GeoNet: Deep Geodesic Networks for Point Cloud Analysis

Tong He[1], Haibin Huang[2,†], Li Yi[3], Yuqian Zhou[4], Chihao Wu[2], Jue Wang[2], Stefano Soatto[1]
[1]UCLA [2]Megvii (Face++) [3]Stanford [4]UIUC [†]Corresponding Author

Abstract

Surface-based geodesic topology provides strong cues for object semantic analysis and geometric modeling. However, such connectivity information is lost in point clouds. Thus we introduce GeoNet, the first deep learning architecture trained to model the intrinsic structure of surfaces represented as point clouds. To demonstrate the applicability of learned geodesic-aware representations, we propose fusion schemes which use GeoNet in conjunction with other baseline or backbone networks, such as PU-Net and PointNet++, for down-stream point cloud analysis. Our method improves the state-of-the-art on multiple representative tasks that can benefit from understandings of the underlying surface topology, including point upsampling, normal estimation, mesh reconstruction and non-rigid shape classification.

1. Introduction

Determining neighborhood relationship among points in a point cloud, known as topology estimation, is an important problem since it indicates the underlying point cloud structure, which could further reveal the point cloud semantics and functionality. Consider the red inset in Fig. 1: the two clusters of points, though seemingly disconnected, should indeed be connected to form a chair leg, which supports the whole chair. On the other hand, the points on opposite sides of a chair seat, though spatially very close to each other, should not be connected to avoid confusing the sittable upper surface with the unsittable lower side. Determining such topology appears to be a very low-level endeavor but in reality it requires global, high-level knowledge, making it a very challenging task. Still, from the red inset in Fig. 1, we could draw the conclusion that the two stumps are connected only after we learn statistical regularities from a large number of point clouds and observe many objects of this type with connected elongated vertical elements extending from the body to the ground. This motivates us to adopt a learning approach to capture the topological structure within point clouds.

Our primary goals in this paper are to develop representations of point cloud data that are informed by the underlying surface topology as well as object geometry, and propose methods that leverage the learned topological features for geodesic-aware point cloud analysis. The representation should capture various topological patterns of a point cloud and the method of leveraging these geodesic features should not alter the data stream, so our representation can be learned jointly and used in conjunction with the state-of-the-art baseline or backbone models (e.g. PU-Net, PointNet++ [45, 32, 33, 7]) that feed the raw data through, with no information loss to further stages of processing.

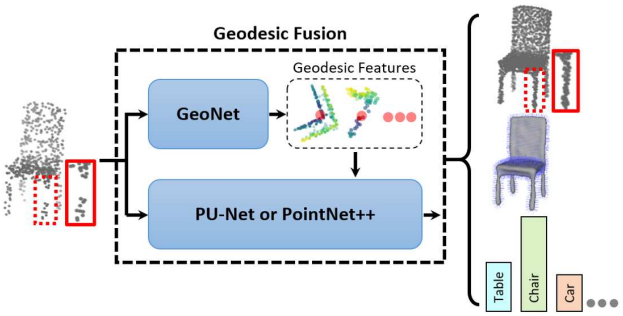

Figure 1. Our method takes a point cloud as input, and outputs representations used for multiple tasks including upsampling, normal estimation, mesh reconstruction, and shape classification.

For the first goal, we propose a geodesic neighborhood estimation network (GeoNet) to learn deep geodesic representations using the ground truth geodesic distance as supervision signals. As illustrated in Fig. 2, GeoNet consists of two modules: an autoencoder that extracts a feature vector for each point and a geodesic matching (GM) layer that acts as a learned kernel function for estimating geodesic neighborhoods using the latent features. Due to the supervised geodesic training process, intermediates features of the GM layer contain rich information of the point cloud topology and intrinsic surface attributes. We note that the representation, while trained on geodesic distances, does not by construction produce geodesics (e.g. symmetry, triangle inequality, etc.). The goal of the representation is to inform subsequent stages of processing of the global geometry and topology, and is not to conduct metric computations directly.

For the second task, as shown in Fig. 3, we propose geodesic fusion schemes to integrate GeoNet into the state-

of-the-art network architectures designed for different tasks. Specifically, we present PU-Net fusion (PUF) for point cloud upsampling, and PointNet++ fusion (POF) for normal estimation, mesh reconstruction as well as non-rigid shape classification. Through experiments, we demonstrate that the learned geodesic representations from GeoNet are beneficial for both geometric and semantic point cloud analyses.

In summary, in this paper we propose an approach for learning deep geodesic-aware representations from point clouds and leverage the results for various point set analyses. Our contributions are:

- We present, to the best of our knowledge, the first deep learning method, GeoNet, that ingests point clouds and learns representations which are informed by the intrinsic structure of the underlying point set surfaces.
- To demonstrate the applicability of learned geodesic representations, we develop network fusion architectures that incorporate GeoNet with baseline or backbone networks for geodesic-aware point set analysis.
- Our geodesic fusion methods are benchmarked on multiple geometric and semantic point set tasks using standard datasets and outperform the state-of-the-art methods.

2. Related work

We assume our input is a point cloud, which can be obtained by multiple-view geometry, single image depth estimation [29, 20, 15, 38, 12] or collected from various sensors (e.g. depth camera, Lidar, etc.). We mainly review traditional graph-based methods for geodesic distance computation, as well as general works on point cloud upsampling, normal estimation, and non-rigid shape classification, as we are unaware of other prior works on point cloud-based deep geodesic representation learning.

Geodesic distance computation. There are two types of methods: some allow the path to traverse mesh faces [35, 31, 6, 14, 37, 41, 8] for accurate geodesic distance computation, while others find approximate solutions via shortest path algorithms constrained on graph edges [11, 13, 23]. For the first type, an early method [35] suggests a polynomial algorithm of time $O(n^3 logn)$ where n is the number of edges, but their method is restricted to a convex polytope. Based on Dijkstra's algorithm [11], [31] improves the time complexity to $O(n^2 logn)$ and extends the method to an arbitrary polyhedral surface. Later, [6] proposes an $O(n^2)$ approach using a set of windows on the polyhedron edges to encode the structure of the shortest path set. By filtering out useless windows, [41] further speeds up the algorithm. Then [8] introduces a heat method via solving a pair of standard linear elliptic problems. As for graph edge-based methods, typical solutions include Dijkstra's [11], Floyd-Warshall [13] and Johnson's algorithms [23], which have much lower time complexity than the surface traversing methods. For a 20000-vertex mesh, computing its all-pair geodesic distances can take several days using [41] while [23] only uses about 1 minute on CPU. When a mesh is dense, the edge-constrained shortest path methods generate low-error geodesic estimates. Thus in our work, we apply [23] to compute the ground truth geodesic distance.

Point upsampling. Previous methods can be summarized into two categories. i) Optimization based methods [1, 27, 22], championed by [1], which interpolates a dense point set from vertices of a Voronoi diagram in the local tangent space. Then [27] proposes a locally optimal projection (LOP) operator for point cloud resampling and mesh reconstruction leveraging an L_1 median. For improving robustness to point cloud density variations, [22] presents a weighted LOP. These methods all make strong assumptions, such as surface smoothness, and are not data-driven, and therefore have limited applications in practice. ii) Deep learning based methods. To apply the (graph) convolution operation, many of those methods first voxelize a point cloud into regular volumetric grids [40, 39, 19, 9] or instead use a mesh [10, 44]. While voxelization introduces discretization artifacts and generates low resolution voxels for computational efficiency, mesh data can not be trivially reconstructed from a sparse and noisy point cloud. In [43, 42] a sparse point cloud is reprojected onto a range map and modeled as a 2.5D inpanting problem. To directly upsample a 3D point cloud, PU-Net [45] learns multilevel features for each point and expands the point set via a multi-branch convolution unit implicitly in feature space. But PU-Net is based on Euclidean space and thus does not leverage the underlying point cloud surface attributes in geodesic space, which we show in this paper are important for upsampling.

Normal estimation. A widely used method for point cloud normal estimation is to analyze the variance in a tangential plane of a point and find the minimal variance direction by Principal Component Analysis (PCA) [21, 24]. But this method is sensitive to the choice of the neighborhood size, namely, large regions can cause over-smoothed surfaces and small ones are sensitive to noises. To improve robustness, [16, 4, 2] propose to fit higher-order shapes. However, methods described above all require careful parameter tuning at the inference time and only estimate normal orientation up to sign. Thus, so far robust estimation for oriented normal vectors using traditional methods is still challenging, especially across different noise levels and shape structures. There are only few data-driven methods that are able to integrate normal estimation and orientation alignment into a unified pipeline [17, 33]. They take a point cloud as input and directly regress oriented normal vectors, but these methods are not designed to learn geodesic topology-based

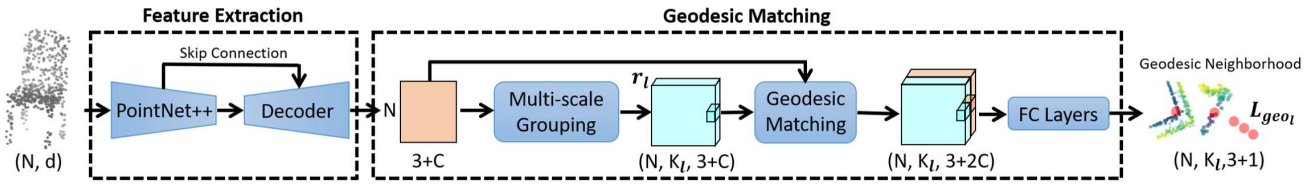

Figure 2. GeoNet: geodesic neighborhood estimation network.

representations that capture the intrinsic surface features for better normal estimation.

Non-rigid shape classification. Classifying the point cloud of non-rigid objects often consists of two steps: extracting intrinsic features in geodesic space and applying a classifier (e.g. SVM, MLP, etc.). Some commonly used features include wave kernel signatures [3], heat kernel signatures [36], spectral graph wavelet signatures [30], ShapeDNA [34], etc. For example, DeepGM [28] uses geodesic moments and stacked sparse autoencoders to classify non-rigid shapes, such as cat, horse, spider, etc. The geodesic moments are feature vectors derived from the integral of the geodesic distance on a shape, while stacked sparse autoencoders are deep neural networks consisting of multiple layers of sparse autoencoders. However, the above methods all require knowing graph-based data, which is not available from widely used sensors (e.g. depth camera, Lidar, etc.) for 3D data acquisition. Though PointNet++ [33] is able to directly ingest a point cloud and conduct classification, it is not designed to model the geodesic topology of non-rigid shapes and thus its performance is inferior to traditional two-step methods which heavily reply on the offline computed intrinsic surface features.

3. Method

3.1. Problem Statement

$\chi = \{x_i\}$ denotes a point set with $x_i \in \mathbb{R}^d$ and $i = 1, \ldots, N$. Although the problem and the method developed are general, we focus on the case $d = 3$ using only Euclidean coordinates as input. A neighborhood subset within radius r from a point x_i is denoted $B_r(x_i) = \{x_j | d_E(x_i, x_j) \leq r\}$ where $d_E(x_i, x_j) \in \mathbb{R}$ is the Euclidean (embedding) distance between x_i and x_j. The cardinality of $B_r(x_i)$ is K. The corresponding geodesic distance set around x_i is called $G_r(x_i) = \{g_{ij} = d_G(x_i, x_j) | x_j \in B_r(x_i)\}$ where $d_G \in \mathbb{R}$ means the geodesic distance. Our goal is to learn a function $f : x_i \mapsto G_r(x_i)$ that maps each point to (an approximation of) the geodesic distance set $G_r(x_i)$ around it.

3.2. Method

We introduce GeoNet, a network trained to learn the function f defined above. It consists of an autoencoder with skip connections, followed by a multi-scale Geodesic Matching (GM) layer, leveraging latent space features $\{\psi(x_i)\} \subseteq \mathbb{R}^{3+C}$ of the point set. GeoNet is trained in a supervised manner using ground truth geodesic distances between points in the set χ. To demonstrate the applicability of learned deep geodesic-aware representations from GeoNet, we test our approach on typical tasks that require understandings of the underlying surface topology, including point cloud upsampling, surface normal estimation, mesh reconstruction, and non-rigid shape classification. To this end, we leverage the existing state-of-the-art network architectures designed for the aforementioned problems. Specifically, we choose PU-Net as the baseline network for point upsampling and PointNet++ for other tasks. The proposed geodesic fusion methods, called PU-Net fusion (PUF) and PointNet++ fusion (POF), integrate GeoNet with the baseline or backbone models to conduct geodesic-aware point set analysis.

3.3. Geodesic Neighborhood Estimation

As illustrated in Fig. 2, GeoNet consists of two modules: an autoencoder that extracts a feature vector $\psi(x_i)$ for each point $x_i \in \chi$ and a GM layer that acts as a learned geodesic kernel function for estimating $G_r(x_i)$ using the latent features.

Feature Extraction. We use a variant of PointNet++, which is a point set based hierarchical and multi-scale function, for feature extraction. It maps an input point set χ to a feature set $\{\varphi(x_i) | x_i \in \widetilde{\chi}\}$ where $\varphi(x_i) \in \mathbb{R}^{3+\widetilde{C}}$ is a concatenation of the xyz coordinates and the $\widetilde{C}$ dimensional embedding of x_i, and $\widetilde{\chi}$ is a sampled subset of χ by farthest-point sampling. To recover features $\{\psi(x_i)\}$ for the point cloud χ, we use a decoder with skip connections. The decoder consists of recursively applied tri-linear feature interpolators, shared fully connected (FC) layers, ReLU and Batch Normalization. The resulting $(N, 3 + C)$ tensor is then fed into the GM layer for geodesic neighborhood estimation.

Geodesic Matching. We group the latent features $\psi(x_i)$ into neighborhood feature sets $F_{r_l}(x_i) = \{\psi(x_j) | x_j \in B_{r_l}(x_i)\}$, under multiple radius scales r_l. At each scale r_l we set a maximum number of neighborhood points K_l, and thus produce a tensor of dimension $(N, K_l, 3+C)$. The grouped features, together with the latent features, are sent to a geodesic matching module, where $\psi(x_i)$ is concatenated with $\psi(x_j)$ for every $x_j \in B_{r_l}(x_i)$. The resulting

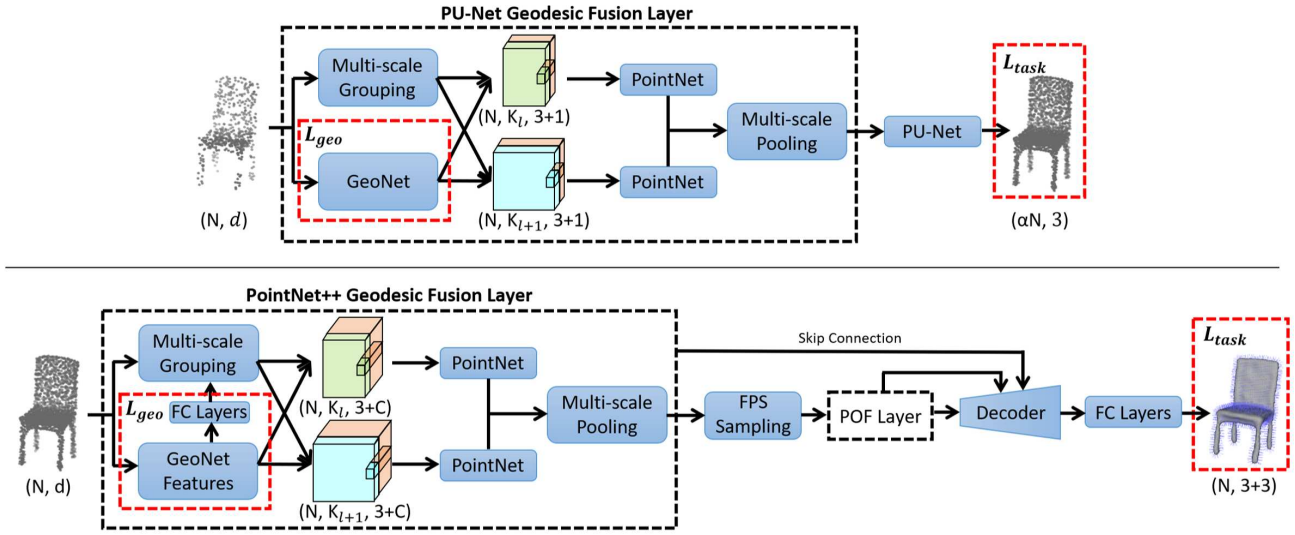

Figure 3. PU-Net (top) and PointNet++ (bottom) geodesic fusion architectures.

feature $\xi_{ij} \in \mathbb{R}^{3+2C}$ becomes the input to a set of shared FC layers with ReLU, Batch Normalization and Dropout. As demonstrated in [18], the multilayer perceptron (MLP) acts as a kernel function that maps ξ_{ij} to an approximation of the geodesic distance, $\hat{g}_{ij}$. Finally, the GM layer yields $G_{r_l}(x_i)$ for each point of the input point cloud χ. We use a multi-scale L_1 loss $L_{geo} = \sum_l L_{geo_l}$ to compare the ground truth geodesic distances to their estimates:

$$L_{geo_l} = \sum_{x_i \in \chi} \sum_{x_j \in B_{r_l}(x_i)} \frac{|g_{ij} - \hat{g}(x_i, x_j)|}{NK_l} \tag{1}$$

3.4. Geodesic Fusion

To demonstrate how the learned geodesic representations can be used for point set analysis, we propose fusion methods based on the state-of-the-art (SOTA) network architectures for different tasks. For example, PU-Net is the SOTA upsampling method and thus we propose PUF that uses PU-Net as the baseline network to conduct geodesic fusion for point cloud upsampling. With connectivity information provided by the estimated geodesic neighborhoods, our geodesic-fused upsampling network can better recover topological details, such as curves and sharp structures, than PU-Net. We also present POF leveraging PointNet++ as the fusion backbone, and demonstrate its effectiveness on both geometric and semantic tasks where PointNet++ shows the state-of-the-art performance.

PU-Net Geodesic Fusion. A PUF layer, as illustrated in Fig. 3 (top), takes a (N, d) point set as input and sends it into two branches: one is a multi-scale Euclidean grouping layer, and the other is GeoNet. At each neighborhood scale r_l, the grouped point set $B_{r_l}(x_i)$ is fused with the estimated geodesic neighborhood $G_{r_l}(x_i)$ to yield $S_{r_l}(x_i) = \{(x_j, g_{ij}) | x_j \in B_{r_l}(x_i)\}$ with $(x_j, g_{ij}) \in \mathbb{R}^{d+1}$. Then the $(N, K_l, d+1)$ fused tensor is fed to a PointNet to generate a (N, C_l) feature tensor which will be stacked with features from other neighborhood scales. The remaining layers are from PU-Net. As indicated by the red rectangles in Fig. 3, the total loss has two weighted terms:

$$L = L_{geo} + \lambda L_{task} \tag{2}$$

where L_{geo} is for GeoNet training (1), λ is a weight and L_{task}, in general, is the loss for the current task that we are targeting. In this case, the goal is point cloud upsampling: $L_{task} = L_{up}(\theta)$ where θ indicates network parameters. PUF upsampling takes a randomly distributed sparse point set χ as input and generates a uniformly distributed dense point cloud $\hat{P} \subseteq \mathbb{R}^3$. The upsampling factor is $\alpha = \frac{|P|}{|\chi|}$:

$$L_{up}(\theta) = L_{EMD}(P, \hat{P}) + \lambda_1 L_{rep}(\hat{P}) + \lambda_2 \|\theta\|^2 \tag{3}$$

in which the first term is the Earth Mover Distance (EMD) between the upsampled point set $\hat{P}$ and the ground truth dense point cloud P:

$$L_{EMD}(P, \hat{P}) = \min_{\phi: \hat{P} \to P} \sum_{p_i \in \hat{P}} \|p_i - \phi(p_i)\|^2 \tag{4}$$

where $\phi : \hat{P} \to P$ indicates a bijection mapping.

The second term in (3) is a repulsion loss which promotes a uniform spatial distribution for $\hat{P}$ by penalizing close point pairs:

$$L_{rep}(\hat{P}) = \sum_{p_i \in \hat{P}} \sum_{p_j \in \widetilde{P}_i} \eta(\|p_i - p_j\|) \omega(\|p_i - p_j\|) \tag{5}$$

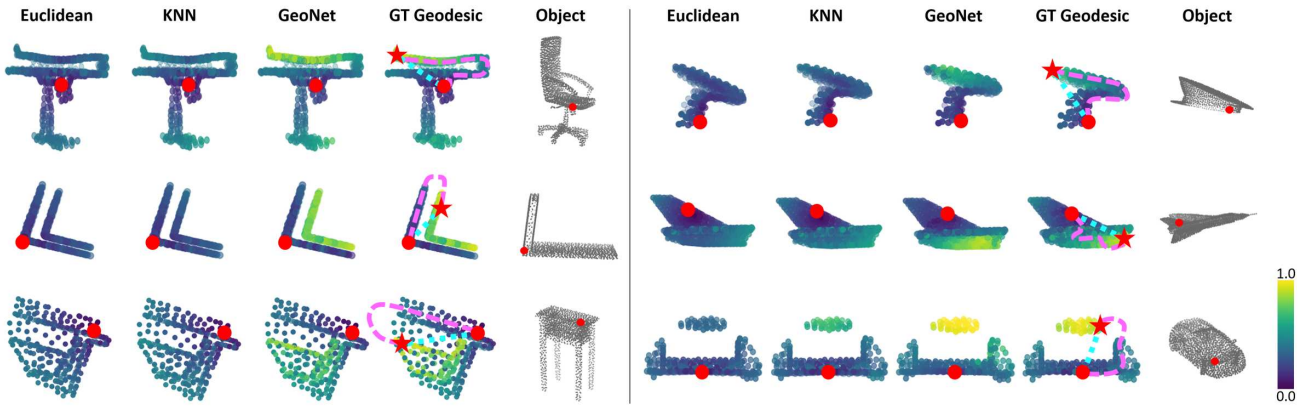

Figure 4. Representative results of geodesic neighborhood estimation. Red dots indicate the reference point and stars represent target points selected for the purpose of illustration. Points in dark-purple are closer to the reference point than those in bright-yellow. Shortest paths between the reference point and the target point in euclidean space are colored in sky-blue. Topology-based geodesic paths are in pink.

where $\widetilde{P}_i$ is a set of k-nearest neighbors of p_i, $\eta(r) = -r$ penalizes close pairs (p_i, p_j), and $\omega(r) = e^{-r^2/h^2}$ is a fast-decaying weight function with some constant h [22, 27].

PointNet++ Geodesic Fusion. Fig. 3 (bottom) illustrates the PointNet++ based fusion pipeline. Due to task as well as architecture differences between PU-Net and PointNet++, we make following changes to PUF to design a suitable fusion strategy that leverages PointNet++. First, for multi-scale grouping, we use the learned geodesic neighborhoods $\hat{G}_r(x_i)$ instead of Euclidean ones. Geodesic grouping brings attention to the underlying surfaces as well as structures of the point cloud. Second, while the PUF layer fuses estimated $\hat{G}_r(x_i) = \{\hat{g}_{ij} = \hat{d}_G(x_i, x_j) | x_j \in B_r(x_i)\}$, where $\hat{g}_{ij} \in \mathbb{R}$, of each neighborhood point set $B_r(x_i)$ into the backbone network, the POF layer uses the latent geodesic-aware features $\widetilde{\xi}_{ij} \in \mathbb{R}^{\widetilde{C}}$ extracted from the second-to-last FC layer in GeoNet. Namely, $\widetilde{\xi}_{ij}$ is an intermediate high-dimensional feature vector from ξ_{ij} to $\hat{g}_{ij}$ via FC layers, and therefore it is better informed of the intrinsic point cloud topology. Third, in PointNet++ fusion we apply the POF layer in a hierarchical manner, leveraging farthest-point sampling. Thus, the learned features encode both local and global structural information of the point set. The total loss for POF also has two parts: One is for GeoNet training and the other is for the task-at-hand. We experiment on representative tasks that can benefit from understandings of the topological surface attributes. We use the L_1 error for point cloud normal estimation:

$$L_{normal} = \sum_{x_i \in \chi} \sum_{j=1}^{3} \frac{\left| n_i^{(j)} - \hat{n}(x_i)^{(j)} \right|}{3N} \tag{6}$$

in which $n_i \in \mathbb{R}^3$ is the ground truth unit normal vector of x_i, and $\hat{n}(x_i) \in \mathbb{R}^3$ is the estimated normal. We then use the normal estimation to generate mesh via Poisson surface reconstruction [25]. To classify point clouds of non-rigid objects, we use cross-entropy loss:

$$L_{cls} = -\sum_{c=1}^{S} y_c log(p_c(\chi)) \tag{7}$$

		K-3	*K-6*	*K-12*	*Euc*	GeoNet
v1	r $\leqslant$ 0.1	8.75	8.97	9.04	9.06	**5.67**
	r $\leqslant$ 0.2	16.22	17.33	17.90	18.16	**9.25**
	r $\leqslant$ 0.4	15.15	16.80	17.88	18.95	**9.75**
v2	r $\leqslant$ 0.1	11.71	11.49	11.55	11.57	**7.06**
	r $\leqslant$ 0.2	19.22	17.76	18.28	18.56	**9.74**
	r $\leqslant$ 0.4	21.03	17.19	18.20	19.44	**10.04**
v3	r $\leqslant$ 0.1	13.28	14.23	14.62	14.78	**10.86**
	r $\leqslant$ 0.2	14.85	17.27	18.54	19.49	**13.61**
	r $\leqslant$ 0.4	**13.48**	16.10	17.72	19.68	14.73

Table 1. Neighborhood geodesic distance estimation MSE (x100) on the heldout ShapeNet training-category samples. We compare with *KNN-Graph* based shortest path methods under different choices of K values. *Euc* represents the difference between Euclidean distance and geodesic distance. MSE(s) are reported under multiple radius ranges r. **v1** takes uniformly distributed point sets with 512 points as input, and **v2** uses randomly distributed point clouds. **v3** is tested using point clouds that have 2048 uniformly distributed points.

		K-3	*K-6*	*K-12*	*Euc*	GeoNet
v1	r $\leqslant$ 0.1	8.81	9.01	9.05	9.06	**7.52**
	r $\leqslant$ 0.2	11.84	12.88	13.49	13.75	**11.44**
v2	r $\leqslant$ 0.1	10.52	10.21	10.25	10.26	**8.94**
	r $\leqslant$ 0.2	15.02	12.99	13.59	13.86	**11.69**
v3	r $\leqslant$ 0.1	11.82	12.39	12.65	12.75	**10.88**
	r $\leqslant$ 0.2	**11.80**	12.84	13.55	14.50	12.26

Table 2. Geodesic neighborhood estimation MSE (x100) on the leftout ShapeNet categories. **v1** takes uniformly distributed point sets with 512 points as input, and **v2** uses randomly distributed point clouds. **v3** is tested using point clouds that have 2048 uniformly distributed points.

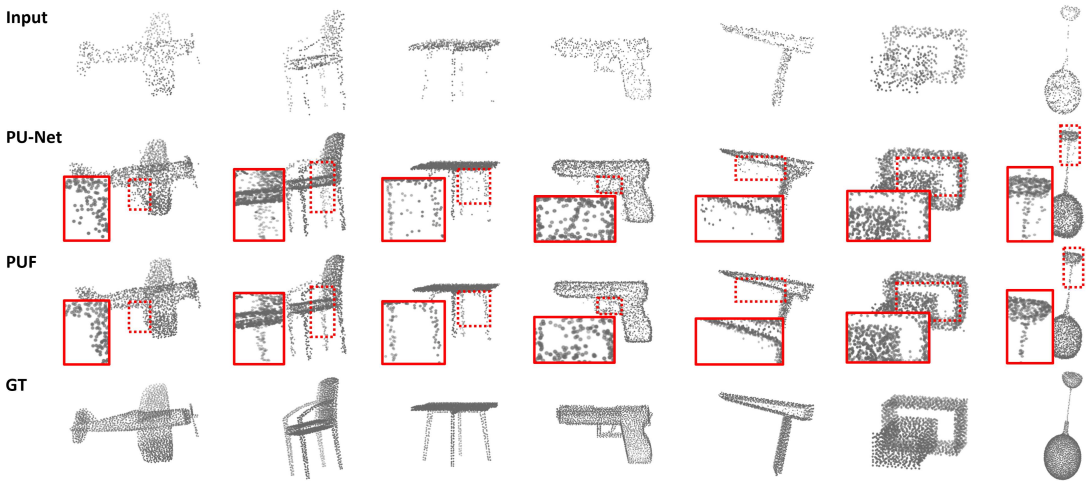

Figure 5. Point cloud upsampling comparisons with PU-Net. The input point clouds have 512 points with random distributions and the upsampled point clouds have 2048 points. Red insets show details of the corresponding dashed region in the reconstruction.

where S is the number of non-rigid object categories, and c is class label; $y_c \in \{0, 1\}$ is a binary indicator, which takes value 1 if class label c is ground truth for the input point set. $p_c(\chi) \in \mathbb{R}$ is the predicted probability w.r.t. class c of the input point set.

3.5. Implementation

For GeoNet training, the multiscale loss L_{geo_l} is enforced at three radius ranges: 0.1, 0.2 and 0.4. We use Adam [26] with learning rate 0.001 and batchsize 3 for 8 epochs. To train the geodesic fusion networks, we set the task term weight λ as 1, and use Adam with learning rate 0.0001 and batchsize 2 for around 300 to 1500 epochs depending on the task and the dataset. Source code in Tensorflow will be made available upon completion of the anonymous review process.

4. Experiments

We put GeoNet to the test by estimating point cloud geodesic neighborhoods. To demonstrate the applicability of learned deep geodesic-aware representations, we also conduct experiments on down-stream point cloud tasks such as point upsampling, normal estimation, mesh reconstruction and non-rigid shape classification.

4.1. Geodesic Neighborhood Estimation

In Tab. 1 (**v1**) we show geodesic distance set, $G_r(x_i)$, estimation results on the ShapeNet dataset [5] using point clouds with 512 uniformly distributed points. Mean-squared errors (MSE) are reported under multiple radius scales r w.r.t. $x_i \in \chi$. GeoNet demonstrates consistent improvement over the baselines. Representative results are visualized in Fig. 4. Our method captures various topological patterns, such as curved surfaces, layered structures, inner/outer parts, etc.

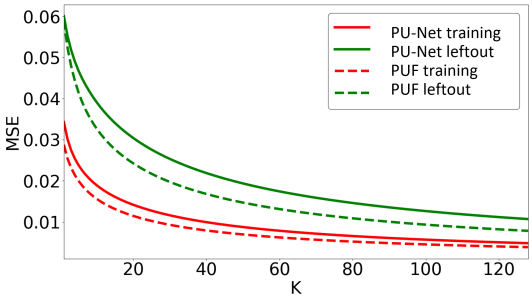

Figure 6. Top-k mean square error (MSE) of upsampled points that have large errors, for both the heldout training-category samples (red) and the leftout ShapeNet categories (green).

		MSE	EMD	CD
Training	PU-Net	7.14	8.06	2.72
	PUF	**6.23**	**7.62**	**2.46**
Leftout	PU-Net	12.38	11.43	3.98
	PUF	**9.55**	**8.90**	**3.27**

Table 3. Point cloud upsampling results on both the heldout training-category samples and the unseen ShapeNet categories. MSE(s) (x10000) are scaled for better visualization.

Generality. We test GeoNet's robustness under different point set distributions and sizes. In Tab. 1 (**v2**) we use point clouds with 512 randomly distributed points as input. We also test on dense point sets that contain 2048 uniformly distributed points in Tab. 1 (**v3**). Our results are robust to different point set distributions as well as sizes. To show the generalization performance, in Tab. 2 we report results on the leftout ShapeNet categories. Our method performs better on unseen categories, while *KNN-Graph* based shortest path approaches suffer from point set distribution randomness, density changes and unsuitable choices of *K* values.

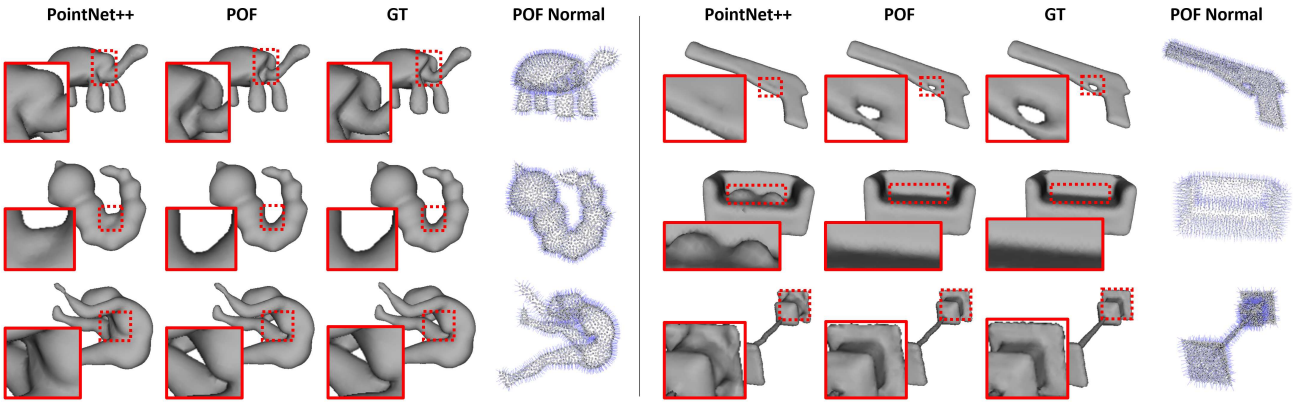

Figure 7. Mesh reconstruction results on the Shrec15 (left) and the ShapeNet (right) datasets using the estimated normal by PointNet++ and our method POF. GT presents mesh reconstructed via the ground truth normal. We also visualize POF normal estimation in the fourth and the last columns.

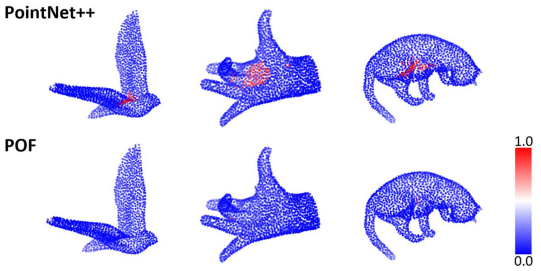

Figure 8. Point set normal estimation errors. Blue indicates small errors and red is for large ones.

	$\leqslant 2.5^\circ$	$\leqslant 5^\circ$	$\leqslant 10^\circ$	$\leqslant 15^\circ$
PCA	6.16±0.01	14.85±0.02	27.16±0.17	34.17±0.28
PointNet++	12.81±0.18	33.37±0.92	61.58±2.02	75.49±1.95
POF	**16.26**±0.30	**39.02**±1.09	**66.98**±1.46	**79.66**±1.21

Table 4. Point cloud normal estimation accuracy (%) on the Shrec15 dataset under multiple angle thresholds.

		$\leqslant 2.5^\circ$	$\leqslant 5^\circ$	$\leqslant 10^\circ$	$\leqslant 15^\circ$
Training	PCA	5.33	10.11	18.52	24.82
	PointNet++	30.68	43.19	55.91	62.30
	POF	**32.04**	**45.02**	**57.52**	**63.62**
Leftout	PCA	5.24	10.59	18.99	25.17
	PointNet++	17.35	28.82	43.26	51.17
	POF	**19.13**	**31.83**	**46.22**	**53.78**

Table 5. Point cloud normal estimation accuracy (%) on the ShapeNet dataset for both heldout training-category samples and leftout categories.

4.2. Point Cloud Upsampling

We test PUF on point cloud upsampling and present results in Tab. 3. We compare against the state-of-the-art point set upsampling method PU-Net on three metrics: MSE, EMD as well as the Chamfer Distance (CD). Our method outperforms the baseline under all metrics by 9.25% average improvement on the heldout training-category samples. Since geodesic neighborhoods are better informed of the underlying point set topology than Euclidean ones, PUF upsampling produces less outliers and recovers more details in Fig. 5, such as curves and sharp structures.

Generality. To analyze outlier robustness (i.e. points with large reconstruction errors), we plot top-k MSE in Fig. 6. Our method generates fewer outliers on both the heldout training-category samples and the unseen categories. We also report quantitative results on the leftout categories in Tab. 3. Again, PUF significantly surpasses the state-of-the-art upsampling method PU-Net under three different evaluation metrics.

4.3. Normal Estimation and Mesh Reconstruction

For normal estimation we apply PointNet++ geodesic fusion, POF, then we conduct Poisson mesh reconstruction leveraging the estimated normals. Quantitative results for normal estimation on the Shrec15 dataset and the ShapeNet dataset are given in Tab. 4 and Tab. 5, respectively. We compare our method with the traditional PCA algorithm as well as the state-of-the-art deep learning method PointNet++. Our results outperform the baselines by around 10% relative improvement. In Fig. 8, we visualize typical normal estimation errors, showing that PointNet++ usually fails at high-curvature and complex-surface regions. For further evidence, we visualize Poisson mesh reconstruction in Fig. 7 using the estimated normals.

Generality. In Tab. 5 we evaluate normal estimation performance on the leftout ShapeNet categories. Our method has higher accuracy over competing methods under multiple angle thresholds. Though trained with point clouds of 2048 points, POF is also tested on denser input. In Fig. 9 we take point clouds with 8192 points as input, and visualize the normal estimation and mesh reconstruction results, which shows that our method generalizes to dense point

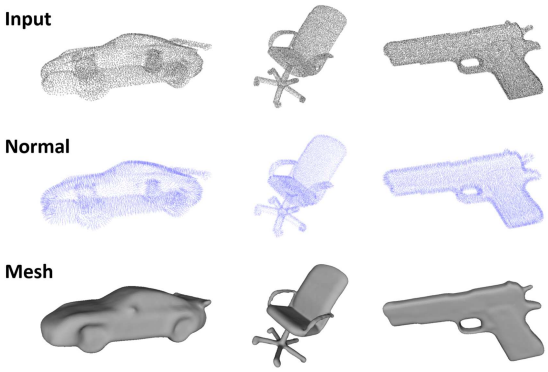

Figure 9. Normal estimation and Poisson mesh reconstruction results by POF using dense point clouds with 8192 points.

	Input feature	Accuracy (%)
PointNet++	XYZ	73.56
POF	XYZ	**94.67**
DeepGM	Intrinsic features	93.03

Table 6. Point cloud classification of non-rigid shapes on the Shrec15 dataset.

	Gaussian Noise Level				
	0.8%	0.9%	1.0%	1.1%	1.2%
PointNet++	70.54	69.27	67.83	65.66	62.38
POF	91.89	90.93	89.40	87.72	84.98

Table 7. Noisy point clouds classification accuracy (%). We add Gaussian noise of 0.8% to 1.2% of unit ball radius.

clouds without re-training and produces fine-scaled mesh.

4.4. Non-rigid Shape Classification

Results of non-rigid shape classification are reported in Tab. 6. While POF and PointNet++ only take point cloud-based xyz Euclidean coordinates as input, DeepGM requires offline computed intrinsic features from mesh data in the ground truth geodesic metric space. Though using less informative data, our method has higher classification accuracy than other methods, which further demonstrates that the proposed geodesic fusion architecture, POF, is suitable for solving tasks that require understandings of the underlying point cloud surface attributes.

Generality. We add Gaussian noise of different levels to the input and conduct noisy point clouds classification. Comparisons are shown in Tab. 7. POF outperforms PointNet++ under several noise levels. Our method also demonstrates better noise robustness. It shows a 10.24% decrease in relative accuracy at the maximum noise level, while PointNet++ decreases by up to 15.20%.

4.5. Failure Modes

Failure cases of geodesic neighborhood estimation are shown in Fig. 10. Due to large ratios between length and width/height, after normalizing a stick-shaped object (e.g. rocket, knife, etc.) into a unit ball we need high precision small values to represent its point-pair geodesic distance along the width/height sides. Since stick-shaped objects like rocket and knife only take up a small portion of the training data, GeoNet tends to make mistakes for held-out samples from these categories at inference time. Using an anistropic normalization might alleviate this issue but is challenging in practice as the principal directions would have to be estimated. We have not found additional failure cases, and quantitative improvements continue to take effect due to rich surface-based topological information learned during the geodesic-supervised training process.

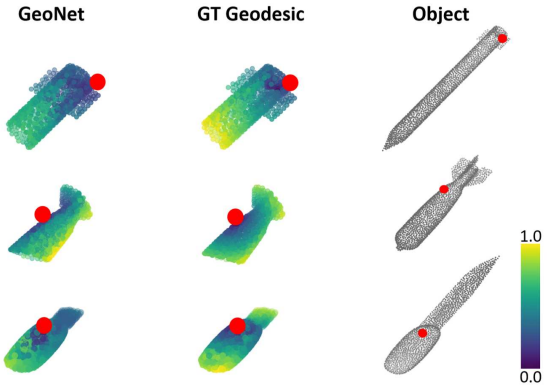

Figure 10. Failure cases of geodesic neighborhood estimation for stick-shaped objects (e.g. rocket, knife, etc.) which have large ratios between length and width/height. Red dots indicate the reference point. Points in dark-purple are closer to the reference point than those in bright-yellow.

5. Conclusion

We have presented GeoNet, a novel deep learning architecture to learn the geodesic space-based topological structure of point clouds. The training process is supervised by the ground truth geodesic distance and therefore the learned representations reflect the intrinsic structure of the underlying point set surfaces. To demonstrate the applicability of such a topology estimation network, we also propose fusion methods to incorporate GeoNet into computational schemes that involve the standard backbone architectures for point cloud analysis. Our method is tested on both geometric and semantic tasks and outperforms the state-of-the-art methods, including point upsampling, normal estimation, mesh reconstruction and non-rigid shape classification. For future works we will move on to complicated scenes where a mesh might need to be approximated for supervision.

Acknowledgment

Research supported by ONR N00014-17-1-2072, N00014-13-1-034 and ARO W911NF-17-1-0304.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on visualization and computer graphics*, 9(1):3–15, 2003. 2

[2] N. Amenta and M. Bern. Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry*, 22(4):481–504, 1999. 2

[3] M. Aubry, U. Schlickewei, and D. Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1626–1633. IEEE, 2011. 3

[4] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005. 2

[5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 6

[6] J. Chen and Y. Han. Shortest paths on a polyhedron. In *Proceedings of the sixth annual symposium on Computational geometry*, pages 360–369. ACM, 1990. 2

[7] W. Chen, X. Han, G. Li, C. Chen, J. Xing, Y. Zhao, and H. Li. Deep rbfnet: Point cloud feature learning using radial basis functions. *arXiv preprint arXiv:1812.04302*, 2018. 1

[8] K. Crane, C. Weischedel, and M. Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics (TOG)*, 32(5):152, 2013. 2

[9] A. Dai, C. R. Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2017. 2

[10] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016. 2

[11] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1(1):269–271, Dec. 1959. 2

[12] X. Fei, A. Wong, and S. Soatto. Geo-supervised visual depth prediction. *IEEE Robotics and Automation Letters*, 4(2):1661–1668, 2019. 2

[13] R. W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962. 2

[14] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216. ACM Press/Addison-Wesley Publishing Co., 1997. 2

[15] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2

[16] G. Guennebaud and M. Gross. Algebraic point set surfaces. In *ACM Transactions on Graphics (TOG)*, volume 26, page 23. ACM, 2007. 2

[17] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra. Pcpnet learning local shape properties from raw point clouds. In *Computer Graphics Forum*, volume 37, pages 75–85. Wiley Online Library, 2018. 2

[18] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3279–3286, 2015. 4

[19] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[20] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[21] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992. 2

[22] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM transactions on graphics (TOG)*, 28(5):176, 2009. 2, 5

[23] D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM (JACM)*, 24(1):1–13, 1977. 2

[24] I. Jolliffe. Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer, 2011. 2

[25] M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):29, 2013. 5

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[27] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer. Parameterization-free projection for geometry reconstruction. *ACM Transactions on Graphics (TOG)*, 26(3):22, 2007. 2, 5

[28] L. Luciano and A. B. Hamza. Deep learning with geodesic moments for 3d shape classification. *Pattern Recognition Letters*, 105:182–190, 2018. 3

[29] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An invitation to 3-d vision: from images to geometric models*, volume 26. Springer Science & Business Media, 2012. 2

[30] M. Masoumi, C. Li, and A. B. Hamza. A spectral graph wavelet approach for nonrigid 3d shape retrieval. *Pattern Recognition Letters*, 83:339–348, 2016. 3

[31] J. S. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM Journal on Computing*, 16(4):647–668, 1987. 2

[32] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017. 1

[33] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 1, 2, 3

[34] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace–beltrami spectra as shape-dnaof surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006. 3

[35] M. Sharir and A. Schorr. On shortest paths in polyhedral spaces. *SIAM Journal on Computing*, 15(1):193–215, 1986. 2

[36] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009. 3

[37] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, and H. Hoppe. Fast exact and approximate geodesics on meshes. In *ACM transactions on graphics (TOG)*, volume 24, pages 553–560. Acm, 2005. 2

[38] A. Wong, B.-W. Hong, and S. Soatto. Bilateral cyclic constraint and adaptive regularization for unsupervised monocular depth prediction. *arXiv preprint arXiv:1903.07309*, 2019. 2

[39] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 2

[40] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. 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. 2

[41] S.-Q. Xin and G.-J. Wang. Improving chen and han's algorithm on the discrete geodesic problem. *ACM Transactions on Graphics (TOG)*, 28(4):104, 2009. 2

[42] Y. Yang and S. Soatto. Conditional prior networks for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 271–287, 2018. 2

[43] Y. Yang, A. Wong, and S. Soatto. Dense depth posterior (ddp) from single image and sparse range. *arXiv preprint arXiv:1901.10034*, 2019. 2

[44] L. Yi, H. Su, X. Guo, and L. J. Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *CVPR*, pages 6584–6592, 2017. 2

[45] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018. 1, 2