ISSN 1000-9000(Print)
/1860-4749(Online)
CODEN JCTEEM

JCST

Vol.34 No.6 Nov. 2019

Journal of Computer Science & Technology

SPONSORED BY INSTITUTE OF COMPUTING TECHNOLOGY
THE CHINESE ACADEMY OF SCIENCES &
CHINA COMPUTER FEDERATION
SUPPORTED BY NSFC
CO-PUBLISHED BY SCIENCE PRESS &
SPRINGER

Pan ZG, Xian CH, Jin S *et al.* Progressive furniture model decimation with texture preservation. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 34(6): 1258–1268 Nov. 2019. DOI 10.1007/s11390-019-1974-0

Progressive Furniture Model Decimation with Texture Preservation

Zhi-Guang Pan[1], Chu-Hua Xian[1,*], *Member, CCF*, Shuo Jin[2], and Gui-Qing Li[1], *Senior Member, CCF*

[1] *School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China*
[2] *BlueFire AI, Central, Hong Kong, China*

E-mail: 201821033919@mail.scut.edu.cn; chhxian@scut.edu.cn; jerry.shuojin@gmail.com; ligq@scut.edu.cn

Received April 4, 2019; revised September 26, 2019.

Abstract In digital furniture design, skillful designers usually use professional software to create new furniture designs with various textures and then take advantage of rendering tools to produce eye-catching design results. Generally, a fine-grained furniture model holds many geometric details, inducing significant time cost to model rendering and large data size for storage that are not desired in application scenarios where efficiency is greatly emphasized. To accelerate the rendering process while keeping good rendering results as many as possible, we develop a novel decimation technique which not only reduces the number of faces on furniture models, but also retains their geometric and texture features. Two metrics are utilized in our approach to measure the distortion of texture features. Considering these two metrics as guidance for decimation, high texture distortion can be avoided in simplifying the geometric models. Therefore, we are able to build multi-level representations with different detail levels based on the initial design. Our experimental results show that the developed technique can achieve excellent visual effects on the decimated furniture model.

Keywords furniture design, model decimation, texture preservation, progressive representation

1 Introduction

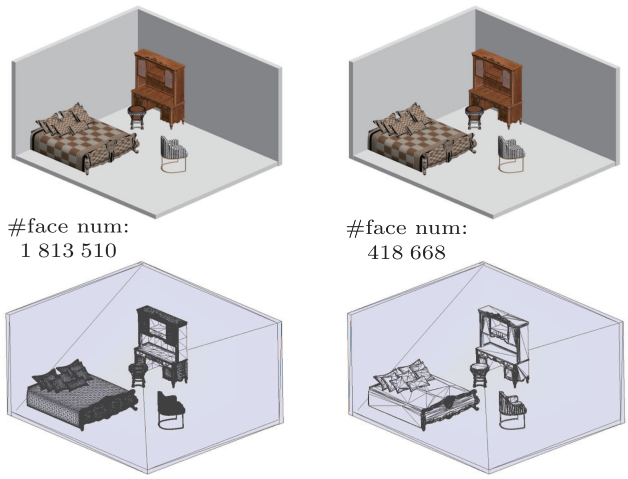
Digitalization of furniture models has increasingly become popular nowadays as it provides easiness and directness for people to interactively obtain a visual sense of the furniture that they are interested in without the trouble of checking out real objects on site. Such digital data are usually available on the Internet or provided by suppliers in digital demos. To get eye-appealing visualization of furniture models, skillful designers often use professional software (e.g., 3D Max) to create 3D geometric designs and define proper textures for different parts. Then professional rendering tools are utilized to generate rendered results for users.

Commonly, the furniture suppliers maintain a large database of furniture models, and the initial design of each furniture model possesses rich geometric details. A fine-grained furniture model can provide abundant details and a good sense of reality to users. However, the more details a furniture model holds, the more time it requires for data loading and rendering. In real applications, it is expected to provide smooth experience for customers to see rendered furniture models, requiring the whole data process from loading a model to rendering the design to be efficient enough. To decrease the time cost of loading and rendering a furniture model, a reasonable solution is to geometrically simplify it by reducing the number of faces in the original mesh. As shown in Fig.1, if we use the software 3D Max to render a room scene with 1 813 510 triangular faces, it will cost 9 seconds. In contrast, it only takes 2 seconds for rendering when the face number (#face num) is reduced to 418 668.

In recent decades, a lot of mesh decimation approaches have been proposed[1]. The decimation procedure is often controlled by some pre-defined quality criteria which target at preserving specific properties of the original models. Most of these approaches only consider minimizing the geometric deviation when simplifying the models. For example, quadric error metrics[2,3] are widely utilized in vertex-clustering based and edge-collapsing based approaches. Progres-

Regular Paper
Recommended by CAD/Graphics 2019
This work was supported by the Natural Science Foundation of Guangdong Province of China under Grant No. 2017A030313347.
*Corresponding Author

sive mesh representation[4] is also a popular solution to get different mesh models in multiple detail levels, which encodes a mesh model by a simplified mesh and a sequence which records the refinement information of the mesh. With this recorded sequence, progressive models can be dynamically reconstructed.



(a) (b)

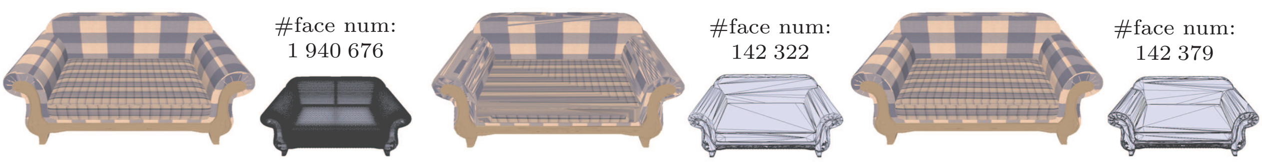
Fig.1. Render a room scene using 3D Max 2016 in the resolution $3\,072\times 2\,048$. (a) Original scene, rendered in 9 seconds. (b) Our decimated result with preserved texture, rendered in 2 seconds.

Although the geometry of a model is important, it is not the only factor that decides the rendered appearance of a model, especially for furniture models. Generally, it is defined by both the geometric details and the corresponding textures of the model. Therefore, we believe both factors should be considered in the decimation process. As shown in Fig.2, contrary to serious texture distortion with geometric criteria considered only in Fig.2(b), a better result without visible distortion is achieved with our texture-preserving simplification method in Fig.2(c).

In this paper, we mainly focus on preserving the visual appearance of furniture models, i.e., the rendered image for a decimated mesh from any view should get a small visual difference compared with that for the original mesh. To get a good visual presentation, the biggest challenge is dealing with texture maps. Since the texture details of a model are stored as texture images, the quality of texture mapping to 3D surface plays a decisive role on display effects. In this paper, a new metric is designed to evaluate the potential texture and shape distortion on edges during mesh simplification so that significant distortions can be effectively avoided. Besides, when a mesh surface is parameterized for texture mapping, it is inevitable to make cuts to reduce foldovers or high distortions, which will cause discontinuities (seams) in the parameterization. For a mesh model with multiple textures, the boundary between two texture images will also introduce such discontinuities. When edge collapse operation happens on these seams, it will very likely lead to unacceptable texture distortions. To tackle this difficulty, another metric is defined to control the edge collapse of seams.

Specifically, the main contributions in our work are as follows.

- *Minimization of Visual Texture Distortion.* To produce a high-quality screen display of a simplified model, we attempt to preserve the visual effects (including shape and texture) at any view compared with its original design. We propose a metric to measure the visual texture distortion and shape deviation on the decimated model, and by minimizing this metric the overall display effect will be retained well.
- *Minimization of Seam Stretch.* For a mesh surface, there may be some discontinuous texture coordinates on adjacent vertices that can lead to high texture distortion when edge collapse happens on them. We develop a metric of seam stretch to evaluate the possible distortion and use it to decide whether a seam edge can be collapsed.
- *Novel Decimation Algorithm.* By combining the above two factors, a simple and effective decimation algorithm is proposed and is able to generate the simplification of furniture models while keeping their overall visual effects well.



(a) (b) (c)

Fig.2. Example of texture preserved furniture decimation. (a) Original sofa design with its underlying mesh. (b) Simplified model generated by the QEM method in MeshLab software with no texture preservation, rendered in 1 second. (c) Our texture preserving decimated result, rendered in 1 second.

2 Related Work

Since the first algorithm to decimate general polygonal mesh models proposed by Schroeder *et al.* in [5], many approaches have been presented in literature to solve this problem[6−8]. A lot of proposed solutions only focus on minimizing potential geometric errors defined on models which can perform well in many application scenarios, but may fail to guarantee a good enough result in the case of a mesh model with texture appearance attributes. There are some other techniques taking appearance attributes into account that are able to conduct the appearance-preserved decimation of models. In this section, we will not aim at composing a complete review of all existing methods, but mainly cover those that are most related to our proposed approach.

Vertex clustering based approaches are usually straightforward to decimate mesh models, whose basic idea is to group vertices into clusters and compute a new vertex for each cluster[9]. Although these methods are quite computationally efficient, they generally can preserve neither topology nor small-scale geometric details. Low and Tan[10] proposed an improved approach called floating-cell clustering which takes the visual and geometric quality of simplification into consideration. To cope with perceptual degradation, couples of internal edges in each cluster are merged if a testing verification based on the curvature and size is satisfied[11]. To provide a view-dependent simplification of arbitrary polygonal scenes, Luebke and Erikson[12] proposed a hierarchical dynamic simplification framework using different vertex-folding criteria. In [3], Lindstrom extended the Rossignac-Borrel algorithm[9] and described an out-of-core implementation which requires only enough memory to store the simplified mesh. Although the vertex-clustering based approaches are efficient and robust, the main drawback of these approaches is that it may generate non-manifold results on a manifold input mesh, and the outputs of decimation may differ greatly on different choices of clustering centers.

Quadric error metric (QEM) based approaches[2,13] iteratively perform edge collapse or vertex removal guided by the quadric error metric to reach an optimal decimation. In each iteration, the position of one vertex of an edge is replaced by that of the other vertex or assigned a computed new position[14]. This research thread has made significant success in the preservation of mesh geometry, but it is still a challenge to retain the model's visual attributes. Regarding a color feature, an extended QEM based work[15] represents each vertex with a 6-dimensional vector (vertex coordinates and RGB values) for color preservation. Furthermore, such representation can be used in arbitrary dimensional space that encodes more geometric and visual information. Cohen *et al.*[16] introduced the texture deviation metric that involves the following features: surface position, surface curvature, and surface color. These attributes are sampled from the input surface and used to guide edge collapse and vertex removal in decimation. Taking a step further, Hoppe gave an improved version of this method in [17]. Because the curvature computation depends on the quality of triangulation and the graphical models often consist of dense triangles with a good quality, the method could work well on the graphical models. Different from such graphical models, furniture models often contain many planar and cylindrical regions where needle-like triangles often appear, which make the curvature computation inaccurate and unrobust. Sander *et al.*[18] introduced texture stretch metric to chart parameterization. By minimizing this metric it can partition a mesh into charts with a balanced parameterization on the surface, indicating that the texture deviation in every direction is under control. Lindstrom and Turk[19] developed image-driven simplification and used root mean square (RMS) error to guide the decimation. RMS measures the distance between images captured from the original mesh and simplifies the mesh at the same view angle. It aims at minimizing the metric at different angles to retain the overall appearance of models. Yao *et al.*[20] improved QEM by using the discrete curvature to reserve more features for mesh simplification.

Progressive meshes are adopted by Hoppe[21] to store a mesh sequence in different detail levels. One energy function consisting of distance energy, spring energy, scalar energy and another energy of discontinuous curves are minimized to generate the progressive mesh sequence. Cohen *et al.*[16] stored an error bound for each vertex in a mesh sequence. Sander *et al.*[18] resized chart polygons and optimized parameterization for all meshes in a progressive mesh sequence by considering texture deviation and stretch to obtain the optimal texture mapping.

3 Texture Preservation Model Decimation

Given a surface model together with its multiple textures, we aim at simplifying the surface mesh while minimizing the visual texture distortion on it. Each vertex is defined by its 3D spatial coordinate (x, y, z)

and texture coordinate (u, v). The 3D coordinates of vertices and their connectivity decide the 3D shape of the model. Mapping 2D texture images to the surface can be regarded as a mapping: $\mathbb{R}^2 \mapsto \mathbb{R}^3$, which is determined by the texture coordinates of vertices. Suppose there are two models M and M' sharing the same texture images, $f : \mathbb{R}^2 \mapsto \mathbb{R}^3$ is the texture mapping function for M, and $g : \mathbb{R}^2 \mapsto \mathbb{R}^3$ is the one for M'. Then the appearance difference with texture E for M and M' can be formulated as:

$$E = \iint \|f(u,v) - g(u,v)\|^2 \mathrm{d}u\mathrm{d}v, \quad (1)$$

where u, v are 2D texture coordinates.

Let M' be the simplified model from the original model M. Because the mapping $f(u, v)$ has been determined when M is created, minimizing the appearance difference with texture in (1) ideally means computing a proper mapping g^* such that

$$g^* = \arg\min_g \iint \|f(u,v) - g(u,v)\|^2 \mathrm{d}u\mathrm{d}v. \quad (2)$$

Since furniture models are represented in mesh format, it requires to discretize (2) on the surface of the model, which is not intuitive to realize, resulting in the difficulty to directly determine the optimal global mapping function g^* on M'. Hence, we look at building Level of Detail (LoD) models that can be regarded as a model sequence $S = M_1, M_2, ..., M_n$, which is derived from M, produced by controlling certain criteria and ordered by their complexity of geometry. With such LoD sequence, it is possible to make proper selections of models for various applications.

Unfortunately, a furniture model is usually designed together with multiple texture images for its different parts. Applying simplification directly to a model without considering shape and texture distortion could lead to undesired visual artifacts (see Fig.2(b)). To achieve satisfactory visual quality, both texture maintenance and geometric approximation are emphasized in our edge-collapse decimation method.

3.1 Shape Approximation

Edge-collapse based mesh decimation selectively merges the edges of the mesh to give a simplified output. When edge collapse operation is applied on an edge denoted by $(\boldsymbol{v}_1, \boldsymbol{v}_2)$, a new vertex $\boldsymbol{v}_{\text{new}}$ is generated to replace $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ (see Fig.3). As explained in [2], it is problematic to simply set $\boldsymbol{v}_{\text{new}}$ to either or their average. We notice the fact that a change on $\boldsymbol{v}_1$ will make an impact on all its one-ring faces, shedding light on our consideration that this influence should be incorporated in our formulation. We define $w(\cdot)$ as the weight function that is a monotonically increasing function with regard to the area S of triangular faces, and $d_{\boldsymbol{f}}(\boldsymbol{v})$ as the point-to-plane distance of vertex $\boldsymbol{v}$ to face $\boldsymbol{f}$. The 3D spatial coordinate of $\boldsymbol{v}_{\text{new}}$ is computed by minimizing the following formula:

$$E_{\text{shape}} = \frac{\sum\limits_{\boldsymbol{f} \in N_F(\boldsymbol{v}_1) \cup N_F(\boldsymbol{v}_2)} w(S_{\boldsymbol{f}}) d_{\boldsymbol{f}}(\boldsymbol{v}_{\text{new}})}{\sum\limits_{\boldsymbol{f} \in N_F(\boldsymbol{v}_1) \cup N_F(\boldsymbol{v}_2)} w(S_{\boldsymbol{f}})}, \quad (3)$$

where $N_F(\boldsymbol{v})$ represents the one-ring faces of vertex $\boldsymbol{v}$. It is apparent that a large triangular face makes a stronger influence on the determination of $\boldsymbol{v}_{\text{new}}$ due to the definition of $w(\cdot)$. The comparison in Fig.4 proves the positive impact of the weights. For texture coordinate of $\boldsymbol{v}_{\text{new}}$, it can be decided with the same weights in (3). In practice, it turns out that assigning the texture coordinate of the closest vertex on the original surface to $\boldsymbol{v}_{\text{new}}$ is already good enough for most applications.

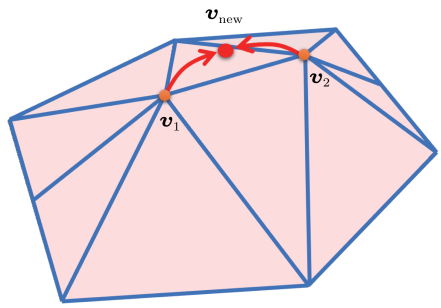


Fig.3. Edge collapse operation based on E_{shape}.

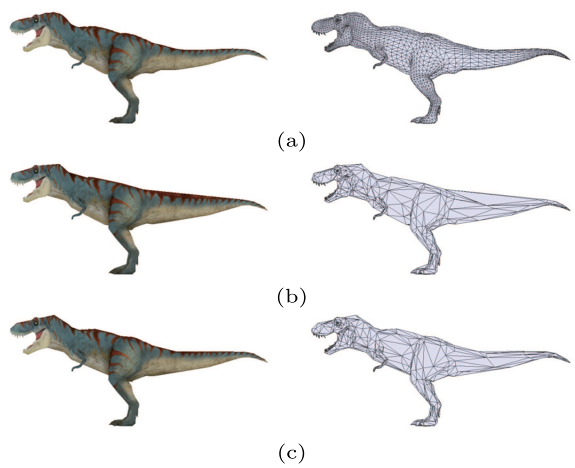


Fig.4. Decimating a dinosaur model using different weights in (3). (a) Original input model. (b) Result with $w(\cdot) = 1$. (c) Result with our weight choice $w(S) = S$. Our result demonstrates a better shape restoration.

As illustrated in Fig.3, we determine the new vertex by minimizing the energy function E_{shape} defined in (3). The smaller E_{shape} is, the closer the new vertex will be to the original model. In contrast, if E_{shape} is too large, the computed new vertex will be far away from the original model, leading to the poor preservation of local geometric features. In this case, the corresponding edge will be forbidden to be collapsed.

3.2 Seam Stretch

Seams are defined as the edges indicating the intersection of two different texture images (refer to the red edges in Fig.5). The vertices on seams contain multiple texture coordinates for different textures. As shown in Fig.5, edge collapse on a seam or an edge with one vertex on seam can cause significant texture distortion. An intuitive solution to overcome this is to fix such edges and only allow the rest of edges to collapse, which will unluckily induce two problems (see Fig.6): 1) the rate of simplification will be negatively affected, and 2) the edge collapse in other regions may cause texture distortion.

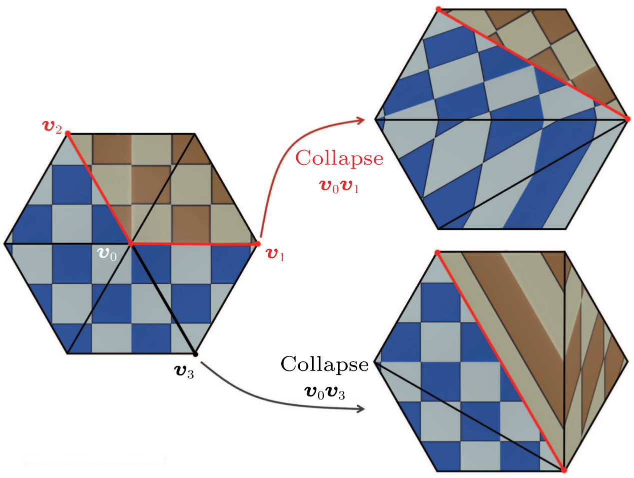


Fig.5. To collapse edges related to seams causes high texture distortion.

To properly handle seams, we first define our collapse rule: for a seam edge $(\boldsymbol{v}_1, \boldsymbol{v}_2)$, the collapse operation makes it degenerate to either $\boldsymbol{v}_1$ or $\boldsymbol{v}_2$. It is obvious that the collapse on two connected seam edges with a small angle will not lead to large distortion (even no distortion if they form a straight line). More specifically, for seam edges $(\boldsymbol{v}_0, \boldsymbol{v}_1)$ and $(\boldsymbol{v}_1, \boldsymbol{v}_2)$, a collapse operation on $(\boldsymbol{v}_0, \boldsymbol{v}_1)$ makes it become $\boldsymbol{v}_0$. It is straightforward to measure the deviation as:

$$E_{\text{seam}} = 1 + \frac{(\boldsymbol{v}_0 - \boldsymbol{v}_1)^{\mathrm{T}}(\boldsymbol{v}_2 - \boldsymbol{v}_1)}{\|\boldsymbol{v}_0 - \boldsymbol{v}_1\| \cdot \|\boldsymbol{v}_2 - \boldsymbol{v}_1\|}.$$

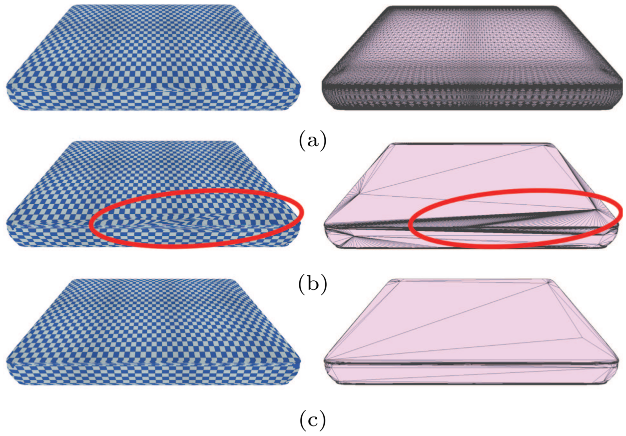


Fig.6. Comparison of fixing seam related edges and collapsing seams properly. (a) Input design and its mesh. (b) Result by fixing seam related edges. (c) Result from our seam collapse strategy.

For a general edge with none of its vertices on seams, collapse will not lead to texture distortion around seams. Any edge with only one vertex on seam-s is forbidden to collapse. Therefore, given an arbitrary edge $\boldsymbol{v}_0\boldsymbol{v}_1$, we finalize E_{seam} as:

$$E_{\text{seam}} = \begin{cases} 0, & \text{general edges}, \\ 1 + \frac{(\boldsymbol{v}_0 - \boldsymbol{v}_1)^{\mathrm{T}}(\boldsymbol{v}_2 - \boldsymbol{v}_1)}{\|\boldsymbol{v}_0 - \boldsymbol{v}_1\| \cdot \|\boldsymbol{v}_2 - \boldsymbol{v}_0\|}, & \text{seam edges}, \\ \infty, & \text{others}. \end{cases} \quad (4)$$

3.3 Model Decimation

Our decimation approach takes both shape approximation and seam stretch into account, leading to our edge collapse metric as a combination of (3) and (4):

$$E_{\text{edge}} = \lambda E_{\text{shape}} + (1 - \lambda) E_{\text{seam}},$$

where the parameter $0 \leqslant \lambda \leqslant 1$ weighs the contribution of E_{shape} and E_{seam} ($\lambda = 0.05$ in our setting). We summarize our algorithm as follows.

- For each edge, $\boldsymbol{v}_{\text{new}}$ is computed by minimizing E_{shape}. E_{seam} is also determined.
- E_{edge} is exploited to decide the priority of collapse. The edge with a lower value will be assigned a higher priority, while the edge will be forbidden to collapse if its corresponding energy E_{edge} is infinite.
- Edges are collapsed based on their priority, and their spatial and texture coordinates are updated accordingly. For seam edges, they are updated following the rule in Subsection 3.2; for general edges, we obey the strategy detailed in Subsection 3.1.

Designers often design mesh models with symmetry. As shown in Fig.7, the texture coordinates of vertices $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$ are identical. When $(\boldsymbol{v}_0, \boldsymbol{v}_1)$ is collapsed from $\boldsymbol{v}_0$

to $\boldsymbol{v}_1$, it produces a triangular patch that has the same texture coordinates on its different vertices, resulting in a huge texture stretch. Let Ω be the set of edges where there exist vertices with the same texture coordinates to their two vertices (two end vertices of the edge) in their one-ring. We define δ as a symmetry factor for edge $(\boldsymbol{v}_0, \boldsymbol{v}_1)$:

$$\delta(\boldsymbol{v}_0, \boldsymbol{v}_1) = \begin{cases} \infty, & \text{if } (\boldsymbol{v}_0, \boldsymbol{v}_1) \in \Omega, \\ 0, & \text{if } (\boldsymbol{v}_0, \boldsymbol{v}_1) \notin \Omega. \end{cases}$$

By incorporating this factor, we rewrite the edge metric as:

$$E_{\text{edge}} = \lambda E_{\text{shape}} + (1 - \lambda) E_{\text{seam}} + \delta.$$

Fig.8 shows the new formula successfully handles the texture stretch on the bed model.

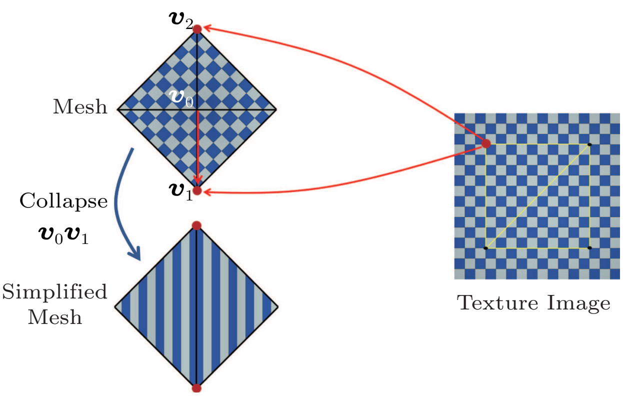


Fig.7. Collapsing an edge with symmetric texture coordinates causes huge texture stretch.

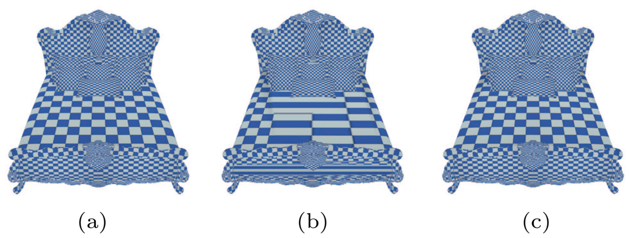


Fig.8. The edge collapse metric without considering symmetry generates unwanted artifacts on the bed model. By incorporating the symmetry factor, it gives satisfactory decimated result. (a) Original model. (b) Decimation result without symmetry factor. (c) Decimation result with symmetry factor.

4 Results

4.1 Implementation

We have implemented our decimation technique in C++ on a PC with Intel® Core i7-8700 3.5 GHz, RAM 16 GB, and NVIDIA Geforce GTX 1060 6 GB. All results in this paper are rendered with 3D Max 2016.

As introduced before, a specified criterion must be given to determine whether an edge can be degenerated. Suppose E^*_{shape} is the maxima among all $\{E^i_{\text{shape}}\}$ on the model edges, only the edges in the set $\{\boldsymbol{e}_j \mid E_{\text{shape}}(\boldsymbol{e}_j) \leqslant \alpha E^*_{\text{shape}} \text{ and } E_{\text{seam}}(\boldsymbol{e}_j) \leqslant \varepsilon\}$ may collapse in the decimation process. We set $\alpha = 0.9$ and $\varepsilon = 0.1$ for all furniture models in our implementation. However, different thresholds may induce different results. The smaller α or ε is, the more details the decimated model preserves. Fig.9 shows an example using different thresholds to guide the decimation on a wolf model.

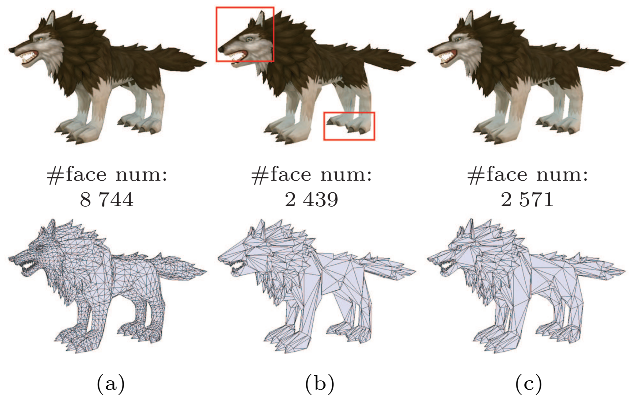


Fig.9. Decimating the wolf model using different thresholds. (a) Original model. (b) Decimation result with $\alpha = 0.9$ and $\varepsilon = 0.1$. (c) Decimation result with $\alpha = 0.75$ and $\varepsilon = 0.1$.

Users are also allowed to specify a target face number for simplification. Suppose n_f is the face number of the original model, n'_f is that of the simplified model and n_e stands for the number of the edges which can be collapsed. If $n_f - n'_f \leqslant n_e$, we will collapse $n_f - n'_f$ edges referring to their priority. Otherwise, each edge in the set will collapse.

4.2 Experimental Results

Our developed decimation technique has generated high-quality results with various furniture models. Fig.10 shows a simplified bed model with a flower pot on it. Clearly, the flower pot as a small feature is well handled by our method, resulting in a very good rendering output. Many furniture designs contain complicated geometry for aesthetic purpose, which brings difficulty for algorithms to deal with. Our method can successfully process complex designs as shown in Fig.11 and Fig.12. Besides, there are some furniture models that can be decimated a lot because of their simplicity. A high rate of simplification will provide significant efficiency improvement. The examples in Fig.13 and

Fig.14 demonstrate the effectiveness of our decimation algorithm to reach this goal, where the face number is reduced to 10% in both cases. Besides, our method can be applied to decimate graphical models with proper settings of the parameters as shown in Fig.4 and Fig.9.

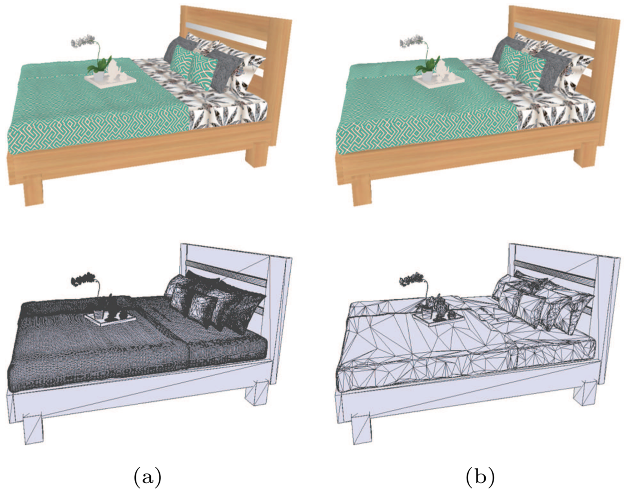


Fig.10. Bed model with a flower pot. (a) Original model. (b) Simplified model generated by our approach.

Fig.11. Table model with complex geomtric details. (a) Original model. (b) Simplified model generated by our approach.

Progressive Decimation. We can apply our proposed approach to build a progressive model sequence of decimation. Fig.15 shows an example of a progressive mesh sequence for a bed model iteratively generated by our technique. The provided statistics proves that less simplified results achieve better appearance preservation while more decimated ones can be rendered with much less time. This enables users to pick proper model based specific application needs, i.e., for those where efficiency is more important, the models with less faces can be loaded, and vice versa.

Fig.12. Big cabinet model with rich geomtric details. (a) Original model. (b) Simplified model generated by our approach.

Fig.13. The face number is reduced to 10% on this chair model. (a) Original model. (b) Simplified model generated by our approach.

Fig.14. Another greatly simplified chair model. (a) Original model. (b) Simplified model generated by our approach.

4.3 Verification

Evaluation. The RMS error is adopted to evaluate the appearance quality of the decimated models. As explained in [19], the difference between two images of the original and the simplified model is computed in order to measure how well the appearance is retained. Suppose there are two images Y^0 and Y^1 with an identical resolution $m \times n$, then the RMS error is defined as:

$$d_{\mathrm{RMS}}(Y^0, Y^1) = \sqrt{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}(y^0_{ij} - y^1_{ij})^2},$$

where y^0_{ij} and y^1_{ij} are luminance pixel values of corresponding images of Y^0 and Y^1 respectively. A small RMS error indicates a high-quality visual restoration of a rendered decimated result. Table 1 demonstrates the RMS errors for part of our results in this paper, showing that our approach can achieve excellent visual results on the decimated furniture models compared with their original inputs.

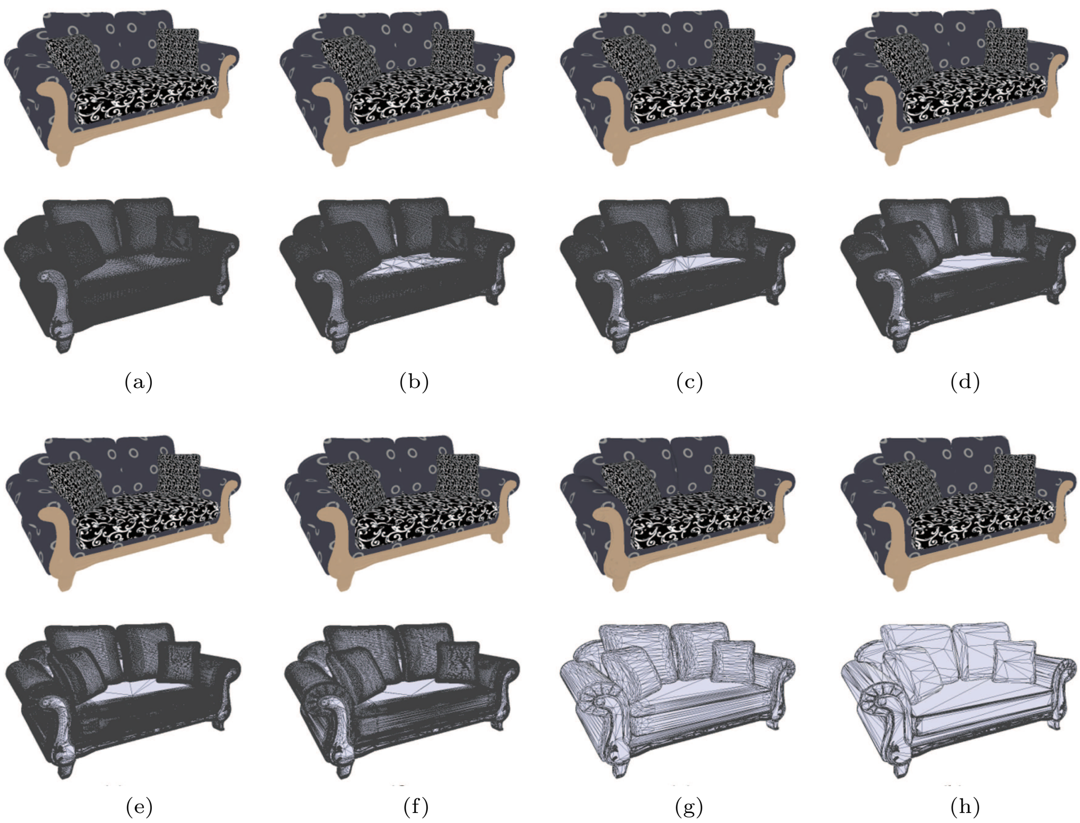


Fig.15. A progressive model sequence is built from a sofa model. (a) Original input with 2 137 284 faces. (b) 1 882 564 faces with RMS error 0.246 7 and rendering time 10 s. (c) 1 602 989 faces with RMS error 0.239 8 and rendering time 8 s. (d) 1 246 795 faces with RMS error 1.073 7 and rendering time 6 s. (e) 1 002 041 faces with RMS error 1.625 2 and rendering time 5 s. (f) 670 586 faces with RMS error 3.797 4 and rendering time 5 s. (g) 300 546 faces with RMS error 7.421 9 and rendering time 2 s. (h) 152 471 faces with RMS error 12.382 7 and rendering time 2 s.

Table 1. Numerical Summary of Our Results

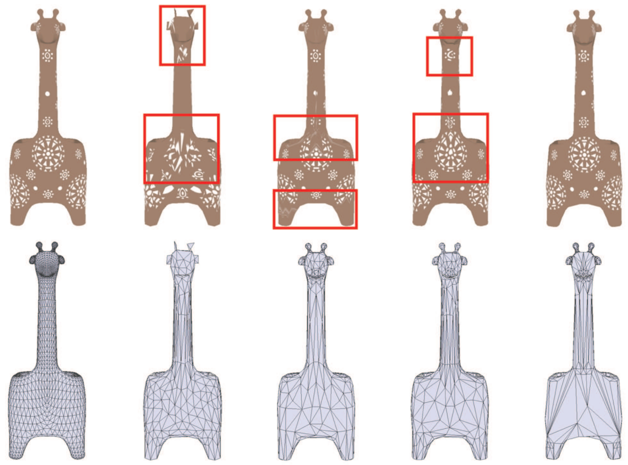
	n_f	n'_f	RMS	Resolution	t
Fig.10	341 804	72 253	9.975 0	800 × 600	3.392
Fig.11	516 070	189 100	7.505 4	800 × 600	4.895
Fig.12	578 048	114 859	5.246 5	800 × 800	6.705
Fig.13	43 096	4 014	8.460 4	800 × 600	0.552
Fig.14	332 134	36 154	8.796 0	700 × 700	3.797

Note: n_f and n'_f represent the face number of the original model and the decimated model respectively. t is the running time of decimation in seconds.

Comparison. We compare our approach with the QEM method based on discrete curvature[20], the QEM method with texture[15], and the QEM methods with texture implemented in MeshLab software. Fig.16 and Fig.17 show the comparative results of a giraffe model and a bed model respectively. Table 2 shows the corresponding numerical data, where n_f is the face number of the original model. n_1, n_2, n_3, and n_4 are the face numbers of the decimated models using the method in [20], the method in [15], QEM with texture, and our method, respectively. R_1, R_2, R_3, and R_4 are the RMS errors of the method in [20], the method in [15], QEM with texture, and our method, respectively. From these two examples, it can be seen that the method in [20], [15], and QEM with texture will generate appearance distortions, and they cannot properly handle symmetric textures. In contrast, our texture-preserved decimation approach can keep the texture appearance quite well on the simplified models with much less faces.

4.4 Limitation

We suffer a limitation that is shared by many other simplification techniques. As explained in Subsection 4.1, a guarantee on the satisfaction of user-specified target face number is missing in our method. When the specified target exceeds the maximum number of collapsible edges determined by our approach, it will not be able to give an expected result. In some cases, an iterative strategy could be used here to partially tackle this issue, i.e., building a progressive mesh sequence.

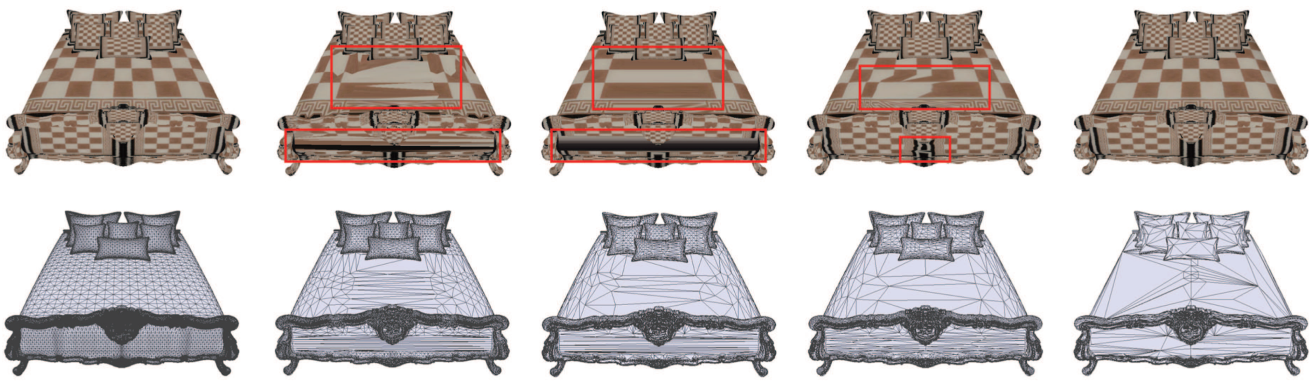


(a) (b) (c) (d) (e)

Fig.16. Decimated results of a giraffe model by different methods. (a) Original model. (b) Method in [20]. (c) Method in [15]. (d) QEM with texture. (e) Our approach.

5 Conclusions

When we tried to simplify a furniture design, not only its mesh but its texture should be considered. In this paper, we developed a novel technique that is able to produce the texture-preserved decimation of furniture models, which can be exploited to build progressive mesh sequences for various applications. Two metrics are formulated to evaluate the potential texture



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

Fig.17. Decimated results of a bed model by different methods. (a) Original model. (b) Method in [20]. (c) Method in [15]. (d) QEM with texture. (e) Our approach.

Table 2. Comparison with QEM Methods

	Resolution	n_f	n_1	n_2	n_3	n_4	R_1	R_2	R_3	R_4
Giraffe model (Fig.16)	400×800	8 720	1 414	1 414	1 414	1 414	19.068 7	13.895 8	13.606 3	12.284 9
Bed model (Fig.17)	800×600	387 224	81 000	77 444	81 000	70 968	24.628 5	19.394 0	13.519 7	11.266 2

distortion on edges, which are combined to determine the priority for edge collapse. Our experimental results showed that high-quality model decimation is achieved on different types of furniture models.

Future Work. There are several potential improvements that we will try out in the future. First of all, the current formulation in (3) to compute the vertex position acquires a straightforward strategy, which is to minimize the sum of weighted point-to-plane distances. There exists the possibility of incorporating other geometric properties (e.g., length, curvature) to achieve a better result. Second, as illustrated in Subsection 3.2, any edge with only one vertex on a seam is not allowed to collapse in our present setting. We would like to investigate a solution to collapse these edges which can help further simplify the models. Third, our method mainly focuses on decimating single-mesh models. In many applications, furniture models consist of multiple physically connected mesh patches rather than a single one. We would like to investigate the extension of our current technique for such a case.

References

[1] Botsch M, Kobbelt L, Pauly M, Alliez P, Lévy B. Polygon Mesh Processing (1st edition). A K Peters/CRC Press, 2010.

[2] Garland M, Heckbert P S. Surface simplification using quadric error metrics. In *Proc. the 24th Annual Conference on Computer Graphics and Interactive Techniques*, August 1997, pp.209-216.

[3] Lindstrom P. Out-of-core simplification of large polygonal models. In *Proc. the 27th Annual Conference on Computer Graphics and Interactive Techniques*, July 2000, pp.259-262.

[4] Hoppe H. Efficient implementation of progressive meshes. *Computers & Graphics*, 1998, 22(1): 27-36.

[5] Schroeder W J, Zarge J A, Lorensen W E. Decimation of triangle meshes. *ACM SIGGRAPH Computer Graphics*, 1992, 26(2): 65-70.

[6] Luebke D. A survey of polygonal simplification algorithms. Technical Report, University of North Carolina at Chapel Hill, 1997. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.104.3802&rep=rep1&type=pdf, Sept. 2019.

[7] Lubeke D P. A developer's survey of polygonal simplification. *IEEE Computer Graphics and Applications*, 2001, 21(3): 24-35.

[8] Talton J O. A short survey of mesh simplification algorithms. Technical Report, University of Illinois at Urbana-Champaign, 2004. https://pdfs.semanticscholar.org/ddc3/59b398e5c7d95ceb4015e65ad4e103d669dd.pdf, Sep. 2019.

[9] Rossignac J, Borrel P. Multi-resolution 3D approximations for rendering complex scenes. In *Modeling in Computer Graphics: Methods and Applications*, Falcidieno B, Kunii T L (eds.), Springer, 1993, pp.455-465.

[10] Low K L, Tan T S. Model simplification using vertex-clustering. In *Proc. the 1997 Symposium on Interactive 3D Graphics*, April 1997, pp.75-82.

[11] Reddy M. SCROOGE: Perceptually-driven polygon reduction. *Computer Graphics Forum*, 1996, 15(4): 191-203.

[12] Luebke D, Erikson C. View-dependent simplification of arbitrary polygonal environments. Technical Report, University of North Carolina at Chapel Hill, 2006. https://apps.dtic.mil/dtic/tr/fulltext/u2/a446958.pdf, Sep. 2019.

[13] Li Y, Zhu Q. A new mesh simplification algorithm based on quadric error metrics. In *Proc. the 2008 International Conference on Advanced Computer Theory and Engineering*, December 2008, pp.528-532.

[14] Schroeder W J. A topology modifying progressive decimation algorithm. In *Proc. the 1997 IEEE Visualization*, October 1997, pp. 205-212.

[15] Garland M, Heckbert P S. Simplifying surfaces with color and texture using quadric error metrics. In *Proc. the 1998 IEEE Visualization*, October 1998, pp.263-269.

[16] Cohen J, Olano M, Manocha D. Appearance-preserving simplification. In *Proc. the 25th Annual Conference on Computer Graphics and Interactive Techniques*, July 1998, pp.115-122.

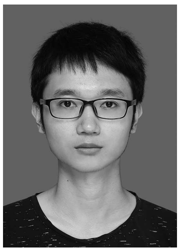
[17] Hoppe H. New quadric metric for simplifying meshes with appearance attributes. In *Proc. the 1999 IEEE Visualization*, October 1999, pp.59-510.

[18] Sander P V, Snyder J, Gortler S J, Hoppe H. Texture mapping progressive meshes. In *Proc. the 28th Annual Conference on Computer Graphics and Interactive Techniques*, August 2001, pp.409-416.

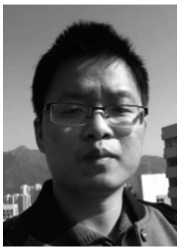
[19] Lindstrom P, Turk G. Image-driven simplification. *ACM Transactions on Graphics*, 2000, 19(3): 204-241.

[20] Yao L, Huang S, Xu H, Li P. Quadratic error metric mesh simplification algorithm based on discrete curvature. *Mathematical Problems in Engineering*, 2015, 2015: Article No. 428917.

[21] Hoppe H. Progressive meshes. In *Proc. the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, August 1996, pp.99-108.



Zhi-Guang Pan received his B.E. degree in intelligent science and technology from South China University of Technology (SCUT), Guangzhou, in 2018. He is currently a Master student in the School of Computer Science and Engineering at SCUT, Guangzhou. His current research interests are geometry modeling and processing, 3D computer vision, and image processing.



Chu-Hua Xian is currently an associate professor in the School of Computer Science and Engineering at South China University of Technology, Guangzhou. He received his Ph.D. (2012) degree in computer science and technology from the State Key Laboratory of CAD&CG at Zhejiang University, Hangzhou. He was a postdoctoral researcher in The Chinese University of Hong Kong from Nov. 2013 to May 2014 and from Sept. 2015 to Apr. 2016. His current research interests are image processing, intelligent computer graphics, geometry modeling and processing, and 3D computer vision.

Shuo Jin is an engineer at Blue-Fire AI, Hong Kong. He received his Ph.D. (2016) degree in mechanical and automation engineering from the Chinese University of Hong Kong, Hong Kong, and his B.S. (2010) degree in information and computing science from the Department of Mathematics at Zhejiang University, Hangzhou. His research interests include geometric modeling and processing, and natural language processing.

Gui-Qing Li received his B.S. (1987) degree in mathematics from the University of Science and Technology of China, Hefei, M.S. (1990) degree in mathematics from Nankai University, Tianjin, and Ph.D. (2001) degree in computer application from Chinese Academy of Sciences, Beijing. He is currently a professor in the School of Computer Science and Engineering at South China University of Technology, Guangzhou. His main research interests include digital geometry processing and reverse engineering.

JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

Volume 34, Number 6, November 2019

JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

《计算机科学技术学报》

Volume 34 Number 6 2019 (Bimonthly, Started in 1986)

Indexed in: **SCIE, Ei, INSPEC, JST, AJ, MR, CA, DBLP**

Edited by:

THE EDITORIAL BOARD OF JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

Guo-Jie Li, Editor-in-Chief, P.O. Box 2704, Beijing 100190, P.R. China

Managing Editor: Feng-Di Shu E-mail: jcst@ict.ac.cn http://jcst.ict.ac.cn Tel.: 86-10-62610746

Sponsored by: Institute of Computing Technology, CAS & China Computer Federation

Supervised by: Chinese Academy of Sciences

Undertaken by: Institute of Computing Technology, CAS

Published by: SCIENCE PRESS, BEIJING, CHINA

Printed by: Beijing Kexin Printing House

Distributed by:

China: All Local Post Offices

Other Countries: ORDER DEPT., SPRINGER, P.O. BOX 322, AH DORDRECHT, THE NETHERLAND

Available Online: www.springerlink.com

ISSN 1000-9000

国内统一连续出版物号: CN 11-2296/TP 国内邮发代号: 2-578 RMB ¥150.00