CanonNet: Spectral Canonicalization and Curvature-Driven Learning for Compact Local-Geometry Point-Cloud Operators

Benjy Friedmann and Michael Werman

Hebrew University of Jerusalem, Jerusalem, Israel
{benjamin.friedmann, michael.werman}@mail.huji.ac.il

Abstract. *To address the persistent challenges of scalability and robust local geometry representation in point-cloud processing, we propose* ***CanonNet****, a highly efficient local feature operator. CanonNet first employs* ***spectral canonicalization*** *to establish an* invariant local frame *for each neighborhood. It then uses a* ***geometric learning framework****, trained on synthetic surfaces, to distill* fundamental curvature priors *into a lightweight MLP. This design allows CanonNet to achieve competitive performance on various benchmarks with a 100× reduction in parameters, while also exhibiting robust domain transfer. Its efficiency and design make it an effective building block for deep, hierarchical models, acting as a geometric analogue to the convolution operator for capturing multi-scale features.*

Keywords: Point Clouds · Spectral Canonicalization · Shape Analysis · Geometric Neural Networks · Curvature.

1 Introduction

Point clouds, which are unstructured collections of 3D points, have become fundamental to numerous applications including autonomous driving [27], robotics [30], and medical imaging [37]. While point clouds capture detailed geometric information, their unstructured nature presents significant challenges for processing and analysis. Existing approaches have not fully resolved two critical challenges (1) establishing a consistent ordering of points and (2) effectively learning fine-grained geometric features. In this paper, we present *CanonNet*, a novel approach that establishes canonical point ordering and orientation while enhancing the learning of geometric features through synthetic data with *precise* curvature annotations.

The unstructured nature of point clouds necessitates neural architectures that are permutation-invariant [31, 32, 18, 29, 40]. This is typically achieved through operations like pooling, which aggregate point features regardless of their order. PointNet [31] pioneered the application of such operations in point cloud processing (eg: classification, segmentation), though similar permutation-invariant mechanisms had already been explored in Graph Neural Networks (GNNs) [3,

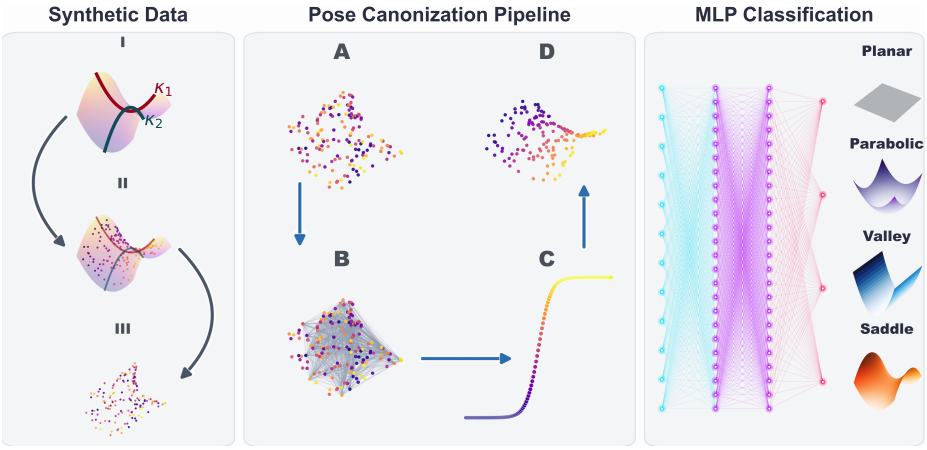

Fig. 1. The complete CanonNet pipeline. **Synthetic Data:** Point clouds are sampled from analytical surfaces with known principal curvatures (κ_1, κ_2) and randomly oriented. **Pose Canonization:** A spectral graph pipeline aligns the arbitrary point cloud into a canonical reference frame. **MLP Classification:** The canonized representation is processed by an MLP to classify it into one of four geometric surfaces.

11]. While effective in ensuring order invariance, these symmetric aggregation functions inherently limit a network's expressivity [23, 19], restricting its ability to capture fine-grained geometric relationships [21].

To enhance neural network expressivity when processing graphs, researchers have developed various positional encoding (PE) methods [16] such as Laplacian-based, random walk-based, and other approaches. These techniques, particularly those using Laplacian eigenvectors, effectively encode structural properties [4, 24, 12, 28]. However, in point cloud processing, PE has primarily been limited to Transformer-based architectures [25, 44, 33, 29], with some Transformer variants deliberately omitting it [18]. We show that Laplacian PE can be harnessed in point cloud processing to achieve canonical order and orientation, eliminating the need for complex transformation-invariant architectures.

The geometric properties inherent in point clouds constitute another valuable source of information that can be harnessed by neural architectures. Rather than relying solely on learned representations, various approaches exploit explicitly computed features such as normals [8, 9, 42], angles [8, 9, 42, 33], and pairwise distances [8, 9, 42, 33] as supplementary inputs to enhance model capabilities. Among these geometric property approaches, approximating curvature directly from point clouds via triangulation and supplying them as features to neural networks has achieved significant performance gains [34]. In this context, a primary constraint is the limited availability of high-quality training data with accurate geometric annotations, which has restricted the evolution of learning-based approaches that can effectively utilize these geometric properties. Real-world point cloud datasets often lack precise geometric ground truth, making it difficult to

train models that can reliably learn and interpret local surface properties. This limitation has particularly affected the development of approaches that aim to understand fine-grained geometric features.

Key Contributions

1. **Canonical Preprocessing**: A novel pipeline that establishes both *canonical ordering* and *orientation* for local point patches, ensuring invariance to point permutations and rigid transformations.
2. **Synthetic Data Generation**: A framework leveraging analytic surfaces with known curvatures, enabling *unlimited training samples* with precise geometric properties.
3. **Lightweight Architecture**: A neural network that effectively learns local geometric features through curvature-based classification.

2 Related Works

Point cloud processing is challenging due to the data's irregularity and lack of inherent order. Our work addresses these issues with two innovations: geometry-aware canonical ordering and curvature-based synthetic data generation. We review related literature across three areas.

Deep Learning for Point Cloud Processing. Early approaches converted point clouds into voxels or projections, losing geometric detail. PointNet [31] introduced direct point-based processing, extended by PointNet++ [32] for hierarchical feature learning. DGCNN [40] improved local modeling through dynamic graphs, while KPConv [38] introduced geometry-adaptive convolutions. Transformer-based models like Point Transformer [44] and PCT [18] leveraged self-attention for geometric relations. Despite progress, these methods are computationally heavy and limited in capturing intrinsic geometry.

Surface Geometry in Point Clouds. Surface properties such as normals and curvatures are crucial for tasks like registration and classification. PCPNet [17] learned local geometry from patches, while DeepFit [5] applied neural surface fitting for accurate normals and curvature estimation. Incorporating geometric cues (distances, angles, normals, curvature) has consistently improved downstream performance [36, 35, 9, 42, 33, 34]. Our method builds on these works, introducing an efficient curvature-based approach that avoids heavy architectures by enforcing invariances in preprocessing.

Ordering and Orientation in Point Clouds. Canonical alignment is another strategy for invariance. STN [20] and its PointNet extension [31] used learned transformations, while PCPNet [17] stabilized orientation via rotation-only constraints. Ordering methods [45, 41, 10] often focus on downsampling rather than true canonical ordering. In contrast, our approach unifies canonical ordering and orientation with curvature-based synthetic supervision, enabling lightweight, permutation-invariant processing with strong geometric consistency.

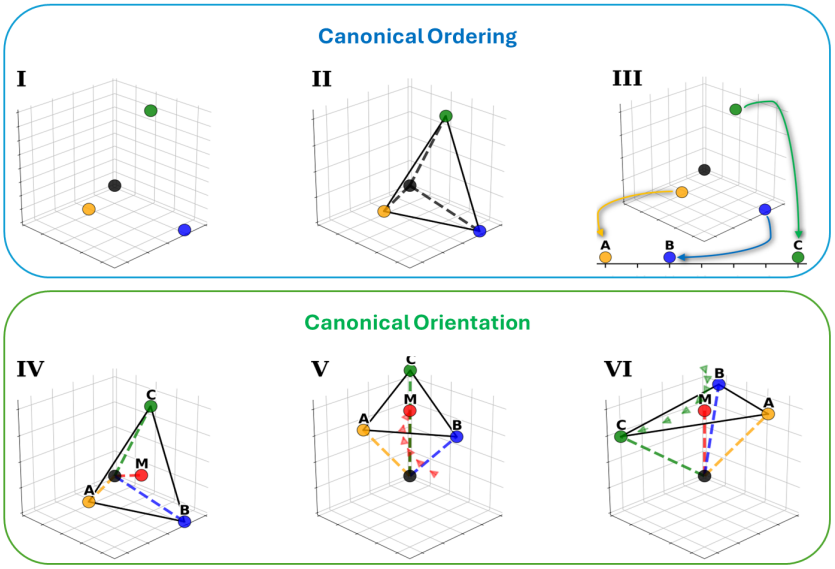

Fig. 2. Illustration of the preprocessing pipeline for establishing canonical point cloud representation as described in Section 3.1: **(I)** Input point cloud with arbitrary ordering and orientation. **(II)** Construction of fully connected graph with heat kernel weights and computation of normalized graph Laplacian. **(III)** Reordering points along a 1D axis based on Laplacian eigenvector values. **(IV)** Identification of geometric landmarks. **(V)** First standardization rotation. **(VI)** Second standardization rotation.

3 Method

We present a framework that combines differential geometry with deep learning to enable efficient point cloud processing. Our approach consists of two main components: (1) a preprocessing pipeline that establishes canonical point ordering and orientation, and (2) a training methodology utilizing synthetically generated surfaces with known geometric properties.

3.1 Preprocessing Pipeline

This section details our preprocessing approach for creating consistent point ordering and orientation.

Local Patch Extraction. For each query point p_q in the 3D point cloud $\mathbf{P}$, we extract a local patch $\mathbf{X}$ using k-nearest neighbors ($k = 20$). This effectively captures the surrounding geometry of the point.

Graph Construction and Spectral Embedding. Given a local patch from a 3D point cloud $\mathbf{X} = \{x_i\}_{i=1}^{N}, x_i \in \mathbb{R}^3$ or in matrix form $\mathbf{X} \in \mathbb{R}^{N\times 3}$, we aim

to establish a consistent point ordering that is invariant to permutations and rigid transformations. To achieve this, we construct a fully connected, undirected graph $\mathcal{G} = (\mathcal{V}, \mathbf{W})$ to capture the local geometric relationships between points. The edge weights $\mathbf{W}$ are defined by the heat kernel:

$$\mathbf{W}_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{t}\right) \quad \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{V} \tag{1}$$

Here $t > 0$ is a temperature parameter controlling the locality of point interactions. We compute the normalized graph Laplacian [7]:

$$\mathbf{L} = \Delta^{-1/2}(\Delta - \mathbf{W})\Delta^{-1/2} \tag{2}$$

where $\Delta \in \mathbb{R}^{N\times N}$ is the diagonal degree matrix with entries $\Delta_{ii} = \sum_{j=1}^{N} \mathbf{W}_{ij}$ for $i = 1, 2, \ldots, N$, and $\Delta_{ij} = 0$ for all $i \neq j$.

Next, we compute the eigenvector $\boldsymbol{\phi} \in \mathbb{R}^N$ corresponding to the smallest nonzero eigenvalue of $\mathbf{L}$ [13, 14]. This eigenvector establishes a spectral embedding where each point $\mathbf{x}_i$ in the point cloud corresponds to the i-th component of $\boldsymbol{\phi}$, effectively projecting the 3D points onto a single line. As demonstrated by [4], this embedding minimizes the weighted sum $\sum_{i,j} \mathbf{W}_{ij}(\phi_i - \phi_j)^2$, ensuring that points with strong connections in the original 3D space remain close in the 1D embedding, thereby optimally preserving the local geometric structure. This preservation of local geometric structure enhances robustness to noise.

It is worth emphasizing that the eigenvector computation for small patches is computationally inexpensive. This efficiency can be further optimized through established iterative methods such as the power method or Lanczos algorithm, preserving the lightweight nature of our preprocessing approach.

Canonical Ordering and Orientation. As illustrated in Fig. 2, we establish a standardized point cloud representation through three complementary transformations that address ordering, position, and orientation. Together, these transformations ensure that geometrically equivalent shapes converge to identical representations regardless of their initial configurations.

The spectral embedding from the previous step forms the basis of our canonical ordering. Sorting the points by their values in the eigenvector $\boldsymbol{\phi}$ defines a permutation $\sigma \in S_N$, such that $\phi_{\sigma(1)} \leq \phi_{\sigma(2)} \leq \ldots \leq \phi_{\sigma(N)}$. We construct the corresponding permutation matrix $\boldsymbol{\Pi}$, where $\Pi_{ij} = 1$ if $j = \sigma(i)$ and 0 otherwise. This matrix is applied to the point cloud, yielding the reordered set of points as follows:

$$\bar{\mathcal{X}} = \boldsymbol{\Pi}\mathcal{X} \tag{3}$$

Next, we normalize the positions by aligning the center of mass with the z-axis. From the reordered point cloud, we compute the center of mass as follows:

$$m = \frac{1}{N}\sum_{i=1}^{N} \bar{\mathbf{x}}_i \tag{4}$$

Next, we compute a rotation matrix $\mathbf{R}_{cm}$ that places m at $(0, 0, \|m\|_2^2)$. Applying this rotation results in:

$$\mathcal{Y} = \mathbf{R}_{cm}\bar{\mathcal{X}} \tag{5}$$

To ensure consistent orientation, we apply a final rotation about the z-axis based on a landmark point. Specifically, we identify the point $p_1 = (x_1, y_1, z_1)$ in $\mathcal{Y}$ that corresponds to the largest value in $\boldsymbol{\phi}$ and compute a rotation $\mathbf{R}_z$ that places p_1 into the XZ-plane with a positive x-coordinate. This yields the final standardized representation:

$$\mathcal{P} = \mathbf{R}_z\mathcal{Y} \tag{6}$$

The full transformation pipeline is as follows:

$$\mathcal{P} = \mathbf{R}_z\mathbf{R}_{cm}\Pi\mathcal{X} \tag{7}$$

Invariance Properties. A key advantage of our preprocessing pipeline is its theoretical guarantees of invariance to both point permutations and rigid transformations. We formally establish these properties below.

Theorem 1. *The proposed preprocessing pipeline is invariant to point permutation and rigid transformation.*

Proof. Let $\mathbf{X} \in \mathbb{R}^{N\times 3}$ be a 3D point cloud, $\mathbf{P} \in \mathbb{R}^{N\times N}$ a permutation matrix, $\mathbf{R} \in \mathbb{R}^{3\times 3}$ a rotation matrix, and $\mathbf{t} \in \mathbb{R}^3$ a translation vector. We define the transformed point cloud as $\tilde{\mathbf{X}} = \mathbf{R}\mathbf{X} + \mathbf{t}$.

Rigid Transformation Invariance: For any pair of points $\mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}$, their pairwise distance remains unchanged under rigid transformation:

$$\|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\| = \|(\mathbf{R}\mathbf{x}_i + \mathbf{t}) - (\mathbf{R}\mathbf{x}_j + \mathbf{t})\| = \|\mathbf{R}(\mathbf{x}_i - \mathbf{x}_j)\| = \|\mathbf{x}_i - \mathbf{x}_j\| \tag{8}$$

Thus, the weight matrix $\mathbf{W}$ remains invariant, leading to an identical Laplacian matrix.

Point Permutation Invariance: Let $\boldsymbol{\phi}$ be the eigenvector corresponding to the smallest non-zero eigenvalue λ of the Laplacian matrix $\mathbf{L}$. We assume that the multiplicity of λ is 1, ensuring that $\boldsymbol{\phi}$ is unique up to scaling (a standard assumption that holds almost surely for real-world, noisy data). For a permuted point cloud $\bar{\mathbf{X}} = \mathbf{P}\mathbf{X}$, with permutation matrix $\mathbf{P}$, the weight matrix transforms as $\bar{\mathbf{W}} = \mathbf{P}\mathbf{W}\mathbf{P}^{-1}$, resulting in a similarity-transformed Laplacian $\bar{\mathbf{L}} = \mathbf{P}\mathbf{L}\mathbf{P}^{-1}$. For the eigenvector $\boldsymbol{\phi}$ of $\mathbf{L}$ with eigenvalue λ, we have:

$$\bar{\mathbf{L}}(\mathbf{P}\boldsymbol{\phi}) = \mathbf{P}(\mathbf{L}\boldsymbol{\phi}) = \mathbf{P}(\lambda\boldsymbol{\phi}) = \lambda(\mathbf{P}\boldsymbol{\phi}) \tag{9}$$

Therefore, $(\mathbf{P}\boldsymbol{\phi})$ is an eigenvector of $\bar{\mathbf{L}}$ with eigenvalue λ. Since λ has multiplicity 1, eigenvectors for the original and permuted Laplacians differ only by the permutation $\mathbf{P}$ and scaling.

To establish a canonical ordering, we first normalize the eigenvector $\boldsymbol{\phi}$ so its largest-magnitude entry is positive, resolving sign ambiguity. We then permute the points according to these normalized eigenvector values, yielding a point ordering invariant to initial permutations.

These invariance properties ensure that our preprocessing pipeline produces consistent results regardless of the initial point ordering or orientation of the input point cloud. This theoretical guarantee enables our lightweight MLP architecture to focus on learning the geometric properties of the surface rather than accounting for permutation and transformation variations.

3.2 Training

This section outlines our geometric learning framework. We first present our synthetic data generation approach with controlled curvature properties. Then we describe our geometric feature extraction process and the design of our parameter-efficient network for surface classification.

Synthetic Data Generation. Surface curvature quantifies how a surface deviates from being flat at a point. The principal curvatures κ_1 and κ_2 represent the maximum and minimum bending of the surface. From these, we derive the Gaussian curvature $K = \kappa_1\kappa_2$ and the mean curvature $H = \kappa_1 + \kappa_2$. For our dataset, we generate quadratic surfaces with analytically tractable curvature properties:

$$z = f(x, y) = ax^2 + by^2 + cxy + dx + ey \tag{10}$$

Among possible sampling methods, we chose to sample coefficients (a, b, c, d, e), which proved sufficient to create surfaces with the full range of desired curvature characteristics. These surfaces are classified into one of four categories: $\mathcal{C} = \{\text{plane}, \text{parabolic}, \text{valley}, \text{saddle}\}$ based on its curvature signature at the origin.

To ensure an unbiased dataset, we sample each class separately with equal representation. For each generated surface, we uniformly sample points within the region $[-0.5, 0.5] \times [-0.5, 0.5]$ to create our synthetic dataset.

Feature Extraction and Network Design. We train a lightweight Multi-Layer Perceptron (MLP) in a supervised manner on synthetic point cloud data. The input consists of the preprocessed point cloud $\mathcal{P}$ augmented with second-order polynomial features of each point's coordinates (e.g., x^2, y^2, xy). These terms capture local curvature variations and encode higher-order geometric relationships, allowing the network to better distinguish surface classes with different curvature characteristics. The MLP is trained to classify each $\mathcal{P}$ into one of four fundamental surface categories:

1. **Plane:** A flat surface characterized by zero Gaussian and mean curvatures.
2. **Parabolic:** A convex surface with positive Gaussian curvature (e.g., spheres, domes).
3. **Valley:** A surface with zero Gaussian curvature and nonzero mean curvature (e.g., a cylinder).
4. **Saddle:** A hyperbolic surface with negative Gaussian curvature.

These four surface types represent fundamental geometric structures commonly encountered in real-world point cloud data. Their classification is critical for downstream tasks such as shape reconstruction and geometric reasoning. Fig. 3 provides visual illustrations of these surfaces, highlighting their distinct curvature properties.

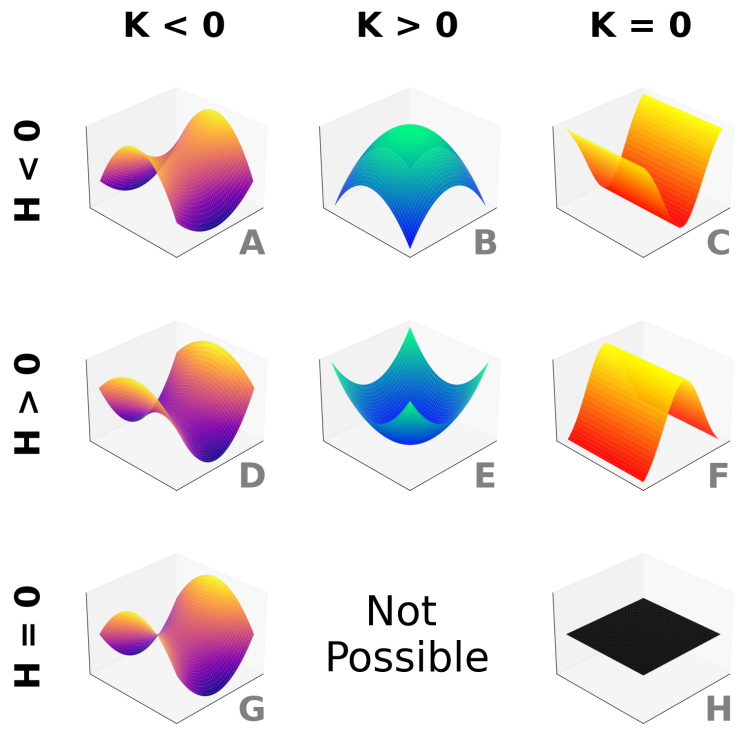

Fig. 3. The different possible surfaces given the Gaussian (**K**) and mean (**H**) curvature as described in Section 3.2. Note that up to rigid motion there are 4 different types of surfaces.

4 Experiments

We evaluate CanonNet on three tasks that capture essential aspects of local geometric understanding: ***curvature estimation***, ***Descriptor retrieval***, and ***surface classification***. While downstream tasks like classification or segmentation are important, these chosen experiments provide a more direct and rigorous assessment of an operator's ability to capture fine-grained surface properties. We conduct evaluations across three datasets and simulate real-world corruptions,

including **partial overlap**, **random rotations**, and **Gaussian noise**, to assess the model's resilience.

4.1 Gaussian and Mean Curvature Estimation

Experimental Setup. We evaluate CanonNet on the PCPNet dataset [17], which includes point clouds sampled from various 3D shapes with ground truth normals and curvatures. Unlike PCPNet and DeepFit, which are trained directly on this dataset, CanonNet is trained solely on synthetic quadratic surfaces representing local geometry (Section 3.2). Our model is a lightweight MLP with only 0.03M parameters and operates on local patches of just 20 points, ordered canonically.

Curvature Estimation Task. We estimate Gaussian curvature (K) and absolute mean curvature ($|H|$), which are invariant to normal orientation and better aligned with surface-based reasoning. CanonNet jointly predicts these curvatures along with the surface type (4 categories), encouraging deeper geometric understanding and aiding learning through multi-task supervision.

Comparison Baselines. We compare against PCPNet [17] and DeepFit [5]. PCPNet uses 500-point patches and 22M parameters; DeepFit uses 128-point patches and 3.5M parameters. In contrast, CanonNet uses only 20 points per patch and 0.03M parameters—roughly **700**× and **116**× smaller, respectively.

Evaluation Metric. We evaluate curvature estimation performance using the rectified error metric, which is calculated as:

$$D_K = \frac{|K - K_{GT}|}{\max\{|K_{GT}|, 1\}}, \quad D_H = \frac{|H - H_{GT}|}{\max\{|H_{GT}|, 1\}}, \tag{11}$$

where K and H are the predicted Gaussian and mean curvatures, and K_{GT} and H_{GT} are the ground truth values. The final error metrics are reported as the root mean square error (RMSE) of these normalized differences.

Results and Analysis. Table 1 presents the quantitative results. CanonNet achieves a mean curvature error of 0.40 on the PCPNet dataset—surpassing both PCPNet (1.91) and DeepFit (0.67)—despite being trained on synthetic data and using dramatically smaller patches. Its Gaussian curvature error is higher (8.2) but remains competitive considering the scale of the model and input.

On synthetic surfaces with analytical curvature, CanonNet achieves strong accuracy: 0.97 (Gaussian) and 0.14 (mean). These results demonstrate that our preprocessing and geometric learning pipeline effectively substitute for architectural complexity, enabling compact models to match or exceed the performance of much larger networks.

4.2 Geometric Descriptor Retrieval

Experimental Setup. Unlike other methods explicitly trained for descriptor matching, CanonNet is not trained as a geometric descriptor. Instead, we leverage its learned geometric understanding from curvature estimation and surface classification. Descriptors are formed by applying CanonNet to multi-resolution patches (via progressive downsampling) and concatenating the resulting features.

We evaluate CanonNet on the 3DMatch benchmark [43] using the Feature Match Recall (FMR) [8] metric, which measures the fraction of ground-truth correspondences correctly matched via mutual nearest neighbors in descriptor space.

Results and Analysis. Table 2 compares CanonNet to both traditional and learned descriptors. CanonNet achieves a competitive 65.7% FMR on unseen data, despite never being trained for this task. CanonNet is by far the smallest model: just 0.03Mb in size. This compactness results from a combination of our canonical preprocessing, which removes the need for complex architectures to handle point permutations or rigid transformations, and a geometric learning pipeline that effectively captures local surface structure.

Efficiency extends beyond model size. CanonNet processes only 300 points per patch (across resolutions), compared to 1000–2000 for others. Despite a lower inlier rate, it needs just 5 RANSAC iterations on average to find valid correspondences—offset by a $\sim 30\times$ speedup in descriptor generation over SpinNet on standard hardware.

4.3 Surface Classification

Experimental Setup. To further test CanonNet's robustness, we evaluate it on a synthetic surface classification task under randomized and noisy conditions. At test time, each surface instance is randomly sampled from one of several unseen surface shapes, then independently subjected to random point permutations, arbitrary 3D rotations, and additive Gaussian noise with varying magnitudes (0%–10%).

Results and Analysis. Fig. 4 shows the classification accuracy across noise levels. CanonNet achieves high accuracy on clean data (98%), and maintains strong performance under moderate noise—retaining 80% accuracy even at 10% noise. This demonstrates the model's effectiveness in extracting stable local geometry from deformed data, validating our canonical preprocessing and learning design.

References

1. Ao, S., Hu, Q., Yang, B., Markham, A., Guo, Y.: Spinnet: Learning a general surface descriptor for 3d point cloud registration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11753–11762 (2021)

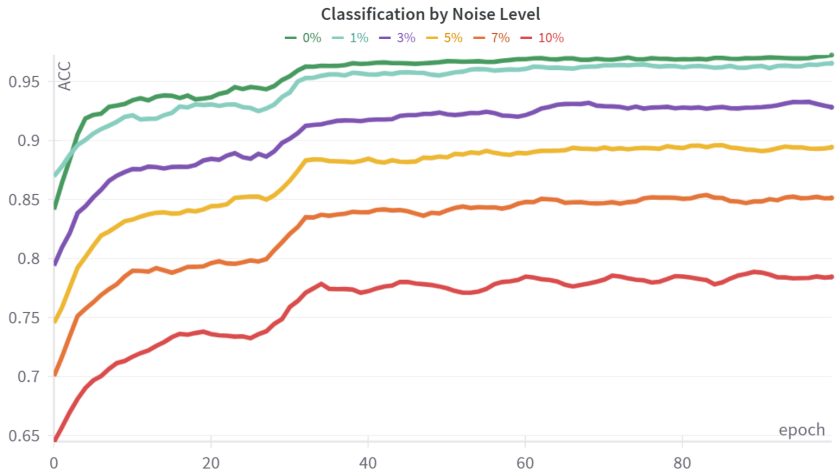

Fig. 4. Noise effect on surface classification.

2. Bai, X., Luo, Z., Zhou, L., Fu, H., Quan, L., Tai, C.L.: D3feat: Joint learning of dense detection and description of 3d local features. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6359–6367 (2020)
3. Behler, J., Parrinello, M.: Generalized neural-network representation of high-dimensional potential-energy surfaces. Physical review letters **98**(14), 146401 (2007)
4. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation **15**(6), 1373–1396 (2003)
5. Ben-Shabat, Y., Gould, S.: Deepfit: 3d surface fitting via neural network weighted least squares. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16. pp. 20–34. Springer (2020)
6. Choy, C., Park, J., Koltun, V.: Fully convolutional geometric features. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 8958–8966 (2019)
7. Chung, F.R.: Spectral graph theory, vol. 92. American Mathematical Soc. (1997)
8. Deng, H., Birdal, T., Ilic, S.: Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In: Proceedings of the European conference on computer vision (ECCV). pp. 602–618 (2018)
9. Deng, H., Birdal, T., Ilic, S.: Ppfnet: Global context aware local features for robust 3d point matching. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 195–205 (2018)
10. Dovrat, O., Lang, I., Avidan, S.: Learning to sample. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2760–2769 (2019)
11. Duvenaud, D.K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints. Advances in neural information processing systems **28** (2015)

Table 1. Curvature errors and efficiency (Section 4.1). CanonNet is small, with SOTA results.

PCPNET dataset				
Method	$D_K \downarrow$	$D_H \downarrow$	#Params (M)	#Points
PCPNET [17]	6.88	1.91	22	500
DeepFit [5]	0.56	0.67	3.5	128
CanonNet	8.2	**0.4**	**0.03**	**20**
Synthetic dataset				
CanonNet	0.97	**0.14**	**0.03**	**20**

12. Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks. Journal of Machine Learning Research **24**(43), 1–48 (2023)
13. Fiedler, M.: Algebraic connectivity of graphs. Czechoslovak mathematical journal **23**(2), 298–305 (1973)
14. Fiedler, M.: A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. Czechoslovak mathematical journal **25**(4), 619–633 (1975)
15. Gojcic, Z., Zhou, C., Wegner, J.D., Wieser, A.: The perfect match: 3d point cloud matching with smoothed densities. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5545–5554 (2019)
16. Grötschla, F., Xie, J., Wattenhofer, R.: Benchmarking positional encodings for gnns and graph transformers. arXiv preprint arXiv:2411.12732 (2024)
17. Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N.J.: Pcpnet learning local shape properties from raw point clouds. In: Computer graphics forum. vol. 37, pp. 75–85. Wiley Online Library (2018)
18. Guo, M.H., Cai, J.X., Liu, Z.N., Mu, T.J., Martin, R.R., Hu, S.M.: Pct: Point cloud transformer. Computational Visual Media **7**, 187–199 (2021)
19. de Haan, P., Cohen, T.S., Welling, M.: Natural graph networks. Advances in neural information processing systems **33**, 3636–3646 (2020)
20. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. Advances in neural information processing systems **28** (2015)
21. Joshi, C.K., Bodnar, C., Mathis, S.V., Cohen, T., Lio, P.: On the expressive power of geometric graph neural networks. In: International conference on machine learning. pp. 15330–15355. PMLR (2023)
22. Khoury, M., Zhou, Q.Y., Koltun, V.: Learning compact geometric features. In: Proceedings of the IEEE international conference on computer vision. pp. 153–161 (2017)
23. Kondor, R., Son, H.T., Pan, H., Anderson, B., Trivedi, S.: Covariant compositional networks for learning graphs. arXiv preprint arXiv:1801.02144 (2018)

Table 2. Parameter count and FMR (Section 4.2). CanonNet is compact yet competitive.

Method	Param. (Mb)	In-Domain (%)	Cross-Domain (%)
FPFH [35]	-	35.9	22.1
SHOT [39]	-	23.8	61.1
3DMatch [43]	13.40	59.6	16.9
CGF [22]	1.86	58.2	20.2
PerfectMatch [15]	3.26	94.7	79.0
FCGF [6]	33.48	95.2	16.1
D3Feat (rand) [2]	13.42	95.3	26.2
LMVD [26]	2.66	97.5	79.9
SpinNet [1]	2.16	97.6	92.8
CanonNet	**0.03**	-	65.7

24. Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., Tossou, P.: Rethinking graph transformers with spectral attention. Advances in Neural Information Processing Systems **34**, 21618–21629 (2021)
25. Lai, X., Liu, J., Jiang, L., Wang, L., Zhao, H., Liu, S., Qi, X., Jia, J.: Stratified transformer for 3d point cloud segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8500–8509 (2022)
26. Li, L., Zhu, S., Fu, H., Tan, P., Tai, C.L.: End-to-end learning local multi-view descriptors for 3d point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1919–1928 (2020)
27. Li, Y., Ma, L., Zhong, Z., Liu, F., Chapman, M.A., Cao, D., Li, J.: Deep learning for lidar point clouds in autonomous driving: A review. IEEE Transactions on Neural Networks and Learning Systems **32**(8), 3412–3432 (2020)
28. Maskey, S., Parviz, A., Thiessen, M., Stärk, H., Sadikaj, Y., Maron, H.: Generalized laplacian positional encoding for graph representation learning. arXiv preprint arXiv:2210.15956 (2022)
29. Pan, X., Xia, Z., Song, S., Li, L.E., Huang, G.: 3d object detection with pointformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7463–7472 (2021)
30. Pomerleau, F., Colas, F., Siegwart, R., et al.: A review of point cloud registration algorithms for mobile robotics. Foundations and Trends® in Robotics **4**(1), 1–104 (2015)
31. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
32. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems **30** (2017)
33. Qin, Z., Yu, H., Wang, C., Guo, Y., Peng, Y., Xu, K.: Geometric transformer for fast and robust point cloud registration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11143–11152 (2022)

34. Ran, H., Liu, J., Wang, C.: Surface representation for point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 18942–18952 (2022)
35. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (fpfh) for 3d registration. In: 2009 IEEE international conference on robotics and automation. pp. 3212–3217. IEEE (2009)
36. Rusu, R.B., Marton, Z.C., Blodow, N., Beetz, M.: Persistent point feature histograms for 3d point clouds. In: Intelligent Autonomous Systems 10, pp. 119–128. IOS Press (2008)
37. Sitek, A., Huesman, R.H., Gullberg, G.T.: Tomographic reconstruction using an adaptive tetrahedral mesh defined by a point cloud. IEEE Transactions on medical imaging **25**(9), 1172–1179 (2006)
38. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6411–6420 (2019)
39. Tombari, F., Salti, S., Di Stefano, L.: Unique signatures of histograms for local surface description. In: Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part III 11. pp. 356–369. Springer (2010)
40. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (tog) **38**(5), 1–12 (2019)
41. Yang, P., Snoek, C.G., Asano, Y.M.: Self-ordering point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15813–15822 (2023)
42. Yuan, Y., Wu, Y., Fan, X., Gong, M., Ma, W., Miao, Q.: Egst: Enhanced geometric structure transformer for point cloud registration. IEEE transactions on visualization and computer graphics **30**(9), 6222–6234 (2023)
43. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.: 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1802–1811 (2017)
44. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 16259–16268 (2021)
45. Zheng, T., Chen, C., Yuan, J., Li, B., Ren, K.: Pointcloud saliency maps. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1598–1606 (2019)