

# 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 visualizations and quantitative results
- Section 4: Several ablation experiments are presented, replacing components of Deep-UME by alternatives in order to evaluate the contribution of the proposed construction
- Section 5: Architecture implementation 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*

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

then

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

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, [8], in the specific case where the translation vector  $\mathbf{t} = 0$ .

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

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

**Theorem 2.2 (Continuous UME)** [1], Let  $f, g : \mathbb{R}^n \rightarrow \mathbb{R}$ , be two functions with compact supports related by a rotation  $\mathbf{R}$ , i.e.  $g(\mathbf{x}) = f(\mathbf{R}\mathbf{x})$  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  $i$ ,

$$\mathbf{UME}_f = \mathbf{R} \cdot \mathbf{UME}_g. \quad (4)$$

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  $w_1, \dots, w_D : \mathbb{R} \rightarrow \mathbb{R}$  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

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

**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  $w_1, \dots, w_D : \mathbb{R} \rightarrow \mathbb{R}$  satisfying  $w_i(0) = 0$  for all  $i$

$$\mathbf{UME}_{\mathcal{P}_2}^{\mathcal{F}} = \mathbf{R} \cdot \mathbf{UME}_{\mathcal{P}_1}^{\mathcal{F}}. \quad (6)$$

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

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

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  $f_\varepsilon, g_\varepsilon : \mathbb{R} \rightarrow \mathbb{R}$  such that  $g_\varepsilon(\mathbf{x}) = f_\varepsilon(\mathbf{R}^{-1}\mathbf{x})$ . We then apply Theorem 2.2 to  $f_\varepsilon$  and  $g_\varepsilon$ , and taking  $\varepsilon \rightarrow 0$  we will conclude.

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

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

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  $f_\varepsilon$ ,

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

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  $\mathbf{R}^{-1}\mathbf{x} \in B_\varepsilon(\mathbf{p})$ :

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

It immediately follows that

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

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

$$f_\varepsilon(\mathbf{R}^{-1}\mathbf{x}) = \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{p}) \mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{R}^{-1}\mathbf{x}) \quad (12)$$

$$= \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{Rp}) \mathbb{1}_{B_\varepsilon(\mathbf{Rp})}(\mathbf{x}) \quad (13)$$

$$= \sum_{\mathbf{p} \in \mathcal{P}_2} \mathcal{F}(\mathbf{p}) \mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}) = g_\varepsilon(\mathbf{x}). \quad (14)$$

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

$$\text{UME}_{g_\varepsilon} = \mathbf{R} \cdot \text{UME}_{f_\varepsilon}. \quad (15)$$

For sufficiently small  $\varepsilon$ , the balls defining  $f_\varepsilon$  do not intersect and therefore we have,

$$[\text{UME}_{f_\varepsilon}]_{ij} \stackrel{(8)}{=} \int_{\mathbb{R}^3} x_i w_j \left( \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{p}) \mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}) \right) d\mathbf{x} \quad (16)$$

$$= \int_{\bigsqcup_{\mathbf{v} \in \mathcal{P}_1} B_\varepsilon(\mathbf{v})} x_i w_j \left( \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{p}) \mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}) \right) d\mathbf{x} \quad (17)$$

$$+ \underbrace{\int_{\mathbb{R}^3 \setminus \bigsqcup_{\mathbf{v} \in \mathcal{P}_1} B_\varepsilon(\mathbf{v})} x_i w_j \left( \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{p}) \underbrace{\mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x})}_{=0} \right) d\mathbf{x}}_{w_j(0)=0} \quad (18)$$

$$= \sum_{\mathbf{v} \in \mathcal{P}_1} \int_{B_\varepsilon(\mathbf{v})} x_i w_j \left( \sum_{\mathbf{p} \in \mathcal{P}_1} \mathcal{F}(\mathbf{p}) \mathbb{1}_{B_\varepsilon(\mathbf{p})}(\mathbf{x}) \right) d\mathbf{x} \quad (19)$$

$$= \sum_{\mathbf{v} \in \mathcal{P}_1} w_j(\mathcal{F}(\mathbf{v})) \int_{B_\varepsilon(\mathbf{v})} x_i d\mathbf{x}. \quad (20)$$

where  $\bigsqcup$  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

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

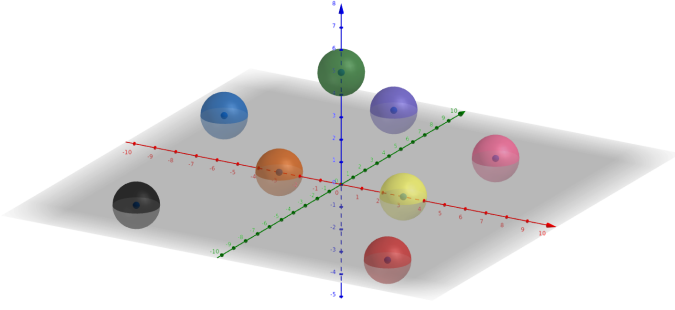


Figure 1:  $B_{\varepsilon}(\mathbf{p})$  for  $\mathbf{p} \in \mathcal{P}$ . Balls centers are the point cloud points, and each ball color represents the value of  $\mathcal{F}(\mathbf{p})$ .

This shows that

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

and similarly it is easily proved that

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

Finally, applying Theorem 2.2 on  $f_{\varepsilon}$  and  $g_{\varepsilon}$  we obtain:

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

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

This completes the proof.

## 2.2 $\text{SO}(3)$ -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

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

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 co-variant:

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

That is,

$$\mathbf{D}^{\mathbf{R}\mathcal{P}'} = \mathbf{R}\mathbf{D}_{\mathcal{P}'}. \quad (28)$$

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

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

### 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  $\pm 1$ . 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  $\left\{ \mathbf{D}_{\mathcal{P}'_2}^i \right\}_{i=1}^8$  given by different sign multiplication constellations. We choose the one axes system that satisfies (29) by

$$i = \arg \min_{j=1, \dots, 8} d_{\mathcal{C}}(\mathcal{C}_1, \mathcal{C}_2^j) \quad (30)$$

where  $d_{\mathcal{C}}$  stands for Chamfer distance.

## 3 Additional Registration Results

### 3.1 ModelNet40 ambiguity

In Figure 2, 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  $180^\circ$  about the  $z$  axes. Therefore, as shown the in Figure 2, in the sampling noise scenario (zero-intersection model) two possible registration solutions (obtained from one to another by applying a  $180^\circ$  rotation about the  $z$  axes) 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.

### 3.2 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 [4] is significantly more affected by AWGN, than other types of data. Recalling the ambiguity problem of ModelNet40 (see section 3.1 and section 5 in the main paper) this is quite expected: Noise makes the inherent ambiguity problem harder to resolve. In the

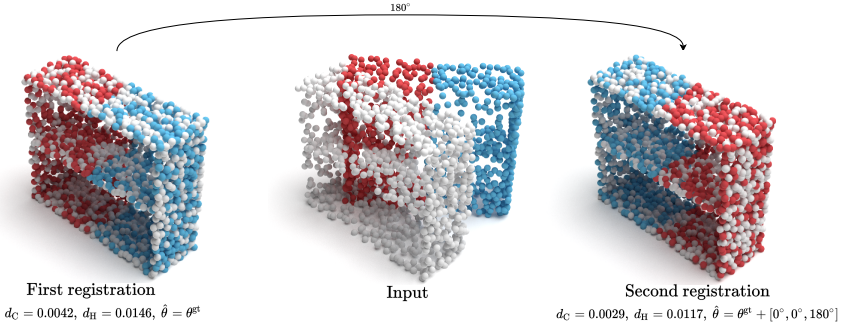


Figure 2: ModelNet40 registration ambiguity arising in sampling noise scenarios. Due to the sampling noise there is no one true registration solution for the input, but any that aligns the clouds together. Therefore, the two presented solutions should result in a low error metric, which is reflected in the Chamfer and Hausdorff distances and not in RMSE(R).

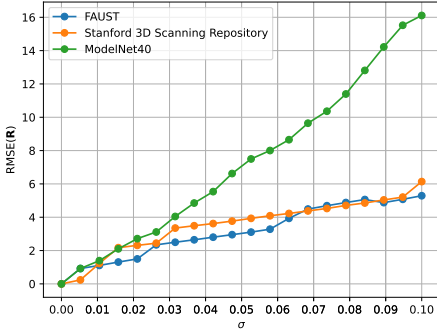


Figure 3: The effect of the WAGN variance on DeepUME performance (measured in  $\text{RMSE}(\mathbf{R})$ ) in all three datasets tested. In the presence of noise, more symmetric examples in ModelNet40 dataset are incorrectly registered and therefore the average error increases dramatically.

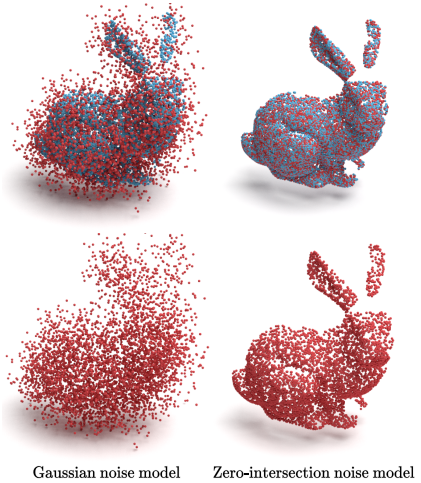


Figure 4: In the top row  $\mathcal{P}_1$  and  $\mathcal{P}_2$  are presented together, and in the bottom row only  $\mathcal{P}_2$  is shown. In order to cause the same registration error as in the zero-intersection noise model on the Stanford dataset, an AWGN with variance of approximately 0.09 is required. While the former preserves well the original shape, the latter severe distortion makes it barely recognizable.

presence of noise, many of the ambiguous examples are falsely registered and therefore the average error increases dramatically.

We find that for sufficiently large noise variance, the rotation RMSE in the AWGN case and the rotation RMSE for both sampling noise scenarios discussed in the main paper, are comparable. In order to illustrate the very different impact of the different types of noise on registration, we refer the reader to Figure 4. In order to have the same registration error as in the zero-intersection model on the Stanford dataset, an AWGN with variance of approximately 0.09 is required. As shown in Figure 4, such a noise causes a severe distortion, which practically makes the original shape unrecognizable. On the other hand, the original shape of the bunny in the sampling noise scenario is well preserved.

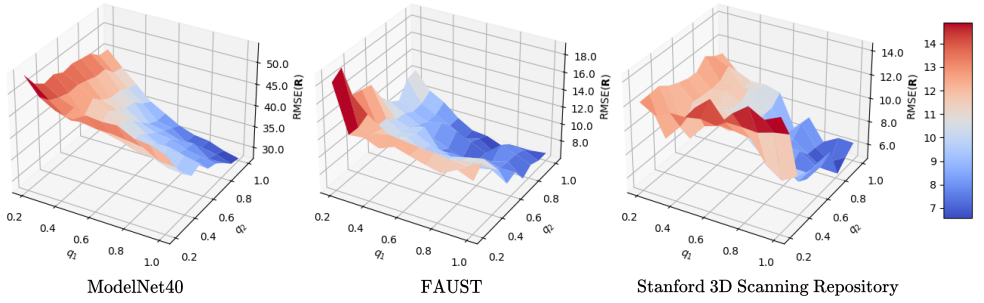


Figure 5: DeepUME performance (measured in  $\text{RMSE}(\mathbf{R})$ ) under Bernoulli noise model with respect to the probabilities  $q_1$  and  $q_2$  in all three datasets tested. The surfaces visualization implies that the sparseness of the sparsest point clouds is the dominant cause of error, rather than the difference between the clouds densities.

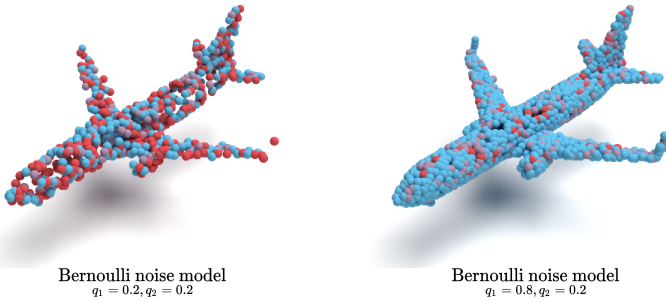


Figure 6: The effect of Bernoulli noise on a point cloud in extreme values of the probabilities  $q_1$  and  $q_2$ . Red points denote  $\mathcal{P}_1$ , blue points denote  $\mathcal{P}_2$  and purple points belong to both.

### 3.3 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 than 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.

### 3.4 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 our performance separately for the case of registration under rotation by small angles (up to  $60^\circ$  about each axes).

For a fair comparison, we use the pre-trained models released by the authors and test

Model	Noise free				Zero-intersection noise			
	$d_C$	$d_H$	RMSE(R)	RMSE(t)	$d_C$	$d_H$	RMSE(R)	RMSE(t)
UME [9]	<9e-05	<4e-04	0.333	<5e-06	0.030	0.269	42.559	0.010
ICP [8]	0.280	1.274	72.265	0.277	0.267	1.256	71.532	0.276
PointNetLK [10]	<7e-04	0.005	0.729	0.005	0.012	0.113	11.504	0.095
DCP [11]	<4e-04	0.003	5.739	0.002	0.065	0.479	71.307	0.011
DeepGMR [12]	<4e-06	<2e-05	0.110	<6e-05	0.294	0.270	33.128	0.015
RGM [6]	0.268	1.203	0.025	<2e-04	0.267	1.187	<b>4.690</b>	0.032
DeepUME (ours)	<b>&lt;1e-07</b>	<b>&lt;1e-07</b>	<b>&lt;8e-05</b>	<b>&lt;1e-07</b>	<b>0.002</b>	<b>0.118</b>	<u>5.159</u>	<b>0.010</b>

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(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.

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 3.1 and in section 5 in the main paper. The symmetry in the examples of ModelNet40 creates ambiguity where rotations by angles larger than  $90^\circ$  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 [13], 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 [6] 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 [4] 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	<u>0.003</u>	<u>0.046</u>	<u>12.964</u>	<u>0.023</u>	<u>0.001</u>	<u>0.011</u>	<u>1.488</u>	<u>0.003</u>
DeepUME (full model)	<b>0.002</b>	<b>0.024</b>	<b>8.630</b>	<b>0.019</b>	<b>0.001</b>	<b>0.011</b>	<b>1.069</b>	<b>0.002</b>

Table 2: Ablation study results on the unseen dataset FAUST [4]. 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 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. Similarly to DCP [14], we use 5 *EdgeConv* (denoted as DGCNN [14]) layers. The number of filters in each layer are [64, 64, 128, 256, 512]. The number of heads in multi-head attention is 1 and the embedding dimension is 64. We use LayerNorm [2] without Dropout [3]. Adam [4] is used to optimize the network parameters, with initial learning rate 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. Training on DeepUME takes about 11 hours to converge with Pytorch [8] and 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] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
  - [3] 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.
  - [4] 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.
  - [5] 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.
  - [6] Kexue Fu, Shaolei Liu, Xiaoyuan Luo, and Manning Wang. Robust point cloud registration framework based on deep graph matching. *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
  - [7] Diederik P Kingma. Ba j.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 531, 2014.
  - [8] 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-learning-library.pdf>.
  - [9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
  - [10] Stanford Scanning Repository. The Stanford 3D Scanning Repository. <http://graphics.stanford.edu/data/3Dscanrep/>, 1994.
  - [11] 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.
  - [12] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

- [13] 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.
- [14] 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.