The Visual Computer
https://doi.org/10.1007/s00371-023-03003-9

ORIGINAL ARTICLE

MEAN: An attention-based approach for 3D mesh shape classification

Jicheng Dai[1] · Rubin Fan[1] · Yupeng Song[1] · Qing Guo[1] · Fazhi He[1,2]

Accepted: 23 June 2023

Abstract
3D shape processing is a fundamental computer application. Specifically, 3D mesh could provide a natural and detailed way for object representation. However, due to its non-uniform and irregular data structure, applying deep learning technologies to 3D mesh is difficult. Furthermore, previous deep learning approaches for 3D mesh mainly focus on local structural features and there is a loss of information. In this paper, to make better mesh shape awareness, a novel deep learning approach is proposed, which aims to full-use the information of mesh data and exploit comprehensive features for more accurate classification. To utilize self-attention mechanism and learn global features of mesh edges, we propose a novel attention-based structure with the edge attention module. Then, for local feature learning, our model aggregates edge features from adjacent edges. We refine the network by discarding pooling layers for efficiency. Thus, it captures comprehensive features from both local and global fields for better shape awareness. Moreover, we adopt spatial position encoding module based on spatial information of edges to enhance the model to better recognize edges and make full use of mesh data. We demonstrate effectiveness of our model in classification tasks with numerous experiments which show outperforming results on popular datasets.

Keywords 3D mesh · 3D shape processing · Self-attention mechanism · Non-local features

1 Introduction

3D shapes play an important role in various scenes of computer vision and graphics [1–3], computer-aided design/co mputer-aided engineering (CAD/CAE) [4–7], virtual reality [8, 9], simulation [10] and 3D printing [11, 12]. One typical 3D shape representation is the 3D mesh. To get an approximation of smooth surfaces, mesh is defined as a set of successive polygons (often triangles) in 3-dimensional space and is composed of vertices, edges, and faces as primitives. Additional information like texture is optional for a better rendering (Fig. 1).

In mesh data, rich information including coordinates of vertices and surface connectivity is stored. With appropriate algorithms [13], mesh has more flexibility and potential to represent smooth surfaces of real 3D models when high resolution and quality are needed. However, due to the difficulties of manual acquisition and the non-uniform structure, it is a challenging task for mesh shape analysis [14]. These obstacles lead to the fact that there are only a few deep learning methods for 3D mesh analysis while numerous works like Pointnet [15], Song et al. [16], and VoxNet [17] have been applied to other 3D representations. Furthermore, the existing 3D mesh analysis models mainly focus on the features in a local region because it's natural to exploit the connectivity between mesh primitives. However, the absence of non-local relations and features may be harmful to these models to understand the shape. In addition, many mesh analysis methods based on edges don't make full use of the rich information that mesh contains. The above shortages may lead to incomplete awareness of the mesh object.

In detail, the following manuscript shows the motivation of this paper.

- Compared with other 3D representations, mesh contains both coordinates as spatial information and surface connectivity as structural information, which makes it more powerful to represent 3D objects. However, in existing approaches, either spatial or structural information is overlooked which causes the loss of information.
- Typical previous 3D mesh analysis, such as MeshCNN [18], MedmeshCNN [19], considers the explicit connec-

✉ Fazhi He
fzhe@whu.edu.cn

1 School of Computer Science, Wuhan University, Bayi Road, Wuhan 430072, China

2 National Engineering Research Center for Multimedia Software, Bayi Road, Wuhan 430072, China

Published online: 16 July 2023

tivity between mesh primitives and provides a natural way for feature aggregation and convolution. However, these models overlook the global features. In addition, these methods may lead to a severe efficiency drop because the pooling operations serially access and update the mesh structure at high frequency.

- Self-attention mechanism is becoming a hot topic in 2D image processing [20] and other 3D representations [21, 22] with promising capabilities in non-local feature extraction. As mentioned above, existing approaches for 3D mesh analysis mainly focus on local features and lead to incomplete awareness of the mesh shape. Thus, how to leverage the self-attention to the irregular structure of mesh with global feature extraction is unknown.

In this paper, inspired by the above motivations, we propose a novel structure, namely mesh edge attention network (MEAN) for mesh shape analysis, which aims to learn comprehensive features of mesh and exploit both spatial and structural information. We define local feature aggregation block and edge attention block to expand the sight of machine to all components of mesh and learn comprehensive features, overcoming shortages of existing models. Experimental results show that the proposed MEAN outperforms the recent methods for mesh classification tasks.

In summary, the main contributions of our work are as follows:

- We propose a novel structure, which introduces the self-attention mechanism by utilizing edge attention blocks to capture the semantic affinities between mesh edges and exploit non-local features from all the input edges for better shape awareness.
- We design the edge feature aggregating block on the features generated from adjacent mesh edges for input embedding, which enables the network to capture local geometric features. Especially, we discard the time-consuming pooling operations in previous mainstream models, and our MEAN gets a significant improvement on time efficiency and also achieves advanced performance.
- A spatial encoding module is proposed to fully use the information of mesh. Spatial information is helpful for our network to recognize edges and learn more effective features than the previous edge-based models which focus on geometric features only.
- We conduct numerous experiments on classification tasks to evaluate the effectiveness and robustness of the proposed network. On three mainstream datasets, MEAN outperforms recent popular methods. Ablation experiments illustrate that MEAN is still effective in challenging cases such as non-uniformity of edge lengths and lacks of training samples.

The paper is organized as follows. In Sect. 2, Related work, we summarize the development of 3D deep learning methods. Section 3 presents the ideas and details of MEAN. Section 4 illustrates the performance of MEAN for classification and analysis with ablation experiments. Finally, Sect. 5 gives the conclusion and future works.

2 Related work

In this section, firstly we summarize the works in recent years of mesh deep learning. Then, we introduce some typical approaches applying self-attention mechanism for other 3D representations.

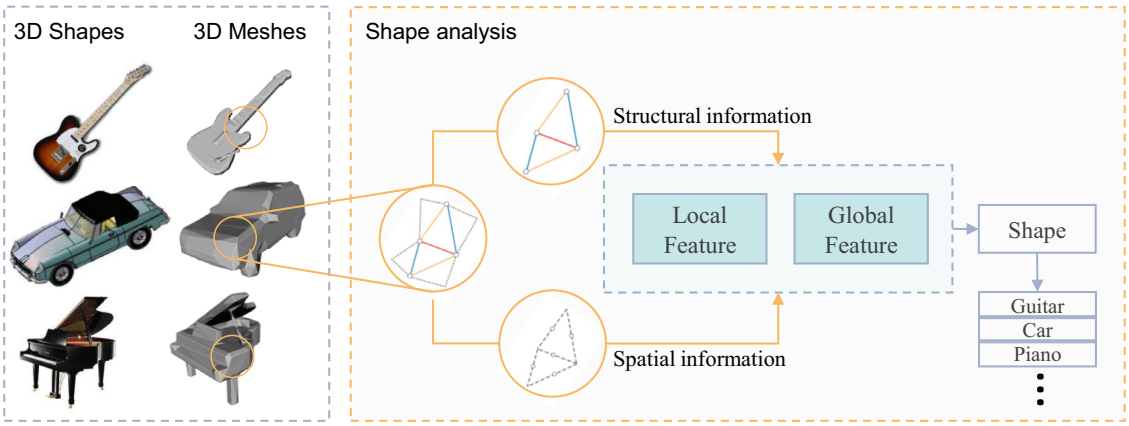


Fig. 1 3D models in reality and their corresponding 3D meshes. Our proposed method learns local and global features from structural and spatial information for mesh shape analysis

2.1 Mesh deep learning

In this subsection, we will summarize recent works of mesh understanding based on deep learning methods. Firstly, we will introduce the models which do not directly apply networks to mesh shape. Then, we will discuss deep learning methods directly based on the primitives of mesh, and there are three main categories: vertices, edges, and faces.

Converting mesh to other structures. The structure of 3D mesh is not irregular; thus, the networks designed for 2D image could not be directly applied to 3D mesh shape understanding. One popular way is converting the mesh structure to other data structures with the neighborhood relations of mesh primitives. Wang [23] applies graph convolutional neural networks (GCNs) and pooling on the graphs generated by 3D mesh and builds the networks with full convolutional network for mesh segmentation. FeastNet [24] designs graph-convolution operator to establish correspondences between graph neighborhoods with arbitrary connectivity. Another series of models encode the mesh to Laplacian space before convolution. Qiao [25] and Laplacian2Mesh [26] maps the input mesh surface to the multi-dimensional Laplacian–Beltrami space and applies convolution for shape analysis. Further works like Laplacian Mesh Transformer [27] applies transformer for both geometric and structural features based on Laplacian space.

Vertex-based methods. Methods learning mesh features from vertices are similar to methods for 3D point cloud. However, the points in mesh are much sparser while more connectivity is provided. One series of works aims to apply convolution on mesh vertices. The idea of PointNet [15] and PointNet [28] to sample the points in a neighboring domain and apply convolution is instructive for mesh deep learning because the connectivity between mesh vertices provides a natural way for such operations. Chen [29] adopts graph neutral networks on vertices to utilize the connectivity between mesh vertices. DiffusionNet [30] and HodgeNet [31] introduce the Laplacian operator to mesh surface for shape awareness.

Another series of methods do not apply convolution operations but introduce the idea of DeepWalk [32] by randomly walking on the mesh surfaces, exploring the geometry and topology of the object. MeshWalker [33] adopts recurrent neural network (RNN) on the walk sequences which are from randomly walking on the connected vertices. And the extensional work AttWalk [34] learns to weigh the features of different walks. And the importance of different regions is calculated by the walk-attention mechanism.

Edge-based models. Learning shape awareness from edge features is also an important category of mesh deep learning. For triangle mesh, each face has exactly three vertices and three edges which explicitly provides connectivity and topological information like angles and length ratios. MeshCNN [18] defines neighbors of edge in a manifold mesh, applying convolution function on these edges to aggregate features. And the edge pooling operation enables edge collapse to downsample the mesh, which folds edges and their neighbors and generates a new abstract object. PDMeshNet [35] defines Primal Graphs and Dual Graphs, applying convolutions using graph attention networks (GAT) for mesh shape awareness [36].

These edge-based models focus mainly on the local structural features in a regular domain and rely on the pooling operation to expand the receptive field. In this paper, we propose our method to overcome these shortages by learning global features from structural and spatial information without pooling operation.

Face-based models. Face-based models try to gather face information from neighbors. The faces of mesh could store rich structure information, and the coordinates of face center also provide spatial information. Methods based on face could utilize different types of information to generate mesh shape descriptors. MeshSNet [37] provides a graph-constrained learning module to extract multi-scale features. MeshNet [38] defines convolution on the neighbors of mesh face to combine and aggregate the spatial and structural information of mesh, whose initial descriptors are generated from face rotation and kernel correlation. Xu [39] defines a two-stream segmentation framework with the face normals and face distance histograms as input for segmentation tasks. Based on subdivision connectivity, SubdivNet [40] defines a general convolution operation on mesh surfaces which is very analogous to 2D convolution and permits variable kernel size, stride, and dilation.

The above three categories of existing models for mesh deep learning still have limitations. And many of these models mainly focus on local structural features.

2.2 Self-attention for 3D deep learning

The original self-attention mechanism is proposed for sentence translation in natural language processing [41, 42] and then becomes a mainstream approach in this field with various extension works like BERT [43], Si et al. [44] and YDTR [45].

In recent years, many researchers introduce this idea to computer graphics and computer vision and make progress. For 2D images, ViT [20] processes image pixels to small patches and calculates the scale-dot product and generates feature maps. TransUNet [46] and Segformer [47] use the idea for semantic segmentation tasks. A comprehensive review of these works is provided in recent surveys [48, 49]. For 3D data representations, the structure PCT [21] and Point Transformer [50] treat the disordered input point clouds as

sentences and calculate the semantic similarities between points. It adopts offset attention as an analogy of the discrete Laplacian operator for better performance. PointBert [22] tokenizes the local structure in subsets of point cloud and adopts random masks for these sub-clouds. It learns global features by predicting and reconstructing the point clouds.

For specific tasks like human pose transfer, Lin [51] presents a transformer to learn non-local relation between vertices and joints. Mesh Graphformer [52] combines graph convolution and self-attention in a transformer to reconstruct 3D mesh from a single image.

This paper addresses the 3D mesh classification with self-attention mechanism for comprehensive feature learning from mesh edges. Specifically, we propose a novel structure that adopts both local feature aggregation and edge attention to capture non-local relations based on self-attention mechanism for better mesh shape awareness.

3 The proposed approach

In this section, we present the details of our network which learns features from mesh edges for classification tasks. Firstly, we present the overview of our pipeline for a brief understanding of the network. Then, we give the definition of input and the core of our model in detail about how to gather local and global features, which enables our network to learn comprehensive features.

3.1 Network overview

Existing deep learning methods for 3D mesh mainly focus on the local features from vertices, edges or faces while global features are often overlooked. In this paper, to better learn comprehensive features of mesh, we propose a novel structure MEAN to learn from mesh edges. The proposed MEAN is composed of two parts, as shown in Fig. 2.

Feature Learning. In the feature learning network, MEAN aims to learn the global features as well as local features. As shown in Fig. 2, the proposed pipeline adopts three aggregation blocks in input embedding module firstly, which will be described in detail in Sect. 3.2. These blocks enable the networks to capture local features in the receptive field and finally process the input structural information descriptor to a 256-dimensional vector. Then, these features are fed to edge attention module to learn global features, which is composed of 4 sets of edge attention blocks and a spatial position encoding module utilizing spatial information of edges. The details of edge attention module will be discussed later in Sect. 3.3. Then, we add up global features of the 4 attention blocks and local features of input embedding module to 256-dimensional comprehensive edge features. Finally, we will feed these features to the following max-pooling and average-pooling layer and concatenate the outputs to a 512-dimensional vector, which is the general descriptor for the mesh.

Classification Network. To classify a mesh from n_c different categories, we feed the 512-dimensional output vector F_{out} from the feature learning network to three cascaded full-connected layers with a dropout rate $\mathcal{P}_d$. And each of the full-connected layers combines the linear projection, the batch normalization layer and the $Relu$ activation in sequence. Finally, the last single full connected layer will predict the classification scores and the category of mesh depending on the maximum score.

3.2 Input embedding

Unlike other 3D representations, mesh data have complex and irregular structures. However, the connectivity between different edges gives us a natural way to find neighbors of edges. In this paper, inspired by previous work [18, 35], to learn local geometric information, convolution is applied on defined neighbors and convolution invariance is guaranteed. The difference is that we discard the pooling functions because the edge collapse process is much time-consuming. Moreover, residual links are added in input embedding blocks to make our network deeper.

Definition of input. The raw input of a mesh M is described as a triple (V, F, E), where $V = \{v_1, v_2, v_3, ...v_n\}$ is the set of vertices. Each v_i describes the coordinate of the vertex in 3D space, namely $v_i \in \mathbb{R}^3$. And F denotes the set of faces, using triples to define the connectivity of vertices. Generally, for face $F_i = \{(v_a, v_b, v_c)\}$, the triangle face is composed of these three vertices. Then, given (V, F), the edge set E is defined as the distinct edges of each two connected vertices of faces. Note that $E_i = \{(v_a, v_b)\}$ and $E_j = \{(v_b, v_a)\}$ refers to the same edge.

In MEAN, all the input meshes should be under the constraint of manifold, which guarantees that each edge is incident to two different triangle surfaces. Therefore, there are four adjacent edges. Other faces and edges will be removed in data preprocessing if they are non-manifolds. After manifold is satisfied, we define the four neighbor edges counterclockwise, namely as (e_1, e_2, e_3, e_4) or (e_3, e_4, e_1, e_2). As shown in Fig. 3, we could see that for edge e, the order of neighbors only depends on which face is the first (Fig. 4).

Our model is edge-based, and a set of initial geometric features is defined as a 5-dimensional vector for every edge [18]: the dihedral angle, $D_d \in (0, 2\pi)$; two inner angles, $D_{i1}, D_{i2} \in (0, 2\pi)$ and the two ratios of edge lengths, $r_1, r_2 > 0$ for each of the perpendicular line for each adjacent face. Fig 5 shows these features.

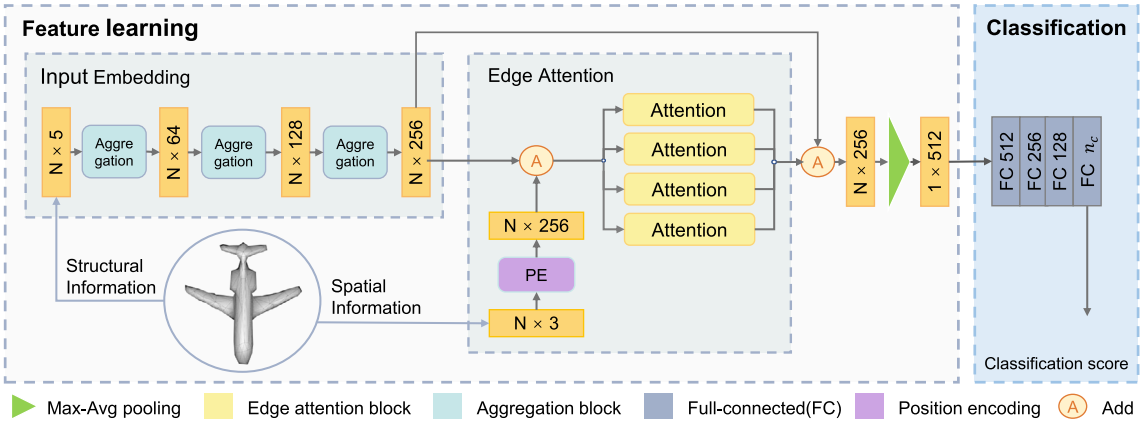
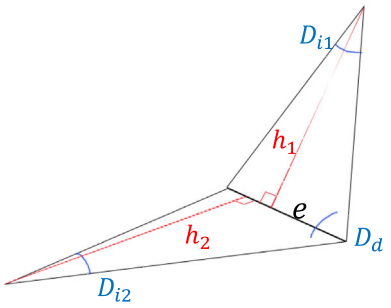


Fig. 2 The overall structure of MEAN. N refers to the number of mesh edges and n_c refers to the number of categories. MEAN learns comprehensive features via the input embedding module and edge attention module, from structural and spatial information for better awareness. Best viewed in color



$$r_1 = \frac{h_1}{e}$$

$$r_2 = \frac{h_2}{e}$$

Fig. 3 Initial features extracted from mesh edges. D_{i1} and D_{i2} are two inner angles, D_d is the dihedral angle. h_1 and h_2 are the perpendicular lines to e of each face (painted in red)

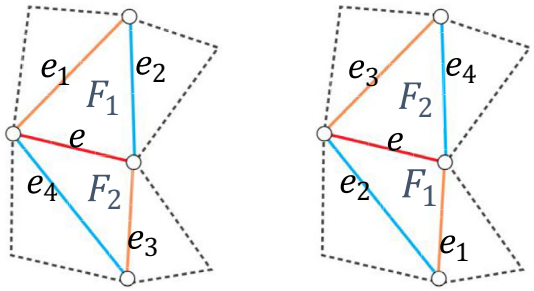


Fig. 4 Two probabilities of edge orders. Depending on which face is the first one, neighbor edges of e are defined as (e_1, e_2, e_3, e_4) or (e_3, e_4, e_1, e_2)

Aggregating edge features. To aggregate structural information of edges, a convolution function is defined for input edges of mesh. As mentioned above, each input edge of manifold has exactly four neighbor edges. Depending on which face counts the first, there are two possible forms for the neighbors of an edge. In order to guarantee the convolution invariance, several symmetric functions are applied to these neighbors. As a result, the four neighbor edge that is the receptive field of an edge e is defined as follows.

$$Neighbor(e) = (|e_1 - e_3|, |e_1 + e_3|, |e_2 - e_4|, |e_2 + e_4|) \tag{1}$$

For the input vector of $N \times n_d$, where N refers to the number of mesh edges, and n_d refers to the number of feature dimensions, we could easily find the four neighbors when the connectivity is given. Then, we attach these neighbor features to the input vector to get a $N \times n_d \times 5$ vector where 5 refers to the edge and its four neighbors. We could calculate the input features from four adjacent edges neighbors before convolution and then multiply the new matrix by the weight of convolution and get results. After carefully avoiding ambiguity, no matter in which order we input the faces, the convolution operation is actually applied to the same features from neighbor edges.

$$e' = e \times k_0 + \sum_{i=1}^{4} Neighbors(e)[i] \times k_i \tag{2}$$

In practice, calculating neighbor edge features by some symmetric functions in $\mathbb{R}^{N \times n_d \times 5}$, we could use the operator $conv2d$ with 1×5 kernel size to implement this convolution and get the output in $\mathbb{R}^{N \times n_{d'} \times 5}$, where d' refers to the output dimensions.

Discard of edge pooling. In a convolution-based network, especially for 2D images, pooling layers are often following convolution layers for downsampling. After pooling operations, it will generate an abstract feature map in high dimensions and the whole computation cost will be reduced while the receptive field is rapidly expanded. However, we could not directly use the typical maximum or average pooling from 2D images for mesh edges due to the irregular structure of 3D mesh.

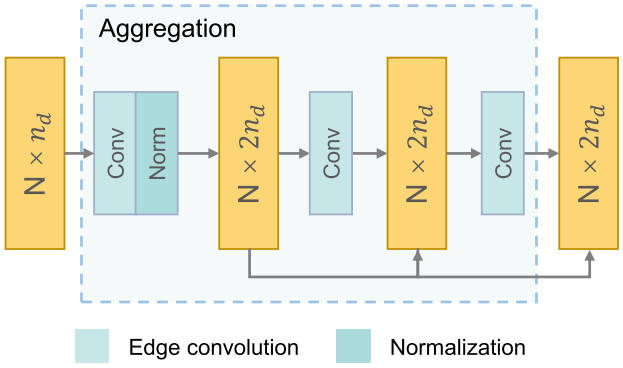


Fig. 5 The structure of feature aggregation block. Residual links are helpful for making deeper networks, and the output dimensions are doubled after aggregation

Although MeshCNN [18] defines an edge-collapse pooling function for mesh abstracting, this function is rather time-consuming which serially accesses the edge matrix and applies a series of operations to reduce the edge number.

In MEAN, we discard the pooling layers in our input embedding module. At the first glance, the absence of pooling may slow the expansion of receptive fields and lead to deterioration of effectiveness. However, the overall structure of MEAN benefits from global feature extraction and compensates for this deterioration.

In Sect. 4, experiments will demonstrate that there is a significant efficiency improvement compared with the original MeshCNN with pooling layers when effectiveness is guaranteed.

Residual links in aggregation block. We add residual links between convolution operations to make our network deeper. The deeper network can learn more hidden features in high dimensions. These links are also helpful for avoiding the degeneration of our networks [53]. As shown in Fig. 5, for the input vector to output vector in one convolution block, the shape is expanded from $N \times n_d$ to $N \times 2n_d$, where N refers to the edge number and n_d is the channel number.

As shown in Fig. 5, convolutions are applied several times in one block and only the first one will double the feature channels. Others will output a vector in the same dimensions as the input. For the second and following outputs after convolution and normalization, initial inputs are added through a residual link (Fig. 6).

$$\begin{aligned} F_i &= Norm(ReLU(conv(F_{i-1})), i = 1 \\ F_i &= Norm(ReLU(conv(F_{i-1})) + F_{i-1}, i > 1 \end{aligned} \tag{3}$$

3.3 Edge attention

To extend the original self-attention mechanism from NLP [41, 42] to 3D mesh, we input all the mesh edges as a sequence and each edge could be treated as a word in a sentence. In Sect. 3.2, we get the 256-dimensional input embeddings via the input embedding module which learns local geometric features from neighbor edges in the receptive field. In consideration of non-local features, we design the edge attention module (Fig. 7).

3.3.1 Spatial encoding

The original self-attention mechanism [41] and Transformer [42] process the sequence of words; thus, the position of each word in the sentence is vital to the semantics. And transformers for images like ViT [20] also adopt position encodings on pixels patches because the order of pixels is disrupted when there is no gird structure. However, in Transformers for 3D data like PCT [21], different input orders of points, or the primitives of mesh do not make difference to the structure of this object. The structure of a mesh object is mainly defined by the connectivity of its vertices and faces. Therefore, we do not follow the encoding methods in original Transformer and propose a special position encoding module.

Many of the existing methods [18, 40] mainly focus on the geometric feature of mesh edge, overlooking the spatial information of edge position. For better awareness of mesh, we aim to make full use of the information in mesh and propose a novel position encoding module for MEAN. This module utilizes the spatial position of mesh edges in 3D space to get an extra learnable position encoding. These encodings are adopted as additional marks for edge attention module to measure the semantic affinities between mesh edges.

The spatial position encoding module is composed of two simple MLP layers, encoding the edge coordinates to higher dimensions. Considering an edge $e_i = \{(v_a, v_b)\}$, we could use the midpoint coordinates of the edge in $\mathbb{R}^3$ as spatial position. To make our network robust to scaling and translation, all the meshes are normalized in data preparation.

In our settings, the output channel numbers of each MLP are $n_{PE_1} = 64, n_{PE_2} = 256$, respectively. Then, the position features F_{PE} are directly added to input embedding F_{IE} as the input feature F_{in} of edge attention module.

$$F_{in} = F_{PE} + F_{IE} \tag{4}$$

Another possible approach is adding the 3-dimension coordinates to the initial 5-dimension edge features directly. However, coordinates could not guarantee convolution invariance when mesh is shifted, rotated or scaled. Coordinates of edges will change when these rigid transformations are applied to mesh. Thus, we design this position encoding module instead of directly concatenating the spatial information to initial descriptors. We evaluate different strategies of using position encoding in Sec 4.3.

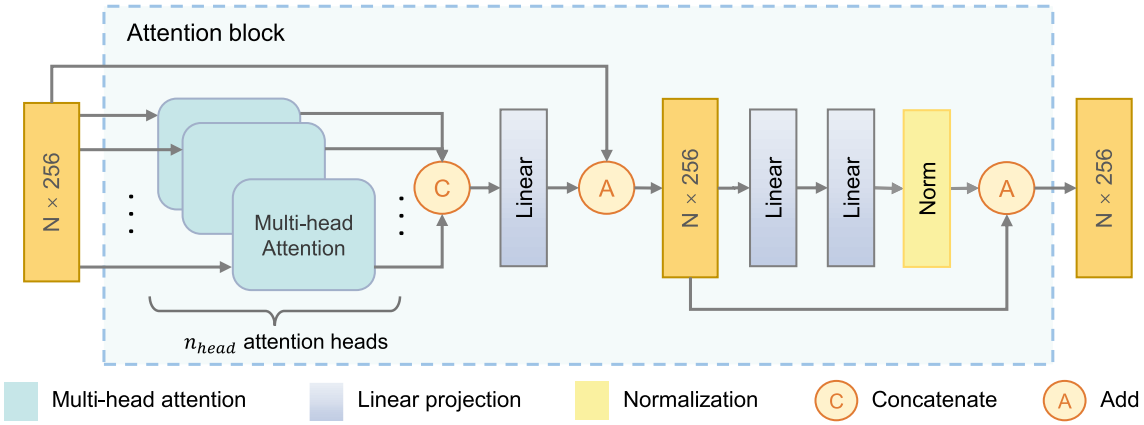


Fig. 6 The architecture of edge attention block. We adopt scale-dot product attention and multi-head attention where n_head refers to the number of heads. In the proposed model, there are 4 edge attention blocks for better performance

In summary, the main difference between our spatial encoding and position encoding in transformers is that the input of our spatial encoding module is the spatial information of mesh edges, while the position encoding in other transformers often encode the order of words or pixel patches.

3.3.2 Edge attention block

As discussed above, we get the 256-dimensional feature F_{in}. It contains the local geometric features of each edge, which are beneficial to identify and classify the mesh, and the spatial information is also utilized as additional marks. In this section, we will introduce the edge attention block, which learns semantic affinities between edge pairs and captures global features. For each input edge, we will calculate the attention weight using the typical scaled dot-product attention.

Firstly, we use linear projection to define the query, key and value matrices Q, K, V from input features F_{in} as follows, where W_q, W_k, W_v are learnable weights of linear projection. Also, MEAN adopts multi-head attention in typical transformers [20, 22, 42], which projects F_{in} to n_{head} groups of Q_i, K_i, V_i and creates n_{head} attention heads. Each attention head calculates the attention weights in a lower dimension, respectively.

$$\begin{aligned} Q_i &= F_{in} \cdot W_{q_i} \\ K_i &= F_{in} \cdot W_{k_i} \\ V_i &= F_{in} \cdot W_{v_i} \\ i &= 1, 2, ..., n_{head} \end{aligned} \tag{5}$$

For each attention head, the affinities and similarities between mesh edges are calculated as the dot-product of the query matrix and key matrix, divided by $\sqrt{n_d}$. We adopt the $softmax$ function as normalization; hence, we get the importance of each edge to others which ranges from 0 to 1. As shown in Eq. 6, we get the attention weight $Attention_i$ of head. And the output feature F_{att_i} of one head is the product of its attention weight and value matrix V_i. In F_{att_i}, the new feature of each edge is the sum of the products of other edge

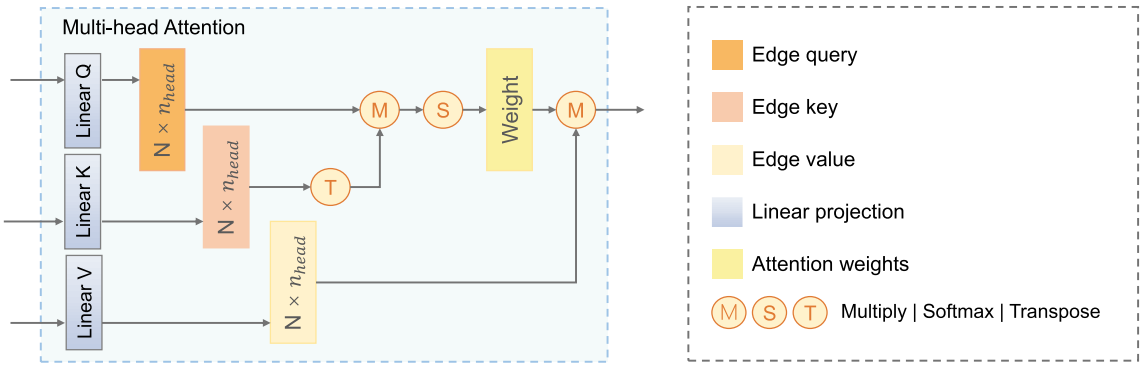


Fig. 7 Multi-head attention in edge attention block. The output edge feature of one attention head is the product of the weights and the value matrix

features and their weights.

$$Attention_i = softmax\left(\frac{Q_i \cdot K_i^T}{\sqrt{n_d/n_{head}}}\right)$$
$$F_{att_i} = Attention_i \cdot V_i \tag{6}$$

Then, we concatenate the outputs of all attention heads as F_{att}. The global feature is projected again with Φ to n_d-dimensional vector with a dropout rate set to 0.5.

$$F_{att} = \Phi\left(Concat_{i=1}^{n_{head}} F_{att_i}\right) \tag{7}$$

The attention for mesh edge means that the attention weights indicate the importance of all edges to each query edge. High weights mean high contributions to the query edge feature. Because the attention weights are calculated over all edges, we actually gather features in a global field which could be a substitution for expanding the receptive field. As we discuss in Sect. 3.2, we discard the pooling operations for efficiency reasons, which may slow down the expansion of the receptive field, and the edge-wise self-attention makes up for this problem.

Furthermore, we use two linear layers with $Relu$ activation, followed by normalization. The first one ϕ projects edge feature to a higher dimension $h_{d'}$, and the second ψ projects it back to h_d, these operations are applied to each edge separately. Also, we use residual links after these linear projections.

$$F_{out_i} = Norm(F_{in} + \psi(Relu(\phi(F_{att}))) + F_{att} \tag{8}$$

Finally, we adopt four edge attention blocks to enhance our network. Each of them learns different weights and features, adding diversities and making the network more robust. The final output of edge attention module F_{out} is the sum of these four edge attention blocks.

$$F_{out} = \sum_{i=1}^{4} F_{out_i} \tag{9}$$

The final feature F_{out} contains global features gathered in edge attention module and the local geometric features in input embedding module. It will be fed to the classification network then to predict the label of objects.

4 Experiments and analysis

In this section, we conduct experiments of mesh classification tasks on three common datasets, namely SHREC11 [54], Cube Engraving [18] and Manifold40 [40]. We illustrate the performance of MEAN and compare it with recent works. Then, we evaluate the effectiveness and efficiency of MEAN with ablation experiments.

4.1 Classification

In this subsection, we evaluated our network on three common datasets and compared the results with several mainstream methods. These three datasets cover from small scale to large scale and are representative of different cases in mesh classification tasks. We trained and tested them with the experimental platform, as shown below.

4.1.1 Experimental environment

Platform. For training and testing, all of our experiments are conducted under the same hardware and platform. We use an Intel Core i7-10,700 (3.8GHz) and an NVIDIA Geforce GTX 3090 graphics card with 24 GB memory. All the models are written and implemented with Pytorch on Ubuntu 16.04 LTS. The following experiments are all conducted on this platform.

Settings. Before training, we follow the basic data augmentation method and settings of previous methods [18], which adopt 20% edge flips and 20% vertices slides for enhancement. These augmentations add diversity to the datasets for better performance. Then, we adopt the Adam optimizer to update the gradient and train our model with the initial learning rate set to 0.0001. The batch size is set to 16, and the maximum epoch number is 300. For the dropout rate of the fully connected layers in the classification network, we set $\mathcal{P}_d = 0.3$.

4.2 Classification results

We demonstrate the effectiveness and capabilities of the proposed model for classification tasks. For all samples in the three datasets, they are guaranteed to be manifold and composed of 750 edges.

SHREC11. The SHREC11 dataset, provided by Lian [54], is composed of 600 mesh samples in 30 categories and each category contains 20 samples. We follow mainstream setup [18] to split the samples of each category in two different ways. One is that 16 samples are used as train sets and 4 are used as test sets. The other is that 10 samples are used as train sets and 10 are used as test sets. Then, we randomly rotate each training sample of SHREC11 from three different axes for data enhancement before each iteration.

The evaluation results of SHREC11 are shown in Table 1. We compared our network against other recent mainstream models of vertex-based ones like MeshWalker [33], edge-based ones like MeshCNN [18] and PDMeshNet [35]. For

Table 1 Classification results on the SHREC11

Network	SHREC11 Acc (%)	
	Split 16	Split 10
GWCNN [55]	96.6	90.3
MeshCNN [18]	98.6	91.0
MeshWalker [33]	98.6	97.1
PDMeshNet [35]	99.7	99.1
HodgeNet [31]	99.2	94.7
DiffusionNet [30]	–	**99.4**
MEAN (ours)	**100.0**	99.1

Our proposed method achieves state-of-the-art results on Split16 and is also advantageous on Split10
Bold values indicate the best one

Table 2 Classification results on Cube Engraving

Network	Input type	Cube Acc (%)
PointNet++ [28]	Point cloud	64.3
MeshCNN [18]	Mesh	92.2
PDMeshNet [35]	Mesh	94.4
MeshWalker [33]	Mesh	98.6
Laplacian2Mesh [26]	Mesh	**98.9**
MEAN(ours)	Mesh	**98.9**

Our model outperforms those recent mainstream methods for both mesh and point cloud
Bold values indicate the best one

the 16/4 split, MEAN achieves the perfect classification. All samples are classified correctly. And for the 10/10 split, the result is still competitive when compared with other methods.

Cube Engraving. The released version of Cube Engraving dataset [18] consists of around 4381 samples and 3722 of them are used as train sets and the others are test sets. In the released dataset, there are 22 categories and in which each sample is a small object inserted on a random surface of a cube. Thus there are more flat areas in these objects as disruption. The results are given in Table 2. It demonstrates that MEAN is sufficiently accurate for classification tasks of this dataset.

Manifold40. The Manifold40 is a variant of ModelNet40 [56]. ModelNet40 is a widely used benchmark in 3D deep learning and consists of 12,311 CAD models in 40 categories. However, most samples in ModelNet40 do not satisfy the Manifold constraint. Hu [40, 57] provided this Manifold40 dataset as a refined subset of ModelNet40 after remeshing and reconstruction [58] to guarantee that all the objects are closed manifolds. Manifold40 has the biggest sample scale in our experiments with roughly 9800 samples and is rather challenging. We compared our method with MeshNet [38], MeshWalker [33] and SubdivNet [40]. We do not take MeshCNN [18] into comparison since there are still holes in some samples, and errors occur in the pooling layers of the released code.

Table 3 Classification results on Manifold40 [40]

Network	Input type	Manifold40 Acc (%)
PointNet++ [28]	Point cloud	87.9
MeshNet [38]	Mesh	88.4
MeshWalker [33]	Mesh	90.5
SubdivNet [40]	Mesh	91.2
MEAN (ours)	Mesh	**91.2**

MEAN achieves higher accuracy than other models. The results of the compared methods are from Hu
Bold values indicate the best one

Table 3 shows that MEAN achieves competitive results for the classification tasks on Manifold40 with a 90.9% accuracy on the test sets. MEAN outperforms MeshNet and MeshWalker on Manifold40 and the accuracy is very close to SubdivNet.

4.3 Ablation experiments and analysis

To better analyze our model, several ablation experiments are conducted. In this subsection, we will evaluate the efficiency, effectiveness and robustness of our method.

Computation time. In the input embedding module in Sect. 3.2, to gather local structural features of mesh edge. We discard the pooling layers in typical models like MeshCNN [18] due to the high time cost of this operation. For a fair comparison, we use the default settings for both MeshCNN and MEAN. Then, we measure the computation time and accuracy when training datasets. The numbers of time costs are averaged over 1000 iterations.

As shown in Table 4, for SHREC11 and Cube Engraving, each training iteration of MEAN runs about ten times faster than original MeshCNN.

In addition, we test the efficiency and accuracy of the modified MeshCNN, which follows most of the original MeshCNN, but does not adopt pooling operation. The modified MeshCNN without pooling layers also runs faster than the original one, but the accuracy deteriorates. The results prove that MEAN achieves a significant improvement of time efficiency by discarding pooling layers, while it can get higher accuracy.

Effectiveness of each component. The proposed MEAN is composed of several modules, namely the edge convolution with residual blocks, edge attention block and spatial position encoding. To evaluate the effectiveness of each component, we evaluate several different modified versions of MEAN. We use the same train settings in Sects. 4.1 and 4.2 for each of these models for a fair comparison.

We could summarize the results in Table 5 that:

Table 4 Analysis of computation time

Network	SHREC (Split16)		Cube Engraving	
	Time (ms)	Accuracy (%)	Time (ms)	Accuracy (%)
MEAN (ours)	**17.8**	**100.0**	**18.9**	**98.9**
MeshCNN [18]	198.4	98.6	205.3	92.2
MeshCNN (w/o pooling)	20.1	97.5	20.9	89.9

The pooling operation of MeshCNN is much time-consuming. And if we discard these operations in the original MeshCNN, the performance is dropping. However, MEAN sufficiently outperforms MeshCNN and runs ten times faster than it
Bold values indicate the best one

Table 5 Analysis of the impact of each components in MEAN

Module name					Accuracy	
Edge conv	Pooling	Residual	Edge attn	Spatial encoding	Manifold40(%)	SHREC11 (%)
✓	✓	×	×	×	–	98.6
✓	×	×	×	×	77.8	97.5
✓	×	✓	×	×	79.4	98.3
✓	×	✓	✓	×	85.0	99.1
✓	×	✓	✓	✓	**91.2**	**100**

We use ✓ for that we utilize the corresponding module and × for that we do not use it. The first row represents the original MeshCNN [18], and the last row represents the full structure of the proposed MEAN. We use split-16 version of SHREC11 for experiments. Some samples in Manifold40 do not support pooling operation; thus, we do not report the accuracy in the first row
Bold values indicate the best one

- After removing the pooling layers, the performance on SHREC11 and Manifold40 both deteriorates severely, which indicates that the pooling layers are important in traditional convolutional neutral networks.
- When residual blocks are added to the model, the deterioration of test accuracy is slightly alleviated on SHREC11. However, the model could not get good performance for Manifold40.
- When edge attention module is applied, there is a significant improvement on the testing accuracy for each dataset and it achieves higher accuracy on SHREC11 than original MeshCNN [18].
- The full structure of MEAN achieves the best results. When the spatial encoding module is added to the model, MEAN outperforms the models with convolution-pooling structure such as MeshCNN.

As we discussed in Sect. 3, the traditional CNNs rely on pooling layers to expand the receptive field. However, pooling is a much time-consuming operation in MeshCNN [18]. We discard this operation and introduce the attention mechanism to help to expand the receptive field rapidly and get global features. In conclusion, the results in Table 5 support that the edge attention module is the most important module to improve the performance after removing pooling layers. At the same time, residual blocks and can also slightly alleviate the performance deterioration.

Robustness. To evaluate the robustness of our network to rotation, we rotate the input model 90 degrees on three axes in one step and evaluate our model on these new samples. In our training settings, we use various operations as data augmentation including edge flips, vertices slides and rotation. For those methods without using spatial location, like MeshCNN [18] and SubdivNet [40], these networks learn relative geometric features only, while in our method, we propose a spatial position encoding module for MEAN to utilize the spatial information. Therefore, it is necessary to test its robustness. After experiments, we do not notice a severe drop in test accuracy on all three datasets, which proves that MEAN is robust to the process of rotation.

Number of attention blocks. There are 4 attention blocks in MEAN by default and we evaluate the performances of other optional numbers. Table 6 shows the accuracy of MEAN with different numbers of attention blocks. It can be found that more than one block enables MEAN to learn richer features for better mesh awareness and make the network more robust. However, if too many blocks are adopted, there is not a big improvement. Too many blocks may lead to high computation and memory costs.

Number of training samples. Another challenging case for deep learning models is that researchers could only get a small number of samples to train the network due to the high cost for acquiring 3D meshes. Table 7 shows the accuracy of the tested models on Manifold40, in which we can find that the full training set leads to the best results for all models

Table 6 Analysis of the number of attention blocks

Num of blocks	Cube Acc (%)	Manifold40 Acc (%)
1	98.0	88.7
2	98.6	90.2
4 (default)	**98.9**	**91.2**
6	98.2	90.9
8	97.6	90.7

More attention blocks achieve higher accuracy, while the improvement is limited when there are more than 4 blocks. However, more blocks cost more memory and time. Thus, adopting 4 blocks is a good choice
Bold values indicate the best one

Table 7 Analysis of number of training samples

Samples of category	MEAN	MeshNet	MeshCNN
full	**91.2**	88.4	–
16	**78.4**	77.8	37.4
4	**61.4**	57.1	24.5

The models are tested on Manifold40. We completely follow the settings of the other two models to get fair results. When training set is sparse and performance deteriorates, our model still gets promising results
Bold values indicate the best one

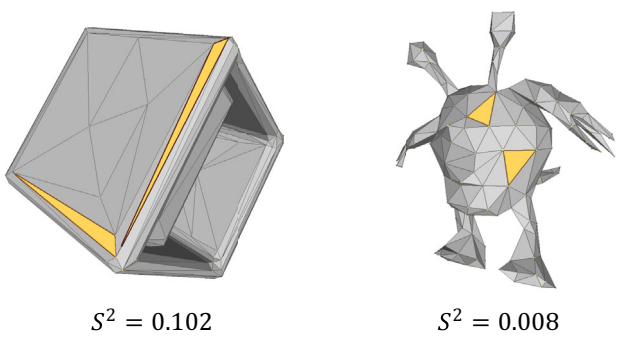


Fig. 8 Samples with narrow triangles faces (left) and regular triangle faces (right). The two samples have the maximum variance of edge length in Manifold40 ($S^2{}_{max} = 0.102$) and SHREC11 ($S^2{}_{max} = 0.008$), respectively

as we expected. And MEAN gets higher accuracy than the other models when using a much smaller size of training set. In addition, when we only use 4 samples in each category for training which means less than 5% of full training set, our model still gets pretty good result (61.4%). The results might be explained by the fact we utilize attention mechanism for global descriptor extracting; thus, when there are only a few training samples, our model could get good results for this challenging case.

Non-uniform edge length. The original dataset of Manifold40 [40, 56] is from scanning real-world CAD objects in industry; hence, the length of edge might not be uniform, which means there could be some long and some short edges in the object. These samples are more challenging for deep learning models because the irregular mesh structure might impact the network to understand the shape.

Table 8 Analysis of non-uniform edge length

Test set	MEAN (%)	MeshNet (%)
Full	**91.2**	88.4
10% (max variance)	**85.4**	84.2

To evaluate the performance on challenging cases, we use 10 percent of the test set with the maximum variance of edge length. The two models are trained on the same full training set
Bold values indicate the best one

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{N}(X_i - \overline{X})^2 \tag{10}$$

To measure the non-uniformity of edge length, we calculate the sample variance of edge length for each sample. Figure 8 shows that the samples with greater variance tend to have non-uniform edges and narrow faces, which are more challenging for networks. And we use only those with the maximum sample variance in test set to evaluate the performance of our model. Table 8 shows that compared with the full test set, the samples with higher non-uniformity of edge length are more challenging and accuracy gets lower. However, our model still achieves competitive results compared with MeshNet [38]. The explanation could be that the performance of MeshNet degrades when some triangle faces have two long sides and one short side and these triangles get narrow. However, our model could benefit from the combination of edge-convolution and edge attention module and learn to weigh edges with different attributes and features, which helps to understand the non-uniform mesh better.

The choice of edge coordinates. In our proposed method, we design the spatial position encoding module to utilize the spatial information which is often overlooked in previous works. How to represent the spatial location of an edge? Each edge of mesh is defined by two endpoints and in our implementation, we choose the midpoint coordinates as our spatial input. Table 9 shows that compared with choosing one endpoint or random point on the line segment, the midpoint version achieves better results. This could be explained that the mid-point of each edge preserves most original spatial information. However, in other versions, the density of points might not be well controlled or the distance of edges could not be represented correctly.

In summary, experimental results prove that MEAN outperforms recent methods for mesh classification tasks. Compared with convolution-based methods like MeshCNN [18], it learns comprehensive features more efficiently. Moreover, ablation experiments illustrate the effectiveness of the residual blocks, edge attention and spatial encoding module and the robustness to rotation. At last, for some challenging cases, MEAN is still capable for its tasks.

Table 9 Analysis of choices of edge coordinates

Edge coordinate	SHREC Acc (%)	Cubes Acc (%)
Midpoint (default)	**100.0**	**98.9**
Endpoint	99.2	98.7
Random point	99.2	97.1

The model gets the best results when using the midpoint coordinates as the input of spatial position encoding module
Bold values indicate the best one

4.4 Limitation

Manifold. Our proposed model is trained and tested on the manifold datasets [18, 40, 54] because this limitation is common for these edge convolutions that are designed for edges with a fixed number of neighbors. And failures will occur when a mesh object contains some non-manifold parts and these faces are not closed [57]. Its neighbor edges are illegal or undefined at this time. This failure will also occur in many other edge-based models. [18, 40]

5 Conclusion and future works

This paper presents MEAN, which learns comprehensive features on mesh edges to make better mesh shape awareness. We design the local feature aggregation block and edge attention block, leveraging the idea of self-attention mechanism to mesh deep learning. Moreover, we propose a spatial position encoding module to fully use the mesh data and it is effective for mesh classification tasks. Compared with other recent approaches, our MEAN learns mesh features from both structural and spatial information, avoiding loss of information. We conducted numerous experiments of classification tasks on popular datasets to evaluate the capabilities of our method. The results prove that the structure of MEAN outperforms mainstream methods.

Encouraged by the good performance of MEAN, we will continue our work and improve the network for more 3D mesh applications. Specifically, we will consider refining the structure and details of MEAN and extending the network to various mesh applications like semantic segmentation and retrieval. However, one limitation of our network is that the memory cost of attention blocks will rapidly rise when the network gets deeper and denser. Thus, we will try to control the memory cost in future works. In addition, for other primitives of mesh like vertices and faces, learning comprehensive features is also helpful for shape awareness. Thus, we will extend our idea to other areas [59–62] and propose more effective models.

Funding This work is supported by the National Natural Science Foundation of China (Grant No.62072348). The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Wang, K., Zhang, G., Yang, J., Bao, H.: Dynamic human body reconstruction and motion tracking with low-cost depth cameras. Vis. Comput. **37**(3), 603–618 (2021)
2. Haouchine, N., Roy, F., Courtecuisse, H., Niebner, M., Cotin, S.: Physics-based image and video editing through cad model proxies. Vis. Comput. **36**(1), 211–226 (2020)
3. Zabulis, X., Lourakis, M.I.A., Koutlemanis, P.: Correspondence-free pose estimation for 3d objects from noisy depth data. Vis. Comput. **34**(2), 193–211 (2018)
4. Hua, H., Jia, T.: Wire cut of double-sided minimal surfaces. Visu. Comput. **34**(6-8, SI), 985–995 (2018). In: 35th Computer Graphics International Conference (CGI), Indonesia, 11–14 June, 2018
5. Yeo, C., Kim, B.C., Cheon, S., Lee, J., Mun, D.: Machining feature recognition based on deep neural networks to support tight integration with 3d cad systems. Sci. Rep. **11**(1), 22147 (2021)
6. Regli, W., Rossignac, J., Shapiro, V., Srinivasan, V.: The new frontiers in computational modeling of material structures. Comput. Aided Des. **77**, 73–85 (2016)
7. Liang, Y., He, F., Zeng, X., Luo, J.: An improved loop subdivision to coordinate the smoothness and the number of faces via multi-objective optimization. Integr. Comput. Aided Eng. **29**(1), 23–41 (2022)
8. Hai, W., Jain, N., Wydra, A., Thalmann, N.M., Thalmann, D.: Time-scaled interactive object-driven multi-party vr. Vis. Comput. **34**(6-8, SI), 887–897 (2018). In: 35th Computer Graphics International Conference (CGI), Indonesia, 11–14 June, 2018
9. Diaz, J., Ropinski, T., Navazo, I., Gobbetti, E., Vazquez, P.-P.: An experimental study on the effects of shading in 3d perception of volumetric models. Vis. Comput. **33**(1), 47–61 (2017)
10. Wang, Y., Rosen, D.W.: Multiscale heterogeneous modeling with surfacelets. Comput. Aided Des. Appl. **7**(5), 759–776 (2010)
11. Buonamici, F., Furferi, R., Governi, L., Lazzeri, S., McGreevy, K.S., Servi, M., Talanti, E., Uccheddu, F., Volpe, Y.: A practical methodology for computer-aided design of custom 3d printable casts for wrist fractures. Vis. Comput. **36**(2), 375–390 (2020)
12. Kim, Y., Kwon, K., Mun, D.: Mesh-offset-based method to generate a delta volume to support the maintenance of partially damaged parts through 3d printing. J. Mech. Sci. Technol. **35**(7), 3131–3143 (2021)
13. Bukenberger, D.R., Lensch, H.P.A.: Be water my friend: mesh assimilation. Vis. Comput. **37**, 2725–2739 (2021). (**(9-11, SI)**)
14. Xiao, D., Lin, H., Xian, C., Gao, S.: Cad mesh model segmentation by clustering. Comput. Graph. **35**(3), 685–691 (2011)
15. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–660 (2017)
16. Song, Y., He, F., Duan, Y., Liang, Y., Yan, X.: A kernel correlation-based approach to adaptively acquire local features for learning 3d point clouds. Comput. Aided Des. **146**, 103196 (2022)
17. Maturana, D., Scherer, S.: Voxnet: a 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ Inter-

national Conference on Intelligent Robots and Systems (IROS), pp. 922–928 (2015)
18. Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., Cohen-Or, D.: Meshcnn: a network with an edge. ACM Trans. Graph. (TOG) **38**(4), 1–12 (2019)
19. Schneider, L., Niemann, A., Beuing, O., Preim, B., Saalfeld, S.: Medmeshcnn-enabling meshcnn for medical surface models. Comput. Methods Programs Biomed. **210**, 106372 (2021)
20. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S, et al.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
21. Guo, M.H., Cai, J.X., Liu, Z.N., Mu, T.J., Martin, R.R., Hu, S.M.: Pct: point cloud transformer. Comput Vis Media **7**(2), 187–199 (2021)
22. Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., Lu, J.: Point-bert: pre-training 3d point cloud transformers with masked point modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 19313–19322 (2022)
23. Wang, P., Gan, Y., Shui, P., Fenggen, Yu., Zhang, Y., Chen, S., Sun, Z.: 3d shape segmentation via shape fully convolutional networks. Comput. Graph. **70**, 128–139 (2018)
24. Verma, N., Boyer, E., Verbeek, J.: Feastnet: feature-steered graph convolutions for 3d shape analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
25. Qiao, Y.-L., Gao, L., Yang, J., Rosin, P.L., Lai, Y.-K., Chen, X.: Learning on 3d meshes with Laplacian encoding and pooling. IEEE Trans. Vis. Comput. Graph. **28**(2), 1317–1327 (2022)
26. Dong, Q., Wang, Z., Li, M., Gao, J., Chen, S., Shu, Z., Xin, S., Tu, C., Wang, W.: Laplacian2mesh: Laplacian-based mesh understanding. arXiv preprint arXiv:2202.00307 (2022)
27. Li, X.-J., Yang, J., Zhang, F.-L.: Laplacian mesh transformer: dual attention and topology aware network for 3d mesh classification and segmentation. In: Computer Vision - ECCV, pp. 541–560 (2022)
28. Ruizhongtai, Q.C., Yi, L., Su, H., Guibas, L.J.: Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, 4–9 Dec, 2017, Long Beach, CA, USA, pp. 5099–5108 (2017)
29. Chen, Yu., Zhao, J., Shi, C., Yuan, D.: Mesh convolution: a novel feature extraction method for 3d nonrigid object classification. IEEE Trans. Multimed. **23**, 3098–3111 (2020)
30. Sharp, N., Attaiki, S., Crane, K., Ovsjanikov, M.: Diffusionnet: discretization agnostic learning on surfaces. ACM Trans. Graph. (TOG) **41**(3), 1–16 (2022)
31. Smirnov, D., Solomon, J.: Hodgenet: learning spectral geometry on triangle meshes. ACM Trans. Graph. (TOG) **40**(4), 1–11 (2021)
32. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
33. Lahav, A., Tal, A.: Meshwalker: deep mesh understanding by random walks. ACM Trans. Graph. (TOG) **39**(6), 1–13 (2020)
34. Ben Izhak, R., Lahav, A., Tal, A.: Attwalk: attentive cross-walks for deep mesh analysis. In: 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2937–2946. IEEE (2022)
35. Milano, F., Loquercio, A., Rosinol, A., Scaramuzza, D., Carlone, L.: Primal-dual mesh convolutional neural networks. Adv. Neural Inf. Process. Syst. **33**, 952–963 (2020)
36. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
37. Lian, C., Wang, L., Wu, T.H., Liu, M., Durain, F., Ko, C.C., Shen, D.: Meshsnet: deep multi-scale mesh feature learning for end-to-end tooth labeling on 3d dental surfaces. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 837–845. Springer (2019)
38. Feng, Y., Feng, Y., You, H., Zhao, X., Gao, Y.: Meshnet: mesh neural network for 3d shape representation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8279–8286 (2019)
39. Xu, H., Dong, M., Zhong, Z.: Directionally convolutional networks for 3d shape segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2698–2707 (2017)
40. Hu, S.M., Liu, Z.N., Guo, M.H., Cai, J.X., Huang, J., Mu, T.J., Martin, R.R.: Subdivision-based mesh convolution networks. ACM Trans. Graph. (TOG) **41**(3), 1–16 (2022)
41. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
42. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser Ł, Polosukhin, I.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
43. Devlin, J., Chang, M.W., Lee, K. and Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
44. Si, T., He, F., Zhang, Z. and Duan, Y.: Hybrid contrastive learning for unsupervised person re-identification. IEEE Trans. Multimed. (2022)
45. Tang, W., He, F., Liu, Y.: Ydtr: infrared and visible image fusion via y-shape dynamic transformer. IEEE Trans. Multimed. (2022)
46. Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A.L., Zhou, Y.: Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306 (2021)
47. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: simple and efficient design for semantic segmentation with transformers. Adv. Neural Inf. Process. Syst. **34**, 12077–12090 (2021)
48. Chaudhari, S., Mithal, V., Polatkan, G., Ramanath, R.: An attentive survey of attention models. ACM Trans. Intell. Syst. Technol. (TIST) **12**(5), 1–32 (2021)
49. Niu, Z., Zhong, G., Hui, Yu.: A review on the attention mechanism of deep learning. Neurocomputing **452**, 48–62 (2021)
50. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 16259–16268 (2021)
51. Lin, K., Wang, L. and Liu, Z.: End-to-end human pose and mesh reconstruction with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1954–1963 (2021)
52. Lin, K., Wang, L., Liu, Z.: Mesh graphormer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 12939–12948 (2021)
53. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
54. Lian, Z., Godil, A., Bustos, B., Daoudi, M., Hermans, J., Kawamura, S., Kurita, Y., Lavoue G., Van Nguyen, H., Ohbuchi, R, et al. Shrec'11 track: shape retrieval on non-rigid 3d watertight meshes. In: 3DOR@ Eurographics, pp. 79–88 (2011)
55. Ezuz, D., Solomon, J., Kim, V.G., Ben-Chen, M.: Gwcnn: a metric alignment layer for deep shape analysis. In: Computer Graphics Forum, vol. 36, pp. 49–57. Wiley, Hoboken (2017)
56. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920 (2015)

57. Huang, J., Su, H., Guibas, L.: Robust watertight manifold surface generation method for shapenet models. arXiv preprint arXiv:1802.01698 (2018)
58. Lee, A.W., Sweldens, W., Schröder, P., Cowsar, L. and Dobkin, D.: Maps: multiresolution adaptive parameterization of surfaces. In: Proceedings of the 25th annual Conference on Computer Graphics and Interactive Techniques, pp. 95–104 (1998)
59. Zhang, J., He, F., Duan, Y., Yang, S.: Aidednet: anti-interference and detail enhancement dehazing network for real-world scenes. Front. Comput. Sci. **17**(2), 172703 (2023)
60. Zhang, Y., Yin, C., Qilin, W., He, Q., Zhu, H.: Location-aware deep collaborative filtering for service recommendation. IEEE Trans. Syst. Man Cybern. Syst. **51**(6), 3796–3807 (2021)
61. Mancini, S., Stecca, G.: A large neighborhood search based matheuristic for the tourist cruises itinerary planning. Comput. Ind. Eng. **122**, 140–148 (2018)
62. Li, S., Zhang, D., Xian, Y., Li, B., Zhang, T., Zhong, C. Parameter-adaptive multi-frame joint pose optimization method. Vis. Comput

Jicheng Dai received his BE degree in Computer Science and Technology from Wuhan University, China, in 2021. Currently, he is pursuing his ME degree in the School of Computer Science in Wuhan University. His research interests include computer graphics, 3D computer vision, and deep learning.

Rubin Fan is currently pursuing his PhD degree at the Institute of Artificial Intelligence, School of Computer Science, Wuhan University, China. His research interests include artificial intelligence, computer vision, 3D vision processing, and computer-aided design for the generation of 3D graphics.

Yupeng Song is currently pursuing his PhD degree with the Institute of Artificial Intelligence, School of Computer Science, Wuhan University, China. His research interests include artificial intelligence, computer vision, and 3D vision processing.

Qing Guo received his BE degree in Software Engineering from Wuhan University, China, in 2021. He is currently pursuing a ME degree in Computer Science and Technology in Wuhan University. His research interests include artificial intelligence and computer graphics.

Fazhi He received the bachelor's, master's, and PhD degrees from the Wuhan University of Technology. He was a Postdoctoral Researcher at The State Key Laboratory of CADCG, Zhejiang University, a Visiting Researcher with the Korea Advanced Institute of Science and Technology, and a Visiting Faculty Member at the University of North Carolina at Chapel Hill. He is currently a Professor with the School of Computer Science, Wuhan University. His research interests are artificial intelligence, intelligent computing, computer graphics, image processing, computer-aided design, computer-supported cooperative work, and co-design of software/hardware.