DeepGMR: Learning Latent Gaussian Mixture Models for Registration

Wentao Yuan[1]* Ben Eckart[2] Kihwan Kim[2]
Varun Jampani[2] Dieter Fox[1,2] Jan Kautz[2]

[1] University of Washington
[2] NVIDIA

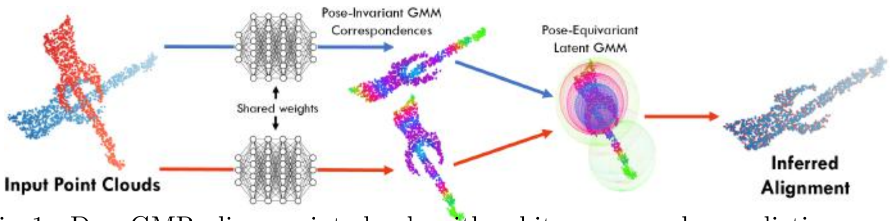

Input Point Clouds
Shared weights
Pose-Invariant GMM Correspondences
Pose-Equivariant Latent GMM
Inferred Alignment

Fig. 1: DeepGMR aligns point clouds with arbitrary poses by predicting pose-invariant point correspondences to a learned latent Gaussian Mixture Model (GMM) distribution.

Abstract. Point cloud registration is a fundamental problem in 3D computer vision, graphics and robotics. For the last few decades, existing registration algorithms have struggled in situations with large transformations, noise, and time constraints. In this paper, we introduce Deep Gaussian Mixture Registration (DeepGMR), the first learning-based registration method that explicitly leverages a probabilistic registration paradigm by formulating registration as the minimization of KL-divergence between two probability distributions modeled as mixtures of Gaussians. We design a neural network that extracts pose-invariant correspondences between raw point clouds and Gaussian Mixture Model (GMM) parameters and two differentiable compute blocks that recover the optimal transformation from matched GMM parameters. This construction allows the network learn an SE(3)-invariant feature space, producing a global registration method that is real-time, generalizable, and robust to noise. Across synthetic and real-world data, our proposed method shows favorable performance when compared with state-of-the-art geometry-based and learning-based registration methods.

Keywords: Point cloud registration, Gaussian Mixture Model

1 Introduction

The development of 3D range sensors [2] has generated a massive amount of 3D data, which often takes the form of point clouds. The problem of assimilating

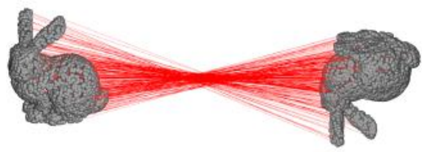
*Work partially done during an internship at NVIDIA
Code and data are available at https://wentaoyuan.github.io/deepgmr

raw point cloud data into a coherent world model is crucial in a wide range of vision, graphics and robotics applications. A core step in the creation of a world model is point cloud registration, the task of finding the transformation that aligns input point clouds into a common coordinate frame.

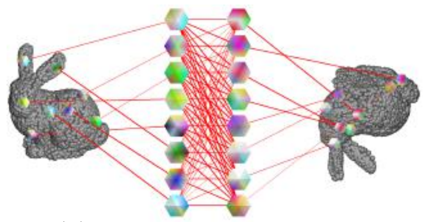
As a longstanding problem in computer vision and graphics, there is a large body of prior works on point cloud registration [30]. However, the majority of registration methods rely solely on matching local geometry and do not leverage learned features capturing large-scale shape information. As a result, such registration methods are often *local*, which means they cannot handle large transformations without good initialization. In contrast, a method is said to be *global* if its output is invariant to initial poses. Several works have investigated global registration. However, they are either too slow for real-time processing [47,5] or require good normal estimation to reach acceptable accuracy [48]. In many applications, such as re-localization and pose estimation, an accurate, fast, and robust global registration method is desirable.

The major difficulty of global registration lies in data association, since ICP-style correspondences based on Euclidean distances are no longer reliable. Existing registration methods provide several strategies for performing data association (see Fig. 2), but each of these prove problematic for global registration on noisy point clouds. Given point clouds of size N, DCP [42] attempts to perform point-to-point level matching like in ICP (Fig. 2a) over all N^2 point pairs, which suffers from $O(N^2)$ complexity. In addition, real-world point clouds don't contain exact point-level correspondences due to sensor noise. FGR [48] performs sparse feature-level correspondences that can be much more efficient (Fig. 2b), but are highly dependent on the quality of features. For example, the FPFH features used in [48] rely on consistent normal estimation, which is difficult to obtain in practice due to varying sparsity or non-rectilinear geometry [41]. Moreover, these sparse correspondences are still point-level and suffer the same problem when exact point-level correspondences don't exist. To solve this problem, one can use probabilistic methods to perform distribution-to-distribution matching (Fig. 2c). However, the distribution parameters are not guaranteed to be consistent across different views due to the well-known problem of Gaussian Mixture *non-indentifiability* [4]. Thus, probabilistic algorithms like HGMR [15] have only local convergence and rely on iterative techniques to update and refine point-to-distribution correspondences.

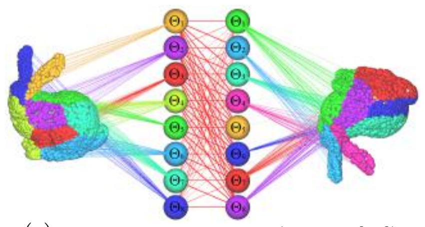
In this paper, we introduce a novel registration method that is designed to overcome these limitations by learning pose-invariant point-to-distribution parameter correspondences (Fig. 2d). Rather than depending on point-to-point correspondence [42] and iterative optimization [1], we solve for the optimal transformation in a single step by matching points to a probability distribution whose parameters are estimated by a neural network from the input point clouds. Our formulation is inspired by prior works on Gaussian mixture registration [15,24], but different from these works in two ways. First, our method does not involve expensive iterative procedures such as Expectation Maximization (EM) [12]. Second, our network is designed to learn a consistent GMM representation across multiple point clouds rather than fit a GMM to a single reference point cloud.



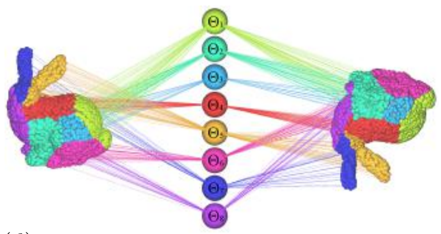
(a) Point Matching (ICP [3], DCP[42])



(b) Feature Matching (FGR [48])



(c) Distribution Matching (HGMR [15])



(d) Direct Latent Correspondence (**Proposed**)

Fig. 2: Comparison of data association techniques in various registration approaches. Unlike previous approaches that attempt to find point-to-point, feature-to-feature, or distribution-to-distribution level correspondences, we learn direct and pose-invariant correspondences to a distribution inside a learned latent space.

Our proposed method has the following favorable properties:

Global Registration Point clouds can be aligned with arbitrary displacements and without any initialization. While being accurate on its own, our method can work together with local refinement methods to achieve higher accuracy.

Efficiency The proposed method runs on the order of 20-50 frames per second with moderate memory usage that grows linearly with the number of points, making it suitable for applications with limited computational resources.

Robustness Due to its probabilistic formulation, our method is tolerant to noise and different sizes of input point clouds, and recovers the correct transformation even in the absence of exact point-pair correspondences, making it suitable for real-world scenarios.

Differentiability Our method is fully differentiable and the gradients can be obtained with a single backward pass. It can be included as a component of a larger optimization procedure that requires gradients.

We demonstrate the advantages of our method over the state-of-the-art on several kinds of challenging data. The baselines consist of both recent geometry-based methods as well as learning-based methods. Our datasets contain large transformations, noise, and real-world scene-level point clouds, which we show cause problems for many state-of-the-art methods. Through its connection to Gaussian mixture registration and novel learning-based design, our proposed method performs well even in these challenging settings.

2 Related Work

Point cloud registration has remained a popular research area for many years due to its challenging nature and common presence as an important component of many 3D perception applications. Here we broadly categorize prior work into *local* and *global* techniques, and discuss how emerging *learning-based* methods fit into these categories.

Local Registration Local approaches are often highly efficient and can be highly effective if limited to regimes where transformations are known *a priori* to be small in magnitude. The most well-known approach is the Iterative Closest Point (ICP) algorithm [3,8] and its many variants [33,34]. ICP iteratively alternates between two phases: point-to-point correspondence and distance minimization. Countless strategies have been proposed for handling outliers and noise [9], creating robust minimizers [18], or devising better distance metrics [26,35].

Another branch of work on local methods concerns probabilistic registration, often via the use of GMMs and the EM algorithm [12]. Traditional examples include EM-ICP [20], GMMReg [24], and methods based on the Normal Distributions Transform (NDT [36]). More recent examples offer features such as batch registration (JRMPC [16]) or robustness to density variance and viewing angle (DARE [23]). Other recent approaches have focused on efficiency, including filter-based methods [19], GPU-accelerated hierarchical methods [15], Monte Carlo methods [13] or efficient distribution-matching techniques [38].

In our experiments, we compare our algorithm against local methods belonging to both paradigms: the Trimmed ICP algorithm with point-to-plane minimization [26,9] and Hierarchical Gaussian Mixture Registration (HGMR) [15], a state-of-the-art probabilistic method.

Global Registration Unlike local methods, global methods are invariant to initial conditions, but often at the cost of efficiency. Some approaches exhaustively search $SE(3)$ via branch-and-bound techniques (Go-ICP [47], GOGMA [5] and GOSMA [6]). Other approaches use local feature matching with robust optimization techniques such as RANSAC [17] or semidefinite programming [46].

One notable exception to the general rule that global methods must be inefficient is Fast Global Registration (FGR) [48], which achieves invariance to initial pose while remaining as fast or faster than many local methods. We compare against FGR [48], RANSAC [17] and TEASER++ [46] in our experiments as representatives of state-of-the-art geometry-based global methods.

Learning-based Registration Deep learning techniques on point clouds such as [31,32,37,44] provide task-specific learned point representations that can be leveraged for robust point cloud registration. PointNetLK [1], DCP [42] and PRNet [43] are the closest-related registration methods to ours. PointNetLK [1] proposes a differentiable Lucas-Kanade algorithm [27] that tries to minimize the feature distance between point clouds. DCP [42] proposes attention-based feature matching coupled with differentiable SVD for point-to-point registration, while PRNet [43] uses neural networks to detect keypoints followed by SVD for final registration. However, as we show in our experiments, due to their iterative nature, PointNetLK and PRNet are local methods that do not converge under large transformations. While DCP is a global method, it performs point-to-point correspondence which suffers on noisy point clouds.

Our proposed approach can be characterized as a *global* method, as well as the first *learning-based*, *probabilistic* registration method. To further emphasize the difference of our approach in the context of data association and matching, refer to the visual illustrations of various correspondence strategies in Fig. 2.

3 GMM-Based Point Cloud Registration

Before describing our approach, we will briefly review the basics of the Gaussian Mixture Model (GMM) and how it offers a maximum likelihood (MLE) framework for finding optimal alignment between point clouds, which can be solved using the Expectation Maximization (EM) algorithm [15]. We then discuss the strengths and limitations of this framework and motivate the need for learned GMMs.

A GMM establishes a multimodal generative probability distribution over 3D space ($\mathbf{x} \in \mathbb{R}^3$) as a weighted sum of J Gaussian distributions,

$$p(\mathbf{x} \mid \boldsymbol{\Theta}) \overset{\text{def}}{=} \sum_{j=1}^{J} \pi_j \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j), \tag{1}$$

where $\sum_j \pi_j = 1$. GMM parameters $\boldsymbol{\Theta}$ comprise J triplets $(\pi_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where π_j is a scalar mixture weight, $\boldsymbol{\mu}_j$ is a 3×1 mean vector and $\boldsymbol{\Sigma}_j$ is a 3×3 covariance matrix of the j-th component.

Given point clouds $\hat{\mathcal{P}}, \mathcal{P}$ and the space of permitted GMM parameterizations $\mathcal{X}$, we can formulate the registration from $\hat{\mathcal{P}}$ to $\mathcal{P}$ as a two-step optimization problem,

$$\textbf{Fitting: } \boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta} \in \mathcal{X}}{\operatorname{argmax}}\, p(\mathcal{P} \mid \boldsymbol{\Theta}) \tag{2}$$

$$\textbf{Registration: } T^* = \underset{T \in SE(3)}{\operatorname{argmax}}\, p(T(\hat{\mathcal{P}}) \mid \boldsymbol{\Theta}^*) \tag{3}$$

where $SE(3)$ is the space of 3D rigid transformations. The fitting step fits a GMM $\boldsymbol{\Theta}^*$ to the target point cloud $\mathcal{P}$, while the registration step finds the optimal transformation T^* that aligns the source point cloud $\hat{\mathcal{P}}$ to $\boldsymbol{\Theta}^*$.

Note that both steps maximize the likelihood of a point cloud under a GMM, but with respect to different parameters. In general, directly maximizing the likelihood p is intractable. However, one can use EM to maximize a lower bound on p by introducing a set of latent correspondence variables $\mathcal{C}$. EM iterates between E-step and M-step until convergence. At each iteration k, the E-step updates the lower bound q to the posterior over $\mathcal{C}$ given a guess of the parameters $\boldsymbol{\Theta}^{(k)}$ and the M-step updates the parameters by maximizing the expected joint likelihood under q. As an example, the EM updates for the fitting step (Eq. 2) are as follows.

$$\mathbf{E}_{\boldsymbol{\Theta}}\textbf{: } q(\mathcal{C}) := p(\mathcal{C} \mid \mathcal{P}, \boldsymbol{\Theta}^{(k)}) \tag{4}$$

$$\mathbf{M}_{\boldsymbol{\Theta}}\textbf{: } \boldsymbol{\Theta}^{(k+1)} := \underset{\boldsymbol{\Theta} \in \mathcal{X}}{\operatorname{argmax}}\, \mathbb{E}_q[\ln p(\mathcal{P}, \mathcal{C} \mid \boldsymbol{\Theta})] \tag{5}$$

The key for EM is the introduction of latent correspondences $\mathcal{C}$. Given a point cloud $\mathcal{P} = \{p_i\}_{i=1}^N$ and a GMM $\boldsymbol{\Theta}^{(k)} = \{\pi_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^J$, $\mathcal{C}$ comprises NJ binary latent variables $\{c_{ij}\}_{i,j=1,1}^{N,J}$ whose posterior factors can be calculated as

$$p(c_{ij} = 1 \mid p_i, \boldsymbol{\Theta}^{(k)}) = \frac{\pi_j \mathcal{N}(p_i | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{j'=1}^J \pi_{j'} \mathcal{N}(p_i | \boldsymbol{\mu}_{j'}, \boldsymbol{\Sigma}_{j'})}, \tag{6}$$

which can be seen as a kind of softmax over the squared Mahalanobis distances from p_i to each component center $\boldsymbol{\mu}_j$. Intuitively, if p_i is closer to $\boldsymbol{\mu}_j$ relative to the other components, then c_{ij} is more likely to be 1.

The introduction of $\mathcal{C}$ makes the joint likelihood in the M-step (Eq. 5) factorizable, leading to a closed-form solution for $\boldsymbol{\Theta}^{(k+1)} \overset{\text{def}}{=} \{\pi_j^*, \boldsymbol{\mu}_j^*, \boldsymbol{\Sigma}_j^*\}_{j=1}^J$. Let $\gamma_{ij} \overset{\text{def}}{=} \mathbb{E}_q[c_{ij}]$. The solution for Eq. 5 can be written as

$$\pi_j^* = \frac{1}{N} \sum_{i=1}^N \gamma_{ij}\,, \qquad N\pi_j^* \boldsymbol{\mu}_j^* = \sum_{i=1}^N \gamma_{ij} p_i\,, \tag{7,8}$$

$$N\pi_j^* \boldsymbol{\Sigma}_j^* = \sum_{i=1}^N \gamma_{ij} (p_i - \boldsymbol{\mu}_j^*)(p_i - \boldsymbol{\mu}_j^*)^\top \tag{9}$$

Likewise, the registration step (Eq. 3) can be solved with another EM procedure optimizing over the transformation parameters T.

$$\mathbf{E}_T\textbf{:}\ q(\mathcal{C}) := p(\mathcal{C} \mid T^{(k)}(\hat{\mathcal{P}}), \boldsymbol{\Theta}^*) \tag{10}$$

$$\mathbf{M}_T\textbf{:}\ T^{(k+1)} := \operatorname*{argmax}_{T \in SE(3)} \mathbb{E}_q[\ln p(T(\hat{\mathcal{P}}), \mathcal{C} \mid \boldsymbol{\Theta}^*)] \tag{11}$$

Many registration algorithms [23,19,15,28,22,20,14] can be viewed as variations of the general formulation described above. Compared to methods using point-to-point matches [8,35,18,42], this formulation provides a systematic way of dealing with noise and outliers using probabilistic models and is fully differentiable. However, the iterative EM procedure makes it much more computationally expensive. Moreover, when the transformation is large, the EM procedure in Eq. 10,11 often gets stuck in local minima. This is because Eq. 6 used in the E Step performs point-to-cluster correspondence based on locality, i.e. a point likely belongs to a component if it is close to the component's center, which leads to spurious data association between $\hat{\mathcal{P}}$ and $\boldsymbol{\Theta}^*$ when T is large. In the following section, we show how our proposed method solves the data association problem by learning pose-invariant point-to-GMM correspondences via a neural network. By doing so, we also remove the need for an iterative matching procedure.

4 DeepGMR

In this section, we give an overview of Deep Gaussian Mixture Registration (DeepGMR) (Fig. 3), the first registration method that integrates GMM registration

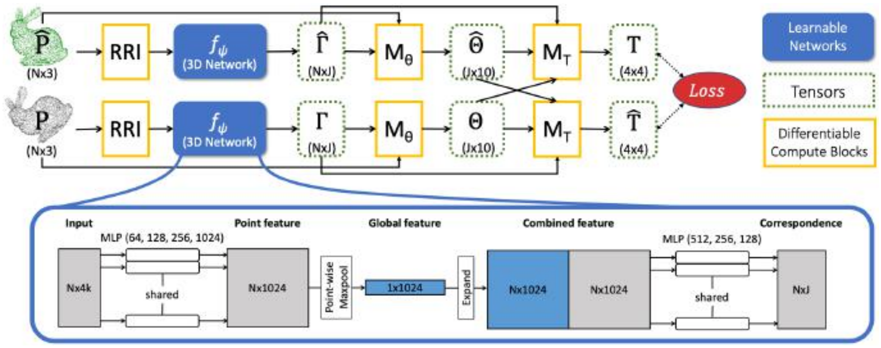


Fig. 3: Given two point clouds $\hat{P}$ and P, DeepGMR estimates the optimal transformation T that registers $\hat{P}$ to P and optionally the inverse transformation $\hat{T}$. DeepGMR has three major components: a learnable permutation-invariant point network f_ψ (Sec. 4.1) that estimates point-to-component correspondences $\mathit{\Gamma}$ and two differentiable compute blocks $\mathbf{M}_{\mathbf{\Theta}}$ (Sec. 4.2) and $\mathbf{M}_{\mathbf{T}}$ (Sec. 4.3) that compute the GMM parameters $\boldsymbol{\mathit{\Theta}}$ and transformation T in closed-form. We backpropagate a mean-squared loss through the differentiable compute blocks to learn the parameters of f_ψ.

with neural networks. DeepGMR features a correspondence network (Sec. 4.1) and two differentiable computing blocks (Sec. 4.2, 4.3) analogous to the two EM procedures described in Sec. 3. The key idea is to replace the E-step with a correspondence network, which leads to consistent data association under large transformations, overcoming the weakness of conventional GMM registration.

4.1 Correspondence Network f_ψ

The correspondence network f_ψ takes in a point cloud with N points, $\mathcal{P} = \{p_i\}_{i=1}^N$, or a set of pre-computed features per point and produces a $N \times J$ matrix $\mathit{\Gamma} = \left[\gamma_{ij}\right]$ where $\sum_{j=1}^J \gamma_{ij} = 1$ for all i. Each γ_{ij} represents the latent correspondence probability between point p_i and component j of the latent GMM.

Essentially, the correspondence network replaces the E-step of a traditional EM approach reviewed in Sec. 3. In other words, f_ψ induces a distribution q_ψ over latent correspondences (Eq. 4). Importantly, however, the learned latent correspondence matrix $\mathit{\Gamma}$ does not rely on Mahalanobis distances as in Eq. 6. This relaxation has several advantages. First, the GMMs generated by our network can deviate from the maximum likelihood estimate if it improves the performance of the downstream task. Second, by offloading the computation of $\mathit{\Gamma}$ to a deep neural network, higher-level contextual information can be learned in a data-driven fashion in order to produce robust non-local association predictions. Third, since the heavy lifting of finding the appropriate association is left to the neural network, only a single iteration is needed.

Finally, under this framework, the computation of $\mathit{\Gamma}$ can be viewed as a J-class point classification problem. As a result, we can leverage existing point cloud segmentation networks as our backbone.

4.2 $\mathbf{M}_{\boldsymbol{\Theta}}$ Compute Block

Given the outputs $\mathit{\Gamma}$ of f_ψ together with the point coordinates $\mathcal{P}$, we can compute the GMM parameters $\boldsymbol{\Theta}$ in closed form according to Eq. 7, 8, and 9. Since these equations are weighted combinations of the point coordinates, the estimated GMM must overlap with the input point cloud spatially, providing an effective inductive bias for learning a stable representation. We refer to the parameter-free compute block that implements the conversion from $(\mathit{\Gamma}, \mathcal{P}) \rightarrow \boldsymbol{\Theta}$ as $\mathbf{M}_{\boldsymbol{\Theta}}$.

In order to have a closed-form solution for the optimal transformation (Sec. 4.3), we choose the covariances to be isotropic, *i.e.*, $\boldsymbol{\Sigma}_j = \text{diag}([\sigma_j^2, \sigma_j^2, \sigma_j^2])$. This requires a slight modification of Eq. 9, replacing the outer product to inner product. The number of Gaussian components J remains fixed during training and testing. We choose $J = 16$ in our experiments based on ablation study results (refer to the supplement for details). Note that J is much smaller than the number of points, which is important for time and memory efficiency (Sec. 5.3).

4.3 $\mathbf{M}_{\mathbf{T}}$ Compute Block

In this section, we show how to obtain the optimal transformation in closed form given the GMM parameters estimated by $\mathbf{M}_{\boldsymbol{\Theta}}$. We refer to this parameter-free compute block that implements the computation from $(\mathit{\Gamma}, \boldsymbol{\Theta}, \hat{\boldsymbol{\Theta}}) \rightarrow T$ as $\mathbf{M}_{\mathbf{T}}$.

We denote the source point cloud as $\hat{\mathcal{P}} = \{\hat{p}_i\}$ and the target point cloud as $\mathcal{P} = \{p_i\}$. The estimated source and target GMMs are denoted as $\hat{\boldsymbol{\Theta}} = (\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ and $\boldsymbol{\Theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$. The latent association matrix for source and target are $\hat{\mathit{\Gamma}} = \{\hat{\gamma}_{ij}\}$ and $\mathit{\Gamma} = \{\gamma_{ij}\}$. We use $T(\cdot)$ to denote the transformation from the source to the target, consisting of rotation R and translation $\mathbf{t}$ (*i.e.*, $T \in SE(3)$).

The maximum likelihood objective of $\mathbf{M}_{\mathbf{T}}$ is to minimize the KL-divergence between the transformed latent source distribution $T(\hat{\boldsymbol{\Theta}})$ and the latent target distribution $\boldsymbol{\Theta}$,

$$T^* = \underset{T}{\text{argmin}}\, \text{KL}(T(\hat{\boldsymbol{\Theta}}) \mid \boldsymbol{\Theta}) \tag{12}$$

It can be shown that under general assumptions, the minimization in Eq. (12) is equivalent to maximizing the log likelihood of the transformed source point cloud $T(\hat{\mathcal{P}})$ under the target distribution $\boldsymbol{\Theta}$:

$$T^* = \underset{T}{\text{argmax}} \sum_{i=1}^{N} \ln \sum_{j=1}^{J} \pi_j \mathcal{N}(T(\hat{p}_i) \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{13}$$

Thus, Equations 12-13 conform to the generalized registration objective outlined in Sec. 3, Eq. 11. We leave the proof to the supplement.

To eliminate the sum within the log function, we employ the same trick as in the EM algorithm. By introducing a set of latent variables $\mathcal{C} = \{c_{ij}\}$ denoting the association of the transformed source point $T(\hat{p}_i)$ with component j of the target GMM $\boldsymbol{\Theta}$, we can form a lower bound over the log likelihood as the expectation of the joint log likelihood $\ln p(T(\hat{\mathcal{P}}), \mathcal{C})$ with respect to some distribution q_ψ over $\mathcal{C}$, where ψ are the parameters of the correspondence network

$$T^* = \underset{T}{\operatorname{argmax}}\, \mathbb{E}_{q_\psi}[\ln p(T(\hat{\mathcal{P}}), \mathcal{C} \mid \boldsymbol{\Theta})] \tag{14}$$

$$= \underset{T}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{j=1}^{J} \mathbb{E}_{q_\psi}[c_{ij}] \ln \mathcal{N}(T(\hat{p}_i) \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{15}$$

Note that we parametrize q with the correspondence network f_ψ instead of setting q as the posterior based on Mahalanobis distances (Eq. 6). In other words, we use f_ψ to replace the $\mathbf{E}_T$ step in Eq. 10. Specifically, we set $\mathbb{E}_{q_\psi}[c_{ij}] = \hat{\gamma}_{ij}$ and rewrite Eq. (15) equivalently as the solution to the following minimization,

$$T^* = \underset{T}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{j=1}^{J} \hat{\gamma}_{ij} \| T(\hat{p}_i) - \boldsymbol{\mu}_j \|^2_{\boldsymbol{\Sigma}_j} \tag{16}$$

where $\| \cdot \|_{\boldsymbol{\Sigma}_j}$ denote the Mahalanobis distance.

The objective in Eq. (16) contains NJ pairs of distances which can be expensive to optimize when N is large. However, we observe that the output of $\mathbf{M}_{\boldsymbol{\Theta}}$ (*i.e.*, using Eqs. 7, 8, and 9), with respect to $\hat{\mathcal{P}}$ and $\hat{\Gamma}$, allows us to simplify this minimization to a single sum consisting only of the latent parameters $\hat{\boldsymbol{\Theta}}$ and $\boldsymbol{\Theta}$. This reduction to a single sum can be seen as a form of barycentric coordinate transformation where only the barycenters' contributions vary with respect to the transformation. Similar types of reasoning and derivations can be found in previous works using GMMs [20,14] (see the supplement for further details). The final objective ends up with only J pairs of distances, which saves a significant amount of computation. Furthermore, by using isotropic covariances in $\mathbf{M}_{\boldsymbol{\Theta}}$, the objective simplifies from Mahalanobis distances to weighted Euclidean distances,

$$T^* = \underset{T}{\operatorname{argmin}} \sum_{j=1}^{J} \frac{\hat{\pi}_j}{\sigma_j^2} \| T(\hat{\boldsymbol{\mu}}_j) - \boldsymbol{\mu}_j \|^2 \tag{17}$$

Note that Eq. 17 implies a correspondence between components in the source GMM $\boldsymbol{\Theta}$ and the target GMM $\hat{\boldsymbol{\Theta}}$. Such a correspondence does *not* naturally emerge from two independently estimated GMMs (Fig. 2c). However, by representing the point-to-component correspondence between $\hat{\mathcal{P}}$ and $\hat{\boldsymbol{\Theta}}$ and between $T(\hat{\mathcal{P}})$ and $\boldsymbol{\Theta}$ with the same Γ matrix, an implicit component-to-component correspondence between $\boldsymbol{\Theta}$ and $\hat{\boldsymbol{\Theta}}$ is enforced, which reduces the alignment of point clouds to the alignment of component centroids (weighted by the covariances).

From Eq. 17, we can solve T^* in closed-form using a weighted version of the SVD solution in [40] (refer to the supplement for more detail). Note that our formulation is fundamentally different from previous learning-based approaches such as DCP [42] that use the SVD solver on point-to-point matches. We use a rigorous probabilistic framework to reduce the registration problem to matching latent Gaussian components, which has dimension J that is usually orders of magnitude smaller than the number of points N (in our experiments, $J = 16$ and $N = 1024$). Our formulation is not only much more efficient (SVD has complexity $O(d^3)$ on a $d \times d$ matrix) but also much more robust, since exact point-to-point correspondences rarely exist in real world point clouds with sensor noise.

4.4 Implementation

Our general framework is agnostic to the choice of the particular architecture of f_ψ. In our experiments, we demonstrate state-of-the-art results with a simple PointNet segmentation backbone [31] (see inset of Fig. 3), but in principle other more powerful backbones can be easily incorporated [11,39]. Because it can be challenging for PointNet to learn rotation invariance, we pre-process the input points into a set of rigorously rotation-invariant (RRI) features, proposed by [7], and use these hand-crafted RRI features as the input to the correspondence network. Note that RRI features are only used in the estimation of the association matrix Γ and not in the computation of the GMM parameters inside $\mathbf{M}_{\mathbf{\Theta}}$, which depends on the raw point coordinates. Our ablation studies (see supplement) show that RRI does help, but is not essential to our method.

During training, given a pair of point clouds $\hat{\mathcal{P}}$ and $\mathcal{P}$, we apply the correspondence network f_ψ along with $\mathbf{M}_{\mathbf{\Theta}}$ and $\mathbf{M}_{\mathbf{T}}$ compute blocks to obtain two transformations: T from $\hat{\mathcal{P}}$ to $\mathcal{P}$ and $\hat{T}$ from $\mathcal{P}$ to $\hat{\mathcal{P}}$. Given the ground truth transformation T_{gt} from $\hat{\mathcal{P}}$ to $\mathcal{P}$, our method minimizes the following mean-squared error:

$$L = \|TT_{gt}^{-1} - I\|^2 + \|\hat{T}T_{gt} - I\|^2, \tag{18}$$

where T is the 4×4 transformation matrix containing both rotation and translation and I is the identity matrix. We also experimented with various other loss functions, including the RMSE metric [10] used in our evaluation, but found that the simple MSE loss outperforms others by a small margin (see the supplement).

DeepGMR is fully-differentiable and non-iterative, which makes the gradient calculation more straightforward than previous iterative approaches such as PointNetLK [1]. We implemented the correspondence network and differentiable compute blocks using autograd in PyTorch [29]. For all our experiments, we trained the network for 100 epochs with batch size 32 using the Adam optimizer [25]. The initial learning rate is 0.001 and is halved if the validation loss has not improved for over 10 epochs.

5 Experimental Results

We evaluate DeepGMR on two different datasets: synthetic point clouds generated from the ModelNet40 [45] and real-world point clouds from the Augmented ICL-NUIM dataset [10,21]. Quantitative results are summarized in Table 1 and qualitative comparison can be found in Fig. 4. Additional metrics and full error distribution curves can be found in the supplement.

Table 1: Average RMSE and recall with threshold 0.2 on various datasets. DeepGMR achieves the best performance across all datasets thanks to its ability to perform robust data association in challenging cases (Sec. 5.2). Local methods are labeled in red; Inefficient global methods are labeled in orange and efficient global methods (average runtime < 1s) are labeled in blue. Best viewed in color.

	ModelNet clean		ModelNet noisy		ModelNet unseen		ICL-NUIM	
	RMSE ↓	Re@0.2 ↑	RMSE ↓	Re@0.2 ↑	RMSE ↓	Re@0.2 ↑	RMSE ↓	Re@0.2 ↑
ICP [8]	0.53	0.41	0.53	0.41	0.59	0.32	1.16	0.27
HGMR [15]	0.52	0.44	0.52	0.45	0.54	0.43	0.72	0.50
PointNetLK [1]	0.51	0.44	0.56	0.38	0.68	0.13	1.29	0.08
PRNet [43]	0.30	0.64	0.34	0.57	0.58	0.30	1.32	0.15
RANSAC10M+ICP	0.01	0.99	0.04	0.96	0.03	0.98	0.08	0.98
TEASER++ [46]	**0.00**	**1.00**	**0.01**	**0.99**	**0.01**	**0.99**	0.09	0.95
RANSAC10K+ICP	0.08	0.91	0.42	0.49	0.30	0.67	0.17	0.84
FGR [48]	0.19	0.79	0.2	0.79	0.23	0.75	0.15	0.87
DCP [42]	0.02	0.99	0.08	0.94	0.34	0.54	0.64	0.16
DeepGMR	**0.00**	**1.00**	**0.01**	**0.99**	**0.01**	**0.99**	**0.07**	**0.99**

Baselines We compare DeepGMR against a set of strong geometry-based registration baselines, including point-to-plane ICP [8], HGMR [15], RANSAC [17], FGR [48][3] and TEASER++ [46]. These methods cover the spectrum of point-based (ICP) and GMM-based (HGMR) methods, efficient global methods (FGR) as well as provably "optimal" methods (TEASER++). We also compare against state-of-the-art learning-based methods [1,42,43]. For RANSAC, we test two variants with 10K and 10M iterations and refine the results with ICP.

Metrics Following [10,48], we use the RMSE the metric designed for evaluating global registration algorithms in the context of scene reconstruction. Given a set of ground truth point correspondences $\{p_i, q_i\}_{i=1}^n$, RMSE can be computed as

$$E_{RMSE} = \frac{1}{n}\sqrt{\sum_{i=1}^{n} \|T(p_i) - q_i\|^2} \tag{19}$$

Since exact point correspondences rarely exist in real data, we approximate the correspondence using $\{p_i, T^*(p_i)\}_{i=1}^n$, where T^* is the ground truth transformation and p_i is a point sampled from the source point cloud, so Eq. 19 becomes

$$E_{RMSE} \approx \frac{1}{n}\sqrt{\sum_{i=1}^{n} \|T(p_i) - T^*(p_i)\|^2} \tag{20}$$

In our evaluation we use $n = 500$. In addition to the average RMSE, we also calculate the recall across the dataset, *i.e.*, the percentage of instances with RMSE less than a threshold τ. We followed [10] and used a threshold of $\tau = 0.2$.

[3] We performed a parameter search over the voxel size v, which is crucial for FGR's performance. We used the best results with $v = 0.08$.

5.1 Datasets

Synthetic data We generated 3 variations of synthetic point cloud data for registration from ModelNet40 [45], a large repository of CAD models. Each dataset follows the train/test split in [45] (9843 for train/validation and 2468 for test across 40 object categories).

ModelNet Clean. This dataset contains synthetic point cloud pairs without noise, so exact point-to-point correspondences exist between the point cloud pairs. Specifically, we sample 1024 points uniformly on each object model in ModelNet40 and apply two arbitrary rigid transformations to the same set of points to obtain the source and target point clouds. Note that similar setting has been used to evaluate learning-based registration methods in prior works [1,42], but the rotation magnitude is restricted, whereas we allow *unrestricted* rotation.

ModelNet Noisy The setting in *ModelNet clean* assumes exact correspondences, which is unrealistic in the real world. In this dataset, we perturb the sampled point clouds with Gaussian noise sampled from $\mathcal{N}(0, 0.01)$. Note that unlike prior works [1,42], we add noise independently to the source and target, breaking exact correspondences between the input point clouds.

ModelNet Unseen Following [42,43], we generate a dataset where train and test point clouds come from different object categories to test each method's generalizability. Specifically, we train our method using point clouds from 20 categories and test on point clouds from the held-out 20 categories unseen during training. The point clouds are the same as in *ModelNet noisy*.

Real-world Data To demonstrate applicability on real-world data, we tested each method using the point clouds derived from RGB-D scans in the Augmented ICL-NUIM dataset [10]. However, the original dataset contains only 739 scan pairs which are not enough to train and evaluate learning-based methods (a network can easily overfit to a small number of fixed transformations). To remedy this problem, we take every scan available and apply the same augmentation as we did for the ModelNet data, *i.e.*, resample the point clouds to 1024 points and apply two arbitrary rigid transformations. In this way, our augmented dataset contains both point clouds with realistic sensor noise and arbitrary transformation between input pairs. We split the dataset into 1278 samples for train/validation and 200 samples for test.

5.2 Analysis

From the quantitative results in Table 1, we can see that DeepGMR consistently achieves good performance across challenging scenarios, including noise, unseen geometry and real world data. TEASER++, the only baseline that is able to match the performance of DeepGMR on synthetic data, is over 1000 times slower (13.3s per instance) than DeepGMR (11ms per instance). In the rest of this section, we analyze common failure modes of the baselines and explain why DeepGMR is able to avoid these pitfalls. Typical examples can be found in the qualitative comparison shown in Fig. 4.

Table 2: Average running time (ms) of efficient registration methods on ModelNet40 test set. DeepGMR is significantly faster than other learning based method [1,42,43] and comparable to geometry-based methods designed for efficiency [15,48]. Baselines not listed (RANSAC10M+ICP, TEASER++) have running time on the order of 10s. OOM means a 16GB GPU is out of memory with a forward pass on a single instance.

# points	ICP	HGMR	PointNetLK	PRNet	RANSAC10K+ICP	FGR	DCP	DeepGMR
1000	184	33	84	153	95	22	67	**11**
2000	195	35	90	188	101	32	90	**19**
3000	195	37	93	OOM	113	37	115	**26**
4000	198	39	106	OOM	120	40	135	**34**
5000	201	**42**	109	OOM	124	**42**	157	47

Repetitive structures and symmetry Many point clouds contains repetitive structures (e.g. the bookshelf in the first row of Fig. 4) and near symmetry (e.g. the room scan in the last row of Fig. 4). This causes confusion in the data association for methods relying on distance-based point matching (e.g. ICP) or local feature descriptors (e.g. FGR). Unlike these baselines, DeepGMR performs data association globally. With the learned correspondence, a point can be associated to a distant component based on the context of the entire shape rather than a local neighborhood.

Parts with sparse point samples Many thin object parts (e.g. handle of the mug in the second row in Fig. 4) are heavily undersampled but are crucial for identifying correct alignments. These parts are easily ignored as outliers in an optimization whose objective is to minimize correspondence distances. Nevertheless, by doing optimization on a probabilistic model learned from many examples, DeepGMR is able to recover the correct transformation utilizing features in sparse regions.

Irregular geometry Among all the methods, FGR is the only one that performs significantly better on ICL-NUIM data than ModelNet40. This can be attributed to the FPFH features used by FGR to obtain initial correspondences. FPFH relies on good point normal estimates, and since the indoor scans of ICL-NUIM contain mostly flat surfaces with corners (e.g. the fourth row of Fig. 4), the normals can be easily estimated from local neighborhoods. In ModelNet40, however, many point clouds have irregular geometry (e.g. the plant in the third row of Fig. 4) which makes normal estimates very noisy. As a result, FGR's optimization fails to converge due to the lack of initial inlier correspondences.

5.3 Time Efficiency

We compute the average amount of time required to register a single pair of point clouds versus the point cloud size across all test instances in the ModelNet40. The results are shown in Table 2. Two baselines, RANSACK10M+ICP and TEASER++, are not listed because their average running time is over 10s for point clouds with 1000 points. ICP [8], FGR [48] and RANSAC10K+ICP are tested on an AMD RYZEN 7 3800X 3.9 GHz CPU. HGMR [15], PointLK [1] and

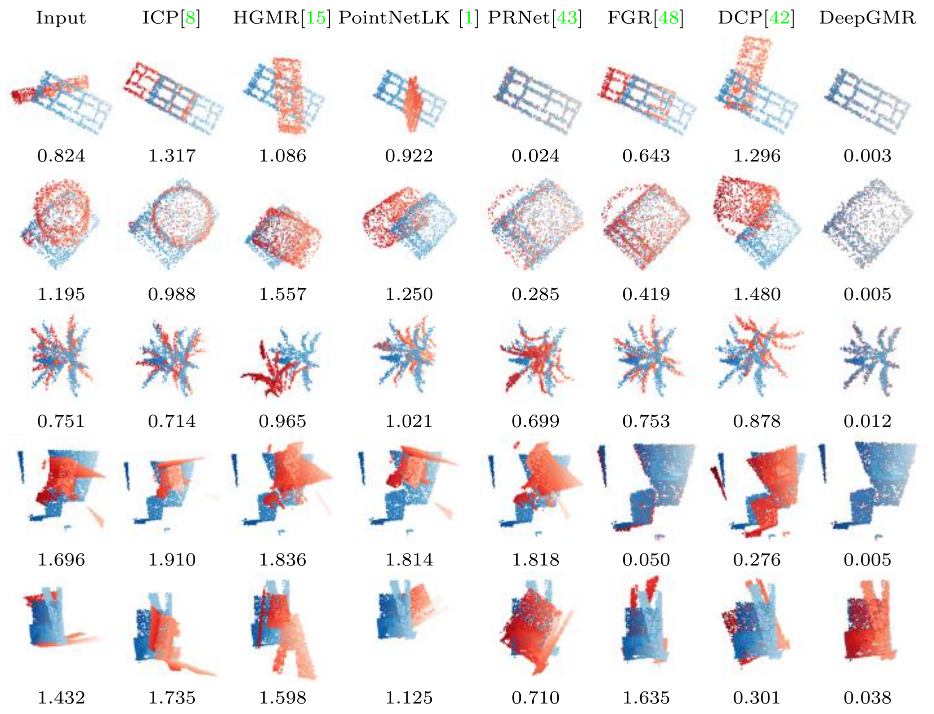


Fig. 4: Qualitative registration results on noisy ModelNet40 (top 3 rows) and ICL-NUIM point clouds (bottom 2 rows). The RMSE of each example is labeled below the plot. These examples highlight some typical failure modes of existing methods such as 1) ignoring parts with sparse point samples 2) erroneous data association due to repetitive structures and symmetry. DeepGMR avoids these errors by estimating consistent point-to-distribution correspondences and performing robust registration on learned GMMs.

DeepGMR are tested on a NVIDIA RTX 2080 Ti GPU. DCP [42] and PRNet [43] are tested on a NVIDIA Tesla V100 GPU because they require significant GPU memory. It can be seen that DeepGMR is the fastest among all methods thanks to its non-iterative nature and the reduction of registration to the task of learning correspondences to a low-dimensional latent probability distribution.

6 Conclusion

We have proposed DeepGMR, a first attempt towards learning-based probabilistic registration. Our experiments show that DeepGMR outperforms state-of-the-art geometry-based and learning-based registration baselines across a variety of data settings. Besides being robust to noise, DeepGMR is also generalizable to new object categories and performs well on real world data. We believe DeepGMR can be useful for applications that require accurate and efficient global alignment of 3D data, and furthermore, its design provides a novel way to integrate 3D neural networks inside a probabilistic registration paradigm.

References

1. Aoki, Y., Goforth, H., Srivatsan, R.A., Lucey, S.: Pointnetlk: Robust & efficient point cloud registration using pointnet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7163–7172 (2019) 2, 4, 10, 11, 12, 13, 14, 22, 24, 26, 27, 29
2. Beraldin, J.A., Blais, F., Cournoyer, L., Godin, G., Rioux, M.: Active 3d sensing. Modelli e metodi per lo studio e la conservazione dell'architettura storica, University: Scola Normale Superiore, Pisa **10** (01 2000) 1
3. Besl, P., McKay, H.: A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14**(2), 239–256 (1992). https://doi.org/10.1109/34.121791 3, 4
4. Bishop, C.M.: Pattern recognition and machine learning. springer (2006) 2
5. Campbell, D., Petersson, L.: Gogma: Globally-optimal gaussian mixture alignment. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5685–5694 (2016) 2, 4
6. Campbell, D., Petersson, L., Kneip, L., Li, H., Gould, S.: The alignment of the spheres: Globally-optimal spherical mixture alignment for camera pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 11796–11806 (2019) 4
7. Chen, C., Li, G., Xu, R., Chen, T., Wang, M., Lin, L.: Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4994–5002 (2019) 10, 23
8. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image and Vision Computing **10**(3), 145 – 155 (1992), range Image Understanding 4, 6, 11, 13, 14, 22, 24, 26, 27, 29
9. Chetverikov, D., Stepanov, D., Krsek, P.: Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm. Image and vision computing **23**(3), 299–309 (2005) 4
10. Choi, S., Zhou, Q.Y., Koltun, V.: Robust reconstruction of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5556–5565 (2015) 10, 11, 12
11. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. arXiv preprint arXiv:1904.08755 (2019) 10
12. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. pp. 1–38 (1977) 2, 4
13. Dhawale, A., Shaurya Shankar, K., Michael, N.: Fast monte-carlo localization on aerial vehicles using approximate continuous belief representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5851–5859 (2018) 4
14. Eckart, B., Kim, K., Troccoli, A., Kelly, A., Kautz, J.: Mlmd: Maximum likelihood mixture decoupling for fast and accurate point cloud registration. In: 3D Vision (3DV), 2015 International Conference on. pp. 241–249. IEEE (2015) 6, 9
15. Eckart, B., Kim, K., Kautz, J.: Hgmr: Hierarchical gaussian mixtures for adaptive 3d registration. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 705–721 (2018) 2, 3, 4, 5, 6, 11, 13, 14, 22, 24, 26, 27, 29
16. Evangelidis, G.D., Horaud, R.: Joint alignment of multiple point sets with batch and incremental expectation-maximization. IEEE transactions on pattern analysis and machine intelligence **40**(6), 1397–1410 (2017) 4

17. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981) 4, 11
18. Fitzgibbon, A.W.: Robust registration of 2d and 3d point sets. Image and Vision Computing **21**(13), 1145–1153 (2003) 4, 6
19. Gao, W., Tedrake, R.: Filterreg: Robust and efficient probabilistic point-set registration using gaussian filter and twist parameterization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 11095–11104 (2019) 4, 6
20. Granger, S., Pennec, X.: Multi-scale EM-ICP: A fast and robust approach for surface registration. ECCV 2002 pp. 69–73 (2002) 4, 6, 9
21. Handa, A., Whelan, T., McDonald, J., Davison, A.: A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). Hong Kong, China (May 2014) 10
22. Horaud, R., Forbes, F., Yguel, M., Dewaele, G., Zhang, J.: Rigid and articulated point registration with expectation conditional maximization. IEEE Trans. on Pattern Analysis and Machine Intelligence **33**(3), 587–602 (2011). https://doi.org/http://doi.ieeecomputersociety.org/10.1109/TPAMI.2010.94 6
23. Järemo Lawin, F., Danelljan, M., Shahbaz Khan, F., Forssén, P.E., Felsberg, M.: Density adaptive point set registration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3829–3837 (2018) 4, 6
24. Jian, B., Vemuri, B.C.: Robust point set registration using Gaussian mixture models. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1633–1645 (2011), `http://gmmreg.googlecode.com` 2, 4
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 10
26. Low, K.L.: Linear least-squares optimization for point-to-plane icp surface registration. Chapel Hill, University of North Carolina **4**(10) (2004) 4
27. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI. pp. 674–679 (1981) 4
28. Myronenko, A., Song, X.: Point set registration: Coherent point drift. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(12), 2262–2275 (2010) 6
29. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017) 10
30. Pomerleau, F., Colas, F., Siegwart, R.: A review of point cloud registration algorithms for mobile robotics. Found. Trends Robot **4**(1), 1–104 (May 2015) 2
31. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. Proc. Computer Vision and Pattern Recognition (CVPR), IEEE **1**(2), 4 (2017) 4, 10
32. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems. pp. 5099–5108 (2017) 4
33. Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: International Conference on 3-D Digital Imaging and Modeling. pp. 145–152 (2001) 4
34. Rusinkiewicz, S.: A symmetric objective function for ICP. ACM Transactions on Graphics (Proc. SIGGRAPH) **38**(4) (Jul 2019) 4
35. Segal, A., Haehnel, D., Thrun, S.: Generalized ICP. Robotics: Science and Systems **2**(4) (2009) 4, 6

36. Stoyanov, T.D., Magnusson, M., Andreasson, H., Lilienthal, A.: Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. International Journal of Robotics Research (2012) 4
37. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: Splatnet: Sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2530–2539 (2018) 4
38. Tabib, W., OMeadhra, C., Michael, N.: On-manifold gmm registration. IEEE Robotics and Automation Letters **3**(4), 3805–3812 (2018) 4
39. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. arXiv preprint arXiv:1904.08889 (2019) 10
40. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on Pattern Analysis & Machine Intelligence **13**(4), 376–380 (1991) 9, 21
41. Unnikrishnan, R., Lalonde, J., Vandapel, N., Hebert, M.: Scale selection for the analysis of Point-Sampled curves. In: 3D Data Processing Visualization and Transmission, International Symposium on. vol. 0, pp. 1026–1033. IEEE Computer Society, Los Alamitos, CA, USA (2006). https://doi.org/http://doi.ieeecomputersociety.org/10.1109/3DPVT.2006.123 2
42. Wang, Y., Solomon, J.M.: Deep closest point: Learning representations for point cloud registration. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019) 2, 3, 4, 6, 9, 11, 12, 13, 14, 22, 24, 26, 27, 29
43. Wang, Y., Solomon, J.M.: Prnet: Self-supervised learning for partial-to-partial registration. In: Advances in Neural Information Processing Systems. pp. 8812–8824 (2019) 4, 11, 12, 13, 14, 22, 24, 25, 26, 27, 29
44. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5), 146 (2019) 4
45. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015) 10, 12
46. Yang, H., Shi, J., Carlone, L.: Teaser: Fast and certifiable point cloud registration. arXiv preprint arXiv:2001.07715 (2020) 4, 11
47. Yang, J., Li, H., Campbell, D., Jia, Y.: Go-icp: A globally optimal solution to 3d icp point-set registration. IEEE transactions on pattern analysis and machine intelligence **38**(11), 2241–2254 (2015) 2, 4
48. Zhou, Q.Y., Park, J., Koltun, V.: Fast global registration. In: European Conference on Computer Vision. pp. 766–782. Springer (2016) 2, 3, 4, 11, 13, 14, 22, 24, 26, 27, 29

DeepGMR: Learning Latent Gaussian Mixture Models for Registration –Supplementary Material–

Wentao Yuan[1] Ben Eckart[2] Kihwan Kim[2]
Varun Jampani[2] Dieter Fox[1,2] Jan Kautz[2]

[1] University of Washington
[2] NVIDIA

1 Overview

In this document, we provide additional details, discussions, and experiments to support the original submission. Below is a summary of the contents.

- Sec. 2 provides detailed derivation and proofs for the $\mathbf{M_T}$ compute block described in Sec. 4.3 of the main paper.
- Sec. 3 contains auxiliary results, including error distribution curves (Fig. 1), ablation studies (Fig. 2), robustness tests (Fig. 3) and category-specific results (Table 1) in support of the major results in Sec. 5 of the main paper.
- Sec. 4 shows additional visualizations of registration results (Fig. 5) and the learned latent GMMs (Fig. 6).
- Sec. 5 discusses a limitation of our method and suggests directions for future research.

2 Additional Derivation

2.1 KL-divergence to Maximum Likelihood

We prove the conditions whereby Eq. (12) in the main paper is equivalent to Eq. (13), *i.e.* the conditions under which minimizing the KL-divergence from the transformed source distribution $T(\hat{\boldsymbol{\Theta}})$ to the target distribution $\boldsymbol{\Theta}$ is equivalent to maximizing the likelihood of the transformed source point cloud $T(\hat{\mathcal{P}})$ under the target distribution $\boldsymbol{\Theta}$.

Given two probability distributions $p(x)$ and $q(x)$ on $\mathcal{X}$, the KL-divergence between p and q is defined to be

$$\mathrm{KL}\left(p \mid q\right) = \int_{\mathcal{X}} p(x)\left[\ln p(x) - \ln q(x)\right] dx \tag{1}$$

$$= \mathbb{E}_{x\sim p(x)} \ln p(x) - \mathbb{E}_{x\sim p(x)} \ln q(x) \tag{2}$$

Thus, we can write the KL-minimization problem in Eq. (12) of the main paper as follows,

$$T^* = \operatorname*{argmin}_T \mathrm{KL}\left(T(\hat{\boldsymbol{\Theta}}) \mid \boldsymbol{\Theta}\right) \tag{3}$$

$$= \operatorname*{argmin}_T \mathbb{E}_{x\sim p(x\mid T(\hat{\boldsymbol{\Theta}}))} \ln p(x \mid T(\hat{\boldsymbol{\Theta}})) - \mathbb{E}_{x\sim p(x\mid T(\hat{\boldsymbol{\Theta}}))} \ln p(x \mid \boldsymbol{\Theta}) \tag{4}$$

Note that the first term, the negative entropy of $p(x|T(\hat{\boldsymbol{\Theta}}))$, is invariant with respect to T, so we end up with

$$T^* = \underset{T}{\operatorname{argmax}} \mathbb{E}_{x \sim p(x|T(\hat{\boldsymbol{\Theta}}))} \ln p(x \mid \boldsymbol{\Theta}) \tag{5}$$

Thus, minimizing the KL-divergence from $T(\hat{\boldsymbol{\Theta}})$ to $\boldsymbol{\Theta}$ is equivalent to maximizing the expected log likelihood of data distributed according to $T(\hat{\boldsymbol{\Theta}})$ under $\boldsymbol{\Theta}$. Or, in other words, minimizing the cross-entropy. Note that the transformed source point cloud $T(\hat{\mathcal{P}}) = \{T(\hat{p}_i)\}_{i=1}^N$ are sampled *iid* from the distribution $p(x|T(\hat{\boldsymbol{\Theta}}))$. Using the law of large numbers, given a suitably large point cloud, we can approximate the expectation in Eq. (5) as

$$\underset{T}{\operatorname{argmax}} \mathbb{E}_{x \sim p(x|T(\hat{\boldsymbol{\Theta}}))} \ln p(x|\boldsymbol{\Theta}) \approx \underset{T}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^N \ln p(T(\hat{p}_i) \mid \boldsymbol{\Theta}) \tag{6}$$

$$= \underset{T}{\operatorname{argmax}} \sum_{i=1}^N \ln \sum_{j=1}^J \pi_j \mathcal{N}(T(\hat{p}_i) \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \tag{7}$$

which gives us Eq. (13) in the main paper.

2.2 Single Sum Reduction

We show how to reduce the NJ pairs of distances in Eq. (16) of the main paper to J pairs of distances in Eq. (17) of the main paper using the output of $\mathbf{M}_{\boldsymbol{\Theta}}$ (Eqs. (7,8,9) in the main paper).

The calculations inside $\mathbf{M}_{\boldsymbol{\Theta}}$ (Eqs. (7,8,9) in the main paper) determine the relationships between the correspondence matrix $\hat{\Gamma} = \{\hat{\gamma}_{ij}\}_{i,j=1,1}^{N,J}$, the point coordinates $\hat{\mathcal{P}} = \{\hat{p}_i\}_{i=1}^N$ and the GMM parameters $\hat{\boldsymbol{\Theta}} = \{\hat{\pi}_j, \hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j\}_{j=1}^J$. Specifically, we can rewrite Eqs. (7,8) in the main paper as

$$\sum_{i=1}^N \hat{\gamma}_{ij} = N\hat{\pi}_j \tag{8}$$

$$\sum_{i=1}^N \hat{\gamma}_{ij} T(\hat{p}_i) = N\hat{\pi}_j T(\hat{\boldsymbol{\mu}}_j) \tag{9}$$

To prove the latter identity, note that the 3D rigid transformation T is a linear operator. Therefore,

$$\sum_{i=1}^N \hat{\gamma}_{ij} T(\hat{p}_i) = T\left(\sum_{i=1}^N \hat{\gamma}_{ij} \hat{p}_i\right) \tag{10}$$

$$= T(N\hat{\pi}_j \hat{\boldsymbol{\mu}}_j) \tag{11}$$

$$= N\hat{\pi}_j T(\hat{\boldsymbol{\mu}}_j) \tag{12}$$

Next, we expand the right hand side of Eq. (16) in the main paper, which contains NJ pairs of distances, using Eqs. (8,9).

$$\sum_{i=1}^{N}\sum_{j=1}^{J}\hat{\gamma}_{ij}\|T(\hat{p}_i)-\boldsymbol{\mu}_j\|^2_{\boldsymbol{\Sigma}_j} \tag{13}$$

$$=\sum_{j=1}^{J}\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-2\sum_{j=1}^{J}\boldsymbol{\mu}_j^\top\boldsymbol{\Sigma}_j^{-1}\sum_{i=1}^{N}\hat{\gamma}_{ij}T(\hat{p}_i)+\sum_{j=1}^{J}\|\boldsymbol{\mu}_j^\top\|^2_{\boldsymbol{\Sigma}_j}\sum_{i=1}^{N}\hat{\gamma}_{ij} \tag{14}$$

$$=\sum_{j=1}^{J}\left(\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-2\boldsymbol{\mu}_j^\top\boldsymbol{\Sigma}_j^{-1}N\hat{\pi}_jT(\hat{\boldsymbol{\mu}}_j)+\|\boldsymbol{\mu}_j^\top\|^2_{\boldsymbol{\Sigma}_j}N\hat{\pi}_j\right) \tag{15}$$

Now, we complete the square by adding $N\hat{\pi}_j\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j}$ to the latter two terms in the summation and subtracting it from the first term. For the latter two terms, we have

$$N\hat{\pi}_j\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j}-2\boldsymbol{\mu}_j^\top\boldsymbol{\Sigma}_j^{-1}N\hat{\pi}_jT(\hat{\boldsymbol{\mu}}_j)+\|\boldsymbol{\mu}_j^\top\|^2_{\boldsymbol{\Sigma}_j}N\hat{\pi}_j \tag{16}$$

$$=N\hat{\pi}_j\left(\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j}-2\boldsymbol{\mu}_j^\top\boldsymbol{\Sigma}_j^{-1}T(\hat{\boldsymbol{\mu}}_j)+\|\boldsymbol{\mu}_j^\top\|^2_{\boldsymbol{\Sigma}_j}\right) \tag{17}$$

$$=N\hat{\pi}_j\|\boldsymbol{\mu}_j-T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{18}$$

For the first term, we have

$$\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-N\hat{\pi}_j\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{19}$$

$$=\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-2N\hat{\pi}_j\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j}+N\hat{\pi}_j\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{20}$$

$$=\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-2N\hat{\pi}_jT(\hat{\boldsymbol{\mu}}_j)^\top\boldsymbol{\Sigma}_j^{-1}T(\hat{\boldsymbol{\mu}}_j)+\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{21}$$

$$=\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)\|^2_{\boldsymbol{\Sigma}_j}-\sum_{i=1}^{N}\hat{\gamma}_{ij}T(\hat{p}_j)^\top\boldsymbol{\Sigma}_j^{-1}T(\hat{\boldsymbol{\mu}}_j)+\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{22}$$

$$=\sum_{i=1}^{N}\hat{\gamma}_{ij}\|T(\hat{p}_i)-T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{23}$$

Eq. (21) and (22) uses the relationship in Eq. (8) and (9) respectively. Notice that the result in Eq. (23) is invariant to T because we assume Σ_j is isotropic. Therefore, if we are optimizing over T, we can reduce Eq. (15), *i.e.* the right hand side of Eq. (16) in the main paper, to the single term in Eq. (18), which gives us

$$T^* = \operatorname*{argmin}_T \sum_{i=1}^{N} \sum_{j=1}^{J} \hat{\gamma}_{ij} \|T(\hat{p}_i) - \boldsymbol{\mu}_j\|^2_{\boldsymbol{\Sigma}_j} \tag{24}$$

$$= \operatorname*{argmin}_T \sum_{j=1}^{J} N\hat{\pi}_j \|\boldsymbol{\mu}_j - T(\hat{\boldsymbol{\mu}}_j)\|^2_{\boldsymbol{\Sigma}_j} \tag{25}$$

$$= \operatorname*{argmin}_T \sum_{j=1}^{J} \frac{\hat{\pi}_j}{\sigma_j^2} \|T(\hat{\boldsymbol{\mu}}_j) - \boldsymbol{\mu}_j\|^2 \tag{26}$$

This is exactly Eq. (17) in the main paper.

2.3 SVD Solution

We derive the solution to the weighted ICP criterion in Eq. (17) of the main paper using a weighted version of Umeyama's method [40]. First, we center the data and construct the cross-covariance matrix M,

$$\boldsymbol{\mu}_c \overset{\text{def}}{=} \sum_{j=1}^{J} \hat{\pi}_j \boldsymbol{\mu}_j \tag{27}$$

$$\hat{\boldsymbol{\mu}}_c \overset{\text{def}}{=} \sum_{j=1}^{J} \hat{\pi}_j \hat{\boldsymbol{\mu}}_j \tag{28}$$

$$M \overset{\text{def}}{=} \sum_{j=1}^{J} \frac{\hat{\pi}_j}{\sigma_j^2} (\boldsymbol{\mu}_j - \boldsymbol{\mu}_c)(\hat{\boldsymbol{\mu}}_j - \hat{\boldsymbol{\mu}}_c)^T \tag{29}$$

Assuming $T \in SE(3)$, given the SVD decomposition of $M = USV^T$, the optimal rotation R^* and translation $\mathbf{t}^*$ are as follows,

$$R^* = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det VU^T \end{bmatrix} U^T \tag{30}$$

$$\mathbf{t}^* = \boldsymbol{\mu}_c - R^* \hat{\boldsymbol{\mu}}_c \tag{31}$$

The center matrix in Equation (30) comes from the fact that we want to enforce $\det R^* = +1$ to prevent reflections.

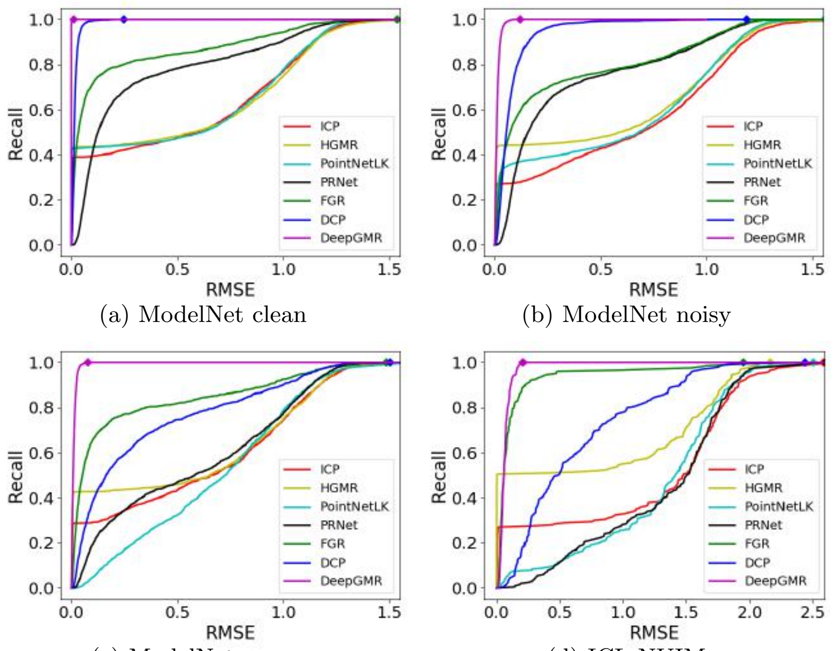
3 Additional Quantitative Results

3.1 Full Error Distribution

Fig. 1 contains the cumulative distribution function (CDF) curves of the RMSE metric for the methods tested in Sec. 5 of the main paper. To be specific, a point (x, y) on the curve implies that fraction y of the instances in the test set has RMSE less than x. The CDF curves show the complete error distribution which reveals more information than a single metric. In fact, recall@0.2 shown in the main paper is a single point on the CDF curve with $x = 0.2$.

A couple of observations can be drawn from the error distribution

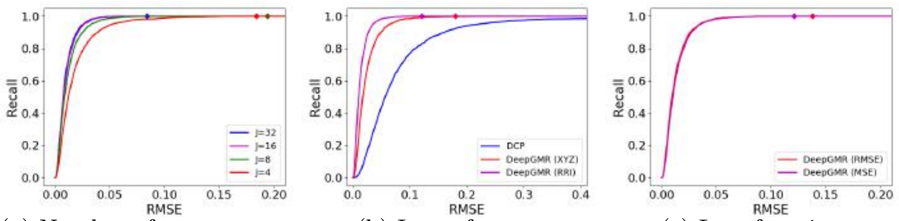
- Some local methods such as ICP [8] and HGMR [15] are quite accurate on a fraction of instances (in particular, those with small transformations).
- Methods based on point-to-point (ICP [8], DCP [42], PRNet [43]) and feature correspondences (FGR [48], PointNetLK [1]) performs worse on noisy data, whereas methods based on probabilistic data association (HGMR [15], DeepGMR) are unaffected.
- Learning-based methods (PointNetLK [1], DCP [42], PRNet [43]) except DeepGMR perform significantly worse on data from unseen categories, which shows that the generalization to unseen data demonstrated in these works does not hold in the case of unrestricted rotation.



(a) ModelNet clean (b) ModelNet noisy

(c) ModelNet unseen (d) ICL-NUIM

Fig. 1: Cumulative distribution function (CDF) of RMSE metric on the test set of the evaluation datasets in Sec. 5 of the main paper. A point (x, y) on the curve indicates the method achieve a recall of y with threshold x on that dataset. The diamonds show where the CDF reaches 1, i.e. the maximum error across the entire test set for that method. If there is no diamond, it means the maximum error is beyond the x-axis limit.



(a) Number of components (b) Input feature (c) Loss function

Fig. 2: Ablation studies on **(a)** the number of GMM components J, **(b)** the input to correspondence network (RRI [7] or raw XYZ) and **(c)** loss function. Ablated models are compared using CDF of RMSE on the ModelNet noisy test set. The diamonds show where the CDF reaches 1, i.e. the maximum error across the entire test set for that method.

3.2 Ablation Studies

We perform ablations on several design choices mentioned in Sec. 4 of the main paper, including the number of GMM components J, the input to correspondence network f_ψ and the loss function. The dataset used is ModelNet noisy. The results are compared using CDF of RMSE on the test set shown in Fig. 2.

Number of GMM components It can be seen that the performance of DeepGMR saturates with $J > 16$, so we use $J = 16$ across our experiments.

Input feature While the performance of DeepGMR is indeed improved with RRI features [7], DeepGMR taking raw xyz coordinates still outperforms the most competitive baseline DCP.

Loss function We trained DeepGMR directly with the RMSE metric used for evaluation (Eq. 20 of the main paper) and compared it to DeepGMR trained with the simple MSE loss in Eq. 18 of the main paper. We found the two models perform almost identically (the MSE-trained model has slightly lower maximum error). This shows that DeepGMR is not sensitive to the particular choice of loss function. Although it is possible to design a better loss function, it is not the focus of our work.

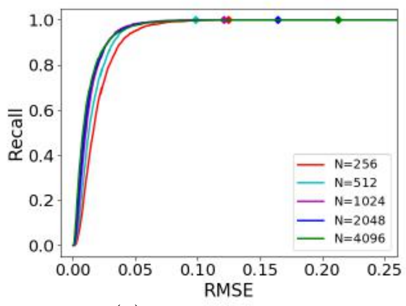
3.3 Robustness Tests

We perform additional tests on the robustness of DeepGMR to input point density and transformation magnitude. The results are shown in Fig. 3.

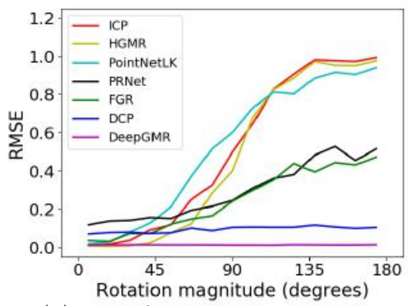
Point density Because DeepGMR performs registration in the latent GMM space, it is invariant to the density of input point clouds. To demonstrate this, we test the DeepGMR model trained on ModelNet noisy in Sec. 5 of the main paper on point clouds with various density without any finetuning. Here, we can use the number of points N as a proxy for density since the point clouds are uniformly sampled from same surface. From the results in Fig. 3a, it can be seen that the performance of DeepGMR is unaffected on point clouds up to 4 times denser than training and is only slightly worse on point clouds up to 4 times sparser, which can be attributed to missing geometric details in sparse point clouds.

We note that the accuracy of methods that depend on hand-crafted feature correspondences, e.g. FGR, may improve with more input points as better normals can be estimated. However, we found that this is only true on data without noise. With 4096 input points, the accuracy of FGR improves on ModelNet clean (0.04 RMSE, 0.96 recall), but stays the same on ModelNet noisy (0.22 RMSE, 0.77 recall). This test indicates that FGRs normal estimation accuracy is more contingent on the noise level than on the sampling density.

Input transformation magnitude DeepGMR learns latent correspondences between points and GMM components that are *pose-invariant*, which means that its output is invariant to the magnitude of the transformation between the input point clouds. From Fig. 3b, we can see that DCP [42], another learning-based global method, shares the same invariance property while the performance of local methods (ICP [8], HGMR [15], PointNetLK [1]) degrades significantly with larger transformation. We also observe an interesting class of registration methods including FGR [48] and PRNet [43]. The formulations of these methods are global but they rely on feature matching or keypoint detection that become unstable with larger transformation, which makes their performance worse on these cases.



(a) Point density



(b) Transformation magnitude

Fig. 3: Robustness tests. **(a)** RMSE CDF curves of the same DeepGMR model (trained with $N = 1024$) tested on point clouds of different density (density is measured by the number of subsampled points N). **(b)** RMSE vs magnitude of ground truth rotation between source and target on ModelNet noisy test set.

3.4 Per-category Results

Table 1 compares the average RMSE within each of the 40 categories in ModelNet noisy. DeepGMR achieves consistently good performance across all categories, while the baseline methods struggle with objects that have rotation or reflection symmetry (e.g. bowl, glass box), repetitive structure (e.g. bookshelf, stairs) or thin parts (e.g. radio, lamp).

4 Additional Qualitative Results

4.1 Registration Results

More qualitative registration results on ModelNet noisy and ICL-NUIM are shown in Fig. 5. It can be seen that DeepGMR is able to deal with challenging cases that trap other methods in local minima, such as repetitive structure, undersampled thin parts and non-planar geometry, which demonstrate the consistency and robustness of the correspondences learned by DeepGMR.

4.2 GMM Visualization

We show more visualization of the learned GMM and correspondence in Fig. 6. We can see that different object parts are assigned to different GMM components consistently across views. Note that no explicit supervision is provided on the correspondence. Everything is learned end-to-end with the registration objective.

5 Future Work

One limitation of DeepGMR is that it does not explicitly consider partial overlap, i.e. when the IoU between source and target point clouds is less than 1 after alignment. The reason is that DeepGMR estimates the correspondence between *all* points and *all* components in the latent GMM. In the case of partial overlap, however, it is more ideal to estimate a partial correspondence, i.e. the correspondence between *some* of the points and *some* of the components in the latent GMM.

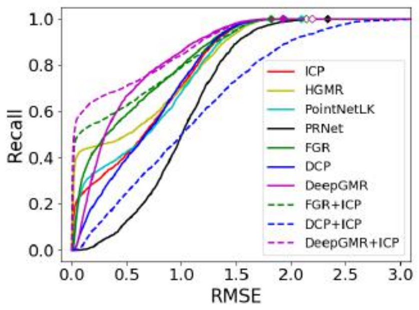
To measure the consequence of this limitation, we performed a preliminary experiment on partial data artificially created from ModelNet40. Specifically, we generate partial point clouds by approximating the rendering procedure of an orthographic depth camera. First, we randomly rotate the complete point cloud and project the points onto a zero-centered grid of dimension 200×200 and size 2×2 (same size as the bounding box of the point cloud, which is normalized to $[-1, 1]^3$ across the dataset) on the xy-plane. Then, for each grid cell, we keep one point with the smallest z value and throw away the others. In this way, we end up with a partial point cloud that closely resembles the observation of a depth camera. Finally, we add independent Gaussian noise to the points.

Experimental results on this partial dataset are summarized in Fig. 4. On one hand, we note that even though DeepGMR does not explicitly consider partial overlap, its performance is still competitive. In addition, if we apply a refinement stage afterwards (i.e. use the prediction of a global method as the initialization of a local method such as ICP), DeepGMR achieves the best performance on this partial dataset. This demonstrates the power of the robust data association learned by DeepGMR. On the other hand, PRNet [43], a prior work that explicitly considers partial overlap, fails on this dataset. This shows that their method of dealing with partial overlap only works with limited transformation magnitude, i.e. it is a local registration method.

Although DeepGMR is able to outperform baselines on partial overlap data with the help of local refinement, its performance is still far below its performance on completely overlapping data. Therefore, a promising future research direction is to combine the robust point-to-latent-GMM correspondence learned by DeepGMR with techniques that deal with partial overlap (e.g. the attention mechanisms in [42,43]).

	RMSE ↓	Re@0.2 ↑
ICP [8]	2.45	0.29
HGMR [15]	0.58	0.45
PointNetLK [1]	0.66	0.33
PRNet [43]	0.79	0.12
FGR [48]	0.50	0.43
DCP [42]	0.68	0.19
DeepGMR	0.46	0.34
FGR+ICP	0.44	0.55
DCP+ICP	3.46	0.09
DeepGMR+ICP	**0.34**	**0.64**

(a) RMSE and recall



(b) CDF curves

Fig. 4: Results on ModelNet partial: **(a)** Average RMSE and recall with threshold 0.2; **(b)** CDF of RMSE. Local methods outperform global methods on a fraction of instances with small transformations but fail on the remaining ones. DeepGMR+ICP, a global+local method that uses the output of DeepGMR as the initialization for ICP, achieves the best overall performance. Although DeepGMR by itself is not as accurate as in the case of complete overlap, it is able to bring most instances in the convergence basin of local methods. Best viewed in color.

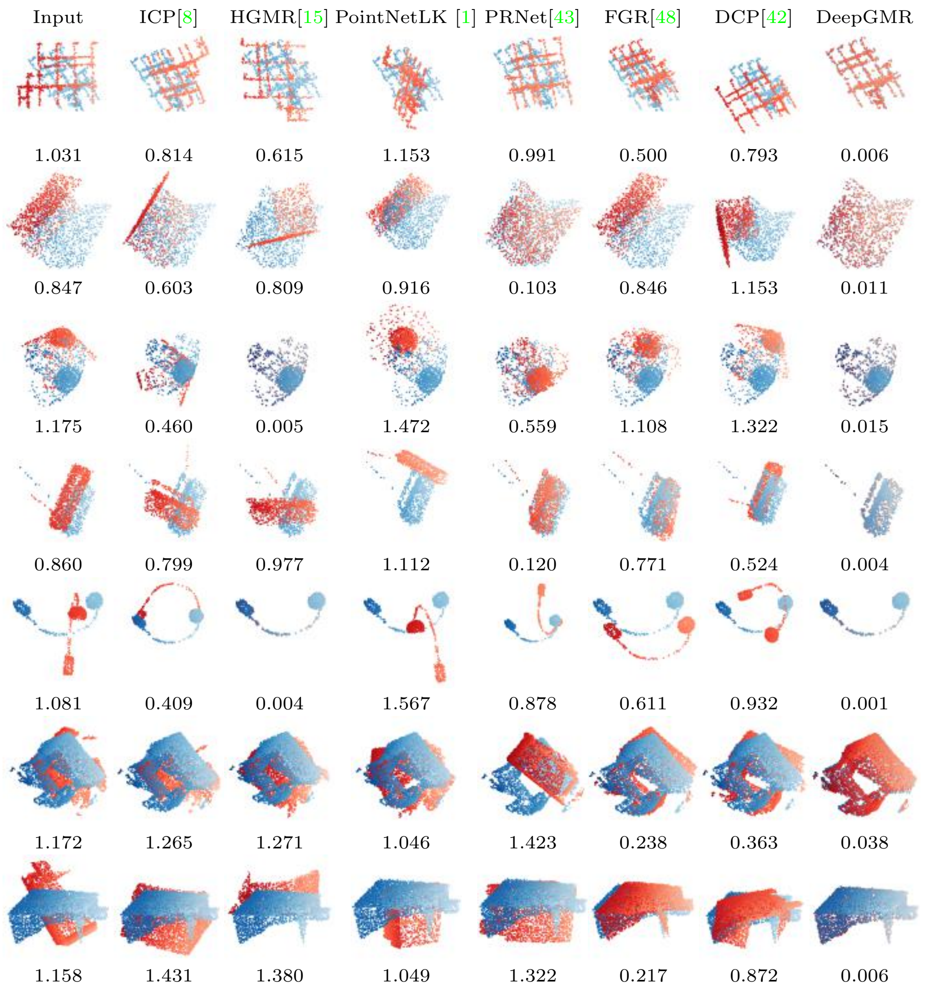


Fig. 5: Qualitative registration results on ModelNet40 noisy (top 5 rows) and ICL-NUIM point clouds (bottom 2 rows). The RMSE of each example is labeled below the plot. These examples highlight some typical failure modes of existing methods such as 1) ignoring parts with sparse point samples 2) erroneous data association due to repetitive structures and symmetry. DeepGMR avoids these errors by estimating consistent point-to-distribution correspondence and performing robust registration on GMMs.

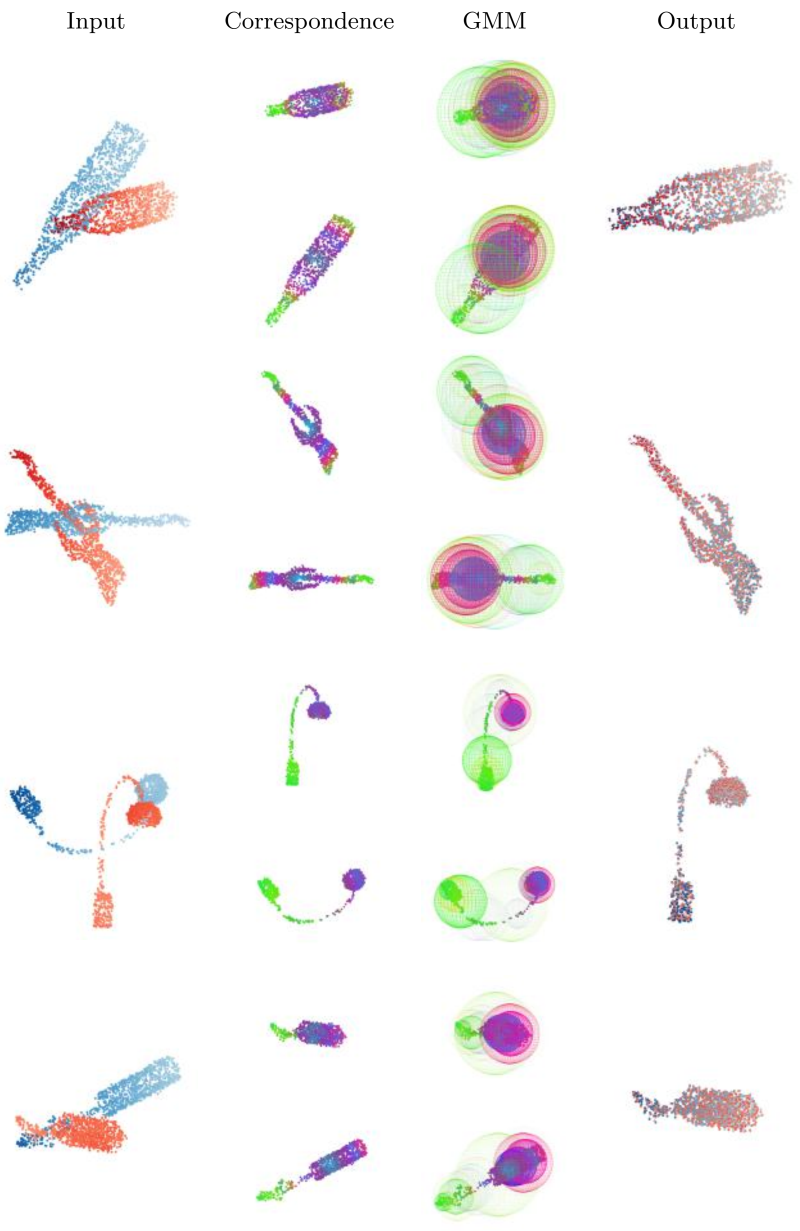


Fig. 6: Visualization of learned correspondences and GMMs. In the second and third column, each color indicates a different GMM component. The point colors are calculated as weighted averages of the component colors according to the learned correspondences. The radius of each sphere is equal to the standard deviation of the GMM component. Note how DeepGMR learns to correspond the points and GMMs in the source (red in first column, top in each row) and target (blue in first column, bottom in each row) without any explicit supervision.

Table 1: Comparison of average RMSE within each category on ModelNet noisy

Category	ICP[8]	HGMR[15]	PointNetLK [1]	PRNet[43]	FGR[48]	DCP[42]	DeepGMR
airplane	0.49	0.48	0.51	0.24	0.15	0.08	**0.01**
bathtub	0.53	0.43	0.62	0.31	0.30	0.13	**0.01**
bed	0.46	0.66	0.52	0.20	0.24	0.08	**0.01**
bench	0.51	0.62	0.42	0.41	0.38	0.15	**0.02**
bookshelf	0.58	0.50	0.56	0.33	0.22	0.14	**0.01**
bottle	0.51	0.55	0.51	0.27	0.25	0.07	**0.01**
bowl	0.77	0.76	0.80	0.57	0.61	0.15	**0.02**
car	0.41	0.44	0.47	0.23	0.16	0.09	**0.01**
chair	0.55	0.51	0.55	0.25	0.20	0.09	**0.01**
cone	0.50	0.77	0.54	0.24	0.30	0.13	**0.02**
cup	0.64	0.67	0.72	0.48	0.39	0.10	**0.01**
curtain	0.58	0.41	0.41	0.34	0.36	0.10	**0.01**
desk	0.58	0.54	0.53	0.21	0.18	0.11	**0.01**
door	0.55	0.53	0.55	0.26	0.60	0.14	**0.02**
dresser	0.57	0.51	0.59	0.34	0.23	0.10	**0.02**
flower pot	0.27	0.51	0.37	0.25	0.15	0.09	**0.01**
glass box	0.65	0.68	0.61	0.41	0.47	0.11	**0.02**
guitar	0.40	0.52	0.36	0.31	0.47	0.05	**0.00**
keyboard	0.55	0.44	0.53	0.47	0.45	0.07	**0.01**
lamp	0.59	0.78	0.37	0.27	0.18	0.07	**0.01**
laptop	0.32	0.53	0.51	0.37	0.37	0.09	**0.01**
mantel	0.56	0.48	0.56	0.27	0.23	0.08	**0.01**
monitor	0.61	0.48	0.59	0.27	0.30	0.10	**0.01**
night stand	0.63	0.58	0.56	0.36	0.27	0.11	**0.01**
person	0.43	0.27	0.42	0.27	0.19	0.07	**0.00**
piano	0.55	0.58	0.59	0.24	0.15	0.08	**0.01**
plant	0.51	0.50	0.47	0.27	0.21	0.07	**0.01**
radio	0.37	0.71	0.43	0.26	0.33	0.09	**0.01**
range hood	0.56	0.50	0.54	0.31	0.25	0.07	**0.01**
sink	0.51	0.60	0.54	0.30	0.16	0.09	**0.01**
sofa	0.43	0.50	0.49	0.35	0.18	0.10	**0.01**
stairs	0.63	0.53	0.58	0.29	0.46	0.15	**0.01**
stool	0.76	0.71	0.63	0.20	0.25	0.13	**0.01**
table	0.59	0.46	0.60	0.34	0.51	0.14	**0.01**
tent	0.52	0.53	0.51	0.29	0.21	0.10	**0.01**
toilet	0.53	0.49	0.48	0.23	0.11	0.08	**0.01**
tv stand	0.47	0.50	0.50	0.29	0.20	0.09	**0.01**
vase	0.61	0.65	0.64	0.36	0.34	0.12	**0.02**
wardrobe	0.65	0.63	0.61	0.36	0.43	0.10	**0.02**
xbox	0.46	0.57	0.47	0.28	0.25	0.06	**0.01**