Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Image driven shape deformation using styles*

Guang-hua TAN[1], Wei CHEN[2], Li-gang LIU[†‡2,3]

([1]*School of Computer and Communication, Hunan University, Changsha 410082, China*)

([2]*State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, China*)

([3]*Department of Mathematics, Zhejiang University, Hangzhou 310027, China*)

[†]E-mail: ligangliu@zju.edu.cn

Received Feb. 13, 2009; Revision accepted June 3, 2009; Crosschecked Sept. 29, 2009

Abstract: In this paper, we propose an image driven shape deformation approach for stylizing a 3D mesh using styles learned from existing 2D illustrations. Our approach models a 2D illustration as a planar mesh and represents the shape styles with four components: the object contour, the context curves, user-specified features and local shape details. After the correspondence between the input model and the 2D illustration is established, shape stylization is formulated as a style-constrained differential mesh editing problem. A distinguishing feature of our approach is that it allows users to directly transfer styles from hand-drawn 2D illustrations with individual perception and cognition, which are difficult to identify and create with 3D modeling and editing approaches. We present a sequence of challenging examples including unrealistic and exaggerated paintings to illustrate the effectiveness of our approach.

Key words: Shape depiction, Illustrative visualization, Shape deformation
doi:10.1631/jzus.C0910089 **Document code:** A **CLC number:** TP391.41

1 Introduction

Shape style is a distinctive manner, form or type of shape that can be identified, conducted and popularized. The development of computational methods for depicting shape with visual symbols that can be processed by both human and computer has become one of the main tasks in computer graphics. For traditional photorealistic graphics, shape depiction means faithfully reproducing the realism of shape and shape styles. In contrast, non-photorealistic rendering seeks to approximate shape in a perception-driven context and may lead to artistic, exaggerated and even unrealistic results. In both cases, shape stylization elicits challenging problems, i.e., how to define really meaningful shape styles and how to choose and design shape with appropriate styles.

There have been many studies on mesh deformation in the past decade (Sumner and Popovic, 2004; Yu *et al.*, 2004; Zhou *et al.*, 2005). However, traditional 3D shape design relies mainly on the creativity and capability of 3D modeling and editing approaches. To alleviate the inefficiency of 3D editing, sketch-based modeling and editing (Nealen *et al.*, 2005; Nealen *et al.*, 2007; Zimmermann *et al.*, 2007) have also been studied with the aim of developing a more intuitive user interface. These approachs are generally suitable for novices, but only practical for creating simple shapes. Besides, in terms of artistic or illustrative depiction, most abstract and cognitive information is non-physical such as aesthetic properties, for example. The traditional approach to 3D modeling may result in a close match to realism, while a 2D painting interface typically offers more meaningful simplification. It might be more intuitive to operate on a simpler representation (2D illustrations) and upscale to handle a complex formulation (3D models). This approach would be of great convenience to non-professional 3D modelers such as medical illustrators.

‡ Corresponding author

* Project (Nos. 60776799 and 60873123) supported by the National Natural Science Foundation of China

We propose to perform shape deformation with styles captured from 2D examples. The main reason that the 2D example and the projected shape are triangulated is that the fine details can be encoded appropriately and the appearance (texture) presented in the 2D example can be mapped to the 3D model. The key idea is to construct a planar mesh for the input 2D example and transfer its shape styles in the context of differential mesh editing. To account for the global, medium, local and perception-driven understanding of the shape, we employ four shape styles, namely, the object contour, context curves, user-specified features and local shape details. Intuitively, the proposed 2D-to-3D mode combines sketch-based modeling and style transfer techniques, facilitating shape design using information captured from 2D drawings. By extending this approach to a sequence of 2D illustrations, the stylization of shape and shape deformation can be conveniently achieved. Fig. 1 demonstrates two results obtained using our approach.

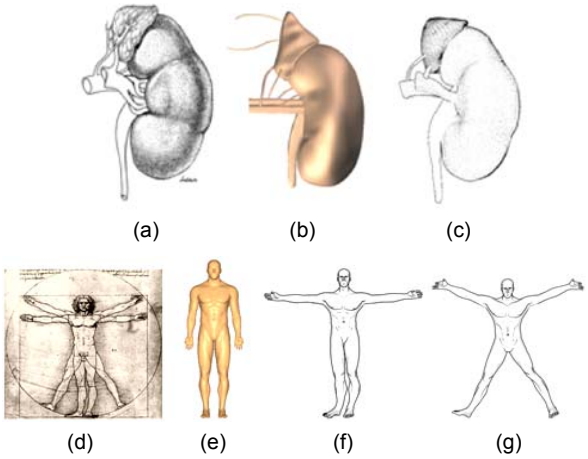

Fig. 1 Two examples using our approach
(a) A stippling illustration by Bill Andrews; (b) A 3D kidney model; (c) The deformed model using our approach; (d) A painting by Leonardo da Vinci that illustrates the golden proportions of the human body; (e) A 3D human body model; (f) and (g) Two results using our approach

2 Related studies

2.1 Example-based shape modeling

A large number of 3D models have been created by skilled artists or 3D scanning devices. It is therefore natural to leverage existing shapes by modifying them and allowing users to focus on the conceptual design. Shape interpolation (Alexa, 2003; Sheffer and Kraevoy, 2004) can be regarded as a simple form of example-based shape modeling. The method proposed by Funkhouser *et al.* (2004) seeks to composite the desired parts from different models to form new objects. Another widely used scheme is detail or deformation transfer (Biermann *et al.*, 2002; Sumner and Popovic, 2004; Yu *et al.*, 2004), which modifies the shape by replicating details in selected regions.

Rather than simply directly transferring or modulating existing shape details, it is also efficient to change the shape based on information from other sources, such as 2D drawings or motion data. For instance, Li *et al.* (2003) integrated user modification of 2D animation with 3D motion captured animation and created expressive 3D shape deformation. Zhou *et al.* (2005) presented a volume graphics Laplacian-based mesh deformation approach, which allows deformation of a 3D mesh by modifying selected curves of the mesh with curves from 2D drawings. Our approach advances this 2D driven deformation scheme from 2D curves to 2D illustrations with styles.

2.2 Sketch-based shape modeling

Most commercial modeling systems use 3D anchors to manipulate shapes and are not easy to learn. Sketch-based shape modeling (Igarashi *et al.*, 1999; Nealen *et al.*, 2005) offers an intuitive interface to allow users to easily determine the region of interest (ROI) and handles. Because human vision recognizes objects mainly by feature lines like silhouettes and contours, previous work generally employed a sequence of reference strokes on the 2D plane to indicate the design intentions (Nealen *et al.*, 2005). More intuitive design is made possible by directly specifying feature lines on 3D models (Ju *et al.*, 2007; Nealen *et al.*, 2007; Rose *et al.*, 2007; Zimmermann *et al.*, 2007).

Note that sketch-based shape modeling is generally suitable for novices, but is practical for creating only simple shapes. To enhance the creativity and imagination of users, an example-based modeling strategy can be efficiently incorporated by referencing 2D images within the sketching interfaces. For instance, Hornung *et al.* (2007) presented an interactive method to animate photos of 2D characters using 3D motion capture data.

2.3 Detail surface representations

Representing surfaces in local coordinate systems has proven to be useful for various shape processing applications. Standard multi-resolution representation (Kobbelt *et al.*, 1998; Guskov *et al.*, 1999; Biermann *et al.*, 2002) decomposes one surface into multiple levels with different details. For each level, the difference from the original mesh is encoded by local frame displacements in the vertices.

Another approach to surface details is differential-based presentation, first used by Alexa (2003) for shape interpolation. Thereafter, researchers have been investigating surface representations based on the Laplacian operator and discrete forms (Sorkine *et al.*, 2004; Lipman *et al.*, 2005; Nealen *et al.*, 2005; Hu *et al.*, 2007). Poisson-based discrete differential operators have also been widely used to perform detail-preserving mesh editing (Yu *et al.*, 2004). To make the representation invariant under rigid transformations, Sheffer and Kraevoy (2004) introduced pyramid coordinates, which are dependent on a set of angles and lengths relating a vertex to its immediate neighbors. Our approach is built upon differential-based mesh representation and manipulation because it bridges the gap between 3D models and 2D drawings, and favors global coupling of shape details.

3 Key idea

In general, representing shape styles by considering a single aspect, e.g., the local geometric details or the global shape contour, would lead to a loss in comprehension of the complexity embedded in the object. In our approach, we seek to comprehend four kinds of shape descriptions in the context of triangular mesh:

1. The object contour is a curve that represents the outline of an object. For a 3D model, the 2D contour of its projection under certain viewpoint constructs a silhouette. We employ the object contour as a global shape style to approximately estimate the proportion, scale and orientation of an object.

2. Context curves are designed to express the approximated understanding of the user to an object. One important curve is a 2D tree-like curve that outlines the approximated skeleton of the 2D example. Incorporating context curves into shape stylization has the advantage that they describe the object structure in a shape context manner and facilitate large-scale shape deformation. Thus, we refer to context curves as a medium shape style.

3. Local geometric details reflect subtle shape features and encode local shape styles such as the coil of a cockle shell. We make use of Laplacian coordinates and local affine frames to tie up local geometric details into an implicit linear system and manipulate them within a global minimization framework.

4. Perception-driven features are perceptively meaningful feature points or lines that are difficult to compute automatically. We allow the user to manually specify these perception-driven styles by drawing points or strokes on both the source model and examples. Note that these features describe more detailed information than the context curves.

We employ mesh editing to fulfill the shape stylization driven by a reference 2D illustration because the triangular mesh representation can be used to encode fine details and perform texture mapping. Our approach exploits the user's intentions in shape stylization by manually determining the curves, features and their correspondences. One main feature distinguishing our approach from others (Hornung *et al.*, 2007) is that we represent the 2D illustration with a texture-mapped mesh and regard the difference of shape styles between two meshes as a system of contour, context curves, features and intrinsic differential coordinates. These styles and their transfer from 2D to 3D can be integrated into a uniform differential based mesh manipulation framework. In this way, the distortion caused by the projection from 3D to 2D is greatly reduced. In addition, Hornung *et al.* (2007) used 3D motion captured data while our approach seeks to stylize 3D models using 2D illustrations.

Beginning with an input 3D mesh and a 2D illustration, we project the model onto the 2D plane and construct silhouettes of the projected 2D shape and the 2D illustration. The user manually specifies the context curves, the features and their correspondences. Subsequently, the shape of the 2D illustration is transferred to the projected 2D shape with a template fitting algorithm (Allen *et al.*, 2003). The final 3D shape is reconstructed under the dual Laplacian mesh editing framework. Fig. 2 depicts the entire pipeline using the shape of a head as an example.

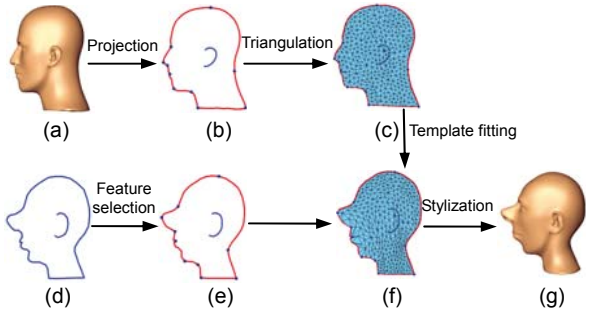

Fig. 2 Algorithm pipeline of our approach

The input 3D model (a) is first projected onto the 2D plane, yielding an outline (b). Based on the extracted silhouette in red, the context curves (shown in blue) and user-specified features (shown in blue points), a planar 2D mesh (c) is constructed. Likewise, the user constructs a 2D shape (e) from the input 2D illustration (d). The projected 2D mesh (c) is deformed by fitting the shape features presented in (e), yielding (f). The stylized shape (g) is obtained by driving the deformation of (a) with the constraints from (f)

4 Algorithms

We denote an arbitrary non-degenerate 2-manifold triangular mesh by $\boldsymbol{S}=(\boldsymbol{K}, \boldsymbol{V})$, where $\boldsymbol{V}=\{\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{n-1}\}$ is the set of vertex coordinates with n elements, and $\boldsymbol{K}$ describes its vertex connectivity. For the purpose of clarity, we list all vertices as an $n\times3$ matrix: $[\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{n-1}]^{\mathrm{T}}$. Throughout this paper, we use the same symbol (e.g., $\boldsymbol{V}$) to denote a vertex set or its matrix form.

4.1 Style generation

Given a 2D illustration I and an input 3D mesh S, the first task is to link them by projecting S onto the 2D plane on which I locates (Sander *et al.*, 2000). We allow the user to determine appropriate viewpoint and projection parameters, and render S into the frame buffer, resulting in a 2D drawing A. We denote the viewing matrix that projects a 3D vertex $\boldsymbol{v}$ of S onto the 2D plane as $\boldsymbol{H}$. The visible parts of S can be regarded as a triangular mesh and are denoted as S^{v}. S^{v} may consist of one or multiple parts, depending on the occlusion caused by the projection. We will discuss the occlusion issue in Section 5.1.

We then slightly enlarge A in the 2D frame buffer in four directions (namely, leftwards, rightwards, upwards, and downwards) with a pixel unit. The silhouette is obtained by subtracting A from the resultant image. The entire procedure can be accomplished with the standard stencil buffer feature of OpenGL (McReynolds *et al.*, 1999). Thereafter, we build a polyline to approximate the silhouette. Correspondingly, the outer contour of I is manually constructed and represented as one polyline. Fig. 3b depicts the silhouette constructed from A, which is the projection of the model shown in Fig. 3a.

We allow the user to manually draw the context curves of I and A. The curves of I are then manually made compatible with those of A by adjusting their vertices. Fig. 3c shows the curves built from Fig. 3b. In the meantime, the user is allowed to interactively draw a sequence of feature points and lines in both A and I and determine their correspondences. By setting the contours, the context curves, and user-specified features as constraints, we generate a triangular mesh S^A from A with a constrained conforming Delaunay triangulation algorithm (Shewchuk, 2002). All triangles of S^A are constrained Delaunay and all constraining points and lines are fairly preserved as vertices or lines. Besides, a minimum on the triangle inner angles and a maximum on the triangle area are ensured. With the correspondence of the feature points specified in both I and A, S^A can be gradually deformed into the shape of I with the template fitting algorithm (Allen *et al.*, 2003). In this way, a one-to-one correspondence between the shape styles of I and S^A is obtained.

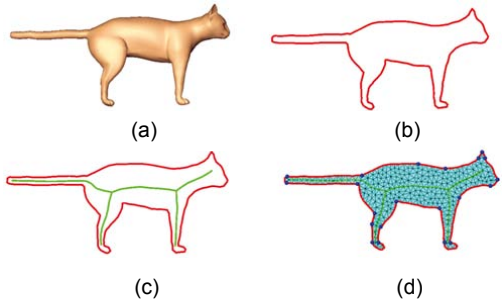

Fig. 3 Illustration of different kinds of shape descriptions

(a) The input 3D model; (b) The extracted silhouette in red; (c) The constructed context curves in green; (d) The constructed mesh with user-specified features in blue

4.2 Shape stylization

We project each vertex of the silhouette, the context curves and user-specified features in S^A into

the 3D space with the inverse of the viewing matrix $\boldsymbol{H}^{-1}$, and find the matched vertices in S^{v}. If there are multiple candidates for one vertex, we employ the approximate nearest neighbor (ANN) algorithm (Mount and Arya, 2006) to compute the nearest one. In this way, there is an injective mapping between shape styles of S^A and S^{v} on a per-vertex basis. Note that it is not necessary for S^{v} to be compatible with S^A. To ensure the establishment of the correspondence, we assume that the vertex distribution of the input model S is adequately dense and uniform. Since I and S^A have a one-to-one correspondence, correspondence among I, S^A and S^{v} is obtained.

Transferring the shape styles from I to S involves two considerations. First, the styles of I are represented in 2D space while the target model S is a 3D mesh. There is not a straightforward way to perform the style transfer. Our solution is to formulate the stylization in terms of the 2D projections of the vertices of S. This scheme effectively applies an optimization function to the vertex positions of S. Second, we expect the overall distortion of local shape details of S to be as minimal as possible under given stylization objectives. This also leads to a constraint on the vertex positions of S. We formulate both expectations on the vertex set $\boldsymbol{V}$ of S as an energy optimization framework with a set of soft constraints:

$$\min(w_{\mathrm{c}}E_{\mathrm{c}} + w_{\mathrm{z}}E_{\mathrm{z}} + w_{\mathrm{t}}E_{\mathrm{t}} + w_{\mathrm{q}}E_{\mathrm{q}} + w_{\mathrm{b}}E_{\mathrm{b}}), \qquad (1)$$

where E_{c}, E_{z}, E_{t}, and E_{q} express the style differences for the silhouette, the context curves, 2D local geometric details, and user-specified features respectively, and E_{b} denotes the 3D local geometric details. w_{c}, w_{z}, w_{t}, w_{q} and w_{b} are their respective associated weights that are used to balance their influences.

The silhouette item E_{c} measures the similarity between the contours $\boldsymbol{C}^A$ and $\boldsymbol{C}^I$ of S^A and I. Rather than representing a contour by a list of vertices, we employ the curve Laplacian coordinates (Weng *et al.*, 2006) to encode a curve C, so that the details of the contours can be well preserved during deformation. Specifically, the Laplacian coordinates $\boldsymbol{L}_{\mathrm{c}}(\boldsymbol{v}_i)$ of each vertex $\boldsymbol{v}_i$ are defined as the difference between $\boldsymbol{v}_i$ and the average of its neighboring vertices on the curve:

$$\boldsymbol{L}_{\mathrm{c}}(\boldsymbol{v}_i) = \boldsymbol{v}_i - (\boldsymbol{v}_{i-1} + \boldsymbol{v}_{i+1})/2. \qquad (2)$$

We denote the 2D Laplacian curve operator that is applied to the vertex set as a matrix $\boldsymbol{L}_{\mathrm{c}}$ (Weng *et al.*, 2006). Based on the fact that $\boldsymbol{C}^A$ and $\boldsymbol{C}^I$ are compatible and their vertices are also the vertices of S^A and I respectively, the transfer of the contour style from I to S^A can be fulfilled by transferring their Laplacian coordinates:

$$E_{\mathrm{c}} = \| \boldsymbol{L}_{\mathrm{c}}\boldsymbol{C}^A - \boldsymbol{L}_{\mathrm{c}}\boldsymbol{C}^I \|^2. \qquad (3)$$

For each 2D vertex $\boldsymbol{v}_j^A \in \boldsymbol{C}^A \subseteq S^A$, we can compute its corresponding 3D vertex in S^{v} with the inverse of the viewing matrix $\boldsymbol{H}$:$\boldsymbol{H}^{-1}\boldsymbol{v}_j^A$. Because there is an injective mapping for each vertex of the shape styles between S^A and S^{v}, we can denote the set of the vertices of S whose projections are on $\boldsymbol{C}^A$ as $\tilde{\boldsymbol{C}}^A$. Thus, we obtain an equation that relates to parts of the vertices of S:

$$E_{\mathrm{c}} = \| \boldsymbol{L}_{\mathrm{c}}\boldsymbol{H}\tilde{\boldsymbol{C}}^A - \boldsymbol{L}_{\mathrm{c}}\boldsymbol{C}^I \|^2. \qquad (4)$$

The context curve item E_{z} accounts for the shape modification caused by the analogy of the context curves from I to S^A. Basically, there are two types of curves. The first one consists of an ordered list of vertices and can be represented by the Laplacian coordinates described above. The second has a formulation of a 2D tree and is used to approximate the skeletons of the 2D drawings. We form the latter as a graph G=(P, E), where P is the 2D vertex set and E denotes the edge set. Similar to the contour item, we encode a 2D tree-like graph with two intrinsic items, namely, a length ratio r_{ijk}=$\|\boldsymbol{e}_{jk}\|/\|\boldsymbol{e}_{ij}\|$ and a directed angle θ_{ijk}=arccos[$\boldsymbol{e}_{ij}\cdot\boldsymbol{e}_{jk}/(\|\boldsymbol{e}_{ij}\|\cdot\|\boldsymbol{e}_{jk}\|)$]. It is not difficult to prove that this intrinsic representation is invariant under rotation and scale transformations. The relationships among two intrinsic items and the vertex positions can be written as

$$\begin{bmatrix} r_{ijk}\cos\theta_{ijk} & r_{ijk}\sin\theta_{ijk} \\ -r_{ijk}\sin\theta_{ijk} & r_{ijk}\cos\theta_{ijk} \\ -r_{ijk}\cos\theta_{ijk}-1 & -r_{ijk}\sin\theta_{ijk} \\ r_{ijk}\sin\theta_{ijk} & -r_{ijk}\cos\theta_{ijk}-1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_i \\ y_i \\ x_j \\ y_j \\ x_k \\ y_k \end{bmatrix} = \boldsymbol{0}. \qquad (5)$$

Applying Eq. (5) to all vertices of the context curves yields a large sparse linear system as $\boldsymbol{M}_s(\Psi)\boldsymbol{P}=0$, where $\boldsymbol{M}_s$ is the matrix determined by the set of scale-invariant variables $\Psi=\{\{r_{ijk}\},\{\theta_{ijk}\}\}$ of G.

Similar to the formulation of the contour item, we formulate the transfer of the context curve style as

$$E_z=\|\boldsymbol{M}_s(\Psi^I)\boldsymbol{H}\tilde{\boldsymbol{P}}^A\|^2, \tag{6}$$

where Ψ^I denotes the scale-invariant representation of G^I, and $\tilde{\boldsymbol{P}}^A$ is the set of the vertices of S whose projections belong to $\boldsymbol{P}^A$.

The 2D local details item E_t is designed to transfer the local 2D geometric details from I to S^A. Similar to the contour item, we denote the 2D mesh Laplacian operator (Sorkine *et al.*, 2004) that is applied to the vertex set as a matrix $\boldsymbol{L}_m$, and formulate the transfer of 2D geometric details as

$$E_t=\|\boldsymbol{L}_m\boldsymbol{H}\tilde{\boldsymbol{V}}^A-\boldsymbol{L}_m\boldsymbol{V}^I\|^2, \tag{7}$$

where $\tilde{\boldsymbol{V}}^A$ denotes the set of the vertices of S whose projections are in $\boldsymbol{V}^A$.

The perception-driven item E_q expresses user cognition in shape stylization. The user-specified feature points and lines are represented by a set of point pairs between S^A and I, denoted as $\boldsymbol{D}^A$ and $\boldsymbol{D}^I$ respectively. We let this item simply be the distance between two point sets:

$$E_q=\|\boldsymbol{H}\tilde{\boldsymbol{D}}^A-\boldsymbol{D}^I\|^2, \tag{8}$$

where $\tilde{\boldsymbol{D}}^A$ denotes the set of the vertices of S whose projections are in $\boldsymbol{D}^A$.

The 3D local details item E_b aims to preserve the 3D local details of S during stylization. Similar to the formulation of the 2D local details item, we denote the 3D Laplacian operator that is applied to the vertex set as a matrix $\boldsymbol{L}$, yielding

$$E_b=\|\boldsymbol{L}\tilde{\boldsymbol{V}}-\boldsymbol{L}\boldsymbol{V}\|^2. \tag{9}$$

4.3 Iterative optimization

Basically, the above five items ensure that the modified S not only mimics the shape styles from the 2D illustration, but also preserves its local details in a global variational framework. Note that, $\tilde{\boldsymbol{V}}$ in Eq. (9) denotes the unknown vertex set of the stylized S, and $\tilde{\boldsymbol{C}}^A$, $\tilde{\boldsymbol{P}}^A$, $\tilde{\boldsymbol{D}}^A$, $\tilde{\boldsymbol{V}}^A$ are subsets of $\tilde{\boldsymbol{V}}$. By taking Eqs. (4), (6), (7), (8) and (9) into Eq. (1), we can easily construct a large sparse linear system which can be solved in a least-squares sense.

Eq. (9) expresses a formulation of standard Laplacian mesh editing (Sorkine *et al.*, 2004). Therefore, we can accomplish the energy optimization of Eq. (1) in the context of differential mesh editing. To minimize the distortion of local geometry and parameterization, we employ dual Laplacian editing (Hu *et al.*, 2007) that fulfills the optimization in the dual domain by iteratively updating the vertex positions and the Laplacian coordinates. The other four items for the contour, the context curves, 2D local details and perception-driven styles are considered as linear constraints in the dual Laplacian system. Iterations are performed in two phases to account for the influences of different shape styles. In the first phase, w_c, w_z, w_t, w_q and w_b are set as 1.0, 1.0, 0.2, 0.2 and 0.5, respectively, to ensure efficient transfer of global styles, namely, the silhouette and the context curves. Then in the second phase, local details and user-specified features dominate the iteration by decreasing w_c and w_z gradually to 0.1 and increasing w_t and w_q gradually to 1.0 in each iteration.

5 Implementation details

5.1 Occlusion handling

In many cases, S^N is composed of multiple independent regions, as shown in Fig. 4a. It is apparent that the regions outlined with different lassoes cannot be regarded as an entire patch. Our solution is to first identify each region manually and build their correspondences to the input 3D model individually. All constraints from these regions are then combined to form a global optimization problem which is described in Section 4. Although this scheme is quite simple, it does produce satisfactory experimental results (Figs. 4a–4d).

5.2 Texture mapping

Optionally, the 2D illustration itself can be used to decorate the appearance of the input model by mapping itself to the visible part of the input model.

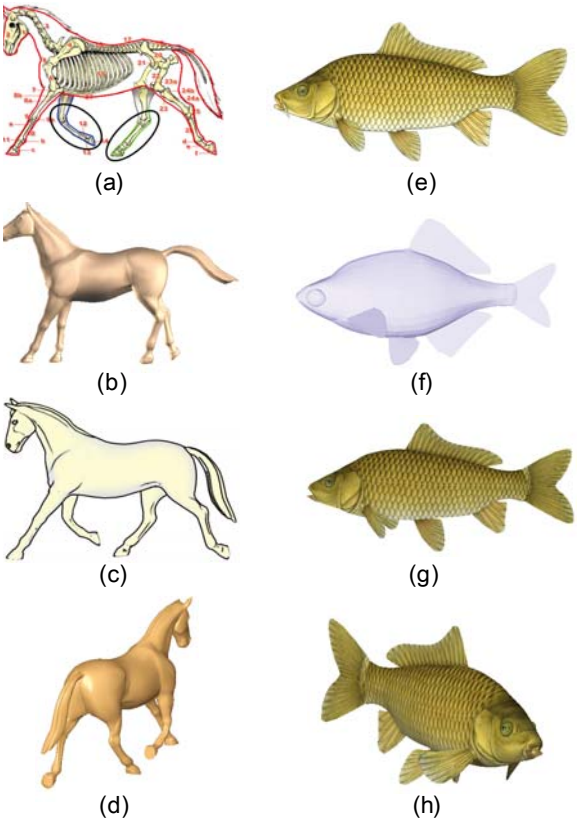

Fig. 4 Our results for horse (a–d) and fish (e–h) models
(a), (b), (e), and (f) show the input 2D examples and models. The stylized results are shown in (c) and (g). The effects from another viewpoint are shown in (d) and (h)

Each vertex in S^v has its projection counterpart in I, whose normalized coordinates in the 2D drawing can be used as the texture coordinates. Note that only parts visible under current viewing parameters can be decorated. For the invisible parts, we can exploit the model symmetry or apply texture synthesis to interactively assign textures, e.g., the half side of the fish shown in Fig. 4h.

6 Experimental results

We implemented the proposed approach on a PC with an Intel Core2 T5500 1.66 GHz CPU, 1.0 GB RAM and ATI X1600 video card. We rendered most stylized models with a point stippling algorithm or a line illustration system. Table 1 lists the stylization configuration and the user time for selected examples. For each experiment, styles for the object contour and local details were used. The user-specified feature points were classified into two categories, points on the object contour (boundary points) and points in the object interior (interior points). Solving the linear sparse system using iterative dual Laplacian algorithm typically required several seconds. To evaluate the efficiency of our approach, we asked three volunteers to test the entire procedure with our system. The first volunteer was a skilled 3D modeler and the other two volunteers had little experience in 3D modeling and little knowledge of our approach. The average user time was about 8 min for each example.

Fig. 1 demonstrates two stylization results from hand-drawn illustrations. It is apparent that not only were the global shapes simulated, but also local shape features such as the cockle in the kidney and the flexure of the calf muscles, were captured in the stylized models. Six other results are shown in Fig. 5. The 2D examples of the pony, teapot, hand, foot and fish were hand-drawn illustrations, and the 2D cat example was a 2D image.

Table 1 The configuration and time consumption for nine examples using our system

Data	Triangle	Vertices	Boundary points	Interior points	Context curves*	Solving	User time (s)		
							1st	2nd	3rd
Kidney	43 724	21881	25	0	3(0)	20	720	960	1080
Body	53 378	26691	25	3	1(1)	26	600	720	660
Head	23 402	11703	10	4	1(0)	12	180	240	180
Horse	16 589	8431	27	3	3(1)	8	600	960	840
Fish	7904	4002	17	4	1(0)	3	360	540	480
Lion	9996	5000	24	0	3(1)	4	300	360	360
Pony	45 338	22671	16	4	1(0)	22	480	720	660
Teapot	39 800	20101	12	0	3(0)	18	480	600	600
Hand	7292	3648	25	9	0(0)	3	180	180	180

* Numbers in parentheses denote the numbers of context curves that outline the approximated skeletons. The last three columns show the user time (s) including time spent on style specification, mesh construction, system solving and user interactions, for each of the three volunteers

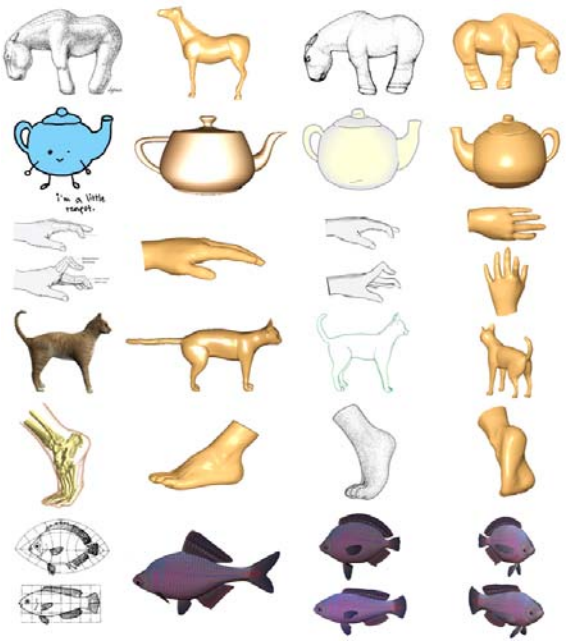

Fig. 5 Results for the pony, teapot, hand, cat, foot and fish models (from top to bottom)
The first and second columns show the input 2D examples and models, respectively. The stylized results are shown in the third column. The fourth column shows the effects from different view points

7 Conclusion and future work

Shape stylization plays an important role in the abstraction of visual information. This paper suggests a new opportunity for rapid and intuitive specification of 3D shape styles by decomposing the differences between 3D models and 2D examples into four components. This allows us to simultaneously provide the flexibility of imposing large global changes, the possibility of adding user cognition, and the accuracy of small shape disparities. The proposed shape styles have been demonstrated to be effective for clarifying meaningful structure and evoking the perception of the object complexity. We believe that our approach will enrich the creativity of computer-generated illustration.

As future work is concerned, we would like to extend our approach to stylize the deformation of a 3D model based on drawings that depict a dynamic sequence. Other aims include incorporating the influences of texture patterns, illumination, and color of the 2D example illustrations. We are also interested in applying shape stylization to volume illustration.

References

Alexa, M., 2003. Differential coordinates for local mesh morphing and deformation. *Vis. Comput.*, **19**(2):105-114.

Allen, B., Curless, B., Popovic, Z., 2003. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Trans. Graph.*, **22**(3):587-594. [doi:10.1145/882262.882311]

Biermann, H., Martin, I., Bernardini, F., Zorin, D., 2002. Cut-and-Paste Editing of Multiresolution Surfaces. Proc. 29th Annual Conf. on Computer Graphics and Interactive Techniques, p.312-321. [doi:10.1145/566570.566583]

Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D., 2004. Modeling by example. *ACM Trans. Graph.*, **23**(3):652-663. [doi:10.1145/1015706.1015775]

Guskov, I., Sweldens, W., Schroder, P., 1999. Multiresolution Signal Processing for Meshes. Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques, p.325-334. [doi:10.1145/311535.311577]

Hornung, A., Dekkers, E., Kobbelt, L., 2007. Character animation from 2D pictures and 3D motion data. *ACM Trans. Graph.*, **26**(1): Article 1, p.1-9.

Hu, J., Liu, L., Wang, G., 2007. Dual Laplacian morphing for triangular meshes. *Comput. Anim. Virtual Worlds*, **18**(4-5): 271-277. [doi:10.1002/cav.182]

Igarashi, T., Matsuoka, S., Tanaka, H., 1999. Teddy: a Sketching Interface for 3D Freeform Design. Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques, p.409-416. [doi:10.1145/311535.311602]

Ju, T., Zhou, Q.Y., Hu, S.M., 2007. Editing the topology of 3D models by sketching. *ACM Trans. Graph.*, **26**(3): Article 42, p.1-9.

Kobbelt, L., Campagna, S., Vorsatz, J., Seidel, H.P., 1998. Interactive Multi-Resolution Modeling on Arbitrary Meshes. Proc. 25th Annual Conf. on Computer Graphics and Interactive Techniques, p.105-114. [doi:10.1145/280814.280831]

Li, Y., Gleicher, M., Xu, Y.Q., Shum, H.Y., 2003. Stylizing Motion with Drawings. Proc. of ACM SIGGRAPH/Eurographics Symp. on Computer Animation, p.309-319.

Lipman, Y., Sorkine, O., Levin, D., Cohen-Or, D., 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph.*, **24**(3):479-487. [doi:10.1145/1073204.1073217]

McReynolds, T., Blythe, D., Grantham, B., Nelson, S., 1999. Advanced Graphics Programming Techniques Using OpenGL. SIGGRAPH Course Notes. Available from http://www.opengl.org/resources/code/samples/sig99/advanced99/notes/notes.html [Accessed on Mar. 20, 2008]

Mount, D., Arya, S,, 2006. ANN: A Library for approximate nearest neighbor searching, Version 1.1.1. Available from http://www.cs.umd.edu/~mount/ANN/ [Accessed on June 5, 2008].

Nealen, A., Sorkine, O., Alexa, M., Cohen-Or, D., 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Trans. Graph.*, **24**(3):1142-1147.

Nealen, A., Igarashi, T., Sorkine, O., Alexa, M., 2007. FiberMesh: designing freeform surfaces with 3D curves. *ACM*

Trans. Graph., **26**(3): Article 41, p.1-9.

Rose, K., Sheffer, A., Wither, J., Cani, M.P., Thibert, B., 2007. Developable Surfaces from Arbitrary Sketched Boundaries. Proc. 15th Eurographics Symp. on Geometry Processing, p.163-172.

Sander, P.V., Gu, X., Gortler, S.J., Hoppe, H., Snyder, J., 2000. Silhouette Clipping. Proc. 27th Annual Conf. on Computer Graphics and Interactive Techniques, p.327-334. [doi:10.1145/344779.344935]

Sheffer, A., Kraevoy, V., 2004. Pyramid Coordinates for Morphing and Deformation. Proc. 2nd Int. Symp. on 3D Data Processing, Visualization, and Transmission, p.68-75. [doi:10.1109/TDPVT.2004.1335149]

Shewchuk, J.R., 2002. Delaunay refinement algorithms for triangular mesh generation. *Comput. Geom.*, **22**(1-3): 21-74. [doi:10.1016/S0925-7721(01)00047-5]

Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rossl, C., Seidel, H.P., 2004. Laplacian Surface Editing. Proc. Eurographics/ACM SIGGRAPH Symp. on Geometry Processing, p.179-188.

Sumner, R.W., Popovic, J., 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, **23**(3):399-405. [doi:10.1145/1015706.1015736]

Weng, Y., Xu, W., Wu, Y., Zhou, K., Guo, B., 2006. 2D shape deformation using nonlinear least squares optimization. *Vis. Comput.*, **22**(9-11):653-660. [doi:10.1007/s00371-006-0054-y]

Yu, Y., Zhou, K., Xu, D., Shi, X., Bao, H., Guo, B., Shum, H.Y., 2004. Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graph.*, **23**(3):644-651.

Zhou, K., Huang, J., Snyder, J., Liu, X., Bao, H., Guo, B., Shum, H.Y., 2005. Large mesh deformation using the volumetric graph Laplacian. *ACM Trans. Graph.*, **24**(3): 496-503. [doi:10.1145/1073204.1073219]

Zimmermann, J., Nealen, A., Alexa, M., 2007. SilSketch: Automated Sketch-Based Editing of Surface Meshes. Proc. 4th Eurographics Workshop on Sketch-Based Interfaces and Modeling, p.496-503.