Sampling Network Guided Cross-Entropy Method for Unsupervised Point Cloud Registration

Haobo Jiang, Yaqi Shen, Jin Xie*, Jun Li, Jianjun Qian*, and Jian Yang

PCA Lab, Nanjing University of Science and Technology, China
{jiahao, syq, csjxie, junli, csjqian, csjyang}@njust.edu.cn

Abstract

In this paper, by modeling the point cloud registration task as a Markov decision process, we propose an end-to-end deep model embedded with the cross-entropy method (CEM) for unsupervised 3D registration. Our model consists of a sampling network module and a differentiable CEM module. In our sampling network module, given a pair of point clouds, the sampling network learns a prior sampling distribution over the transformation space. The learned sampling distribution can be used as a “good” initialization of the differentiable CEM module. In our differentiable CEM module, we first propose a maximum consensus criterion based alignment metric as the reward function for the point cloud registration task. Based on the reward function, for each state, we then construct a fused score function to evaluate the sampled transformations, where we weight the current and future rewards of the transformations. Particularly, the future rewards of the sampled transforms are obtained by performing the iterative closest point (ICP) algorithm on the transformed state. By selecting the top-k transformations with the highest scores, we iteratively update the sampling distribution. Furthermore, in order to make the CEM differentiable, we use the sparse-max function to replace the hard top-k selection. Finally, we formulate a Geman-McClure estimator based loss to train our end-to-end registration model. Extensive experimental results demonstrate the good registration performance of our method on benchmark datasets. Code is available at https://github.com/Jiang-HB/CEMNet.

1. Introduction

Point cloud registration is the problem of finding the optimal rigid transformation (i.e., a rotation matrix and a translation vector) that can align the source point cloud to the target precisely. It plays important roles in a variety of 3D vision applications such as 3D reconstruction [1, 38, 20], Lidar SLAM [12, 57], 3D object location [14, 46, 31]. However, various challenges such as outliers and noise interference still hinder its application in the real world.

Owing to discriminative feature extraction of deep neural networks, deep point cloud registration methods [48, 49, 28] have shown impressive performance. Nevertheless, most of their success mainly depends on large amounts of ground-truth transformations for supervised point cloud registration, which greatly increases their training costs. To avoid it, recent efforts have been devoted to developing an unsupervised registration model. For example, cycle consistency has been used as the self-supervision signal to train the registration models in [19, 54]. However, the cycle-consistency loss may not be able to deal with the partially-overlapping case well, since the outliers cannot form a closed loop. In addition, unsupervised point cloud registration methods [47, 25] learn the transformation by minimizing the alignment error (e.g., Chamfer metric) between the transformed source point cloud and the target point cloud. Nevertheless, for a pair of point clouds with complex geometry, the optimization of the alignment error between them may be difficult and prone to sticking into the local minima.

Inspired by the model based reinforcement learning (RL) [22, 21], in this paper, we propose a novel sampling network guided cross-entropy method for unsupervised point cloud registration. By formulating the 3D registration task as a Markov decision process (MDP), it is expected to heuristically search for the optimal transformation by gradually narrowing its interest transformation region through trial and error. Our deep unsupervised registration model consists of two modules, i.e., sampling network module and differentiable cross-entropy method (CEM) module. Given a

*Corresponding authors.

Haobo Jiang, Yaqi Shen, Jin Xie, Jun Li, Jianjun Qian and Jian Yang are with PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, and Jiangsu Key Lab of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology, China.
pair of source and target point clouds, the sampling network aims to learn a prior Gaussian distribution over the transformation space, which can provide the subsequent CEM module a proper initialization. With the learned sampling distribution, the differentiable CEM further searches for the optimal transformation by iteratively sampling the transformation candidates, evaluating the candidates, and updating the distribution. Specifically, in the sampling network module, we utilize the learned matching map with the local geometric feature for the mean estimation of the sampling distribution and exploit the global feature for variance estimation, respectively. In the CEM module, a novel fused score function combining the current and the future rewards is designed to evaluate each sampled transformation. Particularly, the future reward is estimated by performing the ICP algorithm on the transformed source point cloud and the target point cloud. In addition, the differentiability in our CEM module is achieved by softening the sorting based top-k selection with a differentiable sparsemax function. Finally, we formulate a scaled Geman-McClure estimator [58] based loss function to train our model, where the sublinear convergence speed for the outliers can weaken the negative impact on registration precision from the outliers.

To summarize, our main contributions are as follows:

- We propose a novel end-to-end cross-entropy method based deep model for unsupervised point cloud registration, where a prior sampling distribution is predicted by a sampling network to quickly focus on the promising searching region.
- In the cross-entropy method, we design a novel ICP driven fused reward function for accurate transformation candidate evaluation and propose a sparsemax function based soft top-k selection mechanism for the differentiability of our model.
- Compared to the unsupervised or even some fully supervised deep methods, our method can obtain outstanding performance on extensive benchmarks.

2. Related Work

Traditional point cloud registration algorithms. In the traditional 3D registration field, a lot of research has focused on the Iterative Closest Point algorithm (ICP) [6] and its variants. The original ICP repeatedly performs correspondence estimation and transformation optimization to find the optimal transformation. However, ICP with improper initialization easily sticks into the local minima dilemma and the sensitivity to the outliers also degrades its performance in the partially overlapping case. To avoid it, Go-ICP [53] globally searches for the optimal transformation via integrating the branch-and-bound scheme into the ICP. Furthermore, Chetverikov et al. [7] proposed a trimmed ICP (TrICP) algorithm to handle the partial case, where the least square optimization is only performed on partial minimum square errors rather than the all. Moreover, other variants such as [40, 16, 4, 18, 11] also present competitive registration performance. In addition, RANSAC based methods [15, 10, 9, 45, 44, 13] also have been extensively studies. Among them, one representative method is the 4PCS [2, 42, 43] which determines the correspondence from the sampled four-point sets that are nearly co-planar via comparing their intersectional diagonal ratios. Furthermore, Super4PCS [33], an improved 4PCS method, is proposed to largely reduce the computational complexity of the congruent four-point sets sampling. Compared with the quadratic time complexity of points in 4PCS, Super4PCS only requires linear complexity, which greatly promotes its application in the real world. In addition, other RANSAC variants, including [35, 34, 52, 24, 17, 27] also show good alignment results.

Learning-based point cloud registration algorithms. In recent years, deep learning based point cloud registration methods have received widespread attention. PPFNet method [11] is proposed to utilize the PointNet [37] to extract the feature of the point patches and then perform the RANSAC for finding corresponding patches. DCP [48] calculates the rigid transformation via the singular value decomposition (SVD) where the correspondence is constructed through learning a soft matching map. RPMNet [55] also realizes the registration based on the matching map generated from the Sinkhorn layer and annealing. [56] aligns the point cloud pair by minimizing the KL-Divergence between two learned Gaussian Mixture Models. In addition, ReAgent [5] is proposed to combine the imitation learning and model-free reinforcement learning (i.e., proximal policy optimization [39]) for registration agent training. Instead, our method utilizes the model-based cross-entropy method for registration with the constructed MDP model. [26] proposes to search for the optimal solution by planning with a learned latent dynamic model. Although it’s unsupervised and has a faster inference speed, the approximation error in its latent reward and translation networks may potentially degrade its registration precision. Moreover, [19, 54] propose to apply the cycle consistency as the supervision signal into the point cloud registration for the unsupervised learning. However, since the non-overlapping region cannot construct a closed-loop, they also may not handle the partial case well. In addition, other learning-based methods, including [49, 28, 8, 36, 30, 29, 59] also present impressive performance.

3. Approach

3.1. Problem Setting

For the point cloud registration task, given the source point cloud $X = \{x_i \in \mathbb{R}^3 \mid i = 1, \ldots, N\}$ and the target
point cloud $\mathbf{Y} = \{y_j \in \mathbb{R}^3 \mid j = 1, \ldots, M\}$, we aim to recover the rigid transformation containing a rotation matrix $\mathbf{R} \in SO(3)$ and a translation vector $\mathbf{t} \in \mathbb{R}^3$ for aligning the two point clouds. In this work, we formulate 3D point cloud registration as a Markov decision process (MDP), which contains a state set, an action set, a state transition function, and a reward function.

**State space.** We define the set of point cloud pairs as the state space $\mathcal{S}$, that is $s = \{\mathbf{X}, \mathbf{Y}\} \in \mathcal{S}$.

**Action space.** We denote the rigid transformation space $SE(3) = (SO(3) \times \mathbb{R}^3)$ as the action space $\mathcal{A}$. To reduce the dimension of the search space, we utilize the Euler angle representation $\mathbf{e} = [e_1, e_2, e_3] \in [-\pi, \pi]^3$ to encode the rotation $\mathbf{R} \in SO(3)$. Thus, the action $\mathbf{a} \in \mathcal{A}$ is represented by $\mathbf{a} = [\mathbf{e}, \mathbf{t}]$ and $\mathbf{R}(\mathbf{e})$ is denoted as the corresponding rotation matrix of the Euler angle $\mathbf{e}$.

**State transition function.** After executing the action $\mathbf{a} = [\mathbf{e}, \mathbf{t}]$ at the current state $s = \{\mathbf{X}, \mathbf{Y}\}$, we predict the next registration state $s' = \{\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}\}$ with the state transition function $T$, that is:

$$s' = T(s, a) = \left\{ (\mathbf{R}(\mathbf{e}) x_i + \mathbf{t})_{i=1}^{N}, \mathbf{Y} \right\} = \{\mathbf{X}, \mathbf{Y}\}.$$  

(1)

**Reward function.** We evaluate the effect of executing the action $\mathbf{a}$ at the state $s$ with the reward function $R(s, a)$ in the point cloud registration task. In order to handle the partially overlapping case, we define a maximum consensus criterion based alignment metric $D_{mc}$ as below to quantify the reward, that is, $R(s, a) = -D_{mc}(T(s, a)) = -D_{mc}(\tilde{\mathbf{X}}, \mathbf{Y})$. The lower the alignment error between $\tilde{\mathbf{X}}$ and $\mathbf{Y}$, the higher the reward obtained by that action:

$$D_{mc}(\tilde{\mathbf{X}}, \mathbf{Y}) = 2 - \frac{1}{N} \sum_{\tilde{x}_i \in \mathbf{X}} \rho_\varepsilon(d_{\tilde{x}_i, \mathbf{Y}}) \left( 1 - \frac{d_{\tilde{x}_i, \mathbf{Y}}}{\varepsilon} \right) - \frac{1}{M} \sum_{y_j \in \mathbf{Y}} \rho_\varepsilon(d_{\mathbf{Y}, \tilde{\mathbf{X}}}) \left( 1 - \frac{d_{\mathbf{Y}, \tilde{\mathbf{X}}}}{\varepsilon} \right),$$

$$d_{\tilde{x}_i, \mathbf{Y}} = \min_{y_j} \|\tilde{x}_i - y_j\|_2$$

where $\rho_\varepsilon(x) = 1 \{x \leq \varepsilon\}$ is the indicator function, where the threshold $\varepsilon > 0$ is a hyper-parameter that specifies the maximum allowable distance for inlier. $D_{mc}$ measures the alignment error via counting the number of overlapping point pairs, which can effectively relieve the effect of outliers. Furthermore, each count in $D_{mc}$ is weighted with $1 - d_{\tilde{x}_i, \mathbf{Y}} / \varepsilon$. Compared to the discrete count [27], such weighting operation enforces the overlapping pair to be as close as possible rather than just no more than the maximum allowable distance.

**Lemma 1** If the threshold $\varepsilon$ is no less than the maximum of distances between all closest point pairs, that is, $\max \left\{ \max_i d_{\tilde{x}_i, \mathbf{Y}}, \max_j d_{\mathbf{Y}, \tilde{\mathbf{X}}} \right\} \leq \varepsilon < \infty$, $D_{mc}$ is proportional to the Chamfer metric $D_{cd}$ with ratio $1/\varepsilon$, i.e.,
The proof is provided in Appendix A.

Based on the constructed MDP, given a source and target point cloud pair \( s = \{X, Y\} \), the point cloud registration task aims to find the optimal action \( a^* \) by maximizing its reward as below:

\[
a^* = \arg \max_{a \in A} R(s, a) = \arg \min a \in A D^{mc}(T(s, a)).
\] (3)

### 3.2. Differentiable Cross-Entropy Method based Deep Registration Network

To solve the optimization problem in Eq. 3, we propose a novel CEM based unsupervised cloud registration framework. As shown in Fig. 1, our deep model contains two cascaded modules, including a sampling network module and a differentiable CEM module. With the source and target point cloud pair as the input, the sampling network module predicts a sampling distribution over the transformation space (Section 3.2.1). Guided by this sampling distribution, the CEM module searches the optimal transformation by iteratively narrowing the search space through the transformation sampling, transformation evaluation, and distribution updating. The optimal transformation can be estimated with the mean of the elite transformations in the last iteration (Section 3.2.2).

#### 3.2.1 Sampling Network Module

Given the source and target point cloud pair \( s = \{X, Y\} \), the traditional CEM randomly samples candidates from a pre-defined sampling distribution. Instead, we aim to sample them with a learned Gaussian distribution \( \{\mu_0, \sigma_0\} = \mathcal{P}(s; w) \) parameterized by the deep neural network with parameters \( w \), so that given a registration state, it can increase the chance that sampled candidates fall into the neighborhood of the optimal transformation. As shown in Fig. 1, we utilize the Dynamic Graph CNN (DGCNN) [50] to extract the local geometric features \( F_X \in \mathbb{R}^{N \times P} \) and \( F_Y \in \mathbb{R}^{M \times P} \) via constructing the \( k \)-NN graph for each point in \( \{X, Y\} \). With the learned point embeddings, we further generate a matching map \( M \in \mathbb{R}^{N \times M} \) with the softmax function:

\[
M_{i,j} = \text{softmax} \left( F_Y F_X^T \right)_j,
\] (4)

where \( F_{X_i} \in F_X \) denotes the feature of the point \( x_i \) and the element \( M_{i,j} \) denotes the matching probability of the points \( x_i \) and \( y_j \). With the map \( M \), we can estimate the “matching” point \( \hat{y}_i \) of \( x_i \):

\[
\hat{y}_i = \sum_{j=1}^{M} M_{i,j} \cdot y_j \in \mathbb{R}^3.
\] (5)

Once the correspondences \( \{(x_i, \hat{y}_i) \mid i = 1, \ldots, N\} \) are obtained, we employ the singular value decomposition (SVD) to calculate the transformation \( \{R, t\} \). The expected value \( \mu_0 \) of the Gaussian distribution is equal to the concatenated vector \( [e, t] \in \mathbb{R}^6 \), where \( e \) is the corresponding Euler angle of \( R \). For the prediction of the standard deviation \( \sigma_0 \), we first extract the global features of the point cloud pair \( L_X \in \mathbb{R}^P \) and \( L_Y \in \mathbb{R}^P \) via the maxpooling operation, that is, \( L_X = \text{MaxPool}(F_X) \). Then, we feed the concatenated global feature \( [L_X, L_Y] \) into the 3-layer MLP followed by a sigmoid function to output \( \sigma_0 \).

#### Algorithm 1 CEM based deep registration framework

**Input:** state \( s = \{X, Y\} \), sampling network \( \mathcal{P}(s; w) \), iteration times \( T \), action candidate numbers \( N \), iteration times \( M \) of using future reward.

1. Initialize sampling distribution \( \{\mu_0, \sigma_0\} = p_0(s; w) \).
2. for \( t = 0 : T - 1 \) do
3. for \( i = 1 : N \) do
4. Sample transformation \( a_i^t \sim \mathcal{N}(\mu_i, \sigma_i^2) \).
5. Perform \( a_i^t \) at the state \( s \) and obtain the next state \( s_i^t \) and reward \( R(s, a_i^t) \).
6. if \( t \geq M \) then
7. % Only-current reward:
8. Score \( S(s, a_i^t) = R(s, a_i^t) \).
9. else
10. % Fused reward:
11. Perform ICP to registate \( s_i^t \) and obtain the predicted transformation \( a_{\text{icp}}^t \).
12. Calculate the \( R(s_i^t, a_{\text{icp}}^t) \) as the future reward.
13. Calculate fused score \( S(s, a_i^t) \) via Eq. 6.
14. end if
15. end for
16. Calculate the sparsemax based weight vector via Eq. 10 and update the distribution via Eq. 8.
17. end for

**Return:** \( \mu_T \)

#### 3.2.2 Differentiable Cross-entropy Method Module

Guided by the prior sampling distribution learned by the network guidance module, the CEM model further iteratively searches for the optimal transformation by extensive trial and error. During the iterative process, we propose a novel fused score function combining the current reward and the ICP based future reward for accurate transformation evaluation. Furthermore, we also propose a novel differentiable CEM for the end-to-end training, where the hard top-\( k \) operation is replaced by a differentiable sparsemax function. Specifically, in the \( t \)-th iteration (\( 0 \leq t < T \)), it executes the three steps as below:

1. **Transformation candidates sampling.** In this step, we randomly sample \( N \) transformation candidates \( C_t = \{a_i^t \mid i = 1, \ldots, N\} \) from a Gaussian distribution \( g_t = \mathcal{N}(\mu_i, \sigma_i^2) \). Note that we exploit the prior distribution \( \{\mu_0, \sigma_0\} = \mathcal{P}(s; w) \) predicted by the sampling network.
for the first iteration so that it can quickly focus on the most promising transformation region for the optimal solution searching. For differentiability, we utilize the reparameterization trick to sample each candidate, that is, \( a'_i = \mu_i + \epsilon_i \cdot \sigma_i \), where \( \epsilon_i \sim \mathcal{N}(0, 1) \).

(b) Fused reward based transformation evaluation. After sampling the transformation candidates, based on our constructed MDP, we can obtain the corresponding rewards \( \{ R(s, a'_i) \mid 1 \leq i \leq N \} \) for the sampled transformation candidates. In the traditional CEM, the reward is directly used to score each transformation, i.e., the score function \( S(s, a'_i) = R(s, a'_i) \). However, a good transformation candidate needs to own not only the high current reward, but also the high future reward. For example, as shown in Fig. 2, although the transformation \( a_2 \) can temporarily obtain a higher reward than \( a_1 \) due to the more overlapping pairs, the resulting registration state by performing \( a_2 \) has a worse pose than the state by performing \( a_1 \) intuitively. Thus, for a transformation, just focusing on the current reward and ignoring the future reward of a transformation may lead evil the evolution of the CEM and cause it to stick into the local optima. Therefore, for each transformation, we need to define a future reward to determine whether the transformed source and target point clouds have good poses at the next registration state.

In our method, we utilize the ICP algorithm to quantify the future reward of a transformation. Specifically, we first perform each sampled transformation at the state \( s \) and obtain a set of states \( \{ s'_1, \ldots, s'_N \} \) where \( s'_i = T(s, a'_i) = \{ \tilde{X}_i, \tilde{Y} \} \). Then, we use ICP to predict the transformation \( a'_{icp} \) for the registration state \( s'_i \), and obtain the corresponding next state \( \{ \tilde{Z}_i, \tilde{Y} \} \) and the reward \( R(s'_i, a'_{icp}) = -D_{mc}(\tilde{Z}_i, \tilde{Y}) \). Finally, we use this reward as the future reward. The higher reward means \( a'_i \) may lead to a better pose at the next registration state. After that, we utilize the weighted sum of the current and future rewards to score each action:

\[
S(s, a'_i) = \alpha \cdot R(s, a'_i) + (1 - \alpha) \cdot R(s'_i, a'_{icp}),
\]

where \( \alpha \in [0, 1] \) is the hyper-parameter for balancing the weights of the current and future rewards. Note that benefitting from the CUDA programming, we accelerate the inference speed with the batch ICP consisting of the batch SVD and point-wise parallelization for closest point search.

(c) Sparsemax based distribution updating. In this step, we aim to use the elite transformations to update the sampling distribution so that more “better” candidates are expected to be sampled in the next iteration. In the traditional CEM, the top-\( k \) transformations \( \mathcal{I}_t \) with the highest scores are used as the elites to guide the distribution updating.

\[
R^{current}_t(a_1) < R^{current}_t(a_2) \quad R^{future}(a_1) > R^{future}(a_2)
\]

Figure 2. After performing the actions \( a_1 \) and \( a_2 \) on the source point cloud (blue), the transformed source point cloud via \( a_2 \) has more overlapping pairs (red) with the target point cloud (yellow) than those via \( a_1 \) (i.e., larger current reward, \( R^{current}(a_1) < R^{current}(a_2) \)). However, the action \( a_1 \) can lead to a better pose for future alignment than \( a_2 \) and obtain larger overlappings after performing the ICP alignment. Thus, the future reward of the action \( a_1 \) is higher than that of the action \( a_2 \) (i.e., \( R^{future}(a_1) > R^{future}(a_2) \)).
The more details about the function $\tau$ and the Jacobian of sparsemax for back propagation can be seen in Appendix B. Note that the sparsemax function maps the scores of candidates to a probability distribution whose some elements will be zero if the scores of the corresponding candidates are too low. Compared to the softmax function, the sparsemax function can adaptively ignore the negative effect of the bad candidates, which better simulates the hard top-$k$ operation. Moreover, instead of assigning the same weight ($1/k$) to the candidates, the sparsemax function assigns the higher weights to the better candidates. Thus, the updated distribution can focus on the actions with the higher scores and more potential candidates in the next iteration can be sampled. After $T$ iterations, we use the expected value of the sampling distribution to estimate the optimal transformation. The CEM based registration algorithm is outlined in Algorithm 1.

3.3. Loss Function

Under the unsupervised setting, we utilize the alignment error between the transformed source point cloud and target point cloud rather than the ground-truth transformation for model training. To handle the partially-overlapping case well, a proper robust loss function insensitive to the outliers is desired. In this paper, we integrate the scaled Geman-McClure estimator $\rho_\mu$ [58] into our loss function:

$$
\rho_\mu(x) = \frac{\mu \cdot x^2}{\mu + x^2},
$$

where the Geman-McClure estimator has the sublinear convergent speed for the outliers and thus can weaken their negative impact on the registration precision. The hyper-parameter $\mu$ determines the range of the inliers. Given a training dataset $D = \{(X_i, Y_i)\}$, the loss function $L(w)$ is defined as below:

$$
L(w) = \mathbb{E}_{X,Y \sim D} \left[ \frac{1}{|X|} \sum_{\tilde{x}_i \in X(w)} \rho_\mu(d_{\tilde{x}_i,Y}) + \frac{1}{|Y|} \sum_{y_i \in Y} \rho_\mu(d_{y_i,\tilde{X}(w)}) \right],
$$

where $\tilde{X}(w)$ denotes the transformed source point cloud after performing the transformation predicted by our deep model on the source point cloud $X$.

4. Experiment

In this section, we perform extensive experiments and ablation studies on benchmark datasets, including the ModelNet40 [51], 7Scene [41] and ICL-NUIM [23]. We simply name our CEM based Registration Network as CEMNet.

4.1. Implementation Details

We train our model using Adam optimizer with learning rate $10^{-4}$, and weight decay $5 \times 10^{-4}$ for 50 epochs. The batch size is set to 32. In our differentiable CEM module, we set the numbers of iterations $T$ and candidates $N$ to 15 and 1000, respectively, and the maximum allowable distance $\varepsilon$ is set to 0.1. For the fused reward for transformation evaluation, we set $M$ to 3 and the impact weight $\alpha$ in the score function is set to 0.5. The hyper-parameter $\mu$ in the Geman-McClure estimator used in our loss function is set to 0.01. We utilize PyTorch to implement our project and perform all experiments on the server equipped with a 2080Ti GPU and an Intel i5 2.2GHz CPU.

4.2. Evaluation on ModelNet40

We first test our method on the ModelNet40 dataset [51], which contains 40 categories and is constructed via uniformly sampling 1,024 points from each of 12,311 CAD models. Then, we divide it into two parts where 2,468 models are used for testing and the remaining models are for training. Following the setting in [48], we first generate a random rotation matrix and translation vector via uniformly sampling in the Euler angle range $[0, 45^\circ]$ and the translation range $[-0.5, 0.5]$. Then, we use the transformed source point cloud as the target point cloud and the generated transformation is as the ground truth. The metric for evaluating the registration precision contains the root mean squared error (RMSE) and mean absolute error (MAE) between the predicted transformation and the ground-truth transformation, where the error about Euler angle uses the degree as the unit. We compare our method with eight state-of-the-art methods, where ICP [6], Go-ICP [53], Super4PCS [33], SDRSAC [27] and FGR [58] belong to the traditional methods, and IDAM [28] and DCP-v2 [48] are fully supervised deep methods while FMR [25] is unsupervised. Some visualization results are presented in Fig. 3 and more qualitative examples can be seen in Appendix D.

Partially overlapping unseen object. We first evaluate our method on partially overlapping unseen objects, where the test and training datasets contain all 40 categories, and the non-overlapping region may still exist in the perfectly aligned point clouds. Following [49], to construct the partially overlapping point clouds, we first randomly sample a point in source and target point clouds, respectively. Then, we perform the farthest point subsampling (FPS) to sample 75% (768) points while the remaining 25% points are viewed as the missing points and discarded. We utilize the implementations provided by the authors to train FMR, DCP-v2, and IDAM on our modified training dataset with partial overlap. The left column of Table 1 shows that compared to other algorithms, our unsupervised method can obtain superior registration performance with all evaluation metrics and even exceeds the fully supervised DCP-v2 and IDAM, which owes to that the CEM module guided by the sampling network can effectively search for the accurate transformation.

Partially overlapping unseen category. To further test the
<table>
<thead>
<tr>
<th>Model</th>
<th>RMSE(R)</th>
<th>RMSE(t)</th>
<th>MAE(R)</th>
<th>MAE(t)</th>
<th>RMSE(R)</th>
<th>RMSE(t)</th>
<th>MAE(R)</th>
<th>MAE(t)</th>
<th>RMSE(R)</th>
<th>RMSE(t)</th>
<th>MAE(R)</th>
<th>MAE(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP [6] (+)</td>
<td>14.7732</td>
<td>0.0351</td>
<td>3.5938</td>
<td>0.0132</td>
<td>12.5413</td>
<td>0.0398</td>
<td>4.2826</td>
<td>0.0184</td>
<td>0.0025</td>
<td>0.0434</td>
<td>0.0002</td>
<td>0.0241</td>
</tr>
<tr>
<td>Go-ICP [53] (+)</td>
<td>13.8222</td>
<td>0.0321</td>
<td>3.1579</td>
<td>0.0121</td>
<td>14.5225</td>
<td>0.0329</td>
<td>3.4252</td>
<td>0.0114</td>
<td>0.0419</td>
<td>0.0185</td>
<td>2.3965</td>
<td>0.0164</td>
</tr>
<tr>
<td>Super4PCS [33] (+)</td>
<td>2.5186</td>
<td>0.0032</td>
<td>1.4364</td>
<td>0.0041</td>
<td>13.8551</td>
<td>0.0321</td>
<td>3.1579</td>
<td>0.0121</td>
<td>3.5701</td>
<td>0.0206</td>
<td>1.0642</td>
<td>0.0066</td>
</tr>
<tr>
<td>SDRSAC [27] (+)</td>
<td>4.2475</td>
<td>0.0139</td>
<td>3.0144</td>
<td>0.0121</td>
<td>19.9166</td>
<td>0.1127</td>
<td>7.5760</td>
<td>0.0101</td>
<td>0.0034</td>
<td>0.0013</td>
<td>2.3572</td>
<td>0.0072</td>
</tr>
<tr>
<td>IDAM [28] (+)</td>
<td>14.7732</td>
<td>0.0351</td>
<td>3.5938</td>
<td>0.0132</td>
<td>12.5413</td>
<td>0.0398</td>
<td>4.2826</td>
<td>0.0184</td>
<td>0.0025</td>
<td>0.0434</td>
<td>0.0002</td>
<td>0.0241</td>
</tr>
<tr>
<td>Ours</td>
<td>14.7732</td>
<td>0.0351</td>
<td>3.5938</td>
<td>0.0132</td>
<td>12.5413</td>
<td>0.0398</td>
<td>4.2826</td>
<td>0.0184</td>
<td>0.0025</td>
<td>0.0434</td>
<td>0.0002</td>
<td>0.0241</td>
</tr>
</tbody>
</table>

**Figure 3.** Qualitative registration examples on partially overlapping *ModelNet40* dataset.

### 4.3. Evaluation on 7Scene and ICL-NUIM

We further evaluate our method on two indoor datasets: 7Scene and ICL-NUIM. The former consists of seven scenes: Chess, Fires, Heads, Office, Pumpkin, RedKitchen, and Stairs, and is split into two parts, one containing 296 scans for training and the other containing 57 scans as the test dataset. The latter is split into 1,278 and 200 scans for training and test, respectively. As in Sec. 4.2, we exploit the FPS operation to obtain the partial point clouds.

**Table 1.** Comparison results on partially overlapping *ModelNet40* dataset. (+), (△) and (○) denote the traditional, unsupervised and fully supervised deep methods, respectively.

**Table 2.** Comparison results on 7Scene and ICL-NUIM datasets. (+), (△) and (○) denote the traditional, unsupervised and fully supervised deep methods, respectively.
and use the transformed source point cloud as the target point cloud, where the transformation is also generated via randomly sampling. As demonstrated in Table 2, on ICL-NUIM, our method exhibits higher registration precision on all criteria. In addition, on 7Scene, our method is slightly worse than FGR in terms of the RMSE of translation but shows the lower alignment errors on other criteria.

### 4.4. Ablation Study and Analysis

**Sampling network module.** As shown in the fourth and fifth rows of Table 3, equipped with the sampling network module (SN), dCEM (differentiable CEM) and dCEM+Future (differentiable CEM with future reward) can obtain significant error reduction on all criteria. This is because that SN can learn a better initial sampling distribution, which assists the dCEM to efficiently search for the optimal solution. Note that we use the Gaussian distribution \( N(0, 1) \) as the pre-defined sampling distribution for our method without SN. Moreover, since the sampling network has a similar model design as DCP-v2 except for the branch used for variance prediction, the results in the first row refer to the scores of the fully supervised DCP-v2. One can see that without planning in the CEM module, SN yields poor performance, even guided by the ground truth transformation, which implies the necessity of the proposed CEM module for the precise registration.

**Fused score function.** We further evaluate the effect of ICP based future reward in the fused score function of CEM. As shown in the second and fourth rows of Table 3, the performance drops significantly without considering the future reward of the transformation candidate. Furthermore, Table 4 also shows that as the iteration times \( M \) of using the future reward increases, the error keeps decreasing while the time cost continues to increase. To balance the performance and speed, we set \( M = 3 \) in all experiments. Finally, we test the fluctuation of the error with different weights \( \alpha \) in the fused function. Fig. 4(b) shows that a good balance between the current and future rewards tends to bring higher precision.

**Parameters in CEM.** We test the sensitivity of the hyper-parameters in CEM, including the numbers of the iterations \( T \) and the candidates \( N \). Fig. 4(a) shows that as \( T \) and \( N \) increase, the error tends to decrease (from the light color to the dark color). However, as shown in Appendix C, their performance gain is at the cost of the inference time and we set \( N \) and \( T \) to 1000 and 10 in all experiments.

**Inference time.** We use the averaged running time on each case from ModelNet40 dataset (unseen object) to evaluate the inference time of each algorithm. Note that the traditional registration methods are performed on Intel i7 2.2GHz CPU while the deep learning based methods are executed on 2080Ti GPU. The time cost of our method is 310ms, while the cost of the other methods are ICP (14ms), Go-ICP (2, 278ms), Super4PCS (6, 790ms), FGR (75ms), SDRSAC (23, 651ms), FMR (414ms), DCP-v2 (34ms), and IDAM (53ms), respectively.

### 5. Conclusion

In this paper, we proposed a novel sampling network guided cross-entropy method for unsupervised point cloud registration. In our framework, the sampling network is used to provide a prior sampling distribution for CEM. Guided by this learnable distribution, the CEM module further heuristically searches for the optimal transformation. During this optimization process, we designed a fused score function combining current and ICP based future rewards for more accurate transformation evaluation. Furthermore, we also replaced the hard top-\( k \) selection with the differentiable sparsmax function for the end-to-end model training.

### Acknowledgments

This work was supported by the National Science Fund of China (Grant Nos. U1713208, 61876084, 61876083, 62072242).
References


