Registration Loss Learning for Deep Probabilistic Point Set Registration

Felix Järemo Lawin and Per-Erik Forssén
Computer Vision Laboratory, Department of Electrical Engineering, Linköping University, Sweden
{felix.jaremo-lawin, per-erik.forssen}@liu.se

1. mixture component?

2. registration loss 如何定义?

Abstract

Probabilistic methods for point set registration have interesting theoretical properties, such as linear complexity in the number of used points, and they easily generalize to joint registration of multiple point sets. In this work, we improve their recognition performance to match state of the art. This is done by incorporating learned features, by adding a von Mises-Fisher feature model in each mixture component, and by using learned attention weights. We learn these jointly using a registration loss learning strategy (RLL) that directly uses the registration error as a loss, by back-propagating through the registration iterations. This is possible as the probabilistic registration is fully differentiable, and the result is a learning framework that is truly end-to-end. We perform extensive experiments on the 3DMatch and Kitti datasets. The experiments demonstrate that our approach benefits significantly from the integration of the learned features and our learning strategy, outperforming the state-of-the-art on Kitti. Code is available at https://github.com/felja633/RLLReg.

1. Introduction

Point set registration is a fundamental computer vision problem, that has applications in 3D mapping and scene understanding. The task is to find the relative geometric transformations between scene observations, represented as unstructured point samples. There is a huge amount of different ways to approach point set registration, useful overviews can be found in e.g. [39, 5].

This work extends the paradigm of probabilistic point set registration [18, 14, 25]. In this paradigm, the scene is represented as a *Gaussian Mixture Model* (GMM) of the point set density. This formulation has interesting theoretical properties, such as linear complexity in the number of used points, and it easily generalizes to joint registration of multiple point sets [14]. The recent top-performing methods in registration benchmarks, however, all use learned features [5, 21]. To improve the probabilistic methods, we thus extend them to also benefit from learned features. To this end, we add a von Mises-Fisher feature model in each mixture component to model the local feature distribution.

In addition, this work also introduces a framework for true end-to-end training of feature descriptors in a single training phase. This is possible as the probabilistic registration is fully differentiable. Descriptor learning is normally done in the contrastive paradigm, with a loss such as the triplet loss [29] or its variants [7], that encourages similarity of matches, while reducing similarity of mismatches. The proposed *registration loss learning* (RLL) instead directly uses the registration error as a loss, by back-propagating through the registration iterations. This idea was also used in PointNetLK [1], to learn a global descriptor vector. Here, we instead learn *local features* that are well suited to perform registration, also under partial overlap and varying point density. Using the registration error as a loss also provides an easy way to generate test data, by simply perturbing known registrations. Each training batch is thus composed of registration trajectories estimated using the current descriptor weights. The proposed formulation further allows us to perform training and testing on an arbitrary number of point sets jointly, making our approach easily adaptable to different applications.

Contributions: Our contributions are summarized as follows: **(i)** We extend probabilistic point set registration with a feature model, that can accomodate powerful deep features. **(ii)** The features are learned using *registration loss learning* (RLL), by back-propagating gradients through EM iterations. To avoid instability during training, we propose a robust loss function that is insensitive to large registration errors. **(iii)** As RLL is end-to-end, it allows simultaneous learning of pointwise attention weights, that focus registration on important regions.

Figure 1 gives an overview of the proposed approach. It is tested by extensive experiments on the 3DMatch and Kitti odometry datasets and an ablative analysis highlights the impact of the proposed components. The experiments demonstrate that our approach benefits significantly from the integration of the feature model and outperforms the state-of-the-art on Kitti.

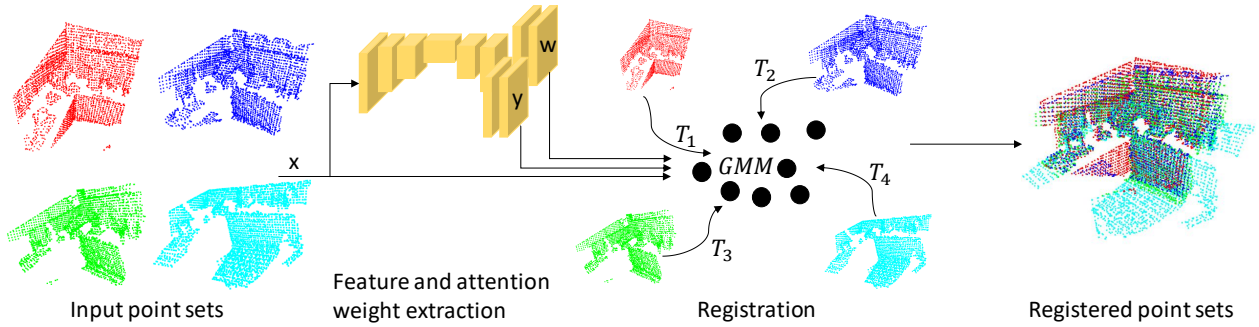


Figure 1. Overview of our registration algorithm. First, a CNN extracts features, y, and attention weights, w, from the input point sets, x. The points and features are then modeled using a GMM with a von Mises-Fisher distribution for each component to model the features. Finally, the parameters of the GMM, the feature model, and the transformations T_i are jointly refined until convergence.

2. Related work

Point set registration has received a high interest over the years. In this section, we outline related works categorized as classical, feature matching, and probabilistic methods.

Classical: The iterative closest point (ICP) based methods [4, 2, 26], iteratively find putative correspondences given the current transformation parameters through a nearest neighbor search in the spatial domain. While being efficient, the ICP methods rely heavily on a good initialization of the transformation parameters. More robust variants of ICP have been proposed, employing multiple restarts [23], graph optimization [31], or branch and bound [3] to find globally optimal transformation parameters.

Another branch of methods are based on point neighborhood density [32]. These methods find the relative transformation between point sets by modeling them as densities using a kernel, and then minimize a loss based on correlation or divergence [18]. The density based losses have improved monotonicity compared to the L^2 loss used in ICP [30].

Feature matching: In order to increase the accuracy of the correspondence set, it is beneficial to use geometric features [28]. Such methods generally employ RANSAC based frameworks to obtain the transformation parameters given putative feature correspondences. The FGR method [37] employs geometric features and iteratively optimizes the correspondence set by minimizing a robust loss.

Since matching approaches are reliant on the discriminative properties of the features, more recent approaches [35, 10, 10, 7, 11, 17, 5] aim at learning these using deep networks. The networks are generally based on e.g. PointNet [27] or sparse 3D convolutions [6] and are usually trained using contrastive learning techniques. With these features, registration can be performed by first establishing correspondences through feature matching and then employing a RANSAC scheme to obtain a robust estimation [35, 10, 10, 7, 11, 21]. The RANSAC step was replaced in recent approaches [17, 5] by further learning an importance weighting of the correspondences. A robust estimate of the transformation parameters can then be found using a weighted Procrustes solver, which in contrast to RANSAC is fully differentiable. Consequently, these methods can benefit from end-to-end learning using the registration objective as a loss. The work in [17] additionally proposes an approach for joint registration of multiple point sets using an iterative refinement of pairwise registrations.

The works in [33, 34, 22] extend ICP by employing features in the correspondence establishment step. In [1], a distance is minimized, between global representations of the points using a gradient decent optimization. While showing promising results on synthetic data, the methods in [33, 34, 1] fail to generalize to real world point sets [5].

Probabilistic: Probabilistic approaches jointly model the underlying point set distribution and infer the transformations. As a result, the putative correspondences become probabilistic, leading to less sensitivity to measurement noise and the initial transformation. In the Coherent Point Drift (CPD) approach [25], the source point cloud is modeled as a GMM, conditioned on the target point set. The parameters of the GMM and the transformation parameters are then optimized using expectation maximization. A downside of the CPD approach is its high computational complexity. To remedy this, recent approaches employ hierarchical GMMs [12] or a fast filtering formulation [15] to achieve similar performance at significantly higher speed. The CPD approach was generalized in [14, 13] to multi-view registration by also optimizing the GMM cluster centers instead of fixing them to the source point set. In [20], a point set density weighting was proposed for the probabilistic methods to handle density variations in the point set.

Feature models were incorporated in the probabilistic methods to increase robustness in [8, 9]. These models, however, were based on color channels and handcrafted geometric features. In this work we aim to bridge the gap between feature matching based and probabilistic approaches by learning both point features and attention weights.

3. Method

We propose a probabilistic method for point set registration, which exploits the powers of learned features. Based on the generative model in [14], we introduce a strategy for integrating and learning deep feature representations of the local point regions. Specifically, we model the joint distribution of points and features as a Gaussian Mixture Model (GMM), where each component represents the density of the spatial coordinates and features in a local region. Hence, instead of employing point matching techniques, correspondences are indirectly represented as soft assignments to the components. This enables a generalization to joint registration of an arbitrary number of point sets. Moreover, the computational complexity is linear with the number of mixture components and points. This allows us to significantly reduce the complexity in comparison to previous deep approaches by keeping the number of components low. Finally, registration is performed by jointly optimizing the parameters of the GMM and the transformation parameters using Expectation Maximization (EM). The EM iterations are both efficient and fully differentiable, allowing us to learn features for registration in an end-to-end manner by minimizing a loss based on the registration error.

Our registration algorithm can be broken down into two steps, as illustrated in Figure. 1; a feature extraction step, and a registration step. The feature extractor network processes each point set separately and generates pointwise features and attention weights. These are incorporated into a probabilistic model, which is optimized with EM in the registration step to obtain the transformation parameters. In the following sections we detail the approach.

3.1. Probabilistic Point Set Registration

In this section we provide a generic description of probabilistic registration. Let $\mathcal{X}_i = \{x_{ij}\}_{j=1}^{N_i}, i = 1, \dots, M$ be M point sets, consisting of 3D-point observations $x_{ij} \in \mathbb{R}^3$ independently drawn from the distributions p_{X_i}. We assume that these distributions are all related to a common scene distribution p_V by the rigid 3D transformations $T_{\omega^i} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ parameterized by ω^i such that $p_V(T_{\omega^i}(x)|\theta) = p_{X_i}(x|\theta)$. The aim is to jointly infer $p_V(v|\theta)$, with parameters θ, and the optimal transformation parameters for $i = 1, \dots M$.

We model the density $p_V(v|\theta)$ as a mixture of Gaussian distributions,

$$p_V(v|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(v; \mu_k, \Sigma_k)\,. \tag{1}$$

Here, $\mathcal{N}(v; \mu, \Sigma)$ is a normal density function with mean μ and covariance Σ. The number of components is denoted by K and π_k is the mixing weight of component k satisfying $\sum_k \pi_k = 1$. In order to reduce the number of parameters in the mixture model, we consider fixed uniform mixing weights $\pi_k = 1/K$ and isotropic $\Sigma_k = \sigma_k^2 \mathbf{I}$. We denote the set of all mixture parameters by $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ and the set of all parameters in the model by $\Theta = \{\theta, \omega^1, \dots, \omega^M\}$. For brevity, in the following sections we omit the parameters ω^i in the notation for the transformation functions, i.e we set $T_i = T_{\omega^i}$.

3.1.1 Inference

To infer the parameters Θ, we maximize the log-likelihood of the observed points x_{ij}:

$$\mathcal{L}(\Theta; \mathcal{X}_1, \dots, \mathcal{X}_M) = \sum_{i}^{M} \sum_{j}^{N_i} \log(p_V(T_i(x_{ij})|\theta))\,. \tag{2}$$

Maximization is done using EM, where we introduce the latent assignment variables $Z \in (1, \dots, K)$, assigning points to the mixture components. Given the current parameter state Θ^n, an EM iteration maximizes the expected complete data log-likelihood objective,

$$\mathcal{Q}(\Theta; \Theta^n) = \sum_{i}^{M} \sum_{j}^{N_i} E_{Z|x,\Theta^n}\left[\log\left(p_{V,Z}(T_i(x_{ij}), Z|\theta)\right)\right] w_{ij}. \tag{3}$$

Here, w_{ij} is a pointwise attention weighting that determines the importance of each point in the objective. In [20], w_{ij} was introduced as an inverse local density estimate to account for variations in the sample densities p_{X_i}. Another option which we will explore is to learn a predictor for w_{ij}.

In the E-step, we evaluate the expectation in (3), taken over the probability distribution of the assignment variables,

$$\begin{aligned} p_{Z|X_i}(k|x, \Theta) &= \frac{p_{X_i,Z}(x, k|\Theta)}{\sum_{l=1}^{K} p_{X_i,Z}(x, l|\Theta)} \\ &= \frac{\pi_k \mathcal{N}(T_i(x); \mu_k, \Sigma_k)}{\sum_{l=1}^{K} \pi_l \mathcal{N}(T_i(x); \mu_l, \Sigma_l)}\,. \end{aligned} \tag{4}$$

For compactness, we introduce the notation $\alpha_{ijk}^n = p_{Z|X_i}(k|x_{ij}, \Theta^n)$. In the subsequent M-step, we maximize,

$$Q(\Theta; \Theta^n) = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \alpha_{ijk}^n w_{ij} \log\left(p_{V,Z}(T_i(x_{ij}), k|\theta)\right)\,, \tag{5}$$

with respect to Θ.

Due to the non-linearities of (5), a closed form solution to the joint maximization problem over θ and T_i is unlikely to exist. As in [14, 13] and [25], we circumvent this by employing the *expectation conditional maximization* (ECM) method. In this approach, the transformations T_i are found first, given the latent assignments and the mixture parameters θ. Afterwards the mixture parameters are found

given the latent assignments and the transformation parameters. As both sub-problems have closed-form solutions, this leads to an efficient optimization procedure. We refer to [14] for the derivations of T_i, μ_k and σ_k.

3.2. Probabilistic Feature Representation

The formulation in the previous section only considered the spatial coordinates of the points in the probabilistic registration model. In this section we extend the model to the joint distribution $(X, Y) \sim p_{X,Y}(x, y)$ over the spatial coordinates x and features y of the points. Specifically, we assume that the features are invariant to the transformations T. Further, as in [9], we also assume that features and spatial locations are independent, conditioned on the assignment variable Z. We then maximize the likelihood for a model over the joint density function $p_{X,Y}(x, y|\Theta, \nu)$. Here, we have introduced feature model parameters ν. The conditional independence between X and Y, and the assumption that Y is invariant to Θ, allow us to factorize p as,

$$\begin{aligned} p(x, y, k|\Theta, \nu) &= p(x, y|k, \Theta, \nu)p(k|\Theta, \nu) \\ &= p(x|k, \Theta)p(k|\Theta, \nu)p(y|k, \nu)\,. \end{aligned} \tag{6}$$

Again, we assume that the mixing weights $\pi_k = p(k|\Theta, \nu)$ are constant and equal for all components. Using the above factorization, the E-step turns into evaluation of the following posterior distribution,

$$\begin{aligned} p_{Z|X_i,Y}(k|x, y, \Theta, \nu_k) &= \frac{p_{X_i,Y,Z}(x, y, k|\Theta, \nu_k)}{\sum_{l=1}^{K} p_{X_i,Y,Z}(x, y, l|\Theta, \nu_l)} \\ &= \frac{p_{X_i|Z}(x|k, \Theta)p_Z(k|\Theta)p_{Y|Z}(y|k, \nu_k)}{\sum_{l=1}^{K} p_{X_i|Z}(x|l, \Theta)p_Z(l|\Theta)p_{Y|Z}(y|l, \nu_l)} \\ &= \frac{\pi_k \mathcal{N}(T_i(x); \mu_k, \Sigma_k)p_{Y|Z}(y|k, \nu_k)}{\sum_{l=1}^{K} \pi_l \mathcal{N}(T_i(x); \mu_l, \Sigma_l)p_{Y|Z}(y|l, \nu_l)}\,. \end{aligned} \tag{7}$$

Note that, the terms in the expression are identical to the terms in (4), except for the added $p_{Y|Z}(y|k, \nu_k)$. In the M-step, we maximize,

$$\begin{aligned} &F(\Theta; \Theta^n) = \\ &\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{k=1}^{K} w_{ij}\alpha_{ijk}^n \log(p_{V,Y,Z}(T_i(x_{ij}), y_{ij}, z = k|\theta, \nu_k)) \\ &= Q(\Theta; \Theta^n) + \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{k=1}^{K} w_{ij}\alpha_{ijk} \log p_Y(y_j|z = k, \nu_k)\,. \end{aligned} \tag{8}$$

Here, Q is identical to the objective in the M-step i equation (5) and only depends on T_i and θ. It can thus be maximized using the procedure in section 3.1.1. The second term contains the feature model and only depends on the feature parameters ν_k. In total, we optimize the parameters of the original spatial distribution of the probabilistic model along with one feature parameter vector ν_k for each component.

3.2.1 Feature Model

We aim to integrate powerful deep features in the probabilistic model. Deep features are generally high dimensional, providing a discriminative representation of the local appearance of a given point. To be able to efficiently model these features and enable end-to-end learning, we propose a von Mises-Fisher distribution [24], with parameters ν_k to represent the density $p_Y(y_{ij}|z = k, \nu_k)$. We therefore assume that the feature vectors are normalized, i.e. $\|y\| = 1$. Thus, in the E-step we compute,

$$p_{Y|Z}(y|k, \nu_k) \propto e^{\nu_k^T y / s^2} \tag{9}$$

In the M-step, we update ν_k by solving

$$\max_{\nu_k} \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\alpha_{ijk}\nu_k^T y_{ij}\,, \quad \text{subject to } \|\nu_k\| = 1\,. \tag{10}$$

The solution to the above maximization problem is

$$\nu_k = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\alpha_{ijk}y_{ij}}{\left\|\sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\alpha_{ijk}y_{ij}\right\|}\,. \tag{11}$$

We provide a derivation of (11) in the supplementary material. Note that, s can be considered as a parameter to be optimized [24]. In this work, however, we treat it as a hyper-parameter kept fixed during inference. In our experiments, we set $s = 0.4$.

3.2.2 Feature Extractor

We construct a feature extractor F_ϕ, parameterized by ϕ. The feature extractor takes 3D coordinates of a point set as input and outputs features y and attention weights w for each point. Similar to the recent works [17, 5], we base our network on the sparse 3D conv-net in [7]. This network comprises four encoder blocks and decoder blocks which are connected in a Unet-based structure. Each block consists of a conv-layer, a ReLU activation, and a batch normalization layer.

On top of the final block, our network generates feature maps y, and attention weight maps w in two separate streams. Both streams consist of two conv-layers with a ReLU in between. The output of the feature stream is L^2-normalized to fit the distribution of our probabilistic feature model in section 3.2.1. In the attention stream we produce a scalar that is input to a SoftPlus activation. This ensures both positive weighting and large flexibility.

3.2.3 Registration Algorithm

In this section we outline the details of our registration algorithm. For a given set of M overlapping point sets $\{\mathcal{X}_i\}_1^M$, we seek the unknown relative transformations T_{ij}.

First, we apply the feature extractor $F_\phi(\mathcal{X}_i)$ on the point sets $\mathcal{X}_i, i = 1, \dots, M$ individually, to obtain features $\{y_{ij}\}_{j=1}^{N_i}$ and attention weights $\{w_{ij}\}_{j=1}^{N_i}$. Next, we employ our probabilistic registration algorithm. When starting from an unknown GMM, its parameters need to be initialized. We initialize the means μ_k of the spatial mixture components by randomly sampling points on a sphere centered at the mean of the point sets, and with a radius equal to the point sets standard deviation. Further, we initialize the standard deviations σ_k of the mixture components as the maximum distance between the points. We assume that the feature distribution is uniform in the first iteration, thus it has no impact on the first transformation estimate. Next, we apply the EM algorithm by iterating the E and M-steps (see Sections 3.1.1 to 3.2.1). In order to reduce the sensitivity to the initialization step, we keep μ_k fixed during the first two iterations. After N_{iter} iterations, we obtain the final transformation estimates $\{T_i\}$, which map the point sets from their local coordinates to the GMM frame.

3.3. Registration Loss Learning

We learn the parameters ϕ of the feature extractor by minimizing a registration loss over a dataset $\mathcal{D} \subset 2^{\{\mathcal{X}_i\}_1^M}$, consisting of D groups $\mathcal{D}_d \in \mathcal{D}$ of overlapping point sets. This loss can be written as

$$\mathcal{L}(\phi; \mathcal{D}) = \sum_{d=1}^{D} \mathcal{R}(\phi; \mathcal{D}_d)\,. \tag{12}$$

We denote the number of point sets in each group by $M_d = |\mathcal{D}_d|$. In principle, M_d can be an arbitrary number of point sets, but on order to reduce training speed and memory consumption, we fix it to two or four sets in our experiments. We further assume that the ground truth relative transformations T_{ik}^{gt} from point set k to i are known in the training set. Similar to [17], the sample loss $\mathcal{R}$ is computed over pointwise errors given the estimated relative transformations $T_{ik}^n = (T_i^n)^{-1} \circ T_k^n$ for each EM iteration n as

$$\mathcal{R}(\phi;\mathcal{D}_d) = \sum_{n}^{N_{\text{iter}}} v_n \sum_{i=1,k=i+1}^{M_d-1,M_d} \frac{1}{N_i} \sum_{j=1}^{N_i} \rho(\left\| T_{ik}^n(x_{ij}) - T_{ik}^{\text{gt}}(x_{ij}) \right\|_2 / c). \tag{13}$$

The estimated transformations T_i^n and T_k^n are found using the proposed EM approach. We observed that large errors in the the registration loss causes instability during training. To counter this we employ a Geman-McClure penalty ρ. This is applied to the Euclidean distance between the points mapped by the estimated transformation and the ground truth transformation. The parameter c is a scale factor that determines the steepness of the error function. Crucially, this robust penalty function ensures that the loss is not dominated by samples far from the solution. To further counter the influence of examples far from convergence, the loss is weighted with $v_n = 1/(40 - n)$, such that the later iterates will have a larger impact.

As all steps in the EM algorithm are differentiable, we can update the parameters ϕ by back-propagating the errors of the loss in (12) through the registration procedure. The feature extractor can thus be trained end-to-end to maximize registration accuracy.

Details: We train our approach on datasets of overlapping point set pairs using the ADAM optimizer with initial learning rate of 0.004 and a batch size of 6. We split the training into 180 epochs with 2000 samples each. Every 40th epoch we reduce the learning rate by a factor of 0.2. In order to increase variations in the dataset we apply random rotations and translations to the point sets. Specifically, to construct the rotations we draw random rotation axes and rotate with an angle between 0 and $\pi/8$ radians. The random translation vectors are drawn from a uniform distribution and have a random length depending on the dataset. For large scale lidar datasets we draw translation vectors with a maximum norm of 2 meters while for smaller scale RGB-D datasets the maximum length is set to 0.8 meters. During training we set the number of mixture components $K = 50$. Further, we found in our experiments that using $N_{\text{iter}} = 23$ EM iterations gives good results (see Section 4.1).

4. Experiments

We evaluate our approach on the RGB-D sequences from the 3DMatch datset [36] and the lidar point clouds from the Kitti odometry dataset [16]. 3DMatch contains sequences from 62 scenes, 54 of which are for training and 8 for testing. We use 7 sequences from the training set for validation and hyper-parameter tuning. Kitti contains scans from 11 scenes (00-10), including 8 for training and 3 for testing. We use (00-05) for training and (06-07) for validation.

In all experiments, we collect samples by first drawing a reference frame from a sequence. We then randomly sample frames that are nearby, in terms of both frame id and distance. For 3DMatch, we draw samples that are within 50 frames and 2 meters from the reference frame. Since the point sets in Kitti have a larger scale, we instead draw samples that are within 100 frames and 15 meters. Following [7], we also make sure that the samples have at least 30% overlap with the reference frame. In both 3DMatch and Kitti, ground truth relative poses are given and we use these to measure the error in rotation angle and Euclidean distance in translation. For all methods in the evaluations, we pre-process the point sets with voxel grid downsampling. We set the voxel side length to 5 cm for 3DMatch and 30 cm for Kitti.

In the following sections, we first provide an ablative analysis on the 3DMatch dataset to show the impact of different components of our method. Next, we compare

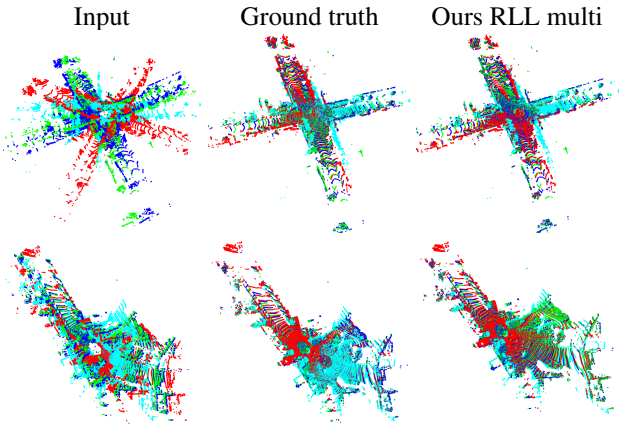


Figure 2. Joint registration of multiple point sets from Kitti. Our approach successfully registers these point sets despite partial overlap and large deviations in initial rotations and translations.

our approach to a number of state-of-the-art methods for pairwise and multi-view registration on both 3DMatch and Kitti. Example registrations on the Kitti dataset are visualized in Figure 2. More examples and a runtime analysis are provided in the supplementary.

4.1. Ablative analysis

In this section we analyze the impact of the different components in our approach. First, we analyze the impact of the proposed feature model and the registration loss learning (RLL). To this end, we generate the different versions of our approach listed below.

Baseline: As a baseline approach we employ a GMM which only models the spatial coordinates of the points. Since the downsampled RGB-D point clouds only have minor variations in density, the baseline does not benefit from density adaptive weighting [20]. Hence, we set the weights w to be equal, and this makes the baseline is equivalent to the JRMPC [14] method.

Contrastive: In this version, we employ our proposed feature model (see Section. 3.2.1) using pre-trained features from [7] with $C = 32$ channels. We denote this version as "contrastive", since the features have been obtained using contrastive learning. Like the baseline, this version does not employ any pointwise weighting.

RLL unweighted: Next, we integrate RLL of the feature extractor (see Section 3.3). The feature extractor is trained from scratch and is outputting $C = 16$ channel feature descriptors. As in the contrastive version, we are not using any pointwise weighting here.

RLL weighted: We further include the attention weights w in the model. These are learned along with the features using RLL.

RLL weighted 32: For fair comparison with the contrastive version, we also include a version using $C = 32$ channel feature descriptors.

RLL L2 weighted: To evaluate the impact of the robust loss ρ in (13), we include a version of RLL with attention weights which have been trained using the L^2 loss without Geman-McClure.

We compare the above variants on the 3DMatch dataset. For simplicity, we only consider pairwise registration in this analysis. We sample 1000 pairs randomly as described above. All variants are assigned $K^{\text{run}} = 100$ mixture components and we set the number of EM iterations to $N_{\text{iter}}^{\text{run}} = 100$ during runtime.

The ablative analysis is summarized in Table 1. We report the success rates, and average errors for the method variants. A registration is considered successful if the rotation error is less than 4° and the translation error is less than 10 cm. The results show that the contrastive version significantly outperforms the baseline with an increased success rate of 11.9%. This demonstrates that the proposed feature model is able to exploit the descriptive features from the contrastive learning process. By further including the RLL, we improve the success rate over contrastive learning with 2.7%. The best performance is obtained when we include the learned attention weighting with an additional improvement of 2.7%. We further observe that increasing to 32 channels feature descriptors do not improve the performance. Finally, we see that the proposed robust loss improves over the L^2 loss in success rate with 3.1%.

Moreover, we analyze the impact of the number of EM iterations (N_{iter}) used during training. The above versions of our approach with RLL are trained with $N_{\text{iter}} = 23$. In this experiment we also include versions with 9, 15 and 29 iterations. From Table 1, we observe that all versions improve over the baseline. A significant improvement is gained by increasing N_{iter} from 9 to 15. Note that, at 15 iterations we already improve over contrastive learning. We further observe that increasing from 15 to 29 iterations slightly reduces the performance. The best performance, however, is achieved at 23 iterations. In the following sections, registration loss learning with learned weights, and $N_{\text{iter}} = 23$ during training, is referred to as Ours RLL.

4.2. State-of-the-art

3DMatch pairwise: We compare our approach to the state-of-the-art for pairwise registration on the 3Dmatch dataset, using the same point set pairs as in the ablative analysis. In this comparison, we include the classical ICP point-to-point and ICP point-to-plane using implementations from Open3D [38]. We further include the probabilistic methods JRMPC [14], FPPSR [8] and FilterReg [15] (point-to-point implementation from [19]). For FPPSR we employ both FPFH [28] and FCGF [7] features. Additionally, we include the global correspondence based methods FGR [37] (Open3D) and the recent DGR [5], which uses deep features

method	Success rate %	Avg. success err (°)	Avg. success err (cm)
baseline	62.5	1.2	3.2
Contrastive	73.1	1.2	3.3
RLL $N_{\text{iter}} = 23$	76.3	1.2	3.4
RLL $N_{\text{iter}} = 23$+weights	79.5	1.1	3.1
RLL $N_{\text{iter}} = 23$+weights $C = 32$	78.9	1.1	3.1
RLL L2 $N_{\text{iter}} = 23$+weights	76.8	1.2	3.2
RLL $N_{\text{iter}} = 29$+weights	75.1	1.1	3.1
RLL $N_{\text{iter}} = 15$+weights	77.2	1.2	3.2
RLL $N_{\text{iter}} = 9$+weights	66.9	1.3	3.5

Table 1. Ablative analysis of on the 3DMatch test split. Successes are registrations with a rotation error less than four degrees and a translation error of less than 10 cm. Within the successful registrations we compute average rotation and translation errors. Our proposed feature model significantly outperforms the baseline without features. The best result is obtained by employing our proposed registration loss learning and learned attention weighting.

method	Success rate %	Avg. inlier err (°)	Avg. inlier err (cm)	Avg. time (s)
ICP-p2p[4]	48.9	1.3	3.7	0.6*
ICP-p2plane[2]	59.3	1.2	3.3	1.1*
FGR[37]	66.1	1.4	3.4	0.9*
DGR[5]	85.8	1.2	3.2	2.5
JRMPC[14]	62.5	1.2	3.2	0.7
FPPSR+FCGF[8]	67.4	1.2	3.3	1.5
FPPSR+FPFH[8]	59.0	1.1	3.0	2.3
FILTERREG[15]	7.1	2.0	6.1	1.3*
Ours Contrastive	73.1	1.2	3.3	0.8
Ours RLL	79.5	1.1	3.1	0.8

Table 2. Pairwise registrations on 3DMatch. Registrations with a rotation error less than four degrees and a translation error of less than 10 cm are categorized as successful. Within the successful registrations we compute average rotation and translation errors. Methods marked with * in the Avg. time column were run using CPU only, while the others were also run on GPU.

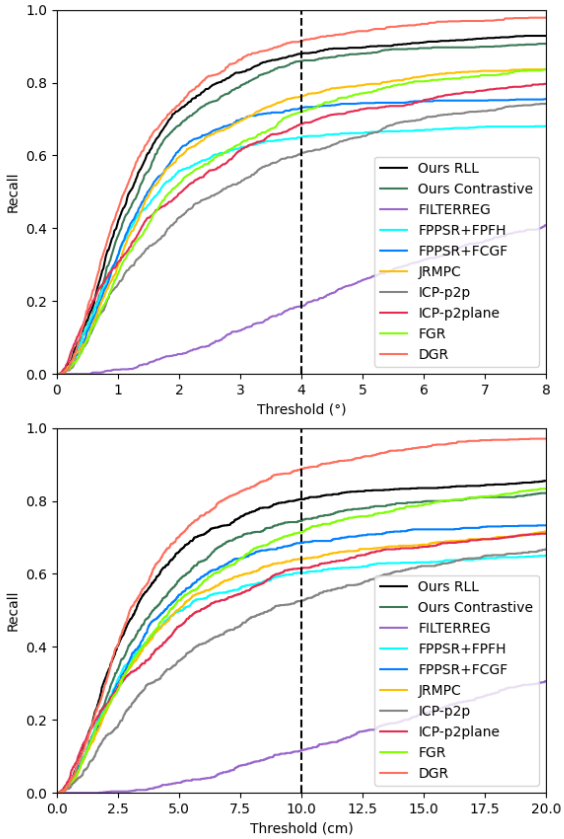


Figure 3. Pairwise recall curves on 3DMatch for errors in rotation (top) and translation (bottom).

and a RANSAC fallback.

The results are presented as recall curves for both the rotation and translation errors in Figure 3. As in the ablative analysis, we also present success rates and success mean rotation and translation errors in Table 2. Additionally, we list the average runtimes over 38 samples. For all methods, the runtimes include pre-processing, feature extraction and registration. Our approaches, DGR, FPPSR and JRMPC are run on GPU, while the other methods run on CPU.

We observe that both the contrastive and registration loss learning versions of our approach significantly outperforms all classical and probabilistic approaches. Further, the registration loss learning version outperforms FGR with a margin of 14% in success rate. Among the compared methods, our registration loss learning version is only outperformed by the recent DGR with a margin of 6%. On the other hand, our approach runs 2.6 times faster than DGR, which should make it more useful in many online applications.

Kitti pairwise: We further evaluate our approach on the Kitti odometry test sequences for pairwise registration. Samples are generated by randomly drawing 1000 pairs as described above (see Section 4). In order to adapt our approach to the lidar scans, we perform training on the split described in Section 3.3. We compare our approach with ICP point-to-point [4], ICP point-to-plane [2], DARE [20], FPPSR [8], FilterReg [15], FGR [37] and DGR [5]. Despite voxel grid downsampling, the density variations of the lidar point set are still high, causing the performance of the GMM based methods to drop significantly. To counter this, we employ the density adaptive weighting in [20] for the methods DARE, FPPSR and Ours Contrastive. We further observed that DGR struggles with finding the correct translations. Therefore, we include a version of DGR with an ICP point-to-plane refinement step, denoted DGR+ICP.

Recall curves over rotation and translation errors are reported in Figure 4. In Table 3, we report success rates, success mean rotation and translation errors[1]. For the Kitti results, we regard registrations with a rotation error smaller than 4 degrees and a translation error smaller than 30 cm as being successful. We see that our approach benefits from the proposed registration loss learning, outperforming contrastive learning with a margin of 23.2%. Moreover, since Ours RLL is using the learned attention weighting instead

[1]The Kitti dataset results reported here are slightly different from the published paper, as the experiments have been rerun. In the published version, the observation weights were missing for the FPPSR methods. Adding the weights did not change any conclusions however.

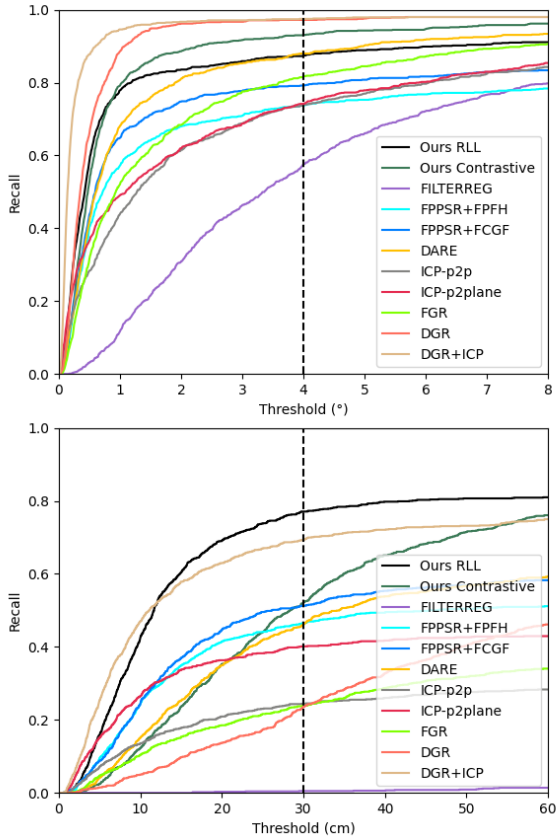


Figure 4. Pairwise recall curves on Kitti for errors in rotation (top) and translation (bottom).

method	Success rate %	Avg. inlier err (°)	Avg. inlier err (cm)	Avg. time (s)
ICP-p2p[4]	24.4	0.3	10.6	0.4*
ICP-p2plane[2]	40.1	0.3	8.8	0.6*
FGR[37]	24.2	0.4	13.4	3.0*
DGR[5]	23.5	0.3	17.6	2.1
DGR+ICP[5]	69.3	0.1	8.8	2.5
DARE[20]	45.9	0.4	14.6	1.1
FPPSR+FCGF[8]	51.6	0.4	14.1	4.5
FPPSR+FPFH[8]	38.3	0.4	14.4	5.8
FILTERREG[15]	0.5	0.8	20.8	2.0*
Ours Contrastive	51.5	0.4	16.2	1.2
Ours RLL	76.9	0.4	10.5	0.6

Table 3. Pairwise registrations on Kitti. Registrations with a rotation error less than four degrees and a translation error of less than 30 cm are categorized as successful. Within the successful registrations we compute average rotation and translation errors. Methods marked with * in the Avg. time column were run using CPU only, while the others were also run on GPU.

of the density adaptive weighting in [20], it also runs faster than both DARE and Ours Contrastive.

Among the previous methods DGR with ICP refinement has the highest performance. While DGR+ICP has a higher inlier rate for rotations and translation errors within 10 cm, Ours RLL has an overall success rate which is 8.4% higher. Moreover, Ours RLL runs 4.3 times faster than DGR+ICP.

3DMatch mutli-view: We evaluate multi-view registration on 3DMatch by creating a dataset of 500 samples with four overlapping point sets each. Each registration results in six pairwise transformations. We use these to evaluate the success rates and mean rotation and translation errors in Table 4 (top). We compare our approaches to FPPSR and JRMPC since these methods directly handle multi-view registration. For this experiment we also include a version of our approach which has been trained on four views per sample, denoted Ours RLL multi. Our RLL based approaches achieve the highest success rates. We also observe that our approach benefits from training on multi-view samples.

3DMatch				
method	Success rate %	Avg. inlier err (°)	Avg. inlier err (cm)	Avg. time (s)
JRMPC[14]	52.9	1.2	3.4	0.8
FPPSR+FCGF[8]	65.1	1.2	3.4	2.3
FPPSR+FPFH[8]	60.1	1.2	3.2	4.1
Ours Contrastive	71.4	1.2	3.4	1.0
Ours RLL	78.6	1.2	3.1	1.1
Ours RLL multi	79.3	1.2	3.1	1.1
Kitti				
DARE[20]	37.8	0.4	15.0	1.4
FPPSR+FCGF[8]	43.7	0.4	14.4	7.2
FPPSR+FPFH[8]	27.3	0.4	15.0	9.6
Ours Contrastive	48.2	0.5	15.8	1.6
Ours RLL	68.6	0.4	11.2	0.8
Ours RLL multi	69.6	0.4	11.3	0.8

Table 4. Multi-view registration on 3DMatch (top) and Kitti (bottom), with four point sets in each sample. For 3DMatch, registrations with a rotation error less than four degrees and a translation error of less than 10 cm are categorized as successful. Successful registrations on Kitti samples have rotation error less that four degrees and translation error less than 30 cm. Within the successful registrations we compute average rotation and translation errors.

Kitti mutli-view: For multi-view registration on Kitti we create a dataset of 500 samples of four overlapping point sets for each sample. As as in the 3DMatch multi-view experiment above, we use all six pairwise relative transformations in the evaluation in Table 4 (bottom)[1]. We compare our approaches to FPPSR and DARE. For this experiment we also include a version of our approach that has been trained on multi-view samples, denoted Ours RLL multi. As for the pairwise experiments, these results are generated from a rerun after the paper reviews due to missing observation weigths in the FPPSR methods. Our RLL based approaches achieve the highest success rates. We further observe that Ours RLL multi outperforms Ours RLL in terms of success rate.

5. Conclusions

We have extended the paradigm of probabilistic point set registration to exploit the discriminative powers of learned features and weights. To learn the features and weights, we propose a registration loss learning strategy that trains the network in an end-to-end manner in a single phase. Our experiments demonstrate that the extension significantly out-

performs previous probabilistic methods for both pairwise and multi-view registration.

Acknowledgments: This work was supported by the ELLIIT Excellence Center and the Vinnova through the Visual Sweden network 2019-02261.

References

[1] Y. Aoki, H. Goforth, R. Arun Srivatsan, and S. Lucey. PointNetLK: Robust and efficient point cloud registration using pointnet. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2

[2] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE TPAMI*, 14(2):239–256, 1992. 2, 7, 8

[3] D. Campbell and L. Petersson. Gogma: Globally-optimal gaussian mixture alignment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[4] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *IEEE ICRA'91*, April 1991. 2, 7, 8

[5] C. Choy, W. Dong, and V. Koltun. Deep global registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'20)*, 2020. 1, 2, 4, 6, 7, 8

[6] C. Choy, J. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 2

[7] C. Choy, J. Park, and V. Koltun. Fully convolutional geometric features. In *ICCV*, 2019. 1, 2, 4, 5, 6

[8] M. Danelljan, G. Meneghetti, F. Shahbaz Khan, and M. Felsberg. Aligning the dissimilar: A probabilistic method for feature-based point set registration. In *ICPR*, 2016. 2, 6, 7, 8, 12

[9] M. Danelljan, G. Meneghetti, F. Shahbaz Khan, and M. Felsberg. A probabilistic framework for color-based point set registration. In *CVPR*, 2016. 2, 4

[10] H. Deng, T. Birdal, and S. Ilic. Ppfnet: Global context aware local features for robust 3d point matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 195–205, 2018. 2

[11] H. Deng, T. Birdal, and S. Ilic. 3d local features for direct pairwise registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3244–3253, 2019. 2

[12] B. Eckart, K. Kim, and J. Kautz. Hgmr: Hierarchical gaussian mixtures for adaptive 3d registration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 705–721, 2018. 2

[13] G. Evangelidis and R. Horaud. Joint alignment of multiple point sets with batch and incremental expectation-maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1397–1410, June 2018. 2, 3

[14] G. D. Evangelidis, D. Kounades-Bastian, R. Horaud, and E. Z. Psarakis. A generative model for the joint registration of multiple point sets. In *European Conference on Computer Vision*, pages 109–122. Springer, 2014. 1, 2, 3, 4, 6, 7, 8, 10, 12

[15] W. Gao and R. Tedrake. Filterreg: Robust and efficient probabilistic point-set registration using gaussian filter and twist parameterization. In *Conference On Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 6, 7, 8

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 5

[17] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal. Learning multiview 3D point cloud registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'20)*, 2020. 2, 4, 5

[18] B. Jian and B. C. Vemuri. Robust point set registration using gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1633–1645, 2011. 1, 2

[19] Kenta-Tanaka. probreg: Python package for point cloud registration using probabilistic model. 6

[20] F. J. Lawin, M. Danelljan, F. Khan, P.-E. Forssén, and M. Felsberg. Density adaptive point set registration. In *IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, USA, June 2018. Computer Vision Foundation. 2, 3, 6, 7, 8, 10, 12

[21] L. Li, S. Zhu, H. Fu, P. Tan, and C.-L. Tai. End-to-end learning local multi-view descriptors for 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'20)*, 2020. 1, 2

[22] W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan, and S. Song. Deepvcp: An end-to-end deep neural network for point cloud registration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 12–21, 2019. 2

[23] J. P. Luck, C. Q. Little, and W. Hoff. Registration of range data using a hybrid simulated annealing and iterative closest point algorithm. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, ICRA 2000, April 24-28, 2000, San Francisco, CA, USA*, pages 3739–3744, 2000. 2

[24] K. V. Mardia and P. E. Jupp. *Directional Statistics*, chapter 10.3.1. Johna Wiley and Sons Inc., 2000. 4

[25] A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, 2010. 1, 2, 3

[26] F. Pomerlau, F. Colas, and R. Siegwart. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics*, 4(1):1–104, 2015. 2

[27] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016. 2

[28] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. In *IEEE International Conference on Robotics and Automation (ICRA'09)*, Kobe, Japan, May 2009. 2, 6

[29] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR'15*, 2015. 1

[30] A. C. Tavares, F. J. Lawin, and P.-E. Forssén. Assessing losses for point set registration. *IEEE Robotics and Automation Letters*, 5(2):3360–3367, 2020. 2

[31] P. W. Theiler, J. D. Wegner, and K. Schindler. Globally consistent registration of terrestrial laser scans via graph optimization. *ISPRS Journal of Photogrammetry and Remote Sensing*, 109:126–138, 2015. 2

[32] Y. Tsin and T. Kanade. A correlation-based approach to robust point set registration. In *European conference on computer vision*, pages 558–569. Springer, 2004. 2

[33] Y. Wang and J. M. Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3523–3532, 2019. 2

[34] Y. Wang and J. M. Solomon. Prnet: Self-supervised learning for partial-to-partial registration. In *Advances in Neural Information Processing Systems*, pages 8812–8824, 2019. 2

[35] Z. J. Yew and G. H. Lee. 3dfeat-net: Weakly supervised local 3d features for point cloud registration. In *ECCV*, 2018. 2

[36] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017. 5

[37] Q.-Y. Zhou, J. Park, and V. Koltun. Fast global registration. In *European Conference on Computer Vision*, pages 766–782. Springer, 2016. 2, 6, 7, 8

[38] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 6

[39] H. Zhu, B. Guo, K. Zou, Y. Li, K.-V. Yuen, L. Mihaylova, and H. Leung. A review of point set registration: From pairwise registration to groupwise registration. *Sensors*, 19(5), 2019. 1

Supplementary Material

S1. Introduction

This supplementary material contains additional information that did not fit into the main paper. A reference implementation is available at https://github.com/felja633/RLLReg.

S2. Derivation of feature model updates

In this section we derive the update step for the parameters ν_k in (11) in the main paper. The parameters are obtained by solving the following constrained maximization problem,

$$\max_{\nu_k} \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\alpha_{ijk}\nu_k^T y_{ij}\,, \quad \text{s.t. } \|\nu_k\| = 1 \tag{S1}$$

Equivalently, we can set the constraint to $\nu_k^T\nu_k = 1$. We write the resulting Lagrangian as,

$$L(\nu_k, \lambda) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{ij}\alpha_{ijk}\nu_k^T y_{ij} - \lambda\cdot(\nu_k^T\nu_k - 1)\,, \tag{S2}$$

where $\lambda \in \mathbb{R}$ is the Lagrange multiplier. At solutions to (S1), the gradient of (S2) should vanish, i.e.

$$\nabla_{\nu_k} L(\nu_k, \lambda) = \sum_{j=1}^{N} w_{ij}\alpha_{ijk}y_{ij} - 2\lambda\nu_k = 0\,. \tag{S3}$$

We solve this equation for ν_k as,

$$\nu_k = \frac{\sum_{j=1}^{N} w_{ij}\alpha_{ijk}y_{ij}}{2\lambda}\,. \tag{S4}$$

We see that the constraint is satisfied when $\lambda = \left|\left|\sum_{j=1}^{N} w_{ij}\alpha_{ijk}y_{ij}\right|\right|/2$ and we arrive at the expression in (11).

S3. Runtime analysis

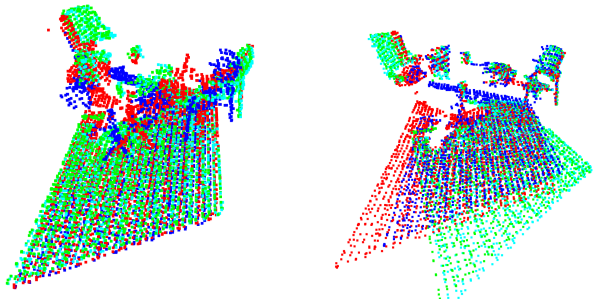
We here provide an empirical runtime analysis of our method for pairwise registration on 3DMatch point sets. The computed values are averages over 95 samples. The main processing steps in our method are the initial voxel downsampling, the feature extraction and the registration step. In our implementation, the voxel downsample step accounts for 45.7% of the total runtime. The subsequent feature extraction steps takes 9.6% of the runtime. Most of the remaining time is due to the registration step at 44.2%. Note that this fraction can be made smaller by employing fewer EM iterations. Another small fraction is caused by overhead operations and memory management.

Timings in seconds are provided in the main paper (Tables 2, and 3).

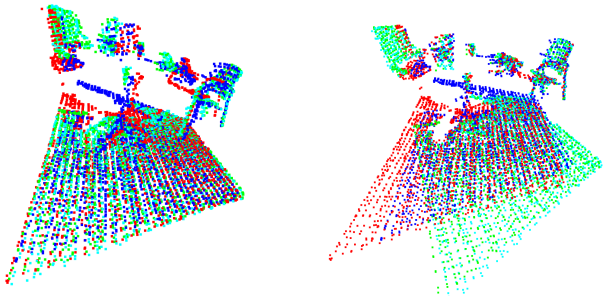
S4. Qualitative examples

We visualize examples of successful registrations produced by our approach on multi-view point sets from 3DMatch (Figure S1) and Kitti (Figure S3). We see that the baseline approaches JRMPC [14] and DARE [20], which are not using the proposed feature model, fail to register these point sets.

We also visualize the corresponding predicted weights in Figures S2 and S4. In both cases, the weighting focuses on specific structures and objects, while reducing the impact of ambiguous flat regions such as floors and ground.

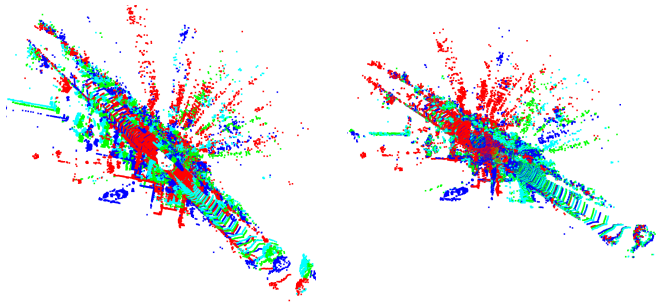


Input Ground-truth

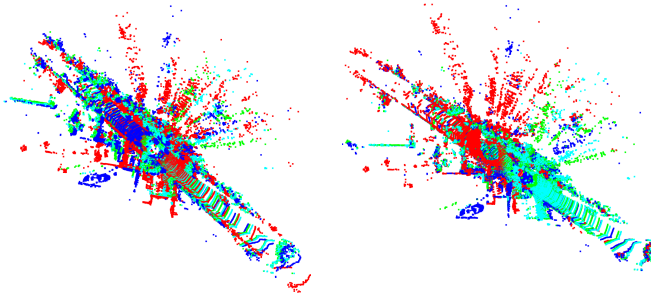


JRMPC Ours RLL Multi

Figure S1. Example of joint registration of four point sets from 3Dmatch.



Input Ground-truth



DARE Ours RLL Multi

Figure S3. Example of joint registration of four point sets from Kitti.

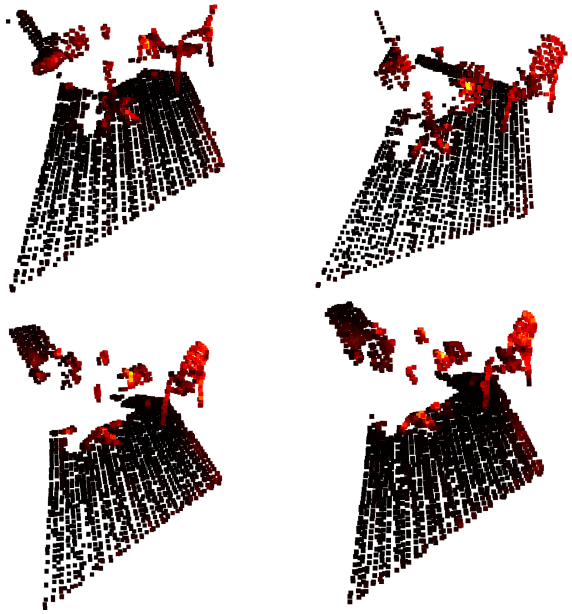


Figure S2. Visualization of the learned point-wise weighting. Brighter colors indicate higher weights.

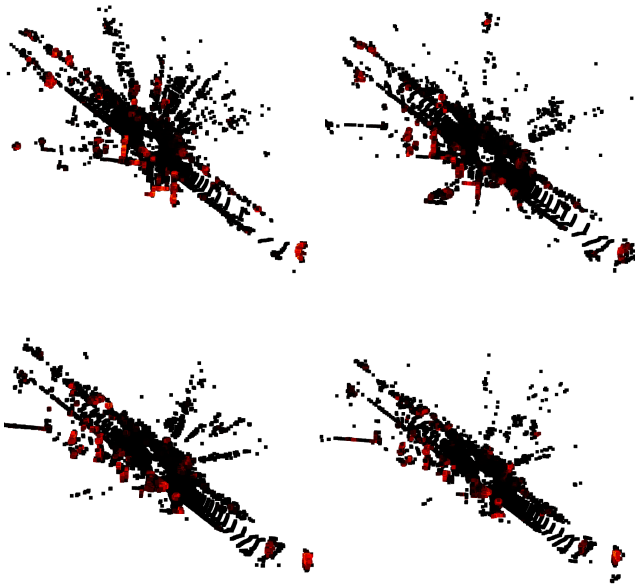


Figure S4. Visualization of the learned point-wise weighting. Brighter colors indicate higher weights.

S5. Experiments

In this section we provide additional results that did not fit into the experiments section of the main paper. To analyze of the impact of the number of mixture components K, we compare the performance of our method (RLL N_iter = 23+weights) with $K = 25, 50, 100, 200, 400$ on the 3D Match test dataset. The results are presented in Figure S5 as recall curves. As we can see, using $K = 400$ components gives the highest recall at smaller error thresholds for both rotation and translation. However, at larger errors using $K = 100$ components gives higher recall.

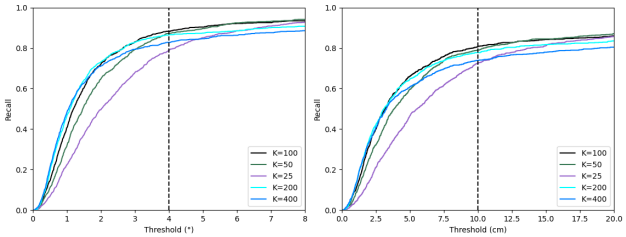


Figure S5. Pairwise recall curves on 3DMatch for errors in rotation (left) and translation (right). The curves show in recall for different number of mixture components K.

method \ 3DMatch	rot. inlier rate %	min rot. inlier rate %	max rot. inlier rate %	t. inlier rate %	min t. inlier rate %	max t. inlier rate %
JRMPC[14]	65.9	96.9	45.2	99.6	98.6	50.5
FPPSR+FCGF[8]	71.8	98.4	52.8	95.1	99.6	55.9
FPPSR+FPFH[8]	66.3	97.5	46.2	94.6	99.0	47.6
Ours Contrastive	80.8	99.0	65.5	99.8	100.0	71.9
Ours RLL	87.5	99.6	75.4	99.9	100.0	77.6
Ours RLL multi	87.2	99.2	75.2	99.8	100.0	78.4

method \ Kitti	rot. inlier rate %	min rot. inlier rate %	max rot. inlier rate %	t. inlier rate %	min t. inlier rate %	max t. inlier rate %
DARE[20]	81.6	98.8	67.1	57.9	85.4	11.1
FPPSR+FCGF[8]	77.8	97.7	61.8	64.2	88.3	15.6
FPPSR+FPFH[8]	64.3	95.3	41.9	49.5	72.7	7.4
Ours Contrastive	0.901	100.0	81.7	76.8	88.7	16.8
Ours RLL	82.5	99.6	70.2	78.1	97.9	43.7
Ours RLL multi	83.5	99.4	71.2	78.7	98.4	43.7

Table S1. Multi-view registration with four point sets in each sample for 3DMatch (top) and Kitti (bottom). Inlier/outlier splits are as in the main paper. The min and max columns only consider the min and max error pair in each sample respectively.

We also extend the results provided for multi-view registration in Table 4 in the paper. In Table S1, we include inlier rates for both rotation and translations[1]. As in the paper, we use thresholds of 4 degrees for the rotations errors and 10 cm for translation on 3D Match and 30 cm for translation on Kitti. In addition we also show maximum and minimum inlier rates, which are the inlier rates for the maximum and minimum errors respectively for each sample.