DeepUME: Learning the Universal Manifold Embedding for Robust Point Cloud Registration - Supplementary Material

Natalie Lang
langn@post.bgu.ac.il
Joseph M.Francos
francos@ee.bgu.ac.il
Ben-Gurion University
Beer-Sheva, Israel

1 Overview

This document contains the following:

- Section 2: Mathematical proofs of the statements mentioned in the main paper
- Section 3: Additional registration qualitative and quantitative results
- Section 4: Several ablation experiments are presented, replacing components of DeepUME by alternatives in order to evaluate the contribution of the proposed construction
- Section 5: DeepUME architecture details

2 Mathematical Proofs

2.1 Discrete UME

Relying on the original UME method, we prove the closed form formula presented in the paper in equation (1).

Theorem 2.1 *Let* $\mathbf{R}$ *be a rotation matrix and* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *be two point clouds satisfying the relation* $\mathcal{P}_2 = \mathbf{R}\cdot\mathcal{P}_1$. *Let* $\mathcal{F}$ *be an* $SO(3)$ *invariant feature on* $\mathcal{P}_1$ *and* $\mathcal{P}_2$, *namely*

$$\mathcal{F}(\mathbf{p}) = \mathcal{F}(\mathbf{R}\mathbf{p}), \quad \forall \mathbf{p} \in \mathcal{P}_1, \tag{1}$$

then

$$\mathbf{M}_{\mathcal{P}_2}(\mathcal{F}) = \mathbf{R}\cdot\mathbf{M}_{\mathcal{P}_1}(\mathcal{F}), \quad \textit{where } \mathbf{M}_{\mathcal{P}_i}(\mathcal{F}) = \frac{1}{|\mathcal{P}_i|}\begin{bmatrix}\sum_{\mathbf{p}\in\mathcal{P}_i} p_1\mathcal{F}(\mathbf{p})\\ \sum_{\mathbf{p}\in\mathcal{P}_i} p_2\mathcal{F}(\mathbf{p})\\ \sum_{\mathbf{p}\in\mathcal{P}_i} p_3\mathcal{F}(\mathbf{p})\end{bmatrix},\ \mathbf{p} = \begin{bmatrix}p_1\\p_2\\p_3\end{bmatrix}. \tag{2}$$

In order to prove Theorem 2.1, we first state and prove a discrete version of the UME theorem from which Theorem 2.1 follows immediately as a special case. We begin by presenting the continuous UME theorem, [4], in the specific case where the translation vector $\mathbf{t} = 0$.

Definition 2.1.1 *Let $k:\mathbb{R}^n \to \mathbb{R}$ be a compactly supported measurable function and $w_1,\dots,w_D : \mathbb{R}\to\mathbb{R}$ are measurable functions. The $n\times D$ UME matrix of k with respect to $w_1,\dots,w_D$ is defined by*

$$[\mathbf{UME}_k]_{i,j} := \int_{\mathbb{R}^n} x_i w_j(k(\mathbf{x}))d\mathbf{x}. \tag{3}$$

Theorem 2.2 ***(Continuous UME)*** *[4], Let $f,g:\mathbb{R}^n\to\mathbb{R}$, be two functions with compact supports related by a rotation $\mathbf{R}$, i.e. $g(\mathbf{x}) = f(\mathbf{Rx})$ for all $\mathbf{x}\in\mathbb{R}^n$. Then, for any set of D measurable functions $w_1,\dots,w_D$ such that $w_i(0)=0$ for all i,*

$$\mathbf{UME}_f = \mathbf{R}\cdot\mathbf{UME}_g\,. \tag{4}$$

We next provide the discrete analog of Theorem 2.2 where the continuous functions f and g are replaced by the invariant functions estimated from the observed point clouds using the DNN, and integration is replaced by summation on the elements in the point clouds.

Definition 2.2.1 *Let $\mathcal{P}\subseteq\mathbb{R}^3$ be a finite point cloud, $\mathcal{F}$ a feature (function) on $\mathcal{P}$ and $w_1,\dots,w_d : \mathbb{R}\to\mathbb{R}$ measurable functions. The discrete $n\times D$ UME matrix of $\mathcal{P}$ and $\mathcal{F}$ with respect to $w_1\dots,w_D$ is defined to be*

$$[\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}}]_{i,j} = \sum_{\mathbf{p}\in\mathcal{P}} p_i w_j(\mathcal{F}(\mathbf{p})). \tag{5}$$

Proposition 2.2.1 *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two point clouds satisfying $\mathcal{P}_2 = \mathbf{R}\mathcal{P}_1$, and $\mathcal{F}$ is an invariant feature on $\mathcal{P}_1$ and $\mathcal{P}_2$. For any D functions $w_1,\dots,w_d:\mathbb{R}\to\mathbb{R}$ satisfying $w_i(0)=0$ for all i*

$$\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_2} = \mathbf{R}\cdot\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_1}\,. \tag{6}$$

We note that if we take w_1 to be the identity function and denote the first column of $\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_1}$ and $\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_2}$ by $[\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_1}]_1$ and $[\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_2}]_1$ respectively, we have

$$\frac{1}{|\mathcal{P}_i|}[\mathbf{UME}^{\mathcal{F}}_{\mathcal{P}_i}]_1 = \mathbf{M}_{\mathcal{P}_i}(\mathcal{F}), \quad i=1,2. \tag{7}$$

Hence, Theorem 2.1 is followed by Proposition 2.2.1 trivially, as a special case.

Proof of Proposition 2.2.1 The main idea in our proof is to approximate the discrete sums defining the discrete UME matrices in (6) by continuous integrals and apply the continuous UME theorem. Given $\varepsilon>0$, $\mathcal{P}_1$, $\mathcal{P}_2$ and the invariant feature $\mathcal{F}$, we construct two compactly supported measurable functions $f_\varepsilon, g_\varepsilon:\mathbb{R}\to\mathbb{R}$ such that $g_\varepsilon(\mathbf{x}) = f_\varepsilon(R^{-1}\mathbf{x})$. We then apply Theorem 2.2 to f_ε and g_ε, and taking $\varepsilon\to 0$ we will conclude.

Denote the ball of radius ε centered at a point $\mathbf{p}\in\mathbb{R}^3$ by $B_\varepsilon(\mathbf{p})$. Define the functions f_ε is and g_ε by

$$f_\varepsilon(\mathbf{x}) := \sum_{\mathbf{p}\in\mathcal{P}_1}\mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}), \quad g_\varepsilon(\mathbf{x}) := \sum_{\mathbf{p}\in\mathcal{P}_2}\mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}). \tag{8}$$

See the illustration of $f_\varepsilon(\mathbf{x})$ in Figure 1. We now prove that the desired relation $g_\varepsilon(\mathbf{x}) = f_\varepsilon(\mathbf{R}^{-1}\mathbf{x})$ holds: Directly from the definition of f_ε,

$$f_\varepsilon(\mathbf{R}^{-1}\mathbf{x}) \overset{(8)}{=} \sum_{\mathbf{p}\in\mathcal{P}_1}\mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{R}^{-1}\mathbf{x}). \tag{9}$$

Since a rigid transformation maps any ball to a ball with the same radius, we have that $\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{R}^{-1}\mathbf{x})$ is non-zero if and only if $\mathbf{R}^{-1}\mathbf{x} \in B_\varepsilon(\mathbf{p})$:

$$\mathbf{R}^{-1}\mathbf{x} \in B_\varepsilon(\mathbf{p}) \iff \mathbf{x} \in \mathbf{R}(B_\varepsilon(\mathbf{p})) \iff \mathbf{x} \in B_\varepsilon(\mathbf{R}\mathbf{p}). \tag{10}$$

It is immediately follows that

$$\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{R}^{-1}\mathbf{x}) = \mathbb{1}_{B_\varepsilon(\mathbf{R}\mathbf{p})}(\mathbf{x}), \quad \forall \mathbf{x} \in \mathbb{R}^3. \tag{11}$$

Substituting (11) into (9), and using the SO(3) invariance of $\mathcal{F}$ we have

$$\begin{aligned} f_\varepsilon(\mathbf{R}^{-1}\mathbf{x}) &= \sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{R}^{-1}\mathbf{x}) & (12)\\ &= \sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{R}\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{R}\mathbf{p})}(\mathbf{x}) & (13)\\ &= \sum_{\mathbf{p}\in\mathcal{P}_2} \mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}) = g_\varepsilon(\mathbf{x}). & (14) \end{aligned}$$

We have so far proved that $f_\varepsilon(\mathbf{x}) = g_\varepsilon(\mathbf{R}\mathbf{x})$ for all $\mathbf{x}$. By Theorem 2.2 we have

$$\mathbf{UME}_{g_\varepsilon} = \mathbf{R}\cdot\mathbf{UME}_{f_\varepsilon}\,. \tag{15}$$

For sufficiently small ε, the balls defining f_ε do not intersect and therefore we have,

$$\begin{aligned} [\mathbf{UME}_{f_\varepsilon}]_{ij} &\overset{(8)}{=} \int_{\mathbb{R}^3} x_i w_j\left(\sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x})\right)d\mathbf{x} & (16)\\ &= \int_{\biguplus_{\mathbf{v}\in\mathcal{P}_1} B_\varepsilon(\mathbf{v})} x_i w_j\left(\sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x})\right)d\mathbf{x} & (17)\\ &\quad+ \int_{\mathbb{R}^3\setminus\biguplus_{\mathbf{v}\in\mathcal{P}_1} B_\varepsilon(\mathbf{v})} x_i \underbrace{w_j\left(\sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{p})\underbrace{\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x})}_{=0}\right)}_{w_j(0)=0}d\mathbf{x} & (18)\\ &= \sum_{\mathbf{v}\in\mathcal{P}_1}\int_{B_\varepsilon(\mathbf{v})} x_i w_j\left(\sum_{\mathbf{p}\in\mathcal{P}_1} \mathcal{F}(\mathbf{p})\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x})\right)d\mathbf{x} & (19)\\ &= \sum_{\mathbf{v}\in\mathcal{P}_1} w_j(\mathcal{F}(\mathbf{v}))\int_{B_\varepsilon(\mathbf{v})} x_i d\mathbf{x}. & (20) \end{aligned}$$

where $\biguplus$ denotes a disjoint union of sets and the last equality stems from the fact that $\mathcal{F}(\mathbf{p})$ is constant on $B_\varepsilon(\mathbf{v})$. By (20) and the integral mean value theorem we have that

$$\begin{aligned} \lim_{\varepsilon\to 0}\frac{1}{\mathrm{Vol}(B_\varepsilon)}[\mathbf{UME}_{f_\varepsilon}]_{ij} &= \lim_{\varepsilon\to 0}\frac{1}{\mathrm{Vol}(B_\varepsilon)}\sum_{\mathbf{p}\in\mathcal{P}_1} w_j(\mathcal{F}(\mathbf{p}))\int_{B_\varepsilon(\mathbf{p})} x_i d\mathbf{x}\\ &= \sum_{\mathbf{p}\in\mathcal{P}_1} w_j(\mathcal{F}(\mathbf{p}))\underbrace{\lim_{\varepsilon\to 0}\frac{1}{\mathrm{Vol}(B_\varepsilon)}\int_{B_\varepsilon(\mathbf{p})} x_i d\mathbf{x}}_{p_i} = \sum_{\mathbf{p}\in\mathcal{P}_1} p_i w_j(\mathcal{F}(\mathbf{p})) \end{aligned} \tag{21}$$

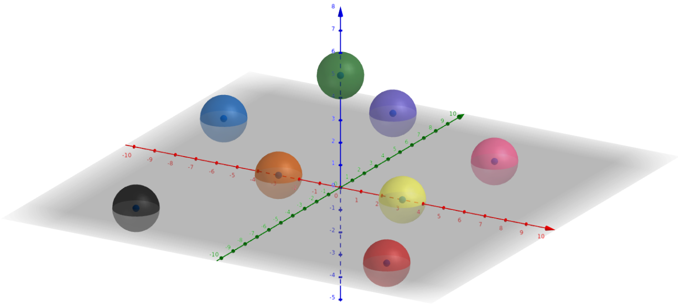


Figure 1: $B_{\varepsilon(\mathbf{p})}$ for $\mathbf{p} \in \mathcal{P}$. Balls centers are the point cloud points, and each ball color represents the value of $\mathcal{F}(\mathbf{p})$.

This shows that

$$\mathbf{UME}_{\mathcal{P}_1}^{\mathcal{F}} = \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon)} \mathbf{UME}_{f_\varepsilon}, \tag{22}$$

and similarly it is easily proved that

$$\mathbf{UME}_{\mathcal{P}_2}^{\mathcal{F}} = \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon)} \mathbf{UME}_{g_\varepsilon}. \tag{23}$$

Finally, applying Theorem 2.2 on f_ε and g_ε we obtain:

$$\mathbf{UME}_{\mathcal{P}_2}^{\mathcal{F}} = \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon)} \mathbf{UME}_{g_\varepsilon} = \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon)} \mathbf{R} \cdot \mathbf{UME}_{f_\varepsilon} \tag{24}$$

$$= \mathbf{R} \cdot \lim_{\varepsilon \to 0} \frac{1}{\mathrm{Vol}(B_\varepsilon)} \mathbf{UME}_{f_\varepsilon} = R \cdot \mathbf{UME}_{\mathcal{P}_1}^{\mathcal{F}}. \tag{25}$$

This completes the proof.

2.2 SO(3)-invariant coordinate system

In the proposed method, point cloud raw coordinates are mapped to a transformation invariant representation by projecting them on the coordinate system defined by the principle vectors of the point cloud PCA.

More specifically, given a point cloud $\mathcal{P}$, the cloud center of mass (denoted by $\mathbf{m}_{\mathcal{P}}$) is subtracted from each point coordinates, to obtain a centered representation $\mathcal{P}'$. The axes of the new coordinate system are the principle vectors of the point cloud covariance matrix given by

$$\mathbf{H}_{\mathcal{P}'} = \sum_{\mathbf{p} \in \mathcal{P}'} \mathbf{p}\mathbf{p}^T \tag{26}$$

For a point $\mathbf{p} \in \mathcal{P}'$, the new coordinates of $\mathbf{p}$ are defined to be $\mathbf{c}_\mathbf{p} = \mathbf{D}_{\mathcal{P}'}^T \cdot \mathbf{p}$ where $\mathbf{D}_{\mathcal{P}'}$ is a matrix whose columns are principle vectors. Formally, $\mathbf{D}_{\mathcal{P}'}$ is an orthogonal matrix for which $\mathbf{H}_{\mathcal{P}'} = \mathbf{D}_{\mathcal{P}'} \mathbf{\Lambda} \mathbf{D}_{\mathcal{P}'}^T$ for a diagonal matrix $\mathbf{\Lambda}$. The resulting point cloud new coordinates are denoted by $\mathcal{C}$.

We shall now verify that the new axes (columns of the PCA matrix) are rotation covariant:

$$\mathbf{H}_{\mathbf{R}\mathcal{P}'} = \sum_{\mathbf{p} \in \mathbf{R}\mathcal{P}'} \mathbf{R}\mathbf{p}\mathbf{p}^T\mathbf{R}^T = \mathbf{R}\mathbf{H}_{\mathcal{P}'}\mathbf{R}^T = \mathbf{R}\mathbf{D}_{\mathcal{P}'}\mathbf{\Lambda}(\mathbf{R}\mathbf{D}_{\mathcal{P}'})^T. \tag{27}$$

That is,

$$\mathbf{D}^{\mathbf{R}\mathcal{P}'} = \mathbf{R}\mathbf{D}_{\mathcal{P}'}. \tag{28}$$

Using (28), we easily prove that the projection coefficients on the new axes are rotation invariant. Given $\mathcal{P}_1$ and $\mathcal{P}_2$ related by a rigid motion, we have

$$\begin{aligned}\mathcal{C}_1 &= \left\{ \left(\mathbf{D}_{\mathcal{P}_1'}\right)^T \mathbf{p} : \mathbf{p} \in \mathcal{P}_1 \right\} = \left\{ \left(\mathbf{R}^T \mathbf{D}_{\mathcal{P}_2'}\right)^T \mathbf{p} : \mathbf{p} \in \mathcal{P}_1 \right\} \\ &= \left\{ \left(\mathbf{D}_{\mathcal{P}_2'}\right)^T \mathbf{R}\mathbf{p} : \mathbf{p} \in \mathcal{P}_1 \right\} = \left\{ \left(\mathbf{D}_{\mathcal{P}_2'}\right)^T \mathbf{p} : \mathbf{p} \in \mathcal{P}_2 \right\} = \mathcal{C}_2.\end{aligned} \tag{29}$$

2.2.1 Axes sign ambiguity

For a point cloud $\mathcal{P}$, the principal vectors defining $\mathbf{D}_{\mathcal{P}'}$, are defined up to a sign. Hence, the equality in (28) is true up to multiplication of the the columns by ± 1. That is to say, only one from the 8 possibilities for the principal vectors matrix satisfies the desired equality (28). We eliminate this sign ambiguity by considering all 8 possible new axes systems $\left\{ D^i_{\mathcal{P}_2'} \right\}_{i=1}^{8}$ given by different sign multiplication constellations. We choose the one axes system that satisfies (29) by

$$i = \operatorname*{arg\,min}_{j=1,\ldots,8} d_{\mathrm{C}}(\mathcal{C}_1, \mathcal{C}_2^j) \tag{30}$$

where d_{C} stands for Chamfer distance.

3 Additional Registration Results

3.1 The Effect of Sampling-Rate on Registration under Sampling-Noise Scenarios

Figure 2 depicts the registration performance of different methods under different point densities.

As mentioned in the introduction of the main paper, many point cloud registration applications process under-sampled point clouds. Figure 2 shows that on dense point clouds, the evaluated registration methods achieve comparable results. However, as the sampling rate decreases, the sampling noise effect becomes dominant and the performance of all methods, but DeepUME, severely deteriorates.

We note that for the closed form registration methods considered, where computational requirements are moderate, registration is testable with up to 80,000 sample points per cloud. In that scenario, PCA and UME registration achieve RMSE($\mathbf{R}$) errors of 5.147 and 2.573 respectively. This is particularly interesting since it shows that PCA and UME, which are both critical parts in our framework, require about *160 times more samples than DeepUME* (80k vs 500) in order to achieve similar performances. It is further implied that the main effect of the proposed method is equivalent to the interpolation of sub-sampled point clouds where the quality criterion of the interpolation is its efficiency for registration purposes.

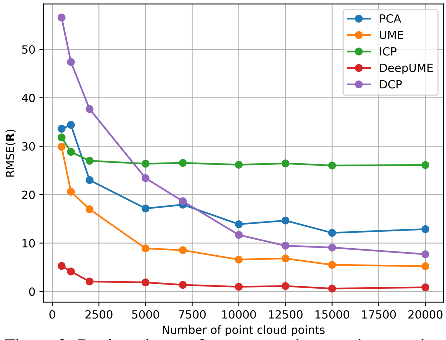


Figure 2: Registration performance under zero-intersection noise model and different point densities on the FAUST dataset. On dense point clouds, the compared registration methods achieve comparable results. However, as the sampling rate decreases, the sampling noise effect becomes dominant and the performance of all methods, but DeepUME, severely deteriorates.

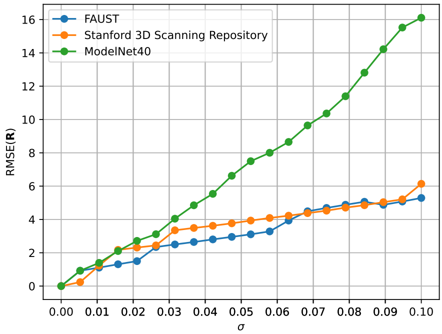


Figure 3: The effect of the WAGN variance on DeepUME performance (measured in RMSE($\mathbf{R}$)) in all three data sets tested. In the presence of noise, more symmetric examples in ModelNet40 data set are incorrectly registered and therefore the average error increases dramatically.

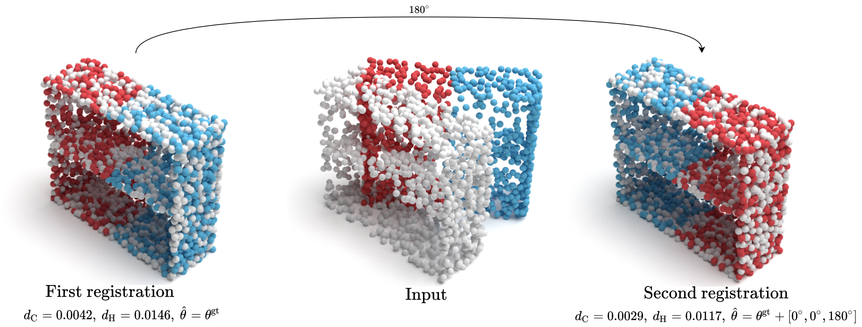


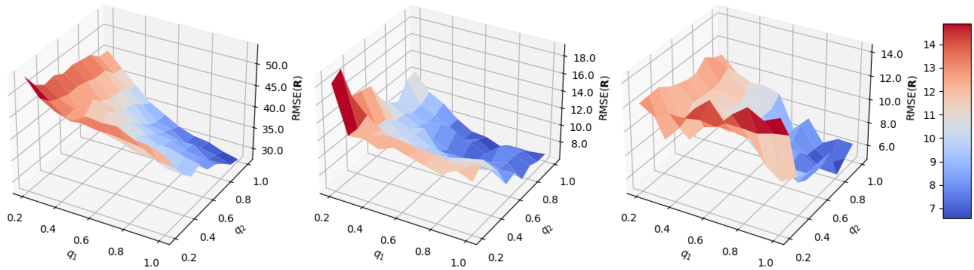
Figure 4: Symmetric objects registration ambiguity arising in sampling noise scenarios. Due to the sampling noise there is no one true registration solution for the input, but any that aligns the clouds together. Therefore, the two presented solutions should result in a low error metric, which is reflected in the Chamfer and Hausdorff distances and not in RMSE($\mathbf{R}$).

3.2 Symmetric objects registration ambiguity

In Figure 4, we demonstrate the ambiguity problem discussed in section 5 in the paper. The box shaped bookshelf (as many other examples in ModelNet40) is symmetric under rotations of 180° about the z axes. Therefore, as shown the in Figure 4, in the sampling noise scenario (zero-intersection model) two possible registration solutions (that differ by a 180° rotation about the z axis) are possible. Both solutions do align the point clouds, yet one achieves zero RMSE($\mathbf{R}$) error, while the other yields a much higher one. Nevertheless, both Chamfer and Hausdorff distances achieve small errors which implies they are better suited for measuring registration error on symmetric shapes.

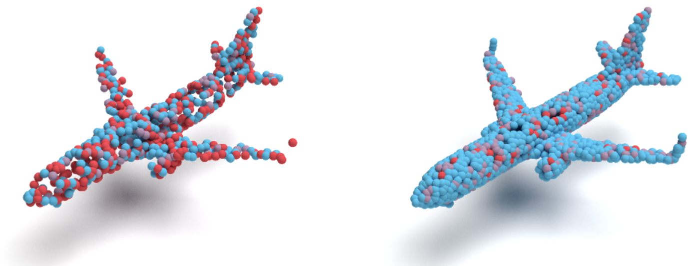
3.3 Gaussian Noise

One of the most intriguing observations presented in the experimental section considers the effect of different types of noise on registration, and in particular, Additive White Gaussian Noise (AWGN) versus sampling noise. In Figure, 3 we evaluate the effect of the noise variance on the rotation RMSE. An interesting observation is that registration on ModelNet40 dataset [] is significantly more affected by AWGN, than other types of data. Recalling the ambiguity problem of ModelNet40 (see section 3.2 and section 5 in the main paper) this



ModelNet40 FAUST Stanford 3D Scanning Repository

Figure 5: DeepUME performance (measured in RMSE(**R**)) under Bernoulli noise model with respect to the probabilities q_1 and q_2 in all three datasets tested. The surfaces visualization implies that the sparseness of the sparsest point clouds is the dominant cause of error, rathe



Bernoulli noise model $q_1 = 0.2, q_2 = 0.2$ Bernoulli noise model $q_1 = 0.8, q_2 = 0.2$

Figure 6: The effect of Bernoulli noise on a point cloud in extreme values of the probabilities q_1 and q_2. Red points denote $\mathcal{P}_1$, blue points denote $\mathcal{P}_2$ and purple points belong to both.

is quite expected: Noise makes the inherent ambiguity problem harder to resolve. In the presence of noise, many of the ambiguous examples are falsely registered and therefore the average error increases dramatically.

3.4 Bernoulli Noise

The error surfaces depicted in Figure 5 describe the rotation RMSE with respect to the probabilities q_1 and q_2 in all three datasets tested. As expected, the error increases as q_1 and q_2 decrease and the point clouds are made sparser. Note that in Figure 5, in all the datasets considered, the rotation RMSE becomes large even when only one of the clouds is sparse (when the probability for keeping a point is small on one cloud and large on the other cloud). That is, the rotation RMSE does not increase much when we take the second cloud to be sparse as well. This might suggest that the sparseness of the sparsest point cloud is the dominant cause of error, rather then the difference between the clouds densities.

For a visualization of the effect of Bernoulli noise on a point cloud in extreme values of the probabilities q_1 and q_2 we refer the reader to Figure 6.

3.5 Small rotation angles range

In our experiments, all DGCNN-based networks, except ours, provide poor registration. This is expected, as without a proper pre-processing procedure such networks fail to create features that are invariant under large rotations. The experimental results demonstrate that on the average, when large rotations are allowed, the registration results of the proposed method outperform the alternatives. However, for a more comprehensive overview, we also compare

	Noise free				Zero-intersection noise			
Model	d_C	d_H	**RMSE**(R)	**RMSE**(t)	d_C	d_H	**RMSE**(R)	**RMSE**(t)
UME [4]	<9e-05	<4e-04	0.333	<ins><5e-06</ins>	0.030	0.269	42.559	<ins>0.010</ins>
ICP [2]	0.280	1.274	72.265	0.277	0.267	1.256	71.532	0.276
PointNetLK [1]	<7e-04	0.005	0.729	0.005	<ins>0.012</ins>	<ins>0.113</ins>	11.504	0.095
DCP [10]	<4e-04	0.003	5.739	0.002	0.065	0.479	71.307	0.011
DeepGMR [12]	<ins><4e-06</ins>	<ins><2e-05</ins>	0.110	<6e-05	0.294	0.270	33.128	0.015
RGM [5]	0.268	1.203	<ins>0.025</ins>	<2e-04	0.267	1.187	**4.690**	0.032
DeepUME (ours)	**<1e-07**	**<1e-07**	**<8e-05**	**<1e-07**	**0.002**	**0.118**	<ins>5.159</ins>	**0.010**

Table 1: Free and zero-intersection noise models results on the unseen dataset Stanford 3D Scanning Repository in small rotation angles regime of $[0,60]$ degrees about each axes. Our method outperforms the competing techniques in the examined scenarios for all metrics, only except the RMSE($\mathbf{R}$) in the zero-intersection noise model. These results suggest that our framework improves the state-of-the-art methods not only on the average over the entire range, but also in the small angles scenario, where most of the methods were designed to be optimal.

our performance separately for the case of registration under rotation by small angles (up to 60° about each axes).

For a fair comparison, we use the pre-trained models released by the authors and test all the proposed methods with rotation in the range $[0,60]$ degrees about each of the axes, for the noise free and zero-intersection noise models, in a similar manner to the experiments described in the main paper. We note that in the small rotations experiment, none of the tested methods encounters the ambiguity problem of ModelNet40, discussed in 3.2 and in section 5 in the main paper. The symmetry in the examples of ModelNet40 creates ambiguity where rotations by angles larger then 90° are considered. Since our method is designed for arbitrary rotations, we do suffer from the ambiguity problem in ModelNet40. Hence, for generating a reliable comparison, we perform that experiment on the Stanford 3D Scanning Repository [8], which does not contain ambiguous examples.

From the results summarized in Table 1, we conclude that the method proposed in this paper outperforms all compared methods in all metrics examined (except for one case, where RGM method [5] achieves slightly smaller rotation RMSE). This shows that the proposed framework outperforms state-of-the-art methods not only on the average on the entire transformation range, but also for the small angles scenario.

4 Ablation Study

We conduct several ablation studies, removing components of the proposed DeepUME and replacing each part with an alternative, to better evaluate our design. The studies were tested for the Gaussian noise and zero-intersection noise models on the unseen FAUST [3] dataset, in a similar manner to the experiments described in the main paper. The results of this section are summarized in Table 2.

4.1 With or without invariant coordinates?

We first try to evaluate whether the new transformation-invariant coordinates generated for the point cloud at the pre-processing phase, provide value over the original representation of the point cloud. We therefore remove the pre-processig module (presented in Figure 2 in the paper) and compare the resulting performance to that obtained using the full model. Table 2 demonstrates that DeepUME performs consistently better with the inclusion of the pre-processig module.

Model	Zero-intersection noise				Gaussian noise			
	d_{C}	d_{H}	**RMSE**(R)	**RMSE**(t)	d_{C}	d_{H}	**RMSE**(R)	**RMSE**(t)
No pre-processing	0.094	0.913	83.506	0.140	0.087	0.896	82.260	0.140
Features joint-sampling	0.007	0.082	14.913	0.027	0.001	0.012	1.533	0.002
MLP instead of UME	0.003	0.046	12.964	0.023	0.001	0.011	1.488	0.003
DeepUME (full model)	**0.002**	**0.024**	**8.630**	**0.019**	**0.001**	**0.011**	**1.069**	**0.002**

Table 2: Ablation study results on the unseen dataset FAUST [3]. DeepUME full model achieves equal or better registration results in all tested scenarios and in all metrics.

4.2 Coordinates joint-resampling or features joint-resampling?

In our proposed framework, the Transformer layer executes a joint-resampling procedure in the coordinated space of $\mathbb{R}^3$. This is unlike other networks, where the joint-resampling process is executed in feature space. Applying the Transformer in the coordinate space has a notable computational advantage as in this case, the embedding size of each point is significantly lower. In our our network the embedding size of each point on the coordinate space is 3 while in the feature space it is 512. Hence, performing a coordinate sampling allows for a dramatic decrease in computational complexity, both in the train and evaluation processes (twice faster). We compare our framework with the two strategies - resampling in coordinate space and resampling in features space. The results presented in Table 2 show that the decrease in computational complexity does not cause a decrease in registration accuracy.

4.3 UME or MLP?

While MLP (Multi Layer Perceptron) provides, in principle, a universal approximation, the UME integration into a DNN framework is designed to provide an accurate computation of a rigid motion under noisy sampling of point clouds. A natural question to ask is whether the UME parameter extraction may be replaced by a general learned module, such that registration with comparable accuracy is achieved. As expected, Table 2 shows that the model performs better with the UME layer than a general MLP.

5 Implementation Details

The architecture of DeepUME is shown in Figure 2 in the paper, and includes the Transformer and DGCNN learned modules. The overall architecture of the Transformer [9] as used in this work, is depicted in Figure 7.

5.1 Point clouds joint-resampling performed by a Transformer in the projected coordinate space

Loosely speaking, the action of the Transformer is meant to improve performances in learned methods for various tasks, by jointly creating weighted sums that modify the original input. Formally, take $\mathcal{C}_1$ and $\mathcal{C}_2$ to be the point clouds generated by the module in 2.2; these representations are computed independently of one another. Our attention model learns a function

$$\phi : \mathbb{R}^{N\times 3} \times \mathbb{R}^{M\times 3} \to \mathbb{R}^{N\times 3} \tag{31}$$

that provides new coordinates for the point clouds

$$\mathcal{C}_1^{\text{Transformer}} = \mathcal{C}_1 + \phi(\mathcal{C}_1, \mathcal{C}_2), \quad \mathcal{C}_2^{\text{Transformer}} = \mathcal{C}_2 + \phi(\mathcal{C}_2, \mathcal{C}_1). \tag{32}$$

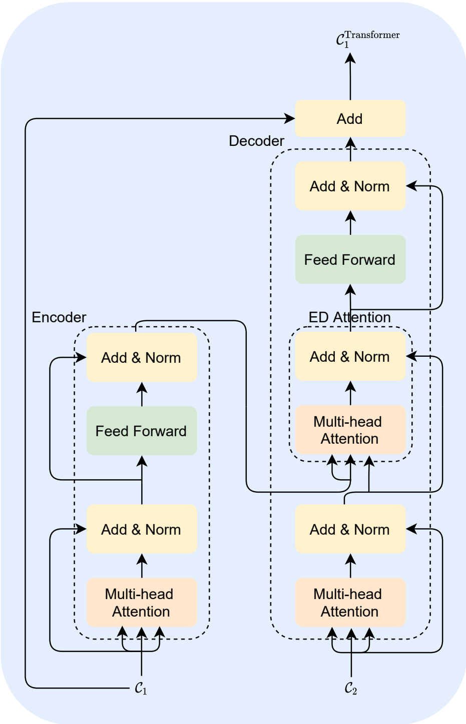


Figure 7: Transformer network architecture. ED stands for Encoder-Decoder.

The objective of the map $\mathcal{C}_1 \mapsto \mathcal{C}_1^{\text{Transformer}}$ is to modify the features associated with the points in $\mathcal{C}_1$ in a way that is knowledgeable of the structure of $\mathcal{C}_2$; the map $\mathcal{C}_2 \mapsto \mathcal{C}_2^{\text{Transformer}}$ serves a symmetric role. The asymmetric function ϕ is given by a Transformer.

During the learning process, the additive terms, $\phi(\mathcal{C}_1, \mathcal{C}_2)$ and $\phi(\mathcal{C}_2, \mathcal{C}_1)$, change the point clouds, without distorting the shape. We therefore refer to this process as resampling. A method for evaluating the similarity between the shapes of the Transformer input and output is to compute their Chamfer distance. We average the Chamfer distance between $\mathcal{C}_i$ and $\mathcal{C}_i^{\text{Transformer}}$ over the ModelNet40 dataset, and get a distance of about 0.004 (where all point clouds are scaled to the unit sphere). This shows that a different point cloud is obtained after the Transformer's action, however the small Chamfer distance between the input and output point clouds (relatively to the point clouds size) shows that this difference is due to a change of sampling points rather than a change of shape.

5.2 Optimization details

Adam [6] is used to optimize the network parameters, with an initial learning rate of 0.001. We divide the learning rate by 10 at epochs $75, 150$ and 200, training for a total of 250 epochs. As the modules of the pre-processing and the UME are of closed-form and non iterative or brute force, their impact on the computational time is negligible. DeepUME is implemented in Pyotrch [7], and its training times is about 11 hours long using an NVIDIA Quadro RTX6000 GPU.

References

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7163–7172, 2019.

[2] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[3] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[4] Amit Efraim and Joseph M Francos. The universal manifold embedding for estimating rigid transformations of point clouds. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5157–5161. IEEE, 2019.

[5] Kexue Fu, Shaolei Liu, Xiaoyuan Luo, and Manning Wang. Robust point cloud registration framework based on deep graph matching. *Internaltional Conference on Computer Vision and Pattern Recogintion (CVPR)*, 2021.

[6] Diederik P Kingma. Ba j.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 531, 2014.

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learn pdf.

[8] Stanford Scanning Repository. The Stanford 3D Scanning Repository. http://graphics.stanford.edu/data/3Dscanrep/, 1994.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[10] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3523–3532, 2019.

[11] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1912–1920, 2015.

[12] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. Deepgmr: Learning latent gaussian mixture models for registration. In *European Conference on Computer Vision*, pages 733–750. Springer, 2020.