Rethinking 2D-3D Registration: A Novel Network for High-Value Zone Selection and Representation Consistency Alignment

Supplementary Material

1. t-SNE and MMD

t-SNE. t-Distributed Stochastic Neighbor Embedding (t-SNE) [8] is a nonlinear dimensionality reduction technique used primarily for data visualization. It is particularly effective for visualizing high-dimensional datasets by embedding them into two or three dimensions. t-SNE aims to preserve the local structure of data points by modeling similar objects with nearby points and dissimilar objects with distant points. This method is beneficial for visualizing multimodal tasks, allowing for intuitive insights into complex datasets that span various domains such as images, text, and audio.

MMD. Maximum Mean Discrepancy (MMD) is a statistical method used to compare two probability distributions. It is a non-parametric technique that measures the difference between distributions by mapping data into a high-dimensional feature space using a kernel function. MMD is widely used in various machine learning applications, such as generative adversarial networks (GANs) [6], domain adaptation, and distribution testing. The core idea of MMD is to compute the distance between the means of two distributions in a reproducing kernel Hilbert space (RKHS) [1]. Given two distributions I and Q, the MMD is defined as:

$$\text{MMD}(I, Q) = \left\| \mathbb{E}_{x\sim I}[\phi(x)] - \mathbb{E}_{y\sim Q}[\phi(y)] \right\|_{\mathcal{H}}, \quad (1)$$

where ϕ is the feature mapping function induced by a kernel, and $\mathcal{H}$ is the RKHS.

MMD is applied in various areas, including generative models for evaluating the similarity between the distributions of generated and real data, domain adaptation for reducing distribution shifts between source and target domains, and hypothesis testing for determining if two samples are drawn from the same distribution.

2. Positional embedding

We augment the 2D and 3D features with their positional information before the attention layer.

$$\hat{F}^{\mathcal{I}}_{\text{pos}} = \hat{F}^{\mathcal{I}} + \phi(\hat{Q}), \quad \hat{F}^{\mathcal{P}}_{\text{pos}} = \hat{F}^{\mathcal{P}} + \phi(\hat{P}). \quad (2)$$

The Fourier embedding function $\phi(x)$ [5] encodes positional information by transforming it into a sequence of sine and cosine terms:

$$\phi(x) = \left[x, \sin(2^0 x), \cos(2^0 x), \ldots, \sin(2^{L-1} x), \cos(2^{L-1} x)\right], \quad (3)$$

where L is the length of the embedding. This transformation incorporates spatial positioning into the features. To facilitate further computations, the first two spatial dimensions of the 2D features are flattened, making the augmented features $\hat{F}^{\mathcal{I}}_{\text{pos}}$ and $\hat{F}^{\mathcal{P}}_{\text{pos}}$ ready for subsequent processing.

3. Network architecture

We utilize a 4-stage ResNet [2] with a Feature Pyramid Network (FPN) [4] as the image backbone. The output channels for each stage are $\{128, 128, 256, 512\}$. The input images have a resolution of 480×640 pixels, which is downsampled to 60×80 in the coarsest level for efficiency. For the 3D backbone, a 4-stage KPFCNN [7] is employed, with output channels configured as $\{128, 256, 512, 1024\}$. Point clouds are voxelized with an initial voxel size of 2.5 cm, which doubles at each stage.

At the coarse level, 2D features are resized to 24×32 pixels before being fed into the transformer to enhance computational efficiency. Each transformer layer has 256 feature channels, 4 attention heads, and uses ReLU activation functions. In the patch pyramid setup, the coarsest level begins with $H_0 = 6$ and $W_0 = 8$, expanding through 3 pyramid levels: $\{6 \times 8, 12 \times 16, 24 \times 32\}$. At the fine level, both 2D and 3D features are projected into a 128-dimensional space for feature matching.

To address significant misalignments caused by structurally similar but non-overlapping regions, we incorporate ground-truth supervision during training by evaluating the overlap ratio between the annotated image-point cloud correspondences, using the dataset-provided ground-truth pose. This overlap measure is computed based on the local neighborhoods of image keypoints and point cloud nodes, which are projected according to the ground-truth pose. However, relying solely on this overlap criterion results in sparse and discontinuous reward signals, as only a limited number of correspondences exhibit valid overlaps, leaving most candidate pairs unlabeled. We incorporate the overlap signal in combination with the rotation-invariant geometric similarity to form the final reward signal for policy optimization. This addition helps prevent large misalignments by providing supervisory information when available, without significantly influencing the reinforcement learning process in regions lacking explicit annotations. As such, it ensures the network benefits from strong supervision in well-annotated areas, while still receiving dense and consistent feedback in less annotated regions, thereby supporting sta-

Algorithm 1: Policy Network for Candidate Correspondence Selection

Input: Candidates $\{\mathcal{C}_t\}_{t=1}^{T}$; pooled descriptors $f_t^{img} \in \mathbb{R}^{d_{img}}$, $f_t^{pcd} \in \mathbb{R}^{d_{pcd}}$; cosine similarity $cos_t \in \mathbb{R}$; threshold τ (default 0.5); mode $\in \{\texttt{train}, \texttt{infer}\}$.

Output: Action mask $\mathbf{a} \in \{0,1\}^T$ and probabilities $\mathbf{p} \in [0,1]^T$.

for $t \leftarrow 1$ **to** T **do**

`// (1) State representation`

$s_t \leftarrow [\, f_t^{img};\ f_t^{pcd};\ cos_t\,] \in \mathbb{R}^{d_{img}+d_{pcd}+1}$

`// (2) Two-layer MLP policy network`

$\text{logit}_t \leftarrow W_2\,\text{ReLU}(W_1 s_t + b_1) + b_2$

`//` Dimensions: $W_1 : \mathbb{R}^{257} \to \mathbb{R}^{128}$, $W_2 : \mathbb{R}^{128} \to \mathbb{R}^1$

$p_t \leftarrow \sigma(\text{logit}_t) = \pi_\theta(a_t = 1 \mid s_t)$

if *mode* = `train` **then**

Sample $a_t \sim \text{Bernoulli}(p_t)$

else

$a_t \leftarrow \mathbb{I}[p_t > \tau]$

Set $\mathbf{a}[t] \leftarrow a_t$, $\mathbf{p}[t] \leftarrow p_t$

return $\mathbf{a}, \mathbf{p}$

Algorithm 2: Candidate Selection with REINFORCE and Reward Influence

Input: Candidates $\{c_t\}_{t=1}^{T}$ with states $\{s_t\}_{t=1}^{T}$; policy network $\text{PolicyMLP}_\theta(\cdot)$; baseline z; matching loss L_x; projection terms P_{roi}, P_{rop}; weights λ_1, λ_2.

Output: Selected set $\mathcal{S}$ and updated policy parameters θ.

`// --- (1) Candidate selection by Bernoulli policy ---`

Initialize $\mathcal{S} \leftarrow \emptyset$

for $t \leftarrow 1$ **to** T **do**

$\text{logit}_t \leftarrow \text{PolicyMLP}_\theta(s_t)$

$p_t \leftarrow \sigma(\text{logit}_t) = \pi_\theta(a_t = 1 \mid s_t)$

Sample $a_t \sim \text{Bernoulli}(p_t)$

if $a_t = 1$ **then**

$\mathcal{S} \leftarrow \mathcal{S} \cup \{c_t\}$

`// --- (2) Reward and its influence on selection ---`

Compute matching loss L_x using selected set $\mathcal{S}$

Compute reward $R \leftarrow \dfrac{1}{L_x} + \lambda_1 P_{roi} + \lambda_2 P_{rop}$

`// REINFORCE loss (baseline` z`) and update direction`

$L_R \leftarrow -(R - z) \sum_{t=1}^{T} \Big(a_t \log p_t + (1 - a_t) \log(1 - p_t) \Big)$

`//` For Bernoulli $p_t = \sigma(\texttt{logit}_t)$: $\frac{\partial}{\partial \texttt{logit}_t} \log \pi_\theta(a_t|s_t) = a_t - p_t$

`//` Thus: $\Delta \texttt{logit}_t \propto (R - z)(a_t - p_t)$, i.e., $R > z$ reinforces sampled actions, $R < z$ suppresses them.

Update θ by minimizing L_R

return $\mathcal{S}, \theta$

ble learning.

We define ground truth using bilateral overlap [3]. A patch pair is positive if both the 2D and 3D overlap ratios are at least 30%, and negative if both are below 20%. The 2D and 3D overlap ratios are used as λ_p, while λ_n is set to 1. At the fine level, a pixel-point pair is positive if the 3D distance is below 3.75 cm and the 2D distance is under 8 pixels, and negative if the 3D distance exceeds 10 cm or the 2D distance is over 12 pixels. Scaling factors are set to 1. Pairs not meeting these criteria are ignored as the safe region during training. The margins are set to $\Delta_p = 0.1$ and $\Delta_n = 1.4$.

4. Pseudocode Representation

References

[1] Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[3] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad 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. 2

[4] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 1

[5] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1

[6] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 1

[7] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud,

Algorithm 3: Coarse-to-Fine Two-Stage Matching

Require: Image $\mathbf{I}$, point cloud $\mathbf{P}$; multi-scale image patch features $\{\mathbf{f}_i^I\}$; multi-level point cloud patch/node features $\{\mathbf{f}_j^{P,l}\}_{l=1}^{L}$; fine-level features $\mathbf{F}^I, \mathbf{F}^P$; neighborhood sizes K_i, K_c; Top-k parameter k; threshold τ.

Ensure: Final fine correspondences $\mathcal{M}$.

Stage 1: Coarse matching (patch/node level)

Construct 3D nodes $\{\mathcal{N}_j^P\}$ by grouping fine points around each coarse point: $\mathcal{N}_j^P \leftarrow \mathrm{KNN}(\mathbf{P}_{fine}, \mathbf{p}_{coarse,j}^P, K_c)$.

Partition image into 2D nodes on multi-scale grids to obtain $\{\mathcal{N}_i^I\}$ and patch features $\{\mathbf{f}_i^I\}$.

forall *2D node* i *and 3D node* j **do**

Compute cosine distance at each point-cloud level l: $d_{ij}^{(l)} \leftarrow 1 - \cos(\mathbf{f}_i^I, \mathbf{f}_j^{P,l})$.

Aggregate multi-level distances (max rule): $d_{ij} \leftarrow \max_{l=1}^{L} d_{ij}^{(l)}$.

Obtain coarse correspondences via mutual Top-k: $\mathcal{C} \leftarrow \textsc{MutualTopK}(\{d_{ij}\}, k)$.

Stage 2: Fine matching (pixel–point level)

Initialize $\mathcal{M} \leftarrow \emptyset$. **forall** $(i, j) \in \mathcal{C}$ **do**

Gather local pixels and local 3D points: $\mathcal{P}_i \leftarrow \textsc{PixelsInNode}(\mathcal{N}_i^I), \quad \mathcal{Q}_j \leftarrow \mathcal{N}_j^P$.

Build local similarity matrix $\mathbf{S}_{ij} \in \mathbb{R}^{K_i \times K_c}$: $\mathbf{S}_{ij}(u, v) \leftarrow \cos\big(\mathbf{F}^I(\mathcal{P}_i[u]), \mathbf{F}^P(\mathcal{Q}_j[v])\big)$.

Select fine correspondences with thresholded mutual Top-k: $\mathcal{M}_{ij} \leftarrow \textsc{MutualTopKThresh}(\mathbf{S}_{ij}, k, \tau)$.

$\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{M}_{ij}$.

Deduplicate global correspondences: $\mathcal{M} \leftarrow \textsc{Unique}(\mathcal{M})$.

return $\mathcal{M}$.

Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. 1

[8] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 1

Algorithm 4: High-Value Zone Reinforced Selection

Input: Image features $\mathbf{F}_i \in \mathbb{R}^{h \times w \times c}$, Point cloud features $\mathbf{F}_p \in \mathbb{R}^{n \times c}$

Output: Selected high-value regions in both image and point cloud

Initialize:

- Self- and cross-attention transformers for feature refinement.
- Candidate set: $\mathcal{C} = (\mathcal{C}_i, \mathcal{C}_p)$ with $\mathcal{C}_i = \{1, \ldots, h \times w\}$ and $\mathcal{C}_p = \{1, \ldots, n\}$.
- Policy network π_θ for candidate selection.
- Moving average of past rewards z.

for $epoch = 1$ ***to*** *End* **do**

For each sample in the batch: for $i \in \mathcal{C}_i$ **do**

Compute state representation s_t for image candidate i ;

Select action $a_t \sim \pi_\theta(a_t \mid s_t)$, where $a_t \in \{0, 1\}$;

for $p \in \mathcal{C}_p$ **do**

Compute state representation s_t for point cloud candidate p ;

Select action $a_t \sim \pi_\theta(a_t \mid s_t)$, where $a_t \in \{0, 1\}$;

Reward Calculation:

$$R = \frac{1}{L_x} + \lambda_1 \cdot Pro_i + \lambda_2 \cdot Pro_p$$

where:

$$Pro_i = \sum_{a=1}^{h \times w} [\mathbf{1}(f(I_a) \in P_s) - \mathbf{1}(f(I_a) \notin P_s)]$$

$$Pro_p = \sum_{b=1}^{n} [\mathbf{1}(f(P_b) \in I_s) - \mathbf{1}(f(P_b) \notin I_s)]$$

Policy Update:

$$L_R = -\mathbb{E}\left[(R - z) \sum_{t=1}^{T} \nabla_\theta \left(a_t \log p_t + (1 - a_t) \log(1 - p_t)\right)\right]$$