applied optics

TDNet: transformer-based network for point cloud denoising

Xueli Xu,[1,2] Guohua Geng,[1,5] Xin Cao,[1,6] Kang Li,[1] and Mingquan Zhou[3,4]

[1]School of Information Science and Technology, Northwest University, Xi'an, Shaanxi, China
[2]College of Mathematics and Computer Science, Yan'an University, Yan'an, Shaanxi, China
[3]College of Information Science and Technology, Beijing Normal University, Beijing, China
[4]Engineering Research Center of Virtual Reality and Applications, Ministry of Education, Beijing Key Laboratory of Digital Preservation and Virtual Reality for Cultural Heritage, Beijing Normal University, Beijing, China
[5]e-mail: ghgeng@nwu.com
[6]e-mail: xin_cao@163.com

Received 22 July 2021; revised 8 November 2021; accepted 21 November 2021; posted 23 November 2021; published 15 December 2021

This study proposes a novel, to the best of our knowledge, transformer-based end-to-end network (TDNet) for point cloud denoising based on encoder–decoder architecture. The encoder is based on the structure of a transformer in natural language processing (NLP). Even though points and sentences are different types of data, the NLP transformer can be improved to be suitable for a point cloud because the point can be regarded as a word. The improved model facilitates point cloud feature extraction and transformation of the input point cloud into the underlying high-dimensional space, which can characterize the semantic relevance between points. Subsequently, the decoder learns the latent manifold of each sampled point from the high-dimensional features obtained by the encoder, finally achieving a clean point cloud. An adaptive sampling approach is introduced during denoising to select points closer to the clean point cloud to reconstruct the surface. This is based on the view that a 3D object is essentially a 2D manifold. Extensive experiments demonstrate that the proposed network is superior in terms of quantitative and qualitative results for synthetic data sets and real-world terracotta warrior fragments.

https://doi.org/10.1364/AO.438396

1. INTRODUCTION

Three-dimensional (3D) point clouds have been widely utilized in the fields of reverse engineering, precision manufacturing, and virtual reality because of their powerful model representation abilities and simplicity of form [1–4]. An initial point cloud obtained by scanning often contains a large number of noise points because it is affected by many factors, such as measuring equipment, external environment, and surface characteristics of the measured object. The greater the number of noise points, the greater the impact on the quality of the point cloud, which directly affects accuracy and efficiency of subsequent tasks, such as feature extraction, registration, surface reconstruction, and visualization [5–10]. Therefore, the initial data must be denoised.

In recent years, scholars have conducted in-depth research on point cloud denoising methods [11–13]. Traditional approaches to achieve point cloud denoising are based on first performing surface fitting to fit the surface on a 3D scanning point cloud of the object, calculating the distance from each point to the fitted surface, and finally removing gross errors or outliers according to certain criteria. Although it is a simple and effective estimation method, it cannot yield satisfactory results. In particular, there are large calculation errors for complex and noisy models. Mattei *et al.* [14] proposed a "mobile robust principal component analysis" method based on sparse representation theory. The estimated position of the point was calculated using the local average value that the sharp feature retained using the weighted minimization mode. Then, the weight was used to update the point's position to determine the similarity between the normal vectors in the local neighborhood. However, the performance tended to decline when the noise level was high due to excessive smoothing or sharpening.

With the rapid development of deep learning, extensive research has been conducted, and great success has been achieved in the field of image denoising [15–23]. In contrast with 2D images, point clouds are disordered and unstructured, and traditional convolution operations cannot consume them directly. Consequently, it is challenging to design a neural network to address this problem. Inspired by the success of convolutional neural networks (CNNs) in image processing, depth-map-based point cloud denoising methods convert 3D point clouds into 2D images [24,25]. Subsequently, deep learning is applied to 2D images to classify the point cloud into feature and nonfeature points and employ different approaches

to estimate and update the normal state of each point. End-to-end automatic learning and updating can be achieved; however, the complexities of time and space are high.

Many recent works have aggregated the local features of a point cloud by defining 3D convolution operators and then conducting denoising [26–29]. These methods either reorder the disordered input point cloud or voxelize it to obtain a convolutional canonical structure. However, this results in significant limitations in terms of timeliness and leads to a waste of space occupation. PointNet is a pioneering work proposed by Qi *et al.* that exploits deep learning for direct feature learning on a point cloud [30]. The model imposes a normalized rotation matrix on the point cloud for the purpose of permutation invariance, which results in excessive independence of the points. In addition, the network employs a global pooling operation to extract global features from the point cloud, which leads to geometric correlation among the points neglected and lost local feature information. Follow-up works based on PointNet, such as PointNet++ [31], neural projection [32], PointCleanNet [33], and total denoising [34], consider the local characteristics of points to improve the model performance. These methods can infer the displacement of the noise points from the underlying surface and reconstruct the points. However, these points are not designed to explicitly reconstruct the surface, which may lead to suboptimal denoising results.

Recently, characteristics (such as permutation invariance) of a transformer [35] have been frequently studied and applied to determine that it is suitable for point cloud learning. All operations of a transformer can be performed in parallel independently of the order. This potentially makes a transformer an exciting option for point cloud feature extraction. Theoretically, a transformer has better versatility and can replace the convolution operation in a CNN. However, downsampling is required to more effectively capture the local context information of the point cloud. Local features are obtained by combining the sampled points with their neighborhoods. The farthest point sampling (FPS) method is the most representative sampling method and can generate relatively uniform sampling points. The corresponding sampling points and their neighborhoods are able to overlap the input point cloud as much as possible. However, the FPS method has two main problems: (1) it is sensitive to abnormal points, making it unstable when dealing with point clouds in the real world; (2) the sampling points using the FPS method must be a subset of the original point cloud. It is difficult to infer the original geometric information if occlusion or loss errors occur during the acquisition process. This will lead to suboptimal feature extraction results and may affect accuracy of subsequent missions.

Stimulated by the success of the transformer in natural language processing (NLP) and computer vision, this study proposes a transformer-based deep-learning network for point cloud denoising. An adaptive sampling method is introduced to adjust the initial sampling points, which is helpful for learning the latent manifold of the point cloud to capture its inherent structure without being affected by outliers. The surface was reconstructed based on the adjusted points to sample and obtain a clean point cloud.

The contributions of this paper are as follows:

(1) A deep neural network called TDNet is proposed with an encoder–decoder architecture for point cloud denoising. The proposed network can generate a clean point cloud $\tilde{P} \in R^{N\times 3}$ when starting with an input point cloud $P \in R^{N\times 3}$ corrupted by noise.
(2) The structure of the transformer in NLP is improved to be appropriate for a point cloud and acquire feature representations with abundant semantics.
(3) Adaptive downsampling is introduced to improve the restrictions of the most widely used sampling method, FPS, and make the sampled noise points as close as possible to the clean points.
(4) The latent manifolds of the noise point cloud are trained to capture the inherent structure and thereby perform surface reconstruction. Finally, the same number of points as the input are sampled.

The remainder of this paper is organized as follows. Section 2 introduces the proposed framework, including the strategy and structure of the encoder and decoder in detail. Extensive experiments are presented in Section 3. Finally, conclusions are presented in Section 4.

2. METHODOLOGY

A point-cloud denoising network, motivated by transformer and manifold reconstruction, is presented. First, the input noise point cloud is consumed for feature learning, and the features are mapped to hidden layers that contain mathematical expressions with point contextual semantics. Second, the learned features with rich semantics are reversely translated to the point cloud using the reconstruction of the latent manifold for each sampled point; then, the points are resampled as the predicted results. The proposed network generates a clean point cloud $\tilde{P} \in R^{N\times 3}$ from the input point cloud $P \in R^{N\times 3}$, which is corrupted by noise. The overall architecture is shown in Fig. 1.

A. Encoder-Feature Learning

All operations of the transformer in NLP can be executed in parallel independently of the order to theoretically replace the convolution operation. Additionally, its intrinsic permutation invariance is suitable for point cloud feature learning. Unfortunately, the point cloud and natural language are different types of data, so the transformer model in NLP must be adjusted to view the point cloud as a sentence and the point as a word. The encoder proposed in this paper transforms the input points into a high-dimensional feature space that can characterize the semantic relevance between points and serve as the input of the decoder. The encoder learns semantically rich and distinctive representations of each point before yielding output features, as shown in Fig. 1(a). Compared with the naive transformer [35], this model discards position embedding because the point coordinate itself already contains this information and then adds a sampling operation.

1. Coordinate-Based Input Embedding Module

Similar to word embedding in NLP, point embedding moves the points closer in the embedding space when the semantics

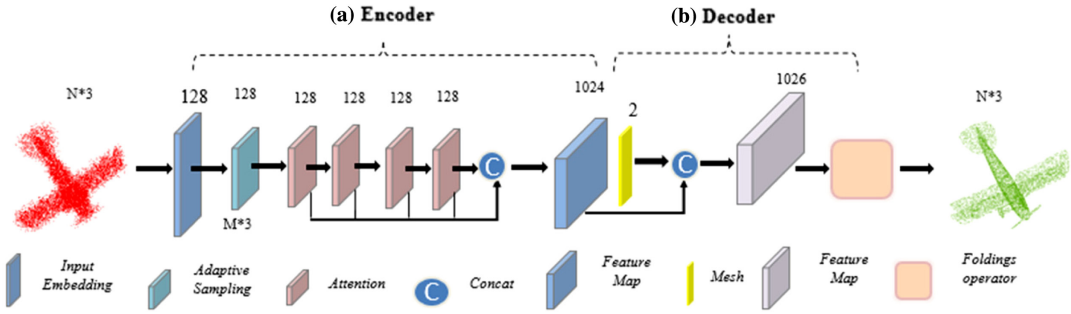

Fig. 1. Components of the overall architecture of the denoising network: (a) encoder and (b) decoder.

are more similar. The position coding module in the naive transformer is used to represent the order of words in natural language, which is leveraged to distinguish the same word in different positions and reflect the positional relationship between words. Each point has a unique position that can generate distinguishable features because the point cloud itself has coordinate information. Therefore, in the input embedding module, original position coding and input embedding are merged into a coordinate-based input-embedding module.

Three-dimensional coordinates of the point cloud are used as the input in this study. Therefore, $d = 3$ features of N points in the given input point cloud $P \in R^{N\times d}$ are mapped to the high-dimensional space through the input embedding module to obtain the de-dimensional feature in $Fe \in R^{N\times de}$. Considering the calculation efficiency and based on experience, the value of de was set to a relatively small value of 128.

2. Adaptive Local Neighborhood Sampling Module

The transformer can effectively extract global features using point embedding; however, it may ignore exceedingly important local geometric information. Motivated by the view of local neighborhood embedding in PointNet++ [31], a point sampling method is employed to acquire local neighborhood features with semantic information to further enhance the feature learning capability of the model. An adaptive local neighborhood downsampling (AS) module is exploited to overcome the limitations of the most widely used sampling method, FPS. The AS module is realized as follows. First, FPS is leveraged to obtain relatively uniform points as the original sampling points. Second, the AS sampling approach is introduced to automatically learn the offset of each sampling point and update the position. A schematic diagram of this process is shown in Fig. 2.

For AS, Ps represents a point set formed by sampling Ns points from N input points. x_i is the sampling point of Ps, and $x_i \in Ps$. f_i is a feature of point x_i, and $f_i \in Fs$. The neighbors of the sampling points are divided into groups using the k-NN query method, and the general self-attention mechanism is used to update the group features [35].

Features $f_{i,1}, \ldots, f_{i,k}$ corresponding to the k nearest neighbors $x_{i,1}, \ldots, x_{i,k}$ of the sampling point x_i, can be expressed as

$$f_{i,j} = A\left(R\left(x_{i,j}, x_{i,k}\right)\gamma\left(x_{i,k}\right)\right), \quad j \in [1, k], \quad \forall x_{i,k} \in N(x_i), \tag{1}$$

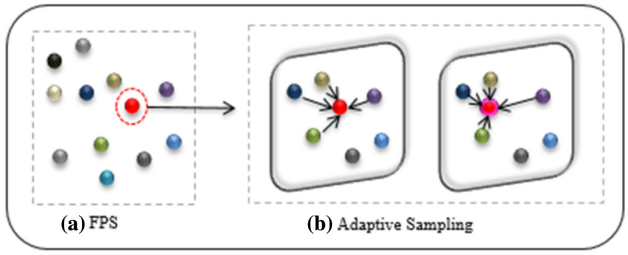

Fig. 2. Illustration of (a) FPS and (b) AS, where each sampled point is combined with its k nearest neighbors to update the position closer to the clean point cloud.

where A denotes the aggregation of features, R describes the high-level relationship between sampling point x_i and its neighbor $x_{i,j}$, and γ tends to change the feature dimension of each neighbor point. $\gamma(x_{i,j}) = W_\gamma f_{i,j}$ to decrease the amount of calculation, and W_γ is a learnable weight parameter. The relational function R is expressed as

$$R\left(x_{i,j}, x_{i,k}\right) = \text{Softmax}\left(\frac{\text{Conv}\left(f_{i,j}\right)^T \text{Conv}\left(f_{i,k}\right)}{\sqrt{D'}}\right), \tag{2}$$

where D' is the output channel of Conv. Subsequently, MLP + Softmax is adopted to obtain the normalized weights, W_p and W_f, for the coordinate and characteristic channel of each point in the group, respectively, expressed as

$$\begin{aligned} W_p &= \text{Softmax}\left(\text{mlp}\left(x_{i,j}\right)\right); \\ W_f &= \text{softmax}\left(\text{mlp}\left(f_{i,j}\right)\right), \quad j \in [1, k]. \end{aligned} \tag{3}$$

Finally, the adaptive update of sampling point x_i and its feature f_i are exploited through the weighted sum operation. x_i^* and f_i^* are the updated point information and are, respectively, expressed as

$$x_i^* = W_p^T X, \quad X = \left\{x_{i,j}\right\}_{j=1}^k; \quad f_i^* = W_f^T F, \quad F = \left\{f_{i,j}\right\}_{j=1}^k. \tag{4}$$

The adaptive downsampling operation obtains points closer to the latent surface with less noise disturbance, which is favorable for narrowing the latent space for reconstruction during the decoding stage.

3. Attention Module

In essence, the attention module strives to filter out and focus on a small amount of important extracted information while ignoring a large quantity of unimportant data. The greater the weight, the more important the corresponding value. Self-attention (SA) in naive transformers is a mechanism for paying attention to other words in the input sentence when encoding each word. The black dashed box in Fig. 3 shows the architecture of the SA. Q, K, and V represent the query, key, and value matrices, respectively, that were generated by the linear transformation of input features when switching to the point data stream, based on the terminology in [35]. The weight coefficient according to the query and key was calculated, and a weighted sum of values was performed according to the weight coefficient. The most common methods for calculating the weight coefficient include calculating the vector dot product of the two weight coefficients, calculating the vector cosine similarity of the two weight coefficients, or evaluating the value by introducing an additional neural network. The vector dot product was leveraged for calculation in this study to prevent the calculation result from becoming too large. Therefore, it was divided by the scale, which is the dimension of a query and key vector. Subsequently, Softmax was utilized to normalize the result to a probability distribution and multiply it by the value matrix to obtain the weighted summation. It can be expressed as

$$F_{\mathrm{sa}} = \mathrm{Attention}\,(Q,\ K,\ V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{5}$$

where $(Q,\ K,\ V) = F_{\mathrm{in}}.(W_q,\ W_k,\ W_v)$, W_q, W_k, and W_v are the weight matrices that can be shared.

SA is the permutation invariant during the calculation process, which makes it suitable for the disorder and irregularity of a point cloud. However, the absolute coordinates of the same point cloud after a rigid transformation are quite different from those before. To this end, the relative attention (RA) of the point cloud is introduced to describe the inherent characteristics of the point cloud.

The SA module in the original transformer is updated with RA to enhance the feature representation of the point cloud, which is inspired by the use of Laplacian matrix $L = D - A$ in the graph convolutional network to replace the adjacency matrix A. Here, D is a diagonal matrix [36], and each diagonal element D_{ii} represents the degree of the i-th node. The RA module calculates the RA feature between the SA and the input, that is, $F_{\mathrm{ra}} = F_{\mathrm{in}} - F_{\mathrm{sa}}$, as shown in Fig. 3. Finally, the RA and input features are further applied to acquire the final feature F_{out}, expressed as

$$F_{\mathrm{out}} = \mathrm{RA}\,(F_{\mathrm{in}}) = \mathrm{relu}\left(\mathrm{bn}\left(\mathrm{mlp}\,(F_{\mathrm{ra}})\right)\right) + F_{\mathrm{in}}. \tag{6}$$

There is an N-dimensional vector f, and $f = (f_1, f_2, \ldots, f_N)$ in graph G with N nodes in line with the discrete Laplace operator, where f_i is the value of the function f at node i. When point i is perturbed, it may become any adjacent node j because the Laplacian operator can calculate the gain from a point to a slight perturbation of all its degrees of freedom, which is represented by a graph. The gain attained by any node j changes to node i based on

$$\Delta f_i = \sum_{j \in N_i} (f_i - f_j). \tag{7}$$

When edge E_{ij} has weight W_{ij}, then

$$\Delta f_i = \sum_{j \in N_i} W_{ij}(f_i - f_j). \tag{8}$$

When $W_{ij} = 0$, it indicates that nodes i and j are not adjacent, so Eq. (8) can be simplified to

$$\Delta f_i = \sum_{j \in N} W_{ij}(f_i - f_j). \tag{9}$$

Subsequently,

$$\Delta f_i = \sum_{j \in N} W_{ij}(f_i - f_j) = \sum_{j \in N} W_{ij} f_i - \sum_{j \in N} W_{ij} f_j = dif i - w_{i:} f, \tag{10}$$

where $d_i = \sum_{j \in N} w_{ij}$ is the degree of vertex i. $w_{i:} = (w_{i1}, \ldots, w_{iN})$ and $f = \begin{pmatrix} f_1 \\ \vdots \\ f_N \end{pmatrix}$ are N-dimensional row and column vectors, respectively. $w_{i:} f$ represents the inner product of two vectors for all N nodes:

$$\begin{aligned} \Delta f &= \begin{pmatrix} \Delta f_1 \\ \vdots \\ \Delta f_N \end{pmatrix} = \begin{pmatrix} d_1 f_1 - w_{1:} f \\ \vdots \\ d_N f_N - w_{N:} f \end{pmatrix} \\ &= \begin{pmatrix} d_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & d_N \end{pmatrix} f - \begin{pmatrix} w_{1:} \\ \vdots \\ w_{N:} \end{pmatrix} f, \\ &= \mathrm{diag}\,(d_i)\ f - Wf = (D - W)\ f = Lf \end{aligned} \tag{11}$$

where $D - W$ is the Laplacian matrix L. The i-th row in the Laplacian matrix reflects the accumulation of gain generated by the i-th node when disturbing the other nodes. Intuitively, the Laplacian graph reveals that the potential can flow smoothly to other nodes in a specific direction when a potential on node i is applied. To this end, RA increases the attention weight and decreases the influence of noise, which is helpful for downstream tasks.

B. Decoder–Manifold Reconstruction

After obtaining the high-dimensional feature representation of the point cloud, it can be employed in the decoder to process different tasks, such as denoising. Previous denoising works mostly relied on the idea of point displacement from the latent surface. However, these points are not designated for surface reconstruction, which may lead to suboptimal denoising [32–34]. The point cloud usually represents a latent surface or 2D manifold of a set of sampling points. The latent manifold of the noise point cloud is learned, and its inherent structure is captured for reconstruction to ensure robust denoising, as shown in Fig. 4(b).

Specifically, the decoder converts the embedding features of each sampling point and its neighborhood into a local surface centered on that point to infer the latent manifold, defined as a patch manifold. Additionally, the inferred patch manifold is sampled multiple times to reconstruct a clean point cloud $\tilde{P} \in R^{N \times 3}$. The entire process is illustrated in Fig. 4.

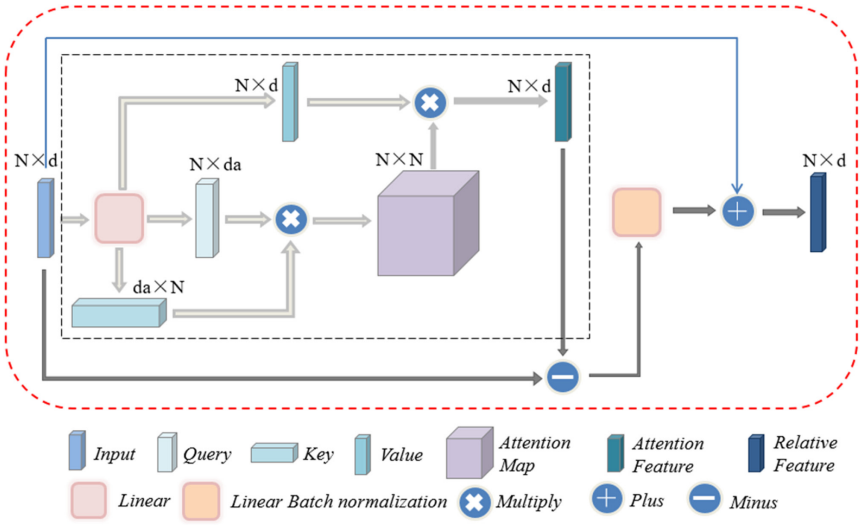

Fig. 3. Structure of the RA module. The SA module is in the black dashed box.

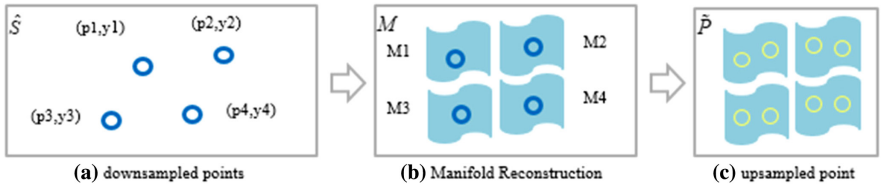

Fig. 4. Illustration of patch manifold reconstruction and resampling.

There is no strict point-to-point correspondence between the downsampling and upsampling point sets.

The 2D manifold M embedded in a 3D space parameterized by the feature vector y is formally defined as

$$M(u, v; y) : [-1, 1] \times [-1, 1] \to \mathbb{R}^3, \tag{12}$$

where (u, v) is a point in the 2D rectangular area $[-1, 1]^2$. Equation (12) maps a 2D rectangle to an arbitrarily shaped patch manifold parameterized by y. The parameterized patch manifold $M_i([u, v; y_i])$ is realized using MLP because it is a general approximator whose expressive ability is sufficient to approximate a manifold of any shape [37]. This is expressed as

$$M_i\,(u, v; y_i) = \mathrm{MLP}_M\,([u, v, y_i])\,. \tag{13}$$

With the definition of manifold M, the patch manifold M_i corresponding to point P_i in the downsampling point set $\hat{S}$ is defined as

$$M_i\,(u, v; y_i) = p_i + M\,(u, v; y_i)\,. \tag{14}$$

Equation (14) reveals that the constructed manifold $M(u, v; y_i)$ moved to a local surface centered on p_i, and the patch manifold corresponding to all points in $\hat{S}$ can be expressed as $\{M_i | p_i \in \hat{S}\}_{i=1}^{M}$, which represents the latent surface of the point cloud. In the previous adaptive downsampling period, the number of input points were reduced by half, that is, $M = N/2$. Likewise, the points on each patch manifold $M_i([u, v, y_i])$ need to be resampled twice to obtain the predicted denoising point cloud $\tilde{P}$, expressed as

$$\tilde{P} = \{p_i + \mathrm{MLP}_M\,([u_{i1}, v_{i1}, y_i])\}_{i=1}^{M} + \{p_i + \mathrm{MLP}_M\,([u_{i2}, v_{i2}, y_i])\}_{i=1}^{M}\,. \tag{15}$$

C. Loss

A total loss function is designed to measure the quality of a denoising point cloud that is composed of two parts, L_{as} and L_{us}. L_{as} quantifies the distance between the adaptive downsampling set $\hat{S}$ and ground truth point cloud P_{gt}. L_{us} quantifies the distance between the final denoising point cloud $\tilde{P}$ and the ground truth P_{gt}. $\hat{S}$ and P_{gt} contain different numbers of points, $|\hat{S}| < |P_{\mathrm{gt}}|$, so the Chamfer distance (CD) was chosen as L_{as} to further optimize the adaptive sampling results [38]. L_{as} is expressed as

$$L_{\mathrm{as}} = L_{\mathrm{CD}}\left(\hat{S}, P_{\mathrm{gt}}\right) = \frac{1}{|\hat{S}|}\sum_{p\in\hat{S}} \min_{q\in P_{\mathrm{gt}}} ||p - q||_2^2 + \frac{1}{|P_{\mathrm{gt}}|}\sum_{q\in P_{\mathrm{gt}}} \min_{p\in\hat{S}} ||p - q||_2^2\,. \tag{16}$$

The first part of Eq. (16) indicates that the CD algorithm finds the nearest neighbor point in the ground truth point set P_{gt} for each point in the sampled point set $\hat{S}$, then sums the squared distances. The second part shows the difference in data distribution between the two sets.

The Earth mover's distance was leveraged as L_{us} to measure the distance between the denoising point cloud $\tilde{P}$ and ground

truth point cloud P_{gt} [38]. It is expressed as

$$L_{us} = L_{EMD}(\tilde{P}, P_{gt}) = \min_{\varphi:\tilde{P}\to P_{gt}} \frac{1}{N}\sum_{p\in\tilde{P}} ||p-\varphi(p)||_2^2, \quad (17)$$

where $|\tilde{P}| = |P_{gt}| = N$ and φ is a bijection.

End-to-end supervised training was conducted on the network to minimize the total loss $L_{denoise} = \lambda L_{as} + (1-\lambda)L_{us}$. λ is an empirical value of 0.01, and an ADAM optimizer was employed with an initial learning rate of 0.001, which decreased during the training.

3. EXPERIMENT

In this section, the proposed framework is quantitatively and qualitatively compared with the latest denoising networks. The proposed network was run in parallel on a system with two NVIDIA RTX 2080Ti graphics cards, Python 3.6, Pytorch 1.5, and Cuda 10.0.

A. Experimental Setup

1. Data Set

Twenty categories of point clouds with eight different shapes in each category were collected from ModelNet-40 for training [39]. Poisson disk sampling (PDS) was utilized to sample point clouds with a resolution from 20 to 100 K [40]. In addition, 20% intervals were used as the basic original point cloud. For the denoising task, noise point clouds were generated by adding Gaussian noise with standard deviations of 0.25%, 0.5%, 1%, 2%, and 3% of the original shape's bounding box diagonal to obtain a total of 4000 point clouds. Twenty types of point clouds with four shapes in each category were collected as the test data set. PDS with resolutions of 20 and 50 K were employed to generate the point clouds, which were perturbed with Gaussian noise. The standard deviations were the same as those of the training data set to obtain a total of 800 shapes.

Furthermore, the terracotta warrior fragments unearthed with a laser scanner in K9901 of the Qin Shihuang Mausoleum Site Museum were used as experimental data to check the versatility of the proposed network for point clouds in the real world.

2. Metrics

Two metrics were utilized to illustrate the effectiveness of the experiment. The first used the CD algorithm as an evaluation metric to measure the distance between the denoising $\tilde{P}$ and ground truth P_{gt} point clouds, expressed as

$$C\left(\tilde{P}, P_{gt}\right) = \frac{1}{\left|\tilde{P}\right|}\sum_{p\in\tilde{P}} \min_{q\in P_{gt}} ||p-q||_2 + \frac{1}{\left|P_{gt}\right|}\sum_{q\in P_{gt}} \min_{p\in\tilde{P}} ||q-p||_2. \quad (18)$$

The first term measures the distance from each predicted point to the target surface; the second term represents the uniform distribution of the output point cloud on the target surface. Equation (18) is different from the $\ell 2$ square distance used in Eq. (16), which is a term in the loss function. Calculating the $\ell 2$ distance involves a square root operation, which is not suitable as a loss function for model training due to its numerical instability.

A point-to-surface (P2S) distance $\rho(\tilde{P}, S)$ was also employed because the proposed framework aimed to reconstruct the latent surface, expressed as

$$\rho\left(\tilde{P}, S\right) = \frac{1}{\left|\tilde{P}\right|}\sum_{p\in\tilde{P}} \min_{q\in S} ||p-q||_2, \quad (19)$$

where S is the latent surface of the ground truth point cloud P_{gt}. For these two metrics, a smaller value indicates a better result. For high noise point clouds, the best results can be obtained by iterative denoising, i.e., feeding the output of the network as input again.

B. Quantitative Results

The proposed network was quantitatively compared with previous deep learning networks, including neural projection (NPD), PointCleanNet (PCNet), and total denoising (TlDn). The CD was calculated for each noise level and the P2S distance was based on 800 point clouds. The metric values of CD and P2S for each method become progressively larger as the noise ratio increased when compared horizontally, as shown in Tables 1 and 2, respectively. When vertically compared with the same noise ratio, the NPD and TlDn models demonstrated similar results and poor metrics in CD and P2S.

The values of the PCNet model were better than those of the previous two models but still inadequate. The proposed TDNet-SA model demonstrated results similar to PCNet under lower noise conditions. The proposed SA and RA methods gradually displayed their advantages with an increase in noise ratio. In particular, the larger the noise, the more obvious the advantage of RA. The proposed framework was superior to previous works and more robust to high noise, as shown in Tables 1 and 2. The terracotta warrior fragment data set obtained by laser scanning was also evaluated to determine the generalizability of the proposed architecture. The proposed architecture demonstrated a good generalization performance, as shown in Table 3.

Table 1. Comparison of CD among Different Denoising Frameworks with Different Noise Ratios

10^{-2}	0.25%	0.5%	1%	2%	3%
NPD	0.24	0.62	1.28	2.32	3.27
PCNet	0.18	0.46	0.97	1.42	2.91
TlDn	0.34	0.78	1.15	2.26	3.12
TDNet-SA (proposed)	0.27	0.48	0.96	1.35	2.64
TDNet-RA (proposed)	0.16	0.39	0.83	1.20	2.15

Table 2. Comparison of P2S among Different Denoising Frameworks with Different Noise Ratios

10^{-2}	0.25%	0.5%	1%	2%	3%
NPD	0.27	0.54	1.24	2.69	4.68
PCNet	0.14	0.38	0.74	1.38	3.27
TlDn	0.33	0.65	1.05	2.78	4.41
TDNet-SA (proposed)	0.26	0.41	0.75	1.36	3.19
TDNet-RA (proposed)	0.13	0.29	0.52	1.03	2.68

Table 3. Comparison of CD and P2S among the Different Denoising Frameworks on Terracotta Warrior Fragments

10^{-2}	NPD	PCNet	TlDn	TDNet-SA (proposed)	TDNet-RA (proposed)
CD	1.32	1.10	1.27	1.15	1.06
P2S	1.28	0.94	1.19	1.04	0.89

It can be concluded from the experimental results that the performance of the proposed TDNet-RA was significantly better than that of the other methods. The performance of the TDNet-SA model was close to that of the latest PCNet at a lower noise rate; the greater the noise rate, the better the performance. Notably, compared with the CD, which is essentially the point-to-point distance, the proposed framework demonstrates more advantages when measured using the P2S distance. The proposed framework reconstructed the latent manifold of the point cloud, and points were resampled from this to obtain a final clean point cloud. Sampling on the manifold did not guarantee that the newly sampled points were close to the points in the original point cloud, which may have resulted in a relatively large point-to-point distance. However, the P2S distance provided better measurement because the point cloud was a representation of a 3D surface.

Similarly, the literature indicates that the distance from the point to the surface is more relevant to the subjective evaluation of the denoising results [41]. The significant advantage of the P2S distance in this work demonstrates that it is visually preferable to previous methods.

C. Qualitative Results

The proposed network was qualitatively compared with previous deep learning networks, such as NPD, PCNet, and TlDn. Comparison of the visual denoising results under Gaussian noise with different noise levels is shown in Fig. 5.

The reconstruction error of each point was measured using the distance from the point to the surface. Deep-learning based models with smaller loss values are represented in dark blue; otherwise, they are red, as shown in the color bar. Results for the proposed method were much cleaner than those of other methods, especially with higher noise ratios. Specifically, the proposed method was more robust compared with the NPD and TlDn methods and similar to the PCNet at lower noise levels. The proposed method reconstructed the underlying surface and produced more significant results. In summary, the qualitative results in Fig. 5 are consistent with the quantitative results shown in Tables 1 and 2.

Additionally, qualitative research was conducted on the data set of terracotta warrior fragments in the real world. The number of noise points (red) for the head and right hand of the terracotta warrior decreased but remained under the NPD and TlDn networks, as shown in Fig. 6.

The PCNet network results were better but not adequate. Additionally, the results for the proposed denoising method demonstrated fewer noise points, and the details were well

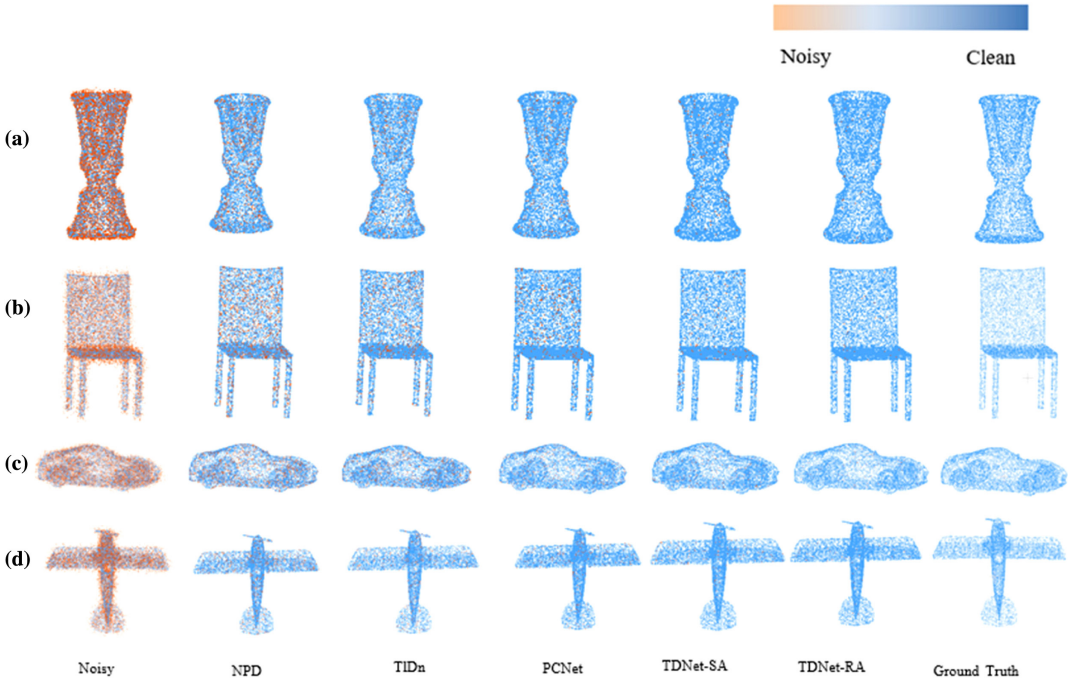

Fig. 5. Visual comparison among the different denoising frameworks under different Gaussian noises: (a) 3%, (b) 2%, (c) 1%, and (d) 0.5%.

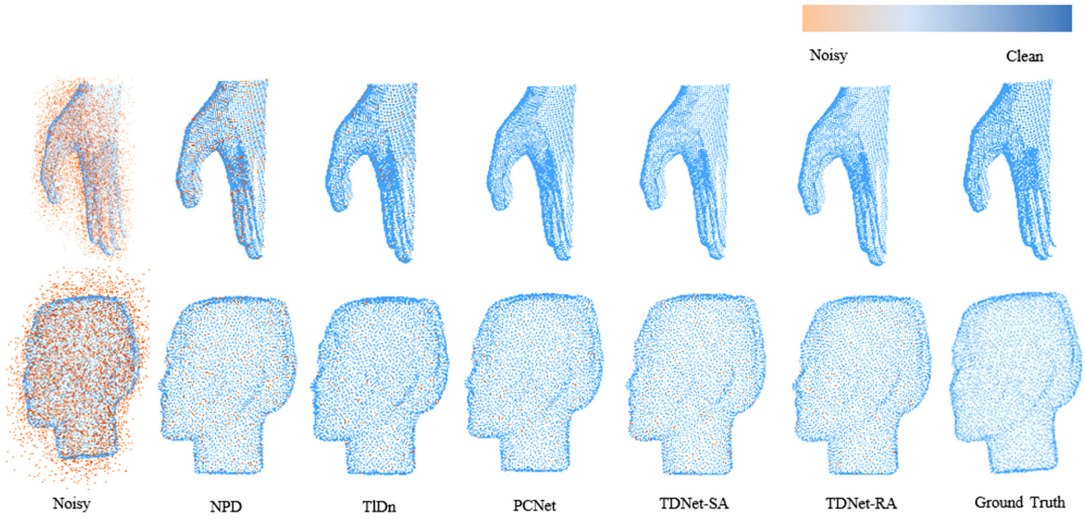

Fig. 6. Qualitative results of different denoising frameworks on terracotta warrior fragments.

preserved. This verified that the proposed approach could be effectively extended to real-world data sets.

In summary, the experimental results consistently demonstrated the effectiveness of the proposed method when quantitatively and qualitatively compared with other methods, especially when the noise was relatively high. Furthermore, the robustness of the proposed method was demonstrated by the superior performance for real-world data sets.

4. CONCLUSION

This study proposed a novel end-to-end denoising architecture that improved the transformer model in NLP to acquire richer high-dimensional feature representation. Additionally, it learned the latent manifold of the noise point cloud from the sampled points. The FPS method was updated with adaptive downsampling operations to make points closer to the surface where they were located. Accordingly, the patch manifold was inferred by converting each sampling point and its embedded neighborhood features to the local surface. A clean point cloud that captured the internal structure was reconstructed by sampling each patch manifold. Extensive experiments demonstrated that the proposed network was superior to other existing frameworks for synthetic noise point clouds and real terracotta warrior fragments. In the future, this framework will be tested and optimized for broader applications.

Funding. National Key Research and Development Program of China (2019YFC1521102, 2019YFC1521103); National Natural Science Foundation of China (61731013, 61731015); China Postdoctoral Science Foundation (2018M643719); Young Talent Support Program of the Shaanxi Association for Science and Technology (20190107); Shaanxi Provincial Key Industrial Chain Project (2019ZDLGY10-01, 2019ZDLSF07-02); Education Department of Shaanxi Province (21JK0975); Yan'an University Scientific Research (YDQ2019-10); Major Research and Development Project of Qinghai (2020-SF-143).

Disclosures. The authors declare no conflicts of interest.

Data Availability. The data underlying the results presented in this paper are not publicly available at this time but may be obtained from the authors upon reasonable request.

REFERENCES

1. A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Tutorial: point cloud library: three-dimensional object recognition and 6 DOF pose estimation," IEEE Robot. Autom. Mag. **19**(3), 80–91 (2012).
2. J. Liu, *Denoising Algorithms in 3D Point Cloud Reconstruction* (Beijing Jiaotong University, 2019).
3. J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon, "High quality depth map upsampling for 3D-TOF cameras," in *International Conference on Computer Vision* (IEEE, 2011), pp. 1623–1630.
4. R. B. Rusu and S. Cousins, "3D is here: point cloud library (PCL)," in *IEEE International Conference on Robotics and Automation* (IEEE, 2011), pp. 1–4.
5. H. Gao and G. Geng, "Classification of 3D terracotta warrior fragments based on deep learning and template guidance," IEEE Access **8**, 4086–4098 (2019).
6. Q. Li, G. Geng, and M. Zhou, "Pairwise matching for 3D fragment reassembly based on boundary curves and concave-convex patches," IEEE Access **8**, 6153–6161 (2019).
7. W. Yang, Z. Mingquan, Z. Pengfei, and G. Guohua, "Matching method of cultural relic fragments constrained by thickness and contour feature," IEEE Access **8**, 25892–25904 (2020).
8. J. Gao, B. Deng, Y. Qin, X. Li, and H. Wang, "Point cloud and 3-D surface reconstruction using cylindrical millimeter-wave holography," IEEE Trans. Instrum. Meas. **68**, 4765–4778 (2019).
9. Y. He, S. H. Kang, and H. Liu, "Curvature regularized surface reconstruction from point clouds," SIAM J. Imag. Sci. **13**, 1834–1859 (2020).
10. Y. Zhang, X. Liu, C. Li, J. Hu, G. Geng, and S. Zhang, "From 2D to 3D: component description for partial matching of point clouds," IEEE Access **7**, 173583–173602 (2019).
11. Y. Liu and S. Sun, "Laser point cloud denoising based on principal component analysis and surface fitting," Laser Technol. **44**, 497 (2020).
12. X. Wang, J. Cai, Z. Wu, and M. Zhou, "Normal estimation and normal orientation for point cloud model based on improved local surface fitting," J. Comput.-Aided Des. Comput. Graph. **27**, 614–620 (2015).
13. Y. Minlv, H. Xianghong, C. Xijiang, and W. Cheng, "Research on method for the denoising of point cloud based on orthogonal TLS fitting," Bull. Surv. Mapp. **11**, 37–39 (2013).
14. E. Mattei and A. Castrodad, "Point cloud denoising via moving RPCA," in *Computer Graphics Forum* (Wiley Online Library, 2017), pp. 123–137.
15. J. Chen, J. Chen, H. Chao, and M. Yang, "Image blind denoising with generative adversarial network based noise modeling," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3155–3164.

16. S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, "Toward convolutional blind denoising of real photographs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 1712–1722.
17. S. Lefkimmiatis, "Non-local color image denoising with convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 3587–3596.
18. J. Liang and R. Liu, "Stacked denoising autoencoder and dropout together to prevent overfitting in deep neural network," in *8th International Congress on Image and Signal Processing (CISP)* (IEEE, 2015), pp. 697–701.
19. X.-J. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Curran Associates, 2016) Vol. **29**, pp. 2802–2810.
20. Q. Xu, C. Zhang, and L. Zhang, "Denoising convolutional neural network," in *IEEE International Conference on Information and Automation* (IEEE, 2015), pp. 1184–1187.
21. K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising," IEEE Trans. Image Process. **26**, 3142–3155 (2017).
22. K. Zhang, W. Zuo, and L. Zhang, "FFDNet: toward a fast and flexible solution for CNN-based image denoising," IEEE Trans. Image Process. **27**, 4608–4622 (2018).
23. K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 1671–1681.
24. A. Ley, O. D'Hondt, and O. Hellwich, "Regularization and completion of TomoSAR point clouds in a projected height map domain," IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **11**, 2104–2114 (2018).
25. Y. He, L. Chen, J. Chen, and M. Li, "A novel way to organize 3D LiDAR point cloud as 2D depth map height map and surface normal map," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)* (IEEE, 2015), pp. 1383–1388.
26. L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: semantic segmentation of 3D point clouds," in *International Conference on 3D Vision (3DV)* (IEEE, 2017), pp. 537–547.
27. Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: convolution on X-transformed points," in *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)* (Curran Associates, 2018) Vol. **31**, pp. 820–830.
28. M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," ACM Trans. Graph. **37**, 71 (2018).
29. W. Wu, Z. Qi, and L. Fuxin, "PointConv: deep convolutional networks on 3D point clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 9621–9630.
30. C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: deep learning on point sets for 3D classification and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 652–660.
31. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems* (2017), pp. 5105–5114.
32. C. Duan, S. Chen, and J. Kovacevic, "3D point cloud denoising via deep neural network based local surface estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2019), pp. 8553–8557.
33. M. J. Rakotosaona, V. La Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, "PointCleanNet: learning to denoise and remove outliers from dense point clouds," in *Computer Graphics Forum* (Wiley Online Library, 2020), pp. 185–203.
34. P. Hermosilla, T. Ritschel, and T. Ropinski, "Total denoising: unsupervised learning of 3D point cloud cleaning," in *IEEE/CVF International Conference on Computer Vision* (2019), pp. 52–60.
35. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (2017), pp. 5998–6008.
36. Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," ACM Trans. Graph. **38**, 1–12 (2019).
37. A. Javaheri, C. Brites, F. Pereira, and J. Ascenso, "Subjective and objective quality evaluation of 3D point cloud denoising algorithms," in *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* (IEEE, 2017), pp. 1–6.
38. H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 605–613.
39. M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, "Tangent convolutions for dense prediction in 3D," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3887–3896.
40. P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte Carlo convolution for learning on non-uniformly sampled point clouds," ACM Trans. Graph. **37**, 1–12 (2018).
41. T. Le and Y. Duan, "PointGrid: a deep network for 3D shape understanding," in *IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 9204–9214.