Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Shape Manipulation of Diffusion Curves Images

SHUFANG LU[1], XUEFENG DING[1], FEI GAO[1], JIAZHOU CHEN[1]
[1]College of Computer Science and Technology, Zhejiang University of Technology

Corresponding author: Shufang Lu (e-mail: sflu@zjut.edu.cn).

This work was supported by Natural Science Foundation of Zhejiang Province (No. LY19F020027), the Key Research and Development Program of Zhejiang Province (No. 2018C03055), and Natural Science Foundation of Zhejiang Province (No. LY18F020035).

ABSTRACT Diffusion curves image (DCI) is a kind of vector graphics with its geometric primitives defined by a set of Bézier curves. As one of the vector graphics, editability is an important property for diffusion curves. However, most of the current DCI shape manipulation methods are typically limited to single curve. It is tedious for users to edit the shape of DCI by deforming each curve individually, especially with the increasing number of curves. In addition, the image contents cannot well preserved because there is no constraints among curves. We present an efficient and flexible DCI manipulation method by combining global and local deformations. The overall shape is easily changed by our global manipulation tool with a few user interactions, while the local manipulation tool is used to modify details. These two tools can be combined to obtain desired effects. In both cases, the constraints during deformation preserve the quality of the editing results. Our experimental results demonstrate that the proposed method provides more convenient control for DCI shape editing than existing techniques based on diffusion curves.

INDEX TERMS Shape Manipulation, Vector Graphics, Diffusion Curves Images

I. INTRODUCTION

Vector graphics provides a representation for describing images by geometric primitives such as points, lines and curves. Compared with raster graphics which is defined via a grid of pixels, vector graphics is resolution-independent, more compact, and easy for editing. Due to these advantages, it is extensively used in graphical user interface, internet applications, three-dimensional modeling and so on.

As one of popular vector graphics primitives, diffusion curves [1] uses cubic Bézier curves and attached both side color control points to create high-quality vector images. It can be created from scratch by drawing tools or through tracing existing images. The rendering of a set of diffusion curves results in a diffusion curve image (DCI) with complex color gradients.

Editability is an important property of vector graphics, which is the reason why it remains popular among artists and designers. However, existing work mainly focus on the DCI creating process [2], [3], rendering acceleration [4], [5] and image diffusion quality improvement [6], with weak support for flexible shape manipulation. As a DCI consists of a sparse set of disconnected paths, most of current methods are limited to curve-level editing. They just deform a curve by moving its control points. Despite their capability of modifying the local details of a DCI, they still have some inevitable limitations. Obviously, it is a time-consuming process because users have to edit each curve individually. Also, without geometric constraints among curves, it may require modifications to a large amount of individual curves in order to obtain desired effects. Moreover, the semantic content represented by a set of curves in a DCI may be destroyed due to editing each curve separately and the editing effects are hard to control or predict.

To address these problems, we present an efficient shape manipulation tool for DCI by combining both global and local deformations. The global shape manipulation is treated as space deformation operation through linear blending affine transformations. We triangulate the closed domain of the DCI and adopt bounded biharmonic weights [7] as the weight function. Then, the global shape is deformed by setting a cage or a few single points as handles controlled by users. With only moving or dragging a few handles, a new shape of DCI can be easily generated followed by a curve refitting process. On the other hand, local details are modified by editing individual curve. To minimize user interaction, we use two simulation points to replace two curve control points that are not located on the curve. In both global and local deformation modes, we add constraints on the junctions between connected curves, which makes the editing results more reliable.

With our DCI shape manipulation method, the user can first edit the global shape roughly by global deformation and then modify the details precisely with local shape deformation. Although it has been proposed to add diffusion curves to SVG standards, there is still no standard representations and rendering methods for diffusion curves. We employ the original diffusion curves representation [1] and a GPU-based rendering system [4] to make our method concise and extensible. The color diffusion result is automatically changed with DCI shape editing because the color inforamtion of diffusion curves is attached with its geometric structure.

Compared with previous diffusion curves image editing methods, our method is more effective and flexible in controlling the shape of DCI. It significantly reduces the user interaction time, especially for the circumstances of creating different gestures of a same object(i.e. keyframes for animation). Furthermore, several constraints are used on the junctions between connected curves in both global and local schemes to guarantee high-quality editing results. To the best of our knowledge, ours is the first DCI shape manipulation technique combining both global and local deformation.

II. RELATED WORK

In this section, we focus on techniques targeting vector graphics, vector graphics editing and shape deformation, the topics that are most relevant to our work.

For a long time, vector graphics is defined by the primitives filled with solid color and cannot offer complex color gradients compared with raster graphics. More recently, new powerful vector graphics representations have been proposed. One of them is gradient mesh [8]–[10], which employs triangular or quad mesh as geometric structure and stores color values on the vertices of mesh. Smooth rendering results are achieved by interpolated colors across the mesh. Gradient mesh produces high-quality image, but is not natural for artists and designers because the relationship between patches of mesh and specific image contents is unclear. As a result, it is difficult for us to manually create vector images represented by gradient meshes and not convenient for subsequent editing. Another line of methods are based on diffusion curves [1]. Diffusion curves employs cubic Bézier curves as geometric primitives to describe the image boundaries, with color constraints attached on both sides of each curve. By solving a Poisson equation, the color control points on either side of a curve diffuse over the entire image canvas smoothly . It is intuitive for users because it supports traditional freehand drawing techniques, and edges/curves are viewed as natural primitives for encoding and editing images [11]. Subsequent works on diffusion curves mainly focus on the rendering acceleration [4], [5], [12], image quality improvement [3], [13], [14], color diffusion control [15]–[19], new forms of diffusion curves extension [6], [20], [21], and its applications [22], [23].

Easy for editing is one of the most important reasons that vector graphics remains popular . Vector primitives are widely used either in raster image editing and vector graphics image itself. Raster image editing has natural advantages in manipulating pixels but limits some operations such as object-based editing. Based on the observation that edges and contours are directly related to significant visual events in image content such as object boundaries, vector geometric primitives are commonly applied for raster image editing operations. Elder et al. [11] first introduced a method for image editing based on edges rather than pixels. Barrett et al. [24] proposed a real-time object-based image editing technique for manipulation and animation of static digital photographs. In their method, Objects is tessellated into a triangular mesh, allowing shape modification to be performed by changing the granularity of editing from the pixel to the object level. Fang et al. [25] applied shape deformation in digital image editing and proposed a detail preserving editing method via feature curves.

The editability of vector graphics is very essential for designers and artists who create visual content with user interaction. Shape editing is convenient for gradient mesh representation because it provides the connectivity for vector image contents. Space deformation techniques can be easily applied to mesh deformation. Liao et al. [9] proposed a subdivision-based representation for vector images by combining the triangular patches and curved boundaries. It performs shape editing for vector images by using as-rigid-as-possible method [26], but cannot guarantee the continuity between the adjacent curve segments. Different from gradient mesh, diffusion curves representation consists of sparse sets of disconnected curves to describe the geometry information. Therefore, it is easy to edit each curve individually in DCI, but hard to edit the semantic content which includes a large number of curves. Despite several existing diffusion curves methods spent some efforts in DCI editing, most of them are still limited to curve-level editing. To facilitate content creation for the artist, the pioneer diffusion curves method [1] provides the user with several standard editing tools including editing of single curve geometry and curve splitting. The individual diffusion curve editing operations in [6], [13] are intuitively be done by a click-and-drag metaphor. In their methods, the Bézier curve control point that is the closest to the clicked position is determined. Moreover, Pang et al. [5], Boyé et al. [16] and Prévost et al. [27] used the triangle mesh as a intermediate representation to assist the rendering process. But as Boyé et al. mentioned in their work [16], the triangle mesh updates lead to flickering artifacts during editing operations and is not suitable for real-time DCI editing. Sun et al. [12] presented a fast multipole representation for diffusion curves and points as well as an interactive editing tool of merging different DCIs together. Xie et al. [2] developed a hierarchical diffusion curves representation to make their results more editable, but it is limited to generating visual abstraction rather than editing the shape of DCI. Lu et al. [14] proposed a diffusion-curve-based image vectorization framework for RGBD images. It supports the editing of objects that are segmented by using both color and depth information. In this paper, we provide a more flexible and efficient DCI shape editing system in terms of global and local aspects.

Shape deformation is extensively used on visual design, animation and so on. Generalized barycentric coordinates, which defines coordinate of points with respect to polytopes, provides an efficient way to compute smooth space deformation. There are many different constructions of barycentric coordinates, and the well-known methods and their extensions include Mean Value Coordinates [28]–[30], Harmonic Coordinates [31]–[34], Green Coordinates [35], complex barycentric coordinates [36], [37], and Biharmonic Coordinates [7], [38]. The deformation equation is linear combination of real or complex-valued coordinates with some coefficients. Another way to perform space deformation is variational methods, which solve a variational problem by finite element discretization over a mesh [26], [39], [40] or by smooth basis functions [33], [36], [38]. In addition, some methods which are based on quasi-conformal maps and their applications have been developed recently [41]–[49]. With trade-off between computational consumption, visual effect and interactivity, Biharmonic Coordinates is used in our method.

III. PROPOSED METHOD

In this section, we will introduce the details of our DCI shape manipulation method. Our method provides two modes to manipulate the shape of DCI: global shape editing and local shape editing. Fig. 1 shows an overview of our DCI shape manipulation framework. In global shape editing mode, by setting and moving a few handle points, the global shape of DCI is controlled by the underlying linear space deformation method. We adopt bounded biharmonic weights as the weight functions and refit diffusion curves after deformation. In local shape editing mode, individual curve is deformed to modify details of DCI. By adding two simulation points, the Bézier curve control point that is the closest to the clicked position is selected. Several constraints are performed during the global and local deformation to produce high-quality results. Users can combine these two editing modes to adjust the editing result until achieving a desired shape.

A. GLOBAL SHAPE DEFORMATION

1) Linear Blending Deformation

The global shape editing is treated as the space deformation operation and we employ the linear method by blending affine transformations at arbitrary handles. Compared with some quadratic methods, such as as-rigid-as-possible shape manipulation [26], linear methods are usually more efficient due to the simplicity. We denote each point of the given shape by p and it is transformed to the new position $p^{'}$ by the following weighted combinations:

$$p^{'} = \sum_{i=1}^{m} \omega_i(p) T_i p \tag{1}$$

Given the handle points $H_i(i = 1, 2, 3, , m)$, T_i is the affine transformation for each handle points H_i and $\omega_i(p)$ is the weight function associated with H_i evaluated at point p. The weights ω_i is obtained as minimizers of the Laplacian energy with constraints on the closed domain Ω:

$$\underset{\omega_i, i=1,2,3,\ldots,m}{argmin} \sum_{i=1}^{m} \frac{1}{2} \int_{\Omega} ||\Delta \omega_i||^2 dV$$

Subject to:

$$\begin{aligned} &\omega_i(H_k) = \delta_{i,k} & \\ &\omega_i(F) \text{ is linear} & \forall F \in F_G \\ &\sum_{i=1}^{m} \omega_i(p) = 1 & \forall p \in \Omega \\ &0 \le \omega_i(p) \le 1, i = 1, \ldots, m & \forall p \in \Omega \end{aligned} \tag{2}$$

where the H_k is a handle point, and $\delta_{i,k}$ is Kronecker delta. The F_G denotes the set of all boundary of cage.

We use the bounded biharmonic weights [7] as the weight function because it is flexible and supports handles of arbitrary topology. In our method, we use cage or single points to control the global shape of DCI. Fig. 1 and Fig. 2 show the process of global shape editing with these two kinds of topology schemes respectively.

Minimizing the biharmonic energy amounts to solving the Euler-largrange equations, which are the biharmonic PDEs in this case: $\Delta^2 \omega_i = 0$. In order to solve the constrained varitional problem numerically with quadratic programming, we discretize it using linear finite elements.

2) Space discretization

Bounded Biharmonic Weight functions require space discretization for weight computation. As diffusion curves is an open structure, a domain enclosing all curves is required to compute the weight function values. For the cage topology, the cage itself is a closed structure(see Fig.1(b)). For single points, the image boundary is treated as the closed region(see Fig.2(c)). We use the triangulation technique [50] to discretize the closed space and the weight value of each vertex in the triangle mesh is obtained by solving Eq.(2).

We can change the path of cubic Bézier curves directly or via moving its control points. For the two different controlling ways, there are two different constraints during triangulation process.

The first one is to use the control points of Bézier curves as additional constraint points, and these control points will be the vertices of mesh after triangulation. Therefore, the path of Bézier curves is changed by mapping their control points to new positions. For each single curve, it is a directly way since the path of Bézier curve is controlled by its control points. However, it has some drawbacks for the global shape deformation. Let $C = [c_1 \quad c_2 \quad c_3 \quad c_4]$ represent the four

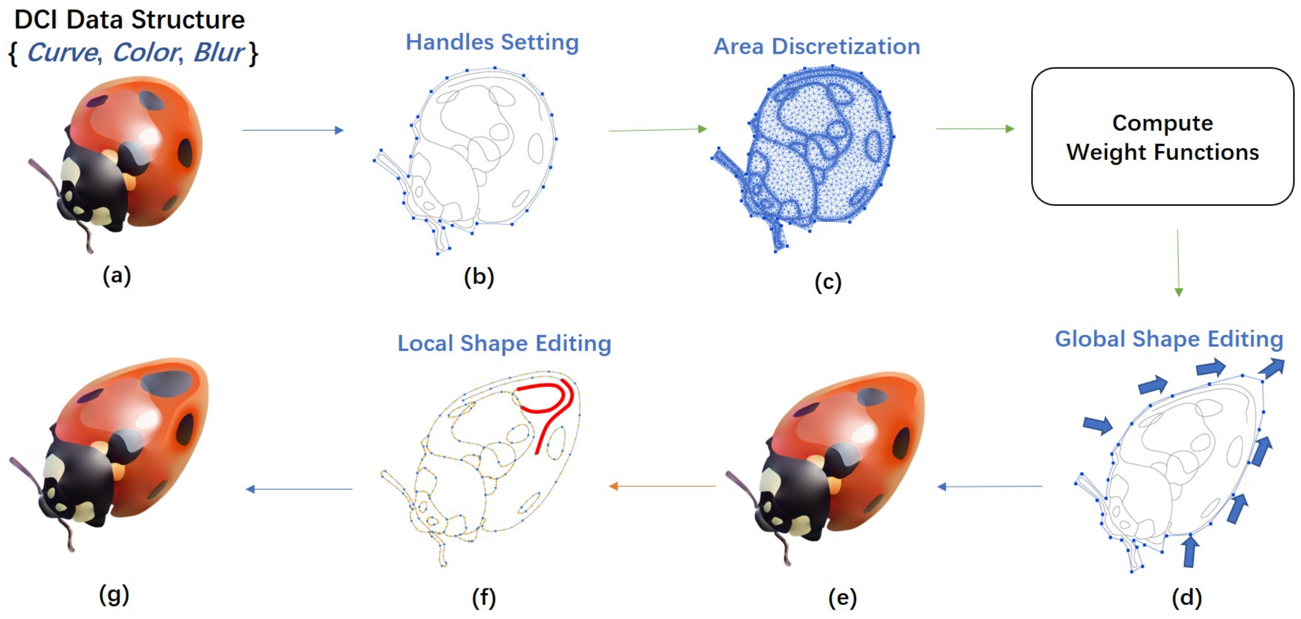

FIGURE 1: An overview of our DCI shape manipulation method.

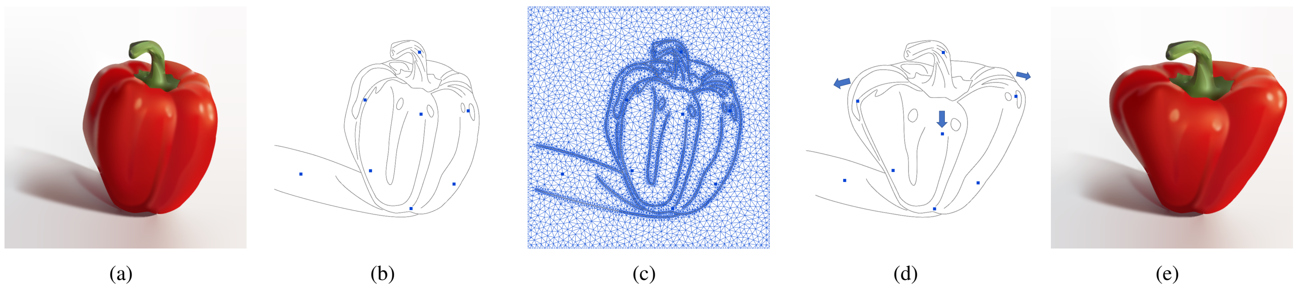

FIGURE 2: Global deformation result with single points. (a) is the input DCI, (b) shows blue handles points, (c) is the space discretization result, (d) is the deformation result with blue arrows indicate the moving direction of handle points, and (e) is the corresponding rendering result.

control points of a Bézier curve (see Fig. 4). c_1 and c_4 is the start point and the end point of the curve and they are located on the curve. But the other two control points c_2 and c_3 are usually not located on the curve, even far away from the curve. The bounded biharmonic weights is distance-dependent, so for each handle point its influence decreases with the increase of distance from it. As shown in Fig. 3, the yellow curve is defined by $A = [A_1 \quad A_2 \quad A_3 \quad A_4]$, and the green curve is defined by $B = [B_1 \quad B_2 \quad B_3 \quad B_4]$. If we use the control points as the triangulation constraints, the red handle point P has more effects on the yellow curve although green curve is closer to P , because the control point A_3 of the yellow path is closer to P than the control points of the green curve. As a result, it may generate unpredictable and undesirable results after deformation. As shown in Fig. 8(e), the continuity is destroyed and local textures are not well preserved after editing, which resulting in visible artifacts indicated by the green arrow.

The second option is to use the sampling points on paths of Bézier curves as additional constraints for triangulation. As we observe that the following refitting result is not smooth with a small number of sampling points, in our experiment, we sample 100 points of each curve as the additional constraints during the triangulation process. Fig. 1(c) and Fig. 2(c) show two area discretization results. In this way, the handle points directly influence the path of Bézier curve rather than the control points, and the problem shown in Fig. 3 can be avoided. With the same editing by dragging the same control point, the deformation result based on this triangulation scheme is improved as shown in Fig. 8(h).

3) Curves Refitting

For each individual curve, let $S(p)$ represent the undeformed shape with each sampling point p. With the arrows in Fig. 1(d) and Fig. 2(d) indicate the moving directions of the handles, all sampling points, that is the discrete representation of the curves are mapped to new positions. The deformed

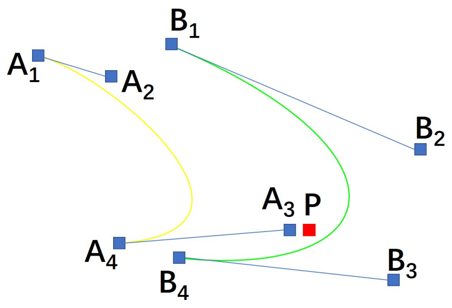

FIGURE 3: Illustration of using control points as triangulation constraints.

shape is:

$$S(p^{'}) = \sum_{i=1}^{m} \omega_i(S(p)T_iS(p) \tag{3}$$

Then we need to find a cubic Bézier curve that is the closest to the deformed polygonal path through refitting process.

The deformed shape will usually not be represented by Bézier curve exactly. For each individual curve, let $S(c)$ represent the undeformed shape with each control point c of Bézier curve. Therefore, our goal is to find the closest shape $S(c^{'})$ whose path best fit the deformed shape. This problem is treated as the energy optimization to obtain the unknown control points $c^{'}$ as follows:

$$E(S) = ||S(c^{'}) - S(p^{'})||^2. \tag{4}$$

$E(S)$ is the integrated distance between the fitting shape $S(c^{'})$ and deformed shape $S(p^{'})$.

In order to preserve the local details of DCI, it is very important to keep the smoothness at the junctions between connected curves. We employ C^0 and C^1 continuity constraints as the linear constraints in the energy minimization process. Let $D = [d_1 \quad d_2 \quad d_3 \quad d_4]$ represent the curve which shares an endpoint with curve C, and $c_4 = d_1$. For deformed curves $C^{'} = [c_1^{'} \quad c_2^{'} \quad c_3^{'} \quad c_4^{'}]$ and $D^{'} = [d_1^{'} \quad d_2^{'} \quad d_3^{'} \quad d_4^{'}]$, we set $c_4' = d_1'$ and $c_4' - c_3' = d_2' - c_1'$ as the linear constraints to enforce C^0 continuity and C^1 continuity, respectively. On the overall domain, in order to solve for all cubic Bézier curves simultaneously, we weight the energy for each curve. Therefore, the complete refitting energy minimization problem with all constraints is as follows:

$$\begin{aligned} &\underset{S}{argmin} \sum_{j=1}^{n} \eta_j E(S_j) \\ &\text{Subject to:} \quad c_4^{'} = d_1^{'} \\ &\qquad\qquad c_4^{'} - c_3^{'} = d_2^{'} - c_1^{'} \end{aligned} \tag{5}$$

where n is the number of curves in the DCI, S_j is the shape of j-th curve, and η_j is the length of S_j.

We adopt the numerical solver proposed by Liu et al. [51] to solve Eq.(5), in which the linear equality constraints are enforced via the Lagrange multiplier method. The different part is that we do not use the tangent magnitude ratios and angle preservation constraints for live manipulation, as we find that these constraints have a small and limited negative effect on refitting result but are time-consuming. Also, the negative effect can be further remedied by our local shape editing method (see Section III-B). Fig. 1(d) and 2(d) show the curve refitting results after global deformation, and Fig. 1(e) and 2(e) are the corresponding rendering results with deformed diffusion curves. After the refitting step, the deformed paths are represented by Bézier curves rather than discrete points.

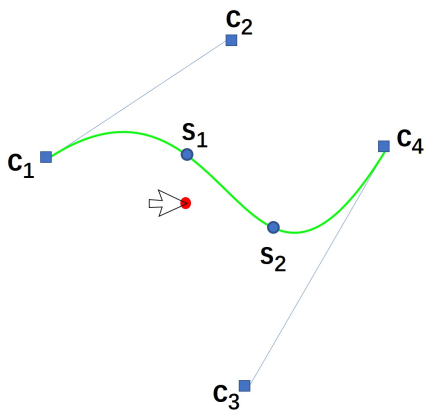

FIGURE 4: Illustration of our local shape editing method.

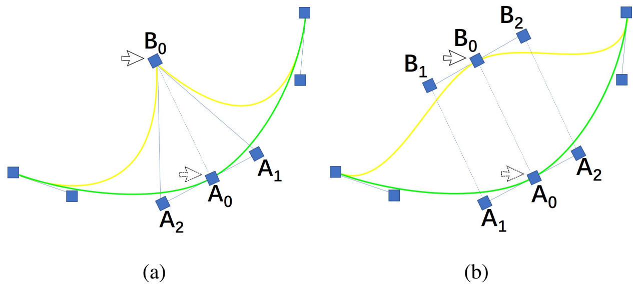

FIGURE 5: Comparison of local editing without and with angle preservation.

B. LOCAL SHAPE EDITING

The overall shape of a DCI can be easily changed by the global shape editing mode, as shown in Fig. 1 and 2. But it is hard to perform more precise editing operations, such as modifying local details. To make the shape editing tool more powerful, our method also supports a local shape editing tool.

Since the individual curve is independent from each other, it is intuitive to change its path by moving the control points. However, it is difficult for users to accurately select the control points especially where the area is densely covered by a large number of curves. Instead of moving the control points directly, we use two simulation points s_1 and s_2 to implicitly represent the two control points c_2 and c_3 that usually are not located on the curve as shown in Fig. 4. s_1 and s_2 are located on the $1/3$ and $2/3$ of the curve, respectively. When the user clicks on the DCI (i.e. the red point in Fig. 4), the simulation point s_1 that is closest to the clicking position is selected automatically. Then the displacement vector from clicking position to the new position is added to the corresponding control point c_2. Therefore, the curve that is closest to the clicking position is automatically chosen for deformation.

Similar to global shape editing, we also set constraints at the junctions between connected individual curves. For connected Bézier curves $C = [c_1 \quad c_2 \quad c_3 \quad c_4]$ and $D = [d_1 \quad d_2 \quad d_3 \quad d_4]$ with $c_4 = d_1$, the local shape editing tool enforces C^0 continuity as global deformation described in Section III-A3. In addition, we preserve the angle at junctions as well. Specifically, as dragging is a translation operation on the selected point, we apply same displacement vector to the control points c_3 and d_2 when dragging the junction point $c_4(d_1)$ to the new position $c'_4(d'_1)$. The angle constraint is expressed as follows:

$$\begin{aligned} c'_3 &= c_3 + (c'_4 - c_4) \\ d'_2 &= d_2 + (c'_4 - c_4) \end{aligned} \tag{6}$$

With junction point A_0, Fig. 5(a) is the result without angle preserving . When dragging point A_0 to B_0 while other control points are fixed, the shape at the junction B_0 has great change which is not desired. Fig. 5(b) shows the improved deformation result by preserving the angle at junction A_0 during dragging. The smoothness at junction B_0 is kept and local details are preserved better. Fig. 1(f) shows the local shape editing result by modifying the red highlighted curve. One more local editing result is shown in Fig. 6, in which the shapes of the front two legs of the chair are edited.

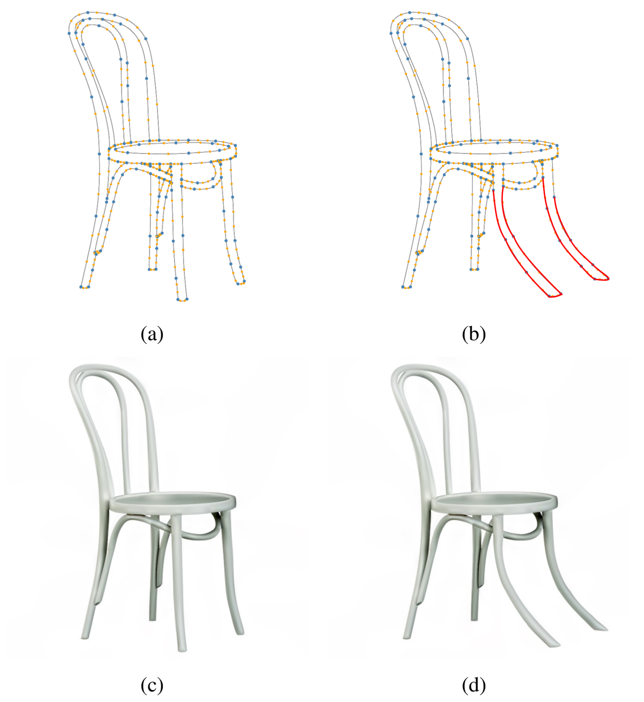

FIGURE 6: The local editing result. (a) is the input, (b) is the local shape editing result with modifying highlighted red curves, (c) and (d)are the rendering results before and after performing local shape editing.

IV. RESULTS AND DISCUSSIONS

More DCI global shape editing results are shown in Fig. 7. Cage is used as handles to deform the vector images of Lotus, Butterfly, Leaf and Cup, while single points are used as handles to control the images of Dolphin, Armchair, Frog and Panda. By moving the handle points, our editing tool easily modifies the global shape of DCI with a few user interactions. On the other hand, images details can be further modified by the provided local editing method. Users can alternately perform the two editing operations to achieve desired editing results. Compared to other DCI editing tools, which only offer single curve editing function, our DCI shape editing method is more efficient and convenient by significantly reducing the user interactions. Furthermore, with the constraints in both global and local editing modes, image features and details are better preserved after deformation.

Our editing system is implemented on a 3.3 GHz Intel Dual Core i5 CPU and an NVIDIA GTX960 GPU. The user interface is implemented on browser with JavaScript and diffusion curves rendering is performed on GPU by employing WebGL. The geometry processing library libigl [52] is used to compute bounded biharmonic weights. The cubic Bézier curves refitting is implemented in Python with Numpy package.

The statistics and performance of the global shape editing results in this paper are shown in Table 1. The space discretization and bounded biharmonic weights calculation are performed in the precomputation step, and it takes more precomputation time with the increasing number of handles and curves. Fortunately, it is just performed once and do not affect the efficiency of live deformation. In the experiments, we notice that the live shape manipulation is smooth when the number of curves is not large, probably less than 300. As shown in Table 1, after moving handles, the update of calculating new positions of all sample points on curves and refitting them back to diffusion curves representation is very fast. The FPS is the average frame number per second when continually performing the shape editing manipulation. Due to the modification of geometric information, the rendering system need to resample the points on curves and send these data to GPU solver to compute every pixel of DCI again. When frequently performing shape editing interaction on a large number of diffusion curves, the frame rate will drop down due to the rendering computation cost. As a result, the DCI rendering is the most time-consuming part, which is the bottleneck of our method.

Fig. 8 compares global deformation results with different triangulation constraints during the process of area discretization as mentioned in Section III-A2. Given an input of

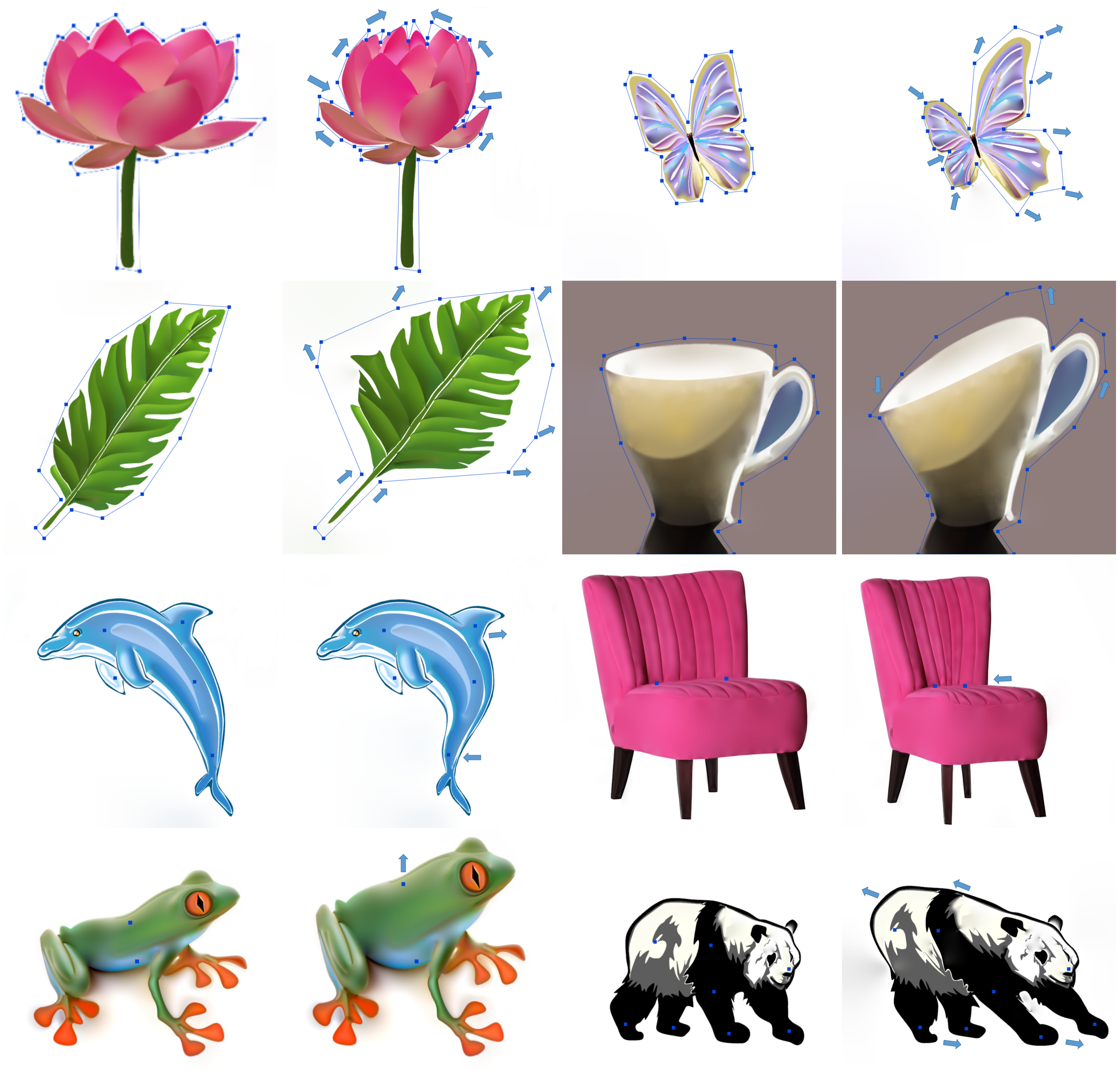

FIGURE 7: A variety of global shape editing results. For each example, left is the DCI before editing and right is the editing result after global deformation. The blue arrows indicate the moving directions of handle points.

TABLE 1: Statistics and performance of the global shape editing results shown in this paper. #Paths is the number of path that consists of a series of end-to-end diffusion curves, #Curves is the number of diffusion curves, #Handles is the number of handle points, Precomputation/h(s) is the bind time per handle in seconds.Updating(s) is the time including calculating the new positions of all the sample points and refitting them back to diffusion curves representation.

Dataset	**#Paths**	**#Curves**	**#Handles**	**Precomputation/h(s)**	**Updating(s)**	**Rendering(FPS)**
Poivron	38	109	8	1.66	3e-3	60
Lotus	22	113	40	1.64	1.6e-2	52
Butterfly	89	176	23	1.65	1.7e-2	55
Leaf	39	197	13	3.01	5e-3	49
Cup	99	132	18	1.72	1.9e-2	58
Dolphin	39	171	5	3.61	2.5e-3	53
Armchair	48	117	2	4.51	1.9e-3	60
Frog	32	79	2	3.63	1.2e-4	60
Panda	34	310	8	3.07	3.3e-3	33

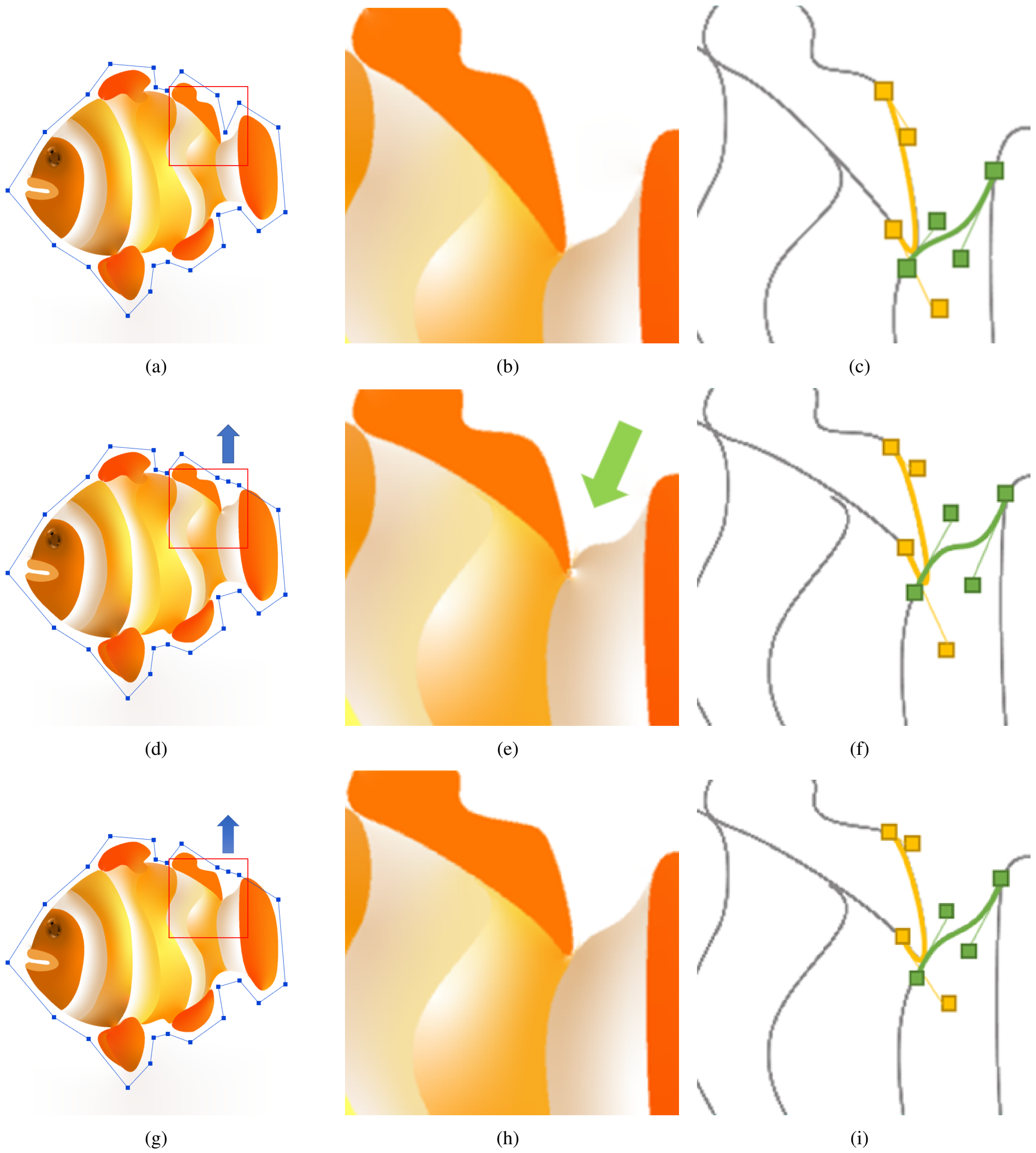

FIGURE 8: The global deformation results comparison between two area discretization schemes. The first row is the initial state, the second row is the deformation result by using control points as triangulation constraints, and the third row is the result by using sampling points of curves as triangulation constraints. Each row from left to right: a DCI, a close-up of the highlighted box region, and corresponding geometrical curves.

a DCI with its cage shown in Fig. 8(a), Fig. 8(b-c) are the zooming-in result of highlighted red box and its diffusion curves. Same deformation indicated by the blue arrow is applied on both triangulation results, and the output are shown in Fig. 8(d) and Fig. 8(g), followed by a close-up of the highlighted boxes and corresponding diffusion curves. When using the control points as the triangulation constraints, the spatial distance between handle points and control points is not consistent with the distance between handle points and curve paths, which resulting in visible artifacts indicated by green arrows in Fig. 8(e) Instead, we directly sample points on curves and use them as additional constraints for triangulation with the improved results shown in Fig. 8(h).

V. CONCLUSIONS

In this paper, we have presented a shape editing system for diffusion curves images. The proposed shape editing method is efficient and convenient by providing both global and local editing modes. A new shape can be easily created by only a few user interactions, and the local details are well preserved by setting constraints on the junctions between connected curves. Moreover, the deformed result is always represented by the data structure of diffusion curves, so it is easy to integrate our editing tool into other DCI editing and rendering systems.

In the future, there are still some aspects that could be improved. First, real-time user feedback is essential for live shape manipulation. With the increasing number of handles and curves, the frame rate drops down as the DCI rendering and bounded biharmonic weights calculation take most of the time. We will develop a fast DCI shape editing system by accelerating rendering process while reducing weights computation time. Secondly, our tool is applied on the single-layer DCI, and all the curves in the region will be deformed and refitted during global deformation. If there are unclear boundaries between the object we want to edit and the background, to select the region of interest is a difficult task. Thus, multi-layer representation of DCI can be developed to avoid this problem.

REFERENCES

[1] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin, "Diffusion curves: a vector representation for smooth-shaded images," ACM Transactions on Graphics (TOG), vol. 27, no. 3, p. 92, 2008.

[2] G. Xie, X. Sun, X. Tong, and D. Nowrouzezahrai, "Hierarchical diffusion curves for accurate automatic image vectorization," ACM Transactions on Graphics (TOG), vol. 33, no. 6, p. 230, 2014.

[3] S. Zhao, F. Durand, and C. Zheng, "Inverse diffusion curves using shape optimization," IEEE transactions on visualization and computer graphics, vol. 24, no. 7, pp. 2153–2166, 2017.

[4] S. Jeschke, D. Cline, and P. Wonka, "A gpu laplacian solver for diffusion curves and poisson image editing," ACM Transactions on Graphics (TOG), vol. 28, no. 5, p. 116, 2009.

[5] W.-M. Pang, J. Qin, M. Cohen, P.-A. Heng, and K.-S. Choi, "Fast rendering of diffusion curves with triangles," IEEE Computer Graphics and Applications, vol. 32, no. 4, pp. 68–78, 2011.

[6] S. Jeschke, "Generalized diffusion curves: An improved vector representation for smooth-shaded images," Computer Graphics Forum, vol. 35, no. 2, pp. 71–79, 2016.

[7] A. Jacobson, I. Baran, J. Popovic, and O. Sorkine, "Bounded biharmonic weights for real-time deformation." ACM Trans. Graph., vol. 30, no. 4, p. 78, 2011.

[8] Y.-K. Lai, S.-M. Hu, and R. R. Martin, "Automatic and topology-preserving gradient mesh generation for image vectorization," ACM Transactions on Graphics (TOG), vol. 28, no. 3, p. 85, 2009.

[9] Z. Liao, H. Hoppe, D. Forsyth, and Y. Yu, "A subdivision-based representation for vector image editing," IEEE transactions on visualization and computer graphics, vol. 18, no. 11, pp. 1858–1867, 2012.

[10] J. Sun, L. Liang, F. Wen, and H.-Y. Shum, "Image vectorization using optimized gradient meshes," ACM Transactions on Graphics (TOG), vol. 26, no. 3, p. 11, 2007.

[11] J. H. Elder and R. M. Goldberg, "Image editing in the contour domain," in Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. 98CB36231). IEEE, 1998, pp. 374–381.

[12] T. Sun, P. Thamjaroenporn, and C. Zheng, "Fast multipole representation of diffusion curves and points." ACM Trans. Graph., vol. 33, no. 4, pp. 53–1, 2014.

[13] S. Jeschke, D. Cline, and P. Wonka, "Estimating color and texture parameters for vector graphics," Computer Graphics Forum, vol. 30, no. 2, pp. 523–532, 2011.

[14] S. Lu, W. Jiang, X. Ding, C. S. Kaplan, X. Jin, F. Gao, and J. Chen, "Depth-aware image vectorization and editing," The Visual Computer, vol. 35, no. 6-8, pp. 1027–1039, 2019.

[15] H. Bezerra, E. Eisemann, D. DeCarlo, and J. Thollot, "Diffusion constraints for vector graphics," in Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering, 2010, pp. 35–42.

[16] S. Boyé, P. Barla, and G. Guennebaud, "A vectorial solver for free-form vector gradients," ACM Transactions on Graphics (TOG), vol. 31, no. 6, p. 173, 2012.

[17] M. Finch, J. Snyder, and H. Hoppe, "Freeform vector graphics with controlled thin-plate splines," ACM Transactions on Graphics (TOG), vol. 30, no. 6, p. 166, 2011.

[18] P. Ilbery, L. Kendall, C. Concolato, and M. McCosker, "Biharmonic diffusion curve images from boundary elements," ACM Transactions on Graphics (TOG), vol. 32, no. 6, p. 219, 2013.

[19] H. Lieng, F. Tasse, J. Kosinka, and N. A. Dodgson, "Shading curves: Vector-based drawing with explicit gradient control," Computer Graphics Forum, vol. 34, no. 6, pp. 228–239, 2015.

[20] F. Hou, Q. Sun, Z. Fang, Y.-J. Liu, S.-M. Hu, A. Hao, H. Qin, and Y. He, "Poisson vector graphics (pvg)," IEEE transactions on visualization and computer graphics, 2018.

[21] Y. Li, X. Zhai, F. Hou, Y. Liu, A. Hao, and H. Qin, "Vectorized painting with temporal diffusion curves," IEEE transactions on visualization and computer graphics, 2019.

[22] S. Jeschke, D. Cline, and P. Wonka, "Rendering surface details with diffusion curves," ACM Transactions on Graphics (TOG), vol. 28, no. 5, p. 117, 2009.

[23] K. Takayama, O. Sorkine, A. Nealen, and T. Igarashi, "Volumetric modeling with diffusion surfaces," ACM Transactions on Graphics (TOG), vol. 29, no. 6, p. 180, 2010.

[24] W. A. Barrett and A. S. Cheney, "Object-based image editing," ACM Transactions on Graphics (TOG), vol. 21, no. 3, pp. 777–784, 2002.

[25] H. Fang and J. C. Hart, "Detail preserving shape deformation in image editing," ACM Transactions on Graphics (TOG), vol. 26, no. 3, p. 12, 2007.

[26] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," ACM transactions on Graphics (TOG), vol. 24, no. 3, pp. 1134–1141, 2005.

[27] R. Prévost, W. Jarosz, and O. Sorkine-Hornung, "A vectorial framework for ray traced diffusion curves," Computer Graphics Forum, vol. 34, no. 1, pp. 253–264, 2015.

[28] M. S. Floater, "Mean value coordinates," Computer aided geometric design, vol. 20, no. 1, pp. 19–27, 2003.

[29] K. Hormann and M. S. Floater, "Mean value coordinates for arbitrary planar polygons," ACM Transactions on Graphics (TOG), vol. 25, no. 4, pp. 1424–1441, 2006.

[30] Y. Lipman, J. Kopf, D. Cohen-Or, and D. Levin, "Gpu-assisted positive mean value coordinates for mesh deformations," in Symposium on geometry processing, 2007.

[31] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki, "Harmonic coordinates for character articulation," ACM Transactions on Graphics (TOG), vol. 26, no. 3, p. 71, 2007.

[32] R. M. Rustamov, "Boundary element formulation of harmonic coordinates," Citeseer, Tech. Rep., 2008.

[33] M. Ben-Chen, O. Weber, and C. Gotsman, "Variational harmonic maps for space deformation," ACM Transactions on Graphics (TOG), vol. 28, no. 3, pp. 1–11, 2009.

[34] O. Weber and C. Gotsman, "Controllable conformal maps for shape deformation and interpolation," in ACM SIGGRAPH 2010 papers, 2010, pp. 1–11.

[35] Y. Lipman, D. Levin, and D. Cohen-Or, "Green coordinates," ACM Transactions on Graphics (TOG), vol. 27, no. 3, pp. 1–10, 2008.

[36] O. Weber, M. Ben-Chen, C. Gotsman et al., "Complex barycentric coordinates with applications to planar shape deformation," in Computer Graphics Forum, vol. 28, no. 2, 2009, p. 587.

[37] O. Weber, M. Ben-Chen, C. Gotsman, and K. Hormann, "A complex view of barycentric mappings," in Computer Graphics Forum, vol. 30, no. 5. Wiley Online Library, 2011, pp. 1533–1542.

[38] O. Weber, R. Poranne, and C. Gotsman, "Biharmonic coordinates," in Computer Graphics Forum, vol. 31, no. 8. Wiley Online Library, 2012, pp. 2409–2422.

[39] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in Symposium on Geometry processing, vol. 4, 2007, pp. 109–116.

[40] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler, "A local/global approach to mesh parameterization," in Computer Graphics Forum, vol. 27, no. 5. Wiley Online Library, 2008, pp. 1495–1504.

[41] O. Weber, A. Myles, and D. Zorin, "Computing extremal quasiconformal maps," in Computer Graphics Forum, vol. 31, no. 5. Wiley Online Library, 2012, pp. 1679–1689.

[42] Y. Lipman, "Bounded distortion mapping spaces for triangular meshes," ACM Transactions on Graphics (TOG), vol. 31, no. 4, pp. 1–13, 2012.

[43] L. M. Lui, W. Zeng, S.-T. Yau, and X. Gu, "Shape analysis of planar multiply-connected objects using conformal welding," IEEE transactions on pattern analysis and machine intelligence, vol. 36, no. 7, pp. 1384–1401, 2013.

[44] O. Weber and D. Zorin, "Locally injective parametrization with arbitrary fixed boundaries," ACM Transactions on Graphics (TOG), vol. 33, no. 4, pp. 1–12, 2014.

[45] L. M. Lui, K. C. Lam, S.-T. Yau, and X. Gu, "Teichmuller mapping (t-map) and its applications to landmark matching registration," SIAM Journal on Imaging Sciences, vol. 7, no. 1, pp. 391–426, 2014.

[46] K. C. Lam and L. M. Lui, "Landmark-and intensity-based registration with large deformations via quasi-conformal maps," SIAM Journal on Imaging Sciences, vol. 7, no. 4, pp. 2364–2392, 2014.

[47] L. M. Lui and T. C. Ng, "A splitting method for diffeomorphism optimization problem using beltrami coefficients," Journal of Scientific Computing, vol. 63, no. 2, pp. 573–611, 2015.

[48] R. Chen and O. Weber, "Bounded distortion harmonic mappings in the plane," ACM Transactions on Graphics (TOG), vol. 34, no. 4, pp. 1–12, 2015.

[49] E. Chien, R. Chen, and O. Weber, "Bounded distortion harmonic shape interpolation," ACM Transactions on Graphics (TOG), vol. 35, no. 4, pp. 1–15, 2016.

[50] J. R. Shewchuk, "Triangle: Engineering a 2d quality mesh generator and delaunay triangulator," in Workshop on Applied Computational Geometry. Springer, 1996, pp. 203–222.

[51] S. Liu, A. Jacobson, and Y. Gingold, "Skinning cubic bézier splines and catmull-clark subdivision surfaces," ACM Transactions on Graphics (TOG), vol. 33, no. 6, p. 190, 2014.

[52] A. Jacobson, D. Panozzo, C. Schüller, O. Diamanti, Q. Zhou, N. Pietroni et al., "libigl: A simple c++ geometry processing library, 2016," 2016.

SHUFANG LU is an associate professor at College of Computer Science and Technology, Zhejiang University of Technology. She received her BSc degree in software engineering from Wuhan University, and MSc and PhD degrees in computer science and technology from Zhejiang University. Her research interests include computer graphics and computer vision.

XUEFENG DING is a postgraduate student of the College of Computer Science & Technology, Zhejiang University of Technology. He received his BSc degree in software engineering from Zhejiang University of Technology. His research interests include computer graphics and computer vision.

FEI GAO is a professor at the College of Computer Science and Technology at Zhejiang University of Technology. He received his PhD degree in mechanical engineering from Zhejiang University in 2004. His research interests include image processing, computer vision, and computer-aided design.

JIAZHOU CHEN is an associate professor in Zhejiang University of Technology. He received his double Ph.D. degrees in INRIA Bordeaux Sud-Ouest, France, and the State Key Laboratory of CAD and CG at Zhejiang University, China, in 2012. His research interests include computer graphics and visual media computing.