

Lepard: Learning partial point cloud matching in rigid and deformable scenes

Yang Li¹ Tatsuya Harada^{1,2}

¹The University of Tokyo, ²RIKEN

{liyang, harada}@mi.t.u-tokyo.ac.jp

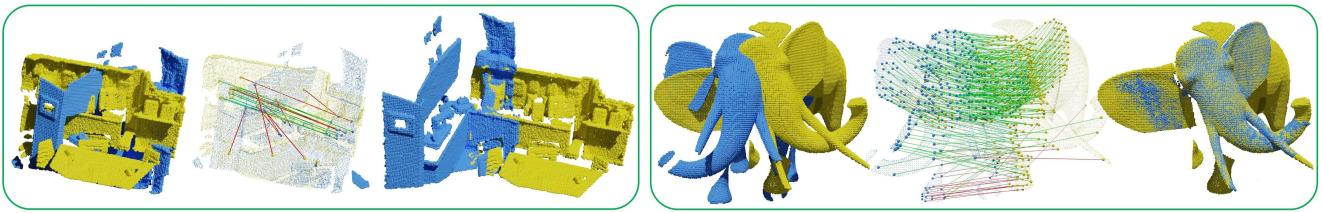


Figure 1. We propose a feature matching method for rigid (left) and deformable (right) point clouds that are captured by range sensors. Results shown are the initial alignment, the predicted matches (blue/red lines indicate inliers/outliers), and the registration results. Registration is done by RANSAC for rigid and non-rigid ICP for deformable.

Abstract

We present *Lepard*, a *Learning based approach for partial point cloud matching in rigid and deformable scenes*. The key characteristics are the following techniques that exploit 3D positional knowledge for point cloud matching: 1) An architecture that disentangles point cloud representation into feature space and 3D position space. 2) A position encoding method that explicitly reveals 3D relative distance information through the dot product of vectors. 3) A repositioning technique that modifies the cross-point-cloud relative positions. Ablation studies demonstrate the effectiveness of the above techniques. In rigid cases, Lepard combined with RANSAC and ICP demonstrates state-of-the-art registration recall of 93.9% / 71.3% on the 3DMatch / 3DLoMatch. In deformable cases, Lepard achieves +27.1% / +34.8% higher non-rigid feature matching recall than the prior art on our newly constructed 4DMatch / 4DLoMatch benchmark. Code and data are available at <https://github.com/rabbityl/leopard>.

1. Introduction

Matching partial point clouds from range sensors lies at the core of many 3D computer vision applications including SLAM and dynamic tracking and reconstruction. The former assumes rigid scenes, e.g. [28, 47], while the latter focuses on scenes that are non-rigidly deforming, e.g. [46]. This work aims at developing a robust point clouds match-

ing method for both rigid and deformable scenes.

Point cloud matching methods often consist of two phases: point cloud feature extraction followed by nearest neighbor search in feature space. Recent learning-based works have made substantial progress for representation learning in 3D data. State-of-the-art point clouds matching approaches [6, 14, 26] employ the geometry features extracted by 3D convolutional networks, such as the KPConv-based [63] or the Minkowski Engine [13]. These 3D feature extractors are strictly translation invariant, and, to a certain extent, also invariant to rotation transformations given the commonly adopted max-pooling layers in the networks and random rotation-based data augmentation during training.

Transformation invariance is well suited for local geometry feature representation. However, it may cause ambiguity in scenes that have repetitive geometry patterns. For instance, the same kind of chairs scattered in different locations of a floor, or left and right hands of a human could yield similar geometry features. We argue that such ambiguity can be resolved by enhancing geometry features with the 3D positional knowledge. Intuitively, humans associate things across observations by referring to not only things' appearance but also their relative locations.

Motivated by the above observations, we design Lepard, a novel partial point clouds matching method that exploits 3D positional knowledge. We first build our baseline using the fully convolutional feature extractor KPFCN [63], the concept of Transformer [65] with self and cross attention, and the idea of differentiable matching [55, 62]. Then, to leverage 3D position information, we introduce the fol-

lowing techniques: 1) A framework that fully disentangles the point cloud representations into a features space and a position space. 2) A position encoding method that explicitly reveals 3D relative distance information through the dot product of vectors. 3) A repositioning module that adjusts the cross-point-cloud relative positions which benefits cross attention and differentiable matching. Ablation studies demonstrate the effectiveness of the above techniques.

In addition, we propose a partial point cloud matching benchmark called 4DMatch, and its low overlap version 4DLoMatch. 4DMatch contains point clouds that are non-rigidly deforming across the time axis. Compared to the rigid situations, the time-varying geometry in 4DMatch poses more challenges for both matching and registration.

We apply Lepard for both rigid and deformable point cloud matching. In rigid cases, Lepard combined with RANSAC and ICP demonstrates state-of-the-art registration recall of **93.9% / 71.3%** on the 3DMatch / 3DLoMatch. On the newly proposed 4DMatch and 4DLoMatch benchmarks, Lepard achieves **+27.1%** and **+34.8%** higher non-rigid matching recall than the prior art.

2. Related work

2.1. Rigid Point Cloud Matching and Registration

Local descriptors prediction followed by the robust RANSAC-based [24, 57] optimization is a long-studied approach. Early works employ hand engineered descriptors [29, 53, 54, 64]. Recent learning-based approaches have made significant progresses for point feature representation [1, 6, 14–16, 18, 19, 21, 26, 30, 32, 39, 66, 71, 74, 76]. 3DMatch [76] made the first attempt to extract descriptors using the siamese network. FCGF [14] leverages the fully convolutional network [41] structure for dense feature extraction. D3feat [6] jointly learns feature description with point saliency detection. Predator [26] adopts the attention mechanism to predict overlapping regions for feature sampling. CoFiNet [74] learns feature descriptors in a coarse-to-fine manner.

Another line of research focuses on direct registration. ICP [8] and FGR [77] optimize the pose using second-order gradients. Go-ICP [72] achieves global registration with a SE(3) space searching schema. Recent works incorporate learned models into end-to-end pose optimization [2, 12, 34, 44, 49, 70, 73]. PointNetLK [2, 34] formulate point cloud registration as a Lucas Kanade-based [7] optimization task; Wang et al. [67, 68] learn registration through graph neural networks. DGR [12] and 3DRegNet [49] learn correspondence weighting networks to reject outliers.

This paper is about enhancing the point cloud feature descriptors with 3D positional knowledge.

2.2. Non-Rigid Correspondence

Estimating non-rigid correspondence from real-world sensor data is a key task for online non-rigid reconstruction [10, 20, 27, 37, 46]. DynamicFusion [46] employs the simple projective correspondence for real-time efficiency. VolumeDeform [27] incorporates SIFT [42] descriptor-based correspondence for robust non-rigid tracking. Schmidt et al. [56] uses DynamicFusion to supervise a siamese network for dense correspondence learning. DeepDeform [10] learns sparse global correspondence for patches in non-rigid deforming RGB-D sequences. Li et al. [35] learns non-rigid features through a differentiable non-rigid alignment optimization. NNRT [9] focuses on end-to-end robust correspondence estimation with an outlier rejection network. Scene flow estimation, e.g. [33, 40, 50, 69], is a closely related technique which usually delivers inter-frame level correspondence.

Non-Rigid correspondence is also a major topic in geometry processing where the input data are usually manifold surfaces. Huang et al. [25] filters outliers using isometric deformation assumption. 3DCODED [22] achieve shape correspondence through latent code optimization. Functional map [48] have been proposed to produce shape correspondence in [4, 17, 43, 52].

This paper is about developing a general non-rigid feature matching method for partial point cloud scans.

3. Problem Definition

Given a source point clouds $\mathbf{S} \in \mathbb{R}^{n \times 3}$ and a target point cloud $\mathbf{T} \in \mathbb{R}^{m \times 3}$, where n, m are the number of points, our goal is to find a set of matches \mathcal{K} , which can be used to recover the warp function $\mathcal{W} : \mathbb{R}^3 \mapsto \mathbb{R}^3$ that aligns \mathbf{S} to \mathbf{T} . In this paper, we focus on both rigid and deformable point clouds. In rigid cases, the warp function \mathcal{W} is parameterized by a $SE(3)$ transformation. In deformable cases, \mathcal{W} is generalized to the dense per-point warp field. Given the ground truth warp function \mathcal{W}_{gt} , an inlier match $(\mathbf{S}_i \in \mathbb{R}^3, \mathbf{T}_j \in \mathbb{R}^3) \in \mathcal{K}$ should satisfies $\|\mathcal{W}_{gt}(\mathbf{S}_i) - \mathbf{T}_j\|_2 < \sigma$, where $\|\cdot\|_2$ is the Euclidean norm, and σ is the tolerance radius for a match.

Partial Overlap. In real-world range sensor data, due to object motion or viewpoint change, a point in \mathbf{S} does not necessarily have a corresponding point in \mathbf{T} . This is referred to as a non-overlapping point. We define the set of overlapping points $\mathcal{O}_{\mathbf{S}}^{\mathbf{T}}$ for the source point cloud by:

$$\mathcal{O}_{\mathbf{S}}^{\mathbf{T}} = \{\mathbf{S}_i | \mathbf{S}_i \in \mathbf{S} \wedge \|\mathcal{W}_{gt}(\mathbf{S}_i) - \text{NN}(\mathcal{W}(\mathbf{S}_i), \mathbf{T})\|_2 < \sigma\}$$

where $\text{NN}(\cdot, \cdot)$ is the nearest neighbor search operator. Then the overlap ratio can be calculated by $|\mathcal{O}_{\mathbf{S}}^{\mathbf{T}}|/|\mathbf{S}|$. Fig. 4 shows examples of non-rigid point clouds pairs with different overlap ratio.

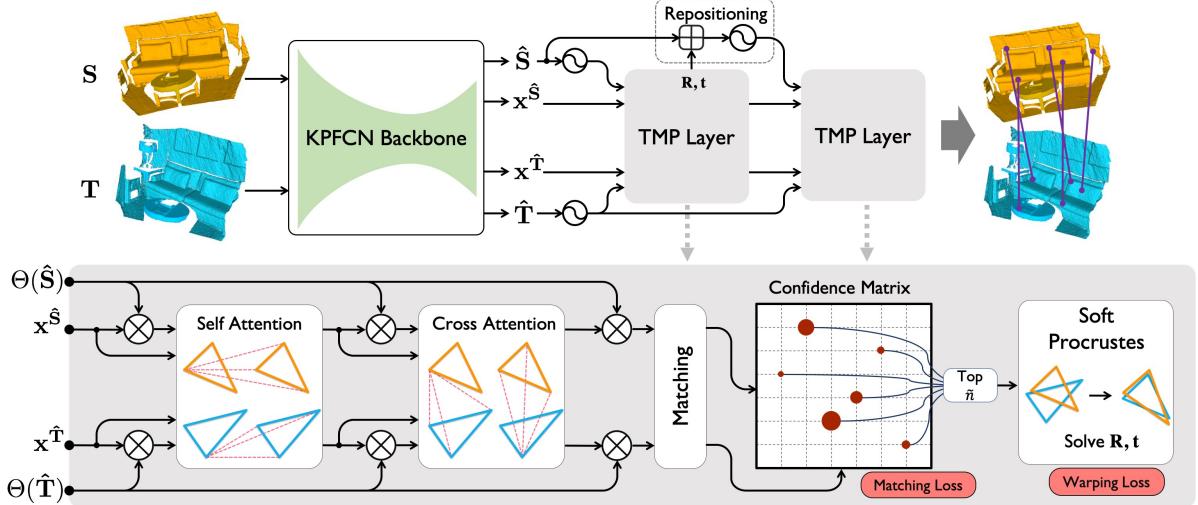


Figure 2. **Overview of the proposed method.** (The symbols are \odot : Positional encoding function $\Theta(\cdot)$; \oplus : Rigid 3D transformation; \otimes : Matrix-vector multiplication; \diamond : Source and target). Given the input point cloud S and T , the KPFCN backbone grid-subsample them to \hat{S} and \hat{T} , and extract geometry features $x^{\hat{S}}$ and $x^{\hat{T}}$ (Sec. 4.1). The positions informations are encoded as $\Theta(\hat{S})$ and $\Theta(\hat{T})$ using the 3D relative positional encoding function (Sec. 4.2). The position codes and geometry features are then processed by the first **TMPLayer** which includes a Transformer block with self and cross attentions (Sec. 4.3), a differentiable **Matching** layer (Sec. 4.4), and a soft **Procrustes** layer to estimate the rigid fitting estimation R, t (Sec. 4.6). Based on the rigid fitting estimation, the **Repositioning** layer adjusts source’s position code $\Theta(\hat{S})$ (Sec. 4.7). Given the updated positions and transformed features, the second **TMPLayer** predicts the final matches.

4. Method

Fig. 2 shows the overview of the proposed method.

4.1. Local geometry feature extraction

Given the input source and target point clouds S and T , We use the function Φ to extract multi-level geometry features. Φ does the following two mappings:

$$(\hat{S}, x^{\hat{S}}) = \Phi(S), \quad (\hat{T}, x^{\hat{T}}) = \Phi(T)$$

Where $\hat{S} \in \mathbb{R}^{\hat{n} \times 3}$ and $\hat{T} \in \mathbb{R}^{\hat{m} \times 3}$ are the output point clouds, $x^{\hat{S}} \in \mathbb{R}^{\hat{n} \times d}$ and $x^{\hat{T}} \in \mathbb{R}^{\hat{m} \times d}$ are the extracted features with dimension $d = 528$.

We build Φ based on the KPFCN backbone [63]. KPFCN possesses the inductive bias of translation equivariance and locality, which are well suited for local geometry feature extraction. The default KPFCN has an UNet-like structure with the same number of pooling/un-pooling layers in the encoder/decoder. We remove the decoder units after the 2nd to the last un-pooling layer from the KPFCN backbone. Thus the output \hat{S} and \hat{T} are the down-sampled point clouds. See supplementary for details. This down-sampling is crucial for efficient computation in the following transformer (Sec. 4.3) and matching ((Sec. 4.4) layers, where the time complexity are both $O(n^2)$ of the number of points.

4.2. Relative 3D Positional Encoding

The KPFCN backbone learns strict translation invariant features. Translation invariant could cause ambiguity in scenes that have symmetric structures or globally repetitive geometry patches. To resolve such ambiguity, we enhance features with the transformation sensitive 3D positional information.

We use the Rotary positional encoding proposed in [60] and extend it to the 3D case. Given a 3D point $S_i = (x, y, z) \in \mathbb{R}^3$, and the it’s feature $x_i^S \in \mathbb{R}^d$. The position encoding function $\mathcal{PE} : \mathbb{R}^3 \times \mathbb{R}^d \mapsto \mathbb{R}^d$ is defined by

$$\mathcal{PE}(S_i, x_i^S) = \Theta(S_i)x_i^S = \begin{pmatrix} M_1 & & & & & \\ & M_2 & & & & \\ & & \ddots & & & \\ & & & M_{d/6} & & \end{pmatrix} x_i^S$$

where $\Theta(S_i)$ is a block diagonal matrix. Each diagonal block with size 6×6 is defined by

$$M_k = \begin{pmatrix} \cos x\theta_k & -\sin x\theta_k & 0 & 0 & 0 & 0 \\ \sin x\theta_k & \cos x\theta_k & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos y\theta_k & -\sin y\theta_k & 0 & 0 \\ 0 & 0 & \sin y\theta_k & \cos y\theta_k & 0 & 0 \\ 0 & 0 & 0 & 0 & \cos z\theta_k & -\sin z\theta_k \\ 0 & 0 & 0 & 0 & \sin z\theta_k & \cos z\theta_k \end{pmatrix}$$

where $\theta_k = \frac{1}{10000^{(k-1)/d}}, k \in [1, 2, \dots, d/6]$ encodes the index in the feature channel.

Compared to the Sinusoidal encoding [11, 65], it has two advantages: 1) $\Theta(\cdot)$ is an orthogonal function, the encoding only changes the feature’s direction but not the fea-

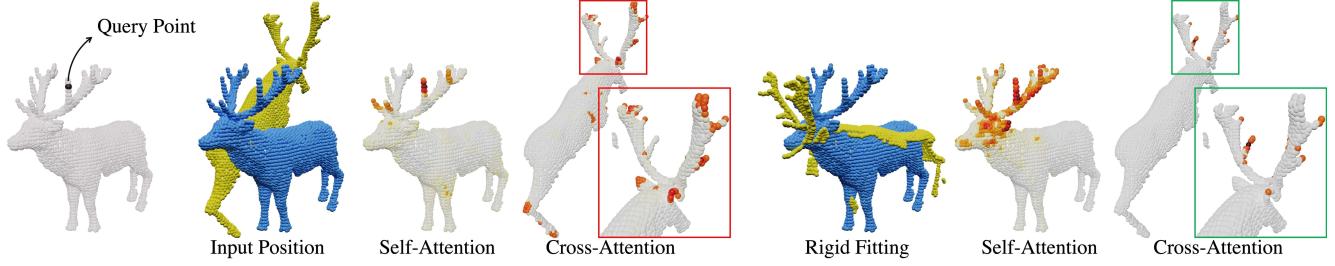


Figure 3. Visualization of self/cross attention heat maps and the rigid fitting based repositioning. In the 2nd TMP layer, self-attention expands to cover larger context, and cross attention converges to the corresponding region.

ture’s length, which could potentially stabilize the learning process. 2) The dot product of two encoded features ($\mathcal{PE}(\mathbf{S}_i, \mathbf{x}_i^S), \mathcal{PE}(\mathbf{S}_j, \mathbf{x}_j^S)$) can be derived to:

$$[\Theta(\mathbf{S}_i)\mathbf{x}_i^S]^T \Theta(\mathbf{S}_j)\mathbf{x}_j^S = (\mathbf{x}_i^S)^T \Theta(\mathbf{S}_j - \mathbf{S}_i)\mathbf{x}_j^S \quad (1)$$

which means the relative 3D distance information can be explicitly revealed by the dot product. We adopt this positional encoding in both the transformer (Sec. 4.3) and matching (Sec. 4.4) layers. The comparison with Sinusoidal encoding can be found in Sec. 6.1.

4.3. Transformer

After the local geometry extraction, $\mathbf{x}^{\hat{S}}$ and $\mathbf{x}^{\hat{T}}$ are passed through the transformer block with a self attention layer to aggregate the global context, followed by a cross attention layer to exchange information between two point clouds. Following [65], the attention operation selects the relevant information by measuring the similarity between the query vector \mathbf{q} , and the key vector \mathbf{k} . The output vector is the sum of the value vector \mathbf{v} weighted by the similarity scores.

Self Attention Layer. In self attention layer, \mathbf{q} and (\mathbf{k}, \mathbf{v}) are obtained from the same point clouds (either from source or the target). Below shows the example of self attention for the source $\hat{\mathbf{S}}$. The vectors $\mathbf{q}, \mathbf{k}, \mathbf{v}$ are first computed by

$$\mathbf{q}_i = \Theta(\hat{\mathbf{S}}_i)W_{\mathbf{q}}\mathbf{x}_i^{\hat{S}} \quad \mathbf{k}_j = \Theta(\hat{\mathbf{S}}_j)W_{\mathbf{k}}\mathbf{x}_j^{\hat{S}} \quad \mathbf{v}_j = W_{\mathbf{v}}\mathbf{x}_j^{\hat{S}} \quad (2)$$

where $W_{\mathbf{q}}, W_{\mathbf{k}}, W_{\mathbf{v}} \in \mathbb{R}^{d \times d}$ are learnable projection matrices. The feature $\mathbf{x}_i^{\hat{S}}$ is finally updated by

$$\mathbf{x}_i^{\hat{S}} \leftarrow \mathbf{x}_i^{\hat{S}} + \text{MLP}(\text{cat}[\mathbf{q}_i, \sum_j a_{ij}\mathbf{v}_j]) \quad (3)$$

where $a_{ij} = \text{softmax}(\mathbf{q}_i \mathbf{k}_j^T / \sqrt{d})$ is the attention weight, $\text{MLP}(\cdot)$ denotes a 3-layer fully connected network, and $\text{cat}[\cdot, \cdot]$ is the concatenation operator.

Cross-Attention Layer. In cross-attention layer, the input vectors \mathbf{q} and (\mathbf{k}, \mathbf{v}) are obtained from different point clouds depending on the direction of cross-attention ($\hat{\mathbf{S}} \rightarrow \hat{\mathbf{T}}$ or $\hat{\mathbf{T}} \rightarrow \hat{\mathbf{S}}$). After replacing the contents for \mathbf{q}, \mathbf{k} , and \mathbf{v} , the formations are the same with self attention.

4.4. Position Aware Feature Matching

After the transformer layer, we compute the scoring matrix \mathcal{S} between two point clouds as

$$\mathcal{S}(i, j) = \frac{1}{\sqrt{d}} \langle \Theta(\hat{\mathbf{S}}_i)W_{\hat{\mathbf{S}}}\mathbf{x}_i^{\hat{S}}, \Theta(\hat{\mathbf{T}}_j)W_{\hat{\mathbf{T}}}\mathbf{x}_j^{\hat{T}} \rangle \quad (4)$$

where $W_{\hat{\mathbf{S}}}, W_{\hat{\mathbf{T}}} \in \mathbb{R}^{d \times d}$ are learnable projection matrices. The features are position-encoded such that the matching algorithm could take the spatial distance into account. We apply softmax on both dimensions (known as the dual-softmax operation [51, 62]) to convert the scoring matrix to confidence matrix \mathcal{C} .

$$\mathcal{C}(i, j) = \text{Softmax}(\mathcal{S}(i, \cdot)) \cdot \text{Softmax}(\mathcal{S}(\cdot, j))$$

Another option for matching is the sinkhorn optimal transport algorithm as in [55]. The comparison can be seen in Sec. 6.1.

Match Prediction. Based on the confidence matrix \mathcal{C} , we select matches with confidence higher than a threshold of θ_c , and further enforce with mutual nearest neighbor (MNN) criteria. The influence of θ_c can be found in the supplemental ablation study.

4.5. Disentanglement between Position and Feature.

As shown in Fig. 2, we hold the position code and geometry features in separated data streams. They are combined only when a similarity matrix needs to be computed. In the transformer layers (c.f. Eqn. 2 and 3), position code $\Theta(\cdot)$ is only multiplied to query \mathbf{q} and key \mathbf{k} but not to the value \mathbf{v} , i.e. $\Theta(\cdot)$ can influence the attention weights but can not be a part of the feature. This technique leads to the *Disentanglement* between position and feature. Opposed to this is regarded as *Entangled*, which does not hold separate data streams for position and feature, i.e. they are mixed at the very beginning of the transformer. The comparison is seen in Sec. 6.1.

4.6. Rigid Fitting With Soft Procrustes

Given the the confidence matrix \mathcal{C} , we select the matches set $\mathcal{K}_{\text{soft}}$ with top \hat{n} matching scores, and then fit them with

a rigid rotation $\mathbf{R} \in SO3$ and translation $\mathbf{t} \in \mathbb{R}^3$. Here \hat{n} is the number of points in the source cloud $\hat{\mathbf{S}}$. Following [3], the rotation is computed from the SVD decomposition of the matrix $H = U\Sigma V^T$. $H \in \mathbb{R}^{3 \times 3}$ is obtained with:

$$H = \sum_{(i,j) \in \mathcal{K}_{soft}} \tilde{\mathcal{C}}(i,j) \hat{\mathbf{S}}_i \hat{\mathbf{T}}_j^T$$

where $\tilde{\mathcal{C}}(i,j)$ is the normalized confidence score. Then the rotation and is computed by

$$\mathbf{R} = U \text{diag}(1, 1, \det(UV^T)) V$$

Then the translation is obtained with

$$\mathbf{t} = \frac{1}{|\mathcal{K}_{soft}|} \left(\sum_{(i,\cdot) \in \mathcal{K}_{soft}} \hat{\mathbf{S}}_i - \mathbf{R} \sum_{(\cdot,j) \in \mathcal{K}_{soft}} \hat{\mathbf{T}}_j \right)$$

4.7. Repositioning

The position encoding in Eqn. 1 can reveal 3D relative distance information between a pair of points. However, this distance knowledge is often incorrect for point pairs from two unaligned point clouds. In other words, this position encoding benefits self-attention but may become irrelevant or noisy signals for cross-attention and matching. To this end, we adjust the position code $\Theta(\hat{\mathbf{S}}_i)$ of $\hat{\mathbf{S}}$ with the rigid fitting \mathbf{R}, \mathbf{t} from the soft Procrustes layer by

$$\Theta(\hat{\mathbf{S}}_i) \leftarrow \Theta(\mathbf{R}\hat{\mathbf{S}}_i + \mathbf{t})$$

We call it as repositioning. An example is shown in Fig. 3. Intuitively, repositioning pushes corresponding points closer in the position space such that it better informs cross attention and also facilitates position aware matching.

4.8. Supervision

Matching Loss. We minimize the focal loss over the confidence matrix \mathcal{C} returned by the matching layer. It is defined as:

$$\mathcal{L}_m = -\frac{1}{|\mathcal{K}_{gt}|} \sum_{(i,j) \in \mathcal{K}_{gt}} \alpha(1 - \mathcal{C}(i,j))^\gamma \log \mathcal{C}(i,j)$$

where $\alpha = 0.25$ and $\gamma = 2$ are empirically decided focal loss parameters as in [38], and \mathcal{K}_{gt} is the set of ground-truth matches. During training, we warp $\hat{\mathbf{S}}$ to $\hat{\mathbf{T}}$ using the ground-truth warp function \mathcal{W}_{gt} , and then collect the set of mutual nearest neighbors of the two-point clouds that are below a distance threshold as \mathcal{K}_{gt} .

Warping Loss. We minimize the L_1 loss for the point clouds that is warped by the \mathbf{R}, \mathbf{t} from the Procrustes layer. It is defined as

$$\mathcal{L}_w = \frac{1}{|\mathcal{O}_{\hat{\mathbf{S}}}^{\hat{\mathbf{T}}}|} \sum_{i \in \mathcal{O}_{\hat{\mathbf{S}}}^{\hat{\mathbf{T}}}} |\mathcal{W}_{gt}(\hat{\mathbf{S}}_i) - \mathbf{R}\hat{\mathbf{S}}_i - \mathbf{t}|$$

where $\mathcal{O}_{\hat{\mathbf{S}}}^{\hat{\mathbf{T}}}$ is the set of overlapping points in $\hat{\mathbf{S}}$, and $\mathcal{W}_{gt}(\cdot)$ is the ground truth warp function. Intuitively, in rigid cases, \mathcal{L}_w regularizes the optimization by suppressing false-positives in \mathcal{K}_{soft} and also encourages sub-point accuracy for soft correspondences; in deformable cases, \mathcal{L}_w tries to approximate a “root” pose that aligns the principal part of the overlapping region, e.g. the deer in Fig. 3.

Total Loss. As shown in Fig. 2, the Transform-Matching-Procrustes (TMP) block repeats two times. The total loss combines the matching loss and warpping loss from the 1st and 2nd TMP blocks: $\mathcal{L} = (\mathcal{L}_m^1 + \mathcal{L}_m^2) + \lambda_w(\mathcal{L}_w^1 + \mathcal{L}_w^2)$ where λ_w is the weighting factor of warpping loss. We show ablation study for λ_w and the number of TMP blocks (2, 3, and 4) in supplementary.

5. 4DMatch

We propose 4DMatch, a benchmark for matching and registration of partial point clouds with time-varying geometry. 4DMatch is constructed using the sequence from DeformingThings4D [36] which contains 1,972 animation sequences with ground truth dense correspondence. We randomly select 1761 animations and generate partial point cloud scans by synthesizing depth images. The selected 1761 sequences are divided into 1232/176/353 as train/valid/test sets. Point cloud pairs in the 353 testing sequence are eventually split into either 4DMatch or 4DLoMatch based on an overlap ratio threshold of 45%. Fig. 4

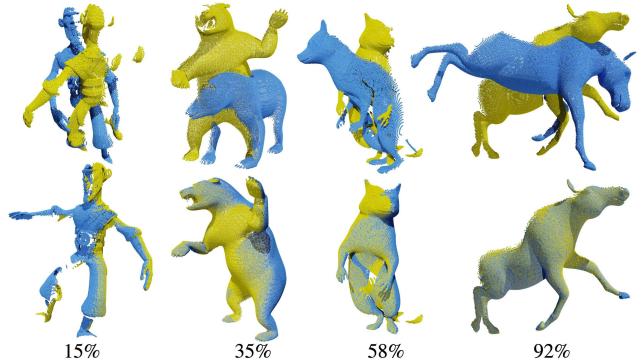


Figure 4. Examples in 4DMatch/4DLoMatch with different overlap ratio. Overlap ratio is computed relative to the source (Blue). Partial overlap is the joint effect of scene deformation and camera viewpoint change.

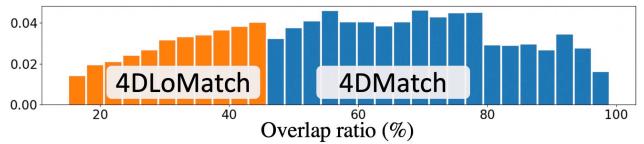


Figure 5. Histogram of the 4DMatch and 4DLoMatch benchmark. The overlap ratio threshold is set to 45%.

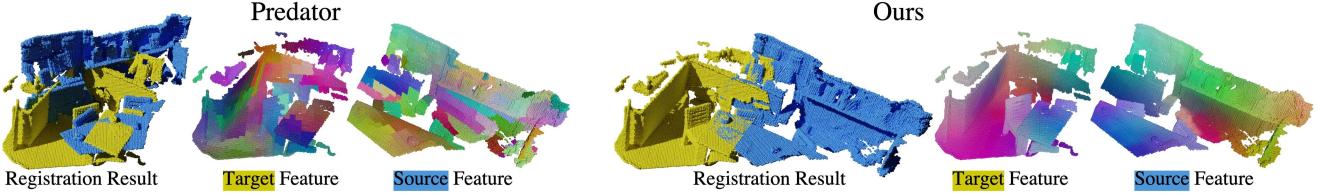


Figure 6. **Qualitative rigid point cloud registration results in 3DLoMatch benchmark.** We use T-SNE to reduce the feature dimension to 3 and then normalize it to [0, 255] as RGB values. We propagate our output feature $\Theta(\hat{\mathbf{S}}_i)W_{\hat{\mathbf{S}}}x_i^{\hat{\mathbf{S}}}$ and $\Theta(\hat{\mathbf{T}}_j)W_{\hat{\mathbf{T}}}x_j^{\hat{\mathbf{T}}}$ as in Eqn. 4 to the raw point clouds via interpolation as Eqn. 6. The feature visualization shows that our approach learns a position-aware feature representation and also accurately reflects the inter-fragment relative position.

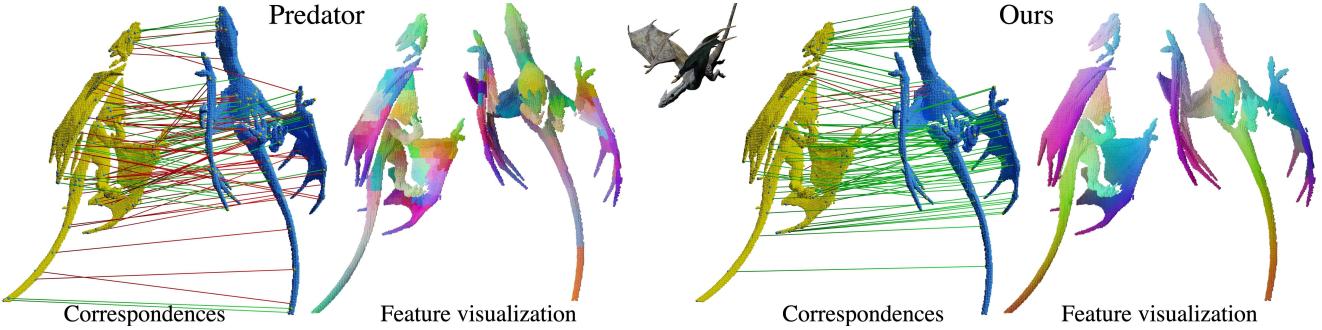


Figure 7. **Qualitative deformable point cloud matching results in 4DMatch benchmark.** Blue/red lines indicate inliers/outliers. In this example, the “Flying Dragon” has a bilateral symmetric shape. T-SNE feature visualization shows that Predator [26] learns similar features for the two wings of the dragon, while our method can discriminate between the left and right wings.

shows examples of point cloud pairs with different overlap ratios. Fig. 5 is the histogram of overlap ratio. The following shows the evaluation metrics for 4DMatch.

Inlier ratio (IR). IR measures the fraction of correct matches in the predicted correspondences set \mathcal{K}_{pred} . A match is correct if it lies within a threshold $\sigma = 0.04m$ after the transformation using the ground truth warp function \mathcal{W}_{gt} . It is defined as

$$IR = \frac{1}{|\mathcal{K}_{pred}|} \sum_{(\mathbf{p}, \mathbf{q}) \in \mathcal{K}_{pred}} [||\mathcal{W}_{gt}(\mathbf{p}) - \mathbf{q}||_2 < \sigma] \quad (5)$$

where $||\cdot||_2$ is the Euclidean norm and $[\cdot]$ is the Iverson bracket.

Non-rigid Feature Matching Recall (NFMR). NFMR measures the fraction of ground-truth matches ($\mathbf{u} \in \mathbb{R}^3, \mathbf{v} \in \mathbb{R}^3 \in \mathcal{K}_{gt}$) that can be successfully recovered by the predicted correspondences ($\mathbf{p} \in \mathbb{R}^3, \mathbf{q} \in \mathbb{R}^3 \in \mathcal{K}_{pred}$). Based on \mathcal{K}_{pred} , a sparse 3D scene flow field $\mathcal{F} = \{\mathbf{q} - \mathbf{p} | (\mathbf{p}, \mathbf{q}) \in \mathcal{K}_{pred}\}$ for the set of anchor points $\mathcal{A} = \{\mathbf{p} | (\mathbf{p}, \mathbf{q}) \in \mathcal{K}_{pred}\}$ are constructed. Then the flow field \mathcal{F} is propagated from \mathcal{A} to a source point \mathbf{u} in \mathcal{K}_{gt} using the inverse distance interpolation

$$\Gamma(\mathbf{u}, \mathcal{A}, \mathcal{F}) = \sum_{\mathcal{A}_i \in \text{knn}(\mathbf{u}, \mathcal{A})} \frac{\mathcal{F}_i ||\mathbf{p} - \mathcal{A}_i||_2^{-1}}{\sum_{\mathcal{A}_i \in \text{knn}(\mathbf{u}, \mathcal{A})} ||\mathbf{u} - \mathcal{A}_i||_2^{-1}} \quad (6)$$

where $\text{knn}(\cdot, \cdot)$ denotes the k -nearest neighbors search with $k = 3$. Finally, we define NFMR as

$$\text{NFMR} = \frac{1}{|\mathcal{K}_{gt}|} \sum_{(\mathbf{u}, \mathbf{v}) \in \mathcal{K}_{gt}} [||\Gamma(\mathbf{u}, \mathcal{A}, \mathcal{F}) - \mathbf{v}||_2 < \sigma]$$

Note that NFMR directly measures the fraction of ground-truth correspondences that are “recalled”. It is a different concept from the Feature Matching Recall (FMR) in the rigid case [76].

6. Experimental Results

3DMatch and 3DLoMatch. 3DMatch [76] and 3DLoMatch [26] are a benchmark of indoor rigid scan matching and registration. 3DMatch contains scan pairs with overlap ratios greater than 30%, while 3DLoMatch contains scan pairs with overlap ratios between 10% and 30%. Following previous works [26, 74], we report the following metrics: Inlier Ratio (IR), Feature Matching Recall (FMR), rigid Registration Recall (RR), Relative Rotation Error (RTE), and Relative Translation Error (RTE). RR is widely recognized as the ultimate metric because it measures the fraction of successfully registered scan pairs.

Implementation details. Our method is implemented using Pytorch and trained using SGD on Nvidia A100 (80G)

GPU. We use a batch size of 8 and apply padding and masking to handle different point clouds sizes. On 3DMatch, we follow the train/validation split as Predator [26]. The training on 3DMatch and 4DMatch both converge around the 15th epoch. We save the model with the best validation loss. More implementation details are seen in the supplementary.

6.1. Ablation Study

–Does disentangling position and feature help? For non-rigid cases, disentangling position and feature achieve similar results to the entangled version on 4DMatch and achieve significantly better results on 4DLoMatch (c.f. Tab. 1). Fea. & Pos. disentanglement also gain +1.1% / +2.5% RR on 3DMatch / 3DLoMatch (c.f. Tab. 3).

–Positional encoding: absolute vs relative. Our relative 3D positional encoding yields +3.8% higher NFMR on 4DLoMatch and +1.5% / +2.3% higher RR on 3DMatch / 3DLoMatch than the absolute sinusoidal encoding [65] (c.f. Tab. 1, 3).

–Does Repositioning make sense? Repositioning gains +2.9% / +3.3% NFMR on 4DMatch / 4DLoMatch and +0.9% / +1.0% RR on 3DMatch / 3DLoMatch (c.f. Tab. 1, 3). Note that the performances drop significantly with the *Random rotation-based* positioning. The *Oracle deformation* in Tab. 1 and *Oracle rigid fitting* in Tab. 3 achieve near perfect results. These results demonstrate the importance of positional knowledge for point cloud registration. In Tab. 1, the *Oracle rigid fitting* refers to the best rigid fitting for the ground truth deformation. We consider it as the upper bound of the rigid fitting-based repositioning approach.

–Matching algorithm: Sinkhorn vs dual softmax. The dual softmax operator achieves higher scores on all benchmarks than the Sinkhorn approach.

Ablation Target		4DMatch		4DLoMatch	
		NFMR↑	IR↑	NFMR↑	IR↑
Fea. & Pos.	Entangled	84.1	83.5	63.3	51.4
	Disentangled*	83.7	82.7	66.9	55.7
PE type	Absolute	83.7	82.2	63.1	51.8
	Relative*	83.7	82.7	66.9	55.7
Positioning	Random rotation w/o Repositioning	79.2	78.2	58.4	46.7
	Repositioning*	80.8	80.5	63.6	53.7
	Oracle rigid fitting	83.7	82.7	66.9	55.7
	Oracle deformation	91.5	89.7	80.2	67.2
		100.0	99.8	100.0	97.6
Matching	Sinkhorn	81.7	77.4	59.6	46.1
	Dual-Softmax*	83.7	82.7	66.9	55.7

Table 1. **Ablation study on 4DMatch.** * indicates the default configuration of our method.

Category	Method	S [†]	4DMatch		4DLoMatch	
			NFMR↑	IR↑	NFMR↑	IR↑
Probabilistic Model	CPD [45]		6.0	5.5	0.4	0.2
	BCPD [23]		11.4	10.1	1.2	0.5
Functional Map [‡]	ZoomOut [43]		4.2	3.8	1.3	0.5
	GeoFM [17]	✓	25.0	20.5	11.7	4.5
Scene Flow	PWC [69]	✓	21.6	20.0	10.0	7.2
	FLOT [50]	✓	27.1	24.9	15.2	10.7
	NSFP [33]		18.5	16.3	1.2	0.5
Feature Matching	D3Feat [6]	✓	55.5	54.7	27.4	21.5
	Predator [26]	✓	56.4	60.4	32.1	27.5
	Ours	✓	83.7	82.7	66.9	55.7

Table 2. **Quantitative Results on 4DMatch.** S[†] denotes supervised methods. [‡] Point cloud Laplacian is obtained via [58].

6.2. Comparison with the state-of-the-art

4DMatch & 4DLoMatch. As shown in Tab. 2, our method achieves +27.1% / +34.8% higher NFMR and +22.3% / +28.2% higher IR than Predator. Fig. 7 shows the qualitative point cloud matching results on 4DMatch. We also report results from probabilistic registration [23, 45], functional map-based [17, 43], and scene flow methods [33, 50, 69]. Overall, feature matching methods [6, 26] show superior performance than other categories. Existing scene flow methods, can not handle large global motion. Functional map methods fail because they need connected shapes to obtain reliable Laplacian matrices, while the shapes in 4DMatch are usually disconnected due to occlusion (c.f. examples in Fig. 4). The Coherent Point Drift (CPD) models [23, 45] do not work well on the partial scan data.

3DMatch & 3DLoMatch. Compared to Predator [26], our method achieves +1.7% / +6.6% higher RR on 3DMatch / 3DLoMatch (c.f. Tab. 3). Fig. 6 shows the qualitative results for a low overlap case. As shown in Tab. 4, Our method also produces better RRE and RTE than Predator; the point-to-point ICP-based postprocessing can further improve the registrations.

6.3. Integrating to non-rigid registration.

We further integrate the predicted matches to non-rigid point cloud registration. For registration, we adopt the non-rigid iterative closest point (N-ICP) [31, 78] method which iteratively minimizes the typical energy function:

$$\mathbf{E}_{total}(\mathcal{G}) = \lambda_c \mathbf{E}_{corr}(\mathcal{G}) + \lambda_r \mathbf{E}_{reg}(\mathcal{G})$$

where \mathbf{E}_{corr} is the correspondence term, \mathbf{E}_{reg} is the regularization term as in [59], and \mathcal{G} is the latent deformation graph model. The correspondence term \mathbf{E}_{corr} is defined by the L2 distance between corresponding points. See the supplementary for formal definitions.

Fig. 8 shows the qualitative results of non-rigid registration on the dragon and elephant examples. The *N-ICP*

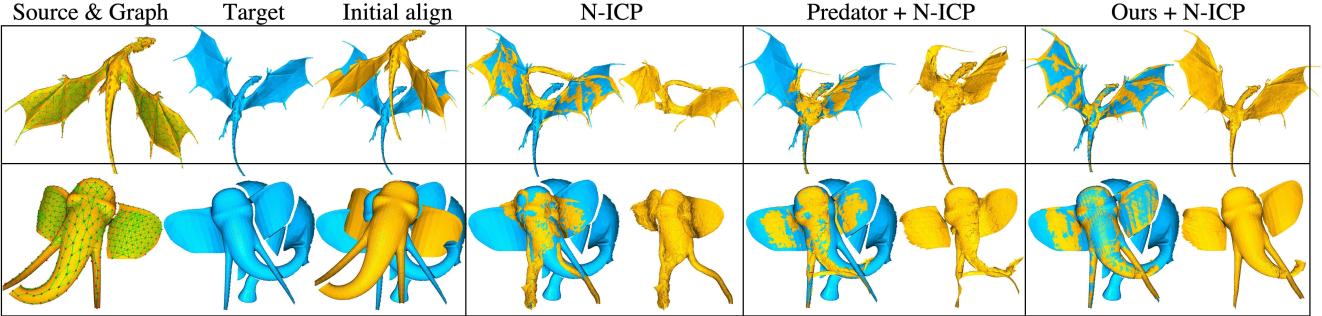


Figure 8. Qualitative non-rigid registration results. As shown on the left end, scene deformation is approximated by the deformation graph. Results shown are the final alignment and the transformed source.

	3DMatch [76]			3DLoMatch [26]		
	FMR↑	IR↑	RR↑	FMR↑	IR↑	RR↑
3DSN [21]	94.7	36.0	81.5	61.9	11.4	36.6
FCGF [14]	95.2	56.9	88.2	60.9	21.4	45.8
D3Feat [6]	95.8	39.0	85.8	69.3	13.2	40.2
Predator [26]	96.7	58.0	91.8	78.6	26.7	62.4
Ours-Entangled	97.8	46.8	92.5	83.8	23.3	66.5
Ours-Absolute	97.4	62.0	92.1	84.1	29.3	67.7
Ours-Random rotation	97.5	60.6	90.8	79.2	25.6	60.3
Ours-w/o repositioning	98.0	57.6	92.8	84.2	27.8	68.0
Ours-Sinkhorn	97.6	46.5	92.2	81.7	17.7	64.9
Ours (Default)	98.3	55.5	93.5	84.5	26.0	69.0
Oracle Rigid fitting	100.0	99.3	100.0	100.0	95.0	99.8

Table 3. **Feature matching and RANSAC registration results on 3DMatch.** Refer to Tab. 1 for the configs of Ours (Default). Note that, in this paper, **RR** is averaged on all scan pairs to reflect the true percentage of successful registrations.

		3DMatch			3DLoMatch		
		RRE ↓	RTE ↓	RR ↑	RRE ↓	RTE ↓	RR ↑
Predator	RANSAC	2.72	7.8	91.8	4.44	11.6	62.4
Ours	RANSAC	<u>2.48</u>	<u>7.2</u>	<u>93.5</u>	<u>4.10</u>	<u>10.8</u>	<u>69.0</u>
Predator	RANSAC+ICP	2.06	6.2	92.3	3.46	9.8	65.2
Ours	RANSAC+ICP	1.96	6.0	93.9	3.17	8.9	71.3

Table 4. **RRE (°), RTE (cm) and RR (%) on 3DMatch.** Point-to-point ICP can further refine the transformations.

baseline adopts the simple Euclidean space nearest neighbor search-based correspondence. *Predator + N-ICP* and *Ours + N-ICP* replace the correspondence term by the predicted matches during the beginning 20 iterations and then run 45 N-ICP iterations for refinements. Our method better informs non-rigid registration than Predator.

7. Conclusion

By leveraging the positional knowledge, Lepard demonstrates state-of-the-art feature matching results for both rigid and deformable point clouds. A promising direction is extending it to end-to-end registration. A few limita-

tions are yet to be addressed: 1) Lepard is a coarse matching approach. Fine-grained correspondence could be obtained with learning-based refinement, e.g. [62, 74]. 2) In deformable cases, Lepard does not explicitly handle topological changes. A potential solution is to jointly learn matching with motion segmentation. 3) Finally, matching and registration in the low overlapping cases is particularly challenging due to data incompleteness. Fig. 9 shows such failure cases. A potential solution for low-overlap scenarios is to expand the mutual information via data completion [36, 75]. Learning outlier rejection as [5, 49] could also benefit registration.

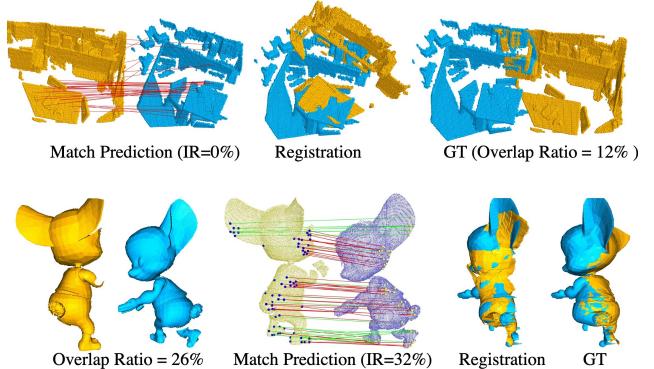


Figure 9. **Failure cases in rigid (top) and non-rigid (bottom) matching.** The low overlapping cases are still challenging, in particular, if the point clouds have similar patterns in the non-overlapping region. Given the similar table and chairs (top), and a similar half-body (bottom), our method fails to treat them as different features.

8. Acknowledgement

This work was partially supported by JST AIP Acceleration Research JPMJCR20U3, Moonshot R&D Grant Number JPMJPS2011, CREST Grant Number JPMJCR2015, and Basic Research Grant (Super AI) of Institute for AI and Beyond of the University of Tokyo.

References

- [1] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo. Spinnet: Learning a general surface descriptor for 3d point cloud registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11753–11762, 2021. [2](#)
- [2] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7163–7172, 2019. [2](#)
- [3] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, pages 698–700, 1987. [5](#)
- [4] S. Attaiki, G. Pai, and M. Ovsjanikov. Dpfm: Deep partial functional maps. *arXiv preprint arXiv:2110.09994*, 2021. [2](#)
- [5] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15859–15869, 2021. [8](#)
- [6] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6359–6367, 2020. [1, 2, 7, 8](#)
- [7] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3):221–255, 2004. [2](#)
- [8] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. [2](#)
- [9] A. Božič, P. Palafox, M. Zollhöfer, A. Dai, J. Thies, and M. Nießner. Neural non-rigid tracking. *arXiv preprint arXiv:2006.13240*, 2020. [2](#)
- [10] A. Božič, M. Zollhöfer, C. Theobalt, and M. Nießner. Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7002–7012, 2020. [2](#)
- [11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. [3](#)
- [12] C. Choy, W. Dong, and V. Koltun. Deep global registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2514–2523, 2020. [2](#)
- [13] C. Choy, J. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3075–3084, 2019. [1](#)
- [14] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8958–8966, 2019. [1, 2, 8](#)
- [15] H. Deng, T. Birdal, and S. Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 602–618, 2018. [2](#)
- [16] H. Deng, T. Birdal, and S. Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 195–205, 2018. [2](#)
- [17] N. Donati, A. Sharma, and M. Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. [2, 7](#)
- [18] M. El Banani, L. Gao, and J. Johnson. UnsupervisedR&R: Unsupervised Pointcloud Registration via Differentiable Rendering. In *CVPR*, 2021. [2](#)
- [19] M. El Banani and J. Johnson. Bootstrap Your Own Correspondences. In *ICCV*, 2021. [2](#)
- [20] W. Gao and R. Tedrake. Surfelwarp: Efficient non-volumetric single view dynamic reconstruction. *arXiv preprint arXiv:1904.13073*, 2019. [2](#)
- [21] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5545–5554, 2019. [2, 8](#)
- [22] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. [2](#)
- [23] O. Hirose. A bayesian formulation of coherent point drift. *IEEE transactions on pattern analysis and machine intelligence*, 43(7):2269–2286, 2020. [7](#)
- [24] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke. Registration with the point cloud library: A modular framework for aligning in 3-d. *IEEE Robotics & Automation Magazine*, 22(4):110–124, 2015. [2](#)
- [25] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas. Non-rigid registration under isometric deformations. In *Computer Graphics Forum*, volume 27, pages 1449–1457. Wiley Online Library, 2008. [2](#)
- [26] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler. Predator: Registration of 3d point clouds with low overlap. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4267–4276, 2021. [1, 2, 6, 7, 8, 4](#)
- [27] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger. Volumed deform: Real-time volumetric non-rigid reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 362–379, 2016. [2](#)

- [28] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011. 1
- [29] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449, 1999. 2
- [30] M. Khouri, Q.-Y. Zhou, and V. Koltun. Learning compact geometric features. In *Proceedings of the IEEE international conference on computer vision*, pages 153–161, 2017. 2
- [31] H. Li, R. W. Sumner, and M. Pauly. Global correspondence optimization for non-rigid registration of depth scans. In *Computer graphics forum*, volume 27, pages 1421–1430. Wiley Online Library, 2008. 7
- [32] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2
- [33] X. Li, J. Kaesemeyer Pontes, and S. Lucey. Neural scene flow prior. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 7
- [34] X. Li, J. K. Pontes, and S. Lucey. Pointnetlk revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12763–12772, 2021. 2
- [35] Y. Li, A. Bozic, T. Zhang, Y. Ji, T. Harada, and M. Nießner. Learning to optimize non-rigid tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4910–4918, 2020. 2
- [36] Y. Li, H. Takehara, T. Taketomi, B. Zheng, and M. Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. *arXiv preprint arXiv:2105.01905*, 2021. 5, 8
- [37] Y. Li, T. Zhang, Y. Nakamura, and T. Harada. Splitfusion: Simultaneous tracking and mapping for non-rigid scenes. *arXiv preprint arXiv:2007.02108*, 2020. 2
- [38] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5
- [39] X. Liu, B. D. Killeen, A. Sinha, M. Ishii, G. D. Hager, R. H. Taylor, and M. Unberath. Neighborhood normalization for robust geometric feature learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13049–13058, 2021. 2
- [40] X. Liu, C. R. Qi, and L. J. Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 529–537, 2019. 2
- [41] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015. 2
- [42] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2
- [43] S. Melzi, J. Ren, E. Rodola, A. Sharma, P. Wonka, and M. Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *arXiv preprint arXiv:1904.07865*, 2019. 2, 7
- [44] T. Min, C. Song, E. Kim, and I. Shim. Distinctiveness oriented positional equilibrium for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5490–5498, 2021. 2
- [45] A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE transactions on pattern analysis and machine intelligence*, 32(12):2262–2275, 2010. 7
- [46] R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, 2015. 1, 2
- [47] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE, 2011. 1
- [48] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):1–11, 2012. 2
- [49] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascento, R. Chellappa, and P. Miraldo. 3dregnet: A deep neural network for 3d point registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7193–7203, 2020. 2, 8
- [50] G. Puy, A. Boulch, and R. Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 527–544. Springer, 2020. 2, 7
- [51] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic. Neighbourhood consensus networks. *arXiv preprint arXiv:1810.10510*, 2018. 4
- [52] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. In *Computer Graphics Forum*, volume 36, pages 222–236. Wiley Online Library, 2017. 2
- [53] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009. 2
- [54] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ international conference on intelligent robots and systems*, pages 3384–3391. IEEE, 2008. 2

- [55] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. [1](#), [4](#)
- [56] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2016. [2](#)
- [57] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007. [2](#)
- [58] N. Sharp and K. Crane. A Laplacian for Nonmanifold Triangle Meshes. *Computer Graphics Forum (SGP)*, 39(5), 2020. [7](#)
- [59] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry Processing*, volume 4, pages 109–116, 2007. [7](#), [3](#)
- [60] J. Su, Y. Lu, S. Pan, B. Wen, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. [3](#)
- [61] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. *ACM Transactions on Graphics (TOG)*, 26(3):80, 2007. [2](#)
- [62] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8922–8931, 2021. [1](#), [4](#), [8](#)
- [63] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019. [1](#), [3](#)
- [64] F. Tombari, S. Salti, and L. Di Stefano. Unique shape context for 3d data description. In *Proceedings of the ACM workshop on 3D object retrieval*, pages 57–62, 2010. [2](#)
- [65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. [1](#), [3](#), [4](#), [7](#)
- [66] H. Wang, Y. Liu, Z. Dong, W. Wang, and B. Yang. You only hypothesize once: Point cloud registration with rotation-equivariant descriptors. *arXiv preprint arXiv:2109.00182*, 2021. [2](#)
- [67] Y. Wang and J. M. Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3523–3532, 2019. [2](#)
- [68] Y. Wang and J. M. Solomon. Prnet: Self-supervised learning for partial-to-partial registration. *arXiv preprint arXiv:1910.12240*, 2019. [2](#)
- [69] W. Wu, Z. Wang, Z. Li, W. Liu, and L. Fuxin. Pointpwc-net: A coarse-to-fine network for supervised and self-supervised scene flow estimation on 3d point clouds. *arXiv preprint arXiv:1911.12408*, 2019. [2](#), [7](#)
- [70] H. Xu, S. Liu, G. Wang, G. Liu, and B. Zeng. Omnet: Learning overlapping mask for partial-to-partial point cloud registration. *arXiv preprint arXiv:2103.00937*, 2021. [2](#)
- [71] H. Yang, W. Dong, L. Carlone, and V. Koltun. Self-supervised geometric perception. In *CVPR*, 2021. [2](#)
- [72] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE transactions on pattern analysis and machine intelligence*, 38(11):2241–2254, 2015. [2](#)
- [73] Z. J. Yew and G. H. Lee. Rpm-net: Robust point matching using learned features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11824–11833, 2020. [2](#)
- [74] H. Yu, F. Li, M. Saleh, B. Busam, and S. Ilic. Cofinet: Reliable coarse-to-fine correspondences for robust point cloud registration. *arXiv preprint arXiv:2110.14076*, 2021. [2](#), [6](#), [8](#)
- [75] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, 2021. [8](#)
- [76] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1802–1811, 2017. [2](#), [6](#), [8](#), [1](#)
- [77] Q.-Y. Zhou, J. Park, and V. Koltun. Fast global registration. In *European conference on computer vision*, pages 766–782. Springer, 2016. [2](#)
- [78] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (ToG)*, 33(4):156, 2014. [7](#)

Supplementary Material: Lepard: Learning partial point cloud matching in rigid and deformable scenes

Supplementary material includes: ablation study (Sec. I); implementation details (Sec. II, III, IV, and V); formal definition of non-rigid registration (Sec. VI); and more results on 3DMatch and 4DMatch (Sec. VII).

I. Ablation study

-Influence of Warping Loss Weight. Applying warping loss in general yields higher NFMR and IR 4DMatch and 4DLoMatch. In particular, in the low overlap situations, the performance grows steadily with the increasing of the motion loss weight (c.f. Tab. 1). In 3DMatch and 3DLoMatch, warping loss significantly increases the Inlier Rate (IR). However, it leads to a decrease in Registration Recall (c.f. Tab. 2). We assume that this is because the warping loss might suppress some border cases correspondences which could have benefited the RANSAC. In deformable cases, a high inlier rate is desired for successful non-rigid registration. However, in rigid cases, the inlier rate is less important since RANSAC is very robust to noise. Therefore, we set $\lambda_w = 0.1$ for 4DMatch and $\lambda_w = 0.0$ for 3DMatch.

	4DMatch		4DLoMatch	
	NFMR↑	IR↑	NFMR↑	IR↑
$\lambda_w = 0$	82.9	82.4	62.1	52.2
$\lambda_w = 0.05$	85.3	83.9	65.1	54.5
$\lambda_w = 0.1^*$	83.7	82.7	66.9	55.7

Table 1. Influence of warping loss on 4DMatch.

	3DMatch [76]			3DLoMatch [26]		
	FMR↑	IR↑	RR↑	FMR↑	IR↑	RR↑
$\lambda_w = 0$	98.0	63.7	93.6	85.6	29.5	69.0
$\lambda_w = 0.05$	97.8	66.6	92.9	84.1	36.5	67.9
$\lambda_w = 0.1$	97.6	71.5	92.9	84.6	38.8	68.2

Table 2. Influence of warping loss on 3DMatch.

-Influence of Confidence Threshold. In rigid cases, increasing the confidence threshold of correspondence leads to a decrease in registration recall (c.f. Tab. 3). Same to the above ablation, we assume that this is because increasing the confidence threshold inevitably suppresses some borderline correspondences which could have benefited the RANSAC. In deformable cases, increasing the confidence threshold results in a higher IR but getting a lower NFMR

	3DMatch (RR↑)	3DLoMatch (RR↑)
$\theta_c = 0.05$	93.6	69.0
$\theta_c = 0.1$	92.3	67.9
$\theta_c = 0.15$	91.7	67.0
$\theta_c = 0.2$	91.2	65.3

Table 3. Influence of confidence thresholds on 3DMatch and 3DLoMatch.

Method	4DMatch			4DLoMatch		
	$ \mathcal{K}_{pred} $	NFMR↑	IR↑	$ \mathcal{K}_{pred} $	NFMR↑	IR↑
D3Feat (1000)	267	51.6	52.7	204	23.6	21.2
D3Feat (3000)	532	55.5	54.7	379	27.4	<u>21.5</u>
D3Feat (5000)	697	<u>56.1</u>	<u>55.3</u>	473	<u>28.1</u>	21.3
Predator (1000)	273	53.3	60.0	205	30.6	<u>29.8</u>
Predator (3000)	534	56.4	<u>60.4</u>	372	<u>32.1</u>	27.5
Predator (5000)	698	<u>56.8</u>	59.3	480	<u>32.1</u>	25.0
Ours ($\theta_c=0.2$)	523	82.2	85.4	325	63.1	60.4
Ours ($\theta_c=0.1$)*	596	83.7	82.7	407	66.9	55.7
Ours ($\theta_c=0.05$)	624	83.9	80.9	447	67.6	52.5

Table 4. Influence of confidence thresholds on 4DMatch and 4DLoMatch. D3Feat [6] and Predator [26] probabilistically sample points either from a saliency heat map or from a machability×overlap heat map (numbers in brackets are the numbers of sampled points). Ours uses the confidence threshold θ_c to get matches from the confidence matrix (c.f. Sec. 4.4). All methods apply the mutual nearest neighbor criteria to filter matches. $|\mathcal{K}_{pred}|$ indicates the average number of final predicted correspondences.

(c.f. Tab. 4). We found $\theta_c=0.1$ a good trade-off between precision and recall.

-Adding more TMP blocks. We tested 3 and 4 TMP layers. The corresponding number of the Repositioning layer is 2 and 3 because it is placed between every two consecutive TMP layers. As shown in Tab. 5, in 3DMatch, additional layers do not improve the results; in 4DMatch, 3 TMP layers achieve the best results. Adding layers inevitably increase the training time.

	Number of TMP layer	2	3	4
	Number of Repositioning layer	1	2	3
Rigid	RR(%)↑ on 3DMatch	93.6	92.8	93.0
	RR(%)↑ on 3DLoMatch	69.0	68.2	68.8
	Training Time (hour)↓	20	25	31
Deformable	NFMR(%)↑ on 4DMatch	83.7	85.9	84.5
	NFMR(%)↑ on 4DLoMatch	66.9	68.1	59.6
	Training Time (hour)↓	18	21	24

Table 5. Ablation study of number of TMP layers.

II. Sparse $\Theta(\cdot)$ Multiplication

Taking the advantage of the sparsity of $\Theta(\cdot)$, given a position $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ and a feature $\mathbf{x} \in \mathbb{R}^d$, the multiplication $\Theta(\mathbf{p})\mathbf{x}$ can be efficiently realized by

$$\begin{pmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{x}(2) \\ \mathbf{x}(3) \\ \mathbf{x}(4) \\ \mathbf{x}(5) \\ \vdots \\ \mathbf{x}(d/6-1) \\ \mathbf{x}(d/6-1) \end{pmatrix} \otimes \begin{pmatrix} \cos x\theta_0 \\ \cos x\theta_0 \\ \cos y\theta_0 \\ \cos y\theta_0 \\ \cos z\theta_0 \\ \cos z\theta_0 \\ \vdots \\ \cos z\theta_{d/6-1} \\ \cos z\theta_{d/6-1} \end{pmatrix} + \begin{pmatrix} -\mathbf{x}(1) \\ \mathbf{x}(0) \\ -\mathbf{x}(3) \\ \mathbf{x}(2) \\ -\mathbf{x}(5) \\ \mathbf{x}(4) \\ \vdots \\ -\mathbf{x}(d/6-1) \\ \mathbf{x}(d/6-1) \end{pmatrix} \otimes \begin{pmatrix} \sin x\theta_0 \\ \sin x\theta_0 \\ \sin y\theta_0 \\ \sin y\theta_0 \\ \sin z\theta_0 \\ \sin z\theta_0 \\ \vdots \\ \sin z\theta_{d/6-1} \\ \sin z\theta_{d/6-1} \end{pmatrix}$$

III. Hyper Parameters

		3DMatch	4DMatch
Metric	Inlier threshold	0.1m	0.04m
	RR threshold	0.2m	—
	FMR threshold	5%	—
	NFMR threshold	—	0.04m
Match Prediction	Confidence threshold θ_c	0.05	0.1
	Apply MNN	False	True
KPFCN Config	Input subsampling radius	0.025m	0.01m
Supervision	GT match radius	0.06m	0.024m
	Warping loss weight λ_w	0.0	0.1

RR: Registration Recall

FMR: Feature Matching Recall

NFMR: Non-rigid Feature Matching Recall

MNN: Mutual Nearest Neighbor

Table 6. The hyper parameters for metric evaluation, match prediction, KPFCN backbone, and training loss

IV. Time and memory expense

	Predator [26]	Lepard (Ours)
Average time (s)	0.18	0.10
Cuda memory (MB)	13,361	6,595

Table 7. Time and Cuda memory usage of inference on an Nvidia A100 (80G) GPU. Time is averaged on 2193 testing samples in 4DLoMatch. Lepard is about twice as efficient as Predator on both time and memory.

KPFCN	Self Att. ($\times 2$)	Cross Att. ($\times 2$)	Matching ($\times 2$)	Procrustes ($\times 2$)
0.0109	0.0016 ($\times 2$)	0.0014 ($\times 2$)	0.0023 ($\times 2$)	0.0191 ($\times 2$)

Table 8. Average time (s) of Lepard function inference on an Nvidia A100 (80G) GPU. Time is averaged on 2193 testing samples in 4DLoMatch.

V. KPFCN backbone architecture

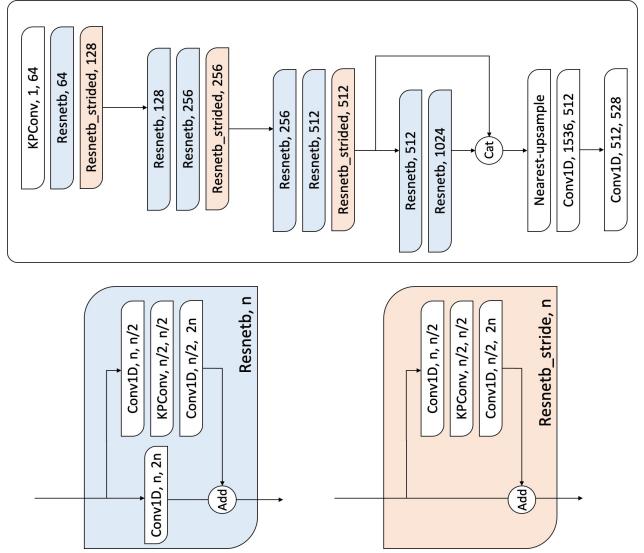


Figure 1. Details of the KPFCN backbone architecture.

VI. Non-Rigid Registration

This section introduce the non-rigid registration technique used in this paper.

Deformation Model. To represent the dense motion from a source to a target, we adapt the embedded deformation model of Sumner et al. [61]. The non-rigid deformation is parameterized by the deformation graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where \mathcal{V} is the set of node and \mathcal{E} is the set of edge. As shown in Fig. 2, we evenly sample graph nodes \mathcal{V} over the source point cloud surface. Each point in the scene has a 3D location: $\mathbf{g}_i \in \mathbb{R}^3$. The motion of a node $i \in \mathcal{V}$ is parameterized by a translation vector: $\mathbf{t}_i \in \mathbb{R}^3$ and a rotation matrix: $\mathbf{R}_i \in \text{SO3}$. In addition, we represent rotations by $\mathbf{R}_i = \exp(\varphi_i^\wedge)\mathbf{R}_i$, where $\varphi_i = [0, 0, 0]$ represents the delta of the rotation in axis-angle form. $(\cdot)^\wedge$ operator convert a 3-dimensional vector to a 3×3 skew-symmetric matrix. $\exp : \mathfrak{so}_3 \mapsto \text{SO3}$ map the skew-symmetric matrix to 3×3 rotation matrix using the Rodrigues formula. Finally, all unkowns in the graph are

$$\mathcal{G} = (\varphi_1, \dots, \varphi_{|\mathcal{V}|}, \mathbf{t}_1, \dots, \mathbf{t}_{|\mathcal{V}|})$$

Non-rigid Warping Function Given a point $\mathbf{p} \in \mathbb{R}^3$, the non-rigid warping function \mathcal{W} is defined as

$$\mathcal{W}(\mathbf{p}) = \sum_{i \in \mathcal{V}} w_{\mathbf{p}, i} (\mathbf{R}_i(\mathbf{p} - \mathbf{g}_i) + \mathbf{g}_i + \mathbf{t}_i)$$

where $w_{\mathbf{p}, i} \in \mathbb{R}$ is the “skinning weight” that measure the influence of node i . They are computed as

$$w_{\mathbf{p}, i} = C e^{\frac{1}{2\gamma^2} ||\mathbf{v}_i - \mathbf{p}||_2^2}$$

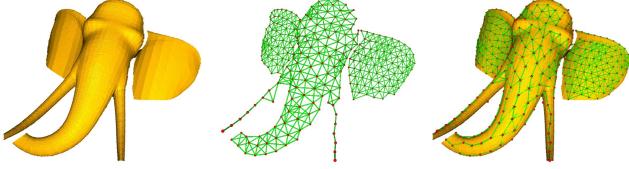


Figure 2. Deformation model. Nodes \mathcal{V} (red dot) are evenly sampled over the source surface. Edges \mathcal{E} (green lines) are computed between nodes based on geodesic connectivity. The point cloud examples in 4DMatch are obtained from depth images. To compute geodesic distance, we construct the surface triangle mesh by connecting the nearby pixels' 3D locations. We filter the triangles with an edge larger than 4cm. During registration, for numerically stable optimization, we ignore point cloud clusters with fewer than 40 deformation nodes.

where γ is the coverage radius of a node, for which we set to 0.9 cm for 4DMatch examples, C denotes the normalization constant, ensuring that skinning weights add up to one

$$\sum_{\mathcal{V}_i \in \mathcal{V}} w_{\mathbf{p},i} = 1$$

Energy Function. The energy function of non-rigid iterative closest point (N-ICP) consists of two terms: the correspondence term and the regularization term. Given a set of matches \mathcal{K} , and the confidence of the correspondences $c_{(\mathbf{p}_s, \mathbf{p}_t)}$ where $(\mathbf{p}_s, \mathbf{p}_t) \in \mathcal{K}$. Correspondence term is defined as

$$\mathbf{E}_{corr}(\mathcal{G}) = \sum_{(\mathbf{p}_s, \mathbf{p}_t) \in \mathcal{K}} c_{(\mathbf{p}_s, \mathbf{p}_t)}^2 \|\mathcal{W}(\mathbf{p}_s) - \mathbf{p}_t\|_2^2$$

We use ARAP [59] as the regularization term

$$\mathbf{E}_{reg}(\mathcal{G}) = \sum_{(i,j) \in \mathcal{E}} \|\mathbf{R}_i(\mathbf{g}_j - \mathbf{g}_i) + \mathbf{g}_i + \mathbf{t}_i - (\mathbf{g}_j + \mathbf{t}_j)\|^2$$

The total energy function is

$$\mathbf{E}_{reg}(\mathcal{G}) = \lambda_c \mathbf{E}_{corr}(\mathcal{G}) + \lambda_a \mathbf{E}_{reg}(\mathcal{G})$$

Residual and Partial Derivatives. The followings show the residuals and partial derivatives for optimization. Derivative of the wrapping function \mathcal{W}

$$\begin{aligned} \frac{\partial \mathcal{W}(\mathbf{p})}{\partial \varphi_i} &= -w_{\mathbf{p},i}(\mathbf{R}_i(\mathbf{p} - \mathbf{g}_i))^\wedge \\ \frac{\partial \mathcal{W}(\mathbf{p})}{\partial \mathbf{t}_i} &= w_{\mathbf{p},i}\mathbf{I}_3 \end{aligned}$$

where \mathbf{I}_3 is the 3×3 identity matrix. Residual term for a correspondence $(\mathbf{p}_s, \mathbf{p}_t) \in \mathcal{K}$

$$\mathbf{r}_{(\mathbf{p}_s, \mathbf{p}_t)}^{corr} = \sqrt{\lambda_c} c_{(\mathbf{p}_s, \mathbf{p}_t)} (\mathcal{W}(\mathbf{p}_s) - \mathbf{p}_t)$$

Derivative of correspondence residual $\mathbf{r}_{(\mathbf{p}_s, \mathbf{p}_t)}^{corr}$

$$\begin{aligned} \frac{\partial \mathbf{r}_{(\mathbf{p}_s, \mathbf{p}_t)}^{corr}}{\partial \varphi_i} &= -\sqrt{\lambda_c} c_{(\mathbf{p}_s, \mathbf{p}_t)} w_{\mathbf{p}_s,i} (\mathbf{R}_i(\mathbf{p}_s - \mathbf{g}_i))^\wedge \\ \frac{\partial \mathbf{r}_{(\mathbf{p}_s, \mathbf{p}_t)}^{corr}}{\partial \mathbf{t}_i} &= \sqrt{\lambda_c} c_{(\mathbf{p}_s, \mathbf{p}_t)} w_{\mathbf{p},i} \mathbf{I}_3 \end{aligned}$$

Residual term for regularization tern $(i, j) \in \mathcal{E}$

$$\mathbf{r}_{(i,j)}^{reg} = \sqrt{\lambda_a} (\mathbf{R}_i(\mathbf{g}_j - \mathbf{g}_i) + \mathbf{g}_i + \mathbf{t}_i - (\mathbf{g}_j + \mathbf{t}_j))$$

Derivative of regularization term $\mathbf{r}_{(i,j)}^{reg}$

$$\begin{aligned} \frac{\partial \mathbf{r}_{(i,j)}^{reg}}{\partial \varphi_i} &= -\sqrt{\lambda_a} (\mathbf{R}_i(\mathbf{g}_j - \mathbf{g}_i))^\wedge \\ \frac{\partial \mathbf{r}_{(i,j)}^{reg}}{\partial \mathbf{t}_i} &= \sqrt{\lambda_a} \mathbf{I}_3 \\ \frac{\partial \mathbf{r}_{(i,j)}^{reg}}{\partial \mathbf{t}_j} &= -\sqrt{\lambda_a} \mathbf{I}_3 \end{aligned}$$

The full Jacobian matrix $\mathbf{J} \in \mathbb{R}^{(3|\mathcal{K}|+3|\mathcal{E}|) \times 6|\mathcal{V}|}$ is shown as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{r}_1^{corr}}{\partial \varphi_1} & \dots & \frac{\partial \mathbf{r}_1^{corr}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}_1^{corr}}{\partial \mathbf{t}_1} & \dots & \frac{\partial \mathbf{r}_1^{corr}}{\partial \mathbf{t}_{|\mathcal{V}|}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{r}_{|\mathcal{K}|}^{corr}}{\partial \varphi_1} & \dots & \frac{\partial \mathbf{r}_{|\mathcal{K}|}^{corr}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}_{|\mathcal{K}|}^{corr}}{\partial \mathbf{t}_1} & \dots & \frac{\partial \mathbf{r}_{|\mathcal{K}|}^{corr}}{\partial \mathbf{t}_{|\mathcal{V}|}} \\ \hline \frac{\partial \mathbf{r}_1^{reg}}{\partial \varphi_1} & \dots & \frac{\partial \mathbf{r}_1^{reg}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}_1^{reg}}{\partial \mathbf{t}_1} & \dots & \frac{\partial \mathbf{r}_1^{reg}}{\partial \mathbf{t}_{|\mathcal{V}|}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{r}_{|\mathcal{E}|}^{reg}}{\partial \varphi_1} & \dots & \frac{\partial \mathbf{r}_{|\mathcal{E}|}^{reg}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}_{|\mathcal{E}|}^{reg}}{\partial \mathbf{t}_1} & \dots & \frac{\partial \mathbf{r}_{|\mathcal{E}|}^{reg}}{\partial \mathbf{t}_{|\mathcal{V}|}} \end{bmatrix}$$

where $|\mathcal{V}|$ is the number of graph node. $|\mathcal{K}|$ is number of correspondence. $|\mathcal{E}|$ is the number of graph edge. Each block in \mathbf{J} is a 3×3 matrix. For the sparse nature of this problem, most blocks are zeros. The full residual vector $\mathbf{r} \in \mathbb{R}^{3|\mathcal{K}|+3|\mathcal{E}|}$ is shown as

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}_1^{corr} \\ \vdots \\ \mathbf{r}_{|\mathcal{K}|}^{corr} \\ \hline \mathbf{r}_1^{reg} \\ \vdots \\ \mathbf{r}_{|\mathcal{E}|}^{reg} \end{bmatrix}$$

where each block is a 3×1 vector. The total length is $(|\mathcal{K}| + |\mathcal{E}|) \times 3$.

Non-rigid Optimization. We use Gauss-Newton algorithm and minimizes the total energy function \mathbf{E}_{total} . The Gauss-Newton method is an iterative scheme. In every iteration n , we re-compute the Jacobian matrix \mathbf{J} and the residual vector \mathbf{r} , and get a solution increment $\Delta\mathcal{G}$ by solving the update equations:

$$\mathbf{J}^T \mathbf{J} \Delta\mathcal{G} = \mathbf{J}^T \mathbf{r}$$

The above linear system is solved using LU decomposition.

VII. Qualitative Results

Tab. 9 shows the scores for the elephant and dragon examples from the main paper. Fig. 3 shows the qualitative matching and registration results on 4DMatch. Tab. 10 shows the corresponding scores for results in Fig. 3. Fig. 4 shows the qualitative matching and registration results on 3DLoMatch.

	elephant			dragon		
	EPE↓	Acc5↑	Acc10↑	EPE↓	Acc5↑	Acc10↑
N-ICP	0.166	22.4	41.5	0.325	4.5	17.8
Predator [26] + N-ICP	0.092	55.7	66.0	0.514	29.6	32.1
Ours + N-ICP	0.018	90.6	98.0	0.038	68.0	96.0

Table 9. Quantitative non-rigid registration results. The metrics are 3D end point error (EPE) and motion estimation accuracy (Acc) ($<0.05m$ or 5%, $<0.1m$ or 10%).

	moose			mutant		
	EPE↓	Acc5↑	Acc10↑	EPE↓	Acc5↑	Acc10↑
N-ICP	0.728	0.1	0.7	0.52	0.0	0.6
Predator [26] + N-ICP	0.0283	86.9	99.4	0.217	44.6	60.1
Ours + N-ICP	0.0263	88.5	99.9	0.119	62.6	71.4

Table 10. Quantitative non-rigid registration results. The metrics are 3D end point error (EPE) and motion estimation accuracy (Acc) ($<0.05m$ or 5%, $<0.1m$ or 10%).

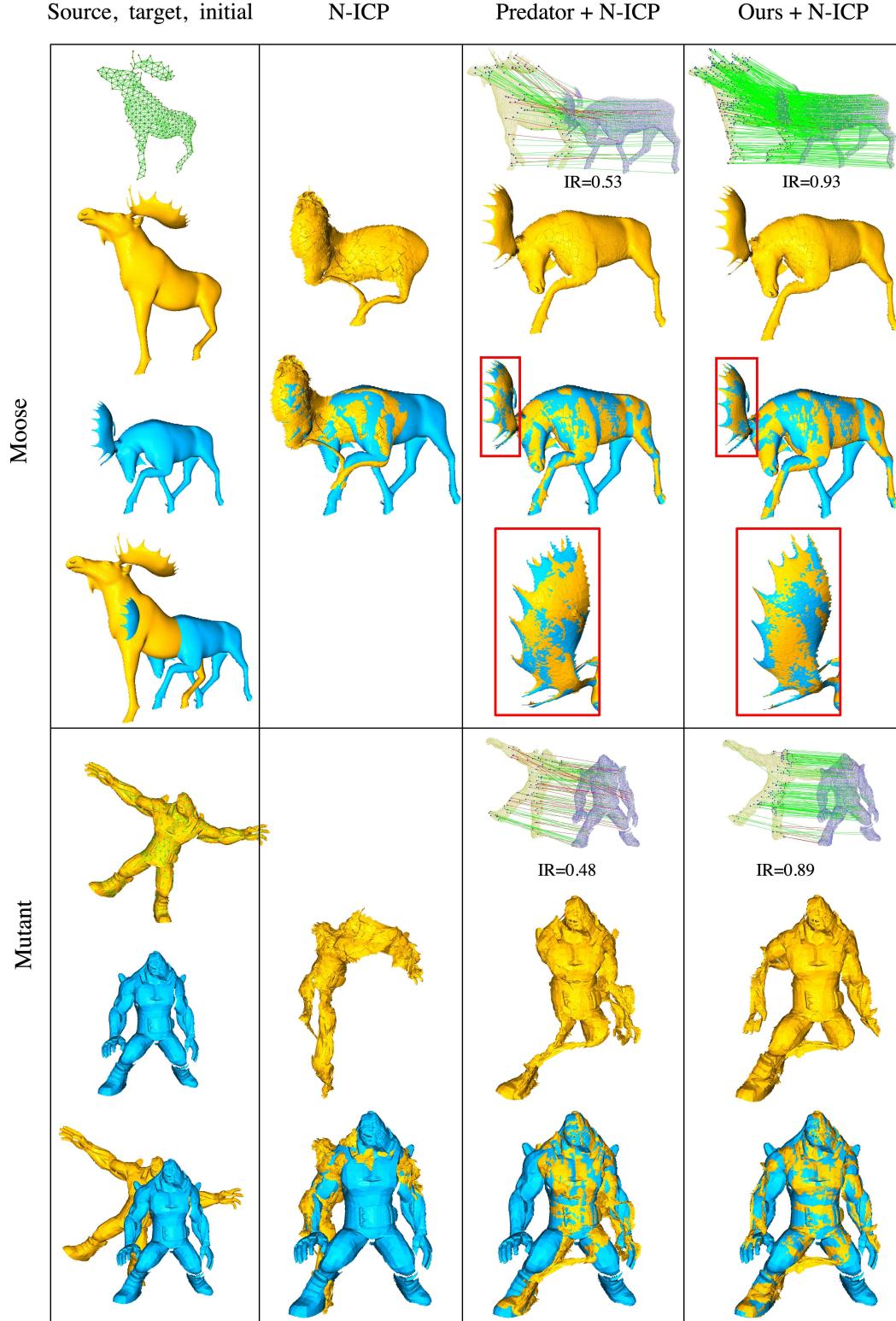


Figure 3. Qualitative point cloud matching and registration results on 4DMatch. The inlier threshold is set to 4cm. The N-ICP-based refinement can remedy outliers to a certain extent if the outlier matches are not too far away from the ground truth (see the results of *Predator + N-ICP* in the Moose example). The N-ICP-based refinement can not handle outliers that connects distant parts. E.g. in the Mutant example, left and right legs are registered together by both methods.

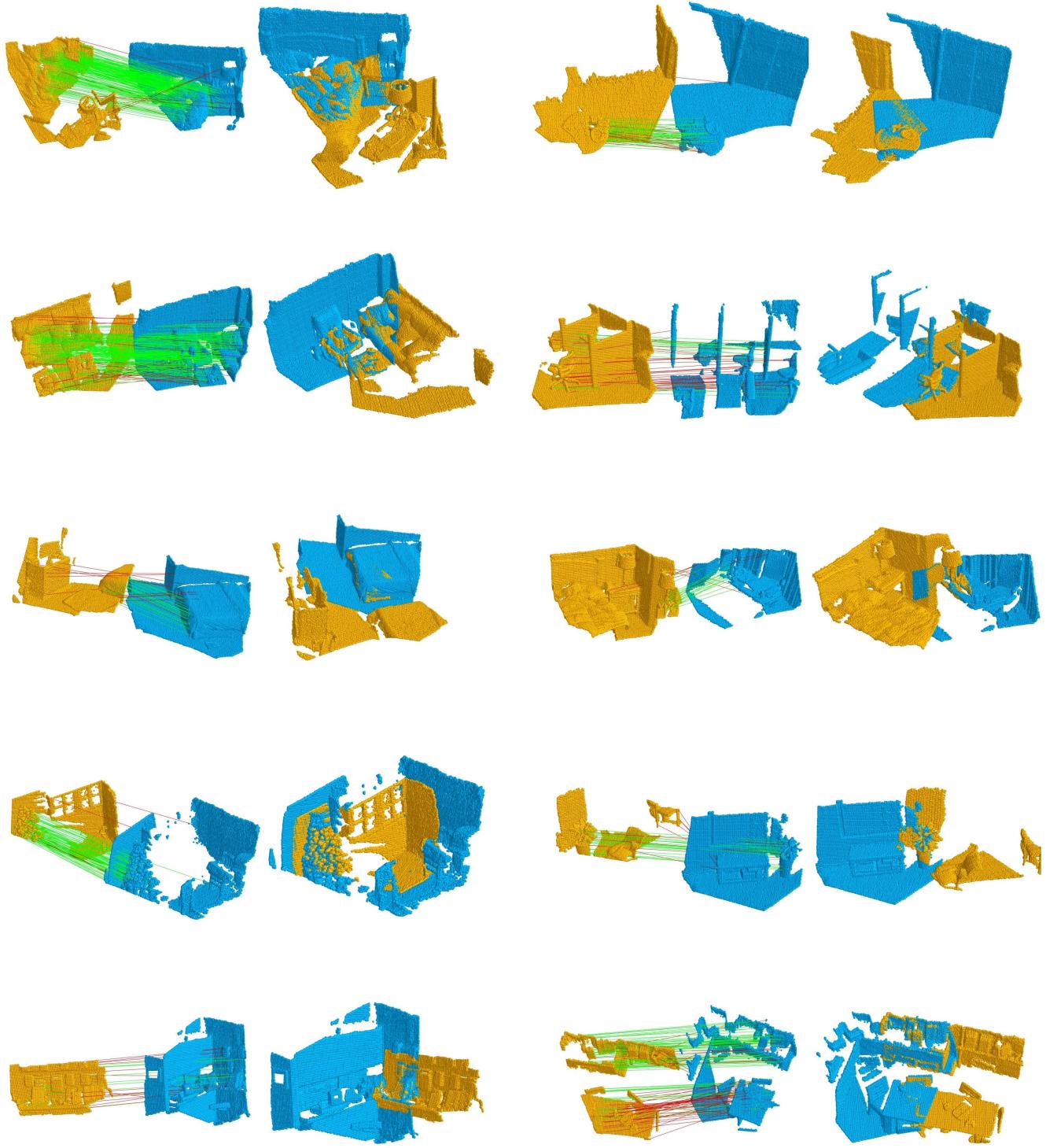


Figure 4. Qualitative point cloud matching and registration results on 3DLoMatch.