Rotation-Invariant Transformer for Point Cloud Matching

Hao Yu[1] Zheng Qin[2] Ji Hou[3] Mahdi Saleh[1,4] Dongsheng Li[2] Benjamin Busam[1,4] Slobodan Ilic[1,5]
[1]TU Munich [2]NUDT [3]Meta Reality Labs [4]3Dwe.ai [5]Siemens AG, Munich

Abstract

*The intrinsic rotation invariance lies at the core of matching point clouds with handcrafted descriptors. However, it is widely despised by recent deep matchers that obtain the rotation invariance extrinsically via data augmentation. As the finite number of augmented rotations can never span the continuous $SO(3)$ space, these methods usually show instability when facing rotations that are rarely seen. To this end, we introduce RoITr, a **Ro**tation-**I**nvariant **Tr**ansformer to cope with the pose variations in the point cloud matching task. We contribute both on the local and global levels. Starting from the local level, we introduce an attention mechanism embedded with Point Pair Feature (PPF)-based coordinates to describe the pose-invariant geometry, upon which a novel attention-based encoder-decoder architecture is constructed. We further propose a global transformer with rotation-invariant cross-frame spatial awareness learned by the self-attention mechanism, which significantly improves the feature distinctiveness and makes the model robust with respect to the low overlap. Experiments are conducted on both the rigid and non-rigid public benchmarks, where RoITr outperforms all the state-of-the-art models by a considerable margin in the low-overlapping scenarios. Especially when the rotations are enlarged on the challenging 3DLoMatch benchmark, RoITr surpasses the existing methods by at least 13 and 5 percentage points in terms of Inlier Ratio and Registration Recall, respectively. Code is publicly available* [1].

1. Introduction

The correspondence estimation between a pair of partially-overlapping point clouds is a long-standing task that lies at the core of many computer vision applications, such as tracking [17, 18], reconstruction [22, 23, 39], pose estimation [19, 27, 50] and 3D representation learning [15, 16, 46], etc. In a typical solution, geometry is first encoded into descriptors, and correspondences are then established between two frames by matching the most similar descriptors. As the two frames are observed from different views, depicting the same geometry under different transformations identically, *i.e.*, the pose-invariance, becomes the key to success in the point cloud matching task.

[1]https://github.com/haoyu94/RoITr

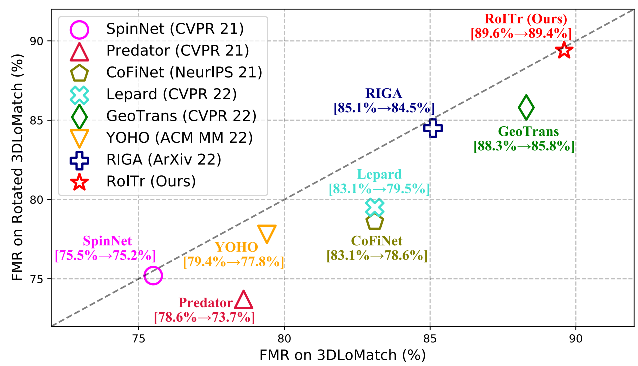


Figure 1. *Feature Matching Recall* (FMR) on 3DLoMatch [19] and Rotated 3DLoMatch. Distance to the diagonal represents the robustness against rotations. Among all the state-of-the-art approaches, RoITr not only ranks first on both benchmarks but also shows the best robustness against the enlarged rotations.

Since the side effects caused by a global translation can always be easily eliminated, *e.g.*, by aligning the barycenter with the origin, the attention naturally shifts to coping with the rotations. In the past, handcrafted local descriptors [11, 30, 31, 41] were designed to be rotation-invariant so that the same geometry observed from different views can be correctly matched. With the emergence of deep neural models for 3D point analysis, *e.g.*, multilayer perceptrons (MLPs)-based like PointNet [25, 26], convolutions-based like KPConv [6, 40], and the attention-based like PointTransformer [33, 53], recent approaches [1, 7, 9, 10, 13, 19, 20, 27, 32, 48–51] propose to learn descriptors from raw points as an alternative to handcrafted features that are less robust to occlusion and noise. The majority of deep point matchers [7, 10, 19, 20, 27, 34, 48, 50–52] is sensitive to rotations. Consequently, their invariance to rotations must be obtained extrinsically via augmented training to ensure that the same geometry under different poses can be depicted similarly. However, as the training cases can never span the continuous $SO(3)$ space, they always suffer from instability when facing rotations that are rarely seen during training. This can be observed by a significant performance drop under enlarged rotations at inference time. (See Fig. 1.)

There are other works [1, 9, 13, 32, 44] that only leverage deep neural networks to encode the pure geometry with the intrinsically-designed rotation invariance. However, the intrinsic rotation invariance comes at the cost of losing global context. For example, a human's left and right halves are almost identically described, which naturally degrades the distinctiveness of features. Most recently, RIGA [49] is proposed to enhance the distinctiveness of the rotation-invariant descriptors by incorporating a global context, *e.g.*, the left and right halves of a human become distinguishable by knowing there is a chair on the left while a table on the right. However, it lacks a highly-representative geometry encoder since it relies on PointNet [25], which accounts for an ineffective local geometry description. Moreover, as depicting the cross-frame spatial relationships is non-trivial, previous works [19, 27, 34, 50] merely leverage the contextual features in the cross-frame context aggregation, which neglects the positional information. Although RIGA proposes to learn a rotation-invariant position representation by leveraging an additional PointNet, this simple design is hard to model the complex cross-frame positional relationships and leads to less distinctive descriptors.

In this paper, we present **Ro**tation-**I**nvariant **Tr**ansformer (RoITr) to tackle the problem of point cloud matching under arbitrary pose variations. By using Point Pair Features (PPFs) as the local coordinates, we propose an attention mechanism to learn the pure geometry regardless of the varying poses. Upon it, attention-based layers are further proposed to compose the encoder-decoder architecture for highly-discriminative and rotation-invariant geometry encoding. We demonstrate its superiority over Point-Transformer [53], a state-of-the-art attention-based backbone network, in terms of both efficiency and efficacy in Fig. 8 and Tab. 4 (a), respectively. On the global level, the cross-frame position awareness is introduced in a rotation-invariant fashion to facilitate feature distinctiveness. We illustrate its significance over the state-of-the-art design [27] in Tab. 4 (d). Our main contributions are summarized as:

- An attention mechanism designed to disentangle the geometry and poses, which enables the pose-agnostic geometry description.
- An attention-based encoder-decoder architecture that learns highly-representative local geometry in a rotation-invariant fashion.
- A global transformer with rotation-invariant cross-frame position awareness that significantly enhances the feature distinctiveness.

2. Related Work

Models with Extrinsic Rotation Invariance. The mainstream of deep learning-based point cloud matching approaches is intrinsically rotation-sensitive. Pioneers [10, 51] learn to describe local patches from a rotation-variant input. FCGF [7] leverages fully-convolutional networks to accelerate the geometry description. D3Feat [3] jointly detects and describes sparse keypoints for matching. Predator [19] incorporates the global context to enhance the local descriptors and predicts the overlap regions for keypoint sampling. CoFiNet [50] extracts coarse-to-fine correspondences to alleviate the repeatability issue of keypoints. GeoTrans [27] considers the geometric information in fusing the intra-frame context globally. However, the awareness of spatial positions is missing in the cross-frame aggregation. Lepard [20] extends the non-rigid shape matching [34, 38, 42] to point clouds [28] and proposes a re-positioning module to alleviate the pose variations. REGTR [48] directly regresses the corresponding coordinates and registers point clouds in an end-to-end fashion. Nonetheless, all of these methods suffer from instability with additional rotations.

Methods with Intrinsic Rotation Invariance. A branch of handcrafted descriptors [8, 14, 41] aligns the input to a canonical representation according to an estimated local reference frame (LRF), while the others [11, 30, 31] mine the rotation-invariant components and encode them as the representation of the local geometry. Inspired by that, some deep learning-based methods [1, 4, 9, 13, 32, 34, 49] are designed to be intrinsically rotation-invariant to make the neural models focus on the pose-agnostic pure geometry. As a pioneer, PPF-FoldNet [9] consumes PPF-based patches and learns the descriptors using a FoldingNet [47]-based architecture without supervision. LRF-based works [1,13,32,34] achieve rotation invariance by aligning their input to the defined canonical representation. YOHO [44] adopts a group of rotations to learn a rotation-equivariant feature group and further obtain the invariance via group pooling. A common problem of the rotation-invariant methods is the less distinctive features. Although RIGA [49] incorporates the global context into local descriptors to enhance the feature distinctiveness, its ineffective local geometry encoding and global position description learned by PointNet [25] still constraint the representation ability of its descriptors.

3. Method

Problem Statement. We tackle the problem of matching a pair of partially-overlapping point clouds $\mathbf{P} \in \mathbb{R}^{n\times 3}$ and $\mathbf{Q} \in \mathbb{R}^{m\times 3}$, and extract a correspondence set $\hat{\mathcal{C}} = \{(\hat{\mathbf{p}}_i, \hat{\mathbf{q}}_j) \big| \hat{\mathbf{p}}_i \in \hat{\mathbf{P}} \subseteq \mathbf{P}, \hat{\mathbf{q}}_j \in \hat{\mathbf{Q}} \subseteq \mathbf{Q}\}$ that minimizes:

$$\frac{1}{|\hat{\mathcal{C}}|}\sum_{(\hat{\mathbf{p}}_i,\hat{\mathbf{q}}_j)\in\hat{\mathcal{C}}} \|\mathcal{M}^*(\hat{\mathbf{p}}_i) - \hat{\mathbf{q}}_j\|_2, \tag{1}$$

where $\|\cdot\|_2$ denotes the Euclidean norm and $|\cdot|$ is the set cardinality. $\mathcal{M}^*(\cdot)$ stands for the ground-truth mapping function that maps $\hat{\mathbf{p}}_i$ to its corresponding position in $\hat{\mathbf{Q}}$. In rigid scenarios, it is defined by a transformation $\mathbf{T}^* \in SE(3)$. For the non-rigid cases it can be denoted as a per-point flow $\mathbf{f}_i^* \in \mathbb{R}^3$ known as the deformation field.

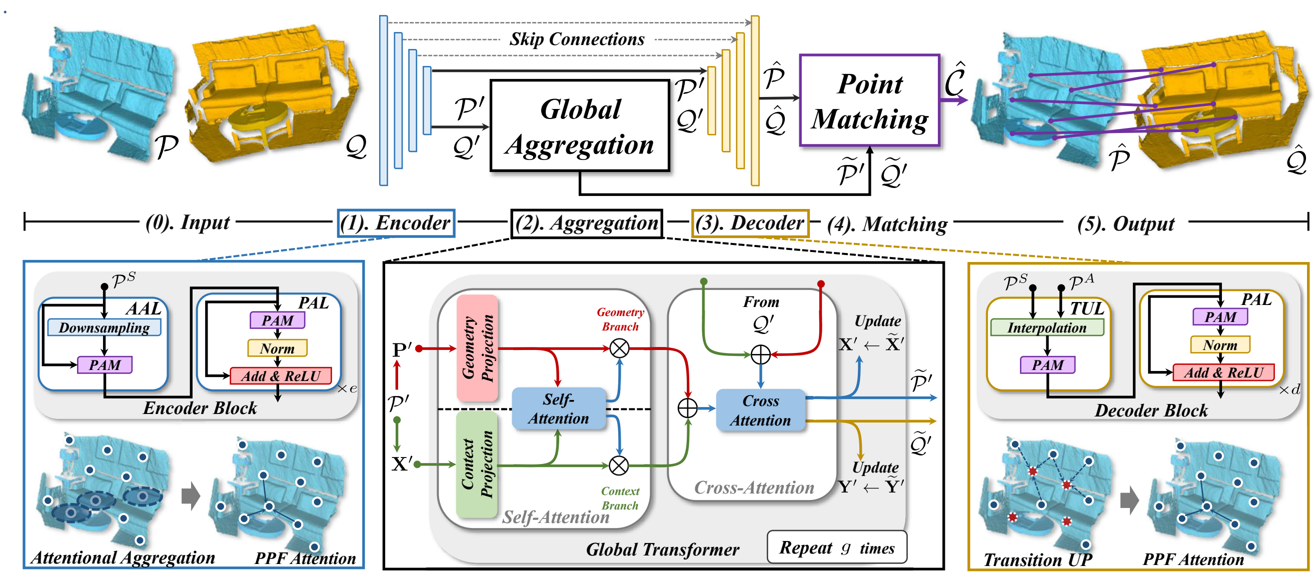
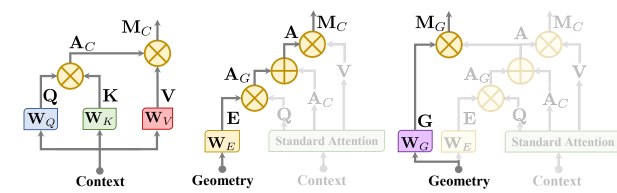


Figure 2. **An Overview of RoITr.** From left to right: ***(0).*** RoITr takes as input a pair of triplets $\mathcal{P} = (\mathbf{P}, \mathbf{N}, \mathbf{X})$ and $\mathcal{Q} = (\mathbf{Q}, \mathbf{M}, \mathbf{Y})$, each with three dimensions referring to the point cloud, the estimated normals, and the initial features. ***(1).***[§. 3.2] A stack of encoder blocks hierarchically downsamples the points to coarser superpoints and encodes the local geometry, yielding superpoint triplets $\mathcal{P}'$ and $\mathcal{Q}'$. Each encoder block consists of an Attentional Abstraction Layer (AAL) for downsampling and abstraction, followed by $e\times$ PPF Attention Layers (PALs) for local geometry encoding and context aggregation. Both of them are based on our proposed PPF Attention Mechanism (PAM), which enables the pose-agnostic encoding of pure geometry. (See Fig. 3 and Fig. 4). ***(2).***[§. 3.3] Global information is fused to enhance the superpoint features of $\mathcal{P}'$ and $\mathcal{Q}'$. The geometric cues are globally aggregated as a rotation-invariant position representation, which introduces spatial awareness in the consecutive cross-frame context aggregation. After a stack of $g\times$ global transformers, the globally-enhanced triplets $\widetilde{\mathcal{P}}'$ and $\widetilde{\mathcal{Q}}'$ are produced. ***(3).***[§. 3.2] Superpoint triplets $\mathcal{P}'$ and $\mathcal{Q}'$ are decoded to point triplets $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$ by a stack of decoder blocks. Each block consists of a Transition Up Layer (TUL) for upsampling and context aggregation, followed by $d\times$ PALs. ***(4).***[§. 3.4] By adopting the coarse-to-fine matching [50], $\widetilde{\mathcal{P}}'$ and $\widetilde{\mathcal{Q}}'$ are matched to generate superpoint correspondences, which are consecutively refined to point correspondences between $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$. ***(5).*** $\hat{\mathcal{C}}$ is established between $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$.

Method Overview. An overview of RoITr is shown in Fig. 2. RoITr consists of an encoder-decoder architecture named Point Pair Feature Transformer (PPFTrans) for local geometry encoding and a stack of $g\times$ global transformers for global context aggregation. Correspondence set $\hat{\mathcal{C}}$ is extracted by the coarse-to-fine matching [50].

3.1. PPF Attention Mechanism

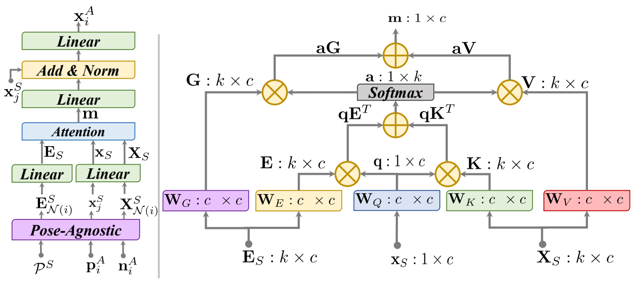
Overview. Fig. 3 compares three different self-attention mechanisms. The standard attention [43] only leverages the input context to obtain the *Query* $\mathbf{Q}$ and *Key* $\mathbf{K}$ to compute the contextual attention $\mathbf{A}_C$, as well as the *Value* $\mathbf{V}$ that encodes information for the contextual message $\mathbf{M}_C$. GeoTrans [27] proposes to learn the positional encoding $\mathbf{E}$ from the geometry and calculates a second attention $\mathbf{A}_G$ to reweigh $\mathbf{A}_C$. However, the cues contained in the raw geometry are totally neglected. To this end, we propose to learn the pose-agnostic geometric cues $\mathbf{G}$ and further generate the geometric message $\mathbf{M}_G$ in the PPF Attention Mechanism (PAM). On the local level, $\mathbf{M}_G$ is combined with $\mathbf{M}_C$ for feature enhancement, while on the global level, it is used to learn the rotation-invariant position representation for the cross-frame context aggregation. More specifically, we define PAM on an *Anchor* triplet $\mathcal{P}^A = (\mathbf{P}^A, \mathbf{N}^A, \mathbf{X}^A)$ and a *Support* triplet $\mathcal{P}^S = (\mathbf{P}^S, \mathbf{N}^S, \mathbf{X}^S)$, both with three dimensions referring to the point cloud, the estimated normals, and the associated features, respectively. PAM aggregates the learned context and geometric cues from $\mathcal{P}^S$ and flows the messages to $\mathcal{P}^A$.



(a). Standard Attention (b). GeoTrans (c). PAM (Ours)

Figure 3. Illustration of different self-attention computation in the standard attention [43], GeoTrans [27], and PAM.

Pose-Agnostic Coordinate Representation. The basis of PAM is the pose-agnostic local coordinate representation that we construct based on PPFs [11]. Let $\mathcal{P}_i^A := (\mathbf{p}_i^A \in \mathbf{P}^A, \mathbf{n}_i^A \in \mathbf{N}^A, \mathbf{x}_i^A \in \mathbf{X}^A) \in \mathcal{P}^A$ denote the triplet constructed by picking the i^{th} item on each dimension. For each $\mathbf{p}_i^A$, a subset of $\mathcal{P}^S$ is first retrieved according to the Euclidean distance w.r.t. $\mathbf{P}^S$, denoted as



(a). PPF Attention (b). Detailed Attention Calculation

Figure 4. **Left**: The workflow of the PPF Attention Mechanism (PAM). **Right**: Detailed calculation of the attention.

$\mathcal{P}^S_{\mathcal{N}(i)} := (\mathbf{P}^S_{\mathcal{N}(i)}, \mathbf{N}^S_{\mathcal{N}(i)}, \mathbf{X}^S_{\mathcal{N}(i)}) \subseteq \mathcal{P}^S$, with $\mathcal{N}(i)$ the indices of k-nearest neighbors. We then adopt PPFs [11] to construct a local coordinate system around each $\mathbf{p}^A_i$ to represent the pose-agnostic position of $\mathbf{P}^S_{\mathcal{N}(i)}$ w.r.t. it. The coordinate of point $\mathbf{p}^S_j \in \mathbf{P}^S_{\mathcal{N}(i)}$ is transferred to:

$$\mathbf{e}^S_j = (\|\mathbf{d}\|_2, \angle(\mathbf{n}^A_i, \mathbf{d}), \angle(\mathbf{n}^S_j, \mathbf{d}), \angle(\mathbf{n}^S_j, \mathbf{n}^A_i)), \quad (2)$$

with $\mathbf{d} = \mathbf{p}^S_j - \mathbf{p}^A_i$, and $\mathbf{n}^A_i$ and $\mathbf{n}^S_j$ the estimated normals of $\mathbf{p}^A_i$ and $\mathbf{p}^S_j$, respectively. $\angle(\mathbf{v}_1, \mathbf{v}_2)$ computes the angles between the two vectors [5, 9]. The transferred coordinates of $\mathbf{P}^S_{\mathcal{N}(i)}$ are denoted as $\mathbf{E}^S_{\mathcal{N}(i)}$.

PPF Attention Mechanism. PPF Attention Mechanism (PAM) takes as input the *Support* triplet $\mathcal{P}^S$ and the *Anchor* point cloud $\mathbf{P}^A$ with estimated normals $\mathbf{N}^A$. PAM generates the *Anchor* features $\mathbf{X}^A$ by aggregating the pose-agnostic local geometry and highly-representative learned context from $\mathcal{P}^S$, which is defined as:

$$\mathcal{P}^A = \delta(\mathbf{P}^A, \mathbf{N}^A | \mathcal{P}^S), \quad (3)$$

with $\delta(\cdot)$ representing PAM. As shown in Fig. 4 (a), for each $\mathbf{p}^A_i \in \mathbf{P}^A$ with normal $\mathbf{n}^A_i$, we find its nearest point $\mathbf{p}^S_j \in \mathbf{P}^S$ whose associated feature $\mathbf{x}^S_j$ is assigned to $\mathbf{p}^A_i$ as the initial description. Then, k-nearest neighbors from $\mathbf{P}^S$ are retrieved according to the Euclidean distance in 3D space, yielding $\mathbf{P}^S_{\mathcal{N}(i)} \subseteq \mathbf{P}^S$ and $\mathbf{X}^S_{\mathcal{N}(i)} \subseteq \mathbf{X}^S$. Following Eq. 2, $\mathbf{P}^S_{\mathcal{N}(i)}$ is transferred to the pose-agnostic position representation $\mathbf{E}^S_{\mathcal{N}(i)}$, which is consecutively projected to the coordinate embedding $\mathbf{E}_S$ via a linear layer. $\mathbf{x}^S_j$ and $\mathbf{X}^S_{\mathcal{N}(i)}$ are projected to the contextual features $\mathbf{x}_S$ and $\mathbf{X}_S$ by a second shared linear layer, respectively. In Fig. 4 (b), the attention mechanism uses five learnable matrices $\mathbf{W}_G$, $\mathbf{W}_E$, $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ to project the input. Specifically, $\mathbf{W}_G$ and $\mathbf{W}_E$ project the input coordinate representation to the geometric cues and positional encoding by:

$$\mathbf{G} = \mathbf{E}_S \mathbf{W}_G \quad \text{and} \quad \mathbf{E} = \mathbf{E}_S \mathbf{W}_E, \quad (4)$$

respectively. Similarly, $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ project the learned context to *Query*, *Key*, and *Value* as:

$$\mathbf{q} = \mathbf{x}_S \mathbf{W}_Q, \ \mathbf{K} = \mathbf{X}_S \mathbf{W}_K, \text{ and } \mathbf{V} = \mathbf{X}_S \mathbf{W}_V, \quad (5)$$

respectively. The attention $\mathbf{a}$ that measures the feature similarity, and the message $\mathbf{m}$ that encodes both the pose-agnostic geometry and the representative context read as:

$$\mathbf{a} = \text{Softmax}(\frac{\mathbf{qE}^T + \mathbf{qK}^T}{\sqrt{c_0}}) \quad \text{and} \quad \mathbf{m} = \mathbf{aG} + \mathbf{aV}, \quad (6)$$

respectively. The message $\mathbf{m}$ is projected and aggregated to $\mathbf{x}^S_j$ via an element-wise addition followed by a normalization through LayerNorm [2]. The final linear layer projects the obtained feature to $\mathbf{x}^A_i$, from which $\mathbf{X}^A$ is obtained to formulate the output $\mathcal{P}^A$ with the known $\mathbf{P}^A$ and $\mathbf{N}^A$.

3.2. PPFTrans for Local Geometry Description

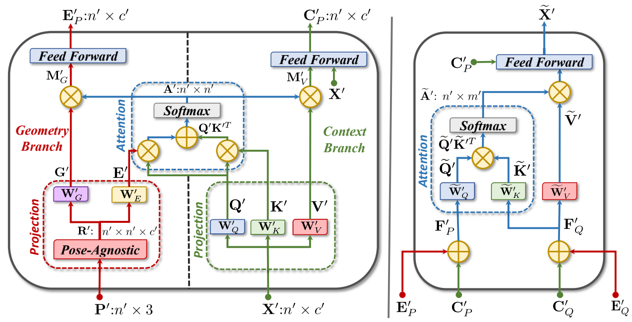
Overview. As illustrated in Fig. 2, PPFTrans consumes triplets $\mathcal{P}$ and $\mathcal{Q}$. Taking $\mathcal{P} = (\mathbf{P}, \mathbf{N}, \mathbf{X})$ as an example, it consists of $\mathbf{P} \in \mathbb{R}^{n \times 3}$ the points cloud, $\mathbf{N} \in \mathbb{R}^{n \times 3}$ the normals estimated from $\mathbf{P}$, and $\mathbf{X} = \vec{\mathbf{1}} \in \mathbb{R}^{n \times 1}$ the initial point features. The encoder produces the superpoint triplet $\mathcal{P}' = (\mathbf{P}', \mathbf{N}', \mathbf{X}')$ with $\mathbf{P}' \in \mathbb{R}^{n' \times 3}$ and $\mathbf{X}' \in \mathbb{R}^{n' \times c'}$. With the consecutive decoder, $\mathcal{P}'$ is decoded to a triplet $\hat{\mathcal{P}} = (\hat{\mathbf{P}}, \hat{\mathbf{N}}, \hat{\mathbf{X}})$ including $\hat{n}$ points with features $\hat{\mathbf{X}} \in \mathbb{R}^{\hat{n} \times \hat{c}}$. Notably, as we adopt a Farthest Point Sampling (FPS) strategy [26], it always satisfies that $\mathbf{P}' \subseteq \hat{\mathbf{P}} \subseteq \mathbf{P}$. The same goes for a second point cloud $\mathbf{Q}$ with an input triplet $\mathcal{Q} = (\mathbf{Q} \in \mathbb{R}^{m \times 3}, \mathbf{M} \in \mathbb{R}^{m \times 3}, \mathbf{Y} = \vec{\mathbf{1}} \in \mathbb{R}^{m \times 1})$ by the shared architecture. In the rest of this paper, we only demonstrate for $\mathcal{P}$ unless the model processes $\mathcal{Q}$ differently.

Encoder. The encoder is constructed by stacking several encoder blocks, each including an Attentional Abstraction Layer (AAL) followed by $e\times$PPF Attention Layers (PALs). Each block consumes the output of the previous block as the *Support* triplet $\mathcal{P}^S$ ($\mathcal{P}^S = \mathcal{P}$ for the first block). $\mathcal{P}^S$ first flows to AAL, where *Anchor* points $\mathbf{P}^A$ with associated normals $\mathbf{N}^A$ are obtained via FPS [26]. The *Anchor* triplet $\mathcal{P}^A$ is then generated in AAL via a PAM following Eq. 3. A sequence of PALs is applied for enhancing the *Anchor* features $\mathbf{X}^A$, each updating the features as:

$$\mathcal{P}^A \leftarrow \theta(\mathcal{P}^A) = \text{ReLU}(\mathbf{X}^A + \phi(\delta(\mathbf{P}^A, \mathbf{N}^A | \mathcal{P}^A))), \quad (7)$$

with ϕ the LayerNorm [2], δ the PAM, and θ the PAL. $\leftarrow$ depicts feature updating. The encoder block outputs the updated $\mathcal{P}^A$, and the output of the whole encoder is defined as $\mathcal{P}'$, which is the output of the final encoder block.

Decoder. We build the decoder by stacking a series of decoder blocks, each consisting of a Transition Up Layer (TUL) followed by $d\times$ PAL. Each block takes the output of the previous block as the *Anchor* triplet $\mathcal{P}^A$ ($\mathcal{P}^A = \mathcal{P}'$ for the first block), and takes the *Support*



(a). Geometry-Aware Self-Attention (b). Position-Aware Cross-Attention

Figure 5. The computation graph of our global transformer consisting of the Geometry-Aware Self-Attention Module (GSM) and Position-Aware Cross-Attention Module (PCM).

triplet $\mathcal{P}^S$ from the encoder via skip connections. The input flows to TUL, where each feature $\widetilde{\mathbf{x}}_j^S \in \widetilde{\mathbf{X}}^S$ assigned to $\mathbf{p}_j^S \in \mathbf{P}^S$ is interpolated by:

$$\widetilde{\mathbf{x}}_j^S = \frac{\sum_{i \in \mathcal{N}(j)} w_i^j \mathbf{x}_i^A}{\sum_{i \in \mathcal{N}(j)} w_i^j}, \text{ with } w_i^j = \frac{1}{\|\mathbf{p}_j^S - \mathbf{p}_i^A\|_2}, \tag{8}$$

with $\mathcal{N}(j)$ the k-nearest neighbors of $\mathbf{p}_j^S$ in $\mathbf{P}^A$. Features are updated by two linear layers as $\mathcal{P}^S \leftarrow \zeta_1(\mathbf{X}^S) + \zeta_2(\widetilde{\mathbf{X}}^S)$. A sequence of PALs is adopted after TUL, each enhancing the features as $\mathcal{P}^S \leftarrow \theta(\mathcal{P}^S)$ according to Eq. 7. The decoder block outputs the updated $\mathcal{P}^S$, and the output of the whole decoder is denoted as $\hat{\mathcal{P}}$, which is the output of the final decoder block.

3.3. Global Transformer for Context Aggregation

Overview. Our designed global transformer takes as input a pair of triplets $\mathcal{P}'$ and $\mathcal{Q}'$, and enhances the features with the global context, yielding $\widetilde{\mathbf{X}}' \in \mathbb{R}^{n' \times c'}$ and $\widetilde{\mathbf{Y}}' \in \mathbb{R}^{m' \times c'}$, respectively. We stack $g\times$global transformers, with each including a Geometry-Aware Self-Attention Module (GSM) and a Position-Aware Cross-Attention Module (PCM) (See Fig. 2 and Fig. 5). Different from previous works [19, 27, 48–50] that totally neglect the cross-frame spatial relationships, we propose to learn a rotation-invariant position representation for each superpoint to enable the position-aware cross-frame context aggregation.

Geometry-Aware Self-Attention Module. On the global level, we modify PAM to learn the rotation-invariant position representation and to aggregate the learned context across the whole frame simultaneously. The design of GSM is detailed in Fig. 5 (a). GSM has two branches, where the geometry branch mines the geometric cues from the pairwise rotation-invariant geometry representation proposed in [27], and the context branch aggregates the global context across the frame. We refer the readers to the Appendix for the detailed construction of $\mathbf{R}' \in \mathbb{R}^{n' \times n' \times c'}$ and the ablation study on it. Similar to Eq. 4, the geometric cues $\mathbf{G}'$ and the positional encoding $\mathbf{E}'$ are linearly projected from $\mathbf{R}'$. $\mathbf{E}'$ is further processed in the geometry branch and finally leveraged as the rotation-invariant position representation. In the context branch, $\mathbf{Q}'$, $\mathbf{K}'$, and $\mathbf{V}'$ are obtained by linearly mapping the input features $\mathbf{X}'$ similar to Eq. 5. The hybrid score matrix $\mathbf{S}' \in \mathbb{R}^{n' \times n'}$ is computed as:

$$\mathbf{S}'(i,j) = \frac{(\mathbf{q}'_i)(\mathbf{e}'_{i,j} + \mathbf{k}'_j)^T}{\sqrt{c'}}, \tag{9}$$

with $\mathbf{e}'_{i,j} := \mathbf{E}'(i,j,:)$, $\mathbf{q}'_i := \mathbf{Q}'(i,:)$, and $\mathbf{k}'_j := \mathbf{K}'(j,:)$ the c'-dimension vectors. The hybrid attention $\mathbf{A}'$ is obtained via a Softmax function over each row of $\mathbf{S}'$, and the geometric messages $\mathbf{M}'_G \in \mathbb{R}^{n' \times c'}$ are computed as:

$$\mathbf{M}'_G(i,:) = \sum_{1 \le j \le n'} a'_{i,j} \mathbf{g}'_{i,j}, \tag{10}$$

with $a'_{i,j} := \mathbf{A}'(i,j)$ and $\mathbf{g}'_{i,j} := \mathbf{G}'(i,j,:)$. The contextual messages $\mathbf{M}'_V \in \mathbb{R}^{n' \times c'}$ are computed by $\mathbf{A}'\mathbf{V}'$. After a feed-forward network [43], the position representation $\mathbf{E}'_P$ and globally-enhanced context $\mathbf{C}'_P$ are generated.

Position-Aware Cross-Attention Module. PCM consumes a pair of doublets $(\mathbf{E}'_P, \mathbf{C}'_P)$ and $(\mathbf{E}'_Q, \mathbf{C}'_Q)$ that are generated from $\mathcal{P}'$ and $\mathcal{Q}'$ by a shared GSM, respectively. As the cross-attention is directional, we apply the same PCM twice, with the first aggregation from $\mathcal{Q}'$ to $\mathcal{P}'$ (See Fig. 5 (b)), and the second reversed. As the first step, the rotation-invariant position representation is incorporated to make the consecutive cross-attention position-aware, yielding position-aware features $\mathbf{F}'_P = \mathbf{E}'_P + \mathbf{C}'_P$ and $\mathbf{F}'_Q = \mathbf{E}'_Q + \mathbf{C}'_Q$. Similar to Eq. 5, $\widetilde{\mathbf{Q}}'$, $\widetilde{\mathbf{K}}'$, and $\widetilde{\mathbf{V}}'$ are computed as the linear projection of $\mathbf{F}'_P$, $\mathbf{F}'_Q$, and $\mathbf{F}'_Q$, respectively. The attention matrix $\widetilde{\mathbf{A}} \in \mathbb{R}^{n' \times m'}$ is computed via a row-wise softmax function applied on $\widetilde{\mathbf{Q}}'\widetilde{\mathbf{K}}'^T$. The fused messages are presented as $\widetilde{\mathbf{A}}\widetilde{\mathbf{V}}'$, which are finally mapped to the output features $\widetilde{\mathbf{X}}'$ through a feed-forward network. As we introduce spatial awareness at the beginning of PCM, both the attention computation and message fusion are aware of the cross-frame positions. After the twice application of PCM, the input features are enhanced as $\mathcal{P}' \leftarrow \widetilde{\mathbf{X}}'$ and $\mathcal{Q}' \leftarrow \widetilde{\mathbf{Y}}'$, respectively. The global aggregation stage finally generates a pair of triplets $\widetilde{\mathcal{P}}' := (\mathbf{P}', \mathbf{N}', \widetilde{\mathbf{X}}')$ and $\widetilde{\mathcal{Q}}' := (\mathbf{Q}', \mathbf{M}', \widetilde{\mathbf{Y}}')$, with the enhanced features from the last global transformer.

3.4. Point Matching and Loss Funcion

Superpoint Matching. As shown in Fig. 2, the point matching stage consumes a pair of superpoint triplets $\widetilde{\mathcal{P}}'$ and $\widetilde{\mathcal{Q}}'$ obtained from the global transformer, as well as a pair of point triplets $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$ produced by the decoder. We adopt the coarse-to-fine matching proposed in [50]. Following [27], we first normalize the superpoint features $\widetilde{\mathbf{X}}'$ and $\widetilde{\mathbf{Y}}'$ onto a unit hypersphere, and measure the pairwise similarity using a Gaussian correlation matrix $\widetilde{\mathbf{S}}$ with $\widetilde{\mathbf{S}}(i,j) =$

$-\exp(-\|\widetilde{\mathbf{x}}'_i-\widetilde{\mathbf{y}}'_j\|_2^2)$. After a dual-normalization [27,29,37] on $\widetilde{\mathbf{S}}$ for global feature correlation, superpoints associated to the top-k entries are selected as the coarse correspondence set $\mathcal{C}'=\{(\mathbf{p}'_i,\mathbf{q}'_j)|\mathbf{p}'_i\in\mathbf{P}',\mathbf{q}'_j\in\mathbf{Q}'\}$.

Point Matching. For extracting point correspondences, denser points $\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$ are first assigned to superpoints. To this end, the point-to-node strategy [50] is leveraged, where each point is assigned to its closest superpoint in 3D space. Given a superpoint $\mathbf{p}'_i\in\mathbf{P}'$, the group of points assigned to it is denoted as $\hat{\mathbf{G}}_i^P\subseteq\hat{\mathbf{P}}$. The group of features associated to $\hat{\mathbf{G}}_i^P$ is further defined as $\hat{\mathbf{G}}_i^X$ with $\hat{\mathbf{G}}_i^X\subseteq\hat{X}$. For each superpoint correspondence $\mathcal{C}'_l=(\mathbf{p}'_i,\mathbf{q}'_j)$, the similarity between the corresponding feature groups $\hat{\mathbf{G}}_i^X$ and $\hat{\mathbf{G}}_j^Y$ is calculated as $\hat{\mathbf{S}}_l=\hat{\mathbf{G}}_i^X(\hat{\mathbf{G}}_j^Y)^T/\sqrt{\hat{c}}$, with $\hat{c}$ the feature dimension. We then follow [35] to append a stack row and column to $\hat{\mathbf{S}}_l$ filled with a learnable parameter α, and iteratively run the Sinkhorn Algorithm [36]. After removing the slack row and column of $\hat{\mathbf{S}}_l$, the mutual top-k entries, *i.e.*, entries with top-k confidence on both the row and the column, are selected to formulate a point correspondence set $\hat{\mathcal{C}}_l$. The final correspondence set $\hat{\mathcal{C}}$ is collected by $\hat{\mathcal{C}}=\cup_{l=1}^{|\mathcal{C}'|}\hat{\mathcal{C}}_l$.

Loss Function. Our loss function reads as $\mathcal{L}=\mathcal{L}_s+\lambda\mathcal{L}_p$, with a superpoint matching loss $\mathcal{L}_s$ and a point matching loss $\mathcal{L}_p$ balanced by a hyper-parameter λ ($\lambda=1$ by default). The detailed definition is introduced in the Appendix.

4. Experiment

We evaluate RoITr on both rigid (3DMatch [51] & 3DLoMatch [19]) and non-rigid (4DMatch [20] & 4DLoMatch [20]) benchmarks. For the rigid matching, we further evaluate our correspondences on the registration task, where RANSAC [12] is used. Details of implementation, metrics, and runtime analysis are introduced in the Appendix.

4.1. Rigid Indoor Scenes: 3DMatch & 3DLoMatch

Dataset. 3DMatch [51] collects 62 indoor scenes, among which 46 are used for training, 8 for validation, and 8 for testing. We use the data processed by [19] where the 3DMatch data is spilt as 3DMatch (> 30% overlap) and 3DLoMatch (10% $\sim$ 30% overlap). To evaluate robustness to arbitrary rotations, we follow [49] for creating the rotated benchmarks, where full-range rotations are individually added to the two frames of each point cloud pair.

Metrics. We follow [19] to use three metrics for evaluation: (1). *Inlier Ratio* (IR) that computes the ratio of putative correspondences whose residual distance is smaller than a threshold (*i.e.*, 0.1m) under the ground-truth transformation; (2). *Feature Matching Recall* (FMR) that calculates the fraction of point cloud pairs whose IR is larger than a threshold (*i.e.*, 5%); (3). *Registration Recall* (RR) that counts the fraction of point cloud pairs that are correctly registered (*i.e.*, with RMSE < 0.2m). [2]

# Samples=5,000	3DMatch Origin	3DMatch Rotated	3DLoMatch Origin	3DLoMatch Rotated
		Feature Matching Recall (%) ↑		
SpinNet [1]	97.4	97.4	75.5	75.2
Predator [19]	96.6	96.2	78.6	73.7
CoFiNet [50]	<u>98.1</u>	97.4	83.1	78.6
YOHO [44]	**98.2**	97.8	79.4	77.8
RIGA [49]	97.9	**98.2**	85.1	84.5
Lepard [20]	98.0	97.4	83.1	79.5
GeoTrans [27]	97.9	97.8	<u>88.3</u>	<u>85.8</u>
RoITr (*Ours*)	98.0	**98.2**	**89.6**	**89.4**
		Inlier Ratio (%) ↑		
SpinNet [1]	48.5	48.7	25.7	25.7
Predator [19]	58.0	52.8	26.7	22.4
CoFiNet [50]	49.8	46.8	24.4	21.5
YOHO [44]	64.4	64.1	25.9	23.2
RIGA [49]	68.4	<u>68.5</u>	32.1	32.1
Lepard [20]	58.6	53.7	28.4	24.4
GeoTrans [27]	<u>71.9</u>	68.2	<u>43.5</u>	<u>40.0</u>
RoITr (*Ours*)	**82.6**	**82.3**	**54.3**	**53.2**
		Registration Recall (%) ↑		
SpinNet [1]	88.8	<u>93.2</u>	58.2	61.8
Predator [19]	89.0	92.0	59.8	58.6
CoFiNet [50]	89.3	92.0	67.5	62.5
YOHO [44]	90.8	92.5	65.2	66.8
RIGA [49]	89.3	93.0	65.1	66.9
Lepard [20]	**92.7**	84.9	65.4	49.0
GeoTrans [27]	<u>92.0</u>	92.0	**75.0**	<u>71.8</u>
RoITr (*Ours*)	91.9	**94.7**	<u>74.8</u>	**77.2**

Table 1. Quantitative results on (Rotated) 3DMatch & 3DLoMatch. 5,000 points/correspondences are used for the evaluation.

Comparison with the State-of-the-Art. We compare RoITr with 7 state-of-the-art methods, among which Predator [19], CoFiNet [50], Lepard [20], and GeoTrans [27] are rotation-sensitive models, while SpinNet [1], YOHO [44], and RIGA [49] guarantee the rotation invariance by design. In Tab. 1 we demonstrate the matching and registration results on 3DMatch and 3DLoMatch, as well as on their rotated versions, with 5,000 sampled points/correspondences. Regarding IR, RoITr outperforms all the others by a large margin on both datasets, which indicates our method matches points more correctly. For FMR, we significantly surpass all the others on 3DLoMatch, while staying on par with CoFiNet and YOHO on 3DMatch, which indicates that our model is good at coping with hard cases, *i.e.*, we find at least 5% inliers on more test data. For the registration evaluation in terms of RR, RoITr achieves comparable performance with GeoTrans and Lepard on 3DMatch, but leads the board together with GeoTrans on 3DLoMatch with an overwhelming advantage over the others. Our stability against additional rotations is further demonstrated on the rotated data, where we outperform all the others with a substantial margin. Qualitative results can be found in Fig. 6.

[2] We follow [49] to calculate the RR strictly with RMSE < 0.2m on the rotated data, which is slightly different from the RR on the original data.

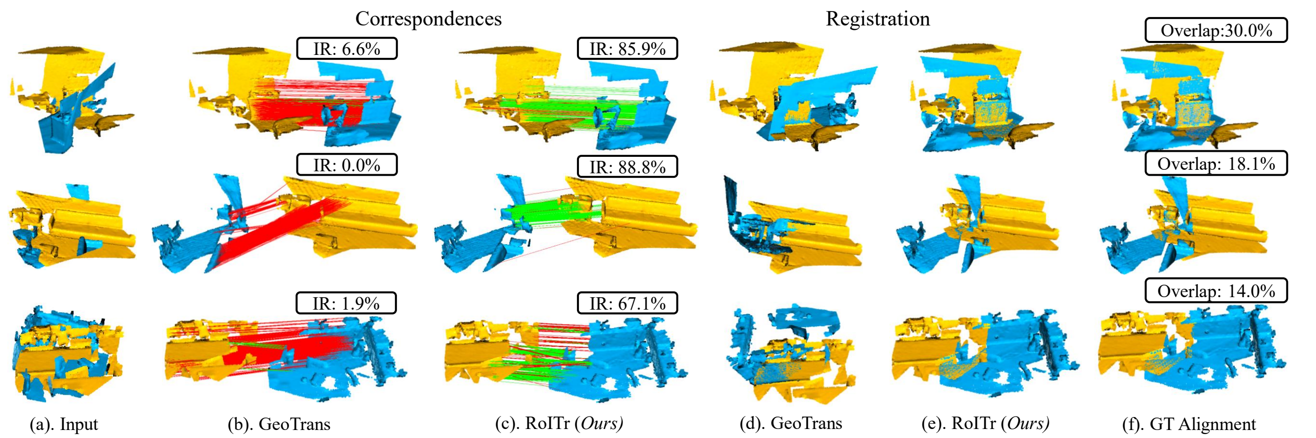


Figure 6. Qualitative results on 3DLoMatch. GeoTrans [27] is used as the baseline. Columns (b) and (c) show the correspondences, while columns (d) and (e) demonstrate the registration results. Green/red lines indicate inliers/outliers. More cases are shown in the Appendix.

	3DMatch					3DLoMatch				
# Samples	5000	2500	1000	500	250	5000	2500	1000	500	250
Feature Matching Recall (%) ↑										
SpinNet [1]	97.4	97.0	96.4	96.7	94.8	75.5	75.1	74.2	69.0	62.7
Predator [19]	96.6	96.6	96.5	96.3	96.5	78.6	77.4	76.3	75.7	75.3
CoFiNet [50]	98.1	**98.3**	**98.1**	**98.2**	**98.3**	83.1	83.5	83.3	83.1	82.6
YOHO [44]	**98.2**	97.6	97.5	97.7	96.0	79.4	78.1	76.3	73.8	69.1
RIGA [49]	97.9	97.8	97.7	97.7	97.6	85.1	85.0	85.1	84.3	85.1
GeoTrans [27]	97.9	97.9	97.9	97.9	97.6	88.3	88.6	88.8	88.6	88.3
RoITr (*Ours*)	98.0	98.0	97.9	98.0	97.9	**89.6**	**89.6**	**89.5**	**89.4**	**89.3**
Inlier Ratio (%) ↑										
SpinNet [1]	48.5	46.2	40.8	35.1	29.0	25.7	23.7	20.6	18.2	13.1
Predator [19]	58.0	58.4	57.1	54.1	49.3	26.7	28.1	28.3	27.5	25.8
CoFiNet [50]	49.8	51.2	51.9	52.2	52.2	24.4	25.9	26.7	26.8	26.9
YOHO [44]	64.4	60.7	55.7	46.4	41.2	25.9	23.3	22.6	18.2	15.0
RIGA [49]	68.4	69.7	70.6	70.9	71.0	32.1	33.4	34.3	34.5	34.6
GeoTrans [27]	71.9	75.2	76.0	82.2	**85.1**	43.5	45.3	46.2	52.9	**57.7**
RoITr (*Ours*)	**82.6**	**82.8**	**83.0**	**83.0**	83.0	**54.3**	**54.6**	**55.1**	**55.2**	55.3
Registration Recall (%) ↑										
SpinNet [1]	88.8	88.0	84.5	79.0	69.2	58.2	56.7	49.8	41.0	26.7
Predator [19]	89.0	89.9	90.6	88.5	86.6	59.8	61.2	62.4	60.8	58.1
CoFiNet [50]	89.3	88.9	88.4	87.4	87.0	67.5	66.2	64.2	63.1	61.0
YOHO [44]	90.8	90.3	89.1	88.6	84.5	65.2	65.5	63.2	56.5	48.0
RIGA [49]	89.3	88.4	89.1	89.0	87.7	65.1	64.7	64.5	64.1	61.8
GeoTrans [27]	**92.0**	**91.8**	**91.8**	**91.4**	**91.2**	**75.0**	**74.8**	74.2	74.1	73.5
RoITr (*Ours*)	91.9	91.7	**91.8**	**91.4**	91.0	74.7	**74.8**	**74.8**	**74.2**	**73.6**

Table 2. Quantitative results on 3DMatch & 3DLoMatch with a varying number of points/correspondences. See the results on rotated data in the Appendix.

Analysis on the Number of Correspondences. We further analyze the influence of a varying number of correspondences. As illustrated in Tab. 2, RoITr shows outstanding performance on both datasets with various correspondences, proving its stability when only a few correspondences are accessible. The same test on the rotated benchmarks is given in the Appendix.

4.2. Deformable Objects: 4DMatch & 4DLoMatch

Dataset. 4DMatch [20] contains 1,761 animations randomly selected from DeformingThings4D [21]. The 1,761 sequences are divided into 1,232/176/353 as train/val/test, where the test set is further split into 4DMatch and 4DLoMatch based on an overlap ratio threshold of 45%. **Metrics.** We follow [20] to use two different metrics: (1). *Inlier Ratio* (IR) which is defined as same as the IR on 3DMatch,

Category	Method	4DMatch		4DLoMatch	
		NFMR(%) ↑	IR(%) ↑	NFMR(%) ↑	IR(%) ↑
Scene Flow	PWC [45]	21.6	20.0	10.0	7.2
	FLOT [24]	27.1	24.9	15.2	10.7
Feature Matching	Predator [19]	56.4	60.4	32.1	27.5
	GeoTrans [27]	83.2	82.2	65.4	63.6
	Lepard [20]	**83.7**	82.7	66.9	55.7
	RoITr (*Ours*)	83.0	**84.4**	**69.4**	**67.6**

Table 3. Quantitative results on 4DMatch & 4DLoMatch.

but with a different threshold (*i.e.*, 0.04m); (2). *Non-rigid Feature Matching Recall* (NFMR) that measures the fraction of ground-truth matches that can be successfully recovered by the putative correspondences.

Comparison with the State-of-the-Art. We compare RoITr with 5 baselines, among which PWC [45] and FLOT [24] are scene flow-based methods, while Predator [19], Lepard [20], GeoTrans [27] are based on feature matching. The results shown in Tab. 3 indicate that although our rotation-invariance is mainly designed for rigid scenarios, RoITr could also achieve outstanding performance in the non-rigid matching task, which further confirms the superiority of our model design. Qualitative results are demonstrated in Fig. 7.

4.3. Ablation Study

Local Attention. We first replace our PPFTrans with Point-Transformer (PT) [53] in Tab. 4 (a.1), which leads to a sharp performance drop. We then ablate by embedding our PPF-based local coordinates into PT (Tab. 4 (a.2)) and by adopting the relative coordinates, *i.e.*, $\mathbf{p}_j - \mathbf{p}_j$, used by PT in our PAM (Tab. 4 (a.3)). Our local coordinate representation significantly boosts the performance of PT in the task of point cloud matching and meanwhile makes it rotation-invariant, although its performance is still far behind ours. However, the relative coordinates fail to work in our PAM, as we adopt a more efficient attention mechanism [43] that learns a scalar attention value for each feature $\mathbf{x} \in \mathbb{R}^c$ and is consequently hard to work under varying poses with a

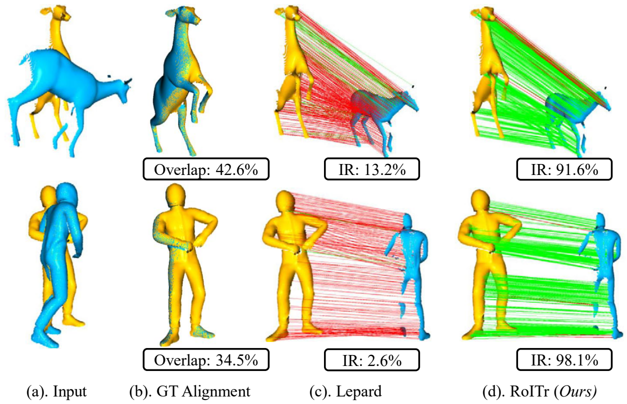


Figure 7. Qualitative results of non-rigid matching on 4DLoMatch with Lepard [20] as the baseline. Green/red lines indicate inliers/outliers. See the Appendix for more examples.

rotation-sensitive design. As a comparison, PT learns a per-channel vector attention $\mathbf{a} \in \mathbb{R}^c$ for the same feature $\mathbf{x}$ and could deal with the pose variations, but at the cost of the efficiency as shown in Fig. 8. When the number of channels is increased, our advantage in terms of efficiency is enlarged. As we achieve that with more parameters, the gap becomes more significant when runtime is normalized with the number of parameters in the right figure. With our PPF-based local coordinate, the scalar attention could focus on the pose-agnostic pure geometry and therefore achieves the best performance shown in Tab. 4 (a.4).

Abstraction Layer. We ablate our Attentional Abstraction Layer (AAL) by replacing it with the pooling-based abstraction design used in [25, 26, 53]. We test the max pooling in Tab. 4 (b.1) and the average pooling in Tab. 4 (b.2), both showing a degrading performance compared with our AAL, which demonstrates our superiority.

Backbone. In Tab. 4 (a.1) we have shown our superiority compared with PT [53]. We further replace our PPFTrans with the KPConv-based backbone network which is widely used in previous deep matchers [19, 27, 50]. The fact that KPConv falls behind our design demonstrates the advantage of PPFTrans in geometry encoding.

Global Transformer. We replace our design with the global transformer of GeoTrans [27] which performs state-of-the-art but without the cross-frame spatial awareness. The dropping results in Tab. 4 (d.1) proves the excellence of our design with the cross-frame position awareness.

The Number of Global Transformers. To demonstrate the importance of being globally aware, we first remove the global transformer. The substantial performance drop confirms the significance of global awareness. Then we add one global transformer and observe an increased performance. In our default setting with 3 global transformers, the model performs the best. However, when the number is increased to 5, the model shows a slight performance drop, which we owe to overfitting. As the data augmentation of rotations has less effect on an intrinsically rotation-invariant method, more data is required for training a larger model.

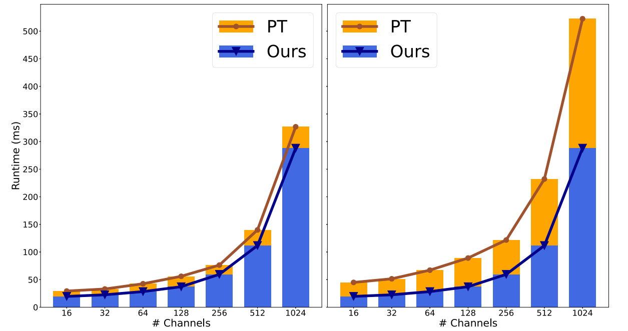


Figure 8. **Left**: Runtime comparison between our PPF attention Mechanism (PAM) and the local attention in PointTransformer [53]. **Right**: Runtime normalized by aligning the number of parameters.

Category	Model	Origin FMR	Origin IR	Origin RR	Rotated FMR	Rotated IR	Rotated RR
a. Local	1. PT [53]	79.0	36.5	61.6	76.5	34.7	60.0
	*2. PPF+PT [53]	87.0	49.9	69.9	86.8	49.4	71.2
	3. Δxyz+Ours	-	-	-	-	-	-
	*4. *Ours*	**89.6**	**54.3**	**74.7**	**89.4**	**53.2**	**77.2**
b. Aggregation	*1. max pooling	85.2	50.1	70.5	85.4	50.2	71.9
	*2. avg pooling	87.8	52.6	73.8	87.2	52.5	74.7
	*3. *Ours*	**89.6**	**54.3**	**74.7**	**89.4**	**53.2**	**77.2**
c. Backbone	1. KPConv [40]	85.2	44.4	70.6	83.0	42.3	71.5
	*2. *Ours*	**89.6**	**54.3**	**74.7**	**89.4**	**53.2**	**77.2**
d. Global	*1. GeoTrans [27]	87.7	53.6	73.0	87.5	**53.2**	75.1
	*2. *Ours*	**89.6**	**54.3**	**74.7**	**89.4**	**53.2**	**77.2**
e. #Global	*1. $g = 0$	87.2	37.6	70.7	87.5	37.6	72.7
	*2. $g = 1$	87.1	42.1	70.8	86.8	42.1	73.0
	*3. $g = 3$ (*Ours*)	**89.6**	**54.3**	**74.7**	**89.4**	**53.2**	**77.2**
	*4. $g = 5$	87.1	52.5	72.1	87.0	52.4	73.3

Table 4. Ablation study on (rotated) 3DLoMatch. 5,000 points/correspondences are leveraged. * indicates the methods with intrinsic rotation invariance. See the Appendix for the results on (rotated) 3DMatch.

5. Conclusion

We introduced RoITr - an intrinsically rotation-invariant model for point cloud matching. We proposed PAM (PPF Attention Mechanism) that embeds PPF-based local coordinates to encode rotation-invariant geometry. This design lies at the core of AAL (Attention Abstraction Layer), PAL (PPF Attention Layer), and TUL (Transition Up Layer) which are consecutively stacked to compose PPFTrans (PPF Transformer) for representative and pose-agnostic geometry description. We further enhanced features by introducing a novel global transformer architecture, which ensures the rotation-invariant cross-frame spatial awareness. Extensive experiments are conducted on both rigid and non-rigid benchmarks to demonstrate the superiority of our approach, especially the remarkable robustness against arbitrary rotations. Limitations are discussed in the Appendix.

Acknowledgment. This paper is supported by the National Natural Science Foundation of China under Grant No. 62025208. We appreciate the help from Lennart Bastian, Mert Karaoglu, Ning Liu, and Zhiying Leng.

References

[1] Sheng Ao, Qingyong Hu, Bo Yang, Andrew Markham, and Yulan Guo. Spinnet: Learning a general surface descriptor for 3d point cloud registration. In *CVPR*, 2021. 1, 2, 6, 7

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[3] 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. 2

[4] Axel Barroso-Laguna, Yannick Verdie, Benjamin Busam, and Krystian Mikolajczyk. Hdd-net: Hybrid detector descriptor with mutual interactive learning. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2

[5] Tolga Birdal and Slobodan Ilic. Point pair features based object detection and pose estimation revisited. In *3DV*, 2015. 4

[6] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 1

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

[8] Chin Seng Chua and Ray Jarvis. Point signatures: A new representation for 3d object recognition. *IJCV*, 1997. 2

[9] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *ECCV*, 2018. 1, 2, 4

[10] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *CVPR*, 2018. 1, 2

[11] Bertram Drost, Markus Ulrich, Nassir Navab, and Slobodan Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *CVPR*, 2010. 1, 2, 3, 4

[12] 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. 6

[13] 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

[14] Yulan Guo, Ferdous Sohel, Mohammed Bennamoun, Min Lu, and Jianwei Wan. Rotational projection statistics for 3d local surface description and object recognition. *IJCV*, 2013. 2

[15] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *CVPR*, 2021. 1

[16] Ji Hou, Saining Xie, Benjamin Graham, Angela Dai, and Matthias Nießner. Pri3d: Can 3d priors help 2d representation learning? In *ICCV*, 2021. 1

[17] Chun-Hao Huang, Benjamin Allain, Jean-Sébastien Franco, Nassir Navab, Slobodan Ilic, and Edmond Boyer. Volumetric 3d tracking by detection. In *CVPR*, 2016. 1

[18] Chun-Hao Paul Huang, Benjamin Allain, Edmond Boyer, Jean-Sébastien Franco, Federico Tombari, Nassir Navab, and Slobodan Ilic. Tracking-by-detection of 3d human shapes: from surfaces to volumes. *TPAMI*, 2017. 1

[19] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap. In *CVPR*, 2021. 1, 2, 5, 6, 7, 8

[20] Yang Li and Tatsuya Harada. Lepard: Learning partial point cloud matching in rigid and deformable scenes. In *CVPR*, 2022. 1, 2, 6, 7, 8

[21] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *ICCV*, 2021. 7

[22] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *CVPR*, 2015. 1

[23] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE International Symposium on Mixed and Augmented Reality*, 2011. 1

[24] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *ECCV*, 2020. 7

[25] 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. 1, 2, 8

[26] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 2017. 1, 4, 8

[27] Zheng Qin, Hao Yu, Changjian Wang, Yulan Guo, Yuxing Peng, and Kai Xu. Geometric transformer for fast and robust point cloud registration. In *CVPR*, 2022. 1, 2, 3, 5, 6, 7, 8

[28] Zheng Qin, Hao Yu, Changjian Wang, Yuxing Peng, and Kai Xu. Deep graph-based spatial consistency for robust non-rigid point cloud registration. *arXiv preprint arXiv:2303.09950*, 2023. 2

[29] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood consensus networks. *NeurIPS*, 2018. 6

[30] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009. 1, 2

[31] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *IROS*, 2008. 1, 2

[32] Mahdi Saleh, Shervin Dehghani, Benjamin Busam, Nassir Navab, and Federico Tombari. Graphite: Graph-induced feature extraction for point cloud registration. In *3DV*, 2020. 1, 2

[33] Mahdi Saleh, Yige Wang, Nassir Navab, Benjamin Busam, and Federico Tombari. Cloudattention: Efficient multi-scale attention scheme for 3d point cloud learning. *arXiv preprint arXiv:2208.00524*, 2022. 1

[34] Mahdi Saleh, Shun-Cheng Wu, Luca Cosmo, Nassir Navab, Benjamin Busam, and Federico Tombari. Bending graphs: Hierarchical shape matching using gated optimal transport. In *CVPR*, 2022. 1, 2

[35] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 6

[36] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 1967. 6

[37] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *CVPR*, 2021. 6

[38] Jiapeng Tang, Lev Markhasin, Bi Wang, Justus Thies, and Matthias Nießner. Neural shape deformation priors. *arXiv preprint arXiv:2210.05616*, 2022. 2

[39] Jiapeng Tang, Dan Xu, Kui Jia, and Lei Zhang. Learning parallel dense correspondence from spatio-temporal descriptors for efficient and robust 4d reconstruction. In *CVPR*, 2021. 1

[40] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 1, 8

[41] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, 2010. 1, 2

[42] Giovanni Trappolini, Luca Cosmo, Luca Moschella, Riccardo Marin, Simone Melzi, and Emanuele Rodolà. Shape registration in the time of transformers. *NeurIPS*, 2021. 2

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 3, 5, 7

[44] Haiping Wang, Yuan Liu, Zhen Dong, and Wenping Wang. You only hypothesize once: Point cloud registration with rotation-equivariant descriptors. In *ACM MM*, 2022. 1, 2, 6, 7

[45] Wenxuan Wu, Zhiyuan Wang, Zhuwen Li, Wei Liu, and Li 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. 7

[46] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pretraining for 3d point cloud understanding. In *ECCV*, 2020. 1

[47] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*, 2018. 2

[48] Zi Jian Yew and Gim Hee Lee. Regtr: End-to-end point cloud correspondences with transformers. In *CVPR*, 2022. 1, 2, 5

[49] Hao Yu, Ji Hou, Zheng Qin, Mahdi Saleh, Ivan Shugurov, Kai Wang, Benjamin Busam, and Slobodan Ilic. Riga: Rotation-invariant and globally-aware descriptors for point cloud registration. *arXiv preprint arXiv:2209.13252*, 2022. 1, 2, 5, 6, 7

[50] Hao Yu, Fu Li, Mahdi Saleh, Benjamin Busam, and Slobodan Ilic. Cofinet: Reliable coarse-to-fine correspondences for robust pointcloud registration. *NeurIPS*, 2021. 1, 2, 3, 5, 6, 7, 8

[51] 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. 1, 2, 6

[52] Yu Zhang, Junle Yu, Xiaolin Huang, Wenhui Zhou, and Ji Hou. Pcr-cg: Point cloud registration via deep explicit color and geometry. In *ECCV*, 2022. 1

[53] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 1, 2, 7, 8