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.

<table>
<thead>
<tr>
<th>$\lambda_w$</th>
<th>3DMatch (NFMR↑ IR↑)</th>
<th>4DLoMatch (NFMR↑ IR↑)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>82.9 82.4</td>
<td>62.1 52.2</td>
</tr>
<tr>
<td>0.05</td>
<td>85.3 83.9</td>
<td>65.1 54.5</td>
</tr>
<tr>
<td>0.1*</td>
<td>83.7 82.7</td>
<td>66.9 55.7</td>
</tr>
</tbody>
</table>

Table 1. Influence of warping loss on 4DMatch.

<table>
<thead>
<tr>
<th>$\lambda_w$</th>
<th>3DMatch [5]</th>
<th>3DLoMatch [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>98.0 63.7 93.6</td>
<td>85.6 29.5 69.0</td>
</tr>
<tr>
<td>0.05</td>
<td>97.8 66.6 92.9</td>
<td>84.1 36.5 67.9</td>
</tr>
<tr>
<td>0.1</td>
<td>97.6 71.5 92.9</td>
<td>84.6 38.8 68.2</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>3DMatch (RR↑)</th>
<th>4DLoMatch (RR↑)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta_c = 0.05$</td>
<td>93.6 69.0</td>
</tr>
<tr>
<td>$\theta_c = 0.1$</td>
<td>92.3 67.9</td>
</tr>
<tr>
<td>$\theta_c = 0.15$</td>
<td>91.7 67.0</td>
</tr>
<tr>
<td>$\theta_c = 0.2$</td>
<td>91.2 65.3</td>
</tr>
</tbody>
</table>

Table 3. Influence of confidence thresholds on 3DMatch and 4DLoMatch.

| Method      | 3DMatch ($|K_{pred}|$) NFMR↑ IR↑ | 4DLoMatch ($|K_{pred}|$) NFMR↑ IR↑ |
|-------------|-------------------|-------------------|
| D3Feat (1000) | 267 51.6 52.7     | 204 23.6 21.2     |
| D3Feat (3000) | 532 55.5 54.7     | 379 27.4 21.5     |
| D3Feat (5000) | 697 56.1 55.5     | 473 28.9 21.3     |
| Predator (1000) | 273 53.3 60.0   | 205 30.6 29.8     |
| Predator (3000) | 534 56.4 60.4   | 372 32.1 27.5     |
| Predator (5000) | 698 56.8 59.5   | 480 32.1 25.0     |
| Ours ($\theta_c=0.2$) | 523 82.2 85.4 | 325 63.1 60.4 |
| Ours ($\theta_c=0.1$) | 596 83.7 82.9 | 407 66.9 55.7 |
| Ours ($\theta_c=0.05$) | 624 83.9 80.9 | 447 67.6 52.5 |

Table 4. Influence of confidence thresholds on 4DMatch and 4DLoMatch. D3Feat [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 $\theta_c$ to get matches from the confidence matrix (c.f. Sec. ??). All methods apply the mutual nearest neighbor criteria to filter matches. $|K_{pred}|$ indicates the average number of final predicted correspondences.

<table>
<thead>
<tr>
<th>Number of TMP layer</th>
<th>Number of Repositioning layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rigid</td>
<td></td>
</tr>
<tr>
<td>RR(%)↑ on 3DMatch</td>
<td>93.6 92.8 93.0</td>
</tr>
<tr>
<td>RR(%)↑ on 3DLoMatch</td>
<td>69.0 68.2 68.8</td>
</tr>
<tr>
<td>Training Time (hour)</td>
<td>20 25 31</td>
</tr>
<tr>
<td>Deformable</td>
<td></td>
</tr>
<tr>
<td>NFMR(%)↑ on 4DMatch</td>
<td>83.7 85.9 84.5</td>
</tr>
<tr>
<td>NFMR(%)↑ on 4DLoMatch</td>
<td>66.9 68.1 59.6</td>
</tr>
<tr>
<td>Training Time (hour)</td>
<td>18 21 24</td>
</tr>
</tbody>
</table>

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 $p = (x, y, z) \in \mathbb{R}^3$ and a feature $x \in \mathbb{R}^d$, the multiplication $\Theta(p)x$ can be efficiently realized by

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

III. Hyper Parameters

<table>
<thead>
<tr>
<th>Metric</th>
<th>3DMatch</th>
<th>4DMatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inlier threshold</td>
<td>0.1m</td>
<td>0.04m</td>
</tr>
<tr>
<td>RR threshold</td>
<td>0.2m</td>
<td>-</td>
</tr>
<tr>
<td>FMR threshold</td>
<td>5%</td>
<td>-</td>
</tr>
<tr>
<td>NFMR threshold</td>
<td>-</td>
<td>0.04m</td>
</tr>
<tr>
<td>Match Prediction</td>
<td>Confidence threshold $\theta_c$</td>
<td>0.05</td>
</tr>
<tr>
<td>Apply MNN</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>KPFCN Config</td>
<td>Input subsampling radius</td>
<td>0.025m</td>
</tr>
<tr>
<td>Supervision</td>
<td>GT match radius</td>
<td>0.06m</td>
</tr>
<tr>
<td>Warping loss weight $\lambda_w$</td>
<td>0.0</td>
<td>0.1</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>Predator [2]</th>
<th>Lepard (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average time (s)</td>
<td>0.18</td>
<td>0.10</td>
</tr>
<tr>
<td>Cuda memory (MB)</td>
<td>13,361</td>
<td>6,595</td>
</tr>
</tbody>
</table>

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.

V. KPFCN backbone architecture

![Figure 1. Details of the KPFCN backbone architecture.](image)

VI. Non-Rigid Registration

This section introduces 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 $G = (V, E)$, where $V$ is the set of node and $E$ is the set of edge. As shown in Fig. 2, we evenly sample graph nodes $V$ over the source point cloud surface. Each point in the scene has a 3D location: $g_i \in \mathbb{R}^3$. The motion of a node $i \in V$ is parameterized by a translation vector: $t_i \in \mathbb{R}^3$ and a rotation matrix: $R_i \in SO3$. In addition, we represent rotations by $R_i = \exp(\varphi_i)^T R_0$, where $\varphi_i = [0, 0, 0]$ represents the delta of the rotation in axis-angle form. $\exp : so3 \mapsto SO3$ map the skew-symmetric matrix to 3×3 rotation matrix using the Rodrigues formula. Finally, all unknowns in the graph are

$$
G = (\varphi_1, \cdots, \varphi_{|V|}, t_1, \cdots, t_{|V|})
$$

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

$$
W(p) = \sum_{i \in V} w_{p,i}(R_i(p - g_i) + g_i + t_i)
$$

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

$$
w_{p,i} = Ce^{\frac{1}{2\sigma^2}||V_i - p||^2}
$$
where $\gamma$ 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
\[
\sum_{s_t \in V} w_{p,i} = 1
\]

**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 $c_{(p_s, p_t)}$ where $(p_s, p_t) \in \mathcal{K}$. Correspondence term is defined as
\[
E_{corr}(\mathcal{G}) = \sum_{(p_s, p_t) \in \mathcal{K}} c_{(p_s, p_t)}^2 \|W(p_s) - p_t\|^2
\]
We use ARAP [3] as the regularization term
\[
E_{reg}(\mathcal{G}) = \sum_{(i,j) \in \mathcal{E}} \|R_i(g_j - g_i) + g_i + t_i - (g_j + t_j)\|^2
\]
The total energy function is
\[
E_{reg}(\mathcal{G}) = \lambda_c E_{corr}(\mathcal{G}) + \lambda_a E_{reg}(\mathcal{G})
\]

**Residual and Partial Derivatives.** The followings show the residuals and partial derivatives for optimization. Derivative of the wrapping function $W$
\[
\frac{\partial W(p)}{\partial \varphi_i} = -w_{p,i}(R_i(p - g_i))^\wedge
\]
\[
\frac{\partial W(p)}{\partial t_i} = w_{p,i} I_3
\]
where $I_3$ is the $3 \times 3$ identity matrix. Residual term for a correspondence $(p_s, p_t) \in \mathcal{K}$
\[
r_{corr}^{(p_s, p_t)} = \sqrt{\lambda_c c_{(p_s, p_t)}} (W(p_s) - p_t)
\]
Derivative of correspondence residual $r_{corr}^{(p_s, p_t)}$
\[
\frac{\partial r_{corr}^{(p_s, p_t)}}{\partial \varphi_i} = -\sqrt{\lambda_c c_{(p_s, p_t)}} w_{p,i} (R_i(p_s - g_i))^\wedge
\]
\[
\frac{\partial r_{corr}^{(p_s, p_t)}}{\partial t_i} = \sqrt{\lambda_c c_{(p_s, p_t)}} w_{p,i} I_3
\]
Residual term for regularization term $(i, j) \in \mathcal{E}$
\[
r_{reg}^{(i, j)} = \sqrt{\lambda_r (R_i(g_j - g_i) + g_i + t_i - (g_j + t_j))}
\]
Derivative of regularization term $r_{reg}^{(i, j)}$
\[
\frac{\partial r_{reg}^{(i, j)}}{\partial \varphi_i} = -\sqrt{\lambda_r (R_i(g_j - g_i))^\wedge}
\]
\[
\frac{\partial r_{reg}^{(i, j)}}{\partial t_i} = \sqrt{\lambda_r} I_3
\]
\[
\frac{\partial r_{reg}^{(i, j)}}{\partial t_j} = -\sqrt{\lambda_r} I_3
\]
The full Jacobian matrix $J \in \mathbb{R}^{(3|\mathcal{K}|+|\mathcal{E}|) \times |\mathcal{V}|}$ is shown as
\[
J =
\]
where $|\mathcal{V}|$ is the number of node, $|\mathcal{K}|$ is number of correspondence, $|\mathcal{E}|$ is the number of graph edge. Each block in $J$ is a $3 \times 3$ matrix. For the sparse nature of this problem, most blocks are zeros. The full residual vector $r \in \mathbb{R}^{3|\mathcal{K}|+|\mathcal{E}|}$ is shown as
\[
r =
\]
where each block is a $3 \times 1$ vector. The total length is $(|K| + |E|) \times 3$.

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

$$ J^T J \Delta G = J^T r $$

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.

<table>
<thead>
<tr>
<th></th>
<th>EPE↓</th>
<th>Acc5↑</th>
<th>Acc10↑</th>
<th>EPE↓</th>
<th>Acc5↑</th>
<th>Acc10↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-ICP</td>
<td>0.166</td>
<td>22.4</td>
<td>41.5</td>
<td>0.325</td>
<td>4.5</td>
<td>17.8</td>
</tr>
<tr>
<td>Predator [2] + N-ICP</td>
<td>0.092</td>
<td>55.7</td>
<td>66.0</td>
<td>0.514</td>
<td>29.6</td>
<td>32.1</td>
</tr>
<tr>
<td>Ours + N-ICP</td>
<td><strong>0.018</strong></td>
<td><strong>90.6</strong></td>
<td><strong>98.0</strong></td>
<td><strong>0.038</strong></td>
<td><strong>68.0</strong></td>
<td><strong>96.0</strong></td>
</tr>
</tbody>
</table>

**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%).

<table>
<thead>
<tr>
<th></th>
<th>EPE↓</th>
<th>Acc5↑</th>
<th>Acc10↑</th>
<th>EPE↓</th>
<th>Acc5↑</th>
<th>Acc10↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-ICP</td>
<td>0.728</td>
<td>0.1</td>
<td>0.7</td>
<td>0.52</td>
<td>0.0</td>
<td>0.6</td>
</tr>
<tr>
<td>Predator [2] + N-ICP</td>
<td>0.0283</td>
<td>86.9</td>
<td>99.9</td>
<td>0.217</td>
<td>44.6</td>
<td>60.4</td>
</tr>
<tr>
<td>Ours + N-ICP</td>
<td><strong>0.0263</strong></td>
<td><strong>88.5</strong></td>
<td><strong>99.9</strong></td>
<td><strong>0.119</strong></td>
<td><strong>62.6</strong></td>
<td><strong>71.4</strong></td>
</tr>
</tbody>
</table>

**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**


Figure 3. Qualitative point cloud matching and registration results on 4DMatch. The inlier threshold is set to 4 cm. 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 cannot handle outliers that connect 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.