Mahalanobis k-NN: A Statistical Lens for Robust Point-Cloud Registrations

Tejas Anvekar and Shivanand Venkanna Sheshappanavar

Geometric Intelligence Research Lab
University of Wyoming, 1000 E University Ave, Laramie, WY 82071

Abstract. In this paper, we discuss Mahalanobis k-NN: a statistical lens designed to address the challenges of feature matching in learning-based point cloud registration when confronted with an arbitrary density of point clouds, either in the source or target point cloud. We tackle this by adopting Mahalanobis k-NN's inherent property to capture the distribution of the local neighborhood and surficial geometry. Our method can be seamlessly integrated into any local-graph-based point cloud analysis method. In this paper, we focus on two distinct methodologies: Deep Closest Point (DCP) and Deep Universal Manifold Embedding (DeepUME). Our extensive benchmarking on the ModelNet40 and Faust datasets highlights the efficacy of the proposed method in point cloud registration tasks. Moreover, we establish for the first time that the features acquired through point cloud registration inherently can possess discriminative capabilities. This is evident by a substantial improvement of about 20% in the average accuracy observed in the point cloud few-shot classification task benchmarked on ModelNet40 and ScanObjectNN. The code is publicly available at https://github.com/TejasAnvekar/Mahalanobis-k-NN

Keywords: Point Cloud Registration · Metric Distance · 3D Vision.

1 Introduction

With the rapid progress of high-precision sensors like LiDAR [13] and Kinect [47], point clouds have become the prevalent data format for 3D representation. Due to sensor limitations in capturing only partial views, registration algorithms are crucial for amalgamating scans into comprehensive 3D scenes. Point cloud registration involves determining transformation matrices between point cloud pairs, facilitating the fusion of partial scans to produce coherent 3D representations.

Registration is a fundamental task spanning various computational fields such as medical imaging [9], robotics [30], autonomous driving [12], and computational chemistry [24]. Estimating transformation matrices serves applications like 3D reconstruction [21] and precise 3D localization [20]. It underpins the development of intricate 3D maps for autonomous driving [12], environment reconstruction in robotics [30], and improved safety in mining. Additionally, registration's

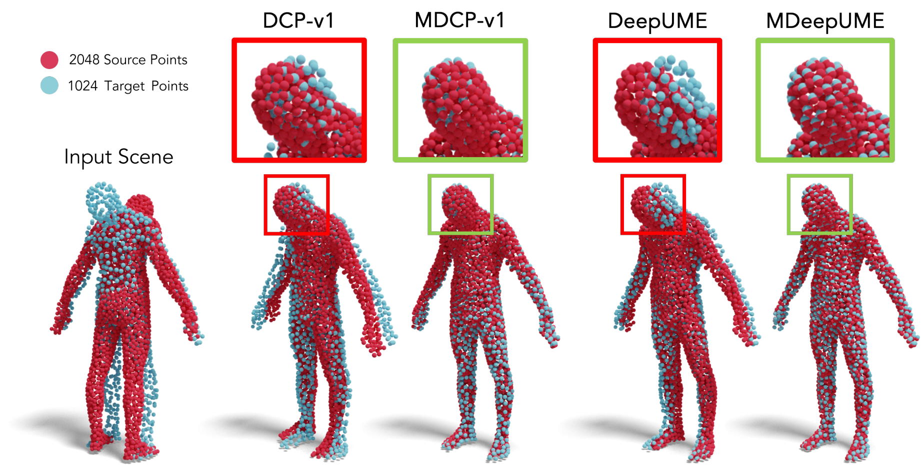

Fig. 1. The visual supremacy of our proposed methodologies, MDCP-v1 and MDeepUME, becomes apparent in a point cloud registration task involving a target point cloud with only half the points compared to the source point cloud. In the visualization, highlighted boxes in red illustrate the limited performance of DCP-v1 [37] and DeepUME [26], while the highlighted boxes in green demonstrate the resilience of the proposed Mahalanobis versions of DCP-v1 [37] and DeepUME [26].

ability to facilitate high-precision localization proves invaluable for entities like driverless cars [12], ensuring precise positioning and interaction within the 3D environment.

The registration of point clouds has garnered significant attention through closed-form [4, 41, 48, 34, 15] and learning-based [2, 37, 45, 26, 18] methodologies. DCP [37] tackles the challenge of feature matching (correspondences) by initially estimating local features via weight-shared DGCNN [39] (Dynamic Graph Convolutional Neural Network). Subsequent steps involve an attention-based module and differentiable SVD modules for point-to-point registration. Conversely, DeepUME [26] adopts a distinct approach by projecting points into a space invariant under $\mathcal{SO}(3)$ transformations. This space is leveraged to compute invariant per-point features that address real-world data sampling issues. Employing two weight-shared transformers and DGCNN [39], correspondences are established and fed into UME [15] for rigid transformation parameter estimation. A limitation of both DCP [37] and DeepUME [26] is susceptibility to failure in scenarios where either the source or target point cloud exhibits arbitrary (lower) density than the others—an occurrence commonly encountered in 3D scan registration from diverse sensors. This limitation is attributed to the methods' reliance on estimating per-point correspondences using local features extracted from edge-conv [39], which operates on graphs constructed using k-nearest neighbors (k-NN).

We introduce Mahalanobis k-NN: A Statistical Lens for Robust Point-Cloud Registration to mitigate this issue. The efficacy of Mahalanobis distance as an evaluation metric has already been demonstrated in reference-based point cloud quality assessment [23]. Our proposed Mahalanobis k-NN can be used as a plugin for any point-cloud registration method. In this paper, we propose mahalnobis versions of DCP-(v1,v2) [37] and DeepUME [26]. Both methods operate on diverse point-cloud representation spaces. This augmentation facilitates point-cloud registration as shown in Figure 1; across various publicly available 3D datasets, encompassing diverse benchmarking scenarios, including: 1) unseen-category evaluation, 2) robustness to various noise types, and 3) efficiency towards varying point densities. Additionally, we demonstrate the discriminative capabilities of the proposed Mahalanobis lens through point cloud few-shot classification tasks. In this context, models pre-trained for registration tasks leverage DGCNN [39] features for few-shot evaluation on both ModelNet40 [40] and ScanObjectNN [36] datasets. Finally, we summarize our contributions as follows:

- We propose Mahalanobis k-NN: A Statistical Lens for Robust Point-Cloud Registration by incorporating Mahalanobis distance in:
  - Deep Closest Point (DCP) [37] that operates on Euclidean coordinate space.
  - Deep Universal Manifold Embedding (DeepUME) [26] that operates on $\mathcal{SO}(3)$ invarient space.
- We demonstrate the efficacy of the proposed method on various point cloud registration tasks on a variety of publicly available data sets and compare the results with state-of-the-art techniques.
- We perform an endurance test to evaluate the robustness and generalization of the proposed method. We achieve state-of-the-art results compared to other point cloud registration methods on all benchmarking strategies and on almost all evaluation metrics.
- To the best of our knowledge, we are the first to demonstrate discriminative prowess inherited in point cloud registration task by benchmarking point cloud few-shot classification on ModelNet40 [40] and ScanObjectNN [36] datasets through features extracted by DCP [37], we report up to an average of 20% increment when Mahalanobis is incorporated.
- We will release our code to facilitate reproducibility and future research upon acceptance.

2 Related Works

Optimization Based. Preceding the advent of the deep learning era in 3D point cloud registration, one common strategy involved extracting and matching spatially local features, as demonstrated in studies like [19, 25, 34, 43, 42, 44]. Many existing methods in this category are adaptations of 2D image processing solutions, such as variants of 3D-SIFT [27] and the 3D Harris key-point detector [35]. However, key-point matching in 3D presents challenges due to the absence of a regular sampling grid, artifacts, and sampling noise, leading to high outlier rates and localization errors. To achieve global alignment, key-point matching typically utilizes outlier rejection methods like RANSAC [16] and is refined using local optimization algorithms [4, 28, 31, 46]. Notably, DGR [11] follows a similar paradigm, but it incorporates learnable inlier detection. Researchers have proposed numerous works for handling outliers and noise [10], formulating robust minimizers [17], and devising more suitable distance metrics. The widely used Iterative Closest Point algorithm (ICP) [4, 46], a popular refinement algorithm, constructs point correspondences based on spatial proximity and employs a transformation estimation step. Over time, various ICP variants [33, 41, 8] have been proposed to enhance convergence rate, robustness, and accuracy in the 3D point cloud registration.

Learning Based. Registration approaches leverage learning-based techniques alongside closed-form methods. Pioneered by PointNet [32] and advanced by DGCNN [39], learning from data for point cloud registration has become powerful (e.g., PointNet++, DCP [37], PRNet [38]). PointNetLK [2] minimizes learned feature distance using a differentiable Lucas-Kanade algorithm, while DCP uses attention and differentiable SVD for point-to-point registration. Recent work [18] introduces graph transformation and deep graph matching. Moreover, DeepUME [26] employ $\mathcal{SO}(3)$ invariant coordinate systems for global registration. DeepGMR [45] tackles pose-invariant correspondences between raw point clouds and Gaussian-mixture models but suffers from noise sensitivity. DeepUME [26] fuses closed-form Universal Manifold Embedding (UME) and deep learning to handle sparse and unevenly sampled point clouds with large transformations. This framework is trained end-to-end and learns joint-resampling strategies and invariant features.

DCP [37] and DeepUME [26] achieve impressive performance but struggle when the source or target point cloud densities are unequal. Both rely on DGCNN [39] for local feature-based correspondence estimation. However, DGCNN's Euclidean metric for k-nearest neighbors lacks surface awareness. We propose Mahalanobis versions of DCP and DeepUME, introducing the Mahalanobis distance for surface-aware features and enhanced registration in real-world scenarios. Furthermore, our proposed approach demonstrates remarkable discriminative capabilities, as evident from its performance in point cloud few-shot classification tasks.

3 Problem Setting

We denote $x_i^T(i \in [1, M])$ and $y_i^T(i \in [1, N])$ as row vectors from matrices $X \in \mathbb{R}^{M\times 3}$ and $Y \in \mathbb{R}^{N\times 3}$ respectively. The matrices X and Y represent two distinct point clouds, often originating from two scans of the same object (we assume Y is transformed from X by an unknown rigid motion). Here, x_i and y_j represent the coordinates of the ith point in X and the jth point in Y, for $i \in [1, M]$ and $j \in [1, N]$ respectively. Assuming that there are S pairs of corresponding points between the two point clouds, the goal of registration is to determine the optimal rigid transformation parameters g (comprising a rotation matrix $R \in \mathcal{SO}(3)$ and a translation vector $t \in \mathbb{R}^3$) that aligns point cloud X to point cloud Y, as illustrated below:

$$\underset{R\in\mathcal{SO}(3),\, t\in\mathbb{R}^3}{\arg\min} \left\| d\left(X, g(Y)\right)\right\|_2^2 \tag{1}$$

The term $d\big(X, g(Y)\big)$ is a measure of the projection error between X and the transformed Y, expressed as $d\big(X, RY+t\big) = \sum_{s=1}^{S} \left\| x_s - \big(Ry_s + t\big)\right\|_2$, where s iterates over the S pairs of corresponding points. The optimization problem in Equation 1 embodies a well-known "chicken-and-egg" scenario: determining the optimal transformation matrix requires knowledge of true correspondences [5, 6]; conversely, accurate correspondences can be established given the optimal transformation matrix. However, the simultaneous resolution of both aspects poses a non-trivial challenge.

4 Method

In this paper, we focus on point cloud registration using DCP [37] and DeepUME [26] for global one-shot rigid transformations (contrasting iterative methods like ICP [4]). Both rely on DGCNN [39] for local features but use static k-nearest neighbor (k-NN) graphs, limiting semantic space creation. Unlike, DGCNN's dynamic k-NN graphs, these methods restrict feature projection depth. We propose using the Mahalanobis distance [29] to enhance k-NN, enabling surface-aware neighbor selection based on principal components. This extracts surface and corner features crucial for accurate matching and robust transformation estimation.

4.1 Mahalanobis k-NN

In this section, we introduce Mahalanobis distance as a statistical lens for robust feature matching in DCP [37] and DeepUME [26]. We modify DGCNN to construct a graph $\mathcal{G}^\dagger$ using Mahalanobis k-nearest neighbors (k-NN) for enhanced feature selection. DGCNN constructs a graph $\mathcal{G}$, applies nonlinearity for edge values, and performs vertex-wise aggregation (max or $\sum$) in each layer. Let x_i^l be the embedding of point i in the l-th layer, and h_θ^l be a nonlinear function parameterized by a shared multi-layer perceptron (MLP). DGCNN can be formulated as:

$$x_i^l = f\Big(\big\{h_\theta^l(x_i^{l-1}, x_j^{l-1}) \forall j \in \mathcal{N}_i\big\}\Big) \tag{2}$$

where $\mathcal{N}_i$ denotes neighbors of vertex i in $\mathcal{G}$.

We propose a novel augmentation by integrating Mahalanobis k-nearest neighbors (k-NN) in Equation.2 to construct graph $\mathcal{G}^\dagger$. This graph prioritizes surface

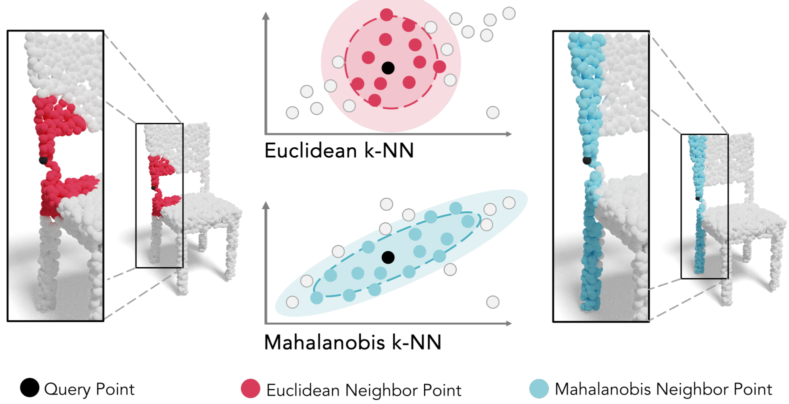

Fig. 2. The illustration contrasts Euclidean distance fields (red) capturing spatial neighbors with Mahalanobis distance fields (blue) considering the underlying data distribution. A *chair* point cloud exemplifies this. The black query point is surrounded by Euclidean neighbors (red points) and Mahalanobis neighbors (blue points). The depiction clearly illustrates the impact of Mahalanobis distance, effectively capturing surface points per the data distribution vital for precise feature matching.

awareness through selection of surficial neighbors $\mathcal{N}_i^{\dagger}$ based on Mahalanobis distance as shown in Figure 2.

To compute Mahalanobis distance between points x_i and x_j, we use:

$$D_M(x_i, x_j) = \sqrt{(x_i - x_j)^T \cdot C^{-1} \cdot (x_i - x_j)} \tag{3}$$

where: $D_M(x_i, x_j)$ is Mahalanobis distance between x_i and x_j, and C is covariance matrix of data points. One can argue, geodesic graph-based methods capture similar or sometimes better surficial information. To counter this, we implement vectorized Floyd-Warshall (refer supplementary for implementation detail on vectorized Floyd-Warshall) on the Euclidean-knn graph and observe that the performance is slower and also not as effective as the proposed Mahalanobis variants of DCP and DeepUME.

The impact of Mahalanobis distance is visually depicted in Figure 2. This surficial awareness proves crucial in feature matching. Incorporating Mahalanobis distance into DCP [37] and DeepUME [26] yields superior performance in point cloud registration. Discriminative power is affirmed in point-cloud few-shot classification, with pre-trained DGCNN models from vanilla DCP and Mahalanobis DCP showcasing remarkable classification capabilities.

5 Experiments

In this section, we showcase the superior performance of our proposed method in the domain of point cloud registration when compared against existing state-of-the-art techniques such as ICP [4], GO-ICP [41], FGR [48], PointNetLK [2], DCP [37], and DeepUME [26] on publicly available datasets. Our evaluation encompasses a spectrum of benchmarking strategies, encompassing scenarios where one point cloud is notably sparser than the other. For a fair comparison, we implemented a vectorized Floyd-Warshall algorithm in PyTorch to compute geodesic distances and compare it to our proposed method for surface feature extraction. We further validate the Mahalanobis metric's discriminative power in few-shot learning tasks and showcase robustness through endurance tests. Our model was trained using an Nvidia RTX 3090 GPU and PyTorch 1.11.

5.1 Point Cloud Registration

We build upon the model architecture, training strategies, evaluation metrics, and hyperparameters established in DCP [37] [1]. We introduce the Mahalanobis version of DCP (MDCP), detailed in Section 4.1. MDCP comes in two variants: v1 without a transformer and v2 with a transformer. We further extend our approach to DeepUME [26] by proposing the Mahalanobis variant, MDeepUME. For a fair comparison, we benchmark geodesic versions of these models using vectorized Floyd-Warshall on Euclidean k-NN graphs.
Dataset. To facilitate benchmarking, we employ publicly available datasets: ModelNet40 [40], FAUST [7], and Stanford3D (S-3D)[2] . Notably, the latter two datasets are exclusively employed for testing purposes. To ensure methodological consistency, we adhere to the official training/testing splits and settings as stipulated in the original works of DCP [37] and DeepUME [26]. Among these datasets, our approach is rigorously evaluated.

We evaluate using root mean squared error (RMSE) between ground truth and predicted transformations (lower is better). Angular errors are reported in degrees. These metrics comprehensively assess our approach's performance.

5.2 Comparison with State-of-the-art Methods

ModelNet40 Results. Table 1 compare our methods (MDCP, MDeepUME) with baselines and their geodesic versions. We outperform all methods across various settings: **Unseen data**: Train-test split from ModelNet40 [40]; **Unseen category**: First 20 classes for training, remaining for testing; **Gaussian Noise**: $\mathcal{N}(0, 0.01)$ clipped to [-0.05, 0.05]. The point density is 1024 in all settings. Baseline evaluations are from [37] (highly comparable results achieved with minimal standard deviation during our reproduction attempts). For each case, a rigid transformation is applied along each axis (rotation: $[0°, 45°]$, translation: [-0.5, 0.5]). The network receives both the original and transformed point cloud as input, evaluated against ground truth using MDCP and MDeepUME.

[1] https://github.com/WangYueFt/dcp
[2] http:// graphics.stanford.edu/data/3Dscanrep/

Table 1. Comparison on ModelNet40 [40] for unseen data, unseen category, and Gaussian noise. MDCP-(v1,v2) and MDeepUME outperform baselines (**bold** best, underline second best). Lower is better. † denotes reproduced results, -Geo refers to Floyd-Warshall geodesic-graph version.

	Unseen data		Unseen Category		Unseen data + Noise	
	RMSE(R)	RMSE(t)	RMSE(R)	RMSE(t)	RMSE(R)	RMSE(t)
ICP [4]	29.9148	0.2909	29.8764	0.2933	29.7080	0.2906
GO-ICP [41]	11.8523	0.0257	13.8657	0.0226	11.4535	0.0231
FGR [48]	9.3628	0.0139	9.8490	0.0135	24.6515	0.1090
PointNetLK [2]	15.0954	0.0221	17.5021	0.0280	16.0049	0.0216
DCP-v1 [37]	2.5457	0.0018	4.3819	0.0050	2.6318	0.0018
DCP-v1† [37]	2.6581	0.0009	6.8386	0.0035	2.6048	0.0009
DCP-v1-Geo	2.6001	0.0007	3.3326	0.0017	2.1778	0.0019
MDCP-v1 (Ours)	**1.6614**	**0.0004**	**1.9679**	**0.0004**	**1.5889**	**0.0004**
DCP-v2 [37]	1.1434	0.0018	3.1502	0.0050	1.0814	**0.0015**
DCP-v2† [37]	1.4125	0.0018	4.2246	0.0063	1.1479	0.0016
DCP-v2-Geo	1.4478	0.0017	5.5233	0.0051	2.1668	0.0019
MDCP-v2 (Ours)	**0.9468**	**0.0004**	**1.3849**	**0.0049**	**0.8481**	0.0043
DeepUME† [26]	0.0062	**0**	0.0183	0.00019	2.5263	**0.0006**
DeepUME-Geo	0.0051	**0**	0.0106	0.00014	2.0086	0.0008
MDeepUME	**0.0023**	**0**	**0.0098**	**0.00009**	**1.9946**	**0.0006**

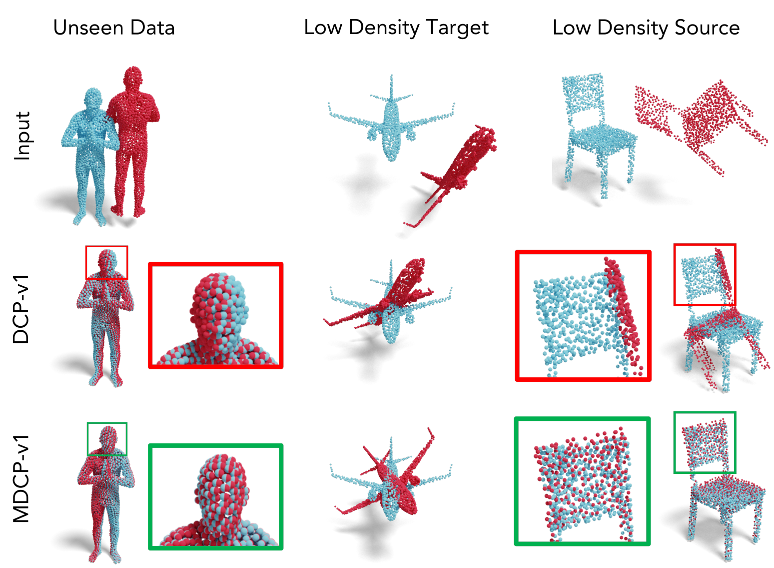

Fig. 3. Robustness evaluation on unseen data (FAUST [7] hand), low-density source (airplane), and low-density target (chair) point clouds (trained on ModelNet40 [40]). Red boxes highlights vulnerabilities in DCP-v1 [37], while green boxes highlights MDCP-v1's efficacy.

Table 2. Evaluation of few-shot classification on ModelNet40 [40] and ScanObjectNN [36] (OBJ ONLY, OBJ+BG, PB75 splits). Our DCGNN in MDCP-(v1,v2) outperforms baselines across all settings (**bold** denotes best accuracy). We report mean accuracy (%) and standard deviation from 50 runs.

ModelNet40 [40]								
	5 way				**10 way**			
Method	**10 shot**		**20 shot**		**10 shot**		**20 shot**	
DCP-v1 [37]	67.6	±6.91	69	±7.87	50.5	±7.74	57.1	±5.01
MDCP-v1 (Ours)	**75.2**	**±7.3**	**80**	**±7.74**	**71.0**	**±4.16**	**70.9**	**±5.72**
Increment in %	**11.24**		**15.94**		**40.59**		**24.17**	
DCP-v2 [37]	76.8	±4.01	79.8	±6.17	60.8	±4.93	69.15	±5.7
MDCP-v2 (Ours)	**82.5**	**±6.96**	**81.8**	**±7.78**	**67.9**	**±5.43**	**72.25**	**±6.33**
Increment in %	**7.42**		**2.51**		**11.68**		**4.48**	
ScanObjectNN OBJ ONLY [36]								
DCP-v1 [37]	51.7	±8.08	54.8	±8.82	32.15	±3.56	34.55	±3.17
MDCP-v1 (Ours)	**54.7**	**±7.47**	**57.1**	**±6.51**	**39.0**	**±4.88**	**41.2**	**±5.72**
Increment in %	**5.80**		**4.20**		**21.31**		**19.25**	
DCP-v2 [37]	55.1	±9.57	56.2	±6.8	38.45	±5.08	42.05	±4.93
MDCP-v2 (Ours)	**55.4**	**±12.11**	**62.4**	**±7.91**	**42.85**	**±3.37**	**46.15**	**±2.96**
Increment in %	**0.54**		**11.03**		**11.44**		**9.75**	
ScanObjectNN OBJ + BG [36]								
DCP-v1 [37]	50.3	±5.25	48.2	±7.35	29.45	±5.34	32.15	±2.9
MDCP-v1 (Ours)	**62.2**	**±6.73**	**60.2**	**±7.54**	**37.25**	**±3.26**	**41.45**	**±3.21**
Increment in %	**23.66**		**24.90**		**26.49**		**28.93**	
DCP-v2 [37]	52.3	±6.92	55	±8.16	33.8	±4.31	36.25	±4.57
MDCP-v2 (Ours)	**58.8**	**±8.07**	**62.8**	**±6.22**	**41.9**	**±6.2**	**44.6**	**±4.1**
Increment in %	**12.43**		**14.18**		**23.96**		**23.03**	
ScanObjectNN PB75 [36]								
DCP-v1 [37]	46.1	±8.31	45.2	±9.37	25.2	±3.78	28.45	±4.5
MDCP-v1 (Ours)	**54.8**	**±7.35**	**51.5**	**±7.08**	**34.05**	**±4.2**	**37.65**	**±4.69**
Increment in %	**18.87**		**13.94**		**35.12**		**32.34**	
DCP-v2 [37]	49.6	±6.64	49.6	±8.7	27.85	±3.75	32.35	±4.03
MDCP-v2 (Ours)	**54.7**	**±5.89**	**53.2**	**±9.46**	**34.45**	**±3.48**	**41.15**	**±4.87**
Increment in %	**10.28**		**7.26**		**23.70**		**27.20**	

Few-shot Classification. Table 2 evaluates the discriminative power of Mahalanobis distance. We compare features learned by DCP-v1/v2 [37] and MDCP-v1/v2 in few-shot classification tasks using the k-way, m-shot setting [1]. Models are trained for registration on ModelNet40 [40] and then tested on few-shot classification for ModelNet40 and three ScanObjectNN [36] splits (OBJ ONLY, OBJ+BG, PB75).

MDCP consistently outperforms DCP across all settings. Summarizing Table 2, MDCP-(v1,v2) achieves average accuracy improvement of $(22.98, 6.52)$ on ModelNet40. For ScanObjectNN, improvements are: OBJ ONLY: $(12.68, 8.19)$; OBJ+BG: $(25.99, 18.40)$; PB75: $(25.06, 17.11)$, [(a, b) denotes average accuracy increment (%) for MDCP-(v1,v2)].

These results suggest two key points: 1) v1 (without transformer) outperforms v2, indicating transformers may struggle with limited data; 2) Even on unseen ScanObjectNN data, MDCP shows consistent improvement across splits.

5.3 Endurance Test

Table 3. Quantitative comparison of MDCP-(v1,v2) robustness against original on ModelNet40 [40] for low-density point cloud registration. We evaluate across various settings. MDCP consistently outperforms its original counterpart (**bold** denotes best). Lower is better for RMSE(t) (reported in 10^{-2} units). **Note:** -Geo refers to the vectorized Floyd-Warshall geodesic-graph version.

		Unseen Data		Unseen Category		Noise	
		RMSE(R)	RMSE(t)	RMSE(R)	RMSE(t)	RMSE(R)	RMSE(t)
Target Low Density ModelNet40 [40]							
1024	**DCP-v1 [37]**	42.0026	10.865	8.3181	1.2206	32.0181	0.9616
	DCP-v1-Geo	38.1172	10.863	6.0088	1.2806	39.9996	**0.9801**
	MDCP-v1	**32.6880**	**10.977**	**5.8425**	**1.2096**	**24.9222**	1.0062
	DCP-v2 [37]	68.0209	3.6777	14.9977	**2.7765**	66.9599	3.0532
	DCP-v2-Geo	44.6676	3.7786	14.7762	8.1678	48.2278	2.9987
	MDCP-v2	**34.9602**	**3.3871**	**14.3836**	6.1023	**25.8595**	**2.8308**
512	**DCP-v1 [37]**	54.5579	1.6058	**8.31497**	1.6812	50.4602	1.4233
	DCP-v1-Geo	48.1628	1.7898	8.1065	1.6711	49.7866	1.5661
	MDCP-v1	**42.3370**	**1.4918**	8.4701	**1.6028**	**38.0739**	**1.3168**
	DCP-v2 [37]	80.8099	8.9802	**23.8370**	**3.7857**	77.4393	8.7660
	DCP-v2-Geo	77.8756	6.7752	21.8861	3.6544	48.2165	8.7765
	MDCP-v2	**47.3271**	**3.9172**	28.7305	8.4551	**44.3240**	**3.4637**
Source Low Density ModelNet40 [40]							
1024	**DCP-v1 [37]**	43.2637	1.1146	8.304669	1.228	38.6104	1.0306
	DCP-v1-Geo	36.6378	1.1123	**8.0011**	1.2761	39.1654	1.1756
	MDCP-v1	**31.3686**	**1.1035**	8.3409	**1.2247**	**26.9862**	**1.0158**
	DCP-v2 [37]	65.4777	3.1448	16.1993	**1.9267**	61.1659	2.6327
	DCP-v2-Geo	57.1998	3.0098	14.7176	6.7861	49.1109	2.9987
	MDCP-v2	**38.6540**	**2.8759**	**7.4574**	5.6750	**27.1185**	**2.6114**
512	**DCP-v1 [37]**	55.2426	1.5553	8.4311	1.4074	51.7135	1.3908
	DCP-v1-Geo	**38.1172**	1.0863	**6.0088**	1.2806	39.9996	**0.9801**
	MDCP-v1	42.4809	**1.4700**	7.8250	1.1089	**39.6634**	**1.3272**
	DCP-v2 [37]	71.8377	7.2431	**22.3028**	4.8566	71.1976	6.9627
	DCP-v2-Geo	**44.6676**	3.7698	26.8861	4.9908	65.9981	5.0098
	MDCP-v2	50.1560	**3.6758**	29.2851	**4.4566**	**45.7104**	**3.3395**

Efficiency towards Point-density. Table 3 evaluates MDCP's robustness against varying point densities, simulating real-world sensor data [22]. One point cloud (source or target) is downsampled by half (e.g., 2048 → 1024). Evaluation metrics use the same number of points for both clouds (e.g., source with 1024 points and target with 2048 points are used to compute transformation, then applied to original points for error calculation).

Figure 3 compares DCP-v1 [37] and MDCP-v1. When densities are similar (man's face), both perform well. However, MDCP-v1 demonstrates superior robustness under varying densities (chair). This highlights its effectiveness in real-world applications where source or target point clouds may have different densities due to factors like sensor capabilities or data acquisition methods (e.g., SFM-LiDAR fusion).

Note: Source point cloud is a transformed version of $g(Y)$, and the goal is to estimate g for perfect alignment with target X.

Table 4. Quantitative comparison of proposed MDeepUME's robustness on ModelNet40 [40], FAUST [7], and Stanford 3D datasets. We evaluate across various noise, benchmarked by authors in DeepUME [26]. MDeepUME consistently outperforms DeepUME (**bold** denotes best results). Lower is better for all metrics: RMSE(t), chamfer distance (CD) [3], and Hausdorff distance (HD) [14] (reported in 10^{-2} units).

	Noise-types	DeepUME [26]				MDeepUME			
		CD	**HD**	**RMSE(R)**	**RMSE(t)**	**CD**	**HD**	**RMSE(R)**	**RMSE(t)**
MN 40	**Bernoulli**	1.113	8.6300	46.39461	**1.5381**	**1.1120**	**8.6200**	**45.7815**	1.5398
	Gaussian	0.1913	1.2445	2.526296	**0.0625**	**0.1900**	**1.2400**	**1.9946**	0.0642
	Sampling	0.6317	5.6678	31.39413	**0.9197**	**0.6290**	**5.6620**	**30.7709**	0.9236
	Z-Intersection	1.2237	11.3676	89.48223	**0.9003**	**1.2230**	**11.3670**	**87.4486**	0.911
FAUST	**Bernoulli**	**0.2608**	**2.7165**	11.30384	**2.2294**	0.2728	2.7818	**10.3421**	2.1964
	Gaussian	0.1243	0.9342	1.819818	**0.1085**	**0.1230**	**0.9300**	**1.7029**	0.2692
	Sampling	0.1026	0.9342	5.6842	**1.172**	**0.0941**	**0.9320**	**3.7985**	0.9461
	Z-Intersection	0.2064	2.3368	12.3721	**2.2183**	**0.2000**	**2.3350**	**11.9663**	2.1383
S-3D	**Bernoulli**	**0.2345**	10.4470	**5.951346**	**1.2268**	0.2470	**10.1853**	6.4044	1.2266
	Gaussian	0.1068	0.9477	0.392542	**0.0591**	**0.1040**	**0.9410**	**0.3106**	0.0604
	Sampling	0.0908	8.5766	6.709815	**0.7145**	**0.0900**	**8.4267**	**6.1997**	0.7143
	Z-Intersection	0.1771	11.7627	5.66907	**1.0417**	**0.1700**	**11.5800**	**4.4850**	1.0428

Efficiency towards Various Noise Types. Table 4 evaluates the robustness of MDeepUME against various noise types from DeepUME [26] (Bernoulli, Gaussian, Sampling, Zero-Intersection). We benchmark on ModelNet40 [40] (training and evaluation), FAUST [7], and Stanford3D (testing only). Training settings follow the original paper [26] ([3]).

MDeepUME outperforms DeepUME across datasets and noise types in terms of Chamfer distance (CD) [3], Hausdorff distance (HD) [14], and root mean squared error in rotation (RMSE(R)). However, limitations are observed in translation accuracy (RMSE(t)) due to DeepUME's inherent projection into SO(3)-invariant space, where Mahalanobis distance is less effective.

6 Conclusion

Our paper introduces Mahalanobis k-NN, a statistical framework for improving point cloud registration in learning-based methods. It addresses challenges like feature matching due to variations in point cloud densities. Mahalanobis k-NN leverages local neighborhood distribution for accurate feature extraction, outperforming methods like Flyod-Warshall. It integrates seamlessly into local

[3] https://github.com/langnatalie/DeepUME

graph-based point cloud analysis methods like DCP and DeepUME, achieving state-of-the-art performance on benchmark datasets. We also demonstrate that registered point cloud features possess discriminative capabilities, leading to significant improvements in few-shot classification tasks. Mahalanobis k-NN offers exceptional surface awareness and outlier robustness, benefiting tasks like segmentation, decomposition, and normal estimation. Additionally, it serves as a surficial geometry-based error metric, enhancing tasks like generation and upsampling. Its versatility makes it a crucial component in advancing point cloud processing, warranting further benchmarking to highlight its significance.

References

[1] Mohamed Afham et al. "CrossPoint: Self-Supervised Cross-Modal Contrastive Learning for 3D Point Cloud Understanding". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 9902–9912.

[2] Yasuhiro Aoki et al. "PointNetLK: Robust & Efficient Point Cloud Registration Using PointNet". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[3] Harry G Barrow et al. "Parametric correspondence and chamfer matching: Two new techniques for image matching". In: *Proceedings: Image Understanding Workshop*. Science Applications, Inc. 1977, pp. 21–27.

[4] Paul J Besl and Neil D McKay. "Method for registration of 3-D shapes". In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. Spie. 1992, pp. 586–606.

[5] Paul J Besl and Neil D McKay. "Method for registration of 3-D shapes". In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. Spie. 1992, pp. 586–606.

[6] Seth D Billings, Emad M Boctor, and Russell H Taylor. "Iterative most-likely point registration (IMLP): A robust algorithm for computing optimal shape alignment". In: *PloS one* 10.3 (2015), e0117688.

[7] Federica Bogo et al. "FAUST: Dataset and evaluation for 3D mesh registration". In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. Piscataway, NJ, USA: IEEE, June 2014.

[8] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. "Sparse iterative closest point". In: *Computer graphics forum*. Vol. 32. 5. Wiley Online Library. 2013, pp. 113–123.

[9] Qiangqiang Cheng et al. "A morphing-Based 3D point cloud reconstruction framework for medical image processing". In: *Computer methods and programs in biomedicine* 193 (2020), p. 105495.

[10] Dmitry Chetverikov, Dmitry Stepanov, and Pavel Krsek. "Robust Euclidean alignment of 3D point sets: the trimmed iterative closest point algorithm". In: *Image and vision computing* 23.3 (2005), pp. 299–309.

[11] Christopher Choy, Wei Dong, and Vladlen Koltun. "Deep global registration". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2514–2523.

[12] Yaodong Cui et al. "Deep learning for image and point cloud fusion in autonomous driving: A review". In: *IEEE Transactions on Intelligent Transportation Systems* 23.2 (2021), pp. 722–739.

[13] Bertrand Douillard et al. "On the segmentation of 3D LIDAR point clouds". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 2798–2805.

[14] M-P Dubuisson and Anil K Jain. "A modified Hausdorff distance for object matching". In: *Proceedings of 12th international conference on pattern recognition*. Vol. 1. IEEE. 1994, pp. 566–568.

[15] 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)*. IEEE. 2019, pp. 5157–5161.

[16] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[17] Andrew W Fitzgibbon. "Robust registration of 2D and 3D point sets". In: *Image and vision computing* 21.13-14 (2003), pp. 1145–1153.

[18] Kexue Fu et al. "Robust Point Cloud Registration Framework Based on Deep Graph Matching". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 8893–8902.

[19] Yulan Guo et al. "A comprehensive performance evaluation of 3D local feature descriptors". In: *International Journal of Computer Vision* 116 (2016), pp. 66–89.

[20] Deepti Hegde et al. "Relocalization of camera in a 3d map on memory restricted devices". In: *Computer Vision, Pattern Recognition, Image Processing, and Graphics: 7th National Conference, NCVPRIPG 2019, Hubballi, India, December 22–24, 2019, Revised Selected Papers 7*. Springer. 2020, pp. 548–557.

[21] Dikshit Hegde et al. "DA-AE: Disparity-Alleviation Auto-Encoder Towards Categorization of Heritage Images for Aggrandized 3D Reconstruction." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5093–5100.

[22] Xiaoshui Huang et al. "A comprehensive survey on point cloud registration". In: *arXiv preprint arXiv:2103.02690* (2021).

[23] Alireza Javaheri et al. "Mahalanobis based point to distribution metric for point cloud geometry quality evaluation". In: *IEEE Signal Processing Letters* 27 (2020), pp. 1350–1354.

[24] Bowen Jing et al. "Torsional diffusion for molecular conformer generation". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24240–24253.

[25] Andrew E Johnson and Martial Hebert. "Using spin images for efficient object recognition in cluttered 3D scenes". In: *IEEE Transactions on pattern analysis and machine intelligence* 21.5 (1999), pp. 433–449.

[26] Natalie Lang and Joseph M Francos. "Deepume: Learning the universal manifold embedding for robust point cloud registration". In: *arXiv preprint arXiv:2112.09938* (2021).

[27] Chris Maes et al. "Feature detection on 3D face surfaces for pose normalisation and recognition". In: *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. IEEE. 2010, pp. 1–6.

[28] Martin Magnusson, Achim Lilienthal, and Tom Duckett. "Scan registration for autonomous mining vehicles using 3D-NDT". In: *Journal of Field Robotics* 24.10 (2007), pp. 803–827.

[29] Goeffrey J McLachlan. "Mahalanobis distance". In: *Resonance* 4.6 (1999), pp. 20–26.

[30] François Pomerleau, Francis Colas, Roland Siegwart, et al. "A review of point cloud registration algorithms for mobile robotics". In: *Foundations and Trends® in Robotics* 4.1 (2015), pp. 1–104.

[31] Helmut Pottmann, Stefan Leopoldseder, and Michael Hofer. "Registration without ICP". In: *Computer Vision and Image Understanding* 95.1 (2004), pp. 54–71.

[32] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[33] Szymon Rusinkiewicz and Marc Levoy. "Efficient variants of the ICP algorithm". In: *Proceedings third international conference on 3-D digital imaging and modeling*. IEEE. 2001, pp. 145–152.

[34] Radu Bogdan Rusu et al. "Aligning point cloud views using persistent feature histograms". In: *2008 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2008, pp. 3384–3391.

[35] Ivan Sipiran and Benjamin Bustos. "Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes". In: *The Visual Computer* 27 (2011), pp. 963–976.

[36] Mikaela Angelina Uy et al. "Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data". In: *International Conference on Computer Vision (ICCV)*. 2019.

[37] 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*. 2019, pp. 3523–3532.

[38] Yue Wang and Justin M. Solomon. "PRNet: Self-Supervised Learning for Partial-to-Partial Registration". In: *33rd Conference on Neural Information Processing Systems (To appear)*. 2019.

[39] Yue Wang et al. "Dynamic graph cnn for learning on point clouds". In: *ACM Transactions on Graphics (tog)* 38.5 (2019), pp. 1–12.

[40] Zhirong Wu et al. "3D ShapeNets for 2.5D Object Recognition and Next-Best-View Prediction". In: *CoRR* abs/1406.5670 (2014). arXiv: 1406.5670. URL: http://arxiv.org/abs/1406.5670.

[41] Jiaolong Yang et al. "Go-ICP: A globally optimal solution to 3D ICP point-set registration". In: *IEEE transactions on pattern analysis and machine intelligence* 38.11 (2015), pp. 2241–2254.

[42] Jiaqi Yang, Zhiguo Cao, and Qian Zhang. "A fast and robust local descriptor for 3D point cloud registration". In: *Information Sciences* 346 (2016), pp. 163–179.

[43] Jiaqi Yang, Yang Xiao, and Zhiguo Cao. "Toward the repeatability and robustness of the local reference frame for 3D shape matching: An evaluation". In: *IEEE Transactions on Image Processing* 27.8 (2018), pp. 3766–3781.

[44] Jiaqi Yang et al. "Rotational contour signatures for both real-valued and binary feature representations of 3D local shape". In: *Computer Vision and Image Understanding* 160 (2017), pp. 133–147.

[45] Wentao Yuan et al. "DeepGMR: Learning Latent Gaussian Mixture Models for Registration". In: *European Conference on Computer Vision*. Springer. 2020, pp. 733–750.

[46] Zhengyou Zhang. "Iterative point matching for registration of free-form curves and surfaces". In: *International journal of computer vision* 13.2 (1994), pp. 119–152.

[47] Zhengyou Zhang. "Microsoft kinect sensor and its effect". In: *IEEE multimedia* 19.2 (2012), pp. 4–10.

[48] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. "Fast global registration". In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer. 2016, pp. 766–782.

Mahalanobis k-NN: A Statistical Lens for Robust Point-Cloud Registrations

Tejas Anvekar and Shivanand Venkanna Sheshappanavar

University of Wyoming, 1000 E University Ave, Laramie, WY 82071

1 Supplimentary

In this supplementary document, we provide additional quantitative results, technical details, and more qualitative test examples to the main paper. We also discuss the implementation of geodesic-knn using vectorized flyod-warshall and compare with proposed method.

1.1 Comparision with Geodesic-knn

While Floyd-Warshall calculates the shortest paths between points, as shown in Figure 2, Mahalanobis distance focuses on a point's similarity to the overall distribution of the other point cloud, considering both mean and covariance as shown in Figure 1. This characteristic makes Mahalanobis distance well-suited for extracting surface features, as points on similar surfaces share statistical properties. Furthermore, the proposed method offers significant advantages in computational efficiency, unlike vectorized Floyd-Warshall, which requires iterating over the batch size during training. Additionally, the main paper's results in Tables 1 and 3 demonstrate that point cloud registration using Mahalanobis distance achieves superior performance compared to Floyd-Warshall geodesic distance. These results suggest that Mahalanobis distance might more effectively capture the underlying relationships between corresponding points across different point clouds, leading to more accurate registrations.

1.2 Results

Linear Classification: Table 1 verifies the proposed Mahalanobis-knn's discriminative prowess. We conduct Linear Classification using the features generated from registration-only trained DGCNN backbone in DCP/MDCP-(v1,v2). We observe almost an increment of 20%, on the ScanObjectNN dataset for Linear Classification over vanilla registration-only trained DGCNN.

Feature Space Visualization: To substantiate the assertion that Mahalanobis distance operates as a statistical lens for enhanced feature matching, we undertake KMeans clustering within the initial embedding space of DGCNN. This embedding space, a 64-dimensional domain following the first edge convolution, is

```python
def pairwise_mahalanobis_distance(point_cloud_1, point_cloud_2):
    """
    Compute pairwise Mahalanobis distance between two 3D point clouds.

    Args:
        point_cloud_1 (torch.Tensor): First input point cloud tensor of size (B, N, 3).
        point_cloud_2 (torch.Tensor): Second input point cloud tensor of size (B, M, 3).

    Returns:
        torch.Tensor: Pairwise Mahalanobis distance tensor of size (B, N, M).
    """
    B, N, _ = point_cloud_1.size()
    _, M, _ = point_cloud_2.size()

    # Calculate the mean of each point cloud
    mean_1 = torch.mean(point_cloud_1, dim=1, keepdim=True)  # Shape: (B, 1, 3)
    mean_2 = torch.mean(point_cloud_2, dim=1, keepdim=True)  # Shape: (B, 1, 3)

    # Calculate the centered point clouds
    centered_point_cloud_1 = point_cloud_1 - mean_1  # Shape: (B, N, 3)
    centered_point_cloud_2 = point_cloud_2 - mean_2  # Shape: (B, M, 3)

    # Calculate the covariance matrix of the first point cloud
    covariance_matrix_1 = torch.matmul(centered_point_cloud_1.transpose(1, 2),
                                       centered_point_cloud_1) / N  # Shape: (B, 3, 3)
    covariance_matrix_2 = torch.matmul(centered_point_cloud_2.transpose(1, 2),
                                       centered_point_cloud_2) / M   # Shape: (B, 3, 3)

    covariance_matrix_1 = (covariance_matrix_1 + covariance_matrix_2)/(2)

    eye = torch.eye(n=covariance_matrix_1.shape[-1],device=point_cloud_1.device)*0.00001
    covariance_matrix_1+=eye

    # Invert the covariance matrix
    inv_covariance_matrix_1 = torch.inverse(covariance_matrix_1)

    # Expand the tensors for broadcasting
    centered_point_cloud_1_expanded = centered_point_cloud_1.unsqueeze(2)  # Shape: (B, N, 1, 3)
    centered_point_cloud_2_expanded = centered_point_cloud_2.unsqueeze(1)  # Shape: (B, 1, M, 3)
    inv_covariance_matrix_1_expanded = inv_covariance_matrix_1.unsqueeze(1)  # Shape: (B, 1, 3, 3)

    # Compute the pairwise Mahalanobis distance
    pairwise_distance = torch.matmul(torch.matmul(centered_point_cloud_1_expanded,
                                                  inv_covariance_matrix_1_expanded),
                                     centered_point_cloud_2_expanded.transpose(-2, -1))  # Shape: (B, N, M)

    pairwise_distance = pairwise_distance.squeeze(2)
    return -pairwise_distance
```

Fig. 1. A more readable Pytorch Implementation of Mahalanobis Distance.

employed in DCP-v1 and MDCP-v1. Our findings are presented visually through Figure 3. The depicted results unequivocally showcase the preservation of surface awareness in the Mahalanobis variant, even in scenarios involving over-clustering (e.g., K=5). This outcome stands in contrast to its Euclidean counterpart. Particularly is the consistent maintenance of surface awareness throughout various clustering cases, particularly evident in the context of the Airplane dataset. This affirmation highlights that the Mahalanobis version of DCP learns to discern and emphasize surface-aware regions crucial in achieving precise feature matching.

```python
def compute_geodesic_distance_vectorised(point_cloud):
    """
    Compute pairwise geodesic (Floyd Warshall) distance.

    Args:
        point_cloud: Input point cloud tensor of size (B, N, 3).

    Returns:
        geodesic_distances: Pairwise geodesic distance tensor of size (B, N, N).
    """
    num_points = point_cloud.shape[1]

    # Compute pairwise distances using Euclidean distance
    pairwise_distances = torch.sum((point_cloud.unsqueeze(2) \
                                    - point_cloud.unsqueeze(1))**2,
                                    dim=-1) #BNN

    # Initialize distance matrix for geodesic distances
    geodesic_distances = pairwise_distances.clone()
    # Floyd-Warshall algorithm for computing geodesic distances
    for k in range(num_points):
        geodesic_distances = torch.min(geodesic_distances,
                                       geodesic_distances[:, :, k:k+1] \
                                        + geodesic_distances[:, k:k+1, :])

    return geodesic_distances
```

Fig. 2. A more readable Pytorch Implementation of Vectorised Flyod-Warshall Geodesic Distance. Here, instead of three loops, we have one, so the time complexity is reduced to $\mathcal{O}(\mathrm{N})$.

Table 1. Evaluation of Linear classification using SVM, on ModelNet40 and ScanObjectNN (OBJ ONLY). Our DCGNN in MDCP-(v1,v2) outperforms baselines across all settings (**bold** denotes best accuracy).

Methods	ModelNet40	ScanObjectNN
DCP-v1	49.39	31.32
MDCP-v1	**65.35**	**42.85**
Increment in %	**24.42**	**26.90**
DCP-v2	69.12	37.69
MDCP-v2	**71.75**	**46.64**
Increment in %	**03.66**	**19.18**

1.3 Limitations

While our approach harnesses the power of Mahalanobis distance for enhanced feature matching and point cloud registration, it is important to acknowledge its limitations. Mahalanobis distance computation critically depends on the accurate estimation of the covariance matrix. When the covariance matrix is ill-conditioned or inadequately estimated due to limited data, the efficacy of the

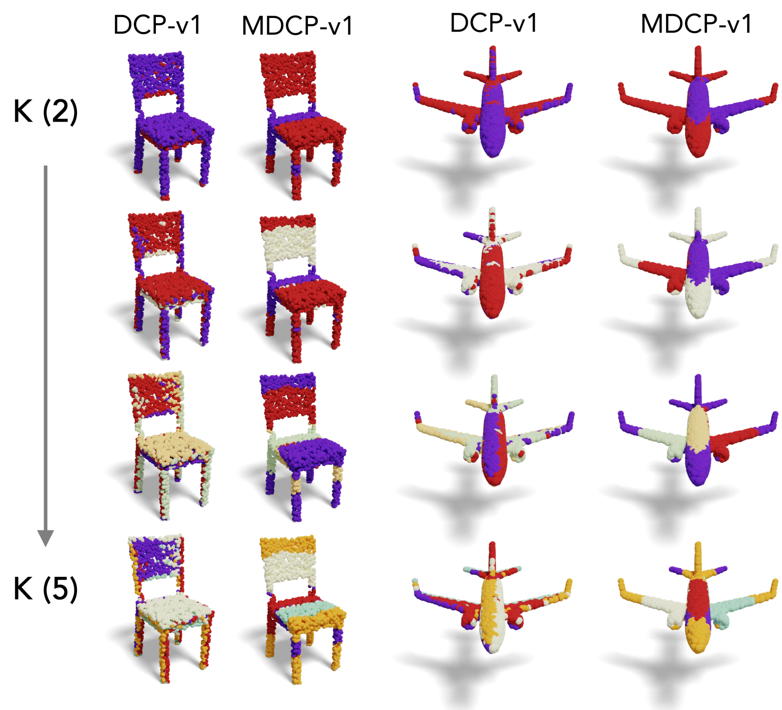

Fig. 3. We present a compelling demonstration showcasing the Mahalanobis-kNN's surface awareness within the proposed MDCP-v1, in contrast to the Euclidean approach in DCP-v1. Here, we conduct KMeansclustering in the initial embedding space of DGCNN, which is employed both in DCP-v1 and MDCP-v1. Each row within the illustration corresponds to K values ranging from 2 to 5, effectively exemplifying the pronounced surficial awareness embodied by MDCP-v1. This surficial awareness is pivotal for robust point cloud registration tasks.

Mahalanobis distance may become compromised. Although we mitigate this concern by introducing a small bias term (10^{-5}) to the diagonal elements of the covariance matrix in our computations, challenges related to accurate covariance estimation persist. This limitation becomes evident in scenarios such as our noise benchmark evaluation in DeepUME, where the Mahalanobis version encounters difficulties in accurately estimating translation parameters, highlighting its sensitivity to covariance matrix quality. Despite these limitations, our approach's innovative integration of Mahalanobis k-NN significantly contributes to addressing key challenges in point cloud registration and feature matching.