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

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

I. Ablation study

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

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

Table 1. Influence of warping loss on 4DMatch.

	3DMatch [5]			3DLoMatch [2]		
	FMR↑	IR↑	**RR**↑	FMR↑	IR↑	**RR**↑
$\lambda_w = 0$	**98.0**	63.7	**93.6**	**85.6**	29.5	**69.0**
$\lambda_w = 0.05$	97.8	66.6	92.9	84.1	36.5	67.9
$\lambda_w = 0.1$	97.6	**71.5**	92.9	84.6	**38.8**	68.2

Table 2. Influence of warping loss on 3DMatch.

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

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

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

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

Table 4. Influence of confidence thresholds on 4DMatch and 4DLoMatch. D3Feat [1] and Predator [2] probabilistically sample points either from a saliency heat map or from a machability×overlap heat map (numbers in brackets are the numbers of sampled points). Ours uses the confidence threshold θ_c to get matches from the confidence matrix (c.f. Sec. **??**). All methods apply the mutual nearest neighbor criteria to filter matches. $|\mathcal{K}_{pred}|$ indicates the average number of final predicted correspondences.

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

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

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

Table 5. Ablation study of number of TMP layers.

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

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

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

III. Hyper Parameters

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

RR: Registration Recall
FMR: Feature Matching Recall
NFMR: Non-rigid Feature Matching Recall
MNN: Mutual Nearest Neighbor

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

IV. Time and memory expense

	Predator [2]	Lepard (Ours)
Average time (s)	0.18	**0.10**
Cuda memory (MB)	13,361	**6,595**

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

KPFCN	Self Att. (×2)	Cross Att. (×2)	Matching (×2)	Procrustes (×2)
0.0109	0.0016 (×2)	0.0014 (×2)	0.0023 (×2)	0.0191 (×2)

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

V. KPFCN backbone architecture

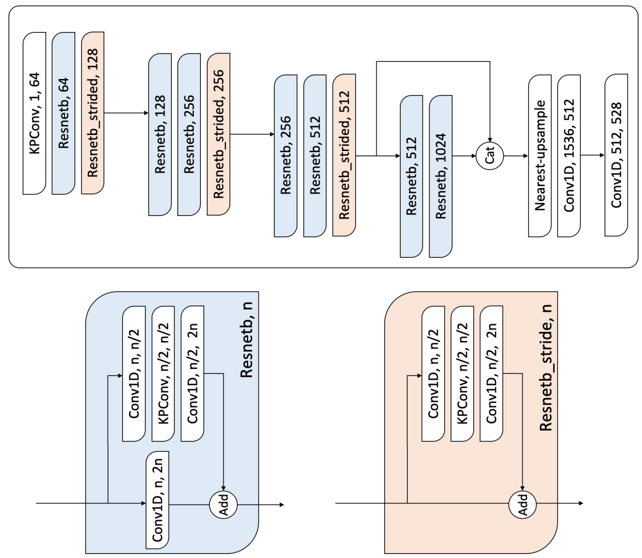

Figure 1. Details of the KPFCN backbone architecture.

VI. Non-Rigid Registration

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

Deformation Model. To represent the dense motion from a source to a target, we adapt the embedded deformation model of Sumner et al. [4]. The non-rigid deformation is parameterized by the deformation graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of node and $\mathcal{E}$ is the set of edge. As shown in Fig. 2, we evenly sample graph nodes $\mathcal{V}$ over the source point cloud surface. Each point in the scene has a 3D location: $\mathbf{g}_i \in \mathbb{R}^3$. The motion of a node $i \in \mathcal{V}$ is parameterized by a translation vector: $\mathbf{t}_i \in \mathbb{R}^3$ and a rotation matrix: $\mathbf{R}_i \in$ SO3. In addition, we represent rotations by $\mathbf{R}_i = \exp(\varphi_i^\wedge)\mathbf{R}_i$, where $\varphi_i = [0,0,0]$ represents the delta of the rotation in axis-angle form. $(\cdot)^\wedge$operator convert a 3-dimensional vector to a 3 × 3 skew-symmetric matrix. $\exp : \mathfrak{so}3 \mapsto$ SO3 map the skew-symmetric matrix to 3 × 3 rotation matrix using the Rodrigues formula. Finally, all unkowns in the graph are

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

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

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

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

$$w_{\mathbf{p},i} = Ce^{\frac{1}{2\gamma^2}||\mathcal{V}_i - \mathbf{p}||_2^2}$$

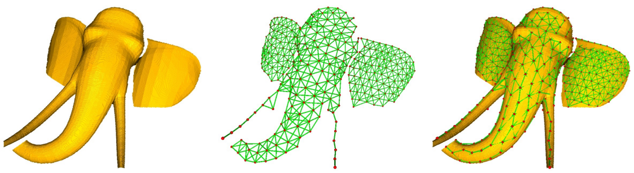

Figure 2. Deformation model. Nodes $\mathcal{V}$ (red dot) are evenly sampled over the source surface. Edges $\mathcal{E}$ (green lines) are computed between nodes based on geodesic connectivity. The point cloud examples in 4DMatch are obtained from depth images. To compute geodesic distance, we construct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

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

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

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

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

We use ARAP [3] as the regularization term

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

The total energy function is

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

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

$$\frac{\partial \mathcal{W}(\mathbf{p})}{\partial \varphi_i} = -w_{\mathbf{p},i}(\mathbf{R}_i(\mathbf{p} - \mathbf{g}_i))^\wedge$$

$$\frac{\partial \mathcal{W}(\mathbf{p})}{\partial \mathbf{t}_i} = w_{\mathbf{p},i}\mathbf{I}_3$$

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

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

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

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

$$\frac{\partial \mathbf{r}^{corr}_{(\mathbf{p}_s,\mathbf{p}_t)}}{\partial \mathbf{t}_i} = \sqrt{\lambda_c} c_{(\mathbf{p}_s,\mathbf{p}_t)} w_{\mathbf{p},i}\mathbf{I}_3$$

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

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

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

$$\frac{\partial \mathbf{r}^{reg}_{(i,j)}}{\partial \varphi_i} = -\sqrt{\lambda_a}(\mathbf{R}_i(\mathbf{g}_j - \mathbf{g}_i))^\wedge$$

$$\frac{\partial \mathbf{r}^{reg}_{(i,j)}}{\partial \mathbf{t}_i} = \sqrt{\lambda_a}\mathbf{I}_3$$

$$\frac{\partial \mathbf{r}^{reg}_{(i,j)}}{\partial \mathbf{t}_j} = -\sqrt{\lambda_a}\mathbf{I}_3$$

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

$$\mathbf{J} = \left[\begin{array}{ccc|ccc}
\frac{\partial \mathbf{r}^{corr}_1}{\partial \varphi_1} & \cdots & \frac{\partial \mathbf{r}^{corr}_1}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}^{corr}_1}{\partial \mathbf{t}_1} & \cdots & \frac{\partial \mathbf{r}^{corr}_1}{\partial \mathbf{t}_{|\mathcal{V}|}} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\frac{\partial \mathbf{r}^{corr}_{|\mathcal{K}|}}{\partial \varphi_1} & \cdots & \frac{\partial \mathbf{r}^{corr}_{|\mathcal{K}|}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}^{corr}_{|\mathcal{K}|}}{\partial \mathbf{t}_1} & \cdots & \frac{\partial \mathbf{r}^{corr}_{|\mathcal{K}|}}{\partial \mathbf{t}_{|\mathcal{V}|}} \\
\hline
\frac{\partial \mathbf{r}^{reg}_1}{\partial \varphi_1} & \cdots & \frac{\partial \mathbf{r}^{reg}_1}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}^{reg}_1}{\partial \mathbf{t}_1} & \cdots & \frac{\partial \mathbf{r}^{reg}_1}{\partial \mathbf{t}_{|\mathcal{V}|}} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\frac{\partial \mathbf{r}^{reg}_{|\mathcal{E}|}}{\partial \varphi_1} & \cdots & \frac{\partial \mathbf{r}^{reg}_{|\mathcal{E}|}}{\partial \varphi_{|\mathcal{V}|}} & \frac{\partial \mathbf{r}^{reg}_{|\mathcal{E}|}}{\partial \mathbf{t}_1} & \cdots & \frac{\partial \mathbf{r}^{reg}_{|\mathcal{E}|}}{\partial \mathbf{t}_{|\mathcal{V}|}}
\end{array}\right]$$

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

$$\mathbf{r} = \left[\begin{array}{c}
\mathbf{r}^{corr}_1 \\
\vdots \\
\mathbf{r}^{corr}_{|\mathcal{K}|} \\
\mathbf{r}^{reg}_1 \\
\vdots \\
\mathbf{r}^{reg}_{|\mathcal{E}|}
\end{array}\right]$$

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

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

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

The above linear system is solved using LU decomposition.

VII. Qualitative Results

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

	elephant			dragon		
	EPE↓	Acc5↑	Acc10↑	EPE↓	Acc5↑	Acc10↑
N-ICP	0.166	22.4	41.5	0.325	4.5	17.8
Predator [2] + N-ICP	0.092	55.7	66.0	0.514	29.6	32.1
Ours + N-ICP	**0.018**	**90.6**	**98.0**	**0.038**	**68.0**	**96.0**

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

	moose			mutant		
	EPE↓	Acc5↑	Acc10↑	EPE↓	Acc5↑	Acc10↑
N-ICP	0.728	0.1	0.7	0.52	0.0	0.6
Predator [2] + N-ICP	0.0283	86.9	99.4	0.217	44.6	60.1
Ours + N-ICP	**0.0263**	**88.5**	**99.9**	**0.119**	**62.6**	**71.4**

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

References

[1] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6359–6367, 2020. 1

[2] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler. Predator: Registration of 3d point clouds with low overlap. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4267–4276, 2021. 1, 2, 4

[3] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry Processing*, volume 4, pages 109–116, 2007. 3

[4] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. *ACM Transactions on Graphics (TOG)*, 26(3):80, 2007. 2

[5] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1802–1811, 2017. 1

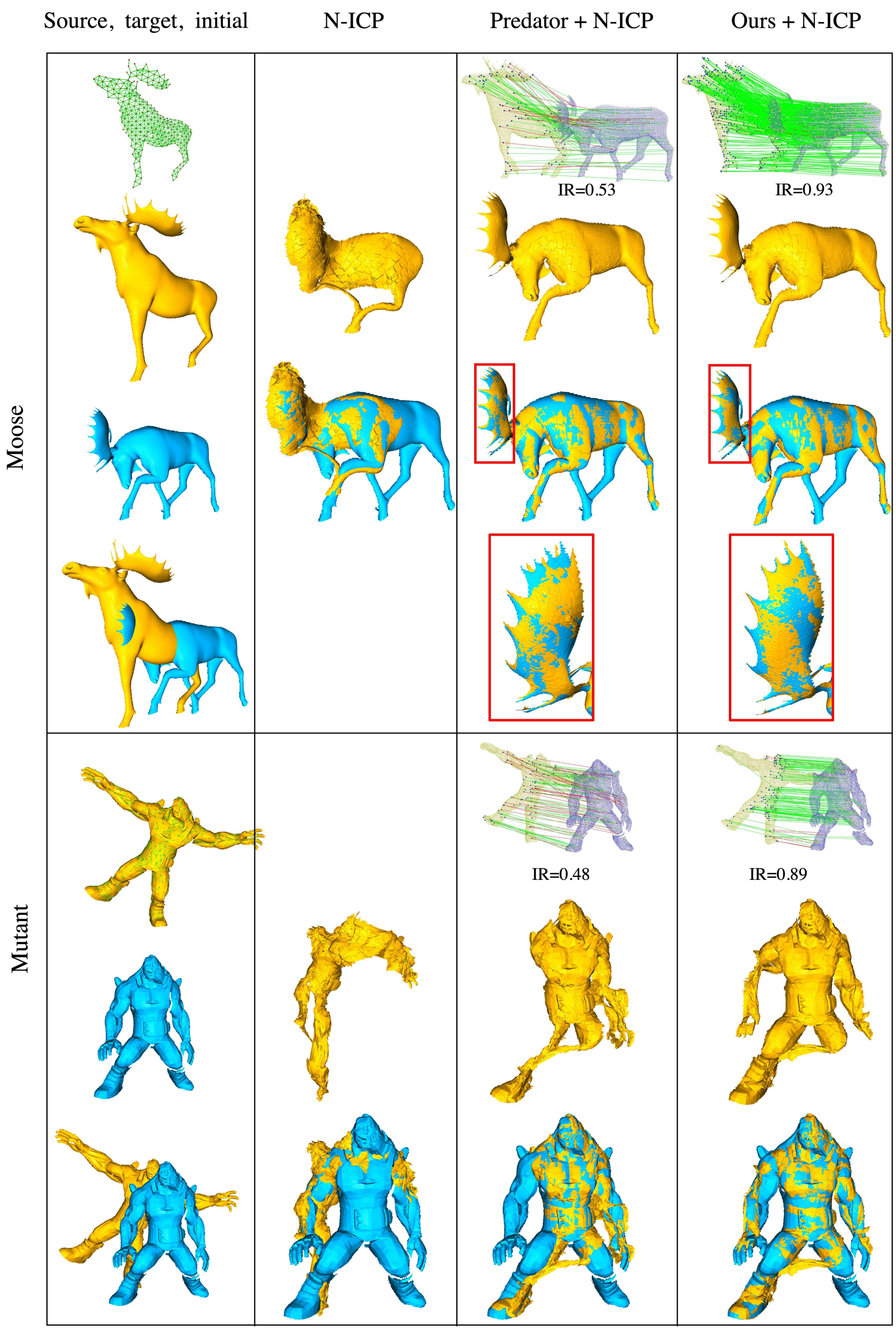

Figure 3. Qualitative point cloud matching and registration results on 4DMatch. The inlier threshold is set to $4cm$. The N-ICP-based refinement can remedy outliers to a certain extent if the outlier matches are not too far away from the ground truth (see the results of *Predator + N-ICP* in the Moose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

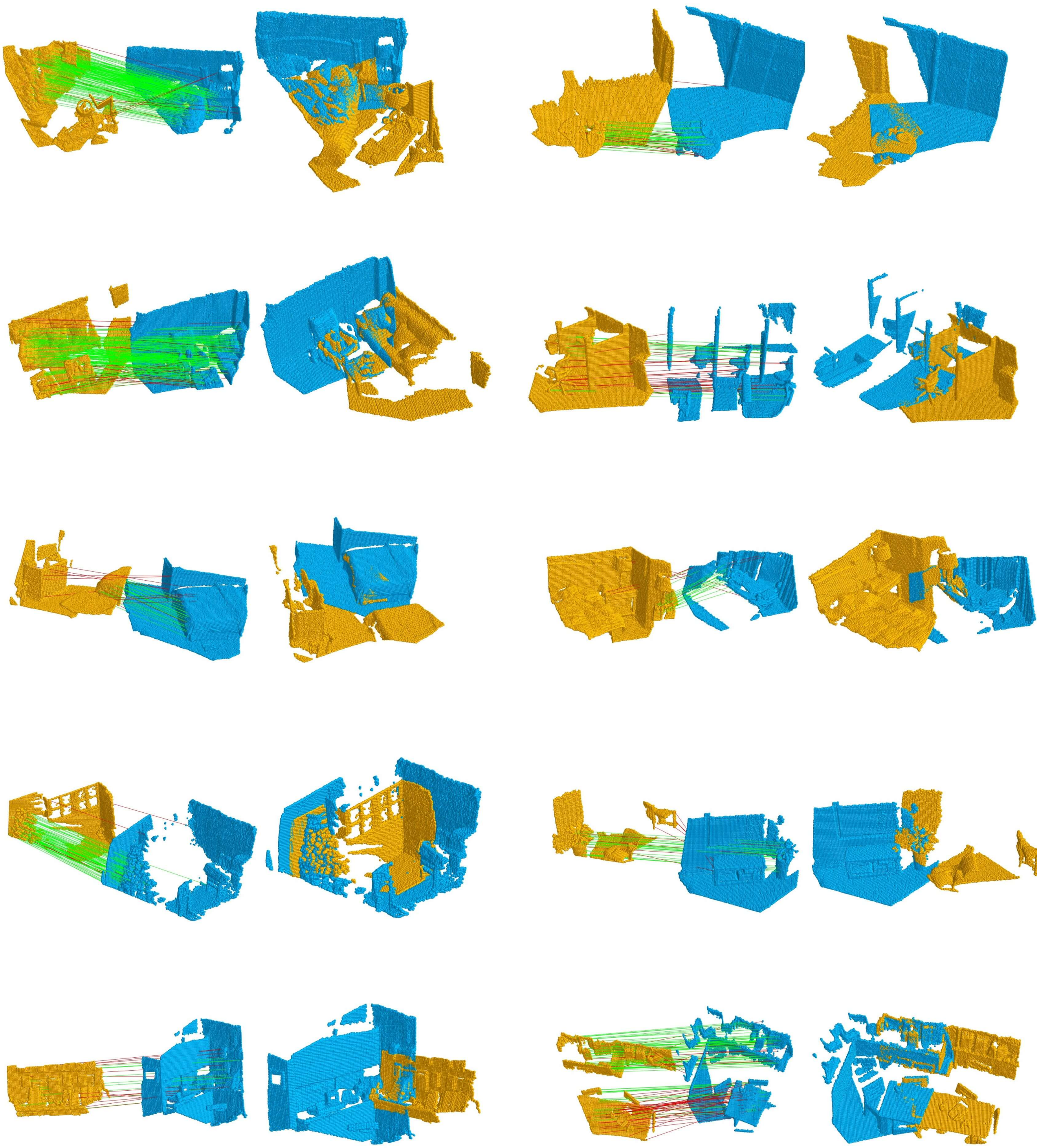

Figure 4. Qualitative point cloud matching and registration results on 3DLoMatch.