Neural Style Transfer for 3D Meshes

Hongyuan Kang
School of Informatics
Xiamen University
kanghongyuan@126.com

Xiao Dong
Department of Computer Science
BNU-HKBU United International College
xiaodong@uic.edu.cn

Juan Cao
School of Mathematical Sciences
Xiamen University
juancao@xmu.edu.cn

Zhonggui Chen*
School of Informatics
Xiamen University
chenzhonggui@xmu.edu.cn

Abstract

Style transfer is a popular research topic in the field of computer vision. In 3D stylization, a mesh model is deformed to achieve a specific geometric style. We explore a general neural style transfer framework for 3D meshes that can transfer multiple geometric styles from other meshes to the current mesh. Our stylization network is based on a pre-trained MeshNet model, from which content representation and Gram-based style representation are extracted. By constraining the similarity in content and style representation between the generated mesh and two different meshes, our network can generate a deformed mesh with a specific style while maintaining the content of the original mesh. Experiments verify the robustness of the proposed network and show the effectiveness of stylizing multiple models with one dedicated style mesh. We also conduct ablation experiment to analyze the effectiveness of our proposed work.

Keywords: ***geometric learning, style transfer, stylization, optimization***

1. Introduction

3D shape representation is a fundamental research topic in the field of computer vision and computer graphics. As a representation of 3D objects, triangular meshes have been widely used because of their strong representation ability for complex 3D models. However, due to the complex topology and diverse data structure, manipulating and modifying the mesh are challenging. Enriching the mesh model by editing existing models is inefficient.

In image processing, image style transfer based on deep learning [8] has achieved fruitful results. We can learn from existing image data and synthesize new images with different styles. The general image style transfer network works as follows: First, a pre-trained model is used to extract high-dimensional features of an image, and these features are decoupled to represent image content and style respectively. Second, given a content image, a style image, and an initialized generated image, a loss function is defined to constrain the similarity between the implicit features of the generated image and the content and style features. Last, the network updates the synthesized image through continuous optimization so that it learns a new style while preserving the original content. The success of image style transfer proves that deep learning is a powerful tool for artistic creation and data augmentation. We are therefore inspired to extend the image-to-image translation work [8] to the style transfer of complex 3D mesh models.

In recent years, researchers proposed various methods for 3D shape stylization and non-realistic modeling, such as traditional energy optimization methods [36, 23] and style transfer neural networks [37, 11]. The stylization methods extract geometric styles from 3D mesh models. Styles may refer to different geometric features in different applications, such as detailed textures, shapes, structures, and sizes. The Laplacian surface editing algorithms [2, 32] focus on transferring surface texture or coating to a smooth mesh. A neural cage network [37] is proposed to warp a source shape to match the general structure of a target shape while preserving the surface details of the source. The Legolization method [26] automatically rates a LEGO brick layout from a 3D model, considering the shapes, colors, and layouts. Cubic stylization [23] deforms a 3D model into the style of a cube and maintains the texture. In this paper, we propose a generic style transfer network that enables one mesh model to present multiple geometric styles.

Our neural style transfer network takes a style mesh and a content mesh as input. Unlike traditional hand-crafted fea-

*Corresponding author

tures, we extract latent geometric features of mesh models using a feature extractor. We adopt MeshNet [7] as the pre-trained network to encode mesh features and feed it into the proposed style learning network to deform generated mesh. MeshNet provides the spatial and structural representation of a 3D model, which is beneficial for our network in geometric learning ability. Our network optimizes the deformation of the synthesized mesh iteratively, allowing it to learn specific geometric styles while maintaining good quality. Furthermore, our network is robust and efficient in handling non-manifold meshes with an arbitrary number of triangular faces and is not affected by the processing order. We demonstrate the effectiveness of the style transfer network through rich experiments. Our specific contributions are summarized as follows:

- We propose a neural style transfer network that generalizes image-to-image translation [8] to style transfer between 3D meshes. Our network is generic in transferring multiple styles to the target mesh with outperforming results.
- We classify the implicit features extracted by MeshNet [7] into geometric style and intrinsic content, and demonstrate that this feature design can effectively guide the mesh to learn specific geometric style.

2. Related Work

Our neural style transfer network is closely related to image-to-image translation, deep learning for mesh analysis and 3D object stylization research.

2.1. Image style transfer network

Image-to-image translation networks aim to convert the style of an artistic image (such as cartoon, oil painting, and watercolor) into a target image so that they have similar artistic style. Gatys *et al*. [8] proposed a convolutional neural network that can separate and recombine natural images' content and style features. They use deep features to construct a Gram matrix to represent styles, which is adopted in many style transfer networks for images and point clouds [3]. Their method is computationally expensive since it requires repeated forward and backward passes through the pre-trained network. To improve efficiency, Johnson *et al*. [15] proposed to use perceptual loss defined on high-level features for training feed-forward transformation networks.

Image style transfer can also be achieved using Generative adversarial networks(GANs) [9]. Pix2Pix [14] uses conditional GANs for image-to-image translation tasks, where the network is conditioned on an input image and generates a corresponding output image. Unlike Pix2Pix trained on aligned image pairs, CycleGAN [40] learns image style without paired samples based on the "cycle-consistent" structural assumption. Pix2Pix and CycleGAN can only translate one image domain to another at a time. Whereas, StarGAN [4, 5] performs image translations for multiple image domains by using only a single model. StyleGAN [18] proposes a GAN formulation to generate high-resolution synthetic images based on the progressive growth mechanism. The pyramidal layer blocks in StyleGAN control disentangled features of different scales, thus style-mixing between two images can be performed naturally.

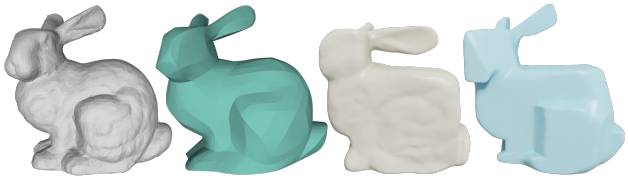
2.2. Deep learning for mesh analysis

Meshes provide an efficient and non-uniform representation for 3D geometries. Researchers have proposed different neural networks for 3D mesh analysis in recent years [21, 7, 31, 12, 28, 10, 20]. These networks encode high-dimensional features based on the vertices, edges, and faces of the mesh to improve the performance on classification or segmentation tasks. Xu *et al*. [35] proposed directional convolution on a surface mesh to solve the shape segmentation problem, defined the rotation-invariant mechanism, and utilized curvature to guide the convolution on faces. MeshNet [7] is a neural network on meshes that extracts face features and aggregates them with neighboring information. It utilizes different per-face processing and symmetry functions to solve the irregularity problem on meshes and can deal with non-manifold cases. MeshCNN[10] performs convolution within local neighboring edges and accomplishes mesh pooling via edge collapse. It learns which edge to collapse to expose and expand the important features while discarding the redundant ones.

PD-MeshNet [28] is a primary-dual framework that relies on two graphs specifically defined for meshes assigning features to both edges and faces. The dual graph allows dynamic feature aggregation to be performed on neighboring features for the graph node by using an attention mechanism. MeshWalker [20] can learn 3D shape and topology directly from a given mesh by random walking along the mesh surface. It works well even when the training dataset is small. Following the introduction of MeshNet [7], MeshNet++[31] is a deeper neural network that can learn the local structure at multiple scales and exhibits robustness to mesh decimation. SubdivNet[12] uses loop subdivision sequence connectivity as a basis to offer general convolution directly defined on meshes. Many other networks, such as the dilated convolution network [30], have also been proposed recently to improve the mesh representation ability [21].

2.3. 3D object stylization

Mesh deformation algorithms [41, 29] allow the local rigid deformation of meshes to synthesize objects with par-



(a) (b) (c) (d)

Figure 1. Visualization of different methods for stylization on an example model. (a) Original model; (b) developable surfaces [33]; (c) cubic stylization [23]; (d) ours.

ticular styles. Mesh editing algorithms based on Laplacian coordinates [2, 32] can transfer texture or coating styles to the target mesh. Some algorithms can design geometry flows or filters to create stylized geometry, such as generating edge-preserving smoothing geometry [39] and creating developable surface modeling [33]. Aiming directly at 3D object stylization, Liu *et al*. [23] presented a cubic stylization algorithm that turns object shapes into the cubic style by optimizing the As-Rigid-As-Possible (ARAP) energy with ℓ_1-regularization aligning rotated vertex normals with coordinate axes. Xu *et al*. [36] proposed a style transfer framework for co-analysis of a 3D set via style-content separation through anisotropic part scales. Berkiten *et al*. [1] proposed a displacement map transfer algorithm that converts existing high-quality, detailed 3D models to simple shapes with no textures.

Recently, researchers proposed many neural networks for 3D stylization. Liu *et al*. [25] proposed a network that allows users to edit an input 3D surface by simply selecting an image processing filter. It is a differentiable renderer with a stochastic multi-view gradient-descent procedure that can back-propagate the changes in the image domain to the mesh vertex positions. Wang *et al*. [37] proposed a neural network for detail-preserving shape deformation inspired by traditional cage-based methods [17, 22]. They designed cage-prediction and deformation-prediction models to warp a source shape to match the target shape. Inspired by MeshCNN [10], Hertz *et al*. [11] designed a deep network to learn geometric texture from local triangular patches and generate mesh vertex displacements for synthesizing local geometries. Yin *et al*. proposed 3DStyleNet [38] to predict a part-aware affine transformation field that naturally warps the source shape to imitate the geometric style of the target. Michel *et al*. [27] proposed Text2Mesh to predict color and local geometric details of 3D mesh which conform to a target text prompt. Some stylization methods emphasize texture learning or coat transfer [32, 25, 11], while our network focuses on the structure or shape style transfer. As in Figure 1, we show different stylization of bunny model generated by some representative methods and our network.

3. Method

In this section, we describe the neural style transfer of 3D meshes. We first present the architecture of the proposed geometric style transfer in Section 3.1. Then we explain how to transfer the geometric style between meshes based on the deep features extracted from the mesh representation network [7] in Section 3.2. Last we describe the strategy for selecting style and content feature layers in Section 3.3.

3.1. Network architecture

Related mesh stylization work have tried to guide style learning by some low level hand-crafted features, such as object size [36], surface normal [23, 19, 24], or curvature [6]. In our work, we try to guide the mesh deformation by high-level latent features generated by a neural network. We focus on transferring the geometric style from the style mesh to other meshes. Inspired by image style translation work [8], we design a neural style transfer network to guide the generated mesh to learn new geometric styles and preserve the original content.

We present the overall architecture of the neural style transfer for meshes in Figure 2. The network takes a content mesh and a style mesh as input and obtains their deep features from a mesh representation network, namely, MeshNet [7]. Then, it extracts the style and content representations as a reference for the synthesized mesh. Our network continuously optimizes the vertex positions of the generated mesh guided by the style similarity to the style mesh and the content consistency to the content mesh. In addition, we propose a shape regularization to ensure mesh quality during deformation.

As shown in the gray box of Figure 2, the pre-trained MeshNet is used to map mesh attributes to latent space consisting of geometric features for subsequent classification. MeshNet [7] is a 3D shape representation network based on direct convolution on mesh faces, and the intermediate features it extracts can represent the spatial and structural information of the mesh. In our work, we find that the decoupling of the latent geometric features of the mesh provides the possibility to learn and transfer geometrical styles between meshes.

3.2. Neural style transfer between meshes

Given a content mesh and a style mesh, we sample the mesh to obtain the same number of triangular faces. The initial features of a triangular face include center coordinate, corner vector, face normal, and neighboring triangle index [7]. We denote the content mesh as $C \in \mathbb{R}^{n_t \times k}$ and the style mesh as $S \in \mathbb{R}^{n_t \times k}$, where k and n_t are the numbers of input feature channels and triangle faces of the mesh, respectively. The initial feature channel k of a triangular face is 18 in the mesh model. Our network attempts to

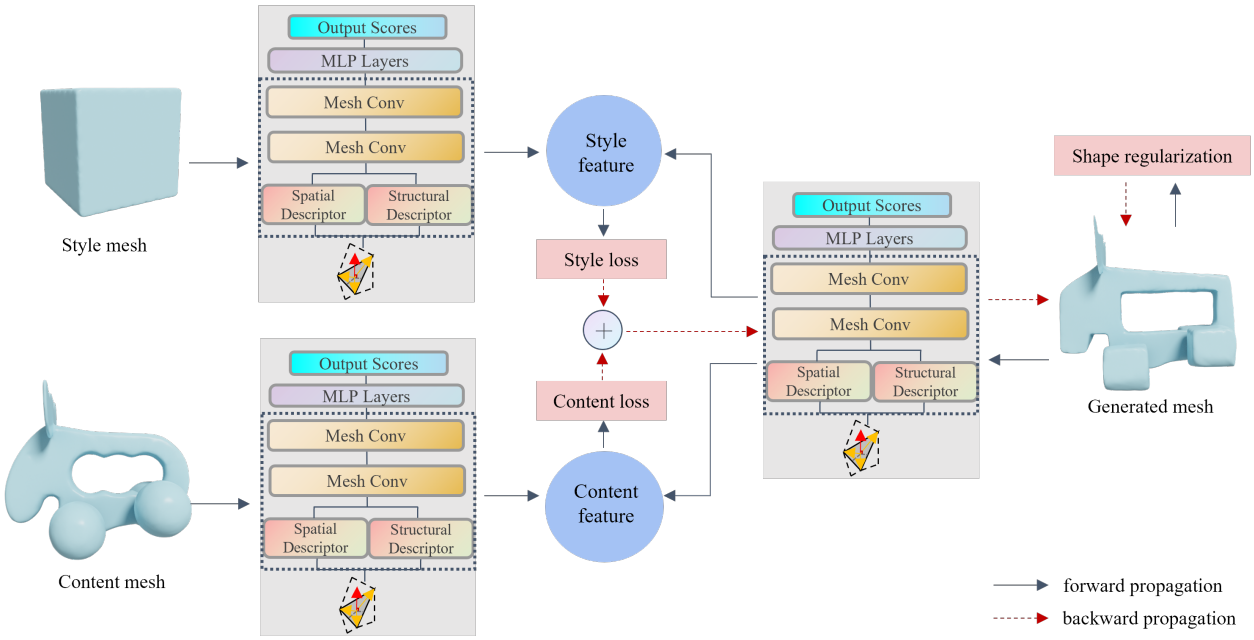


Figure 2. The overall architecture of the neural style transfer network for 3D meshes. The network in gray box is the pre-trained MeshNet [7] for classification task. Inside the dashed box are the feature layers where we extract content and style of mesh from the MeshNet. Our network utilizes MeshNet as feature extractor of input meshes to guide the optimization of the generated mesh. The blue solid arrow represents forward propagation, and the red dashed arrow represents back propagation.

transfer the geometric style of S to C by generating a new mesh M. Using a pre-trained MeshNet as an encoder to extract decoupled high-dimensional features, the style transfer network tries to synthesize a mesh that can exhibit the structural style of S while maintaining content consistency with C. We define the loss function in Equation (1). The neural style transfer continuously optimizes the generated mesh M by minimizing the following loss function:

$$\begin{aligned} Loss(C,S,M) =& \mathcal{L}_{\mathrm{c}}(C,M) + \lambda_s * L_{\mathrm{s}}(S,M) \\ &+ \lambda_r * \mathcal{L}_{\mathrm{r}}(M), \end{aligned} \tag{1}$$

where $\mathcal{L}_{\mathrm{c}}(C,M)$ is the content loss, which is used to measure the difference in content representation between C and M, $\mathcal{L}_{\mathrm{s}}(S,M)$ is the style loss to measure the difference in style representation between S and M, and $\mathcal{L}_{\mathrm{r}}(M)$ is a shape regularizer that enforces the uniform shape of triangular faces. λ_s and λ_r are the hyperparameters that balance the weights of content representation, style representation, and shape regularization. The style transfer network predicts new vertex positions of M by optimizing the loss function to obtain the final stylized mesh.

The attributes of input mesh, such as vertex position, face normal, and neighboring relationship, are encoded by the feature extractor into latent space $\mathbf{F}(\cdot) \in \mathbb{R}^{n_t \times d}$ consisting of content and style representation. d is the total number of feature channels in the latent space. Specifically, we denote $\{l_c\} = \{l_c^1, l_c^2, \ldots, l_c^p\}$ as the content layers, and $\{l_s\} = \{l_s^1, l_s^2, \ldots, l_s^q\}$ as the style layers for computation of the loss. In the following definition, we ignore the superscript and subscript, and use l to denote one feature layer.

Content loss measures the difference in content representation between content mesh C and generated mesh M. M is initialized to C and then updated by predicting the new position of each vertex. We expect M to retain basic content information in C during style learning; then, the generated mesh and content mesh can be regarded as different styles of the same object. Let $\mathbf{F}^l(\cdot)$ denote the latent features through a certain layer l in the feature extractor. The row of $\mathbf{F}^l(\cdot)$ is the number of triangular faces of the input mesh, and the column corresponds to the number of feature channels at layer l. We extract feature layers representing mesh content and define the content loss between C and M as follows:

$$\mathcal{L}_{\mathrm{c}}(C,M) = \sum_{l \in \{l_c\}} \left\| \mathbf{F}^l(C) - \mathbf{F}^l(M) \right\|_2^2, \tag{2}$$

where $\{l_c\}$ is the set of layers in the feature extractor for content representation.

Style loss measures the difference in style representation between style mesh S and generated mesh M. We hope that the geometric style of M is closer to S after each optimization step. We utilize the Gram matrix to compute the geo-

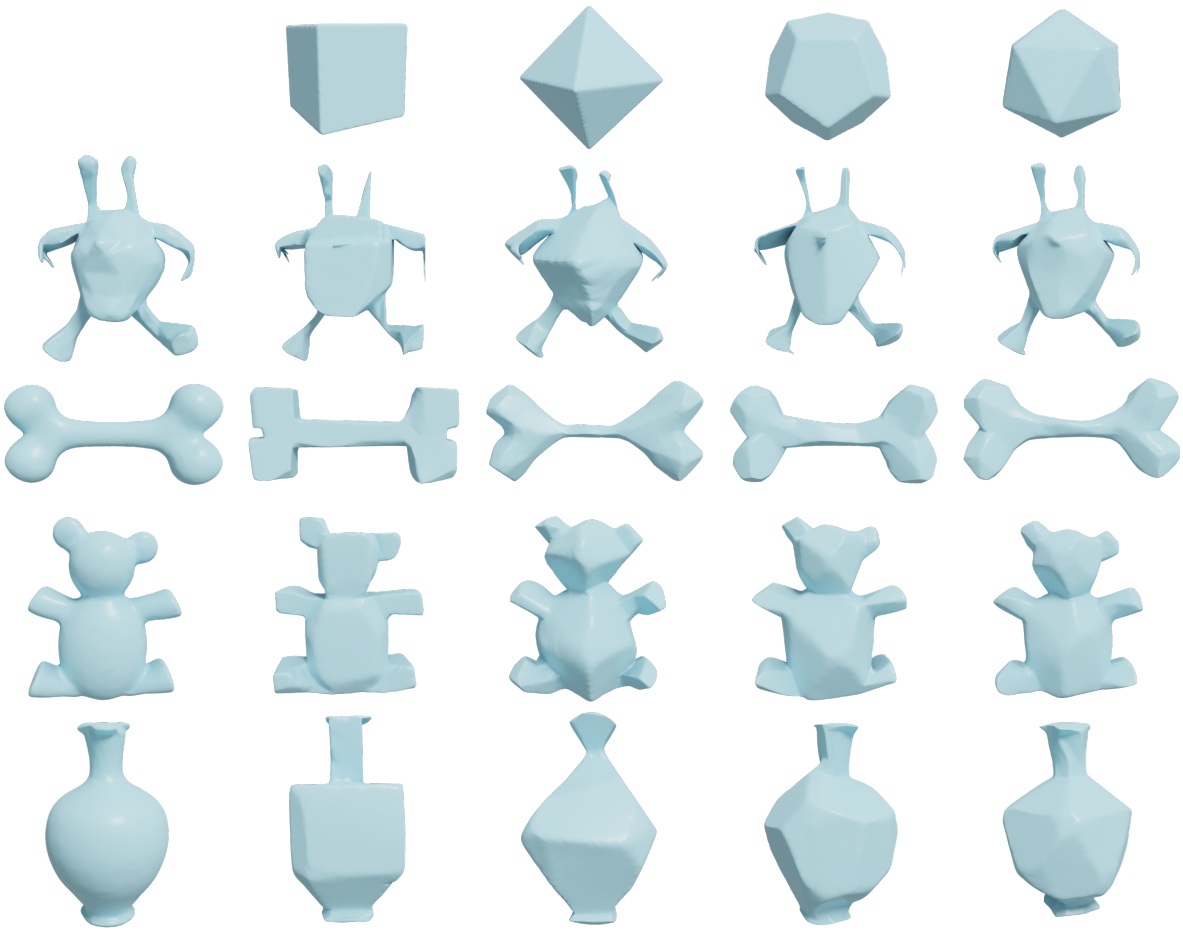


Figure 3. Style transfer results of different geometric styles. The first row shows style meshes, the first column shows content meshes, and the rest shows the results obtained by our style transfer network.

metric style of feature layer l by measuring the characteristics of each feature channel and the spatial correlation of different feature channels in latent feature $\mathbf{F}^l(\cdot)$. We denote Gram matrix of feature layer l as $G\left(\mathbf{F}^l(\cdot)\right) \in \mathbb{R}^{n\times n}$, where n is the feature channels of that layer. The geometric style is calculated by equation $G\left(\mathbf{F}^l(\cdot)\right) = (\mathbf{F}^l(\cdot))^\top(\mathbf{F}^l(\cdot))$. The ij^{th} element of the $G\left(\mathbf{F}^l(\cdot)\right)$ is the inner product of the i^{th} and j^{th} column vectors of $\mathbf{F}^l(\cdot)$.

After calculating the Gram matrix of a specific layer for S and M, we define the style loss as follows:

$$\mathcal{L}_{\mathrm{s}}\left(S, M\right) = \sum_{l\in\{l_s\}} \| \mathrm{G}\left(\mathbf{F}^l(S)\right) - \mathrm{G}\left(\mathbf{F}^l(M)\right) \|_2^2, \quad (3)$$

where $\{l_s\}$ represents the feature layers from MeshNet related to geometric style. The strategy for the selection of content feature layers $\{l_c\}$ and style feature layers $\{l_s\}$ is discussed in Section 3.3.

Shape regularization is necessary to ensure the uniformity and quality of triangular faces in M during optimization in our network. Minimizing only the content and style loss terms may result in a low-quality mesh containing irregularly-sized or locally oscillatory faces and drifting vertices, see Figure 6. Assume M contains n_e and n_v edges and vertices, respectively. Let V be the $n_v \times 3$ matrix consisting of coordinates for all the vertices of the mesh M and $E = \{e_i\}$ be the edge set of M. Also let L be the Laplace matrix of size $n_v \times n_v$ of M. We follow [16] to enforce the smoothness of the predicted mesh M by adding a shape regularization $\mathcal{L}_{\mathrm{r}}$ to the objective function:

$$\mathcal{L}_{\mathrm{r}} = \lambda_e * \mathcal{L}_{edge} + \lambda_n \mathcal{L}_{normal} + \lambda_l * \mathcal{L}_{laplacian}, \quad (4)$$

where

$$\mathcal{L}_{\text{edge}} = \frac{1}{n_e} \sum_{e_i=(v,v')\in E} \|v - v'\|^2, \quad (5)$$

$$\mathcal{L}_{normal} = \frac{1}{n_e} \sum_{e_i\in E} 1 - \cos(\hat{n}, \tilde{n}), \quad (6)$$

with $\hat{n}$ and $\tilde{n}$ the normal of two adjacent faces of e_i, and

$$\mathcal{L}_{laplacian} = \|L \cdot V\|_1. \quad (7)$$

The hyperparameters λ_e, λ_n, and λ_l are used to balance the effects of different shape regularization terms. We provide the ablation study on shape regularization to verify its effectiveness in Section 4.3.1.

3.3. Selection of feature layers

As introduced previously, MeshNet [7] adopts face-unit, feature splitting, and effective mesh block strategies for mesh representation. In our network, we use $\{l_c\}$ and $\{l_s\}$ to denote the feature layers extracted from MeshNet, respectively for calculating the content loss and style loss. In this section, we analyze the latent features encoded by MeshNet and show how to select style features from the implicit features of the style mesh so that the generated mesh can learn its geometric style.

We show the simplified architecture of the MeshNet model in Figure 2. Based on the center, corner, normal, and neighboring information of original triangular faces, MeshNet adopts two descriptors to split features of faces into the spatial feature and structural feature. The spatial descriptor takes the center coordinate of faces as input to generate high-level features relevant to spatial position. In our network, we consider the spatial features to be the content layer $\{l_c\}$ of the mesh, and we want the content of the generated mesh to be as close as possible to the content of the content mesh. The structural descriptor consists of two parts: the first part takes the corner value of each face as input to capture the "inner" structure of faces and focus on shape information; the second part takes the normal value of each face and its neighbors as input to capture the "outer" structure of faces and focus on the local environment. We believe that the structural features generated by the structural descriptor represent the geometric style of the mesh on some level. For example, latent features encoded from both a face's normal and the relations to its neighbors show a specific pattern of a cube model, which can serve as its geometric style.

Given the spatial and structural features of triangular faces, MeshNet adopts the mesh convolution block to expand the receptive field of faces by aggregating information of neighboring faces. Furthermore, MeshNet designs combination and aggregation operations on spatial and structural features to generate global features for the classification task. We experiment by selecting the latent feature of each block from MeshNet as style features in our network and perform ablation study in Section 4.3.2.

4. Experiments

In Section 4.1, we present the dataset and training settings of the proposed neural style transfer network. In Section 4.2, we demonstrate the qualitative results of our network and the comparison experiments with relevant advanced stylization methods. In Section 4.3, we perform ablation studies to validate the network design. In Section 4.4, we discuss the limitation of the proposed neural stylization network.

(a) (b) (c)

Figure 4. Comparison of the cubic stylization [23] method and our method. (a) Original models, (b) cubic stylization and (c) ours. Our results demonstrate better properties in terms of the preservation of object content and the degree of cubic stylization.

4.1. Experimental setting

Dataset. Our network adopts a pre-trained MeshNet for the feature extractor of mesh models. We use a simplified version of ModelNet40 [34] called Manifold40 [12] to train the MeshNet on classification task. Manifold40 [12] dataset contains 12,311 triangular mesh models across 40 categories, among which 9,843 are used for training, and 2,468 are used for testing. The mesh models in this dataset are watertight, and each model contains 500 faces. Manifold40 dataset is used for training MeshNet on classification task, and we use the pre-trained model to extract latent features of our mesh models. We collect some classic and popular triangular mesh models for style transfer learning, such as animals, humans, artifacts, cars, etc. Before train-

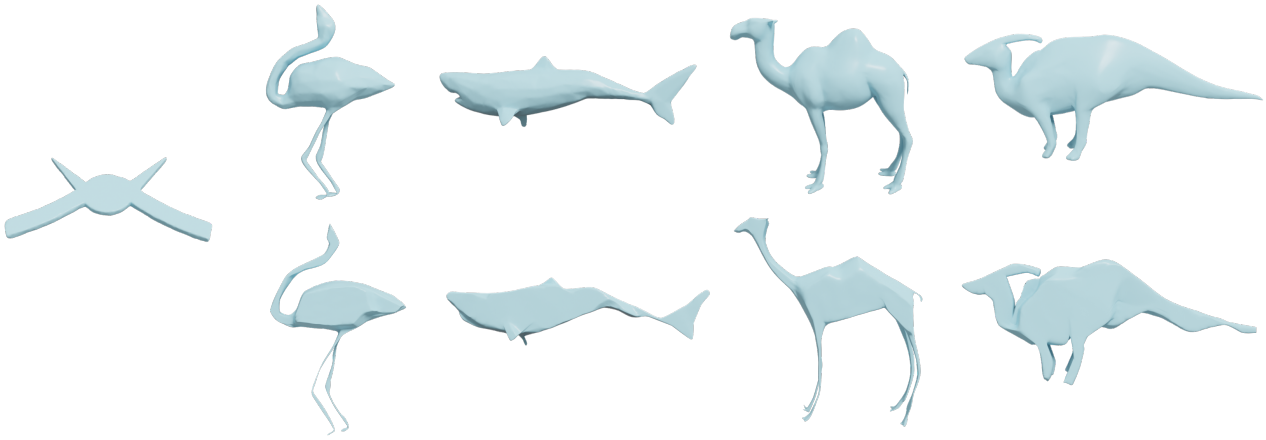


Figure 5. Neural style transfer of a scissor model. The first row shows original models and the second row shows stylized models.

ing, we pre-process the mesh model by sampling it to 2000 faces, moving it to the geometric center, and normalizing it to a unit sphere.

Training settings. We use the same hyperparameters for mesh style transfer throughout the experiments on different models. We apply the pre-trained MeshNet model with 92.75% classification accuracy as the feature extractor in our neural network. We train the style transfer network for 300 epochs taking content mesh C and style mesh S as input. The generated mesh M is initialized to content mesh C at the beginning of model training. The hyperparameters in the loss function are set as follows: $\lambda_s = 0.01$, $\lambda_r = 30,000$, $\lambda_e = 1.0$, $\lambda_n = 0.01$, and $\lambda_l = 0.1$. Before training, we normalize C and S so that the extracted mesh content and style representations have comparable proportions. To obtain the final generated style-transferred mesh, we use the Adam optimizer with an initial learning rate of 0.002. β_1 and β_2 of Adam optimizer are set to 0.9 and 0.999, respectively. We update the vertex positions of the generated mesh iteratively until the energy converges or the maximum number of training epochs is reached. The training process for 300 epochs takes up one or two minutes for all the stylized models in the paper.

4.2. Qualitative results of mesh neural style transfer

We select triangular meshes with specific geometric styles as style meshes, such as cube, octahedron, decahedron, and icosahedron models. The neural style transfer network deforms the content mesh to present the geometric style above.

Figure 3 shows the stylization results for various triangular mesh models, with original models in the first column. By specifying the cubic model as the style model, the network converts the input model into a square-shaped model while preserving the content characteristic of the original mesh as much as possible, see the second column of Figure 3. In other columns, we show the results with the octahedron, decahedron, and icosahedron styles.

Similar style transfer methods have been proposed before, such as mapping models to polycube [13] and deforming models to cube based on ARAP energy with a normal regularization [23]. In the experiment, we mainly compare our method with the cubic stylization algorithm [23]. In Figure 4, the second column shows the cubic stylization results, and the third column shows our results. Two methods take the same mesh models with 2000 triangular faces as input. We can see from the results that our method generates similar cubic style as Liu's method [23], and our method is able to preserve more local structures of the content mesh. For example, the structural information of the Spot's face, Shiba's legs, and camel's neck are well preserved, and the results are more aesthetically pleasing. In addition, Liu's method only yields cubic stylization, but our method is more generic since it can transfer more geometric styles to the source model.

Moreover, we choose a scissor model as the style mesh to learn its geometric style shown in Figure 5. The scissor has a single normal distribution presenting a flattened shape. Figure 5 shows that the normal diversity of the model surface is significantly reduced after stylization. However, unlike the cubic style, the scissor style has flattened the model, and the model body exhibits multiple parallel faces.

4.3. Ablation studies

4.3.1 Importance of shape regularization

In this section, we design the ablation study to verify the effectiveness of the shape regularization term introduced in Section 3.2.

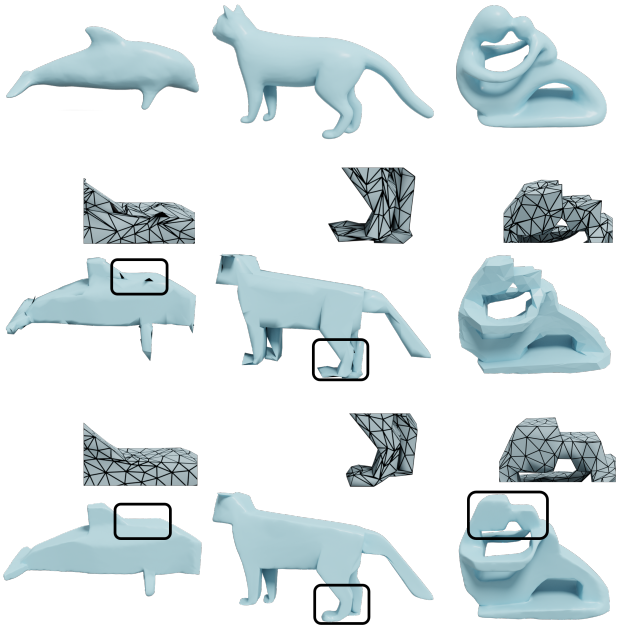


Figure 6. Cubic stylization with (last row) and without (second row) the shape regularization. The results in last row show that shape regularization enhances the smoothness of results.

In Figure 6, we show the cubic stylized meshes obtained with and without shape regularization. If the network uses only style and content losses without shape constraints, irregularly-sized faces, flipped faces, and drifted vertices may be introduced in the result meshes. As can be observed in the second row of Figure 6, the dolphin model contains flipped faces on its back, the feet of the cat model turn to visually inaesthetic, and some vertices at the head part of the fertility model drift away from the surface. In contrast, the last row shows the stylized model under shape regularization with better quality. The surface is smoother, and the shape of triangular faces is more uniform. In summary, the ablation experiments confirm the importance and effectiveness of shape regularization in our style learning network.

4.3.2 Impact of different feature layers

Our style transfer network is based on the geometric representation of the mesh from the pre-trained MeshNet model. Selecting the feature layers $\{l_c\}$ and $\{l_s\}$ for content and style is essential in our network. To verify whether the selection of feature layers is reasonable, we design ablation experiments to show how different choices of feature layers affect the stylization ability of our network. As discussed in Section 3.3, we take the spatial features as the content layers $\{l_c\}$, thus in the ablation study we mainly focus on the selection of style layers $\{l_s\}$.

Given the triangular faces of input mesh, MeshNet encodes basic features into latent features of multiple layers. First, the input features are fed to the spatial and structural descriptors to obtain spatial and structural features. Second, the 1-ring neighbors are aggregated by mesh convolution blocks to get more comprehensive feature. Then the network generates final classification score through MLP layers. Based on the architecture of MeshNet, we try to select features of different layers as the style $\{l_s\}$. We introduce the following four feature layers:

L1) Spatial features generated by spatial descriptor with triangular face information as input;

L2) Structural features generated by structural descriptor with triangular face information as input;

L3) The output features of the first mesh convolution block with spatial and structural features as input;

L4) The output features of the final mesh convolution block as input.

In Figure 3, we show the stylization of mesh models with feature L3 as the style feature. We show the stylization results with other feature layers as the style feature in Figure 7, where we can observe that by choosing different features as the style, the meshes generated by the network are quite different. We analyze the effect of different layers in the following:

- The second and sixth columns in Figure 7 show the stylization utilizing feature L1 in the style loss. The spatial descriptor mainly extracts the spatial position information of the style mesh. The generated mesh is optimized to be similar to the style mesh in spatial, resulting in dramatic shape expansion and non-realistic effect.

- Using L2, the structural features, as the style of mesh, the network generates better results than using feature L1. The structural descriptor encodes the normal and neighboring information of each triangular face, representing the model's local geometric features.

- Combining spatial and structural features, the network generates a smoother mesh with the target geometric style. Our experiments finally adopt this choice of style layer, and it can be seen from many results that the style transfer network can well deform the model to a specific style and ensure the mesh quality.

- After further encoding feature L3, we obtain higher level features L4 of the style mesh. However, the

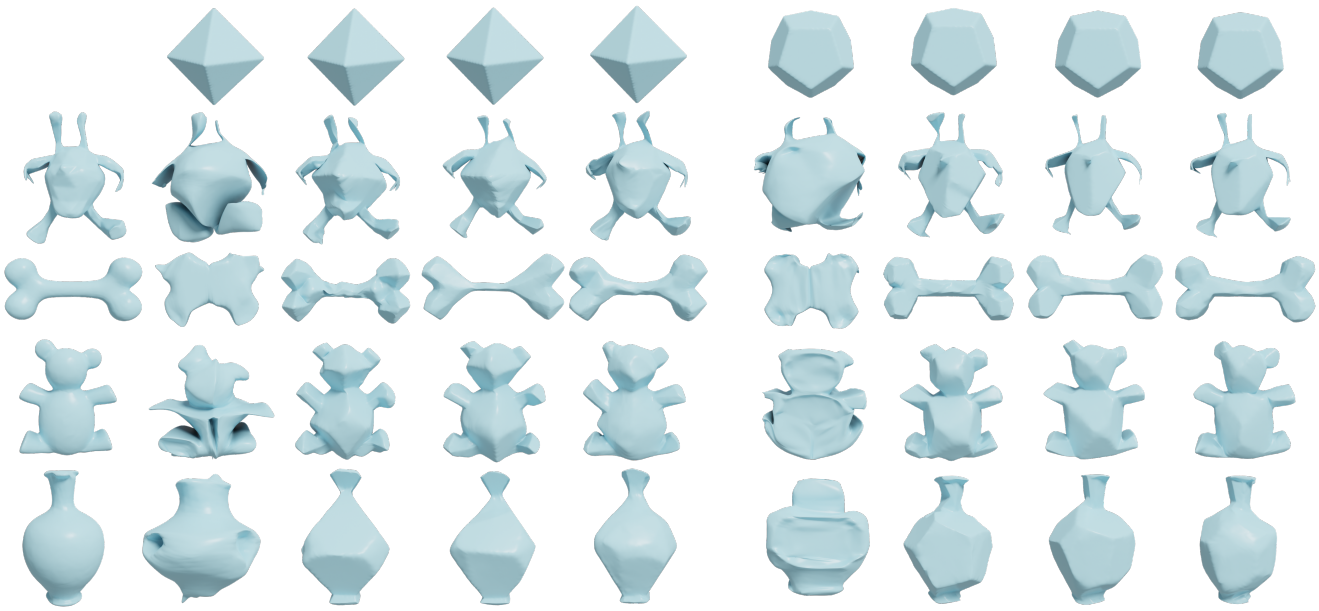


Figure 7. Effects of selecting different feature layers as style of a mesh in our network. The first row is the style mesh, the first column is the content mesh, others correspond to the results of four feature layers. It can be observed that the geometric styles extracted from the different feature layers are substantially different.

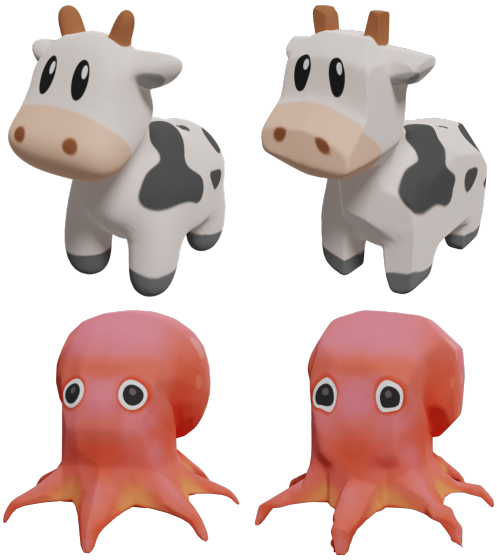
geometric difference between style mesh and content mesh decreases after layer-by-layer encoding, which indicates that the middle-level features can more clearly represent the geometric style of the mesh.

Hence, considering the performance of the feature layers mentioned above, we select feature L3 in our style transfer network to extract the geometric style of the mesh model. It extracts abundant explicit geometric features, matches our intuitive understanding of the model style visually, and the deformed mesh model is quite aesthetically pleasing.

To show the stylized models more clearly , we only show the geometric models without texture in Figure 4. To further demonstrate the advantages of our method in practice, we add the texture mapping on cubic style models in Figure 8. Our method can deform the smooth cartoon model into a cubic style and preserve the structure of the model well. For example, Spot's horns, face, and legs show line structures, and the octopus' head presents a polyhedron silhouette.

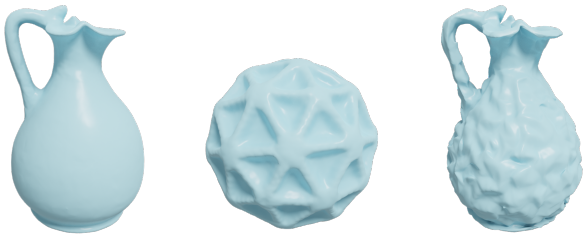
4.4. Limitation

Our network can capture the normal distribution of style models, such as models with multiple flat surfaces or raised sharp corners. However, the pre-trained MeshNet mainly learns feature representation on spatial distribution and shape structure, and has limited ability to encode surface details. Therefore, the proposed style transfer network doesn't perform well in learning surface textures. As shown in Figure 9, we choose a rail sphere with specific texture details as the style model. However, as we can observed in Figure 9 (c), the geometric pattern is not successfully transferred to the vase model.



(a) (b)

Figure 8. Cubic stylization with textures. (a) original models and (b) stylized models.



(a) (b) (c)

Figure 9. Limitation of the proposed network. (a) Content mesh; (b) style mesh and (c) style transfer result. The vase model fails to learn the geometric texture details of the rail sphere model.

5. Conclusion

We propose a neural style transfer network for 3D meshes that can transfer the style of a model to another model while preserving its content features. Our approach successfully generalizes the classic 2D image-to-image translation method [8] to 3D mesh style transfer by using a pre-trained MeshNet model to extract the content and style features of a 3D model. Compared with the existing mesh style transfer algorithms, our network architecture is simple and effective and can be generalized to learn multiple styles. In addition, our network doesn't require a manifold mesh as input. Therefore, our network can serve as a stylization tool that is beneficial to enrich the sample numbers of mesh databases. Considering the limitations of the proposed method, in the future, we will try to use other pre-trained models with a better ability to extract the local texture of the model to learn the surface styles.

References

[1] S. Berkiten, M. Halber, J. Solomon, C. Ma, H. Li, and S. Rusinkiewicz. Learning detail transfer based on geometric features. In *Computer Graphics Forum*, volume 36, pages 361–373. Wiley Online Library, 2017. 3

[2] Z. Bian and S.-M. Hu. Preserving detailed features in digital bas-relief making. *Computer Aided Geometric Design*, 28(4):245–256, 2011. 1, 3

[3] X. Cao, W. Wang, K. Nagao, and R. Nakamura. Psnet: A style transfer network for point cloud stylization on geometry and color. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3337–3345, 2020. 2

[4] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018. 2

[5] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8188–8197, 2020. 2

[6] Q. Fang, Z.-Y. Zhao, Z.-Y. Liu, L. Liu, and X.-M. Fu. Metric first reconstruction for interactive curvature-aware modeling. *Computer-Aided Design*, 126:102863, 2020. 3

[7] Y. Feng, Y. Feng, H. You, X. Zhao, and Y. Gao. Meshnet: Mesh neural network for 3D shape representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8279–8286, 2019. 2, 3, 4, 6

[8] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016. 1, 2, 3, 10

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in Neural Information Processing systems*, 27, 2014. 2

[10] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2, 3

[11] A. Hertz, R. Hanocka, R. Giryes, and D. Cohen-Or. Deep geometric texture synthesis. *ACM Transactions on Graphics (TOG)*, 39(4), aug 2020. 1, 3

[12] S.-M. Hu, Z.-N. Liu, M.-H. Guo, J.-X. Cai, J. Huang, T.-J. Mu, and R. R. Martin. Subdivision-based mesh convolution networks. *ACM Transactions on Graphics (TOG)*, 41(3):1–16, 2022. 2, 6

[13] J. Huang, T. Jiang, Z. Shi, Y. Tong, H. Bao, and M. Desbrun. L_1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics (TOG)*, 33(3):1–11, 2014. 7

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017. 2

[15] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 2

[16] J. Johnson, N. Ravi, J. Reizenstein, D. Novotny, S. Tulsiani, C. Lassner, and S. Branson. Accelerating 3d deep learning with pytorch3d. In *SIGGRAPH Asia 2020 Courses*, SA '20, New York, NY, USA, 2020. Association for Computing Machinery. 5

[17] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki. Harmonic coordinates for character articulation. *ACM Transactions on Graphics (TOG)*, 26(3):71–es, 2007. 3

[18] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 2

[19] M. Kohlbrenner, U. Finnendahl, T. Djuren, and M. Alexa. Gauss stylization: Interactive artistic mesh modeling based on preferred surface normals. In *Computer Graphics Forum*, volume 40, pages 33–43. Wiley Online Library, 2021. 3

[20] A. Lahav and A. Tal. Meshwalker: Deep mesh understanding by random walks. *ACM Transactions on Graphics (TOG)*, 39(6):1–13, 2020. 2

[21] X. Li, R. Li, L. Zhu, C.-W. Fu, and P.-A. Heng. Dnf-net: A deep normal filtering network for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics*, 27(10):4060–4072, 2020. 2

[22] Y. Lipman, D. Levin, and D. Cohen-Or. Green coordinates. *ACM Transactions on Graphics (TOG)*, 27(3):1–10, 2008. 3

[23] H.-T. D. Liu and A. Jacobson. Cubic stylization. *ACM Transactions on Graphics (TOG)*, 38(6):1–10, 2019. 1, 3, 6, 7

[24] H.-T. D. Liu and A. Jacobson. Normal-driven spherical shape analogies. In *Computer Graphics Forum*, volume 40, pages 45–55. Wiley Online Library, 2021. 3

[25] H.-T. D. Liu, M. Tao, and A. Jacobson. Paparazzi: surface editing by way of multi-view image processing. *ACM Transactions on Graphics (TOG)*, 37(6):221–1, 2018. 3

[26] S.-J. Luo, Y. Yue, C.-K. Huang, Y.-H. Chung, S. Imai, T. Nishita, and B.-Y. Chen. Legolization: Optimizing lego designs. *ACM Transactions on Graphics (TOG)*, 34(6):1–12, 2015. 1

[27] O. Michel, R. Bar-On, R. Liu, S. Benaim, and R. Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 3

[28] F. Milano, A. Loquercio, A. Rosinol, D. Scaramuzza, and L. Carlone. Primal-dual mesh convolutional neural networks. *Advances in Neural Information Processing Systems*, 33:952–963, 2020. 2

[29] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu. Anderson acceleration for geometry optimization and physics simulation. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 2

[30] V. V. Singh, S. V. Sheshappanavar, and C. Kambhamettu. Mesh classification with dilated mesh convolutions. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3138–3142. IEEE, 2021. 2

[31] V. V. Singh, S. V. Sheshappanavar, and C. Kambhamettu. Meshnet++: A network with a face. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4883–4891, 2021. 2

[32] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 175–184, 2004. 1, 3

[33] O. Stein, E. Grinspun, and K. Crane. Developability of triangle meshes. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 3

[34] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015. 6

[35] H. Xu, M. Dong, and Z. Zhong. Directionally convolutional networks for 3D shape segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2698–2707, 2017. 2

[36] K. Xu, H. Li, H. Zhang, D. Cohen-Or, Y. Xiong, and Z.-Q. Cheng. Style-content separation by anisotropic part scales. In *ACM SIGGRAPH Asia 2010 Papers*, pages 1–10, 2010. 1, 3

[37] W. Yifan, N. Aigerman, V. G. Kim, S. Chaudhuri, and O. Sorkine-Hornung. Neural cages for detail-preserving 3D deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020. 1, 3

[38] K. Yin, J. Gao, M. Shugrina, S. Khamis, and S. Fidler. 3DStylenet: Creating 3D shapes with geometric and texture style variations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12456–12465, 2021. 3

[39] J. Zhang, B. Deng, Y. Hong, Y. Peng, W. Qin, and L. Liu. Static/dynamic filtering for mesh geometry. *IEEE Transactions on Visualization and Computer Graphics*, 25(4):1774–1787, 2018. 3

[40] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 2

[41] Y. Zhu, R. Bridson, and D. M. Kaufman. Blended cured quasi-newton for distortion optimization. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 2