2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Recurrent Non-Rigid Point Cloud Registration

Yue Cao, Ziang Cheng and Hongdong Li

Abstract— **Non-rigid point cloud registration remains a significant challenge in 3D computer vision due to the complexity of structural deforms, lack of overlaps, and sensitivity to initialization. This paper introduces a framework inspired by the recent success in recurrent architecture, adapted to accommodate the unique characteristics of point clouds. More specifically, we design a recurrent update network block for progressively refining local registration results under a local rigidity assumption, starting from an initial global SE(3) alignment. Through comparison, our method consistently outperforms competing methods in standard metrics, achieving a 33% reduction in EPE on the 4DLoMatch benchmark compared to the second-best method. To the best of our knowledge, the proposed method is the first to successfully demonstrate that the recurrent update strategy can effectively address the non-rigid registration task with large displacement, significant deform, and low overlap. The source code and the model will be released at *http://dummy.url/*.**

I. Introduction

Non-rigid point cloud registration aims to find a transformation that optimally aligns two point sets. This problem is critical for various downstream applications such as augmented reality, medical image analysis, motion analysis, segmentation, and autonomous navigation. However, the inherent challenges of non-rigid registration, such as accommodating deformable structures and shape discrepancies, presence of outliers and partial to low overlaps, make it a complex task.

Classical techniques such as the Iterative Closest Point (ICP) and its variations have been foundational for rigid point cloud registration. Although these methods are adequate for scenarios with uniform body motions, they are less effective for non-rigid data sets. To address this limitation, several extensions and to classical ICP have been developed. Techniques such as Non-rigid ICP, Thin Plate Spline (TPS) [1], [2] have made significant strides in improving flexibility and adaptability to deformations. However, non-rigid registration presents a considerably more challenging optimization problem, where these handcrafted methods require significant computational efforts and often depend heavily on the quality of initial alignment.

On the other hand, the advances in deep learning methods have enabled significant progress in non-rigid point cloud registration. Unlike classical methods that formulate the registration task as an optimization problem, learning methods directly regress the underlying motion patterns from the data, often offering improved precision and speed over traditional techniques. Preliminary attempts include Deep Closest Point (DCP), FlowNet3D [3], [4], [5], and their successors. However, these methods still struggle with challenging scenarios such as significant deformations or minimal overlap between point sets, which are typical in real-world situations.

In response to these challenges, this paper introduces a novel method that combines the classical local rigidity prior and the learning power of deep neural networks under a recurrent refinement scheme. The recurrent strategy is inspired by the success of the Recurrent All-Pairs Field Transforms (RAFT) [6] architecture in 2D optical flow. We show that by cleverly inserting rigidity constraints in such a paradigm, we are able to effectively capture the underlying structural transformations with increased robustness and efficiency.

From a technical aspect, the contribution of this paper also lies in the adaptation of the traditional recurrent architecture to suit the specific characteristics of point clouds. This includes the integration of KPconv [7] layers, which are specifically designed for point cloud processing, to maintain the structural integrity and spatial relationships inherent in the 3D points data. Unlike conventional RAFT frameworks, our model introduces a unique recurrent update block for point cloud data. This recurrent block iteratively performs local registration cost lookups, and progressively refines the local registrations in a course to fine manner, starting from an initial globally rigid transformation. Combined with an enhanced supervision strategy, our method adeptly manages scenarios involving large deformations and minimal overlap.

Our extensive evaluations on standard benchmarks for non-rigid point cloud registration demonstrate that our method consistently outperforms existing models in accuracy, thereby establishing a new state-of-the-art in the field. In particular, our method outperforms the second best method by 33% reduction in end point errors (EPE) on the challenging low overlap benchmark. We also provide a detailed ablation study of our design choices to underscore the effectiveness of the proposed recurrent block, rigidity constraint, and initialization strategy.

To summarize our contributions:

1) We introduce a novel approach to non-rigid point cloud registration that synergistically integrates local rigidity within a recurrent refinement network. This method significantly enhances the alignment accuracy and robustness.
2) We adapt the traditional recurrent network architecture to the unique needs of point cloud data to enable efficient and accurate registration. This is achieved by a recurrent block that is specifically crafted to process point cloud data.

This work was supported by Australia National University

Yue Cao, Ziang Cheng, and Hongdong Li are with The Australian National University, Australia (e-mails: Yue.Cao1@anu.edu.au; Ziang.cheng@anu.edu.au; Hongdong.Li@anu.edu.au).

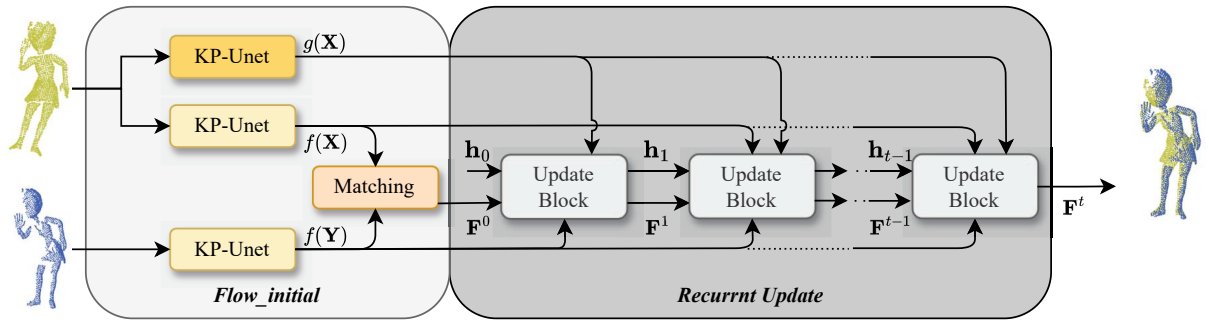

Fig. 1. Overview of our framework. First, the features of the source and target point clouds are extracted from a KP-Unet, and the context features $g(\mathbf{X})$ of source points are extracted at the same time. The features $f(\mathbf{X})$ and $f(\mathbf{Y})$ will pass the matching module and generate an initial flow. Then the initial flow will be refined through several recurrent update blocks.

3) Our method is robust to challenging scenarios involving large deforms and low overlaps, where it significantly outperforms competing methods. And it is robust to initial correspondences.

II. Related Work

In non-rigid point cloud registration, methods can be classified into classical approaches and learning-based techniques. Classical methods are effective for rigid scenarios but struggle with non-linear, non-rigid data and are sensitive to local minima. On the contrary, learning-based methods overcome these challenges by training with large datasets.

A. Classical methods

Classical methods like ICP and its variants [8], [9], [10], [11], [12], [13], [14] offer robust solutions for rigid point cloud registration but often struggle with non-rigid data due to their reliance on globally linear transformation models. Extensions to these methods, such as Non-Rigid ICP [1], introduce flexibility to handle deformations, but at the cost of increased computational complexity and the necessity for good initial alignment. Thin Plate Spline (TPS) Robust Point Matching (RPM) [2] combines the use of thin-plate splines with a soft assignment of correspondences (robust point matching) to deal with non-rigidity and outliers. Coherent Point Drift (CPD) [15] treats one point cloud as a probability density and deforms it to match another using a coherent motion model. Similar to CPD, Gaussian Mixture Models (GMM) model point clouds as mixtures of Gaussians and find the optimal alignment by matching these distributions. Multiscale Non-Rigid Point Set Registration (N-PSR) [16] uses a pyramid approach to gradually align point clouds from coarse to fine, adapting to non-rigid deformations at each scale.

B. Learning methods

Deep learning methods have shown promise in handling non-rigid transformations by learning complex motion patterns directly from data. Approaches such as DCP [3] and FlowNet3D/FlowNet3D++ [4], [5] have introduced neural networks into point cloud scene flow estimation tasks. DCP [3] embebed the point cloud first and then used an attention mechanism with a pointer generation layer which selects sequence specific items from the input data. Finally, differentiable singular value decomposition (SVD) is applied to find transformations. FlowNet3D/FlowNet3D++ [4], [5] introduces two innovative layers: the flow embedding layer and the upconv layer to estimate a transnational flow vector for every point. Another work named FLOT [17] estimates the scene flow from point clouds utilizing the concepts of graph matching and optimal transport. With the development of Transformer, Trappolini et al. design a transformer-based procedure [18] for point cloud registration, Geotransformer [19] and Lepard [20] improves the performance by finding the point-wise relative positions with the help of transformers.

PointPWC-net [21], inspired by the architecture of the well-known PWC-Net [22] used for optical flow in images, applies a similar hierarchical pyramid-based approach to 3D point clouds. PointRNN [23] combines the feature of each point with the attributes and positional differences of its neighboring point patches. The neural deformation pyramid (NDP) [24], based on the MLP architecture, is a pyramid architecture that takes an input of a sinusoidally encoded 3D point and generates the output of the motion increments from the previous frequency level [25].

C. Recurrent network for scene motion

Closely related to ours are methods that rely on Recurrent Neural Networks (RNN) for iteratively refinement of registration results. RAFT [6] demonstrates such recurrent strategy is highly effective for estimating dense motion flow fields. While it focuses on 2D optical flow estimation in image grids, the fundamental ideas and techniques have been extended to 3D scene flow estimation of sparse point clouds.

Feng et al. [26] introduced an unsupervised framework RMA-Net, which gradually recovers scene flows using a gated recurrent unit (GRU) by breaking down complex transformations into a combination of simpler ones. PV-RAFT [27] extends the RAFT framework to 3D point clouds by combining point and voxel representations to estimate scene flow. FlowStep3D [28] and RCP [29] adapt GRU for iterative refinements of sparse 3D scene flows.

However, these approaches struggle with large displacements, low overlap, or non-rigid 3D motions. To our knowledge, our method is the first to successfully show that the recurrent update strategy can effectively address the registration task in these challenging scenarios.

III. Non rigid point cloud registration under local rigidity assumption

Non-rigid point cloud registration involves finding a spatial mapping:

$$\mathbf{F} : \mathbb{R}^3 \rightarrow \mathbb{R}^3, \tag{1}$$

which optimally aligns a source point set $\mathbf{X}$ with a target point set $\mathbf{Y}$. These point sets can be heterogeneous, meaning that $\mathbf{X}$ and $\mathbf{Y}$ may contain differing numbers of points, lack correspondences, only partially overlap and exhibit structural deformations between them.

To make the problem well-posed, we shall assume the registration to be locally rigid. That is to say, for any point $\mathbf{x} \in \mathbf{X}$, a surrounding patch is to be mapped to its corresponding location in $\mathbf{Y}$ via a distance-preserving transformation, formally expressed as:

$$\forall \mathbf{p} \in \Omega\mathbf{x}, \mathcal{F}(\mathbf{p}) \approx \mathbf{F}_\mathbf{x}(\mathbf{p}) = \mathbf{R}_\mathbf{x}\mathbf{p} + \mathbf{T}_\mathbf{x}, \tag{2}$$

where $\Omega\mathbf{x}$ denotes the local neighborhood around $\mathbf{x}$, and $\mathbf{R}_\mathbf{x} \in$ SO(3) and $\mathbf{T}_\mathbf{x} \in \mathbb{R}^3$ represent the rotation and translation components of local transformation, respectively. Under such a local rigidity assumption, it is sufficient to define $\mathbf{F}$ as a discrete set of flows for every point in $\mathbf{X}$; that is, as a matrix $\mathbf{F} \in \mathbb{R}^{|\mathbf{X}|\times 3}$

IV. Method

A. Feature Extraction

For normalization purposes, we assume without loss of generality that both input point clouds $\mathbf{X}$ and $\mathbf{Y}$ are scaled to fit within a unit sphere centered at their respective centers of mass. Subsequently, the per-point features are extracted from the source and target point clouds, respectively, using a shared U-Net architecture f with KPConv layers [7]. This feature extraction U-Net comprises multiple downsampling and upsampling blocks. Each downsampling block contains a ResNet [30] followed by a farthest point downsampling layer for efficient points reduction. On the contrary, each upsampling stage reverts to the last downsampling by interpolating features at earlier locations. The intention is to expand the receptive field while maintaining a sparse feature set for computational efficiency. The final outputs yield $f(\mathbf{X})$ and $f(\mathbf{Y})$ as feature matrices whose rows represent features of individual points.

While the primary function of the feature-extracting U-Net f is to calculate feature matching costs (to be described later), a secondary context-aware U-Net g is also utilized. This network mirrors the structure of the feature U-Net f but is applied solely to the source point cloud $\mathbf{X}$. Its purpose is to extract high-level contextual information about the source geometry, in such a way that the context features $g(\mathbf{X})$ are always spatially aligned with the matching features $f(\mathbf{X})$.

B. Initial Flow Estimation by Matching Cost

With two feature matrices $f(\mathbf{X})$ and $f(\mathbf{Y})$ computed for the source and target point clouds respectively, we first estimate a set of initial flows $\mathbf{F}_0$. The purpose of this initial estimation is to globally align the point sets and bootstrap the refinement of local registrations. Therefore, it does not need to be exact; instead, we make the simple assumption that the initial flow can be parameterized by a global SE(3) transformation.

To estimate initial correspondences, we compute a dense matching cost matrix from source and target features

$$\mathbf{C} = \mathcal{N}\Big(f(\mathbf{X})\big(f(\mathbf{Y})\big)^\top\Big), \tag{3}$$

where $\mathcal{N}(\cdot)$ denotes matrix normalization by row-wise and column-wise softmax, as seen in other works [20]:

$$\mathcal{N}(\mathbf{C}) = \text{RowSoftMax}(\mathbf{C}) \odot \text{ColSoftMax}(\mathbf{C}), \tag{4}$$

where $\odot$ denotes the element-wise product.

A selective set of high confidence correspondences $(\mathbf{x}_i, \mathbf{y}_j)$ are then obtained by applying a high threshold value with $\{(i,j) \mid \mathbf{C}_{ij} > \zeta\}$. Subsequently, we solve for a global SE(3) transformation $\mathbf{R}_0, \mathbf{T}_0$ to parameterize the initial flows:

$$\mathbf{F}_0^\top = (\mathbf{R}_0 - \mathbf{I})\mathbf{X}^\top + \mathbf{T}_0, \tag{5}$$

$$\textbf{s.t. } \mathbf{R}_0, \mathbf{T}_0 = \arg\min_{\mathbf{R},\mathbf{T}} \sum_{i,j} \|\mathbf{R}\mathbf{x}_i + \mathbf{T} - \mathbf{y}_j\|^2. \tag{6}$$

Note, that $\mathbf{R}_0, \mathbf{T}_0$ are amenable to analytical solution by singular value decomposition (SVD) and do not require gradients during training.

C. Recurrent Update

Starting from the initial flow estimation, local registrations are iteratively refined by a series of recurrent updates. Each recurrent stage performs a matching cost lookup to estimate a residual flow while maintaining local rigidity. The recurrent local lookup strategy is inspired by RAFT [6] and shares a similar idea to PatchMatch [31], [32]; however, we found that a few modifications are necessary to accommodate the task of point cloud registration.

An overall illustration of a recurrent update block is given in Fig. 2. The t-th recurrent block takes as input per-point features $f(\mathbf{X})$ and $f(\mathbf{Y})$, the source context features $g(\mathbf{X})$ and the per-point flow estimation $\mathbf{F}^t$, and outputs a residual flow $\Delta\mathbf{F}^t$ to complete the flow update $\mathbf{F}^{t+1} = \mathbf{F}^t + \Delta\mathbf{F}^t$. Additionally, the GRU maintains a hidden representation updated by $\mathbf{h}_t \rightarrow \mathbf{h}_{t+1}$, with initial hidden states $\mathbf{h}_0 = \mathbf{0}$. For notational simplicity, we shall hereafter omit the iteration superscript t from $\mathbf{F}$.

1) Flow Embedding and Lookup Cost: Similar to RAFT [6], we perform local correlation lookups to query the feature matching costs in current flow estimations. However, unlike RAFT that operates on regular image grids, our lookup operation is performed on discrete point sets where sampling a full continuous cost volume quickly becomes intractable.

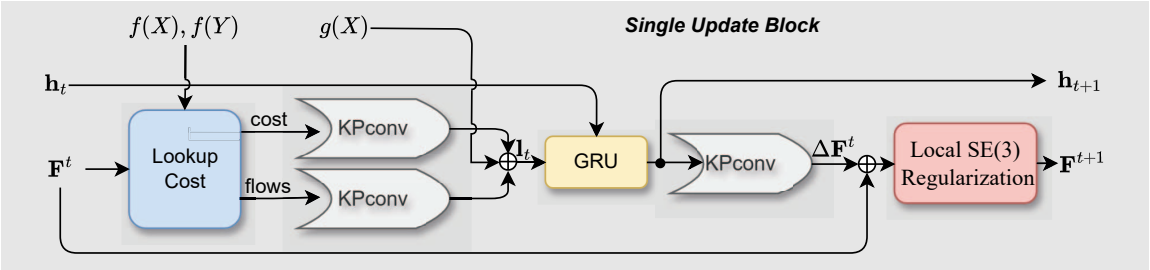

Fig. 2. Detailed illustration of a single update block. The inputs hidden state $\mathbf{h}_t$ and the flow $\mathbf{F}^t$ are both from the previous update block. The lookup block takes the feature of $\mathbf{X}$ and $\mathbf{Y}$ (source and target point cloud, respectively) and generates 4 scale levels of costs and flows. The feature of cost and flows will be concated with context feature $\mathbf{X}$, and passed to the GRU block. Then, GRU takes the concated feature and previous hidden state to generate the current hidden state and a flow feature. The flow feature will be used to estimate a new flow. Finally, the following regularization block will refine the new flow to be locally-rigid.

Instead, our matching costs are indexed from the discrete cost matrix $\mathbf{C}$. For every point in $\mathbf{x}_i \in \mathbf{X}$, we first identify the k-nearest neighbors (knn) of its corresponding location $\mathbf{x}_i + \mathbf{F}_i$ in $\mathbf{Y}$. We then collect all k matching costs and combine them as a cost vector. Formally, our per-point lookup cost is defined as a k-dimensional vector whose elements are taken from the cost matrix $\mathbf{C}$

$$\mathbf{c}_i = [\mathbf{C}_{ij}],\ \forall\ \mathbf{y}_j \in knn\big(\mathbf{x}_i + \mathbf{F}_i, \mathbf{Y}\big). \tag{7}$$

Additionally, we perform the cost lookup on all n levels of downsampled subsets of $\mathbf{Y}$ during the feature-extracting U-Net, thereby enabling multi-scale cost lookups. All $n \times k$ costs and their corresponding flows from $\mathbf{x}_i$ are then concatenated and fed to the GRU, which subsequently updates hidden states and estimates a residual flow to be added back to $\mathbf{F}_i$. Our GRU is similar to RAFT [6], with the exception that the 2D convolution layers are now replaced with KPConv layers.

$$\mathbf{z}_t = \sigma(\text{KPConv}(\mathbf{l}_t \oplus \mathbf{h}_t)) \tag{8}$$
$$\mathbf{r}_t = \sigma(\text{KPConv}(\mathbf{l}_t \oplus \mathbf{h}_t)) \tag{9}$$
$$\tilde{\mathbf{h}}_t = \tanh(\text{KPConv}(\mathbf{l}_t \oplus \mathbf{r}_t \odot \mathbf{h}_t)) \tag{10}$$
$$\mathbf{h}_{t+1} = \mathbf{z}_t \odot \mathbf{h}_t + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t \tag{11}$$
$$\Delta\mathbf{F}^t = \text{KPConv}(\mathbf{h}_{t+1}), \tag{12}$$

where $\mathbf{l}_t$ denotes collectively the multilevel lookup costs, their corresponding flows and context features.

2) Local SE(3) regularization: So far the updated flows have not been regularized by any rigidity constraint. In practice, the lack of such constraints often leads to poor registration results, as the task of unconstrained scene flow regression presents an inherently ill-posed problem. This issue is exemplified later in our ablation study, shown in Fig. 5.

To overcome this challenge, we incorporate a differentiable SE(3) regularization layer to rectify the updated flow. This regularization layer guarantees local rigidity by explicitly enforcing a local SE(3) transformation with the following steps:

Voxelization The per-point flows $\mathbf{F}$ are segmented into non-overlapping voxels that span the space.

SE(3) regression For each voxel, an SE(3) transformation is obtained by tackling the minimization problem as defined in Eq.(6) using a differentiable SVD solver that is amenable to backpropagation.

Rectification The flows within each voxel are recomputed from the SE(3) transformation of that voxel. This ensures that the rectified flows adhere to a locally rigid motion.

This rectification is numerically stable even when the system is underdetermined (such as when there are less than 3 points per voxel or all points are collinear); in such cases, the flows will simply remain unchanged.

As part of our recurrent update strategy, we apply this SE(3) regularization at progressively finer levels of voxelization with each round of iterative refinement. This approach allows the model to learn a hierarchy of structural rigidity, encouraging consistency in the SE(3) transformations across adjacent voxels where possible.

D. Training loss

Similar to RAFT [6], the networks are supervised by an L1 flow loss whose weight increases exponentially with each round of recurrent updates

$$L_F = \frac{1}{\sum_i \beta^i} \sum_i \beta^i |\mathbf{F} - \mathbf{F}_{\text{GT}}|_1, \tag{13}$$

where β is a fixed factor.

Additionally, we include a focal loss [20], [33] to directly supervise the matching cost matrix C to approximate true matching correspondences

$$L_C = -\frac{1}{|M_{\text{GT}}|} \sum_{i,j \in M_{\text{GT}}} \alpha(1 - \mathbf{C}_{ij})^\gamma \log \mathbf{C}_{ij} \tag{14}$$

where α and γ are two constant parameters, and M_{GT} is the set of ground truth matching obtained by thresholding the distance between ground truth warped $\mathbf{X}$ and $\mathbf{Y}$

$$M_{\text{GT}} = \{(i,j) \mid \|\mathbf{F}_{\text{GT}}(\mathbf{x}_i) - \mathbf{y}_j\|_2 < \epsilon\}. \tag{15}$$

The total training loss is:

$$L = \lambda L_F + L_C \tag{16}$$

where $\lambda = 0.5$

V. EXPERIMENTS

A. Dataset

All competing learning methods are trained on the 4DMatch dataset [20], which includes a total of 1,972 animated sequences along with their corresponding ground truth dense flows from Deforming4D [34]. The point cloud pairs within the 353-sequence testing set are further split into 4DMatch and 4DLoMatch benchmarks according to their overlaps, where the latter contains pairs with less than 45% overlapping regions. It should be noted that, unlike traditional scene flow dataset, pairs in 4DMatch are note collected from consecutive time frames and therefore generally contain much larger displacement. For our experiment, we use the same train/test split as lepard [20] and NPD [24], where point cloud pairs with globally rigid motions are removed from the testset, potentially making it even more challenging.

B. Implementation Details

In the implementation of our model, we utilize an encoder-decoder architecture optimized for efficiency in point cloud processing. The feature extraction UNet f and g comprise 5 downsampling blocks and 4 upsampling blocks. Each downsampling block contains a two-layer KPConv ResNet followed by a parameter-free downsampling layer; each upsampling block contains a single KPConv layer that restores features at previously downsampled points. The final feature outputs from both UNets have feature dimension of 256. To generate the initial flow, we set the confidence threshold $\zeta = 0.2$.

Additionally, our model uses GRUs to refine the initial flow. The recurrent block is repeated by $t = 14$ times during botrepeatedand inference. For the lookup operation, we use $n = 4$ levels of scales, and the numbers of k nearest neighbors are $k = [25, 20, 20, 10]$ for each respective scale (sparser scale has less number of neighbors). At the end of each block, the flows are voxelwithed at an increasing spatial resolution of $[1, 2, 2, 2, 2, 3, 3, 3, 4, 4, 4, 8, 8, 8]$, respectively.

For training loss, we select $\beta = 1.25$ for flow loss and $\alpha = 0.25$ and $\gamma = 2$ for focal loss. We use a stochastic gradient descend (SGD) optimizer with ah learning rate of 0.015, and train our networks for 25 epochs using two NVIDIA RTX 3090 GPUs.

C. Evaluation Matrix and Results

To evaluate the performance of all competing methods, we use two common metrics for non-rigid registration:

- End-Point Error (EPE): the L2 norm distance between predicted flow and ground truth flow.
- Outliers Ratio (OR): the percentage of points whose relative error is greater than 0.3. The relative error here refers to EPE divided by the scale of ground truth flow.

We compare our registration results with both the classical and the learning methods. The qualitative results are shown in Fig.3 for both the 4DMatch and the 4DLoMatch bechmarks. Quantitative results on 4DMatch and 4DLoMatch test sets are listed in the Table. I and II, respectively.

TABLE I
QUANTITATIVE NON-RIGID REGISTRATION RESULTS ON 4DMATCH.

Method	EPE↓	OR (%)↓	Time (s)↓
Classical			
ICP [35]	0.296	71.50	**0.10**
ZoomOut [36]	0.598	89.27	151.10
CPD [37]	0.274	74.52	4.52
BCPD [38]	0.291	73.74	5.81
Sinkhorn [38]	0.308	79.86	3.76
NICP [1]	0.325	80.04	4.80
NSFP [39]	0.265	64.96	39.54
Nerfies [40]	0.280	58.91	115.94
NDP [24]	**0.195**	**45.04**	2.31
Learning			
Lepard [20]+SVD	0.137	43.43	**0.06**
PointPWC [21]	0.182	52.07	**0.06**
FLOT [17]	0.133	40.49	0.07
GeomFmaps [41]	0.152	37.90	135.18
Synorim-pw [42]	0.099	26.01	0.41
Lepard+NICP [20]	0.097	23.02	3.93
LNDP+Lepard [24]	0.075	16.78	2.39
Ours	0.088	22.34	1.25
Ours+Lepard[20]	**0.072**	**16.34**	1.68

TABLE II
QUANTITATIVE NON-RIGID REGISTRATION RESULTS ON 4DLOMATCH.

Method	EPE↓	OR (%)↓	Time (s)↓
Classical			
ICP [35]	0.565	90.87	**0.10**
ZoomOut [36]	0.663	90.39	151.10
CPD [37]	0.463	84.49	4.52
BCPD [38]	0.492	86.88	5.81
Sinkhorn [38]	0.505	89.47	3.76
NICP [1]	0.517	92.37	4.80
NSFP [39]	0.495	84.77	39.54
Nerfies [40]	0.498	82.21	115.94
NDP [24]	0.467	80.47	2.31
Learning			
Lepard [20]+SVD	0.160	44.16	**0.06**
PointPWC [21]	0.279	55.70	**0.06**
FLOT [17]	0.210	42.51	0.07
GeomFmaps [41]	0.148	64.63	135.18
Synorim-pw [42]	0.170	31.12	0.41
Lepard+NICP [20]	0.283	52.99	3.93
LNDP+Lepard [24]	0.169	32.14	2.39
Ours	0.126	27.87	1.25
Ours+Lepard [20]	**0.099**	**25.39**	1.68

a) Qualitative results: ICP [35], implemented in Open3D [43], is a rigid registration method and achieves the worst result due to transformation being non-rigid. Additionaly, it cannot handle partially overlapped point cloud. NICP [1] estimates point clouds from depth images, hence, the generated shapes of point clouds are different from other methods. NDP and LNDP are both from both [24], with NDP being a non-learning method and LNPD being learning-based. LNDP achieves better results because it is trained (though in an unsupervised manner) with Chamfer distance. However, it force that each point in the source point cloud must correspond to a point in the target point cloud, leading to poor result on low overlap cases. Our methods achieve competitive results on 4DMatch dataset and performs much better than others on 4DLoMatch dataset. For

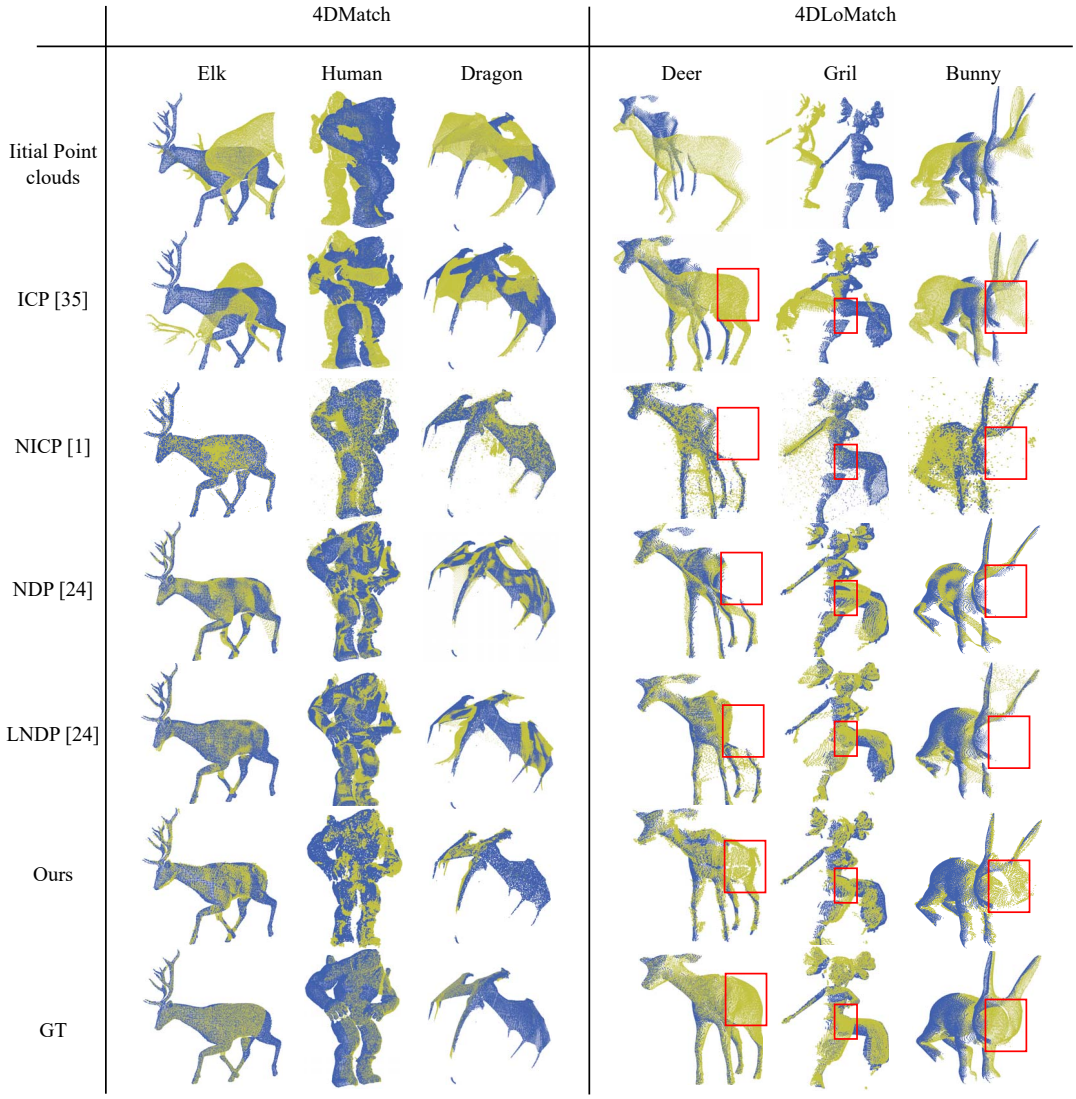

Fig. 3. Comparison of registration results between different methods on 4DMatch (left) and 4DLoMatch (right) datasets. Yellow points are source points, and blue points are target points. Our method achieves competitive results with other learning methods in the 4DMatch dataset. Due to the rigidity assumption, our method is able to extrapolate flows even on non-overlapping regions in 4DLoMatch.

4DLoMatch dataset, our method builds on LNDP's strengths but advances further by enhancing smoothness and handling partial overlaps more effectively. This leads to superior performance in maintaining geometric details and overall registration accuracy. Hence, from Fig. 3, it is clear to see our method more effectively maintains both local and global rigidity.

b) Quantitative results: The metrics reported in TABLE I and II are consistent with the qualitative comparison. In TABLEI, all learning-based methods consistently outperform classical methods. Our method with initial correspondences from Lepard [20] performs the best on both the EPE and the OR metrics, when Lepard+LNDP[24] trails closely behind. However, our method is able to significantly outperform LNDP on the 4DLoMatch benchmark even without Lepard initilization, with a 40% decrease in EPE and 7% less in OR.

D. Ablation study

This section validates the effectiveness of various design choices, including our recurrent strategy, the SE(3) regularization layer, and the initialization strategy. We systematically remove or alter certain components from our pipeline and report the resultant changes in the EPE metric.

1) Recurrent update: In this experiment, we vary the number of iterations of recurrent updates with a fixed voxel reoslution to demonstrate the validity of the recurrent strategy. As shown in Fig. 4, with each recurrent update, the source point cloud is progressively aligned to the target point cloud. We also plot the EPE error on a subset of 7 objects of 4DMatch, as a function of the number of updates, in Fig. 5, where the average EPE consistently drops as more updates are introduced. Interestingly, we note that the EPE does not drop linearly. Instead, most corrections are made between the 7th and 9th updates.

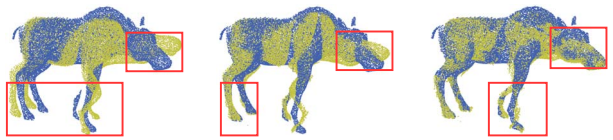

Fig. 4. Visualization of registration result with 0, 6 and 14 iterations of recurrent updates. Note the head and legs are progressively aligned as iteration increases.

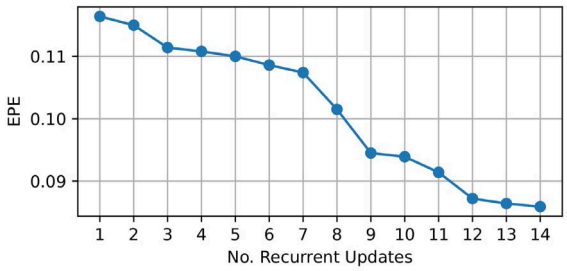

Fig. 5. The relationship between EPE and number of recurrent updates. Error decreases as the recurrent update iterations increase.

2) Local SE(3) Regularization Block: The Local SE(3) regularization block aims to enforce the local rigidity of the transformation. Without such regularization, the predicted flows may still transfer the source points to target geometry, however the rigidity of such transformation is lost and the warped point cloud no longer preserves the structure of source geometry. A comparison is illustrated in Fig.6, where the transformed source points simply collapse onto target surface. The numerical comparison result is shown in TABLE. III, which shows a considerable increase in flow errors.

TABLE III
COMPARISON RESULTS OF EPE W/O REGULARIZATION.

	EPE↓	OR (%)↓
Without Regularization	0.134	27.91
With Regularization	0.086	15.74

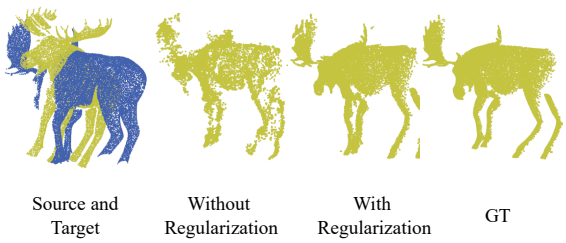

Fig. 6. Illustration of the effectiveness of our SE(3) regularization block. The local structure changes a lot without regularization.

3) Robustness to initialization: In this section, we analysis how the initialization strategy influences the registration performance. To this end, we adjust the matching confidence threshold to allow more sets of correspondences in, effectively introducing noise and outliers to the initial correspondences. Resultant EPE scores are shown in Fig. 7. We note that even with ζ close to zero, the final flow EPE are largely unaffected, suggesting our method is robust to unexact initialization.

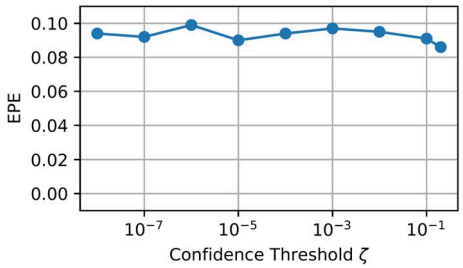

Fig. 7. Comparison results of EPE with different confidence threshold. EPE stays unchanged with different ζ, which means our network is robust to initialization hyper-parameters.

VI. CONCLUSION, LIMITATIONS AND FUTURE WORK

In conclusion, our paper introduces a novel approach to the task of nonrigid point cloud registration, which is a challenging problem due to its ill-posed and non-convex nature. We effectively overcome these challenges by training a deep recurrent network that gradually recovers locally rigid transformations starting from a coarse, global SE(3) transformation. Diverging from traditional RAFT-like strategies, our framework includes the supervision of the matching matrix and incorporates local rigidity regularization to recover the hierarchy of non-rigid deformation. Through extensive experiments, our method obtains state-of-the-art accuracy and is particularly robust to challenging scenarios such as large motion displacement and low overlap. The design choices are validated by ablation study. We hope the success of proposed framework would further advance the field of non-rigid point cloud registration and stimulate new research and applications in 3D computer vision.

However, our method is not without limitations and can be improved in the following ways. Firstly, the local rigidity assumption may not be sufficient to model certain types of motion such as volume expansion and contraction, elastic deformations, and fluid motions. In these cases, the local rigidity assumption, particularly the SO(3) rotation, needs to be replaced with Sim(3) or volume-preserving transformations. Although such modifications are straightforward, their effectiveness remains to be seen. Second, although efficient in practice, our method has quadratic time and space complexity with respect to point sets sizes due to the dense cost matrix. In future work, it is worth investigating whether this can be reduced to linear complexity, by, for instance, low-rank approximations or computing local lookup costs on the fly, as suggested in RAFT [6].

REFERENCES

[1] J. Serafin and G. Grisetti, "Nicp: Dense normal based point cloud registration," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 742–749.

[2] J. Yang, "The thin plate spline robust point matching (tps-rpm) algorithm: A revisit," *Pattern Recognition Letters*, vol. 32, no. 7, pp. 910–918, 2011.

[3] 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*, 2019, pp. 3523–3532.

[4] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 529–537.

[5] Z. Wang, S. Li, H. Howard-Jenkins, V. Prisacariu, and M. Chen, "Flownet3d++: Geometric losses for deep scene flow estimation," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 91–98.

[6] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 402–419.

[7] 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*, 2019, pp. 6411–6420.

[8] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," *Sensor Fusion IV: Control Paradigms and Data Structures*, vol. 1611, pp. 586–606, 1992.

[9] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[10] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[11] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*. IEEE, 2001, pp. 145–152.

[12] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[13] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registration," *Chapel Hill, University of North Carolina*, 2004.

[14] J. Yang, H. Li, and Y. Jia, "Go-icp: Solving 3d registration efficiently and globally optimally," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 1457–1464.

[15] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[16] R. Lai and H. Zhao, "Multi-scale non-rigid point cloud registration using robust sliced-wasserstein distance via laplace-beltrami eigenmap," *arXiv preprint arXiv:1406.3758*, 2014.

[17] G. Puy, A. Boulch, and R. Marlet, "FLOT: Scene Flow on Point Clouds Guided by Optimal Transport," in *European Conference on Computer Vision*, 2020.

[18] G. Trappolini, L. Cosmo, L. Moschella, R. Marin, S. Melzi, and E. Rodolà, "Shape registration in the time of transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5731–5744, 2021.

[19] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, S. Ilic, D. Hu, and K. Xu, "Geotransformer: Fast and robust point cloud registration with geometric transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[20] Y. Li and T. Harada, "Lepard: Learning partial point cloud matching in rigid and deformable scenes," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 5554–5564.

[21] W. Wu, Z. Y. Wang, Z. Li, W. Liu, and L. Fuxin, "Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*. Springer, 2020, pp. 88–107.

[22] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[23] H. Fan and Y. Yang, "Pointrnn: Point recurrent neural network for moving point cloud processing," *arXiv*, vol. 1910.08287, 2019.

[24] Y. Li and T. Harada, "Non-rigid point cloud registration with neural deformation pyramid," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 757–27 768, 2022.

[25] S. Monji-Azad, J. Hesser, and N. Löw, "A review of non-rigid transformations and learning-based 3d point cloud registration methods," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 196, pp. 58–72, 2023.

[26] W. Feng, J. Zhang, H. Cai, H. Xu, J. Hou, and H. Bao, "Recurrent multi-view alignment network for unsupervised surface registration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 297–10 307.

[27] Y. Wei, Z. Wang, Y. Rao, J. Lu, and J. Zhou, "Pv-raft: Point-voxel correlation fields for scene flow estimation of point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6954–6963.

[28] Y. Kittenplon, Y. C. Eldar, and D. Raviv, "Flowstep3d: Model unrolling for self-supervised scene flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4114–4123.

[29] X. Gu, C. Tang, W. Yuan, Z. Dai, S. Zhu, and P. Tan, "Rcp: Recurrent closest point for point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 8216–8226.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[31] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.

[32] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part III 11*. Springer, 2010, pp. 29–43.

[33] 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*, 2017, pp. 2980–2988.

[34] T. T. B. Z. Yang Li, Hikari Takehara and M. Nießner, "4dcomplete: Non-rigid motion estimation beyond the observable surface." *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[35] J.-Y. Bouguet *et al.*, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel corporation*, vol. 5, no. 1-10, p. 4, 2001.

[36] 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.

[37] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.

[38] O. Hirose, "A bayesian formulation of coherent point drift," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 7, pp. 2269–2286, 2020.

[39] Z. Li, S. Niklaus, N. Snavely, and O. Wang, "Neural scene flow fields for space-time view synthesis of dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6498–6508.

[40] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5865–5874.

[41] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[42] J. Huang, T. Birdal, Z. Gojcic, L. J. Guibas, and S.-M. Hu, "Multiway non-rigid point cloud registration via learned functional map synchronization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 2038–2053, 2022.

[43] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.