0.037
Go-ICP	w/o	13.999	-0.157	0.033	0.987
FGR	w/o	11.238	0.256	0.030	0.989
ARL	w/o	8.527	0.570	0.029	-0.046
PPF-FoldNet	w/o	2.285	0.969	<u>0.013</u>	<u>0.998</u>
CapsuleNet	w/o	<u>2.180</u>	<u>0.972</u>	<u>0.013</u>	<u>0.998</u>
S2CNN (512)	w/o	3.069	0.944	0.017	0.997
S2CNN (32)	w/o	3.234	0.938	0.014	0.998
WSDesc	w/o	1.187	0.992	0.008	0.999

TABLE 7: Testing on unseen point clouds of the ModelNet40 dataset augmented with Gaussian noise.

	GT	RMSE(R)↓	R^2 (R)↑	RMSE(t)↓	R^2 (t)↑
PointNetLK	w/	19.939	-1.343	0.057	0.960
DCP-v2	w/	6.883	0.718	0.028	0.991
PRNet	w/	4.323	0.889	0.017	0.995
DeepGMR	w/	19.758	-1.299	0.030	0.989
RPM-Net	w/	1.870	0.979	0.011	0.998
Predator	w/	1.893	0.979	0.009	0.999
OMNet	w/	4.504	0.880	0.021	0.995
ICP	w/o	35.067	-6.252	0.294	-0.045
Go-ICP	w/o	12.261	0.112	0.028	<u>0.991</u>
FGR	w/o	27.653	-3.491	0.070	0.941
ARL	w/o	7.973	0.624	0.027	0.023
PPF-FoldNet	w/o	<u>4.151</u>	<u>0.899</u>	0.009	0.999
CapsuleNet	w/o	4.274	0.893	0.009	0.999
S2CNN (512)	w/o	5.221	0.840	<u>0.007</u>	0.999
S2CNN (32)	w/o	5.040	0.850	0.009	0.999
WSDesc	w/o	3.500	0.928	0.006	0.999

4.4 Generalization to ETH Dataset

To evaluate the generalization ability of our 3D local descriptors, we follow [9] to conduct experiments on the ETH dataset [72]. This dataset consists of point clouds of four outdoor scenes, which are mostly laser scans of vegetation like trees and bushes. The number of points per point cloud is $\sim 100K$ on average, and 5K keypoints are randomly sampled in each point cloud for matching. For learning-based descriptors, to test their generalization ability, we directly reuse the networks trained on 3DMatch (Sec. 4.1) and use the ETH dataset only as a test set.

Comparisons. Table 8 shows comparisons for the 3D local descriptors (Sec. 4.1) in terms of the FMR metric

5. <https://github.com/Dengzhi-USTC/A-robust-registration-loss>

6. <https://github.com/WanquanF/RMA-Net>

($\tau_2 = 0.05$). We observe that WSDesc achieves better performance than the other methods w/o GT in Table 8 (bottom), demonstrating the superior generalization ability. Fig. 7 (bottom) shows qualitative registration examples from this challenging outdoor-scene dataset. We further provide the running time comparisons in Table 9, which were collected on a server with an Intel Xeon CPU @ 2.20GHz and an NVIDIA GeForce RTX 2080Ti GPU. The speed of our method is comparable with PPF-FoldNet and influenced by the significantly increased number of points per point cloud, due to the formulation of differentiable voxelization in Eq. 2. We will investigate further optimizations on this network layer in future work.

TABLE 8: FMR performance (%) on the ETH dataset ($\tau_2 = 0.05$). The learned descriptors are only trained on the 3DMatch dataset.

	GT	Gazebo		Wood		Avg.
		Sum.	Wint.	Sum.	Aut.	
3DMatch	w/	22.8	8.7	22.4	13.9	16.9
CGF	w/	38.6	15.2	19.2	12.2	21.3
3DSmoothNet	w/	91.3	84.1	72.8	67.8	79.0
FCGF	w/	22.8	10.0	14.8	16.8	16.1
D3Feat	w/	85.9	63.0	49.6	48.0	61.6
LMVD	w/	85.3	72.0	84.0	78.3	79.9
SpinNet	w/	92.9	91.7	94.4	92.2	92.8
PPFH	w/o	40.2	15.2	24.0	14.8	23.6
SHOT	w/o	73.9	45.7	64.0	60.9	61.1
PPF-FoldNet	w/o	39.7	24.2	25.6	19.1	27.2
CapsuleNet	w/o	33.2	15.2	22.4	17.4	22.0
S2CNN (512)	w/o	79.9	58.1	63.2	53.9	63.8
S2CNN (32)	w/o	70.7	46.7	54.4	45.2	54.2
WSDesc	w/o	<u>78.3</u>	<u>77.2</u>	95.2	93.9	86.1

TABLE 9: Average running time (ms) of local descriptor extraction on the ETH dataset.

	Dim.	Input Prep.	Inference	Total
PPF-FoldNet	512	16.65	0.30	16.95
CapsuleNet	512	16.93	2.43	19.35
S2CNN	512	5.61	17.56	23.17
S2CNN	32	5.53	17.61	23.14
WSDesc	32	9.50	6.75	16.25

5 CONCLUSION, LIMITATIONS & FUTURE WORK

We have presented WSDesc to learn point descriptors for robust point cloud registration in a weakly supervised manner. Our framework is built upon a voxel-based representation and 3D CNNs for descriptor extraction. To enrich geometric information in the learned descriptors, we propose to learn the local support size in the online point-to-voxel conversion with differentiable voxelization. We introduce a powerful weakly supervised registration loss to guide the learning of descriptors. Extensive experiments show that our descriptors achieve superior performance on existing geometric registration benchmarks.

One limitation of our approach is that the differentiable voxelization layer might be a speed bottleneck for large-size point cloud inputs. This issue could be alleviated by applying down-sampling. Besides, to handle thin-structure

inputs, a higher resolution of voxel grids might be needed, which, though, will also increase the computation cost.

For future work, we will study the application of the differentiable voxelization to other 3D analysis tasks, such as object recognition [61]. Besides, it would be interesting to combine our registration loss with other sub-tasks of point cloud registration, such as keypoint detection [38] or outlier filtering of correspondences [51]. It is also worth investigating the extension of our descriptor to the learning of unsupervised non-rigid shape matching [22].

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments. Parts of this work were supported by the ERC Starting Grants No. 758800 (EXPRO-TEA), the ANR AI Chair AIGRETTE, and City University of Hong Kong (Project No. 7005729).

APPENDIX

Spectral Matching. In Sec. 3.2 - *Transformation Estimation*, we use the spectral matching technique to compute the correspondence compatibility w_i^{sm} in Eq. (4). For completeness, we describe the algorithmic details in the following.

Given the putative correspondence set \mathbf{C} , spectral matching aims to maximize the following inter-cluster score

$$\mathbf{w}^{sm} = \arg \max_{\mathbf{w}^{sm}} (\mathbf{w}^{sm})^\top \mathbf{M} \mathbf{w}^{sm}, \quad (15)$$

where $\mathbf{M} \in \mathbb{R}^{|\mathbf{C}| \times |\mathbf{C}|}$ is a correspondence compatibility matrix, and $\mathbf{w}^{sm} \in \mathbb{R}^{|\mathbf{C}|}$ is an indicator vector whose i^{th} entry denotes the association of the correspondence $c_i \in \mathbf{C}$ with the main inlier cluster. The entry $\mathbf{M}(c_i, c_j)$ measures the *consistency* between correspondences $c_i = (\mathbf{p}_i, \mathbf{q}_i)$ and $c_j = (\mathbf{p}_j, \mathbf{q}_j)$ in terms of length distortion. For $i \neq j$, $\mathbf{M}(c_i, c_j)$ is defined as follows:

$$\mathbf{M}(c_i, c_j) = \left[1 - \frac{d_{ij}^2}{\sigma_d^2}\right]_+, \quad d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|_2 - \|\mathbf{q}_i - \mathbf{q}_j\|_2, \quad (16)$$

where $[\cdot]_+ = \max(\cdot, 0)$, and $\sigma_d = 0.1$ controls the sensitivity to length distortion. For $i = j$, $\mathbf{M}(c_i, c_j) = 0$, because there is no information on an individual correspondence. The entries of \mathbf{M} defined above are non-negative and increase as the length distortions between correspondences decrease. Thus \mathbf{M} intuitively captures the compatibility between any two correspondences. The principal eigenvector of \mathbf{M} , under the constraint $\|\mathbf{w}^{sm}\|_2 = 1$, maximizes the above inter-cluster score and can be computed efficiently by the power iteration algorithm as follows:

$$\mathbf{w}_{k+1}^{sm} = \frac{\mathbf{M} \mathbf{w}_k^{sm}}{\|\mathbf{M} \mathbf{w}_k^{sm}\|_2}, \quad (17)$$

where k denotes the k^{th} iteration. We use $\mathbf{w}_0^{sm} = \mathbf{1}$. In practice, we find that the power iteration algorithm converges in 10 iterations. We assign the i^{th} entry of \mathbf{w}^{sm} to w_i^{sm} .

Qualitative Results. More qualitative registration results are shown in Fig. 9.

REFERENCES

- [1] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *IJCV*, vol. 105, no. 1, 2013.
- [2] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov, "Shape google: Geometric words and expressions for invariant shape retrieval," *ACM TOG*, vol. 30, no. 1, Feb. 2011.
- [3] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3d mesh segmentation and labeling," *ACM TOG*, vol. 29, no. 4, Jul. 2010.
- [4] A. S. Mian, M. Bennamoun, and R. A. Owens, "A novel representation and feature matching algorithm for automatic pairwise registration of range images," *IJCV*, vol. 66, no. 1, 2006.
- [5] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3d point clouds and meshes: A survey from rigid to nonrigid," *IEEE TVCG*, vol. 19, no. 7, 2012.
- [6] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, "A comprehensive performance evaluation of 3d local feature descriptors," *IJCV*, vol. 116, no. 1, 2016.
- [7] A. Zeng, S. Song, M. Niessner, M. Fisher, J. Xiao, and T. Funkhouser, "3DMatch: Learning local geometric descriptors from rgb-d reconstructions," in *CVPR*, 2017.
- [8] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra, "PCPNET learning local shape properties from raw point clouds," in *CGF*, vol. 37, no. 2, 2018.
- [9] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3D point cloud matching with smoothed densities," in *CVPR*, 2019.
- [10] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "SpinNet: Learning a general surface descriptor for 3d point cloud registration," in *CVPR*, 2021.
- [11] M. Khoury, Q.-Y. Zhou, and V. Koltun, "Learning compact geometric features," in *ICCV*, 2017.
- [12] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global context aware local features for robust 3d point matching," in *CVPR*, 2018.
- [13] 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 *CVPR*, 2020.
- [14] R. Spezialetti, S. Salti, and L. D. Stefano, "Learning an effective equivariant 3d descriptor without supervision," in *ICCV*, 2019.
- [15] Y. Wang and J. Solomon, "PRNet: self-supervised learning for partial-to-partial registration," in *NeurIPS*, 2019.
- [16] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *ICCV*, 2019.
- [17] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *CVPR*, 2015.
- [18] H. Deng, T. Birdal, and S. Ilic, "PPF-FoldNet: Unsupervised learning of rotation invariant 3d local descriptors," in *ECCV*, 2018.
- [19] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, "3D point capsule networks," in *CVPR*, 2019.
- [20] H. Yang, W. Dong, L. Carlone, and V. Koltun, "Self-supervised geometric perception," in *CVPR*, 2021.
- [21] W. Feng, J. Zhang, H. Cai, H. Xu, J. Hou, and H. Bao, "Recurrent multi-view alignment network for unsupervised surface registration," in *CVPR*, 2021.
- [22] J.-M. Roufousse, A. Sharma, and M. Ovsjanikov, "Unsupervised deep learning for structured shape matching," in *ICCV*, 2019.
- [23] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE TPAMI*, vol. 21, no. 5, 1999.
- [24] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *ECCV*, 2004.
- [25] F. Tombari, S. Salti, and L. Di Stefano, "Unique shape context for 3d data description," in *3DOR*, 2010.
- [26] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *IROS*, 2008.
- [27] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*, 2009.
- [28] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *ECCV*, 2010.
- [29] S. Salti, F. Tombari, and L. di Stefano, "SHOT: Unique signatures of histograms for surface and texture description," *CVIU*, vol. 125, 2014.
- [30] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer, "Learning local shape descriptors from part correspondences with multiview convolutional networks," *ACM TOG*, vol. 37, no. 1, 2017.
- [31] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical CNNs," *arXiv*, 2018.
- [32] Z. J. Yew and G. H. Lee, "3DFeat-Net: Weakly supervised local 3d features for point cloud registration," in *ECCV*, 2018.
- [33] F. Poiesi and D. Boscaini, "Distinctive 3d local deep descriptors," in *ICPR*, 2021.
- [34] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *NeurIPS*, vol. 30, 2017.
- [35] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *ICCV*, 2019.
- [36] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany, "PointContrast: Unsupervised pre-training for 3d point cloud understanding," in *ECCV*, 2020.
- [37] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *CVPR*, 2019.
- [38] 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 *CVPR*, 2020.
- [39] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," in *CVPR*, 2021.
- [40] 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 *ICCV*, 2019.
- [41] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *CVPR*, 2015.
- [42] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv*, 2017.
- [43] J. Du, R. Wang, and D. Cremers, "DH3D: Deep hierarchical 3d descriptors for robust large-scale 6dof relocalization," in *ECCV*, 2020.
- [44] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *CVPR*, 2018.
- [45] Y. Aoki, H. Goforth, R. Arun Srivatsan, and S. Lucey, "PointNetLK: Robust & efficient point cloud registration using pointnet," in *CVPR*, 2019.
- [46] H. Deng, T. Birdal, and S. Ilic, "3D local features for direct pairwise registration," in *CVPR*, 2019.
- [47] F. J. Lawin and P.-E. Forssén, "Registration loss learning for deep probabilistic point set registration," *arXiv*, 2020.
- [48] Z. J. Yew and G. H. Lee, "RPM-Net: Robust point matching using learned features," in *CVPR*, 2020.
- [49] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal, "Learning multiview 3d point cloud registration," in *CVPR*, 2020.
- [50] X. Huang, G. Mei, and J. Zhang, "Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences," in *CVPR*, 2020.
- [51] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *CVPR*, 2020.
- [52] Z. Yan, Z. Yi, R. Hu, N. J. Mitra, D. Cohen-Or, and H. Huang, "Consistent two-flow network for tele-registration of point clouds," *IEEE TVCG*, 2021.
- [53] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascimento, R. Chellappa, and P. Miraldo, "3DRegNet: A deep neural network for 3d point registration," in *CVPR*, 2020.
- [54] X. Bai, Z. Luo, L. Zhou, H. Chen, L. Li, Z. Hu, H. Fu, and C.-L. Tai, "PointDSC: Robust point cloud registration using deep spatial consistency," in *CVPR*, 2021.
- [55] K. M. Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua, "Learning to find good correspondences," in *CVPR*, June 2018.
- [56] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3d point clouds," in *CVPR*, 2019.
- [57] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so(3) equivariant representations with spherical cnns," in *ECCV*, September 2018.
- [58] R. Spezialetti, F. Stella, M. Marcon, L. Silva, S. Salti, and L. Di Stefano, "Learning to orient surfaces by self-supervised spherical cnns," *NeurIPS*, 2020.
- [59] J. Yang, Q. Zhang, Y. Xiao, and Z. Cao, "TOLDI: An effective and robust approach for 3d local shape description," *Pattern Recogn.*, vol. 65, 2017.

- [60] S. Liu, T. Li, W. Chen, and H. Li, "Soft Rasterizer: A differentiable renderer for image-based 3d reasoning," in *ICCV*, 2019.
- [61] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *CVPR*, 2016.
- [62] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance Normalization: The missing ingredient for fast stylization," *arXiv*, 2016.
- [63] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *ECCV*, 2016.
- [64] —, "Open3D: A modern library for 3D data processing," *arXiv*, 2018.
- [65] T. Plötz and S. Roth, "Neural nearest neighbors networks," *NeurIPS*, 2018.
- [66] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *ICCV*, 2005.
- [67] G. Golub and C. van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.
- [68] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, 1981.
- [69] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimeshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," *NeurIPS*, 2019.
- [70] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.
- [71] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *CVPR*, 2015.
- [72] 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, 2012.
- [73] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *ICRA*, 2011.
- [74] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE TPAMI*, vol. 14, no. 2, 1992.
- [75] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A globally optimal solution to 3d icp point-set registration," *IEEE TPAMI*, vol. 38, no. 11, 2015.
- [76] Z. Deng, Y. Yao, B. Deng, and J. Zhang, "A robust loss for point cloud registration," in *ICCV*, 2021.
- [77] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "DeepGMR: Learning latent gaussian mixture models for registration," in *ECCV*, 2020.
- [78] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng, "OMNet: Learning overlapping mask for partial-to-partial point cloud registration," in *ICCV*, 2021.

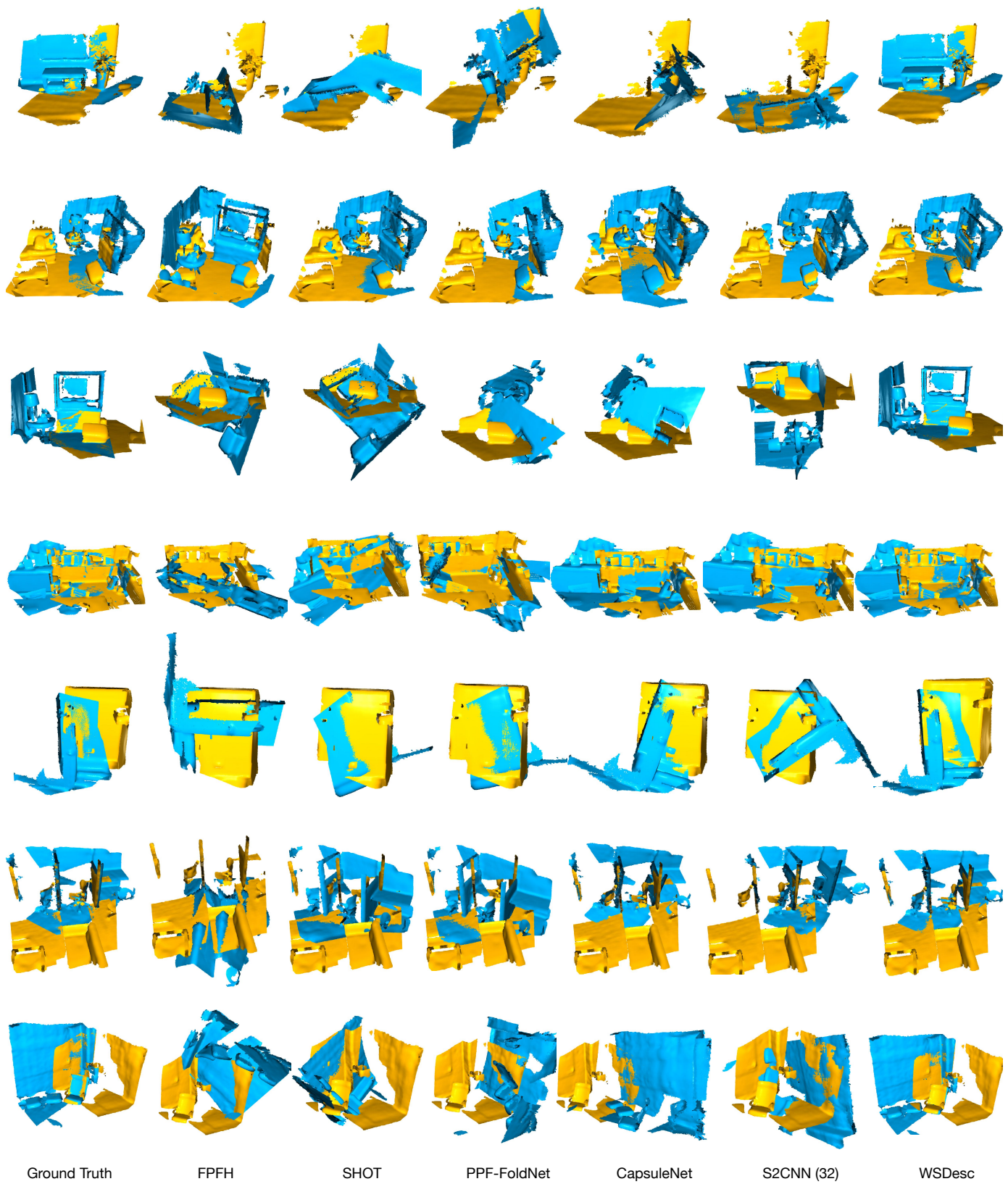


Fig. 9: More point cloud registration results by RANSAC with different descriptors.