DOI: 10.1111/cgf.14660
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

Volume 41 (2022), Number 7

Local offset point cloud transformer based implicit surface reconstruction

Y. X. Yang[†1] and S. G. Zhang[2]

[1,2]University of Chinese Academy of Sciences, Beijing, China

Abstract

Implicit neural representations, such as MLP, can well recover the topology of watertight object. However, MLP fails to recover geometric details of watertight object and complicated topology due to dealing with point cloud in a point-wise manner. In this paper, we propose a point cloud transformer called local offset point cloud transformer (LOPCT) as a feature fusion module. Before using MLP to learn the implicit function, the input point cloud is first fed into the local offset transformer, which adaptively learns the dependency of the local point cloud and obtains the enhanced features of each point. The feature-enhanced point cloud is then fed into the MLP to recover the geometric details and sharp features of watertight object and complex topology. Extensive reconstruction experiments of watertight object and complex topology demonstrate that our method achieves comparable or better results than others in terms of recovering sharp features and geometric details. In addition, experiments on watertight objects demonstrate the robustness of our method in terms of average result.

CCS Concepts

• ***Computing methodologies*** → ***Artificial intelligence;*** • ***Mathematics of computing*** → ***Graphs and surfaces;***

1. Introduction

Three-dimensional objects can be represented as voxels, multi-views, mesh, point clouds and implicit surface. With the rapid development of 3D sensors and depth cameras, 3d point cloud acquisition becomes easy. As an unified representation of 3d objects, 3D point cloud is widely used in classification, detection, segmentation, registration, reconstruction and other fields. Point cloud based reconstruction can be divided into explicit reconstruction [ABK98, ACK01, MAVDF05] and implicit reconstruction [YS01, DTS01, TO05, CBC*01, OBA*05, ABCO*03, RJT*05, FCOS05, HSD00, IJS03, SSB05, SBS05, DTB06, JWB*06, Kaz05, KBH06, KH13, MPS08]. The key of implicit reconstruction is implicit function, which include sign distance function (SDF) and occupancy function. SDF defines the distance from a given point x to the shape surface, with the sign determined by whether x is inside the shape volume or not. Traditional methods manually construct implicit functions, such as SDF [MAVDF05], RBF [CBC*01], MLS [FCOS05], etc., with limited expression ability. With the advent of deep learning, graphics scholars begin to apply deep learning methods to point cloud surface reconstruction. The goal is to represent the implicit functions as neural networks, and then extract surfaces through the zero-level set of implicit functions. Implicit surface reconstruction based on point cloud deep learning can be roughly divided into learning implicit function and regression implicit function. The difference between them is whether real implicit function values are needed as supervision to train the network. The former does not need while the latter does. The representative work of regressing the implicit function is [EGO*20], the encoder-decoder structure is proposed. The encoder phase learns a high-dimensional representation, and the decoder phase uses the real sign distance values as supervision to regress the implicit function. In addition, neural-IMLS [WWD*21] first uses IMLS [ÖGG09] to build real implicit functions, then uses MLP to fit the implicit functions, and finally calculates the loss. However, the two methods mentioned above sometimes fail to recover the topological structure of watertight objects. In contrast to discrete representations such as voxelization, mesh, etc., implicit representations are compact and not limited by pixels.

In this paper, we are interested in learning sign distance functions from a clean or noisy point cloud extracted from a 3d shape. The recent MLP-based methods, IGR [GYH*20] , can well recover the simple topology. However, due to the spectral deviation of neural network [RBA*19], MLP fails to recover the geometric details and complicated topology of the manifold surface. To solve the above problems, SIREN [SMB*20] propose sinusoidal activation function to replace softplus in MLP to enhance the expression ability of MLP. Before using MLP to learn SDF, FPE [MST*20, TSM*20, ZBDB19] uses a set of sinusoidal function

† Corresponding authors

to encode the input coordinates into the high-dimensional fourier space and SPE [WLYT21] maps the input coordinates into the high-dimensional spline space by a set of spline function to facilitate the expressiveness of MLP. However, the above methods could not recover SDFs in good quality. In addition to the above problems, the aforementioned approches learn the implicit function by MLP which deals with the point cloud in a pointwise manner and thus ignores the interdependence of the point cloud. Fortunately, our approach can recover high-quality SDFs and effectively fuse the information in the neighborhood of point cloud.

The emergence of transformer [VSP*17] has led to a further development in the field of natural language processing. With its excellent performance, vector transformer has been used in the field of computer vision [KNH*21], which has promoted the development of the field. The rapid development of transformer in the field of natural language and computer vision has aroused great interest in graphics. Then point cloud transformer [KNH*21, HYC*21, ZJJ*21, HJCX21, HKX21, GCL*21, KB14] were proposed one after another, achieving the best results in point cloud classification, segmentation and detection. However, to our knowledge, there is no work using point cloud transformer to solve the point cloud implicit reconstruction problem. In order to make use of the manifold topology recovery capability of MLP and solve the problems of MLP, we combine the work of point cloud transformer [GCL*21] and point transformer [ZJJ*21] and propose local offset point cloud transformer, LOPCT. Before using MLP to learn the implicit function, the input point cloud is first fed into the local offset transformer, which adaptively learns the dependency of the local point cloud and obtains the enhanced features of each point. In conclusion, our main contributions are as follows:

(1)To our knowledge, we are the first to use point cloud transformer to solve the problem of point cloud implicit reconstruction.

(2)MLP with LOPCT significantly improves the ability of manifold surface reconstruction in terms of geometric details and complex topology.

(3)When there is noise in the normal vector, our method shows robustness and significantly outperforms other methods in terms of average results.

2. Related work

2.1. Implicit neural representation

Implicit neural representation(INR) learned by MLP is a compact representation whose zero level set depicts the shape surface. By using INR and Eikonal equation constraint, IGR [GYH*20] can recover the topology of watertight object well. Some scholars believe that the spectral deviation of neural networks leads to the failure of MLP to recover geometric details. SIREN [SMB*20] proposed sinusoidal activation function and a new initialization scheme, which improves the performance of IGR, but fails to recover SDFs in good quality. In order to enhance the expression ability of MLP, fourier position encoding module [MST*20, TSM*20, ZBDB19] and B-spline position encoding module [WLYT21] are presented before using MLP to learn SDF. Unfortunately, FPE contains a lot of impurities when reconstructing watertight objects and SPE requires multi-scale optimization to gradually recover geometric details of target object. However, We believe that MLP does not consider the mutual information between points, which leads to its inability to recover geometric details and complex topology. Our method mainly solves the problem that MLP does not consider the mutual information between points, while SIREN, FPE and SPE mainly solve the spectral deviation problem of MLP. Therefore, our network structure mainly focuses on point cloud information interaction while other methods transform the time domain problem into the frequency domain problem.

2.2. Point cloud transformer

PCT [GCL*21] used offset attention for point cloud classification, segmentation and normal estimation and got the best results. In order to be robust to noise, a robust normalization method is proposed and neighborhood embedding is proposed to enhance local feature representation. PT [ZJJ*21] proposed local transformer, which first uses KNN to build the neighborhood and then uses transformer in the neighborhood to achieve the best result in the scene segmentation task. Recently, Dual Point Cloud Transformer [HJCX21] is presented to aggregate the well-designed point-wise and channel-wise multi-head self-attention models simultaneously. In addition, lightweight point cloud transformer [WJC*22] is proposed to make a trade-off between speed and accuracy.

3. Method

In this section, we first briefly review the general formulation of self-attention operators. Then we will detail our proposed local offset point cloud transformer and introduce our network framework for surface reconstruction. Finally, loss functions and parameter settings are introduced.

3.1. Background: Self-attention mechanism

Self-attention mechanism can be roughly divided into scalar attention and vector attention. Scalar self-attention is often used in the NLP domain and vector self-attention is often used in the CV domain.

Let $\mathcal{V} = \{\mathbf{v}_i\}$ be a set of feature vectors, which is called input embedding in the NLP, learned by linear layer from inputs. Let $\Delta = \{\delta_i\}$ be a set of position encoding used to encode relative positions between words in a sentence in NLP. The traditional scalar attention layer can be represented as follows:

$$\mathbf{y}_i = \sum_{\mathbf{v}_j \in \mathcal{V}} \beta(w_q(\mathbf{v}_i + \delta_i)^T w_k(\mathbf{v}_j + \delta_j)) w_v(\mathbf{v}_j + \delta_j) \qquad (1)$$

where $\mathbf{y}_i$ is the output feature. w_q,w_k,w_v are pointwise feature transformations, such as fully connected layer and convolution layer. The output of w_q,w_k and w_v are called query vector, key vector and value vector respectively. β is a normalization function used to transform the output of inner product into a discrete probability distribution. The output feature $\mathbf{y}_i$ are obtained by using the discrete probability distribution to weight the value vector. So, the output feature adaptively integrates the semantic information of all input feature, which is vital importance for downstream tasks.

Vector attention is frequently used in computer vision and the computation of attention weights is different which can be formulated as:

$$\mathbf{y}_i = \sum_{\mathbf{v}_j \in \mathcal{V}} \beta(\alpha(\varphi(w_q(\mathbf{v}_i), w_k(\mathbf{v}_j)) + \delta)) \odot w_v(\mathbf{v}_j), \tag{2}$$

w_q, w_k, and w_v have the same meaning as scalar attention. φ is a relation function such as subtraction and concatenation. There are three differences between scalar and vector attention. The first difference between scalar attention and vector attention is the computation of attention vectors, the former by inner product and the latter by mapping function α. The second difference is where the position encoding vector is added. The third difference is the way value vectors are integrated, the former by multiplication and the latter by Hadamard products.

3.2. Local vector self-attention

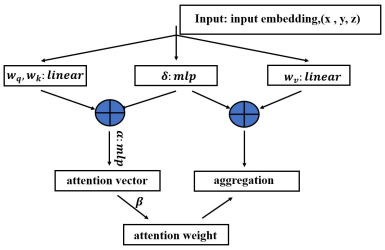

Figure 1: *Local vector self-attention.*

Point cloud is essentially a set of points in a metric space with rigid transformation invariance, so self-attention is a natural fitting of point cloud.

Our self-attention machanism depends upon vector self-attention, as shown in Figure 1. The structure is similar to P-T [ZJJ*21], but there are some differences. w_q, w_k and w_v are fully connected layer which takes as input feature vector obtained by linear layer from input point cloud. Subtraction relation is used and a position encoding δ is added to the input of α and the value vector $w_v(\mathbf{v}_j)$, which combines the characteristics of scalar attention and vector attention. Our local vector self-attention mechanism can be formulated as:

$$\mathbf{y}_i = \sum_{\mathbf{v}_j \in \mathcal{V}(i)} \beta(\alpha(w_q(\mathbf{v}_i) - w_k(\mathbf{v}_j) + \delta)) \odot (w_v(\mathbf{v}_j) + \delta) \tag{3}$$

Here the subset $\mathcal{V}(i) \subseteq \mathcal{V}$ is a subset of features obtained by KNN based on point cloud coordinates. Thus we apply vector self-attention locally. This operation has two advantages. First, the main computational cost of the self-attention mechanism comes from the matrix computation, which will lead to a serious computational burden if all the input embeddings are selected to compute. Second, as the distance between the surrounding points and the center point increases, they carry less useful information and may contain noise. If all input embeddings are used, there is no significant information enhancement and the central point may contain noise. The mapping function α is an MLP with two fully connected layers and the first fully connected layer is followed by a Softplus nonlinearity. We chose Softplus instead of ReLU because Softplus is differentiable everywhere and therefore allows network parameters to be constantly updated.

Position encoding plays a vital role in self-attention. It is mainly used to encode relative positions between sequences. Traditional position encoding are crafted manually. Here position encoding is to be learned adaptively by an MLP with two linear layers and one Softplus nonlinearity, which is defined as follows:

$$\delta = MLP(\mathbf{p}_i - \mathbf{p}_j). \tag{4}$$

Here $\mathbf{p}_i$ and $\mathbf{p}_j$ are the 3D point coordinates. We use the difference between the neighborhood point and the center point to learn the position encoding because it changes with the neighborhood point. As a comparison, when the neighborhood point and the center point are concatenated in the channel dimension, only half of the input changes with the change of the neighborhood point. Therefore, the subtraction relation can be used to learn better position encoding. We used position encoding to learn the attention vector and to integrate the features of each point, hoping that the points farther from the center would have a smaller proportion and a smaller value.

Normalization function is usually used to normalize attention vector. Traditional normalization function is set to be Softmax. In order to increase the robustness to noise, we use the normalization method in PCT [GCL*21]. Let $A = \{a_{i,j}\}$ be attention matrix learned by α. Firstly, softmax is used in the first dimension and the L_1 norm is used in the second dimension to normalize the attention matrix. The equations are as follows:

$$\bar{a}_{i,j} = \text{softmax}(a_{i,j}) = \frac{\exp(a_{i,j})}{\sum_k \exp(a_{k,j})} \tag{5}$$

$$\tilde{a}_{i,j} = \frac{\bar{a}_{i,j}}{\sum_k \bar{a}_{i,k}} \tag{6}$$

3.3. LOPCT

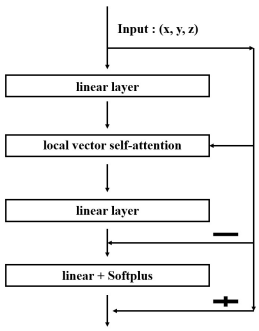

Figure 2: *Local offset point cloud transformer.*

We present a local offset point cloud transformer(LOPCT) with local vector self-attention as its module, as shown in Figure 2. LOPCT consists of the local vector self-attention layer, linear layers and a residual connection.

The input of LOPCT is a set of 3D coordinates $\mathbf{p}$. Firstly, the input point cloud is mapped to the d-dimension space by using the

linear layer to obtain the input embedding. The purpose of this is to make the input point cloud contain more semantic information. Then the local vector self-attention is used in the d-dimension space, which facilitates information integration between these localized input embedding producing new feature vectors for point cloud as its output. Semantic information fusion makes use of the rich information of the neighborhood effectively, so that each point in the point cloud is discriminative. When two points of an object are symmetric, the coordinates are different, but the semantics are the same. After the encoding of local vector self-attention, when two points are symmetric, their corresponding feature vectors are the same. Since each point integrates the semantic features of surrounding points, it can express the same semantics.

Then the linear layer is used to map back to the input space bridging the gap between the input point cloud and new feature vectors. After that, the difference between the input point cloud and self-attention feature matrix transformed by linear layer is fed into the fully connected layer followed by the Softplus nonlinearity. As pointed out by PCT [GCL*21], it is beneficial to use the difference between the input point cloud and the self-attention feature matrix as input to the integration module. Graph convolution learning has proved the validity of the Laplace matrix. The point cloud is regarded as a graph, and the self-attention feature matrix as an adjacency matrix, and the Laplace matrix is approximated by the difference between the point cloud and the self-attention feature matrix.

Finally, the output of the activation function and the input point cloud is added in a channel-wise way to get the final output, which could prevent input information loss during forward calculation.

3.4. Network

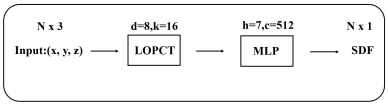

Figure 3: *Network framework for shape reconstruction. The input point cloud encoded by LOPCT, is fed into MLP and outputs SDF. K represents the number of nearest neighbors, D represents the input embedding dimension, H represents the number of hidden layers, and C represents the number of neurons at hidden layers.*

We present a complete network for point cloud reconstruction. Our network structure is shown in Figure 3. Given a point cloud extracted from a manifold surface, the feature-enhanced point cloud is first obtained through LOPCT and then fed into MLP to predict SDF.

We use a single LOPCT as feature aggregation module to fuse input point cloud layout information. We set input embedding dimension k is 8 and the number of neighborhood points is 16. The rationality of the selection is proved by the ablation experiment.

For MLP, we use seven hidden layers, each containing 512 neurons. We use Softplus activation functions behind each hidden layer.

The LOPCT module combines the advantages of local feature integration, graph convolution and robust normalization, effectively extracting and fusing the geometric information and deep semantic information of the input point cloud, and successfully solves the essential defect of MLP, that is, encoding each point in a point-wise manner without considering the mutual information between points.

3.5. Loss function and parameter setting

Suppose the following random experiment occurs: first suppose the original object consists of N points, subject to an unknown probability distribution $f(x_1, x_2, \cdots, x_N)$; Then, $I \leq N$ points are measured from the original object by the 3D sensor with normal information. Due to measurement errors, the points and normals obtained are $x^{'} = x + \varepsilon_1$, $n^{'} = n + \varepsilon_2$, where ε_1 and ε_2 obey an unknown distribution. Therefore, the point cloud obtained can be represented as $\chi = \{(x_i^{'}, n_i^{'})\}_{i=1}^{I}$.

The aim of this paper is to use MLP to learn the implicit function $F(x)$ of manifold surface. To learn the unknown parameters in the network, we use the loss function proposed in [WLYT21].

Given point cloud extracted from 3D object, $\chi = \{(x_i^{'}, n_i^{'})\}_{i=1}^{I}$, the implicit function $F(x)$ meets $F(x_i^{'}) = 0$, $\nabla F(x_i^{'}) = n_i^{'}, i = 1, \cdots, I$. So the loss function for above point cloud is

$$L_1 = \sum_{i=1}^{I} (F(x_i^{'})^2 + \tau \| \nabla F(x_i^{'}) - n_i^{'} \|^2). \tag{7}$$

To ensure $F(x)$ is an implicit function, the margin constraint $\|\nabla F(x)\| = 1$ [GYH*20] is used and the loss for points in the bounding box of 3D object is

$$L_2 = \lambda E_x (\|\nabla F(x)\| - 1)^2. \tag{8}$$

Therefore, the final loss function is in the following form:

$$L_{sdf} = L_1 + L_2. \tag{9}$$

After training, $F(x)$ approximates the potential SDF of the input point cloud, and we use the Marching Cubes to extract the zero level set of SDF to form polygonal mesh.

4. Experiment and evaluation

In this part, we introduce experimental settings in 4.1, experimental data sets and evaluation criteria are introduced in 4.2, some watertight object reconstruction results are showed in 4.3. We present computational resource consumption in 4.4 and experimental results for noisy data in 4.5. Complex topology reconstruction results are presented in 4.6. Finally, in 4.7, we present the ablation experiment results.

4.1. Experiment setting

Our implementation was based on PyTorch and all experiments were based on a GeForce 2080 Ti GPU (11GB of ram). For watertight object, we use all points for training, and for object with complex topology, we randomly select 50K points for training. For watertight object, we first fit the sphere and then initialize our network parameters with trained weights. For complex topology, we do not use sphere initialization. In each iteration of the training phase, 2k

Table 1: Reconstruction results of unstructured point cloud extracted from watertight object. CD are multiplied by 100000, MAE are multiplied by 100. MSPE stands for multiscale SPE and SPE* represents the result of not using multiscale optimization.

method	fandisk		bunny		dragon		gargoyle		armadillo	
	CD	MAE	CD	MAE	CD	MAE	CD	MAE	CD	MAE
IGR	1.35	1.43	1.75	0.43	1.43	**0.58**	**1.37**	**0.61**	**2.14**	**0.69**
SIREN	1.37	19.20	**1.48**	15.10	1.41	20.71	3.67	15.59	2.30	20.08
FPE	4983.33	24.44	8599.51	22.38	353.03	26.81	1243.33	22.21	126.00	29.10
MSPE	**1.34**	0.60	1.60	0.53	1.37	1.35	3.67	1.02	2.41	1.64
SPE*	-	27.38	-	20.33	-	25.30	-	20.15	-	26.41
Ours	1.35	**0.27**	1.60	**0.38**	**1.31**	1.67	3.63	1.30	2.18	1.37

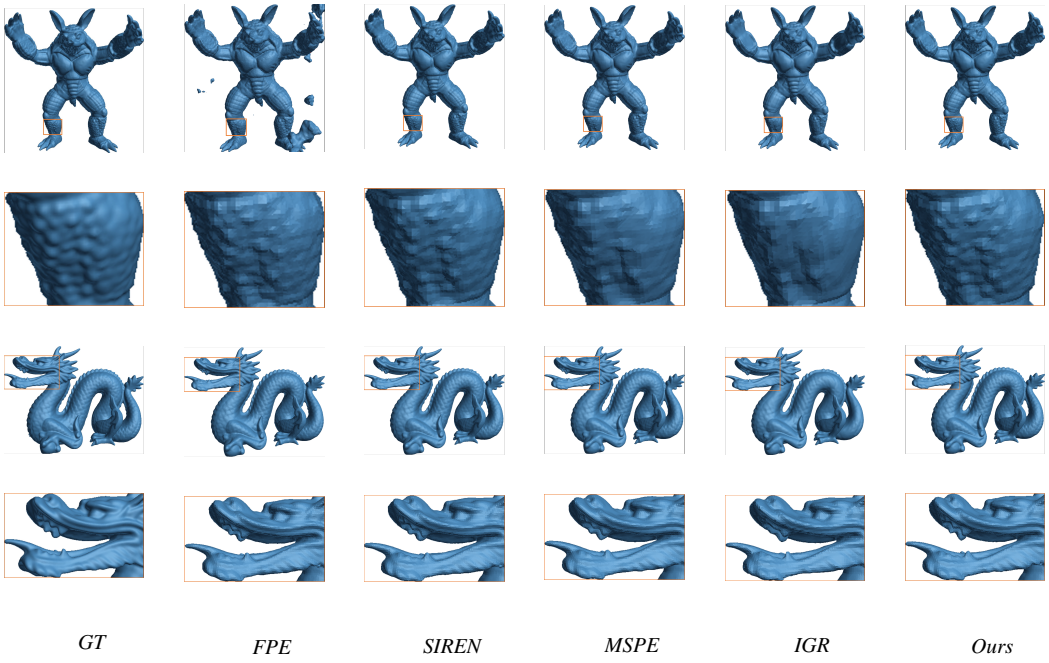

Figure 4: *Visual comparisons on SDF reconstruction from raw point clouds extracted from watertight object. The zero level set of SDF was extracted by marching cubes, and the polygon mesh of watertight object was obtained.*

points were randomly extracted from the input point cloud and the same number of points were extracted from the 3d bounding box containing the object. All inputs are encoded by LOPCT. We set the parameters of LOPCT to d=8 and k=16. The point features encoded by LOPCT are fed into MLP, and then the loss function is calculated. The parameters λ and τ are set to 0.1 and 1. LOPCT and MLP are optimized 20K epochs through Adam [KB14], and the learning rate was set at 1e-4.

4.2. Datasets and evaluation criteria

datasets To verify the validity of the method, five common watertight objects were selected: Armadillo, Bunny, Gargogle, Fandisk, Dragon, and six common complex topology were selected: Railing, Slim, Motor, Mould, Hole, and Part.

criteria In this paper, Chamfer Distance (CD), Hausdoff Distance (HD) and Average Normal Error (NAE) were used to measure the quality of extracted mesh. To calculate CD, HD, and NAE, we randomly sample a set of N points $\chi = \{x_i\}_{i=1}^{N}$, $M = \{n_i\}_{i=1}^{N}$ from the extracted surface and ground-truth surface $\hat{\chi} = \{\hat{x}_i\}_{i=1}^{N}$, $\hat{M} = \{\hat{n}_i\}_{i=1}^{N}$, for watertight object, N=320000, for object with complicated topology, N=50000. The expression of CD is:

Table 2: Computational resource requirements. Params represents space complexity and FLOPS stands for time complexity.

method	# Params	# FLOPs
IGR	2.10M	4.19G
FPE	0.26M	0.52G
SIREN	**0.20M**	**0.40G**
MSPE	1.08M	2.15G
Ours	1.84M	3.68G

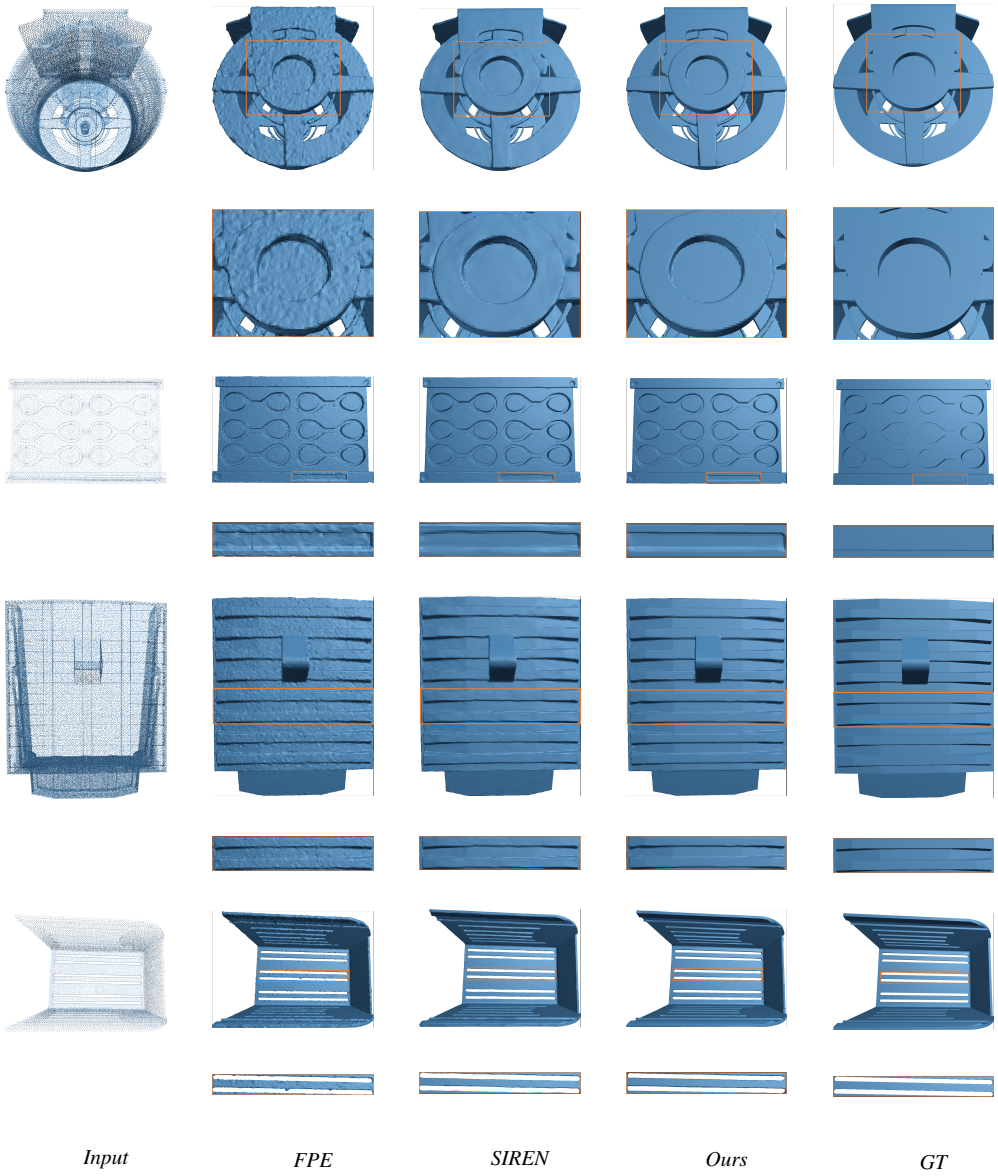

Figure 5: *Visual comparisons on SDF reconstruction from raw point clouds extracted from complex topology. The zero level set of SDF was extracted by marching cubes, and the polygon mesh of manifold was obtained.*

$$CD(\chi,\hat{\chi}) = \frac{1}{N}\sum_i \min_j \|x_i - \hat{x}_j\| + \frac{1}{N}\sum_j \min_i \|\hat{x}_i - x_j\|. \quad (10)$$

The expression of HD is:

$$HD(\chi,\hat{\chi}) = \max(\max_i \min_j \|x_i - \hat{x}_j\|, \max_j \min_i \|\hat{x}_i - x_j\|). \quad (11)$$

The expression of NAE is:

$$NAE(M,\hat{M}) = \frac{1}{N}\sum_i \min_j \|n_i - \hat{n}_j\| + \frac{1}{N}\sum_j \min_i \|\hat{n}_i - n_j\|. \quad (12)$$

In addition, this paper uses the Mean Absolute Error (MAE) between the predicted SDFs and the ground-truth SDFs to measure the quality of the predicted SDFs.

4.3. Watertight object reconstruction

The numerical results are summarized in Table 1, and the visual results are shown in Figure 4. It can be seen from the table that I-GR method has a strong topological recovery ability of watertight objects, but from the Figure 4, it fails to recover geometric details of dragon and armadillo. However, our method can recover both the correct topology and geometric details, indicating that LOPC-T can improve the geometric details recovery ability of MLP. For

Table 3: The reconstruction result of unstructured point cloud after adding (-0.018,0.018) uniform noise to normal vector. CD and HD are multiplied by 100000, NAE are multiplied by 1000 and MAE are multiplied by 100.

method	1%				2%			
	CD	HD	NAE	MAE	CD	HD	NAE	MAE
IGR	14.82	248.03	1.07	**0.86**	12.77	235.29	1.07	**0.83**
SIREN	62.32	2787.34	1.20	18.66	98.49	8373.19	2.05	18.63
Ours	**2.33**	**97.58**	**0.83**	0.93	**2.26**	**86.88**	**0.83**	0.92

Table 4: Reconstruction result of complex topology reconstruction. In order to improve the reconstruction efficiency, 50000 points were extracted from the original object for training, so the expression of complex topology was insufficient in some cases.

data	CD			HD			NAE		
	FPE	SIREN	Ours	FPE	SIREN	Ours	FPE	SIREN	Ours
railing	83154.96	83157.70	**83152.30**	199093.06	199093.70	**199081.03**	3.5375	0.71	**0.58**
slim	2581918.05	2581917.30	**2581898.58**	14389115.79	14389076.18	**14388999.12**	**4.2632**	5.67	5.72
hole	0.42	0.43	**0.41**	1.42	1.43	**1.37**	7.19	**2.50**	2.81
mould	0.01	0.01	**0.01**	0.01	0.01	**0.01**	6.69	**4.80**	5.16
motor	0.62	**0.60**	0.63	1.77	**1.74**	1.79	6.84	2.91	**2.15**
part	0.21	0.21	**0.20**	0.36	0.36	**0.34**	15.00	11.03	**9.09**

CD, our method achieves comparable results with SIREN and M-SPE, but for MAE, our method has better results, which shows that our method can recover SDF in good quality. Moreover, The object recovered from FPE has impurities and its CD index is relatively large.

4.4. Computational resource analysis

In this section we show the time complexity and space complexity of the different methods. For time complexity, we use floating point operation as a metric and for space complexity, we use the number of parameters of the model as a metric. The comparison results are shown in Table 2. As can be seen from the table, IGR has the highest space complexity and time complexity, while SIREN has the lowest. The main reason is that IGR uses more linear layers, while SIREN only uses the new activation function and reduces the use of linear layers. Compared with IGR, our method has lower space complexity and time complexity, which proves the effectiveness of our proposed LOPCT. Although FPE method has low time complexity and space complexity, the watertight objects recovered by them contain impurities.

4.5. Noisy data reconstruction

In order to demonstrate the robustness of the proposed method on real-world datasets, we add 1% or 2% uniform noise to the normal vector of watertight object mentioned in 4.2, and then use the noisy data for training. Every 200 epochs, we used the trained model to reconstruct the watertight object, then calculated CD, HD, NAE and MAE, and finally averaged the results. The numerical results are shown in Table 3. As can be seen from the table, our method is robust to noisy data as the noise level increases and consistently outperforms SIREN and IGR in terms of CD, HD and NAE. In addition, compared with IGR, our method has a faster convergence speed and can recover high-quality topologies at the early stage of training, which can be seen from the average results in the table.

4.6. complex topology reconstruction

Since IGR cannot recover complex topology and SPE relies on multi-scale training, we only compare with SIREN and FPE. The numerical results are summarized in Table 4, and the visual results are shown in Figure 5. For CD and HD, our method achieves the best results on multiple complex topologies, and for NAE, our method and SIREN achieve comparable results. It can be seen from the Figure 5 that comparing with the reconstruction of watertight objects, FPE has more advantages in the reconstruction of complex topology, but its recovery ability for sharp features is insufficient. For SIREN, although the plane recovery ability is better than that of FPE, the recovery of sharp features is also poor. However, our method can better recover the complex topology and sharp features. It shows that LOCPT module can effectively improve the recovery ability of complex topology and sharp features of MLP.

4.7. Ablation study

The results of 3d shape reconstruction were affected by network parameters and network structure. In order to verify the rationality of design of parameter and structure, a large number of ablation experiments of reconstruction of watertight objects were conducted. Every 200 epochs, we test the trained model and output the visual results, then calculate the CD, HD, NAE on the results, and finally take the average.

Our LOPCT module first maps the input to the high-dimensional space, and then does feature fusion in the high-dimensional space, and finally maps the fused features back to the original space. This

Table 5: The ablation experiment of Linear layer . CD and HD are multiplied by 100000, NAE are multiplied by 1000 and MAE are multiplied by 100. With indicates that a linear layer is used, and without indicates that a linear layer is not used.

data	CD		HD		NAE		MAE	
	with	without	with	without	with	without	with	without
fandisk	**1.45**	1.90	**9.97**	14.69	3.97	**3.72**	**0.27**	0.30
bunny	**1.68**	2.30	40.29	**32.07**	**0.10**	0.10	**0.38**	0.55
dragon	**1.45**	1.84	**68.38**	97.46	**0.04**	0.04	**1.67**	1.73
armadillo	2.47	**2.27**	**57.86**	59.48	**0.05**	0.05	1.30	**1.25**
gargoyle	4.33	**4.11**	**288.29**	330.33	**0.06**	0.06	1.37	**1.16**

is different from SIREN, FPE, and SPE, who will continue to learn SDF in higher dimensions. From the average results of Table 5, it can be seen that the recovery of topological structure and geometric details is better when linear layer is used. The effect is similar in terms of SDF recovery quality. But using a linear layer can narrowing the gap between self-attention feature matrix and input point cloud.

The numerical results of whether to initialize network parameters with a sphere are summarized in Table 6. The overall reconstruction results with sphere initialization are better than that without sphere initialization. This is because watertight object can be obtained by deforming spheres and weights trained on spheres can provide geometric prior information for the network. But in terms of normal vector recovery, the effect is similar.

Table 6: Ablation experiment of network parameters initialization. CD and HD are multiplied by 100000, NAE are multiplied by 1000. WithInit indicates that sphere initialization is used and withoutInit indicates that sphere initialization is not used.

method	CD	HD	NAE
withInit	**2.27**	**92.96**	0.84
withoutInit	2.48	106.81	**0.80**

For LOPCT module, different input embedding dimension d and the number of neighborhoods k will affect the final reconstruction effect. We take fandisk reconstruction results as an example to illustrate the influence of LOPCT module parameters. The average results are shown in Table 7. We found that the best results were achieved at d=8 and k=16. When d is equal to 8 and the number of neighborhoods k is equal to 8, 16, 32, the results become better first and then worse. This is because if there are too few neighborhood points, the mutual information between neighborhood points will be lost, and if there are too many neighborhood points, noise will be introduced. When the number of neighborhood points k is equal to 16 and d changes from 8 to 16, the result becomes worse. This is because the mutual information fusion in a higher dimension will produce a gap between the original space and feature space, which is not beneficial for MLP to learn SDF. We find that the increase of the number of neighborhood points has a favorable effect on normal vector reconstruction. This is because the number of points is too small to describe local topological relations.

Table 7: The numerical results of the reconstruction of fandisk. CD and HD are multiplied by 100000, NAE are multiplied by 1000 and MAE are multiplied by 100.

(d,k)	(8,8)	(8,16)	(16,8)	(16,16)	(8,32)
CD	1.50	**1.45**	1.63	1.58	1.48
HD	11.73	**9.97**	11.47	10.74	10.75
NAE	3.95	3.97	4.00	3.93	**3.87**
MAE	0.34	**0.27**	1.90	1.44	0.35

5. Conclusion

In this paper, LOPCT is proposed, which can effectively solve the essential problem of MLP dealing with the point cloud in a pointwise way, so that the sharp features and geometric details of watertight object and complex topology can be well recovered, and the comparable and better results can be obtained. LOPCT has a strong ability to recover geometric details and sharp features. The correct topology can be recovered at the early stage of network iteration and geometric details and sharp features can be recovered with the update of network parameters. In addition, our method also shows robustness in the presence of noise in the normal vector.

6. Acknowledgements

This work was partially supported by the National Key Research and Development Program of China, No. 2020YFA0713703 and Fundamental Research Funds for the Central Universities.

References

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE Transactions on visualization and computer graphics 9*, 1 (2003), 3–15. 1

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new voronoi-based surface reconstruction algorithm. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (1998), pp. 415–421. 1

[ACK01] AMENTA N., CHOI S., KOLLURI R. K.: The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications* (2001), pp. 249–266. 1

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 67–76. 1

[DTB06] DIEBEL J. R., THRUN S., BRÜNIG M.: A bayesian method for probable surface reconstruction and decimation. *ACM Transactions on Graphics (TOG) 25*, 1 (2006), 39–59. 1

[DTS01] DINH H. Q., TURK G., SLABAUGH G.: Reconstructing surfaces using anisotropic basis functions. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001* (2001), vol. 2, IEEE, pp. 606–613. 1

[EGO*20] ERLER P., GUERRERO P., OHRHALLINGER S., MITRA N. J., WIMMER M.: Points2surf learning implicit surfaces from point clouds. In *European Conference on Computer Vision* (2020), Springer, pp. 108–124. 1

[FCOS05] FLEISHMAN S., COHEN-OR D., SILVA C. T.: Robust moving least-squares fitting with sharp features. *ACM transactions on graphics (TOG) 24*, 3 (2005), 544–552. 1

[GCL*21] 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*, 2 (2021), 187–199. 2, 3, 4

[GYH*20] GROPP A., YARIV L., HAIM N., ATZMON M., LIPMAN Y.: Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099* (2020). 1, 2, 4

[HJCX21] HAN X.-F., JIN Y.-F., CHENG H.-X., XIAO G.-Q.: Dual transformer for point cloud analysis. *arXiv preprint arXiv:2104.13044* (2021). 2

[HKX21] HAN X.-F., KUANG Y.-J., XIAO G.-Q.: Point cloud learning with transformer. *arXiv preprint arXiv:2104.13636* (2021). 2

[HSD00] HART P. E., STORK D. G., DUDA R. O.: *Pattern classification*. Wiley Hoboken, 2000. 1

[HYC*21] HUI L., YANG H., CHENG M., XIE J., YANG J.: Pyramid point cloud transformer for large-scale place recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 6098–6107. 2

[IJS03] IVRISSIMTZIS I., JEONG W.-K., SEIDEL H.-P.: Using growing cell structures for surface reconstruction. In *2003 Shape Modeling International.* (2003), IEEE, pp. 78–86. 1

[JWB*06] JENKE P., WAND M., BOKELOH M., SCHILLING A., STRASSER W.: Bayesian point cloud reconstruction. In *Computer Graphics Forum* (2006), vol. 25, Wiley Online Library, pp. 379–388. 1

[Kaz05] KAZHDAN M.: Reconstruction of solid models from oriented point sets. In *Proceedings of the third Eurographics symposium on Geometry processing* (2005), pp. 73–es. 1

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 2, 5

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing* (2006), vol. 7. 1

[KH13] KAZHDAN M., HOPPE H.: Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG) 32*, 3 (2013), 1–13. 1

[KNH*21] KHAN S., NASEER M., HAYAT M., ZAMIR S. W., KHAN F. S., SHAH M.: Transformers in vision: A survey. *ACM Computing Surveys (CSUR)* (2021). 2

[MAVDF05] MEDEROS B., AMENTA N., VELHO L., DE FIGUEIREDO L. H.: Surface reconstruction for noisy point clouds. In *Symposium on Geometry Processing* (2005), Citeseer, pp. 53–62. 1

[MPS08] MANSON J., PETROVA G., SCHAEFER S.: Streaming surface reconstruction using wavelets. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1411–1420. 1

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision* (2020), Springer, pp. 405–421. 1, 2

[OBA*05] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.-P.: Multi-level partition of unity implicits. In *Acm Siggraph 2005 Courses*. 2005, pp. 173–es. 1

[ÖGG09] ÖZTIRELI A. C., GUENNEBAUD G., GROSS M.: Feature preserving point set surfaces based on non-linear kernel regression. In *Computer graphics forum* (2009), vol. 28, Wiley Online Library, pp. 493–501. 1

[RBA*19] RAHAMAN N., BARATIN A., ARPIT D., DRAXLER F., LIN M., HAMPRECHT F., BENGIO Y., COURVILLE A.: On the spectral bias of neural networks. In *International Conference on Machine Learning* (2019), PMLR, pp. 5301–5310. 1

[RJT*05] REUTER P., JOYOT P., TRUNZLER J., BOUBEKEUR T., SCHLICK C.: Surface reconstruction with enriched reproducing kernel particle approximation. In *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics, 2005.* (2005), IEEE, pp. 79–87. 1

[SBS05] SCHALL O., BELYAEV A., SEIDEL H.-P.: Robust filtering of noisy scattered point data. In *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics, 2005.* (2005), IEEE, pp. 71–144. 1

[SMB*20] SITZMANN V., MARTEL J., BERGMAN A., LINDELL D., WETZSTEIN G.: Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems 33* (2020), 7462–7473. 1, 2

[SSB05] STEINKE F., SCHÖLKOPF B., BLANZ V.: Support vector machines for 3d shape processing. In *Computer Graphics Forum* (2005), vol. 24, Citeseer, pp. 285–294. 1

[TO05] TURK G., O'BRIEN J. F.: Shape transformation using variational implicit functions. In *ACM SIGGRAPH 2005 Courses*. 2005, pp. 13–es. 1

[TSM*20] TANCIK M., SRINIVASAN P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHI R., BARRON J., NG R.: Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems 33* (2020), 7537–7547. 1, 2

[VSP*17] VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER Ł., POLOSUKHIN I.: Attention is all you need. *Advances in neural information processing systems 30* (2017). 2

[WJC*22] WANG X., JIN Y., CEN Y., WANG T., TANG B., LI Y.: Lightn: Light-weight transformer network for performance-overhead tradeoff in point cloud downsampling. *arXiv preprint arXiv:2202.06263* (2022). 2

[WLYT21] WANG P.-S., LIU Y., YANG Y.-Q., TONG X.: Spline positional encoding for learning 3d implicit signed distance fields. *arXiv preprint arXiv:2106.01553* (2021). 2, 4

[WWD*21] WANG Z., WANG P., DONG Q., GAO J., CHEN S., XIN S., TU C.: Neural-imls: Learning implicit moving least-squares for surface reconstruction from unoriented point clouds. *arXiv preprint arXiv:2109.04398* (2021). 1

[YS01] YAN J.-Q., SHI P.-F.: Surface reconstruction for 3d objects from unorganized points. In *Visualization and Optimization Techniques* (2001), vol. 4553, International Society for Optics and Photonics, pp. 290–295. 1

[ZBDB19] ZHONG E. D., BEPLER T., DAVIS J. H., BERGER B.: Reconstructing continuous distributions of 3d protein structure from cryo-em images. *arXiv preprint arXiv:1909.05215* (2019). 1, 2

[ZJJ*21] ZHAO H., JIANG L., JIA J., TORR P. H., KOLTUN V.: Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 16259–16268. 2, 3