PointDSC: Robust Point Cloud Registration using Deep Spatial Consistency

Xuyang Bai[1] Zixin Luo[1] Lei Zhou[1] Hongkai Chen[1] Lei Li[1,2] Zeyu Hu[1] Hongbo Fu[3] Chiew-Lan Tai[1]
[1]Hong Kong University of Science and Technology [2]École Polytechnique [3]City University of Hong Kong
{xbaiad,zluoag,lzhouai,hchencf,llibb,zhuam,taicl}@cse.ust.hk hongbofu@cityu.edu.hk

Abstract

Removing outlier correspondences is one of the critical steps for successful feature-based point cloud registration. Despite the increasing popularity of introducing deep learning techniques in this field, spatial consistency, which is essentially established by a Euclidean transformation between point clouds, has received almost no individual attention in existing learning frameworks. In this paper, we present PointDSC, a novel deep neural network that explicitly incorporates spatial consistency for pruning outlier correspondences. First, we propose a nonlocal feature aggregation module, weighted by both feature and spatial coherence, for feature embedding of the input correspondences. Second, we formulate a differentiable spectral matching module, supervised by pairwise spatial compatibility, to estimate the inlier confidence of each correspondence from the embedded features. With modest computation cost, our method outperforms the state-of-the-art handcrafted and learning-based outlier rejection approaches on several real-world datasets by a significant margin. We also show its wide applicability by combining PointDSC with different 3D local descriptors. [code release]

1. Introduction

The state-of-the-art feature-based point cloud registration pipelines commonly start from local feature extraction and matching, followed by an outlier rejection for robust alignment. Although 3D local features [4, 41, 18, 28, 34] have evolved rapidly, correspondences produced by feature matching are still prone to outliers, especially when the overlap of scene fragments is small. In this paper, we focus on developing a robust outlier rejection method to mitigate this issue.

Traditional outlier filtering strategies can be broadly classified into two categories, namely the individual-based and group-based [72]. The individual-based approaches, such as ratio test [42] and reciprocal check [10], identify inlier correspondences solely based on the descriptor similarity, without considering their spatial coherence. In contrast, the group-based methods usually leverage the underlying 2D or 3D scene geometry and identify inlier correspondences through the analysis of spatial consistency. Specifically, in a 2D domain, the spatial consistency only provides a weak relation between points and epipolar lines [13, 9, 78]. Instead, in a 3D domain, the spatial consistency is rigorously defined between *every* pair of points by rigid transformations, serving as one of the most important geometric properties that inlier correspondences should follow. In this paper, we focus on leveraging the spatial consistency in outlier rejection for robust 3D point cloud registration.

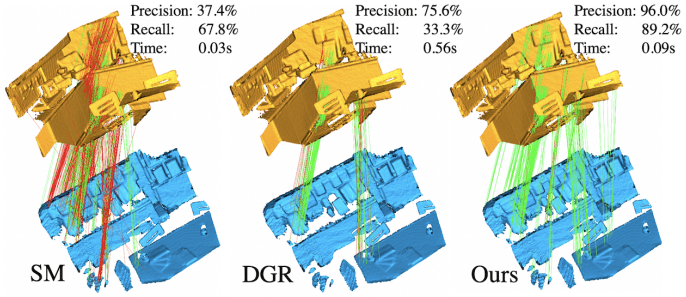


Figure 1: Taking advantage of both the superiority of traditional (e.g. SM [38]) and learning methods (e.g. DGR [16]), our approach integrates important geometric cues into deep neural networks and efficiently identifies inlier correspondences even under high outlier ratios.

Spectral matching (SM) [38] is a well-known traditional algorithm that heavily relies on 3D spatial consistency for finding inlier correspondences. It starts with constructing a compatibility graph using the length consistency, i.e., preserving the distance between point pairs under rigid transformations, then obtains an inlier set by finding the main cluster of the graph through eigen analysis. However, this algorithm has two main drawbacks. First, solely relying on length consistency is intuitive but inadequate because it suffers from the ambiguity problem [58] (Fig. 4a). Second, as explained in [73, 72], spectral matching cannot effectively handle the case of high outlier ratio (Fig. 1, left), where the main inlier clusters become less dominant and thus are difficult to be identified through spectral analysis.

Recently, learning-based 3D outlier rejection methods,

such as DGR [16] and 3DRegNet [51], formulate outlier rejection as an inlier/outlier classification problem, where the networks embed deep features from correspondence input, and predict inlier probability of each correspondence for outlier removal. For feature embedding, those methods solely rely on generic operators such as sparse convolution [17] and pointwise MLP [57] to capture the contextual information, while the essential 3D spatial relations are omitted. Additionally, during outlier pruning, the existing methods classify each correspondence only individually, again overlooking the spatial compatibility between inliers and may hinder the classification accuracy.

All the aforementioned outlier rejection methods are either hand-crafted with spatial consistency adopted, or learning-based without spatial consistency integrated. In this paper, we aim to take the best from both line of methods, and propose PointDSC, a powerful two-stage deep neural network that explicitly leverages the spatial consistency constraints during both feature embedding and outlier pruning.

Specifically, given the point coordinates of input correspondences, we first propose a spatial-consistency guided nonlocal module for geometric feature embedding, which captures the relations among different correspondences by combining the length consistency with feature similarity to obtain more representative features. Second, we formulate a differentiable spectral matching module, and feed it with not only the point coordinates, but also the embedded features to alleviate the ambiguity problem. Finally, to better handle the small overlap cases, we propose a seeding mechanism, which first identifies a set of reliable correspondences, then forms several different subsets to perform the neural spectral matching multiple times. The best rigid transformation is finally determined such that the geometric consensus is maximized. To summarize, our main contributions are threefold:

1. We propose a spatial-consistency guided nonlocal (SC-Nonlocal) module for feature embedding, which explicitly leverages the spatial consistency to weigh the feature correlation and guide the neighborhood search.
2. We propose a differentiable neural spectral matching (NSM) module based on traditional SM for outlier removal, which goes beyond the simple length consistency metric through deep geometric features.
3. Besides showing the superior performance over the state-of-the-arts, our model also demonstrates strong generalization ability from indoor to outdoor scenarios, and wide applicability with different descriptors.

2. Related Work

Point cloud registration. Traditional point cloud registration algorithms (e.g., [8, 1, 50, 33, 46, 46]) have been comprehensively reviewed in [56]. Recently, learning-based algorithms have been proposed to replace the individual components in the classical registration pipeline, including keypoint detection [4, 40, 34] and feature description [21, 22, 23, 55, 4, 18, 28, 32, 2]. Besides, end-to-end registration networks [3, 67, 68, 76] have been proposed. However, their robustness and applicability in complex scenes cannot always meet expectation, as observed in [16], due to highly outlier-contaminated matches.

Traditional outlier rejection. RANSAC [24] and its variants [19, 5, 37, 39] are still the most popular outlier rejection methods. However, their major drawbacks are slow convergence and low accuracy in cases with large outlier ratio. Such problems become more obvious in 3D point cloud registration since the description ability of 3D descriptors is generally weaker than those in 2D domain [42, 6, 44, 43, 45] due to the irregular density and the lack of useful texture [11]. Thus, geometric consistency, such as length constraint under rigid transformation, becomes important and is commonly utilized by traditional outlier rejection algorithms and analyzed through spectral techniques [38, 20], voting schemes [26, 74, 61], maximum clique [54, 12, 64], random walk [14], belief propagation [81] or game theory [59]. Meanwhile, some algorithms based on BnB [11] or SDP [37] are accurate but usually have high time complexity. Besides, FGR [82] and TEASER [70, 71] are tolerant to outliers from robust cost functions such as Geman-McClure function. A comprehensive review of traditional 3D outlier rejection methods can be found in [73, 72].

Learning-based outlier rejection. Learning-based outlier rejection methods are first introduced in the 2D image matching task [48, 78, 79, 65], where outlier rejection is formulated as an inlier/outlier classification problem. The recent 3D outlier rejection methods DGR [16] and 3DRegNet [51] follow this idea, and use operators such as sparse convolution [17] and pointwise MLP [57] to classify the putative correspondences. However, they both ignore the rigid property of 3D Euclidean transformations that has been widely shown to be powerful side information. In contrast, our network explicitly incorporates the spatial consistency between inlier correspondences, constrained by rigid transformations, for pruning the outlier correspondences.

3. Methodology

In this work, we consider two sets of sparse keypoints $\mathbf{X} \in \mathbb{R}^{|\mathbf{X}|\times 3}$ and $\mathbf{Y} \in \mathbb{R}^{|\mathbf{Y}|\times 3}$ from a pair of partially overlapping 3D point clouds, with each keypoint having an associated local descriptor. The input putative correspondence set C can be generated by nearest neighbor search using the local descriptors. Each correspondence $c_i \in C$ is denoted as $c_i = (\boldsymbol{x_i}, \boldsymbol{y_i}) \in \mathbb{R}^6$, where $\boldsymbol{x_i} \in \mathbf{X}, \boldsymbol{y_i} \in \mathbf{Y}$ are the coordinates of a pair of 3D keypoints from the two sets. Our objective is to find an inlier/outlier label for c_i, be-

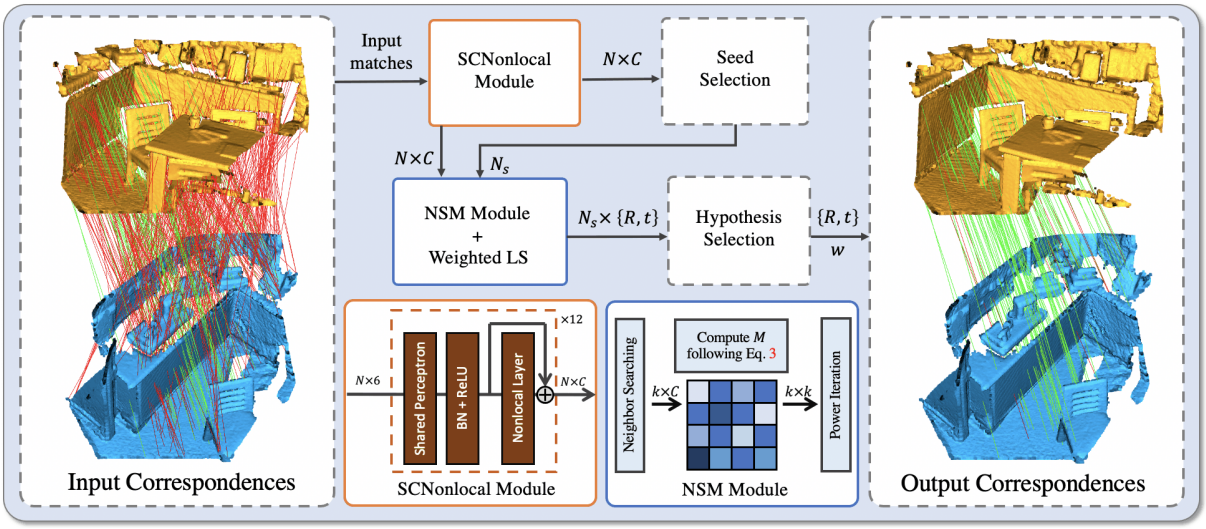


Figure 2: Architecture of the proposed network PointDSC. It takes as input the coordinates of putative correspondences, and outputs a rigid transformation and an inlier/outlier label for each correspondence. The Spatial Consistency Nonlocal (SC-Nonlocal) module and the Neural Spectral Matching (NSM) module are two key components of our network, and perform feature embedding and outlier pruning, respectively. The green lines and red lines are inliers and outliers, respectively. LS represents least-squares fitting.

ing $w_i = 1$ and 0, respectively, and recover an optimal 3D rigid transformation $\hat{\mathbf{R}}, \hat{\mathbf{t}}$ between the two point sets. The pipeline of our network PointDSC is shown in Fig. 2 and can be summarized as follows:

1. We embed the input correspondences into high dimensional geometric features using the SCNonlocal module (Sec. 3.2).
2. We estimate the initial confidence v_i of each correspondence c_i to select a limited number of highly confident and well-distributed *seeds* (Sec. 3.3).
3. For each *seed*, we search for its k nearest neighbors in the feature space and perform neural spectral matching (NSM) to obtain its confidence of being an inlier. The confidence values are used to weigh the least-squares fitting for computing a rigid transformation for each seed (Sec. 3.4).
4. The best transformation matrix is selected from all the hypotheses as the one that maximizes the number of inlier correspondences (Sec. 3.5).

3.1. PointDSC vs. RANSAC

Here, we clarify the difference between PointDSC and RANSAC to help understand the insights behind our algorithm. Despite not being designed for improving classic RANSAC, our PointDSC shares a *hypothesize-and-verify* pipeline similar to RANSAC. In the sampling step, instead of randomly sampling minimal subsets iteratively, we utilize the learned embedding space to retrieve a pool of larger correspondence subsets in one shot (Sec. 3.2 and Sec. 3.3). The correspondences in such subsets have higher probabilities of being inliers thanks to the highly confident seeds and the discriminative embedding space. In the model fitting step, our neural spectral matching module (Sec. 3.4) effectively prunes the potential outliers in the retrieved subsets, producing a correct model even when starting from a not-all-inlier sample. In this way, PointDSC can tolerate large outlier ratios and produce highly precise registration results, without needing exhaustive iterations.

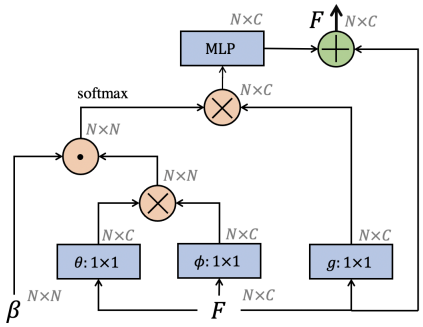


Figure 3: The spatial-consistency guided nonlocal layer. $\boldsymbol{\beta}$ represents the spatial consistency matrix calculated using Eq. 2 and $\boldsymbol{F}$ is the feature from the previous layer.

3.2. Geometric Feature Embedding

The first module of our network is the SCNonlocal module, which receives the correspondences C as input and produces a geometric feature for each correspondence. Previous networks [16, 51] learn the feature embedding through generic operators, ignoring the unique properties of 3D rigid transformations. Instead, our SCNonlocal module explicitly utilizes the spatial consistency between inlier correspondences to learn a discriminative embedding space, where inlier correspondences are close to each other.

As illustrated in Fig. 2, our SCNonlocal module has 12 blocks, each of which consists of a shared Perceptron layer, a BatchNorm layer with ReLU, and the proposed nonlocal layer. Fig. 3 illustrates this new nonlocal layer. Let $\boldsymbol{f_i} \in \boldsymbol{F}$ be the intermediate feature representation for correspondence c_i. The design of our nonlocal layer for updating the features draws inspiration from the well-known nonlocal network [66], which captures the long-range dependencies using nonlocal operators. Our contribution is to introduce a novel spatial consistency term to complement the feature similarity in nonlocal operators. Specifically, we update the features using the following equation:

$$\boldsymbol{f_i} = \boldsymbol{f_i} + \text{MLP}(\sum_j^{|C|} \text{softmax}_j(\boldsymbol{\alpha}\boldsymbol{\beta}) g(\boldsymbol{f_j})) \,, \quad (1)$$

where g is a linear projection function. The feature similarity term $\boldsymbol{\alpha}$ is defined as the embedded dot-product similarity [66]. The spatial consistency term $\boldsymbol{\beta}$ is defined based on the length constraint of 3D rigid transformations, as illustrated in Fig. 4a (c_1 and c_2).

Specifically, we compute $\boldsymbol{\beta}$ by measuring the length difference between the line segments of point pairs in $\mathbf{X}$ and its corresponding segments in $\mathbf{Y}$:

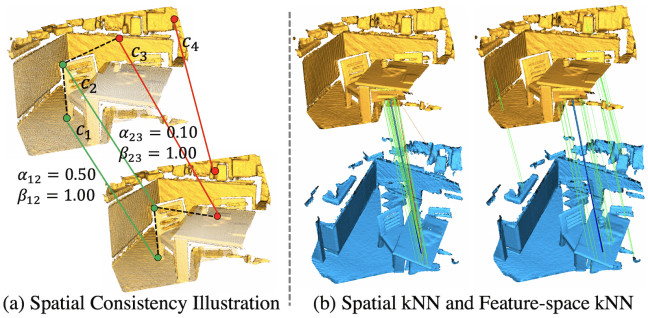
$$\beta_{ij} = [1 - \frac{d_{ij}^2}{\sigma_d^2}]_+, \;\; d_{ij} = \big| \, \|\boldsymbol{x_i} - \boldsymbol{x_j}\| - \|\boldsymbol{y_i} - \boldsymbol{y_j}\| \, \big|, \quad (2)$$

where $[\cdot]_+$ is the $\max(\cdot, 0)$ operation to ensure a non-negative value of β_{ij}, and σ_d is a distance parameter (see Sec. 4) to control the sensitivity to the length difference. Correspondence pairs having the length difference larger than σ_d are considered to be incompatible and get zero for $\boldsymbol{\beta}$. In contrast, β_{ij} gives a large value only if the two correspondences c_i and c_j are spatially compatible, serving as a reliable regulator to the feature similarity term.

Note that other forms of spatial consistency can also be easily incorporated here. However, taking an angle-based spatial consistency constraint as an example, the normals of input keypoints might not always be available for outlier rejection and the normal estimation task is challenging on its own especially for LiDAR point clouds [80]. Our SCNonlocal module produces for each correspondence c_i a feature representation $\boldsymbol{f_i}$, which will be used in both seed selection and neural spectral matching module.

3.3. Seed Selection

As mentioned before, the traditional spectral matching technique has difficulties in finding a dominant inlier cluster in low overlapping cases, where it would fail to provide a clear separation between inliers and outliers [75]. In such cases, directly using the output from spectral matching in weighted least-squares fitting [8] for transformation estimation may lead to a sub-optimal solution since there are still many outliers not being explicitly rejected. To address this issue, inspired by [13], we design a seeding mechanism to apply neural spectral matching locally. We first find reliable and well-distributed correspondences as seeds, and around them search for consistent correspondences in the feature space. Then each subset is expected to have a higher inlier ratio than the input correspondence set, and is thus easier for neural spectral matching to find a correct cluster.



(a) Spatial Consistency Illustration (b) Spatial kNN and Feature-space kNN

Figure 4: (a) Inlier correspondence pairs (c_1, c_2) always satisfy the length consistency, while outliers (e.g. c_4) are usually not spatially consistent with either inliers (c_1, c_2) or other outliers (e.g. c_3). However, there exist ambiguity when inliers (c_2) and outliers (c_3) happen to satisfy the length consistency. The feature similarity term $\boldsymbol{\alpha}$ provides the possibility to alleviate the ambiguity issue. (b) The correspondence subsets of a seed (blue line) found by spatial kNN (Left) and feature-space kNN (Right).

To select the seeds, we first adopt an MLP to estimate the initial confidence v_i of each correspondence using the feature $\boldsymbol{f_i}$ learned by the SCNonlocal module, and then apply Non-Maximum Suppression [42] over the confidence to find the well-distributed seeds. The selected seeds will be used to form multiple correspondence subsets for the neural spectral matching.

3.4. Neural Spectral Matching

In this step, we leverage the learned feature space to augment each seed with a subset of consistent correspondences by performing k-nearest neighbor searching in the feature space. We then adopt the proposed neural spectral matching (NSM) over each subset to estimate a transformation as one hypothesis. Feature-space kNN has several advantages over spatial kNN, as illustrated in Fig. 4b. First, the neighbors found in the feature space are more likely to follow a similar transformation as the seeds, thanks to the SCNonlocal module. Second, the neighbors chosen in the feature space can be located far apart in the 3D space, leading to more robust transformation estimation results.

Given the correspondence subset $C' \subseteq C$ $(|C'| = k)$ of each seed constructed by kNN search, we apply NSM to estimate the inlier probability, which is subsequently used in the weighted least-squares fitting [8] for transformation estimation. Following [38], we first construct a matrix $\mathbf{M}$ representing a compatibility graph associated with C', as illustrated in Fig. 5. Instead of solely relying on the length

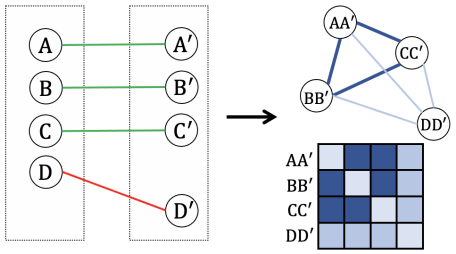


Figure 5: Constructing the compatibility graph and associated matrix (Right) from the input correspondences (Left). We set the matrix diagonal to zero following [38]. The weight of each graph edge represents the pairwise compatibility between two associated correspondences.

consistency as [38], we further incorporate the geometric feature similarity to tackle the ambiguity problem as illustrated in Fig. 4a. Each entry M_{ij} measures the compatibility between correspondence c_i and c_j from C', which is defined as

$$M_{ij} = \beta_{ij} * \gamma_{ij}, \tag{3}$$

$$\gamma_{ij} = [1 - \frac{1}{\sigma_f^2} \left\| \bar{\boldsymbol{f}}_{\boldsymbol{i}} - \bar{\boldsymbol{f}}_{\boldsymbol{j}} \right\|^2]_+ \tag{4}$$

where β_{ij} is the same as in Eq. 2, $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$ and $\bar{\boldsymbol{f}}_{\boldsymbol{j}}$ are the L2-normalized feature vectors, and σ_f is a parameter to control sensitivity to feature difference (see Sec. 4).

The elements of $\mathbf{M}$ defined above are always non-negative and increase with the compatibility between correspondences. Following [38], we consider the leading eigenvector of matrix $\mathbf{M}$ as the *association* of each correspondence with a main cluster. Since this main cluster is statistically formed by the inlier correspondences, it is natural to interpret this *association* as the inlier probability. The higher the association to the main cluster, the higher the probability of a correspondence being an inlier. The leading eigenvector $\boldsymbol{e} \in \mathbb{R}^k$ can be efficiently computed by the power iteration algorithm [47]. We regard $\boldsymbol{e}$ as the inlier probability, since only the relative value of $\boldsymbol{e}$ matters. Finally we use the probability $\boldsymbol{e}$ as the weight to estimate the transformation through least-squares fitting,

$$\mathbf{R}', \mathbf{t}' = \arg\min_{\mathbf{R},\mathbf{t}} \sum_i^{|C'|} \boldsymbol{e}_{\boldsymbol{i}} \left\| \mathbf{R}\boldsymbol{x}_{\boldsymbol{i}} + \mathbf{t} - \boldsymbol{y}_{\boldsymbol{i}} \right\|^2. \tag{5}$$

Eq. 5 can be solved in closed form by SVD [8]. For the sake of completeness, we provide its derivation in the supplementary 7.6. By performing such steps for each seed in parallel, the network produces a set of transformations $\{\mathbf{R}', \mathbf{t}'\}$ for hypothesis selection.

3.5. Hypothesis Selection

The final stage of PointDSC involves selecting the best hypothesis among the transformations produced by the NSM module. The criterion for selecting the best transformation is based on the number of correspondences satisfied by each transformation,

$$\hat{\mathbf{R}}, \hat{\mathbf{t}} = \arg\max_{\mathbf{R}',\mathbf{t}'} \sum_i^{|C|} [\![||\mathbf{R}'\boldsymbol{x}_{\boldsymbol{i}} + \mathbf{t}' - \boldsymbol{y}_{\boldsymbol{i}}|| < \tau]\!], \tag{6}$$

where $[\![\cdot]\!]$ is the Iverson bracket and τ denotes an inlier threshold. The final inlier/outlier labels $\boldsymbol{w} \in \mathbb{R}^{|C|}$ are given by $w_i = [\![||\hat{\mathbf{R}}\boldsymbol{x}_{\boldsymbol{i}} + \hat{\mathbf{t}} - \boldsymbol{y}_{\boldsymbol{i}}|| < \tau]\!]$. We then recompute the transformation matrix using all the surviving inliers in a least-squares manner, which is a common practice [19, 5].

3.6. Loss Formulation

Considering the compatibility graph illustrated in Fig. 5, previous works [16, 51] mainly adopt node-wise losses, which supervise each correspondence individually. In our work, we further design an edge-wise loss to supervise the pairwise relations between the correspondences.

Node-wise supervision. We denote $\boldsymbol{w}^* \in \mathbb{R}^{|C|}$ as the ground-truth inlier/outlier labels constructed by

$$w_i^* = [\![||\mathbf{R}^*\boldsymbol{x}_{\boldsymbol{i}} + \mathbf{t}^* - \boldsymbol{y}_{\boldsymbol{i}}|| < \tau]\!], \tag{7}$$

where $\mathbf{R}^*$ and $\mathbf{t}^*$ are the ground-truth rotation and translation matries, respectively. Similar to [16, 51], we first adopt the binary cross entropy loss as the node-wise supervision for learning the initial confidence by

$$L_{class} = \text{BCE}(\boldsymbol{v}, \boldsymbol{w}^*), \tag{8}$$

where $\boldsymbol{v}$ is the initial confidence predicted (Sec. 3.3).

Edge-wise supervision We further propose the spectral matching loss as our edge-wise supervision, formulated as

$$L_{sm} = \frac{1}{|C|^2} \sum_{ij} (\gamma_{ij} - \gamma_{ij}^*)^2, \tag{9}$$

where $\gamma_{ij}^* = [\![c_i, c_j \text{ are both inliers}]\!]$ is the ground-truth compatibility value and γ_{ij} is the estimated compatibility value based on the feature similarity defined in Eq. 4. This loss supervises the relationship between each pair of correspondences, serving as a complement to the node-wise supervision. Our experiments (Sec. 5.4) show that the proposed L_{sm} remarkably improves the performance.

The final loss is a weighted sum of the two losses,

$$L_{total} = L_{sm} + \lambda L_{class}, \tag{10}$$

where λ is a hyper-parameter to balance the two losses.

4. Implementation Details

Training. We implement our network in PyTorch [52]. Since each pair of point clouds may have different numbers of correspondences, we randomly sample 1,000 correspondences from each pair to build the batched input during training and set the batch size to 16 point cloud pairs. For NSM, we choose the neighborhood size to be $k = 40$. (The choice of k is studied in the supplementary 7.5). We make σ_f learned by the network, and set σ_d as 10cm for indoor scenes and 60cm for outdoor scenes, since σ_d has a clear

physical meaning [38]. The hyper-parameter λ is set to 3. We optimize the network using the ADAM optimizer with an initial learning rate of 0.0001 and an exponentially decayed factor of 0.99, and train the network for 100 epochs. All the experiments are conducted on a single RTX2080 Ti graphics card.

Testing. During testing, we use a full correspondence set as input. We adopt Non-Maximum Suppression (NMS) to ensure spatial uniformity of the selected seeds, and set the radius for NMS to be the same value as the inlier threshold τ. To avoid having excessive seeds returned by NMS and make the computation cost manageable, we keep at most 10% of the input correspondences as seeds. To improve the precision of the final transformation matrix, we further adopt a simple yet effective post-refinement stage analogous to iterative re-weighted least-squares [31, 7]. The detailed algorithm can be found in the supplementary 7.1.

5. Experiments

The following sections are organized as follows. First, we evaluate our method (PointDSC) in pairwise registration tasks on 3DMatch dataset [77] (indoor settings) with different descriptors, including the learned ones and hand-crafted ones, in Sec. 5.1. Next, we study the generalization ability of PointDSC on KITTI dataset [25] (outdoor settings) using the model trained on 3DMatch in Sec. 5.2. We further evaluate PointDSC in multiway registration tasks on augmented ICL-NUIM [15] dataset in Sec. 5.3. Finally, we conduct ablation studies to demonstrate the importance of each proposed component in PointDSC.

5.1. Pairwise Registration

We follow the same evaluation protocols in 3DMatch to prepare training and testing data, where the test set contains eight scenes with $1,623$ partially overlapped point cloud fragments and their corresponding transformation matrices. We first voxel-downsample the point clouds with a 5cm voxel size, then extract different feature descriptors to build the initial correspondence set as input. The inlier threshold τ is set to 10cm.

Evaluation metrics. Following DGR [16], we use three evaluation metrics, namely (1) *Registration Recall (RR)*, the percentage of successful alignment whose rotation error and translation error are below some thresholds, (2) *Rotation Error (RE)*, and (3) *Translation Error (TE)*. *RE* and *TE* are defined as

$$\mathrm{RE}(\hat{\mathbf{R}}) = \arccos \frac{\mathrm{Tr}(\hat{\mathbf{R}}^{\mathbf{T}}\mathbf{R}^{*}) - 1}{2}, \quad \mathrm{TE}(\hat{\mathbf{t}}) = \left\|\hat{\mathbf{t}} - \mathbf{t}^{*}\right\|_2, \tag{11}$$

where $\mathbf{R}^*$ and $\mathbf{t}^*$ denote the ground-truth rotation and translation, respectively, and the average *RE* and *TE* are computed only on successfully registered pairs. Besides, we also report the intermediate outlier rejection results, including *Inlier Precision (IP)*=$\frac{\#kept\ inliers}{\#kept\ matches}$ and *Inlier Recall (IR)*=$\frac{\#kept\ inliers}{\#inliers}$, which are particularly introduced to evaluate the outlier rejection module. For *RR*, one registration result is considered successful if the *TE* is less than 30cm and the *RE* is less than 15°. For a fair comparison, we report two sets of results by combining different outlier rejection algorithms with the learned descriptor FCGF [18] and hand-crafted descriptor FPFH [60], respectively.

Baseline methods. We first select four representative traditional methods: FGR [82], SM [38], RANSAC [24], and GC-RANSAC [5], as well as the state-of-the-art geometry-based method TEASER [71]. For learning-based methods, we choose 3DRegNet [51] and DGR [16] as the baselines, since they also focus on the outlier rejection step for point cloud registration. We also report the results of DGR without RANSAC (i.e., without the so-called safeguard mechanism) to better compare the weighted least-squares solutions. We carefully tune each method to achieve the best results on the evaluation dataset for a fair comparison. More details can be found in the supplementary 7.2.

Comparisons with the state-of-the-arts. We compare our PointDSC with the baseline methods on 3DMatch. As shown in Table 1, all the evaluation metrics are reported in two settings: input putative correspondences constructed by FCGF (left columns) and FPFH (right columns). PointDSC achieves the best *Registration Recall* as well as the lowest average *TE* and *RE* in both settings. More statistics can be found in the supplementary 7.4.

Combination with FCGF descriptor. Compared with the learning-based baselines, PointDSC surpasses the second best method, i.e., DGR, by more than **9%** in terms of *F1 score*, indicating the effectiveness of our outlier rejection algorithm. Besides, although DGR is only slightly worse than PointDSC in *Registration Recall*, it is noteworthy that more than 35% (608/1623) registration pairs are marked as failure and solved by RANSAC (safeguard mechanism). If no safeguard mechanism is applied, DGR only achieves 86.5% *Registration Recall*.

Different from the conclusion in [16], our experiments indicate that RANSAC still shows competitive results when combined with a powerful descriptor FCGF. Nevertheless, our method is about **60 times** faster than RANSAC-100k while achieving even higher *Registration Recall*. We also report the performance of RANSAC with the proposed post-refinement step to clearly demonstrate the superiority of our outlier rejection module. SM and TEASER achieve slightly better *Inlier Precision* than PointDSC, however, they have much lower *Inlier Recall* (38.36% and 68.08% vs. 86.54% (Ours)). We thus conclude that PointDSC achieves a better trade-off between precision and recall.

Combination with FPFH descriptor. We further evaluate all the outlier rejection methods equipped with the traditional descriptor, FPFH. Note that for testing learnable

	FCGF (learned descriptor)							FPFH (traditional descriptor)						
	RR(%↑)	RE(°↓)	TE(cm↓)	IP(%↑)	IR(%↑)	F1(%↑)	Time(s)	RR(%↑)	RE(°↓)	TE(cm↓)	IP(%↑)	IR(%↑)	F1(%↑)	Time(s)
FGR [82]	78.56	2.82	8.36	-	-	-	0.76	40.67	3.99	9.83	-	-	-	0.28
SM [38]	86.57	2.29	7.07	81.44	38.36	48.21	0.03	55.88	2.94	8.15	47.96	70.69	50.70	0.03
TEASER [71]	85.77	2.73	8.66	**82.43**	68.08	73.96	0.11	75.48	2.48	7.31	**73.01**	62.63	66.93	0.03
GC-RANSAC-100k [5]	92.05	2.33	7.11	64.46	**93.39**	75.69	0.47	67.65	2.33	6.87	48.55	69.38	56.78	0.62
RANSAC-1k [24]	86.57	3.16	9.67	76.86	77.45	76.62	0.08	40.05	5.16	13.65	51.52	34.31	39.23	0.08
RANSAC-10k	90.70	2.69	8.25	78.54	83.72	80.76	0.58	60.63	4.35	11.79	62.43	54.12	57.07	0.55
RANSAC-100k	91.50	2.49	7.54	78.38	85.30	81.43	5.50	73.57	3.55	10.04	68.18	67.40	67.47	5.24
RANSAC-100k refine	92.30	2.17	6.76	78.38	85.30	81.43	5.51	77.20	2.62	7.42	68.18	67.40	67.47	5.25
3DRegNet [51]	77.76	2.74	8.13	67.34	56.28	58.33	0.05	26.31	3.75	9.60	28.21	8.90	11.63	0.05
DGR w/o s.g. [16]	86.50	2.33	7.36	67.47	78.94	72.76	0.56	27.04	2.61	7.76	28.80	12.42	17.35	0.56
DGR [16]	91.30	2.40	7.48	67.47	78.94	72.76	1.36	69.13	3.78	10.80	28.80	12.42	17.35	2.49
PointDSC	**93.28**	**2.06**	**6.55**	79.10	86.54	**82.35**	0.09	**78.50**	**2.07**	**6.57**	68.57	**71.61**	**69.85**	0.09

Table 1: Registration results on 3DMatch. *RANSAC-100k refine* represents RANSAC with 100k iterations, followed by the proposed post-refinement step. *DGR w/o s.g.* represents DGR [16] without the safeguard mechanism (RANSAC). The *Time* columns report the average time cost during testing, excluding the construction of initial input correspondences.

	RR(↑)	RE(↓)	TE(↓)	F1(↑)	Time
SM [38]	79.64	0.47	12.15	56.37	0.18
RANSAC-1k [24]	11.89	2.51	38.23	14.13	0.20
RANSAC-10k	48.65	1.90	37.17	42.35	1.23
RANSAC-100k	89.37	1.22	25.88	73.13	13.7
DGR [16]	73.69	1.67	34.74	4.51	0.86
PointDSC	90.27	**0.35**	**7.83**	70.89	0.31
DGR re-trained	77.12	1.64	33.10	27.96	0.86
PointDSC re-trained	**98.20**	**0.35**	8.13	**85.54**	0.31

Table 2: Registration results on KITTI under FPFH setting.

	Living1	Living2	Office1	Office2	AVG
ElasticFusion [69]	66.61	24.33	**13.04**	35.02	34.75
InfiniTAM [35]	46.07	73.64	113.8	105.2	84.68
BAD-SLAM[63]	fail	40.41	18.53	26.34	-
Multiway + FGR [82]	78.97	24.91	14.96	21.05	34.98
Multiway + RANSAC [24]	110.9	19.33	14.42	17.31	40.49
Multiway + DGR [16]	21.06	21.88	15.76	11.56	17.57
Multiway + PointDSC	**20.25**	**15.58**	13.56	**11.30**	**15.18**

Table 3: ATE(cm) on Augmented ICL-NUIM. The last column is the average ATE over all scenes. Since BAD-SLAM fails on one scene, we do not report its average ATE.

outlier rejection methods including PointDSC, we directly re-use the model trained with the FCGF descriptor without fine-tuning, since it is expected that the outlier rejection networks are seamlessly compatible with different feature descriptors. As shown in Table 1, the superiority of PointDSC becomes more obvious when evaluated with the FPFH, where PointDSC achieves 78.5% in *Registration Recall* and remarkably surpasses the competitors. RANSAC-1k and RANSAC-10k perform significantly worse since the outlier ratios are much higher when using FPFH to build the input correspondences. RANSAC-100k with the post-refinement step still achieves the second best performance at the cost of the high computation time. In summary, all the other methods suffer from larger performance degradation than PointDSC when equipped with a weaker descriptor, strongly demonstrating the robustness of PointDSC to the input correspondences generated by different feature descriptors.

5.2. Generalization to Outdoor Scenes

In order to evaluate the generalization of PointDSC to new datasets and unseen domains, we evaluate on a LiDAR outdoor dataset, namely the KITTI odometry dataset, using the model trained on 3DMatch. We follow the same data splitting strategy in [18, 16] for a fair comparison. We use 30cm voxel size and set the inlier threshold τ to 60cm. The evaluation metrics are the same as those used in the indoor setting with a 60cm *TE* threshold and a 5° *RE* threshold.

Comparisons with the state-of-the-arts. We choose SM, DGR, and RANSAC as the baseline methods, and combine them with the FPFH descriptor. We choose FPFH because the results with FCGF are more or less saturated. (The results with FCGF can be found in the supplementary 7.5.)

We report two sets of results for DGR and PointDSC obtained when trained from scratch (labelled "re-trained") and pre-trained on 3DMatch (no extra label). As shown in Table 2, PointDSC trained on 3DMatch still gives competitive results, demonstrating its strong generalization ability on the unseen dataset. When re-trained from scratch, PointDSC can be further improved and outperform the baseline approaches by a significant margin.

5.3. Multiway Registration

For evaluating multiway registration, we use Augmented ICL-NUIM dataset [15], which augments each synthetic scene [29] with a realistic noise model. To test the generalization ability, we again use the models trained on 3DMatch without fine-tuning. Following [16], we first perform pairwise registration using PointDSC with FPFH descriptor to obtain the initial poses, then optimize the poses using pose graph optimization [36] implemented in Open3D [83]. We report the results of baseline methods presented in [16]. The *Absolute Trajectory Error (ATE)* is reported as the evaluation metric. As shown in Table 3, our method achieves the lowest average *ATE* over three of the four tested scene types.

5.4. Ablation Studies

Ablation on feature encoder. To study the effectiveness of the proposed SCNonlocal module, we conduct extensive ablation experiments on 3DMatch. Specifically, we compare (1) **PointCN** (3D version of [48], which is the feature extraction module adopted by 3DRegNet [51]); (2) **Nonlocal** (the SCNonlocal module without the spatial term, i.e., the same operator as in [66]); and (3) **SCNonlocal** (the proposed operator). All the above methods are combined either

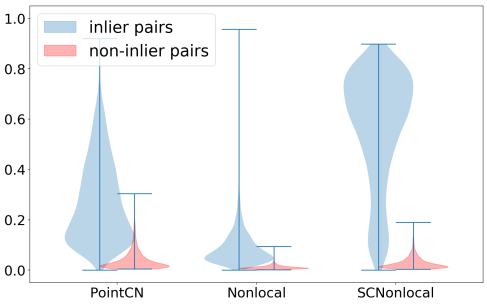


Figure 6: The distribution of feature similarity of inlier pairs and non-inlier pairs (i.e. at least one outlier in the pair).

		RR(↑)	IP(↑)	IR(↑)	F1(↑)	Time
PointCN	**+ classifier**	78.19	58.05	39.59	42.65	0.04
Nonlocal	**+ classifier**	83.30	65.49	67.13	64.28	0.07
SCNonlocal	**+ classifier**	88.17	74.74	77.86	75.04	0.07
PointCN	**+ NSM**	92.48	78.48	82.10	79.98	0.06
Nonlocal	**+ NSM**	92.54	78.68	83.13	80.58	0.09
SCNonlocal	**+ NSM**	**93.28**	**79.10**	**86.54**	**82.35**	0.09

Table 4: Ablation experiments of SCNonlocal module. Rows 1-3 and Rows 4-6 show the registration results of different feature extractors combined with a classification layer and the neural spectral matching module, respectively.

with a classification layer [16, 51] or a proposed NSM layer, resulting in six combinations in total. Other training or testing settings remain unchanged for a fair comparison.

As shown in Table 4, the proposed SCNonlocal module consistently improves the registration results across all the settings and metrics. The spatial term plays a critical role in the SCNonlocal module, without which the Nonlocal module performs drastically worse. Furthermore, we compute the feature similarity defined in Eq. 4 between each pair of correspondences and plot the distribution in Fig. 6. With the SCNonlocal module, the similarity of the inlier pairs is concentrated near 0.8 and is generally much larger than that of the non-inlier pairs. This implies that inliers are closer to each other in the embedding space. In contrast, for the baseline methods, the inliers are less concentrated, i.e., the average similarity between inliers is low.

Ablation on spectral matching. We further conduct ablation experiments to demonstrate the importance of NSM module. As shown in Table 5, the comparison between Rows 1 and 2 shows that augmenting the traditional SM with neural feature consistency notably improves the result. For **+seeding**, we adopt the neural spectral matching over multiple correspondence subsets found by the feature-space kNN search from highly confident *seeds*, and determine the best transformation that maximizes the geometric consensus. This significantly boosts the performance because it is easier to find the inlier clusters for the consistent correspondence subsets.

5.5. Qualitative Results

As shown in Fig. 7, PointDSC is robust to extremely high outlier ratios. Please refer to the supplementary 7.7 for more qualitative results.

	RR(↑)	RE(↓)	TE(↓)	F1(↑)	Time
Traditional SM	86.57	2.29	7.07	48.21	0.03
+ neural	88.43	2.21	6.91	48.88	0.06
+ seeding	92.91	2.15	6.72	82.35	0.08
+ refine	**93.28**	**2.06**	**6.55**	**82.35**	0.09
w/o L_{sm}	92.61	2.07	6.75	81.58	0.09

Table 5: Ablation experiments of NSM module. Note that every row with '+' represents the previous row equipped with the new component. **+refine** is our full model. The last row is the full model trained without L_{sm}.

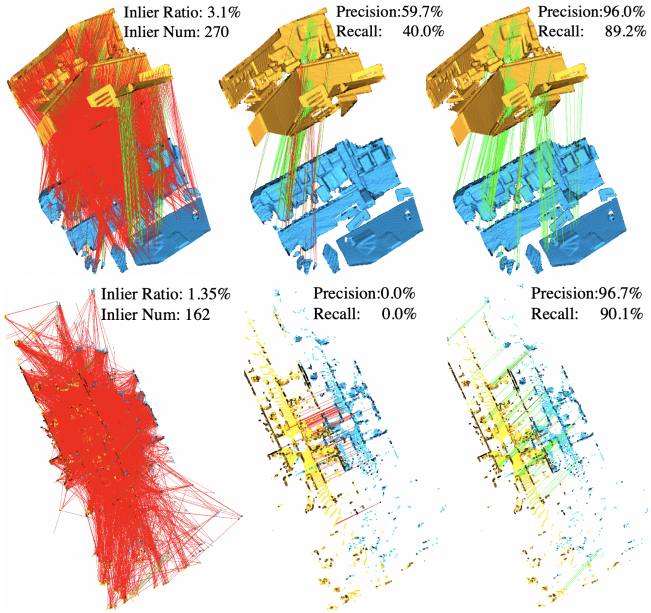


Figure 7: Visualization of outlier rejection results on examples with high outlier ratios from 3DMatch (first row) and KITTI (second row). From left to right: input correspondences, results of RANSAC-100k, and results of PointDSC.

6. Conclusion

We have designed a novel 3D outlier rejection network that explicitly incorporates *spatial consistency* established by Euclidean transformations. We have proposed a spatial-consistency guided nonlocal module (SCNonlocal) and a neural spectral matching module (NSM) for feature embedding and outlier pruning, respectively. We further proposed a seeding mechanism to adopt the NSM module multiple times to boost the robustness under high outlier ratios. The extensive experiments on diverse datasets showed that our method brings remarkable improvement over the state-of-the-arts. Our method can also generalize to unseen domains and cooperate with different local descriptors seamlessly.

Acknowledgements. This work is supported by Hong Kong RGC GRF 16206819, 16203518 and Centre for Applied Computing and Interactive Media (ACIM) of School of Creative Media, City University of Hong Kong.

References

[1] Dror Aiger, Niloy J Mitra, and Daniel Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *SIGGRAPH*. 2008. 2

[2] Sheng Ao, Qingyong Hu, Bo Yang, Andrew Markham, and Yulan Guo. SpinNet: Learning a general surface descriptor for 3d point cloud registration. *arXiv*, 2020. 2

[3] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. PointNetLK: Robust & efficient point cloud registration using pointnet. In *CVPR*, 2019. 2

[4] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai. D3Feat: Joint learning of dense detection and description of 3d local features. In *CVPR*, 2020. 1, 2, 3

[5] Daniel Barath and Jiří Matas. Graph-cut ransac. In *CVPR*, 2018. 2, 5, 6, 7, 1

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *ECCV*, 2006. 2

[7] Per Bergström and Ove Edlund. Robust registration of point sets using iteratively reweighted least squares. *Computational Optimization and Applications*, 2014. 6, 1

[8] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*. International Society for Optics and Photonics, 1992. 2, 4, 5

[9] JiaWang Bian, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan-Dat Nguyen, and Ming-Ming Cheng. GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence. In *CVPR*, 2017. 1

[10] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 1

[11] Álvaro Parra Bustos and Tat-Jun Chin. Guaranteed outlier removal for point cloud registration with correspondences. *PAMI*, 2017. 2

[12] Alvaro Parra Bustos, Tat-Jun Chin, Frank Neumann, Tobias Friedrich, and Maximilian Katzmann. A practical maximum clique algorithm for matching with pairwise constraints. *arXiv*, 2019. 2

[13] Luca Cavalli, Viktor Larsson, Martin Ralf Oswald, Torsten Sattler, and Marc Pollefeys. AdaLAM: Revisiting handcrafted outlier detection. *arXiv*, 2020. 1, 4

[14] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. Reweighted random walks for graph matching. In *ECCV*, 2010. 2

[15] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *CVPR*, 2015. 6, 7

[16] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *CVPR*, 2020. 1, 2, 3, 5, 6, 7, 8

[17] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2

[18] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *ICCV*, 2019. 1, 2, 6, 7

[19] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized ransac. In *JPRS*. Springer, 2003. 2, 5

[20] Timothee Cour, Praveen Srinivasan, and Jianbo Shi. Balanced graph matching. In *NeurIPS*, 2007. 2

[21] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPF-FoldNet: Unsupervised learning of rotation invariant 3d local descriptors. In *ECCV*, 2018. 2

[22] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPFNet: Global context aware local features for robust 3d point matching. In *CVPR*, 2018. 2

[23] Haowen Deng, Tolga Birdal, and Slobodan Ilic. 3d local features for direct pairwise registration. In *CVPR*, 2019. 2

[24] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of The ACM*, 1981. 2, 6, 7, 1

[25] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013. 6

[26] Anders Glent Buch, Yang Yang, Norbert Kruger, and Henrik Gordon Petersen. In search of inliers: 3d correspondence by local and global voting. In *CVPR*, 2014. 2

[27] Zan Gojcic, Caifa Zhou, Jan D Wegner, Leonidas J Guibas, and Tolga Birdal. Learning multiview 3d point cloud registration. In *CVPR*, 2020. 3

[28] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *CVPR*, 2019. 1, 2

[29] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *ICRA*, 2014. 7

[30] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. PVN3D: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *CVPR*, 2020. 2

[31] Paul W Holland and Roy E Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods*, 1977. 6

[32] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. PREDATOR: Registration of 3d point clouds with low overlap. *arXiv*, 2020. 2

[33] Bing Jian and Baba C Vemuri. Robust point set registration using gaussian mixture models. *PAMI*, 2010. 2

[34] Zi Jian Yew and Gim Hee Lee. 3DFeat-Net: Weakly supervised local 3d features for point cloud registration. In *ECCV*, 2018. 1, 2

[35] Olaf Kähler, Victor A Prisacariu, and David W Murray. Real-time large-scale dense 3d reconstruction with loop closure. In *ECCV*, 2016. 7

[36] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *ICRA*, 2011. 7

[37] Huu M Le, Thanh-Toan Do, Tuan Hoang, and Ngai-Man Cheung. SDRSAC: Semidefinite-based randomized approach for robust point cloud registration without correspondences. In *CVPR*, 2019. 2

[38] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *ICCV*, 2005. 1, 2, 4, 5, 6, 7

[39] Jiayuan Li, Qingwu Hu, and Mingyao Ai. GESAC: Robust graph enhanced sample consensus for point cloud registration. *ISPRS*, 2020. 2

[40] Jiaxin Li and Gim Hee Lee. Usip: Unsupervised stable interest point detection from 3d point clouds. In *ICCV*, 2019. 2

[41] Lei Li, Siyu Zhu, Hongbo Fu, Ping Tan, and Chiew-Lan Tai. End-to-end learning local multi-view descriptors for 3d point clouds. In *CVPR*, 2020. 1

[42] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 1, 2, 4

[43] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ContextDesc: Local descriptor augmentation with cross-modality context. In *CVPR*, 2019. 2

[44] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. GeoDesc: Learning local descriptors by integrating geometry constraints. In *ECCV*, 2018. 2

[45] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ASLFeat: Learning local features of accurate shape and localization. In *CVPR*, 2020. 2

[46] Nicolas Mellado, Dror Aiger, and Niloy J Mitra. Super 4pcs fast global pointcloud registration via smart indexing. In *CGF*, 2014. 2

[47] RV Mises and Hilda Pollaczek-Geiringer. Praktische verfahren der gleichungsauflösung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 1929. 5

[48] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, 2018. 2, 7

[49] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 1957. 1

[50] Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *PAMI*, 2010. 2

[51] G Dias Pais, Srikumar Ramalingam, Venu Madhav Govindu, Jacinto C Nascimento, Rama Chellappa, and Pedro Miraldo. 3DRegNet: A deep neural network for 3d point registration. In *CVPR*, 2020. 2, 3, 5, 6, 7, 8, 1

[52] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS-W*, 2017. 5

[53] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019. 2

[54] Samunda Perera and Nick Barnes. Maximal cliques based rigid body motion segmentation with a rgb-d camera. In *ACCV*. Springer, 2012. 2

[55] Fabio Poiesi and Davide Boscaini. Distinctive 3d local deep descriptors. *arXiv*, 2020. 2

[56] François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. 2015. 2

[57] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[58] Siwen Quan and Jiaqi Yang. Compatibility-guided sampling consensus for 3-d point cloud registration. *TGRS*, 2020. 1

[59] Emanuele Rodolà, Andrea Albarelli, Filippo Bergamasco, and Andrea Torsello. A scale independent selection process for 3d object recognition in cluttered scenes. *IJCV*, 2013. 2

[60] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009. 6

[61] Hamdi M Sahloul, Shouhei Shirafuji, and Jun Ota. An accurate and efficient voting scheme for a maximally all-inlier 3d correspondence set. *PAMI*, 2020. 2

[62] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 2

[63] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *CVPR*, 2019. 7

[64] Jingnan Shi, Heng Yang, and Luca Carlone. ROBIN: a graph-theoretic approach to reject outliers in robust estimation using invariants. *arXiv*, 2020. 2

[65] Weiwei Sun, Wei Jiang, Eduard Trulls, Andrea Tagliasacchi, and Kwang Moo Yi. ACNe: Attentive context normalization for robust permutation-equivariant learning. In *CVPR*, 2020. 2

[66] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 4, 7

[67] Yue Wang and Justin M. Solomon. Deep closest point: Learning representations for point cloud registration. In *ICCV*, 2019. 2

[68] Yue Wang and Justin M Solomon. PRNet: Self-supervised learning for partial-to-partial registration. In *NeuIPS*, 2019. 2

[69] Thomas Whelan, Stefan Leutenegger, R Salas-Moreno, Ben Glocker, and Andrew Davison. ElasticFusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015. 7

[70] Heng Yang and Luca Carlone. A polynomial-time solution for robust registration with extreme outlier rates. *arXiv*, 2019. 2

[71] Heng Yang, Jingnan Shi, and Luca Carlone. TEASER: Fast and certifiable point cloud registration. *arXiv*, 2020. 2, 6, 7, 1

[72] Jiaqi Yang, Ke Xian, Peng Wang, and Yanning Zhang. A performance evaluation of correspondence grouping methods for 3d rigid data matching. *PAMI*, 2019. 1, 2

[73] Jiaqi Yang, Ke Xian, Yang Xiao, and Zhiguo Cao. Performance evaluation of 3d correspondence grouping algorithms. In *3DV*, 2017. 1, 2

[74] Jiaqi Yang, Yang Xiao, Zhiguo Cao, and Weidong Yang. Ranking 3d feature correspondences via consistency voting. *Pattern Recognition Letters*, 2019. 2

[75] Zhenpei Yang, Jeffrey Z Pan, Linjie Luo, Xiaowei Zhou, Kristen Grauman, and Qixing Huang. Extreme relative pose estimation for rgb-d scans via scene completion. In *CVPR*, 2019. 4

[76] Zi Jian Yew and Gim Hee Lee. RPM-Net: Robust point matching using learned features. In *CVPR*, 2020. 2

[77] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017. 6

[78] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. OANet: Learning two-view correspondences and geometry using order-aware network. In *ICCV*, 2019. 1, 2

[79] Chen Zhao, Zhiguo Cao, Chi Li, Xin Li, and Jiaqi Yang. NM-Net: Mining reliable neighbors for robust feature correspondences. In *CVPR*, 2019. 2

[80] Ruibin Zhao, Mingyong Pang, Caixia Liu, and Yanling Zhang. Robust normal estimation for 3d lidar point clouds in urban environments. *Sensors*, 2019. 4

[81] Lei Zhou, Siyu Zhu, Zixin Luo, Tianwei Shen, Runze Zhang, Mingmin Zhen, Tian Fang, and Long Quan. Learning and matching multi-view descriptors for registration of point clouds. In *ECCV*, 2018. 2

[82] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *ECCV*, 2016. 2, 6, 7, 1

[83] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3d data processing. *arXiv*, 2018. 7, 1

7. Supplementary Material

7.1. Implementation Details of PointDSC

We provide additional information about the implementation and training details of our PointDSC. The source code will be made publicly available after the paper gets accepted.

Post-refinement. Alg. 1 shows the pseudo-code of our post-refinement step. Inspired by [7], we iteratively alternate between weighing the correspondences and computing the transformation, to improve the accuracy of the transformation matrices. The inlier threshold τ is set to 10cm and 60cm for 3DMatch and KITTI, respectively. We set the maximum iteration number to 20.

Algorithm 1: Post-Refinement Algorithm

Input: $\hat{\mathbf{R}}, \hat{\mathbf{t}}$: initial transformation; $\mathbf{X}, \mathbf{Y}$
Output: $\hat{\mathbf{R}}, \hat{\mathbf{t}}$: refined transformation.
Parameter: τ.
if $iter < max_{iter}$ **then**
  # Compute the residual and the inlier num.
  $res_i = ||\hat{\mathbf{R}}\boldsymbol{x_i} + \hat{\mathbf{t}} - \boldsymbol{y_i}||_2$
  $w_i = [\![res_i < \tau]\!]$
  $num = \sum w_i$
  # If inlier num does not change, then stop.
  if $\Delta num = 0$ **then**
    break
  else
    # Compute the weighting term.
    $\phi_i = (1 + (\frac{res_i}{\tau})^2)^{-1}$
    # Estimate transformation.
    $\hat{\mathbf{R}}, \hat{\mathbf{t}} =$
     $\arg\min_{\mathbf{R},\mathbf{t}} \sum_i^N \phi_i w_i \left\|\mathbf{R}\boldsymbol{x_i} + \mathbf{t} - \boldsymbol{y_i}\right\|^2$
  $iter = iter + 1$
else
  break

Calculation of M. In Sec. 3.4 of the main text, we calculate the compatibility between correspondences by multiplying the spatial consistency term and the feature similarity term mainly because of its simplicity and good performance. Other fusion schemes such as the weighted arithmetic average and weighted geometric average can also be used to define the compatibility metric. We have explored several alternatives but found only a marginal performance difference.

Power iteration algorithm. The power iteration algorithm can compute the leading eigenvector $\boldsymbol{e}$ of the matrix $\mathbf{M}$ in several iterations. For $\mathbf{M} \in \mathbb{R}^{k \times k}$, the power iteration operator is

$$\boldsymbol{e}^{\text{iter+1}} = \frac{\mathbf{M}\boldsymbol{e}^{\text{iter}}}{\|\mathbf{M}\boldsymbol{e}^{\text{iter}}\|}. \tag{12}$$

We initialize $\boldsymbol{e}^{\mathbf{0}} = \mathbf{1}$. By iterating Eq. 12 until convergence, we get the vector $\boldsymbol{e}$, whose elements can take real values in $[0, 1]$. In practice, we find that the power iteration algorithm usually converges in fewer than five iterations.

Data augmentation. During training, we apply data augmentation, including adding Gaussian noise with standard deviation of 0.005, random rotation angle $\in [0°, 360°)$ around an arbitrary axis, and random translation $\in [-0.5m, 0.5m]$ around each axis.

Hyper-parameters. The hyper-parameter σ_d controls the sensitivity to length difference, serving as a pairwise counterpart of the unary inlier threshold τ. The larger σ_d, the more length difference between two pairs of correspondences we can accommodate. It is set manually for a specific scene and kept fixed. Picking a scene-specific value of σ_d is easy due to its clear physical meaning. However, σ_f controlling the sensitivity to feature difference has no clear physical meaning. We thus leave σ_f to be learned by the network.

7.2. Implementation Detail of Baseline Methods

The baseline methods RANSAC [24] and FGR [82] have been implemented in Open3D [83]. For GC-RANSAC [5] and TEASER [71], we use the official implementations. Note that we use TEASER with reciprocal check; otherwise, it takes an extremely long time for testing when the number of input correspondences becomes large. For DGR [16], we use its official implementation and the released pre-trained model. Due to the unsatisfactory results of publicly released code, we re-implement SM [38] and 3DRegNet [51], with the implementation details as follows.

Spectral matching. Traditional spectral matching [38] uses a greedy algorithm based on a one-to-one mapping constraint to discretize the leading eigenvector into the inlier/outlier labels. However, the greedy algorithm often does not show satisfactory performance in real cases. For example, if the input correspondences are pre-filtered by reciprocal check, the greedy algorithm could not reject any correspondences since all of them already satisfy the one-to-one mapping constraint. The Hungarian algorithm [49] can also be used for discretization but provides results similar to the greedy algorithm. In our work, we simply select 10% of the input correspondences with the highest confidence values as the inlier set. This approach empirically shows to be effective throughout our experiments. Then the transformation between two point clouds can be estimated using the selected correspondences.

3DRegNet. We keep the network architecture proposed in 3DRegNet [51] and train it on 3DMatch using the same settings as PointDSC. However, as observed in [16], 3DRegNet does not converge during training and the registration block cannot produce reasonable results. We speculate that directly regressing the pose results in the poor performance

due to the non-linearity of the rotation space [53, 30]. Thus we regard the output of the classification block as the inlier confidence and use the confidence as the weight for weighted least-squares fitting. We then train the network using the classification loss only, since we find the registration loss does not improve the performance. The modified 3DRegNet becomes a 3D variant of PointCN [48] and achieves reasonable results.

7.3. Time Complexity Analysis

We report the average runtime of each component in the proposed pipeline on the 3DMatch test set (roughly 5k putative correspondence per fragment) in Table 6. The reported times are measured using an Intel Xeon 8-core 2.1GHz CPU (E5-2620) and an NVIDIA GTX1080Ti GPU.

SCNonlocal	Seed Selection	NSM	Post Refine	Overall
62.0	2.0	14.4	11.1	89.5

Table 6: Runtime of each component in **milli-seconds**, averaged over 1,623 test pairs of 3DMatch. The time of hypothesis selection is included in the NSM module.

7.4. Additional Statistics

We report the area under cumulative error curve (AUC) of the rotation and translation errors defined in Eq. 11 at different thresholds, as shown in Table 7. PointDSC consistently outperforms the state-of-the-arts on both the AUC of the *Rotation Error* (RE) and *Translation Error* (TE).

	RE AUC			TE AUC					
	5°	10°	15°	5cm	10cm	15cm	20cm	25cm	30cm
SM	50.14	67.24	74.37	16.29	35.98	48.61	56.90	62.57	66.67
DGR	50.22	69.98	77.78	14.13	35.28	49.32	58.50	64.74	69.19
RANSAC	49.99	70.43	78.31	12.16	33.15	47.99	57.81	64.33	68.95
GC-RANSAC	52.81	71.56	78.90	15.33	36.77	50.94	59.95	65.94	70.19
PointDSC	**57.32**	**74.85**	**81.50**	**17.85**	**40.63**	**54.56**	**63.32**	**69.02**	**73.00**

Table 7: Registration results on 3DMatch. We calculate the exact AUC following [62]: the higher, the better. We run 100k iterations for both RANSAC and GC-RANSAC.

We also report the scene-wise registration results of our method on 3DMatch in Table 8.

	RR(%)	RE(°)	TE(cm)	IP(%)	IR(%)	F1(%)
Kitchen	98.81	1.67	5.12	80.57	88.83	84.26
Home1	97.44	1.87	6.45	83.34	88.91	85.88
Home2	82.21	3.36	7.46	71.39	80.20	74.78
Hotel1	98.67	1.88	6.04	83.96	91.48	87.38
Hotel2	92.31	1.98	5.74	81.07	86.97	83.82
Hotel3	92.59	2.00	5.87	82.65	88.57	85.03
Study	89.04	2.29	9.20	77.00	83.72	79.97
Lab	80.52	1.91	8.41	70.31	77.88	73.46

Table 8: Scene-wise statistics for PointDSC on 3DMatch.

7.5. Additional Experiments

Registration results on KITTI. Due to the space limitation and the saturated performance under the FCGF setting, we only report the registration results on KITTI under the FPFH setting in the main text. Here we report the performance of all the methods combined with FCGF in Table 9. For the learning-based models DGR and PointSM, we report the performance of the models trained from scratch (labelled "re-trained") and pre-trained on the indoor dataset 3DMatch (no extra label) with the FCGF descriptor.

	RR(↑)	RE(↓)	TE(↓)	F1(↑)	Time
SM	96.76	0.50	**19.73**	22.84	0.10
RANSAC-1k	97.12	0.48	23.37	84.26	0.22
RANSAC-10k	98.02	0.41	22.94	85.05	1.43
RANSAC-100k	**98.38**	0.38	22.60	**85.42**	13.4
DGR	95.14	0.43	23.28	73.60	0.86
PointDSC	97.84	0.33	20.99	85.29	0.31
DGR re-trained	96.90	0.33	21.29	73.56	0.86
PointDSC re-trained	98.20	**0.33**	20.94	85.37	0.31

Table 9: Registration results on KITTI under the FCGF setting. The reported time numbers do not include the construction of initial correspondences.

Under low-overlapping cases. Recently, Huang et. al [32] have constructed a low-overlapping dataset 3DLoMatch from the 3DMatch benchmark to evaluate the point cloud registration algorithms under low-overlapping scenarios. To demonstrate the robustness of our PointDSC, we further evaluate our method on the 3DLoMatch dataset and report the results[1] in Table 10. Note that we directly use the model trained on 3DMatch without fine-tuning and keep 5cm voxel for the FCGF descriptor. All the other settings are the same as [32] for a fair comparison.

	5000	2500	1000	500	250	Δ
FCGF[18] + RANSAC	35.7	34.9	33.4	31.3	24.4	-
FCGF[18] + PointDSC	52.0	51.0	45.2	37.7	27.5	+10.74
Predator[32] + RANSAC	54.2	55.8	56.7	**56.1**	**50.7**	-
Predator[32] + PointDSC	**61.5**	**60.2**	**58.5**	55.4	50.4	+2.50

Table 10: Registration recall on the 3DLoMatch dataset using different numbers of points to construct the input correspondence set. The last column is the average increase brought by PointDSC.

As shown in Table 10, our method consistently outperforms RANSAC when combined with different descriptors. Moreover, our method can further boost the performance of Predator [32], a recently proposed learning-based descriptors especially designed for low-overlapping registration, showing the effectiveness and robustness of our method under high outlier ratios. PointDSC increases the registration recall by **16.3%** and **7.3%** under 5000 points setting for FCGF and Predator, respectively. Note that PointDSC does

[1]The computation of registration recall is slightly different with ours, we refer readers to [32] for more details.

not bring much performance gain when only a small number of points (e.g. less than 500) are used to construct the input correspondences mainly because some of the point cloud pairs have too few (e.g. less than 3) correspondences to identify a unique registration.

Prioritized RANSAC. Despite the common usage of the inlier probability predicted by networks in weighted least-squares fitting [16, 27], little attention has been drawn to leverage the predicted probability in a RANSAC framework. In this experiment, we derive a strong RANSAC variant (denoted as *Prioritized*) by using the inlier probability for selecting seeds to bias the sampling distribution. For a fair comparison, we implement *Prioritized* using the same codebase (`Open3D`) as RANSAC. As shown in Table 11, *Prioritized* outperforms classic RANSAC by more than 30% in terms of registration recall, indicating that the inlier probability predicted by our method is meaningful and accurate, and thus could help RANSAC to sample all-inlier subsets earlier and to achieve better performance in fewer iterations. This RANSAC variant can also be used for each correspondence subset to replace the weighted LS in Eq. 5, denoted as *Local Prioritized* in Table 11. Still, PointDSC outperforms the strong baselines with better accuracy and faster speed.

	RR(%)	RE(°)	TE(cm)	F1(%)	Time(s)
RANSAC-1k	40.05	5.16	13.65	39.23	0.08
Prioritized-1k	74.31	2.83	8.26	67.58	0.13
Local Prioritized	78.00	2.08	**6.42**	69.44	0.24
PointDSC	**78.50**	**2.07**	6.57	**69.85**	0.09

Table 11: Results on 3DMatch test set using FPFH.

Ablation on loss function. The L_{sm} is proposed to provide additional supervision, i.e., the pairwise relations between correspondences, serving as a complement to the node-wise supervision. The edge-wise supervision encourages the inliers to be concentrated in the embedding space, and this is the key assumption of our NSM module. To demonstrate its effectiveness, we compare the model trained with Eq. 10 and the model trained without the proposed spectral matching loss L_{sm} (Eq. 9) in Table 5. As shown in Table 12, L_{sm} improves the registration recall by 0.67% over the strong baseline.

	RR(↑)	RE(↓)	TE(↓)	F1(↑)	Time
PointDSC	**93.28**	**2.06**	**6.55**	**82.35**	0.09
w/o L_{sm}	92.61	2.07	6.75	81.58	0.09

Table 12: Ablation experiments of NSM module.

Effect of neighborhood size k. The size of correspondence subset, k, (Sec. 3.4) is a key parameter of our proposed method, and controls the size of each correspondence subset for neural spectral matching. We test the performance of our method with k being 10, 20, 30, 40, 50, 60, 100, and 200, respectively. As shown in Table 13, the results show that our method is robust to the choice of k. We ascribe the robustness to the neural spectral matching module, which effectively prunes the potential outliers in the retrieved subsets, thus producing a correct model even when starting from a not-all-inlier sample. We finally choose $k = 40$ for its best *Registration Recall* and modest computation cost.

	RR(↑)	RE(↓)	TE(↓)	IP(↑)	IR(↑)	F1(↑)
10	92.73	2.04	**6.44**	79.01	85.51	81.87
20	92.79	2.04	6.50	78.88	85.86	81.96
30	93.10	2.04	6.50	79.07	86.35	82.25
40	**93.28**	2.06	6.55	79.10	86.54	82.35
50	93.10	2.05	6.54	79.10	86.47	82.34
60	92.91	2.04	6.51	**79.14**	**86.61**	**82.42**
100	92.91	2.04	6.53	78.87	86.25	82.12
200	92.79	2.04	6.51	78.96	86.37	82.22

Table 13: Performance of our PointDSC when varying the size of correspondence subsets in the NSM module.

Joint training with descriptor and detector. In this part, we explore the potential of jointly optimizing the local feature learning and outlier rejection stages. A recently proposed method D3Feat [4], which efficiently performs dense feature detection and description by a single network, best suits our need. By back-propagating gradients to the input descriptors, the detector network can also be updated. Thus we build an end-to-end registration pipeline by taking the output of D3Feat as the input to our outlier rejection algorithm. We establish the correspondences using soft nearest neighbor search proposed in [27] to make the whole pipeline differentiable. We first train the feature network and the outlier rejection network separately, and then finetune them together using the losses in [4] and Eq. 10.

However, we did not observe performance improvement for the feature network in this preliminary joint training experiment. We suspect that the current losses are unable to provide meaningful gradients to the feature network. We believe that it is an interesting future direction to design proper loss formulations for end-to-end learning of both feature and outlier rejection networks.

Nevertheless, it is noteworthy that within a reasonable range, D3Feat + PointDSC achieves improved results when using fewer but more confident keypoints to build the input putative correspondences for outlier rejection. We ascribe the performance improvement to the elimination of keypoints in non-salient regions like smooth surface regions, reducing the failure registration caused by large symmetric objects in the scene. (See the visualization of failure cases Fig. 11 for more detail.) The results of D3Feat + PointDSC under different numbers of keypoints (labelled by **Joint(#num)**) are provided in Table 14 for comparisons.

7.6. Derivation of Eq. 5

For completeness, we summarize the closed-form solution of the weighted least-squares pairwise registration

	RR(↑)	RE(↓)	TE(↓)	IP(↑)	IR(↑)	F1(↑)
PointDSC	93.28	2.06	6.55	79.10	**86.54**	82.35
Joint (5000)	92.42	**1.83**	**5.87**	79.02	85.14	81.72
Joint (4000)	92.67	1.86	5.88	79.67	85.54	82.26
Joint (3000)	93.35	1.85	5.92	80.78	86.26	83.19
Joint (2500)	**93.59**	1.86	6.00	81.05	86.40	**83.38**
Joint (2000)	93.53	1.85	6.02	**81.14**	86.11	83.30
Joint (1000)	90.82	1.96	6.38	78.75	83.41	80.64

Table 14: Registration results of joint training with descriptor and detector on 3DMatch.

problem [8],

$$\hat{\mathbf{R}}, \hat{\mathbf{t}} = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i}^{N} e_i \left\| \mathbf{R}\boldsymbol{x_i} + \mathbf{t} - \boldsymbol{y_i} \right\|^2, \tag{13}$$

where $(\boldsymbol{x_i}, \boldsymbol{y_i})$ is a pair of corresponding points, with $\boldsymbol{x_i}$ and $\boldsymbol{y_i}$ being from point clouds $\mathbf{X} \in \mathbb{R}^{N\times 3}$ and $\mathbf{Y} \in \mathbb{R}^{N\times 3}$, respectively. Let $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$ denote the weighted centroids of $\mathbf{X}$ and $\mathbf{Y}$, respectively:

$$\bar{\boldsymbol{x}} = \frac{\sum_i^N e_i \boldsymbol{x_i}}{\sum_i^N e_i}, \quad \bar{\boldsymbol{y}} = \frac{\sum_i^N e_i \boldsymbol{y_i}}{\sum_i^N e_i}. \tag{14}$$

We first convert the original coordinates to the centered coordinates by subtracting the corresponding centroids,

$$\tilde{\boldsymbol{x}}_{\boldsymbol{i}} = \boldsymbol{x_i} - \bar{\boldsymbol{x}}, \;\; \tilde{\boldsymbol{y}}_{\boldsymbol{i}} = \boldsymbol{y_i} - \bar{\boldsymbol{y}}, \quad i = 1, ..., N. \tag{15}$$

The next step involves the calculation of the weighted covariance matrix $\mathbf{H}$,

$$\mathbf{H} = \tilde{\mathbf{X}}^{\mathbf{T}} \mathbf{E} \tilde{\mathbf{Y}}, \tag{16}$$

where $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ are the matrix forms of the centered coordinates and $\mathbf{E} = \text{diag}(e_1, e_2, ..., e_N)$. Then the rotation matrix from $\mathbf{X}$ to $\mathbf{Y}$ can be found by singular value decomposition (SVD),

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\mathbf{H}), \tag{17}$$

$$\hat{\mathbf{R}} = \mathbf{V} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{V}\mathbf{U}^{\mathbf{T}}) \end{bmatrix} \mathbf{U}^{\mathbf{T}}, \tag{18}$$

where $\det(\cdot)$ denotes the determinant, which is used to avoid the reflection cases. Finally, the translation between the two point clouds is computed as,

$$\hat{\mathbf{t}} = \bar{\boldsymbol{y}} - \hat{\mathbf{R}}\bar{\boldsymbol{x}}. \tag{19}$$

7.7. Qualitative Results

We show the outlier rejection results on 3DMatch and KITTI in Fig. 9 and Fig. 10, respectively. For the KITTI dataset, we use the FPFH descriptor to better demonstrate the superiority of our method. RANSAC suffers from significant performance degradation because the FPFH descriptor results in large outlier ratios, where it is harder to sample an outlier-free set. In contrast, our PointDSC still gives satisfactory results.

We also provide the visualization of failure cases of our method on 3DMatch in Fig. 11. One common failure case happens when there are large symmetry objects (e.g., the wall, floor) in a scene, resulting in rotation errors around 90°or 180°. In this case, the clusters formed by outlier correspondences could become dominant, leading to incorrect transformation hypotheses. Then an incorrect transformation is probably selected as the final solution since a large number of outlier correspondences would satisfy this transformation. To highlight this issue, we draw the distribution of rotation errors of unsuccessful registration pairs on the 3DMatch test set in Fig. 8, from which we can find that a large portion of pairs has around 90°and 180°.

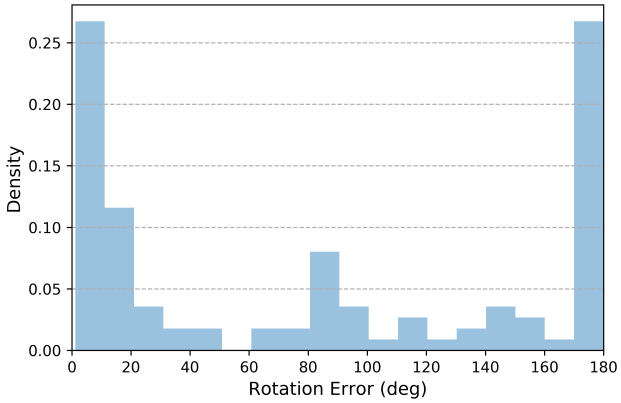


Figure 8: Rotation errors of unsuccessful registration pairs of PointDSC on the 3DMatch test set.

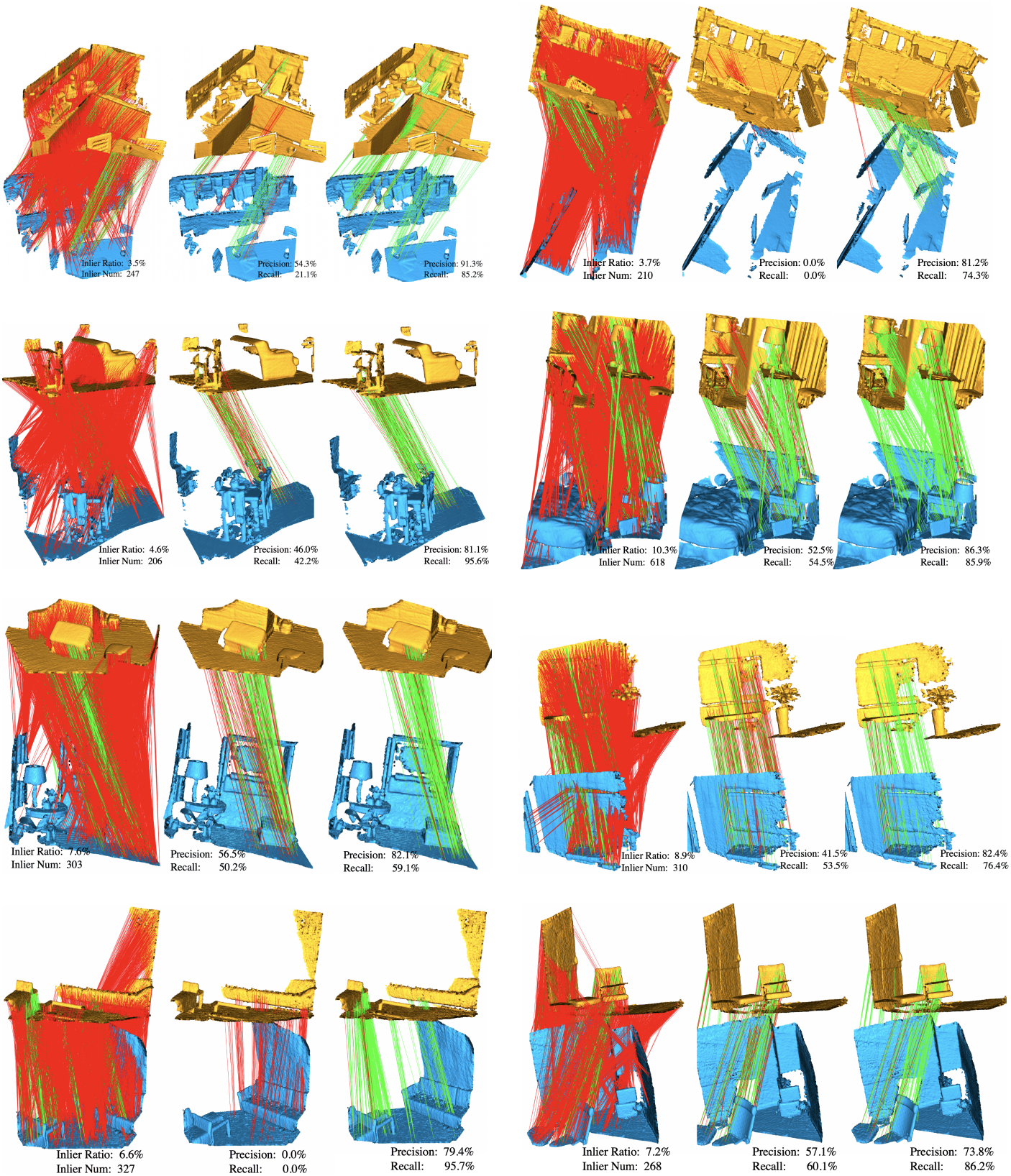


Figure 9: Visualization of outlier rejection results on the 3DMatch dataset. From left to right: input correspondences constructed by FCGF, results of RANSAC-100k, and results of PointDSC. Best viewed with color and zoom-in.

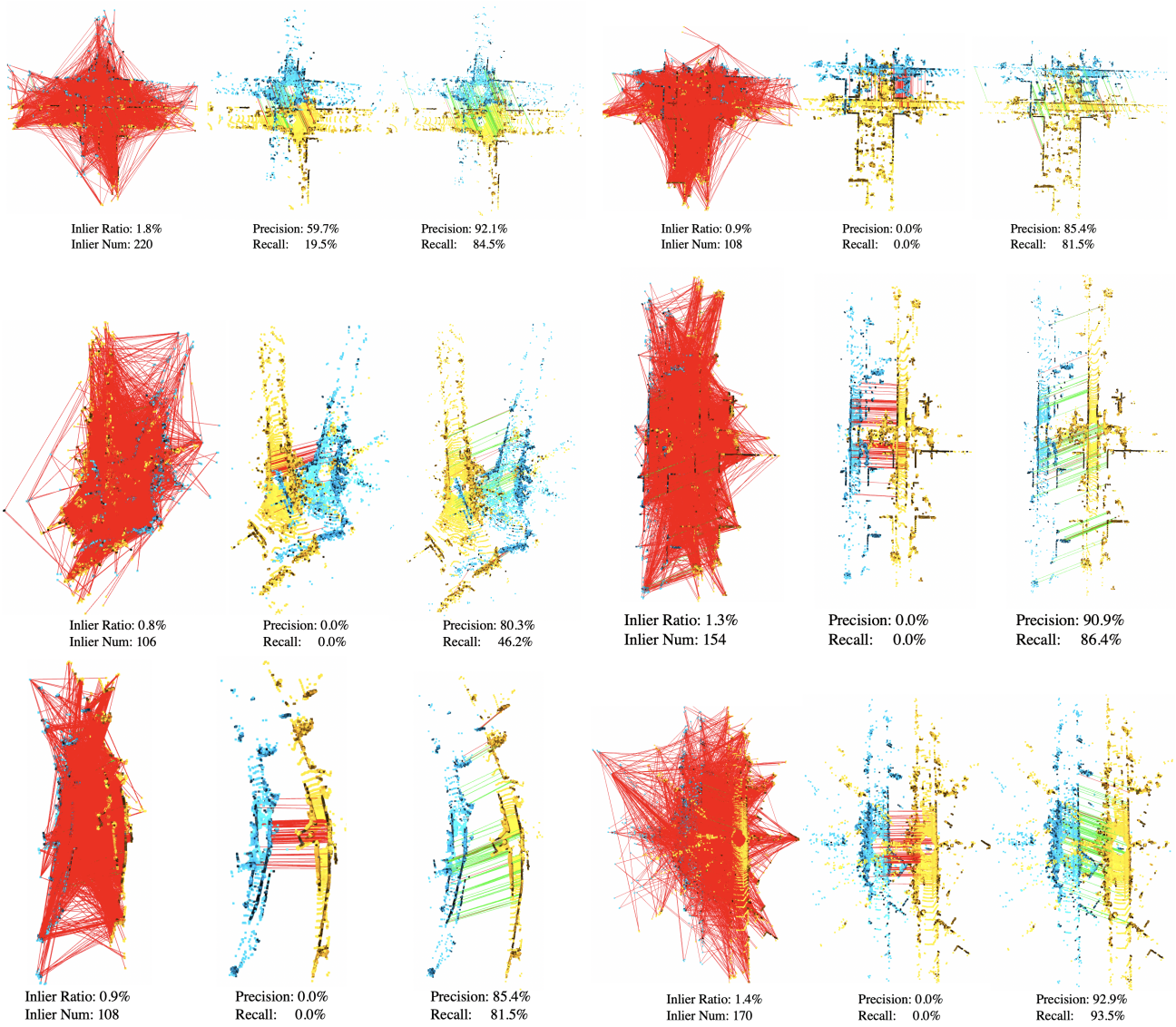


Figure 10: Visualization of outlier rejection results on the KITTI dataset. From left to right: input correspondences constructed by FPFH (we choose FPFH to better demonstrate the robustness of our method to high outlier ratios), results of RANSAC-100k, and results of PointDSC. Best viewed with color and zoom-in.

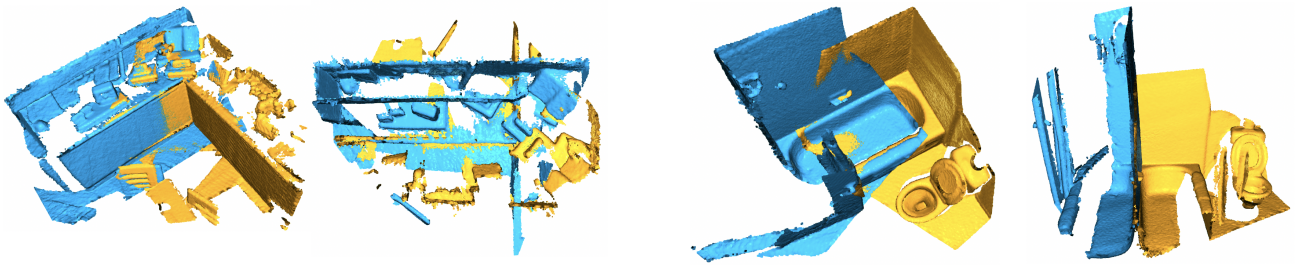


Figure 11: Two representative failure examples of our method on 3DMatch. In each example, ground-truth registration (Left) and estimated registration (Right). We observe that our method fails mainly due to the symmetries in the scene.