Computational Visual Media
https://doi.org/10.1007/s41095-018-0111-2 Vol. 4, No. 3, September 2018, 209–221
Research Article

Relief generation from 3D scenes guided by geometric texture richness

Yongwei Miao[1,2](✉), **Yuliang Sun**[2], **Xudong Fang**[2], **Jiazhou Chen**[2], **Xudong Zhang**[2], **and Renato Pajarola**[3]

Abstract Typically, relief generation from an input 3D scene is limited to either bas-relief or high-relief modeling. This paper presents a novel unified scheme for synthesizing reliefs guided by the geometric texture richness of 3D scenes; it can generate both bas- and high-reliefs. The type of relief and compression coefficient can be specified according to the user's artistic needs. We use an energy minimization function to obtain the surface reliefs, which contains a geometry preservation term and an edge constraint term. An edge relief measure determined by geometric texture richness and edge z-depth is utilized to achieve a balance between these two terms. During relief generation, the geometry preservation term keeps local surface detail in the original scenes, while the edge constraint term maintains regions of the original models with rich geometric texture. Elsewhere, in high-reliefs, the edge constraint term also preserves depth discontinuities in the higher parts of the original scenes. The energy function can be discretized to obtain a sparse linear system. The reliefs are obtained by solving it by an iterative process. Finally, we apply non-linear compression to the relief to meet the user's artistic needs. Experimental results show the method's effectiveness for generating both bas- and high-reliefs for complex 3D scenes in a unified manner.

Keywords geometric texture richness; relief generation; energy minimization; bas-relief; high-relief

1 College of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China. E-mail: ywmiao@zstu.edu.cn (✉).
2 College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China.
3 Department of Informatics, University of Zürich, Zürich CH-8050, Switzerland. E-mail: pajarola@ifi.uzh.ch.
Manuscript received: 2017-12-29; accepted: 2018-01-13

1 Introduction

With the rapid development of 3D digital scanning and modeling techniques, triangular meshes are now the dominant representation for 3D digital scenes. Computer graphics and digital geometry processing require various algorithms to effectively generate and edit 3D digital models [1–7]. As a traditional art form, reliefs bridge the gap between two-dimensional painting and three-dimensional sculpting. Nowadays, various types of reliefs are widely used to adorn shapes in the design of jewelry, industrial packaging, modern pieces of art, etc. [8]. However, the creation of complex reliefs is traditionally a time-consuming and tedious task. Advanced rapid prototyping technologies such as 3D printing [9, 10], which can be used to make a 3D object of almost any shape or geometry, can simplify the generation of reliefs.

According to their thickness, reliefs can be categorized into two main forms: high-reliefs (also called alto-relievo) which elevate perceptibly from a surface, and bas-reliefs (also called basso-relievo) which only rise to a minimal extent from a background [8]. Most research has focused on bas-relief generation, whilst little work has been done on high-relief generation. Many existing algorithms can only generate a single form of relief [8], either bas-relief or high-relief [11, 12]. It is obviously difficult and inconvenient for users to use two independent systems and sets of parameters if they wish to generate both types of reliefs from the same model: separate systems use different control parameters. Apart from reducing the user burden, another aim in constructing a unified framework is to find a detail-preserving method that is applicable to both

bas- and high-relief generation. We present a novel scheme for synthesizing digital reliefs which incorporates a geometric texture richness measure into the relief generation operation and combines it with z-depth compression techniques [13]. This approach can generate both bas- and high-reliefs via a unified modeling framework while preserving salient 3D shape features in a limited depth range (see Fig. 1).

Our main contributions are:

- We consider geometric texture richness in the relief generation process, giving a novel relief generation approach. The geometric texture richness is formulated in terms of variation of curvature, which is related to surface fairness and visual quality [14]. Using geometric texture richness has the advantage of preserving fine details in highly textured regions of complex shapes, whilst those in smoother regions are suppressed during relief generation.
- Bas-reliefs and high-reliefs can be generated in a unified modeling scheme by using an edge relief measure, which combines the geometric texture richness measure and the edge z-depth value.
- An iterative scheme is used, controlled by a parameter from an attenuation function, which prevents self-intersection of the relief and controls its thickness when compressing the original model.

Fig. 1 Bas-reliefs and high-reliefs generated from the Gargoyle model.

The rest of the paper is organized as follows. The main related work on generating bas-reliefs and high-reliefs is reviewed in Section 2. Section 3 gives our relief generation algorithm. Experimental results and discussions are given in Section 4. Finally, Section 5 concludes the paper and gives some future research directions.

2 Related work

2.1 Bas- and high-reliefs

As a kind of sculpture, reliefs squeeze 3D shapes into a solid piece of material exhibiting a narrow depth range while maintaining as much of the appearance of the full 3D scene as possible, and preserving its salient 3D shape characteristics [8, 15]. There are two types of reliefs, bas-reliefs [15–18] and high-reliefs [19]. High-reliefs contain figures that may stand out far from the flat relief plane, with the most prominent elements being undercut, while bas-reliefs (or low-reliefs) project the scene elements into a much narrower depth range. Various methods have been proposed to generate bas-reliefs and high-reliefs in recent years. We refer the reader to the survey in Ref. [8] and the references therein.

2.2 Bas-relief generation

Shapes in bas-relief are carved within a limited depth range and mimic the 3D scene as much as possible [20]. Recent work on bas-relief generation has focused on achieving a balance between height field compression and shape feature preservation.

2.2.1 Height fields

Cignoni et al. [11] first came up with the idea of using a height field by projecting the geometry of a scene onto the viewing plane. A linear compression function is determined from the height field. However, this approach fails to preserving the original model's fine details, despite the result looking like a bas-relief. Instead, Zhang et al. [12] first divided the 3D model into a detailed level and a base level. Then they separately compressed the detail and base levels into the viewing plane. This

still only partially keeps the fine surface details of the original model.

2.2.2 Gradient-domain filtering

Weyrich et al. [15], Kerber et al. [21], Bian and Hu [22], Zhou and Liu [23], and Sun et al. [24] have all used gradient coordinates [3] to keep the fine details of original mesh. They first extracted gradient coordinates from the objects in the input 3D scene. Functions were then used to adjust the gradient coordinates to enhance the surface details in the scene. Finally, they solved a sparse system of linear equations with boundary conditions in the height field based on a Poisson partial differential equation. Most of these methods take the 3D model as input and transform it into a height field. This intermediate representation may change the topology of the input and may lose local detail. Recently, based on gradient-based mesh deformation, Zhang et al. [25] have given a method which performs bas-relief generation directly on the triangular mesh.

2.2.3 Detail filtering and enhancement

Ji et al. [16, 17] presented a digital bas-relief modeling method which minimizes the difference between the Laplacian operators applied to the objects in the 3D scene and applied to the bas-relief. This method can produce different types of bas-relief, but it synthesizes the bas-relief from normals, not directly from the 3D scenes. Schüller et al. [26] proposed a bas-relief generation scheme which minimizes the difference between the 3D model's normal vectors and the relief's normal vectors. Going further, by considering the influence of illumination on bas-relief appearance, Zhang et al. [18] presented an adaptive method for bas-relief generation from 3D objects. Given certain settings for lighting conditions and material, their method can generate a bas-relief surface to match a desired target appearance.

2.3 High-relief generation

Cignoni et al. [11] was also first to propose a method to create high-reliefs. A non-linear compression function is determined by the height field, with small compression for points close to the viewing plane and larger compression for points further from the viewing plane. However, it fails to preserve fine details of the original model while generating high-reliefs. Arpa et al. [19] presented a high-relief generation method which maintains the features of the original model. They first selected a number of relevant scene points as attenuation points and moved them to the viewing plane. They then constructed the relief surface from the new positions of the attenuation points and differential coordinates of the original scene.

2.4 Comparison

Unlike existing relief creation methods, we give a novel unified method for synthesizing reliefs from 3D scenes, which can be applied to both bas- and high-reliefs. Salient fine details of the original models are retained when compressing the depth range of the scene elements. Our geometric texture richness guided relief generation algorithm is introduced in the following sections.

3 Geometric texture richness guided relief generation

3.1 Overview

The aim of this paper is to generate both bas- and high-reliefs using a unified modeling framework. The input is a triangular mesh, which can be represented as a three-tuple: $\boldsymbol{M} = \langle \boldsymbol{V}, \boldsymbol{E}, \boldsymbol{F} \rangle$, where $\boldsymbol{V}$ is the set of vertices, $\boldsymbol{E}$ is the set of edges, and $\boldsymbol{F}$ is the set of faces. The type of relief, i.e., whether the target relief is bas-relief or high-relief, can be specified by the user.

In order to synthesize reliefs from the input 3D scene, we use a relief generation framework based on energy minimization. The energy function comprises a geometry preservation term and an edge constraint term. The former preserves fine details of the original model. The latter maintains regions with rich geometric texture for bas-reliefs, and preserves both texture rich regions and high, standing out regions for high-reliefs. An edge relief measure is utilized in this energy function to achieve a balance between the geometry preservation term and the edge constraint term.

The motivation for incorporating geometric texture richness into our relief generation algorithm is that fine geometric features in texture rich regions should be preserved as much as possible, whilst those in smoother regions should be suppressed during relief generation. To achieve this, geometric texture richness is incorporated selectively by the edge

relief measure. The relationship between geometric texture richness and fine surface details is explained in Section 3.3.1. Subsequently, the edge relief measure is determined by geometric texture richness and edge z-depth. The proposed energy function is then discretized as a sparse linear system which is solved iteratively to determine z coordinates for the relief model. The target compression coefficient for the relief is specified by the user, which determines the scale of the target relief according to artistic needs.

The pipeline of our algorithm is shown in Fig. 2.

3.2 Relief generation by energy minimization

Defining the relief generation problem is essential to our method, as we aim to generate both bas-reliefs and high-reliefs through the same function. Our energy minimization function for generating reliefs can be defined as

$$E = \iint (\triangle \boldsymbol{v}' - \triangle \boldsymbol{v})^2 \mathrm{d}\boldsymbol{v} + \iint w(\boldsymbol{e}' - f(w)\boldsymbol{e})^2 \mathrm{d}\boldsymbol{e} \quad (1)$$

where $\triangle$ indicates the mesh Laplacian operator, $\boldsymbol{v}'$ and $\boldsymbol{v}$ are the vertices of the corresponding relief model $\boldsymbol{M}'$ and input model $\boldsymbol{M}$, $\boldsymbol{e}'$ and $\boldsymbol{e}$ are their respective edges, w denotes the edge relief measure for the 3D scene, and $f(w) = (1-l)w + l$. In our method, $f(w)$ and l define the compression range for scene edges: the ratio of the length of an edge of the relief model to the corresponding edge length in the original scene is in the range $[l, 1]$. An adaptive value for l is generated during iteration: see Section 3.4.

Equation (1) shows the two terms that make up our basic relief generation energy minimization function. The first, $\iint (\triangle \boldsymbol{v}' - \triangle \boldsymbol{v})^2 \mathrm{d}\boldsymbol{v}$, is a geometry preservation term, and the other is an edge constraint term. The former preserves fine surface details of the original 3D scene as it ensures that the relief mesh maintains the original mesh's Laplacian coordinates [2, 4]. Previous works use gradient coordinates [3] in a similar geometry preservation term in conjunction with a Poisson partial differential equation [15, 21–24], which requires solving a sparse system of linear equations with respect to boundary conditions from $\boldsymbol{M}$. Our function uses the Laplacian instead of gradients to keep the fine details in a simple manner. Meanwhile, the purpose of the edge constraint term $\iint w(\boldsymbol{e}' - f(w)\boldsymbol{e})^2 \mathrm{d}\boldsymbol{e}$ is that edges of complex shapes with bigger w always have more fine geometric details and thus should be preserved as much as possible, whilst other edges with smaller w have less surface details and thus may be suppressed during relief generation. Here, an edge constraint, rather than a vertex constraint, is applied. The motivation is that an edge constraint can better prevent self-

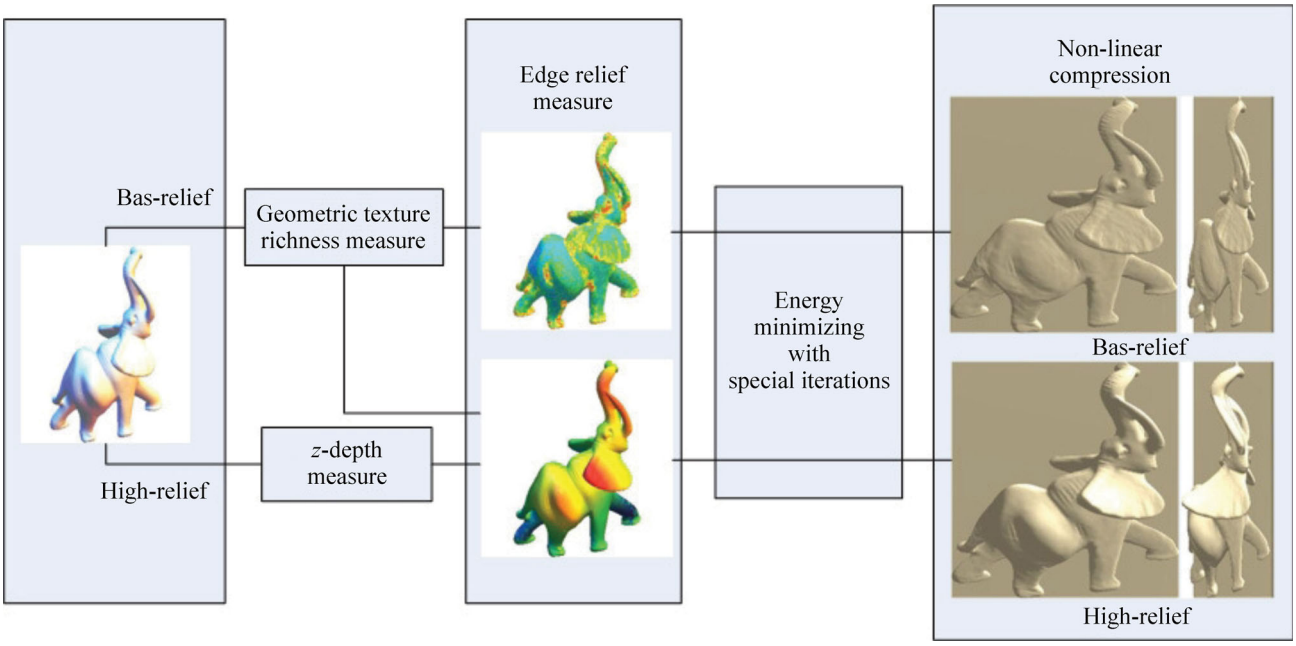


Fig. 2 Pipeline of our relief generation algorithm.

intersections of the relief, since it can constrain depth relationships of vertices.

3.3 Edge relief measure

It is important to select an appropriate edge relief measure w since it defines the degree of fine geometric details to be preserved. In this section, we discuss its definition.

3.3.1 Geometric texture richness measure

Smooth regions of complex surfaces do not have fine details, no matter whether they have low curvature or high curvature. Regions with rich geometric texture, however, always have fine surface details. Thus, we base our geometric texture richness measure on variation in curvature. In general, the higher the richness measure of a region, the finer shape features this region has, and vice versa. This geometric texture richness measure is incorporated in our method in order to preserve regions with high richness and compress those with low richness. Figure 3 illustrates the Bimba model, and others, and their corresponding geometric texture richness maps. Warmer colors in the map indicate higher richness. Regions with higher richness, such as the hair, have more surface details than the regions with lower richness, such as the chest and face. Similar relationships can also be found in the Lion, Pegasus, and Elephant models.

The concept of local final richness (LFR) [14] is used in this paper to represent mesh geometric texture richness. To compute LFR, we first calculate the Laplacian of the Gaussian curvature to give the local richness (LR). Thus LR at $\boldsymbol{v}_i$ is defined as

$$\mathrm{LR}_i = \left| K_i - \frac{\sum_{j\in N(\boldsymbol{v}_i)} \boldsymbol{D}_{i,j} K_j}{\sum_{j\in N(\boldsymbol{v}_i)} \boldsymbol{D}_{i,j}} \right| \tag{2}$$

where K_i is the Gaussian curvature at $\boldsymbol{v}_i$ [27], and $\boldsymbol{D}$ is the mesh Laplacian matrix which can be computed using the discrete exterior calculus [28]. Next, taking into account the visual masking effect explained in Ref. [29] (that is, a textured region can hide geometric distortion much better than a smooth one), LFR is calculated by careful modulation of LR:

$$\mathrm{LFR}_i = \begin{cases} g(\overline{\mathrm{LR}}) + \left|\mathrm{LRM}_i - g(\overline{\mathrm{LR}})\right|/2, & \text{for } \mathrm{LRM}_i > g(\overline{\mathrm{LR}}) \\ \mathrm{LRM}_i, & \text{otherwise} \end{cases} \tag{3}$$

where $\mathrm{LRM}_i = g(\mathrm{LR}_i)$, $\overline{\mathrm{LR}}$ is the average richness, and the modulation function $g(x) = x^{0.15} - (5.0 \times 10^{-4})^{0.15}$. Finally, LFR_i is normalized to $[0, 1]$. Since the final local richness measure LFR_i at each vertex $\boldsymbol{v}_i$ has already been well defined, we can simply determine the geometric texture richness value for each edge $\boldsymbol{e}_{ij}$ linking $\boldsymbol{v}_i$ to $\boldsymbol{v}_j$ as the maximum LFR value of the two endpoints: $r_{ij} = \max(\mathrm{LFR}_i, \mathrm{LFR}_j)$.

3.3.2 Edge relief measure

The height field, also called a range image or depth map, is an essential representation for relief generation technology. According to Ref. [11], z-depth is significant in high-relief generation.

Fig. 3 Geometric texture richness maps for different models (warmer colors indicate higher richness).

Specifically, high edges that are detached completely from the background are those edges with large z-depth in the input model.

Here, we define the vertex z-depth measure d_i of $\boldsymbol{v}_i$ as: $d_i = (\boldsymbol{v}_i^z - z_{\min})/(z_{\max} - z_{\min})$, where $\boldsymbol{v}_i^z$ is z-coordinate at vertex $\boldsymbol{v}_i$, $z_{\min}$ is the minimum z-depth over all $\boldsymbol{v}_i^z$, and $z_{\max}$ is the maximum value. The z-depth measure for each edge $\boldsymbol{e}_{ij}$ is defined as the average z-depth of its endpoints: $d_{ij} = (d_i + d_j)/2$. Thus, by considering the geometric texture richness measure and the edge z-depth value, the relief measure w_{ij} for each edge $\boldsymbol{e}_{ij}$ can be defined as follows:

$$w_{ij} = \max(td_{ij}, r_{ij}) \tag{4}$$

where t is the type of relief, $t = 1$ meaning high-relief and $t = 0$ meaning bas-relief. From its definition, w_{ij} lies in the interval [0, 1].

3.4 Intermediate relief generation

For general 3D scenes, the input data is discretized and represented as triangular meshes. In this section, we discuss discretization of the relief generation energy minimization function in Eq. (1). Firstly, it can be rewritten in discrete form as

$$E = \sum_{\boldsymbol{v}_i \in \boldsymbol{V}} \|L(\boldsymbol{v}_i') - L(\boldsymbol{v}_i)\|^2 + \sum_{\boldsymbol{e}_{ij} \in \boldsymbol{E}} w_{ij} \|\boldsymbol{v}_i' - \boldsymbol{v}_j' - f(w_{ij})(\boldsymbol{v}_i - \boldsymbol{v}_j)\|^2 \tag{5}$$

where L is the discrete mesh Laplacian operator [2, 4], given by $L(\boldsymbol{v}_i) = \boldsymbol{v}_i - \frac{1}{d_i}\sum_{j \in N(\boldsymbol{v}_i)} \boldsymbol{v}_j$. In the above, $\boldsymbol{v}_i'$, $\boldsymbol{v}_j'$ and $\boldsymbol{v}_i$, $\boldsymbol{v}_j$ are vertices of the relief model $\boldsymbol{M}'$ and input model $\boldsymbol{M}$, respectively, w_{ij} is the edge relief measure, and $f(w_{ij})$ is calculated as $f(w_{ij}) = (1 - l)w_{ij} + l$ as explained earlier: if the edge relief measure w_{ij} is higher, $f(w_{ij})$ is closer to 1, so edges with high geometric texture richness are preserved when minimizing the relief generation function. On the other hand, if w_{ij} is lower, $f(w_{ij})$ will be closer to l, so edges with low texture richness will be compressed during relief generation.

Equation (5) contains geometry preservation and edge constraint terms. By maintaining the Laplacian coordinates of the original model, the first term preserves local fine details in the input 3D scene. Meanwhile, the second term determines edge constraints from the original model. Edges with larger w represent more salient features which should be retained, whilst edges with smaller w may be compressed to a larger degree during relief generation.

During relief generation, while maintaining the topological connectivity of the underlying mesh unchanged, vertex positions are updated based on the solution of a sparse linear system, which we now derive.

For each vertex $\boldsymbol{v}_i'$ $(i = 1, \ldots, n)$, the derivative of the energy E should be zero, so $\partial E/\partial \boldsymbol{v}_i' = 0$, and therefore:

$$\begin{aligned} &\boldsymbol{v}_i' + \sum_{j \in N(\boldsymbol{v}_i)} w_{ij}(\boldsymbol{v}_i' - \boldsymbol{v}_j') \\ &\quad + \sum_{j \in N(\boldsymbol{v}_i)} \Big[\Big(-\frac{1}{d_i} - \frac{1}{d_j}\Big)\boldsymbol{v}_j' + \frac{1}{d_j^2} \sum_{k \in N(\boldsymbol{v}_j)} \boldsymbol{v}_k'\Big] \\ &= L(\boldsymbol{v}_i) + \sum_{j \in N(\boldsymbol{v}_i)} w_{ij} f(w_{ij})(\boldsymbol{v}_i - \boldsymbol{v}_j) - \sum_{j \in N(\boldsymbol{v}_i)} \frac{1}{d_j} L(\boldsymbol{v}_j) \end{aligned} \tag{6}$$

Given this linear system, we only solve for the z coordinates (using the Intel MKL library [30]), since reliefs should keep the original vertex x and y coordinates from the input scene, assuming the normal vector of the relief plane to be parallel to the z-axis. Unfortunately, even if we set l to some given fraction, the resulting compression factor of the actual relief with respect to the original scene will be much greater: it is difficult to fully compress the original model into the desired relief. Therefore, we use a modified parameter l and a number of iterations as shown in Algorithm 1.

Algorithm 1 Relief generation

Input: Input model $\boldsymbol{M}$, type of relief t.
Output: Output relief model $\boldsymbol{M}'$.
Step 1: Set intermediate model $\boldsymbol{M}_0 = \boldsymbol{M}$, $k = 1$;
do
  Step 2.1: Calculate edge relief measures w_{ij} for $\boldsymbol{M}_{k-1}$;
  Step 2.2: Calculate function values $f(w_{ij})$;
  Step 2.3: Calculate the modified parameter using the attenuation function $l = h(k) = 1 - 1/2^{k-1}$;
  Step 2.4: Solve the sparse linear system (6) to obtain $\boldsymbol{M}_k$ and its thickness T_k;
while $|T_{k-1}/T_0 - T_k/T_0| > 0.05$
Step 3: Set output model $\boldsymbol{M}' = \boldsymbol{M}_k$.

For the iteration step, we set $l = 0$ initially, which causes the ratio of lengths of all edges of the relief model to their corresponding edges of the original scene to lie in the interval $[0, 1]$. The intention is that, during relief generation, the relief should be compressed as much as possible while preserving the

salient features of the original model to the greatest possible extent. In order to maintain the original surface features, the parameter l is progressively adjusted by the attenuation function $h(k) = 1 - 1/2^{k-1}$. Thus, in the second iteration, l will be 0.5 and edge ratios will be fixed to lie in the interval $[0.5, 1]$. This will retain the original surface details for $l = 0.5$, providing smooth compression of the edges. Thus, the iterations reduce loss of salient features from the original model and avoid self-intersections in the relief model. Iteration stops if there is no significant change of thickness.

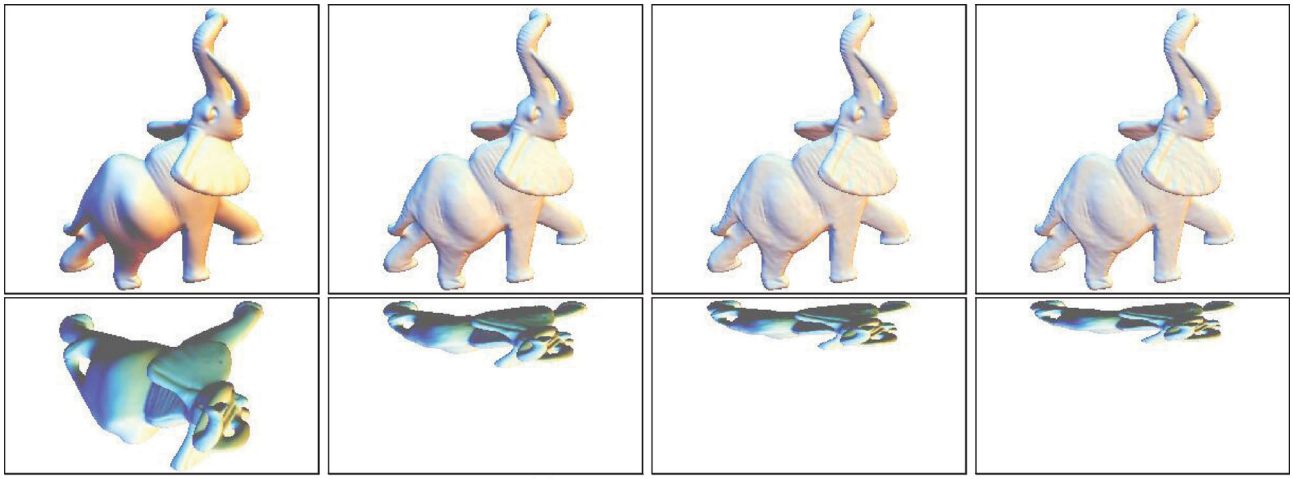
Figure 4 shows bas- and high-relief results for the Elephant model after each iteration. The first two rows show front and top views of the bas-reliefs respectively. The next two rows show the same for high-reliefs. The more iterations are used, the thinner the reliefs become. For this model, after 4 iterations, the thickness of the reliefs remains nearly unchanged. In the bas-relief and high-relief results, parts compress differently, especially the ear and the head, which compress non-homogeneously in the high-relief, demonstrating the effectiveness of Eq. (4).

3.5 Compression control

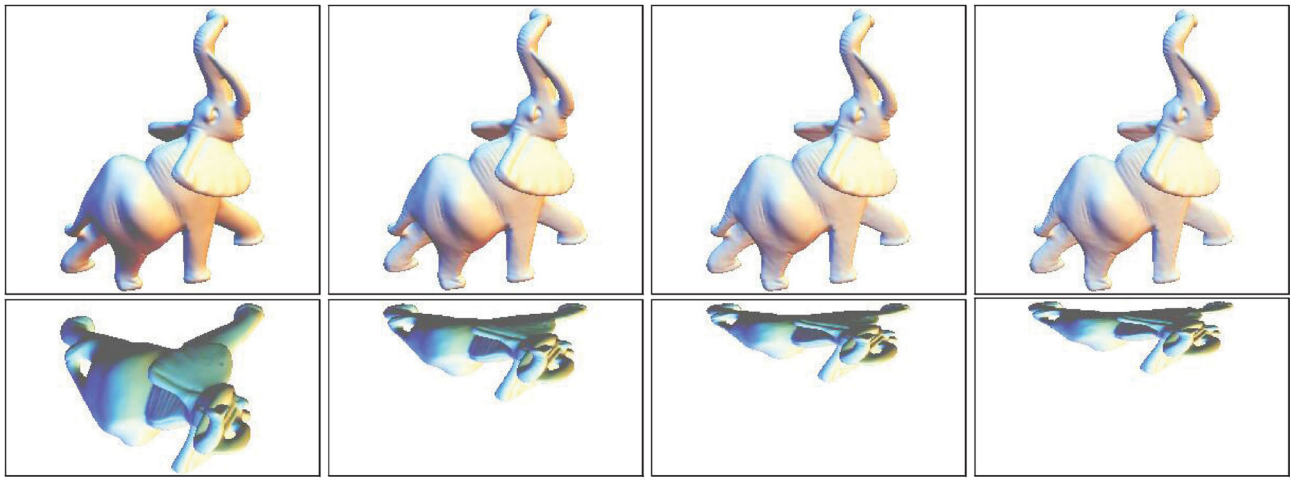
The thickness of the relief produced by the above method may not meet the user's artistic needs for a desired depth range in some cases. Following Ref. [11], we introduce a compression operation on the relief model above $\boldsymbol{M}' = \{\boldsymbol{v}'_1, \ldots, \boldsymbol{v}'_n\}$ to obtain the final relief model $\boldsymbol{M}'' = \{\boldsymbol{v}''_1, \ldots, \boldsymbol{v}''_n\}$:

$$\boldsymbol{v}''^{x}_i = \boldsymbol{v}'^{x}_i, \qquad \boldsymbol{v}''^{y}_i = \boldsymbol{v}'^{y}_i, \qquad \boldsymbol{v}''^{z}_i = s_i \boldsymbol{v}'^{z}_i \tag{7}$$

where the compression ratio for the z-coordinate is determined by $s_i = \widehat{\boldsymbol{v}'^{z}_i} \times (m_1 - m_2) \times t + m_2$ with compression coefficients $m_1 > m_2$, and $\widehat{\boldsymbol{v}'^{z}_i}$ being the



(a) Bas-relief generation



(b) High-relief generation

Fig. 4 Bas- and high-relief results for the Elephant model after each iteration. Increasing the number of iterations flattens the scene to the relief plane and decreases the overall depth range.

z-coordinate of $\boldsymbol{v}_i'^z$ normalized into the interval $[0,1]$, that is $\widehat{\boldsymbol{v}_i'^z} = (\boldsymbol{v}_i'^z - \boldsymbol{v}_{\min}'^z)/(\boldsymbol{v}_{\max}'^z - \boldsymbol{v}_{\min}'^z)$.

When $t = 1$, the resulting reliefs will be high-reliefs, and the compression ratio s_i will be in the range $[m_2, m_1]$ which is related to the z-depth of the mesh $\boldsymbol{M}'$. The compression ratio will lead to the lower vertices being much more flat and the higher vertices standing out much more from the flat relief plane.

When $t = 0$, the reliefs are bas-reliefs, and the compression ratios are defined by the constant m_2.

However, it is difficult for the user to decide on these two parameters. Therefore, we fix $m_1 = m$ and $m_2 = 3m/8$ in our experiments, and set the compression ratio s_i as $s_i = \widehat{\boldsymbol{v}_i'^z}(5m/8)t + 3m/8$; $m \leqslant 1$.

As shown in Fig. 5, we obtain different results according to the choice of the compression coefficient m, which can be set by the user to meet artistic needs. Figure 5(a) shows the high-relief generated with $m = 1.0$. It can be observed from Figs. 5(b)–5(d) that final high-relief height decreases with decreasing m.

4 Results and discussion

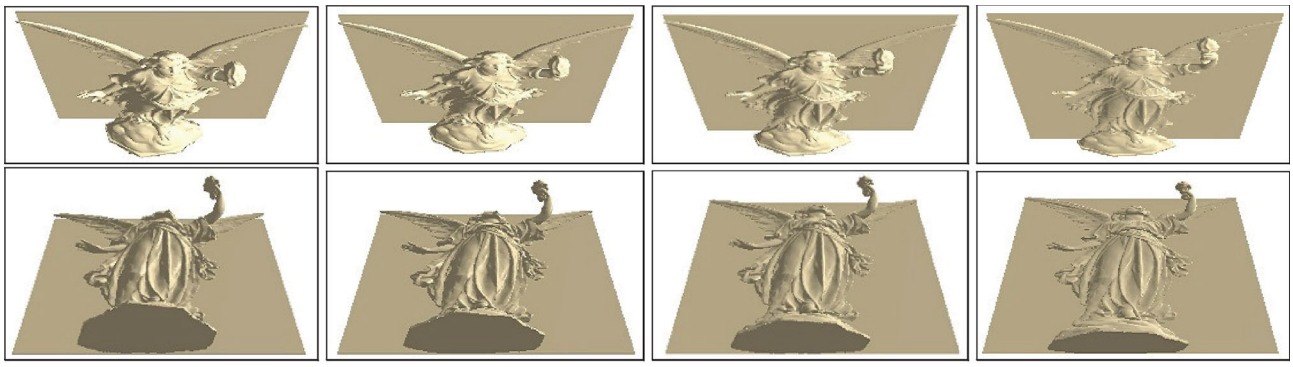
Our algorithm has been implemented on an Intel Core 2 Duo E7500, 2.93 GHz CPU with 2 GB RAM using Microsoft Studio 2005. The proposed energy minimization function is discretized and the resulting linear system is solved iteratively. A compression operation is applied to generate the final reliefs. To demonstrate the effectiveness of our method, we use several models as input and present experimental results and comparisons in the following.

As explained in Section 3, different results can be obtained by varying the user-chosen relief type t and compression coefficient m. In the following, bas-reliefs are generated with $m = 0.8$ and $t = 0$, while high-reliefs are generated with $m = 0.8$ and $t = 1$.

Figure 6 compares results from applying a simple compression method (Figs. 6(a) and 6(c)) and our method (Figs. 6(b) and 6(d)). Figure 6(a) scales the Lucy model to 0.12, whilst Fig. 6(c) scales the Lucy model to 0.32. The reliefs generated by our method are scaled to the same degree. Results in Figs. 6(a) and 6(c) are similar to those from a traditional method [11], and do not preserve the fine details of the original model well. In comparison, our corresponding bas-relief (Fig. 6(b)) and high-relief (Fig. 6(d)) have more salient features, especially in the head, wing, and the clothes, due to our use of geometric texture richness and iteration. Figure 7 provides another comparison with Ref. [11]. Unlike Fig. 7(a) and Fig. 7(c), our generated bas-relief (Fig. 7(b)) and high-relief (Fig. 7(d)) maintain geometric texture richness especially in the clothes and base regions.

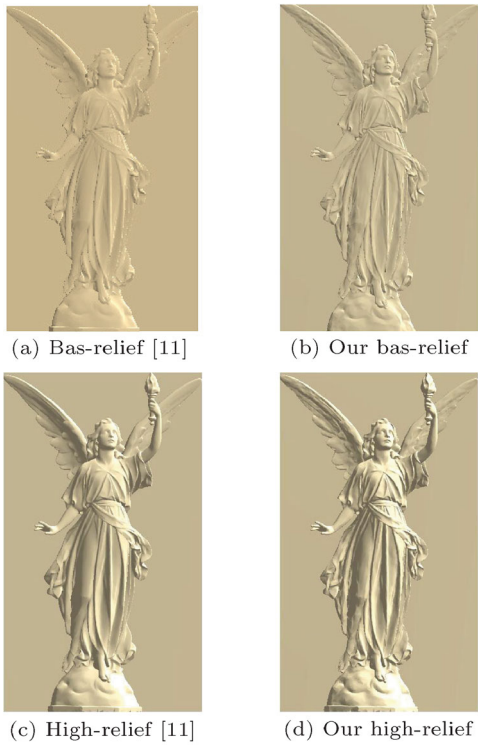
A comparison with a state-of-art method based on height field compression is presented in Fig. 8, which compares bas-relief using Ref. [12] (Fig. 8(a)) and our method (Fig. 8(b)) for the Lucy model. Our bas-relief has more salient features especially in the head and the clothes.

Figure 9 provides a further comparison of these two methods. The bas-relief generated by our method (Fig. 9(b)) preserves more geometric details. In addition, our corresponding high-relief maintains the appearance even when the viewpoint of the high-



(a) $m = 1.0$ (b) $m = 0.8$ (c) $m = 0.6$ (d) $m = 0.4$

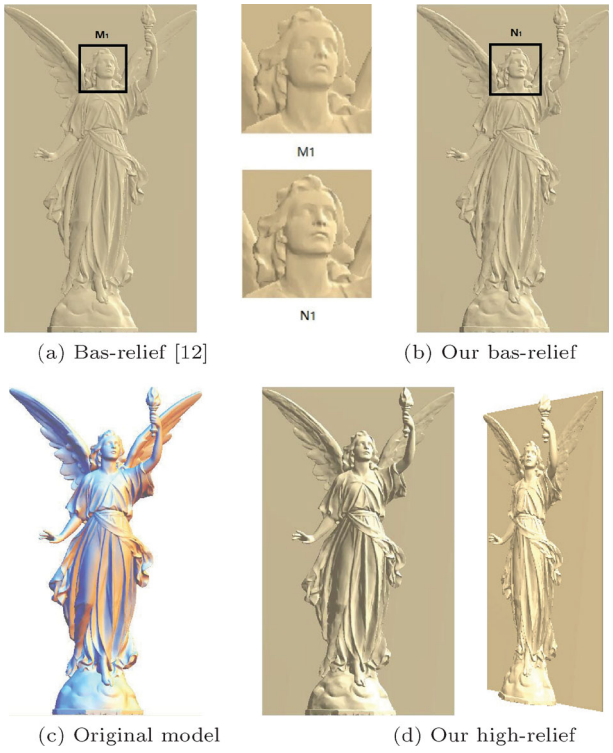
Fig. 5 Compression achieved by varying control coefficient m.



(a) Bas-relief [11] (b) Our bas-relief

(c) High-relief [11] (d) Our high-relief

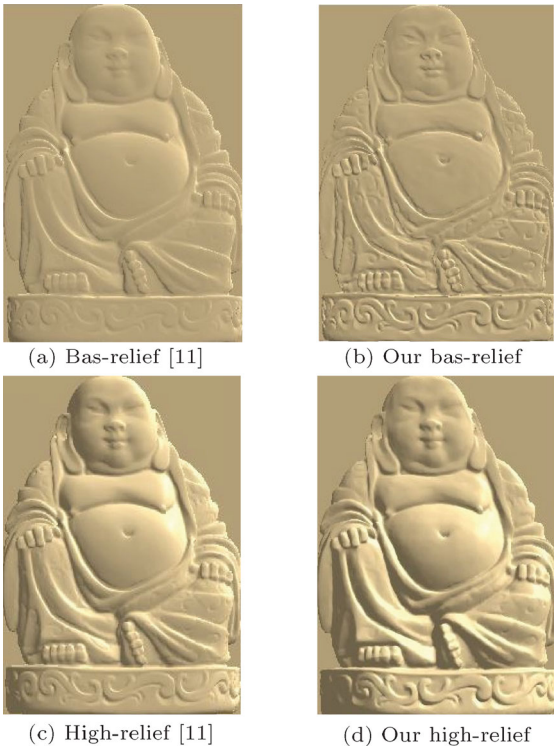
Fig. 6 Results from (a, c) a simple compression method following Ref. [11] and (b, d) our method, for the Lucy model.



(a) Bas-relief [12] (b) Our bas-relief

(c) Original model (d) Our high-relief

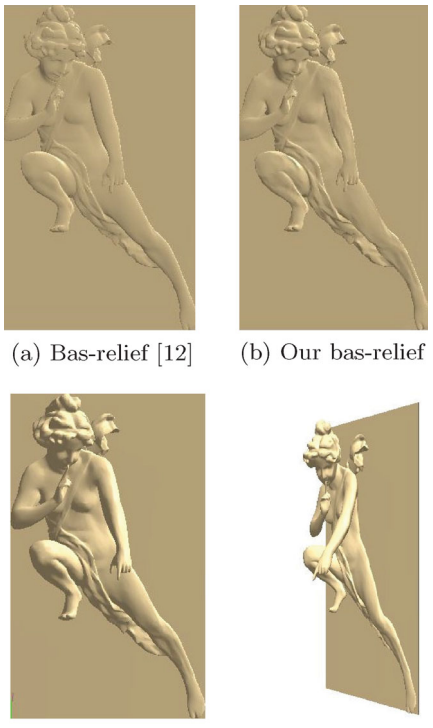
Fig. 8 A bas-relief produced using the method in Ref. [12] (a), and bas-relief (b) and high-relief (d) produced using our method, for the Lucy model (c).



(a) Bas-relief [11] (b) Our bas-relief

(c) High-relief [11] (d) Our high-relief

Fig. 7 Results from (a, c) a simple compression method following Ref. [11] and (b, d) our method, for the Buddha model.



(a) Bas-relief [12] (b) Our bas-relief

(c) Our high-relief

Fig. 9 A bas-relief produced using the method in Ref. [12] (a), and bas-relief (b) and high-relief (c) produced using our method, for the Angel model.

relief changes. Unfortunately, our iteration-based relief generation method is not optimized for performance, and requires several iterations. It takes 78 s to generate the high-relief and bas-relief for the Lucy model. Table 1 provides statistics and timing for other models presented in this paper.

Figure 10 and Fig. 11 show bas-relief (upper row) and high-relief examples (lower row) for different models and various materials. We give the front view and side view in each case. For different 3D models, bas-reliefs and high-reliefs can be generated through the unified method while maintaining salient features of complex shapes within a limited depth range. Users can generate either bas-reliefs or high-reliefs by simply modifying the parameter t. These examples demonstrate the robustness and versatility of our proposed method.

Table 1 Statistics and timing

Model	Vertices	Triangles	Running time
Angel	23697	47404	61 s
Buddha	24955	49918	77 s
Pegasus	31700	63416	52 s
Armadillo	51901	103794	87 s
Lucy	52581	105158	78 s
Lion	61122	122240	101 s
Gargoyle	62498	124994	129 s

5 Conclusions and future work

In this paper, we have presented a novel scheme for synthesizing controllable digital reliefs, guided by geometric texture richness. Our approach can generate both bas- and high-reliefs via a unified modeling framework. Experimental results indicate that our proposed method is an effective and versatile scheme for generating bas- and high-reliefs.

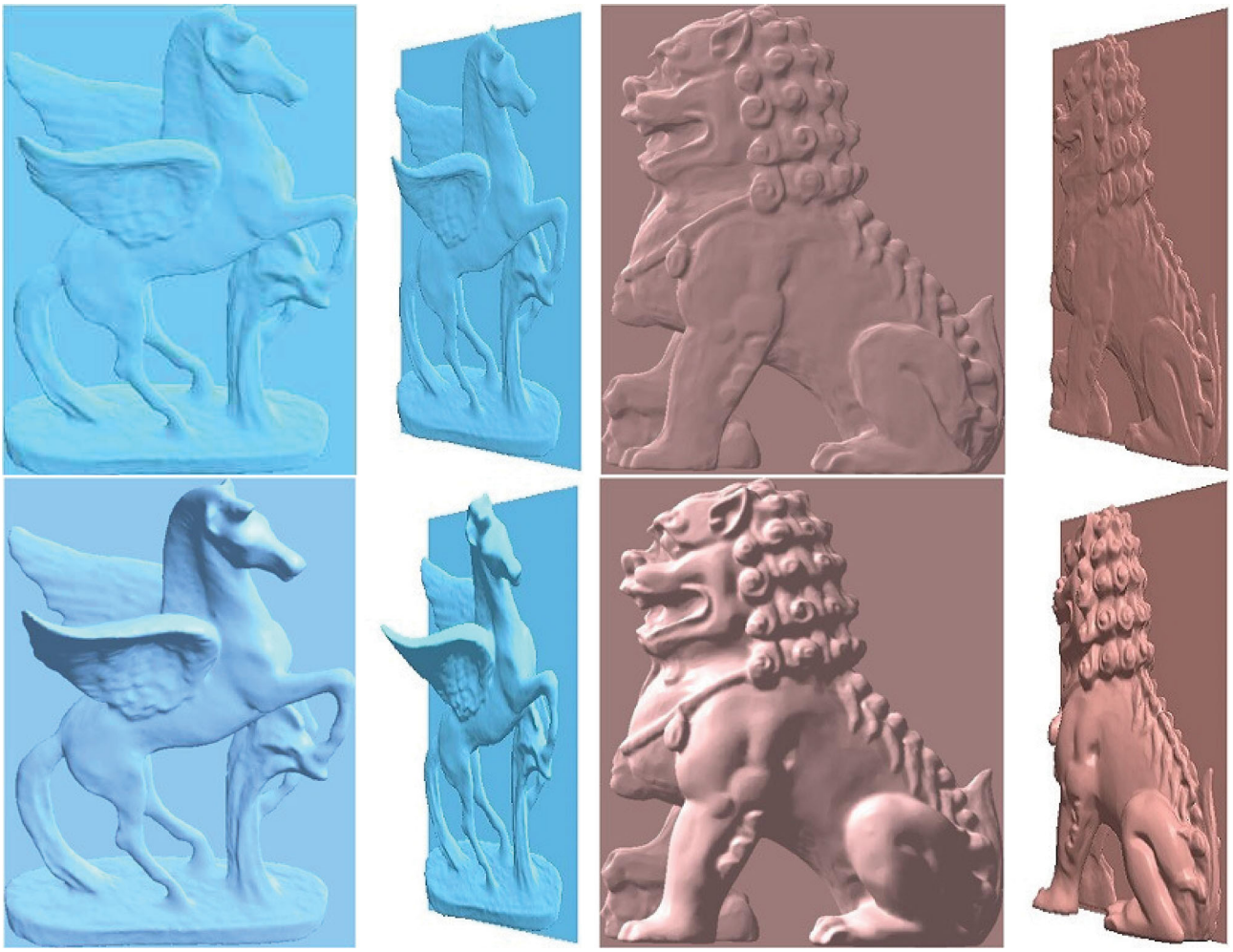


Fig. 10 Bas-relief examples (above) and high-relief examples (below) for the Pegasus and Lion models.

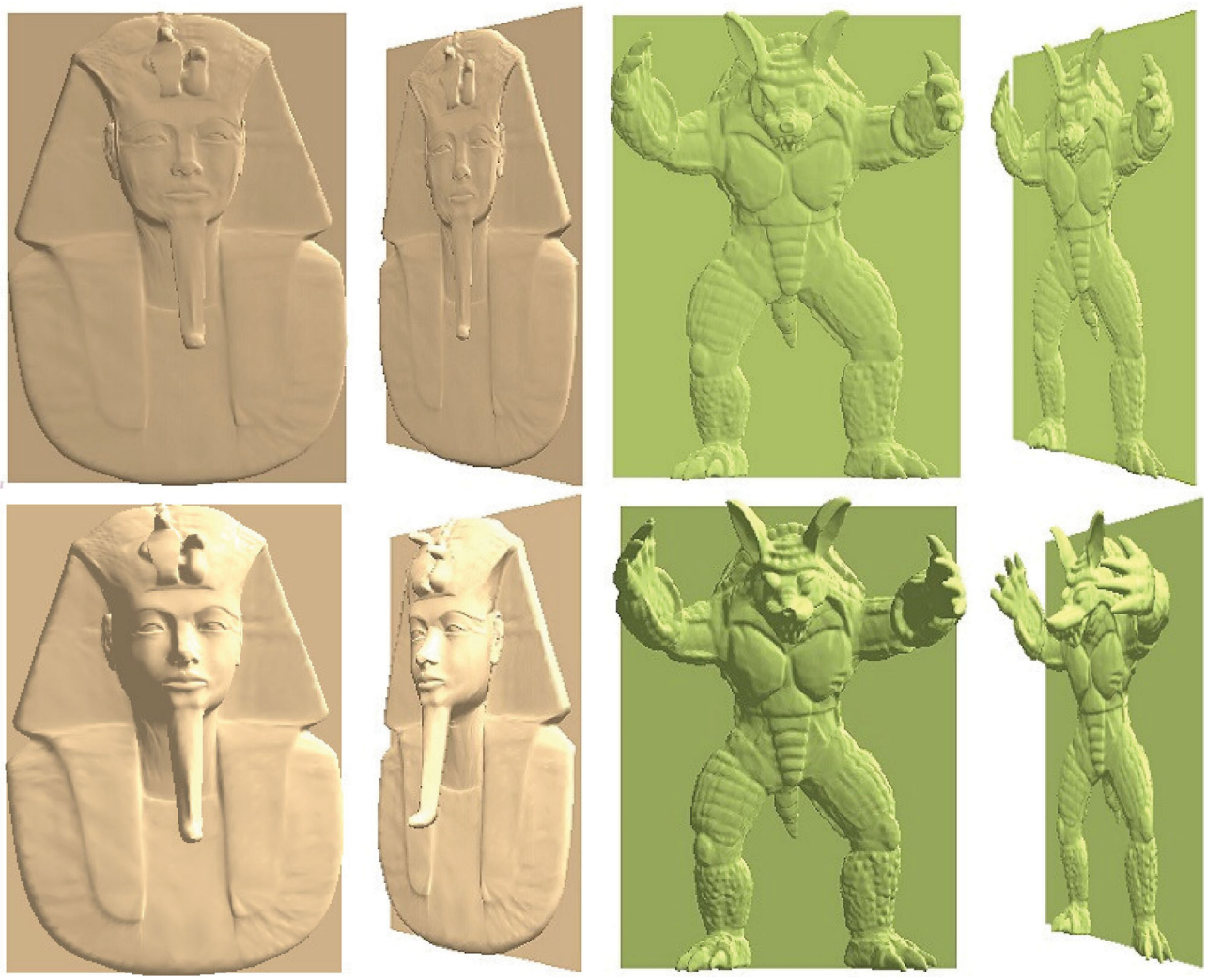


Fig. 11 Bas-relief examples (above) and high-relief examples (below) for the Pharaoh and Armadillo models.

However, our relief generation method is limited to 3D scenes with topological connectivity information, in particular 3D triangular meshes. It is not applicable to unconnected complex models such as point cloud models.

Our unified framework is a step on the path of addressing the problem of bas- and high-relief generation. Using this novel framework, users can obtain both types of reliefs from the same 3D model. This approach has widespread potential usage in house decoration and artistic design. Our future research will focus on a robust unified relief generation algorithm with changing lighting and colour.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No. 61272309. The 3D models are courtesy of the Aim@Shape and Archive3d.

References

[1] Botsch, M.; Pauly, M.; Kobbelt, L.; Alliez, P.; Lévy, B.; Bischoff, S.; Rössl, C. Geometric modeling based on polygonal meshes. In: Proceedings of the ACM SIGGRAPH 2007 Course, Article No. 1, 2007.

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

[3] Yu, Y.; Zhou, K.; Xu, D.; Shi, X.; Bao, H.; Guo, B.; Shum, H.-Y. Mesh editing with Poisson-based gradient field manipulation. *ACM Transactions on Graphics* Vol. 23, No. 3, 644–651, 2004.

[4] Zhou, K.; Huang, J.; Snyder, J.; Liu, X.; Bao, H.; Guo, B.; Shum, H.-Y. Large mesh deformation using the volumetric graph Laplacian. *ACM Transactions on Graphics* Vol. 24, No. 3, 496–503, 2005.

[5] Lipman, Y.; Sorkine, O.; Levin, D.; Cohen-Or, D. Linear rotation-invariant coordinates for meshes. *ACM Transactions on Graphics* Vol. 24, No. 3, 479–487, 2005.

[6] Botsch, M.; Sorkine, O. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics* Vol. 14, No. 1, 213–230, 2008.

[7] Miao, Y.-W.; Feng, J.-Q.; Wang, J.-R.; Pajarola, R. A multi-channel salience based detail exaggeration technique for 3D relief surfaces. *Journal of Computer Science and Technology* Vol. 27, No. 6, 1100–1109, 2012.

[8] Kerber, J.; Wang, M.; Chang, J.; Zhang, J. J.; Belyaev, A.; Seidel, H.-P. Computer assisted relief generation—A survey. *Computer Graphics Forum* Vol. 31, No. 8, 2363–2377, 2012.

[9] Cho, W.; Sachs, E.-M.; Patrikalakis, N.-M.; Troxel, D.-E. A dithering algorithm for local composition control with three-dimensional printing. *Computer-Aided Design* Vol. 35, No. 9, 851–867, 2003.

[10] Jee, H. J.; Sachs, E. A visual simulation technique for 3D printing. *Advances in Engineering Software* Vol. 31, No. 2, 97–106, 2000.

[11] Cignoni, P.; Montani, C.; Scopigno, R. Computer-assisted generation of bas- and high-reliefs. *Journal of Graphics Tools* Vol. 2, No. 3, 15–28, 1997.

[12] Zhang, Y.-W.; Zhou, Y.-Q.; Zhao, X.-F.; Yu, G. Real-time bas-relief generation from a 3D mesh. *Graphical Models* Vol. 75, No. 1, 2–9, 2013.

[13] Miao, Y.; Lin, H. Visual saliency guided global and local resizing for 3D models. In: Proceedings of the 13th International Conference on Computer-Aided Design and Computer Graphics, 212–219, 2013.

[14] Wang, K.; Torkhani, F.; Montanvert, A. A fast roughness-based approach to the assessment of 3D mesh visual quality. *Computers & Graphics* Vol. 36, No. 7, 808–818, 2012.

[15] Weyrich, T.; Deng, J.; Barnes, C.; Rusinkiewicz, S.; Finkelstein, A. Digital bas-relief from 3D scenes. *ACM Transactions on Graphics* Vol. 26, No. 3, Article No. 32, 2007.

[16] Ji, Z.; Ma, W.; Sun, X. Bas-relief modeling from normal images with intuitive styles. *IEEE Transactions on Visualization and Computer Graphics* Vol. 20, No. 5, 675–685, 2014.

[17] Ji, Z.; Sun, X.; Li, S.; Wang, Y. Real-time bas-relief generation from depth-and-normal maps on GPU. *Computer Graphics Forum* Vol. 33, No. 5, 75–83, 2014.

[18] Zhang, Y.-W.; Zhang, C.; Wang, W.; Chen, Y. Adaptive bas-relief generation from 3D object under illumination. *Computer Graphics Forum* Vol. 35, No. 7, 311–321, 2016.

[19] Arpa, S.; Süsstrunk, S.; Hersch, R. D. High reliefs from 3D scenes. *Computer Graphics Forum* Vol. 34, No. 2, 253–263, 2015.

[20] Belhumeur, P. N.; Kriegman, D. J.; Yuille, A. L. The bas-relief ambiguity. *International Journal of Computer Vision* Vol. 35, No. 1, 33–44, 1999.

[21] Kerber, J.; Tevs, A.; Belyaev, A.; Zayer, R.; Seidel, H.-P. Feature sensitive bas relief generation. In: Proceedings of the IEEE International Conference on Shape Modeling and Applications, 148–154, 2009.

[22] Bian, Z.; Hu, S.-M. Preserving detailed features in digital bas-relief making. *Computer Aided Geometric Design* Vol. 28, No. 4, 245–256, 2011.

[23] Zhou, S; Liu, L. Realtime digital bas-relief modeling. *Journal of Computer-Aided Design & Computer Graphics* Vol. 22, No. 3, 434–439, 2010.

[24] Sun, X.; Rosin, P. L.; Martin, R. R.; Langbein, F. C. Bas-relief generation using adaptive histogram equalization. *IEEE Transactions on Visualization and Computer Graphics* Vol. 15, No. 4, 642–653, 2009.

[25] Zhang, Y. W.; Zhou, Y. Q.; Li, X. L.; Liu, H.; Zhang, L. L. Bas-relief generation and shape editing through gradient-based mesh deformation. *IEEE Transactions on Visualization and Computer Graphics* Vol. 21, No. 3, 328–338, 2015.

[26] Schüller, C.; Panozzo, D.; Sorkine-Hornung, O. Appearance-mimicking surfaces. *ACM Transactions on Graphics* Vol. 33, No. 6, Article No. 216, 2014.

[27] Meyer, M.; Desbrun, M.; Schröder, P.; Barr, A. H. Discrete differential-geometry operators for triangulated 2-manifolds. In: *Visualization and mathematics III*. Hege, H. C.; Polthier, K. Eds. Springer-Verlag, 35–57, 2003.

[28] Vallet, B.; Lévy, B. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum* Vol. 27, No. 2, 251–260, 2008.

[29] Lavoué, G. A local roughness measure for 3D meshes and its application to visual masking. *ACM Transactions on Applied Perception* Vol. 5, No. 4, Article No. 21, 2009.

[30] Intel Math Kernel Library (Intel MKL). Available at http://soft-ware.intel.com/en-us/intel-mk.

Yongwei Miao received his Ph.D. degree in computer graphics from the State Key Lab. of CAD & CG at Zhejiang University in March 2007. From February 2008 to February 2009, he worked as a visiting scholar in the University of Zürich, Switzerland. From November 2011 to May 2012, he worked as a visiting scholar in the University of Maryland, USA. From July 2015 to August 2015, he worked as a visiting professor in the University of Tokyo, Japan. Dr. Miao is currently a professor in the College of Information Science and Technology, Zhejiang Sci-Tech University, China, and also a professor in the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include computer graphics, digital geometry processing, visual media computing, 3D reconstruction, and computer vision.

Yuliang Sun is a Ph.D. student in the College of Computer Science and Technology, Zhejiang University of Technology, China. He received his master degree from the University of Manchester, UK, in 2011 and received his bachelor degree from Dalian University of Technology, China, in 2010. His research interests include computer graphics, computer vision, and digital geometry processing.

Xudong Fang received his master degree in software engineering from the College of Computer Science and Technology, Zhejiang University of Technology, China, in 2016. He received his bachelor degree from Zhejiang Chinese Medical University, China, in 2013. His research interests include computer graphics and digital geometry processing.

Jiazhou Chen is an associate professor in the College of Computer Science and Technology, Zhejiang University of Technology, China. Before that, he was a joint Ph.D. student between Bordeaux University, France, and Zhejiang University, China, obtaining a French doctoral diploma from Bordeaux University in July 2012, and also a Chinese doctoral diploma from Zhejiang University in December 2012. His research interests include computer graphics, image and video stylization, augmented reality, and visual media computing.

Xudong Zhang is a faculty member of the College of Computer Science and Technology, Zhejiang University of Technology, China. His research interests include computer graphics, visual media computing, and computer vision.

Renato Pajarola received his Dr.Sc.Techn. in computer science in 1998 from the Swiss Federal Institute of Technology (ETH) Zürich. After a postdoc in the Graphics, Visualization and Usability Center at Georgia Tech., he joined the University of California Irvine in 1999 as an assistant professor where he founded the Computer Graphics Lab. He is currently a professor in the Department of Informatics, University of Zürich, Switzerland. His research interests include real-time 3D graphics, scientific visualization, and interactive 3D multimedia.