IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 13, 2018, accepted December 2, 2018, date of publication December 19, 2018,
date of current version January 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2887078

2D Image Deformation Based on Guaranteed Feature Correspondence and Mesh Mapping

YAQIONG LIU[1], (Member, IEEE), XIN LIN[2], GUOCHU SHOU[1], AND HOCK SOON SEAH[3]
[1]School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]The University of Manchester, Manchester M13 9PL, U.K.
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798

Corresponding authors: Yaqiong Liu (yliu4@e.ntu.edu.sg) and Guochu Shou (gcshou@bupt.edu.cn)

This work was supported in part by the Director Foundation under Grant 2017BKL-NSAC-ZJ-02 and in part by the Fundamental Research Funds for the Central Universities under Grant 2018RC03.

ABSTRACT Image deformation has ubiquitous usage in multimedia applications. It morphs one image into another through a seamless transition. Existing techniques either mainly focus on the correspondence mapping of interior features of the objects in two images, without considering object contours, or sketch contours manually, resulting in tedious work for users. Thus, we propose a 2D image deformation method, which extracts object contours automatically, considers both inner features and contours as constraints and preserves image features in terms of visual importance. Our method first automatically extracts the object contours in the source and target images and then allows users to sketch some interior features in both the images. Then, our method tessellates two images to generate two triangular meshes and builds a guaranteed bijective mesh mapping between them. We also prove the bijectivity of our mesh mapping and discuss its other desirable properties. Then, our method generates the intermediate images between the source and target images by calculating the intermediate meshes and pixels of each intermediate image. Our method realizes automatic contour extraction, provides an intuitive user interface and utilizes harmonic maps to establish a bijective mesh mapping. Therefore, it preserves more significant features with less distortion and works well for many image deformation cases in real time.

INDEX TERMS 2D image deformation/morphing, automatic contour exaction, Delaunay triangulation, harmonic map.

I. INTRODUCTION

Various multimedia applications testify to the ubiquitous use of planar shape deformation and animation. Examples include digital composition in cel animation, computer graphics productions, and Web graphics, etc. 2D image deformation, as a fundamental operation in numerous multimedia applications, has long been an active research topic. Studies on image deformation roughly fall into two categories. One is deforming the space (see [1]) where the shape of interest is embedded, and the other one is deforming the shape itself (see [2], [3]) by taking its structure into consideration.

Generally, there are three well-understood major issues in morphing. The first issue is how to extract or specify features, known as the feature problem. The second issue is how to expediently establish a reasonable correspondence mapping between the source and the target, referred to as the correspondence problem. The third issue is how to define the way of transforming the source shape into the target shape, known as the path problem.

Though many works have researched on image deformation and proposed solutions to the abovementioned three issues, it is noted that (i) some existing methods such as [3]–[7] are algorithmically complex and still can be improved in terms of time complexity; (ii) some works such as [1] only regard the interior features of the shape as constraints, and disregard the important role played by the boundary or contours; (iii) some works such as [8] consider the interior together with its boundary, but do not allow users to specify and preserve some important inner features, and therefore users do not have a large degree of control over the deformation results; and (iv) some other works like [9] require users to manually sketch the contours which leads to tedious work for users.

In order to take advantage of both the contours and interior feature constraints, reduce users' cumbersome manual work as much as possible, give users a high degree of control over the deformation results, and meanwhile achieve a natural and smooth deformation effect, this paper presents a novel

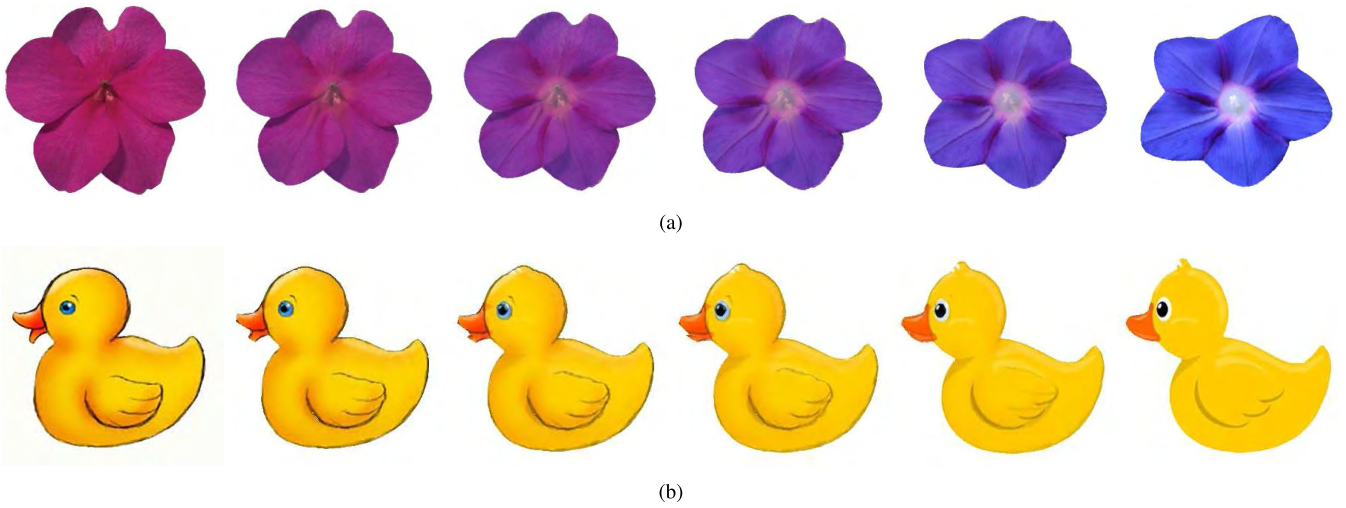

FIGURE 1. 2D image deformation examples generated by our method. (a) Flower-to-flower deformation. (b) Duck deformation.

method for shape-aware 2D image deformation and morphing with guaranteed feature correspondence. Two examples deformed by our method can be found in Fig. 1. Given input source and target images, our method firstly automatically extracts the contours of the objects of interest and then allows the user to specify some inner features within the contours. Our method then constructs a triangular mesh for the source image and target image, respectively, through constrained Delaunay triangulations, taking the contours and the interior features as constraints. The triangular mesh is composed of several patches, each of which is parameterized onto a unit circle domain. With the parameterization, our method can easily establish a correspondence mapping between the patches of the source image and the patches of the target image. Thereafter, our method quickly computes the intermediate meshes and pixels in intermediate images, and finally shows the image deformation process by displaying intermediate images between the source and target images.

This paper makes the following contributions.

- This paper presents a 2D image deformation method which can deform a source image into a target image naturally and smoothly.
- Our deformation method extracts the contour of an object of interest in an image automatically which relieves tedious operations for users, and achieves pleasing morphing results between different objects.
- We prove the bijectivity of our mesh mapping and discuss its distortion metrics and other good properties.

The remainder of this paper is organized as follows. Section II reviews existing image deformation solutions. Section III presents the overview and design scheme, and detailed key steps of our image deformation method. Section IV discusses the bijectivity and minimized distortion of our mesh mapping. Section V reports and discusses our image deformation results, and compares our results with previous methods. Section VI draws a conclusion.

II. RELATED WORK

In this section, we review related work on image deformation.

A. LITERATURE ON IMAGE DEFORMATION TOPIC

Image deformation has received attention from the research community since the early 1990s. To name a few, studies on image deformation include [1], [3], and [9]–[12]. Image deformation has many multimedia applications such as face morphing [13]–[15] and image animation [2], [16].

The underlying technology of image deformation is image warping [10], [17]–[20]. A natural deformation requires a smooth transition when warping the source object into the target object.

B. LITERATURE ON IMAGE DEFORMATION METHODS

Existing techniques for image deformation include mesh-based warping [3], [17], field-based morphing [11], radial basis function [10], and deep learning methods [21], [22].

The basic idea of traditional mesh-based warping techniques (see [3]) is to divide the entire image into quadrilaterals. The user needs to specify a number of control points on both the source image and target image. The traditional mesh-based technique is intuitive and efficient but has its drawbacks: (i) Some points may be moved to where you don't want them to go. (ii) The user cannot have a good control over all points. (iii) It is a cumbersome technique since users need to specify a large number of control points in order to get a good deformation effect. (iv) It is also possible that some features may be lost when generating the intermediate images.

The field-based morphing techniques [11] belong to feature-based techniques. Morphing is conducted based on fields that surround the control elements, that is, feature lines. The advantage of this technique is that it is more expressive. One disadvantage of this technique is that it is time consuming. The distance from all feature lines to each source point

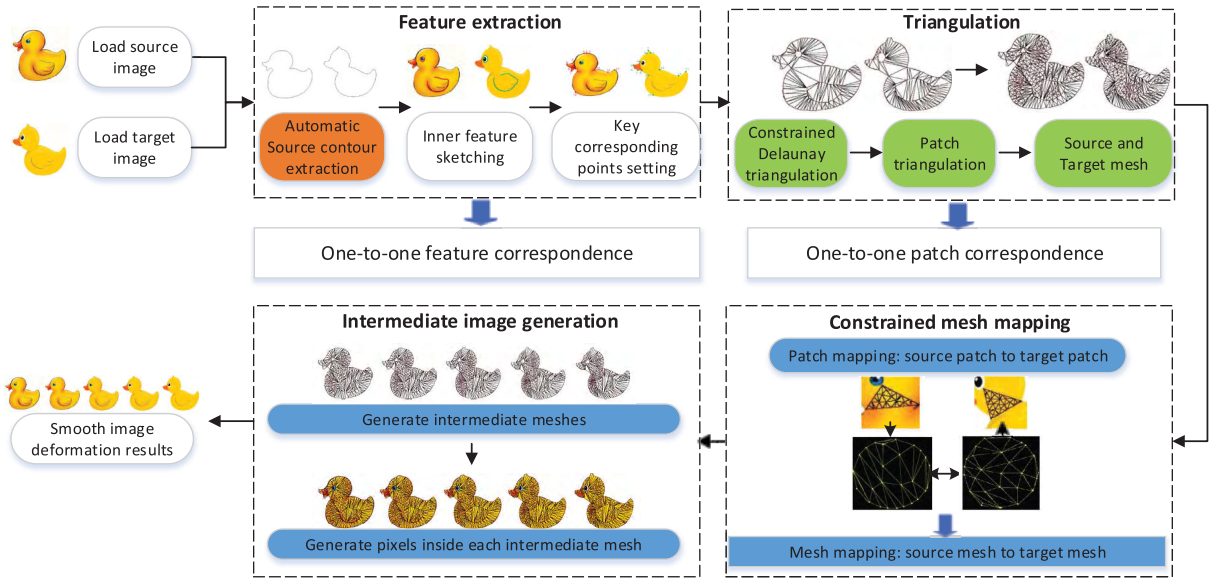

FIGURE 2. The overview and design scheme of our 2D image deformation method.

must be computed before warping. The other disadvantage is that sometimes unwanted interpolation occurs in that one single point is influenced by all feature lines.

The theory of radial basis functions (RBF) [10], [23] provides an attractive framework for image warping. However, this kind of deformation techniques can obtain easy facial deformation, but they cannot produce complex intermediate images and thus cannot achieve complex deformation.

The deep learning methods usually adopt GANs (Generative Adversarial Networks) [24] or their variants such as Cycle-GAN [21], Recycle-GAN [22]) to generate images. Such kind of methods finds many applications in image generation/editing/translation. For example, the "Cycle-GAN" method can translate an image from a source domain X to a target domain Y and thus generate similar images with a different style.

C. COMPARISON WITH EXISTING METHODS

As mentioned in Section I, some existing works [3]–[7] still can be improved in terms of runtime; Some other works [1], [9] either neglect contour constraints or require users to sketch the contours manually. Many mesh-based methods like [3], [17], and [25] do not provide a bijectivity proof for the mapping between the meshes.

In order to overcome the drawbacks of the abovementioned existing methods, our method utilizes both mesh-based morphing and field-based morphing techniques, combines their advantages, and therefore outperforms traditional mesh-based morphing and field-based morphing.

Furthermore, our method takes the significance of the contour of the object in an image into account in addition to considering interior features. Instead of requiring users to sketch object contours manually, our system extracts the boundary contours fully automatically. We also provide an intuitive user-controllable interface for animators to sketch interior feature curves, lines, points, or holes as interior constraints. Meanwhile, we prove the bijectivity of our mesh mapping and discuss the distortion metrics.

In addition, compared with the deep learning methods such as Cycle-GAN [21] or Recycle-GAN [22], our method differs with them in terms of purposes and application scenarios. (i) The "Cycle-GAN" or "Recycle-GAN" method can translate images or retarget a sequence of images. For instance, input a sequence of Donald Trump's facial expression images, then the "Recycle-GAN" method can output a corresponding sequence of Barack Obama's facial expression images which have similar styles to Donald Trump's facial expression images. However, it cannot generate the **intermediate** images between two input images. (ii) Our deformation method aims to generate the sequence of intermediate images between two input images which are referred to as the source image and the target image, so that users can watch the gradual morphing process between the source and the target little by little. Therefore, the application scenarios of the "Cycle-GAN" or "Recycle-GAN" method and our method are different. The "Cycle-GAN" or "Recycle-GAN" method uses currently hot deep learning techniques and looks upmarket, but, given two input images, it surely cannot generate the intermediate image sequences for the morphing process between the two input images, so that it cannot morph the source image **gradually** into the target image. But our method can achieve image morphing little by little.

III. IMAGE DEFORMATION METHOD

In this section, we present our image deformation method in detail.

A. OVERVIEW AND DESIGN SCHEME

The overview and design scheme of our image deformation method is shown in Fig. 2. Given the input source and

target images I and I' containing an object $O \subset I$ and $O' \subset I'$, after we load them into our image deformation system, our method firstly extracts the two objects' features, including contours which are extracted automatically, and interior features which are specified by the user. Then our method generates a triangular mesh for the two images, respectively, taking the feature constraints (including both contour constraint and interior feature constraint) into consideration. Using harmonic maps [26], we can conveniently construct a constrained correspondence mapping for 2D image animations. Thereafter, we generate the intermediate triangular mesh from the source and the target meshes with the corresponding mapping mentioned above. We thus calculate each pixel of the intermediate shape by linear interpolation of the color values of the corresponding pixels in the source and the target shapes. Finally, we generate a sequence of smooth morphing results and achieve image deformation from the source image to the target image gradually, smoothly and naturally.

Hence, in general, our deformation scheme is mainly composed of four modules, i.e., feature extraction module, triangulation module, constrained mesh mapping establishment module, and intermediate image generation module.

- **Feature extraction**: Features include contours and inner features. In this module, our method automatically extracts the contours of the source and target images, and subsequently allows the users to sketch some user-controllable inner features including inner feature points, lines, curves, or holes inside the contour on both images, and finally allows users to set several pairs of user-controllable corresponding points along the contours of the source and target images. In other words, this module consists of three key steps: (i) automatic contour extraction, (ii) inner feature sketching, and (iii) key corresponding points setting. Via this module, we obtain a bijective feature correspondence between the source image and the target image.
- **Triangulation**: In this module, our method first applies constrained Delaunay triangulation to the source image based on its features. Then the triangular topology of the source image is copied to the target image based on the feature points correspondence built in the first module, and thus we also get a constrained Delaunay triangulation of the target image. Subsequently, patch triangulation is applied for each triangle (triangular mesh patch) of the triangulations on both images and then we obtain a dense source triangular mesh and a dense target triangular mesh. As a result, we obtain a one-to-one patch correspondence via this module.
- **Constrained mesh mapping**: This module constructs a constrained mesh mapping between the source triangular mesh and the target triangular mesh using harmonic map techniques.
- **Intermediate image generation**: On the basis of the constrained mesh mapping, our method generates a series of intermediate meshes between the source and target meshes, and computes pixels in each intermediate image based on each intermediate mesh. Finally, we automatically display the smooth deformation sequence results.

B. KEY STEPS

As mentioned in Section III-A, our system has four modules. These modules altogether contain six key steps which are referred to as automatic contour extraction, inner features sketching, corresponding points setting, triangulations, constrained mesh mapping establishment, and intermediate images generation. In this subsection, we take the duck deformation shown in Fig. 1(b) as an example to present these key steps in detail. At the beginning, we load the source image and target image as shown in Fig. 3.

FIGURE 3. An example of source image and target image.

1) AUTOMATIC CONTOUR EXTRACTION

The first key step is automatic contour extraction, which can be further divided into the following three substeps.

(i) **Gaussian blur:** If we apply edge detection directly without Gaussian blur, most natural images after grayscale processing like Fig. 4(a) have noises. Thus a Gaussian blur filter [27] is utilized prior to edge detection to reduce noises in the image. Most edge-detection algorithms are sensitive to noises and a Gaussian blur filter improves the result of the edge-detection algorithm by filtering single pixel noises. The Gaussian blur filter utilizes a Gaussian function as given in Eq. 1, to calculate the transformation and apply it to every pixel of the image.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

In Eq. 1, x and y denote the distances to the origin along the horizontal axis and the vertical axis, respectively, and σ denotes the standard deviation of the Gaussian distribution. Surfaces produced by Eq. 1 have concentric circular contours from the center point, following a Gaussian distribution. Values of $G(x, y)$ are utilized to construct a convolution matrix to apply to the original image. The new value of each pixel is set as a weighted average of the neighborhoods of this pixel. Thus, the value of the original pixel has the largest weight as it has the highest Gaussian value. Neighboring pixels have smaller weights since their distances to the original pixel increase. This leads to a blur which can preserve edges and boundaries better, compared with other more uniform blurring filters.

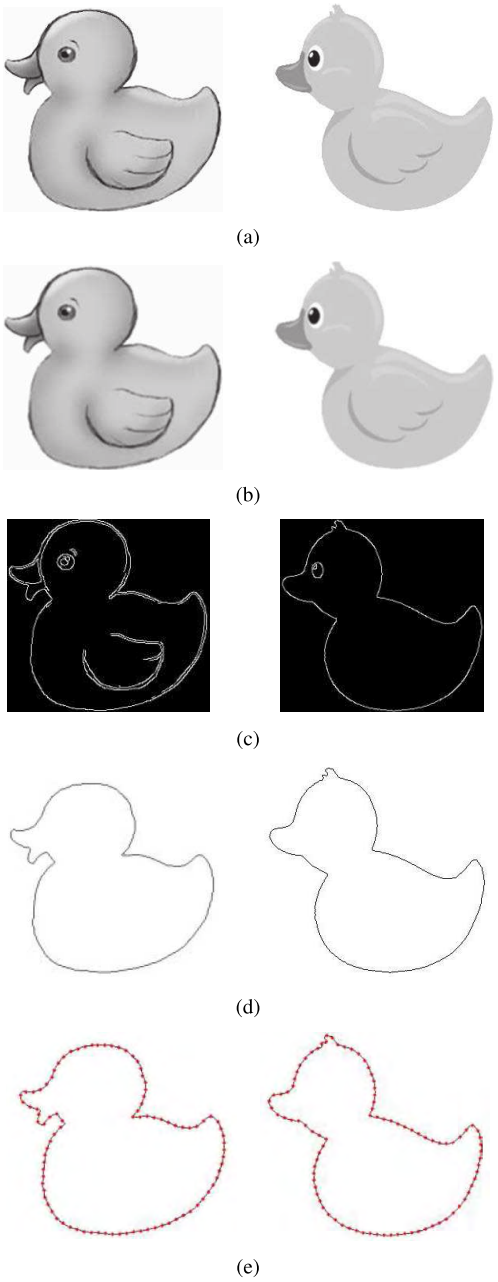

FIGURE 4. Automatic contour extraction. (a) Images after grayscale processing. (b) Images after Gaussian blur. (c) Images after edge detection. (d) Contour extraction. (e) Contour points.

(ii) **Edge detection:** As shown in Fig. 4(b), although most single pixel noises are removed after Gaussian blur, some multi-pixel noises still exist. Most of these multi-pixel noises are likely to be identified as an edge and thus these noises can influence the accuracy of contour extraction to a great extent. As shown in Fig. 4(c), our method can discard some false and fine edges using Canny edge detection operator since this operator has three main advantages: low error rate, good localization and minimal response.

(iii) **Contour extraction:** Based on edge detection, our method conducts the third substep, i.e., contour extraction using a contour extraction function provided by OpenCV. In this function, we use the parameter "CV_RETR_EXTERNAL" which means that we only preserve the external contour. As shown in Fig. 4(d), only the external boundary contour of the object of interest is preserved. The results of contour extraction are the points or pixels of the contour and their connectivity. There is no need to keep all contour points, so our system uniformly samples the contour points. Note that the sampling results of the contour points should reflect the significant features of the contour. Therefore, a suitable sampling step length enables the sampled contour points to be evenly distributed. The relatively even distribution is beneficial for preserving the characteristics of a contour. The number of sampled contour points in the target image should be equal to that in the source image. An example satisfying the aforementioned requirements is shown in Fig. 4(e), in which the red points are the sampled contour points. The concavities and convexities of the contour are maintained.

2) INNER FEATURES SKETCHING

When the object of interest in an image has some inner features, only extracting the contour of the object may produce unsatisfactory deformation effect. The reason is that, in addition to the contour features, interior features also represent significant characteristics of an object of interest in an image. Therefore, these inner features should also be considered as feature constraints and this is why the second key step of our deformation method is to sketch inner features.

Inner features can be inner lines, curves or holes. Similar to the first step, i.e., contour extraction, the number of inner feature lines or curves, and holes on each inner feature line or curve, and the number of hole points for each hole in the target image must be equal to that in the source image. In this step, the user only needs to sketch the inner feature curves/lines in pairs (one on the source image, the other on the target image), and does not need to set any point manually on the feature curves. The reason is that it is our backstage program that automatically memorizes the order of the feature curves/lines that the user sketches, and it is also our program that samples and saves equal number of feature points along the sketched source feature curve/line and target feature curve/line at the backstage. All the user needs to do is to sketch equal number of feature curves/lines on the source image and the target image. This means that the point correspondences between the source inner feature curve and the target inner feature curve are established automatically by our program.

Note that each pair (one in the source image, the other one in the target image) of feature lines, curves, and holes should be sketched in the same direction with the corresponding start point, to guarantee the feature correspondence. Figure 5 shows an example of inner features (e.g. feature curve, hole) sketching. In Fig. 5(a)(b), the feature curves (representing the wings) on both the source image and the target image are drawn in the same direction as the arrows show, and both holes (representing the eyes) are drawn counterclockwise.

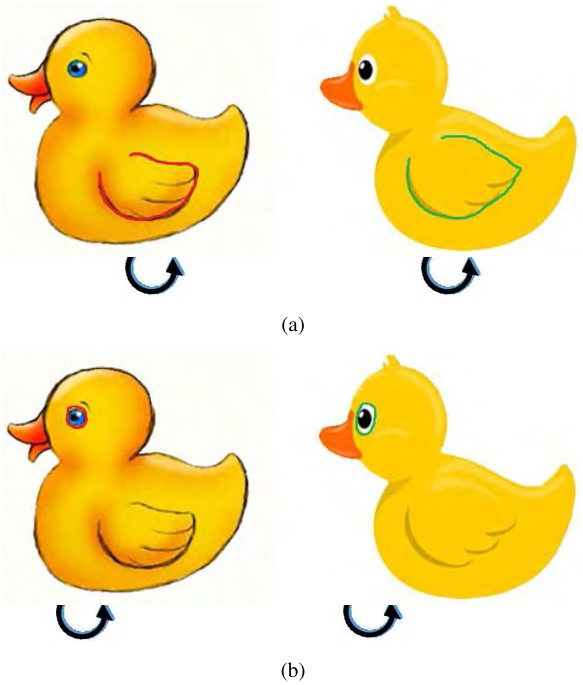

FIGURE 5. Sketching inner features (points, lines, or holes). The counterclockwise arrow below each figure means the inner feature lines or holes are sketched in the counterclockwise direction. (Note that inner features are sketched in the same direction on both images).
(a) Sketching inner feature curves. (b) Sketching inner feature holes.

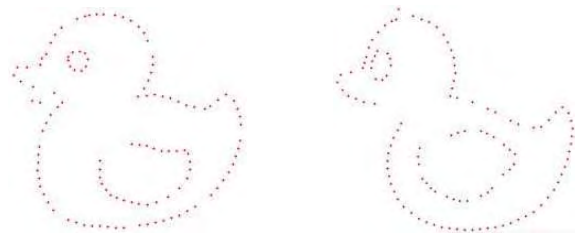

FIGURE 6. All feature points (including contour points and inner feature points). Note that the number of feature points of the source image equals to the number of feature points of the target image.

Finally, a one-to-one point correspondence between inner feature points, lines, curves, and holes is established.

After the first two key steps, we can obtain two sets of feature points including contour points and inner feature points on the two images, as shown in Fig. 6.

3) KEY CORRESPONDING POINTS SETTING

A fine feature correspondence can make the subsequent mesh mapping more accurate and smoother. In order to guarantee feature correspondence, our method allows users to set **several** pairs of corresponding points along the extracted contours on the source image and target image.

Each pair of corresponding points include one point on the contour of the source image and its corresponding point on the contour of the target image. Thus, a one-to-one contour point correspondence is constructed. These corresponding points are usually specified at some places which have visual importance or visual characteristics, like the convexity or concavity of the contour. The purpose of setting corresponding points is to establish the correspondence between each visually important feature in the source image and its corresponding visually important feature in the target image.

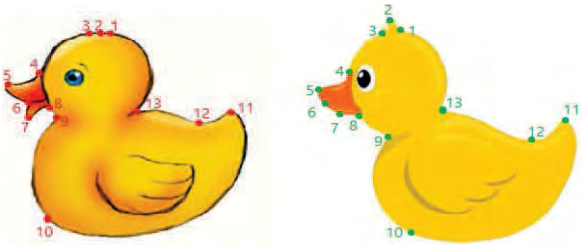

FIGURE 7. User sets the corresponding points on the contours of the source image and the target image, in order to make the important features of the source image correspond to the appropriate features of the target image.

An example is shown in Fig. 7 in which equal number of corresponding points are set on the contours of the objects in the source image and target image, and each point in the source image is mapped to its corresponding point in the target image. For instance, Point 2, 4, 9, 11, and 13 in the source image are mapped to Point 2, 4, 9, 11, and 13 in the target image, respectively.

4) TRIANGULATIONS

Based on the guaranteed feature correspondence, our method performs the fourth key step, i.e., applying triangulations to the source and the target images. We utilize the method of 'Triangle'[1] [28] which can compute Delaunay triangulations and constrained Delaunay triangulations exactly and generate guaranteed-quality meshes (without small angles). It can achieve a low time complexity of $O(n \log n)$.

This step can be decomposed into two substeps, constrained Delaunay triangulation and patch triangulation.

a: CONSTRAINED DELAUNAY TRIANGULATION

First, we apply constrained Delaunay triangulation only to the source image based on the features including the contour constraint and the interior feature lines, curves, and holes (if any) which are the inner feature constraints. If no inner holes exist, our method generates a sparse triangular mesh only composed of triangles. If inner holes exist, it generates a sparse triangular mesh composed of some triangular patches and polygonal hole patches. The generated triangles or polygons are so-called rough patches.

Then, our method does triangulation to the target image: Thanks to the point-to-point correspondence, our method simply copies the connections of the source mesh to connect the target points according to the connectivity among source points and thus we get a target mesh which is compatible with the source mesh. However, in this process, some rough patches in the target mesh may intersect. It is also possible that three points are collinear. Thus, our method allows users to move some target points to avoid this problem and thus

[1] freely available at http://www.cs.cmu.edu/%7Equake/triangle.html

generates a legal triangular mesh for the target image. This operation does not change the connectivity.

Hence, the target mesh and the source mesh have the same topological structure and connectivity. They have equal number of rough patches and the correspondence between source rough patches and target rough patches is bijective. Thus we build a rough-patch-to-rough-patch correspondence, which is very important for the reason that it represents the important feature correspondence to some extent since the vertices of rough patches are all feature points.

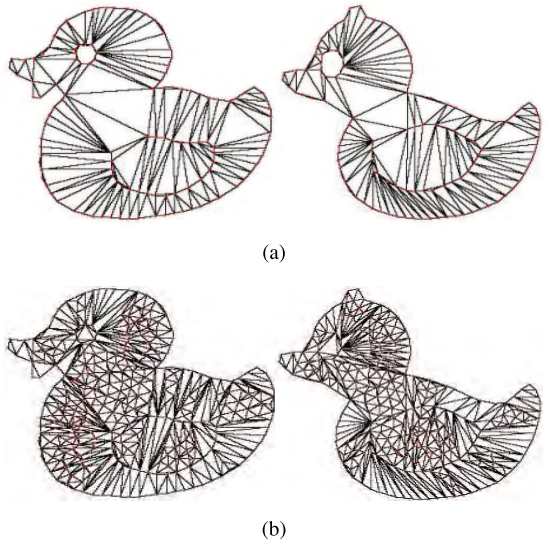

FIGURE 8. Triangulations. (a) Patch layout after constrained Delaunay triangulation. (b) Patch triangulation.

Figure 8 shows the source mesh and the target mesh with the vertices shown in Fig. 6. Thus there exists a one-to-one correspondence between each patch of the source triangular mesh and the corresponding patch of the target triangular mesh.

b: PATCH TRIANGULATION

The patch triangulation substep is to further triangulate each large rough patch into a triangular mesh, aiming to form a denser triangular mesh of the source image and the target image, which enables more accurate calculation of the pixels in intermediate images generated in the sixth key step ''intermediate image generation''.

After the first substep, i.e., ''constrained Delaunay triangulation'', most triangles (rough patches) as shown in Fig. 8(a) are large. In order to achieve a better deformation effect and a low time complexity, in this second substep, our system automatically does conforming constrained Delaunay triangulation only to large rough patches. Based on the three vertices and three edges of this triangle (rough patch), we still use the tool 'Triangle' [28], which provides multiple constrained (conforming) Delaunay triangulation methods with high quality. We use ''Triangle −pq xx.poly'' command to triangulate a planar straight line graph into quality mesh with no angles smaller than 20 degrees. In this process, new vertices (called Steiner points) will be generated along the edge of the triangle or inside the triangle. That's why we can finally turn each triangle into a triangular mesh patch which contains quite a few vertices in addition to the three vertices of the triangle.

Note that in this patch triangulation process, new points may be generated in one rough patch, aiming at dividing it into denser segmented patches. Some of these new points are generated along the boundary edges of a segmented patch. However, two adjacent segmented patches must share one common edge, but the number of points on this common edge in two adjacent segmented patches may be unequal. To deal with this problem and to form a whole triangular mesh for the source image or target image, a re-triangulation process is applied to these segmented patches.

The algorithm of re-triangulation can be described as follows: (i) First, for each segmented patch S, for each boundary edge e_i ($i = 1, \cdots, n$, where n denotes the number of boundary edges of patch S) of this patch, let $P_{e_{i1}}$ and $P_{e_{i2}}$ denote the set of points on edge e_i in S and the set of points on this edge in the adjacent triangular patch which shares the common edge e_i with S, respectively. (ii) Let $P_{e_i} = P_{e_{i1}} \cup P_{e_{i2}}$, which denotes the set of all points that locate on edge e_i. (iii) After obtaining each points set P_{e_i} for all boundary edges of patch S, do triangulation to patch S based on boundary points set $P = \bigcup_{i=1}^{n} P_{e_i}$ and thus we obtain a new triangular patch S_{new}.

After re-triangulation to all segmented patches, all common edges between two adjacent patches share the same edge points. Then we combine each new triangular patch into one entire dense triangular mesh. Finally, the source mesh and target mesh are both dense triangular meshes. Thus we obtain a segmented-patch-to-segmented-patch correspondence. Thus rough patches become segmented patches. For example, as shown in Fig. 8(b), the source hole rough patch (the eye of the source duck) and the target hole rough (the eye of the target duck) patch are triangulated to be a triangular mesh, respectively. Figure 8(b) shows two dense meshes after patch triangulation. Thus, two sparse triangular meshes as shown in Fig. 8(a) turn to two dense triangular meshes as shown in Fig. 8(b).

5) CONSTRAINED MESH MAPPING ESTABLISHMENT

A good deformation effect requires the correspondence between the source mesh and the target mesh, since only by taking advantage of the correspondence, can we gradually deform a source mesh into a target mesh. However, after patch triangulation, the number of new small triangles generated in a source segmented patch and that in its corresponding target segmented patch are always different, resulting in different mesh topologies of the source segmented patch and target segmented patch. Different topological structures mean that the correspondence of the source mesh and the target mesh cannot be constructed directly. Therefore, the fifth key step is to establish a constrained mesh mapping between the source mesh and the target mesh. A preferred method for mesh

parameterization [29]–[31] is harmonic map, which is of a great value due to its advantages (e.g., minimizing angle distortion, preserving aspect ratios of triangles, etc).

Suppose O_s and O_t denote the source mesh and the target mesh, respectively. $P_s \in O_s$ and $P_t \in O_t$ are a pair of segmented patches, both of which are genus-0 surfaces with only one boundary. Our task is to find a bijective and smooth mapping $\phi : P_s \to P_t$.

Instead of computing the mapping directly, we firstly parameterize P_s to the unit disc using harmonic map, i.e., $f : P_s \to \mathbb{D}$ such that $\triangle f = 0$. The process of harmonic map is as follows: Let V_j denote a boundary vertex of P_s, V_i denote an inner vertex of P_s, and E denote the elastic energy of the whole patch. The boundary point V_j is mapped to the boundary point V_j^* of $\mathbb{D}$ using arc length parametrization, and then the interior point V_i is mapped to the interior point V_i^* of $\mathbb{D}$ by minimizing E according to Eq. 2. After letting the partial differential of E be equal to zero as expressed in Eq. 3, we can obtain the interior point V_i^* as given in Eq. 4, where α_{ij} serves as spring constants as defined in Eq. 5, in which L_{ij} denotes the length of edge $\{i, j\}$ as measured in the initial patch mesh P_s, and $Area_{i,j,k}$ denotes the area of face $\{i, j, k\}$ as measured in P_s. Note that each interior edge $\{i, j\}$ is incident to two faces, say $\{i, j, k_1\}$ and $\{i, j, k_2\}$, so the right part of Eq. 5 has two terms. Note also that each boundary edge is incident to only one face so that the right part of Eq. 5 should have only one term in case $\{i, j\}$ is a boundary edge.

$$E = \frac{1}{2} \sum_{\{i,j\} \in Edges(P_s)} \alpha_{ij} ||V_i^* - V_j^*||^2 \tag{2}$$

$$\frac{\partial E}{\partial V_i} = \sum_{\{i,j\} \in Edges(P_s)} \alpha_{ij}(V_i^* - V_j^*) = 0 \tag{3}$$

$$V_i^* = \sum_{j:\{i,j\} \in Edges(P_s)} \alpha_{ij} V_j^* \tag{4}$$

$$\alpha_{ij} = \frac{L_{ik_1}^2 + L_{jk_1}^2 - L_{ij}^2}{Area_{i,j,k_1}} + \frac{L_{ik_2}^2 + L_{jk_2}^2 - L_{ij}^2}{Area_{i,j,k_2}} \tag{5}$$

Likewise, we also parameterize P_t to a unit disc using harmonic map $g : P_t \to \mathbb{D}'$. Via harmonic map, we can minimize the angle distortion and preserve aspect ratios of triangles. We subsequently figure out a bijective mapping between two unit discs $h : \mathbb{D} \to \mathbb{D}'$ using harmonic map $\triangle h = 0$, so as to minimize the angle distortion and preserve aspect ratios of triangles. More details about harmonic map and its applications can be found in [26] and [32].

Thus, the constrained mapping between two patches can be obtained by the composite mapping $\phi = f \circ h \circ g^{-1}$ as shown in the commutative diagram below. We prove its bijectivity and discuss its metric distortion in Section IV.

$$\begin{array}{ccc} P_s & \xrightarrow{\phi} & P_t \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle g} \\ \mathbb{D} & \xrightarrow[h]{} & \mathbb{D}' \end{array}$$

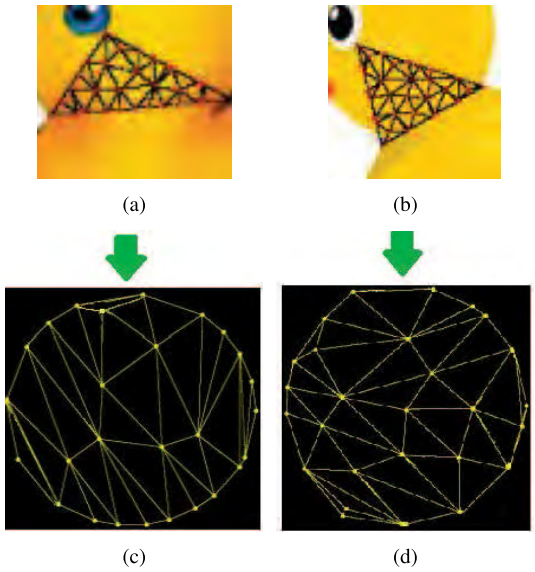

FIGURE 9. Constrained patch mapping. The source triangular patch shown in (a) is mapped onto a circular domain and thus the source circular patch shown in (c) is obtained. The target triangular patch shown in (b) is mapped onto a circular domain and thus the target circular patch shown in (d) is obtained. Then we establish a map between two circular patches shown in (c) and (d). In this way, we can build the constrained patch mapping between the source patch shown in (a) and the corresponding target patch shown in (b).

After building a constrained mapping between each pair of corresponding segmented patches, our method automatically combines all constrained patch mapping together to form a constrained mesh mapping which maps the entire source mesh to an entire target mesh. Thus the correspondence between two meshes is rebuilt. An example of constrained patch mapping is shown in Fig. 9.

6) INTERMEDIATE IMAGE GENERATION

After the constrained mesh mapping between the source mesh and the target mesh has been obtained, the intermediate meshes can be figured out. Hence, the final step is to generate the intermediate shapes and fill in the intermediate shapes with pixels. Thus, this step can be divided into two substeps, i.e., intermediate mesh calculation and intermediate images generation.

a: INTERMEDIATE MESH CALCULATION

Our method utilizes linear interpolation to generate the intermediate meshes based on the mapping between the source mesh and the target mesh. It calculates the position of each vertex v in the intermediate mesh by linear interpolation of the position of the corresponding vertex v_s in the source mesh and the position of the corresponding vertex v_t in the target mesh.

$$p(v) = (1 - \mu) * p(v_s) + \mu * p(v_t) \tag{6}$$

where $\mu \in [0, 1]$ is a parameter controlling the interpolation and function $p(v)$ denotes the position of vertex v. Our method in total generates fifty intermediate meshes in each

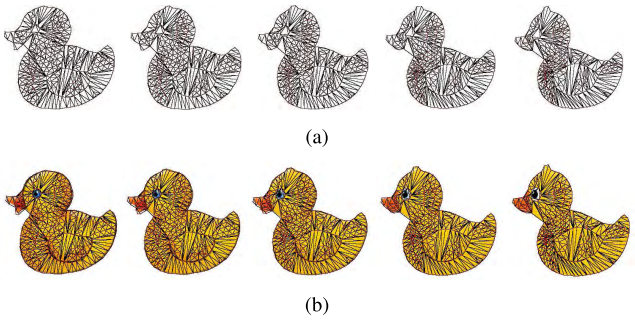

FIGURE 10. Intermediate mesh sequences and intermediate images. (a) From left to right: intermediate mesh #5, intermediate mesh #15, intermediate mesh #25, intermediate mesh #35, intermediate mesh #45. (b) From left to right: intermediate image #5, intermediate image #15, intermediate image #25, intermediate image #35, intermediate image #45.

deformation case. Figure 10(a) shows five uniformly selected intermediate meshes from fifty frames for duck deformation.

b: INTERMEDIATE IMAGES GENERATION

Forward mapping may induce 'hole' problems which means that there exist some missing pixels that cannot be calculated. In order to avoid the 'holes', instead of interpolating the corresponding source pixel and target pixel directly to find the intermediate pixels, our method starts from the position of each intermediate pixel of each intermediate image and finds the corresponding source pixel and target pixel, respectively. For each intermediate pixel p_i, our method first finds the small triangle t in which it locates, and the position of p_i (barycentric coordinates of p_i) can be represented as the linear combination of the coordinates of three vertices v_1, v_2, and v_3 of this triangle. As each triangle in the intermediate mesh corresponds to one triangle in the source mesh and one triangle in the target mesh, our method then finds the corresponding small triangle t_s in the source mesh and the corresponding triangle t_t in the target mesh, and applies the same linear combination of the barycentric coordinates to these two triangles, to obtain the position of corresponding pixel p_s in the source image and p_t in the target image. Finally, our system calculates the pixel value of p_i using the linear interpolation:

$$p_i = (1 - \mu) * p_s + \mu * p_t \tag{7}$$

where $0 \leq \mu \leq 1$. In total, fifty intermediate images are generated in each deformation case. In Fig. 10(b), five uniformly selected intermediate images with pixels and corresponding meshes are displayed.

IV. BIJECTIVITY AND MINIMIZED DISTORTION

A. BIJECTIVITY

In this subsection, we prove that our established mesh mapping presented in Section III-B.5 is a bijective mapping.

Theorem 1: The mapping between a pair of segmented patches, i.e., the source patch P_s and the target patch P_t, is a bijective mapping. ♠

Proof: Consider a pair of segmented patches. The mapping is obtained by $\phi = f \circ h \circ g^{-1}$. Notice that the boundary of the source segmented patch is mapped to the boundary of the target segmented patch via arc length parametrization and that the mapping of the corresponding boundaries are bijective. Lipman [33] proved that as long as the boundary of the source domain is mapped bijectively to a simple polygonal curve, then this map is a global bijection between the source domain and the target domain bounded by this curve. Therefore, the mapping between this pair of segmented patches is a bijection.

Theorem 2: The constrained mesh mapping between the source mesh and the target mesh is bijective. ♠

Proof: Theorem 1 tells us that the mapping between each pair of segmented patches P_s and P_t is bijective. Since there is a one-to-one correspondence between all pairs of segmented patches, after we connect all the patch mappings together to form an entire mesh mapping, we can conclude that the entire mapping between the source mesh O_s and the target mesh O_t is bijective.

B. MINIMIZING DISTORTION

Harmonic maps minimize metric dispersion, which is a measure of metric distortion [34]. Therefore, our final mesh mapping $\phi = f \circ h \circ g^{-1}$ can minimize metric dispersion because $f : P_s \to \mathbb{D}$, $g : P_t \to \mathbb{D}$, and $h : \mathbb{D} \to \mathbb{D}'$ are all harmonic maps.

Harmonic map has quite a few good properties. To name a few, for a harmonic map $f : P_s \to \mathbb{D}$, (i) it is infinitely differentiable on each face of P_s; (ii) it is independent on the triangulations of P_s; (iii) the harmonic map tends to minimize such distortion while maintaining the embedding and attempting to preserve aspect ratios of triangles.

V. RESULTS

This section reports and discusses the results generated by our image deformation method. Many deformation results demonstrate that our deformation method can achieve natural image deformation with user-controllable guaranteed feature correspondence in real time. In addition, we also compare our results with the results generated by existing techniques.

A. SMOOTH DEFORMATION RESULTS

Our deformation method can generate various smooth deformation results. Due to limited space, we only list several deformation examples in this subsection. Note that each deformation example in total has fifty frames, from which we uniformly select five or six frames with the source and the target images being the first and last frames for each deformation example, due to space limits.

Figure 11 shows two flower deformation examples. Observe that one round blue flower is deformed into a fluted purple flower, and each intermediate image shows a complete and natural flower, as shown in Fig. 11(a). Similar successful flower deformation results can be found in Fig. 11(b).

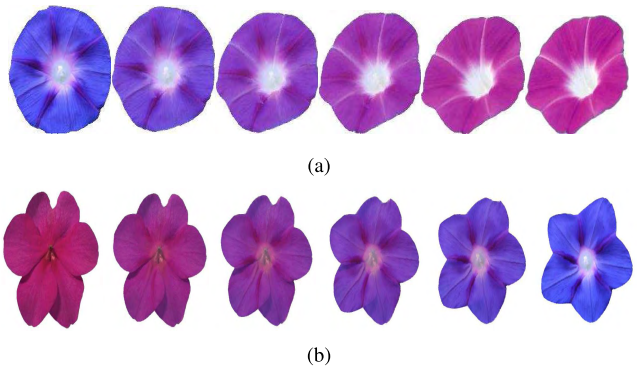

FIGURE 11. Flower deformation examples. In each subfigure, the first image and last image are the source image and target image, and others are intermediate frames. (a) Flower deformation. (b) Flower deformation.

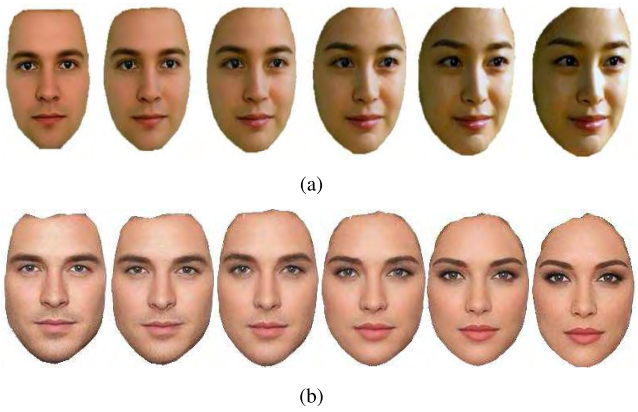

FIGURE 12. Face deformation examples. In each subfigure, the first image and last image are the source image and target image, and others are intermediate frames.

Figure 12 shows two human face morphing examples. Observe that two male faces are gradually deformed into female faces, with smooth and natural deformation effects. Each intermediate image in between the source image and the target image shows a clear, complete and flawless human face.

Figure 13 describes four animal deformation examples. These image deformation sequences perfectly show that a yellow duck image is deformed into another yellow duck image (shown in Fig. 13(a)), a dog image is deformed into another dog image (shown in Fig. 13(b)), a standing fat elephant is deformed into a jumping thin elephant (shown in Fig. 13(c)), and a purple monkey is deformed into a green monkey (shown in Fig. 13(d)). Observe that each intermediate image shows a complete and seamless transition from the source image to the target image.

Some more deformation examples are shown in Fig. 14, where a dancer is seamlessly transited into another dancer with different poses as shown in Fig. 14(a), a real umbrella image is deformed into a cartoon flower as shown in Fig. 14(b), a green leaf is deformed into a pinkish purple flower as shown in Fig. 14(c), and a male student's face is gradually deformed into a female student's face as shown in Fig. 14(d).

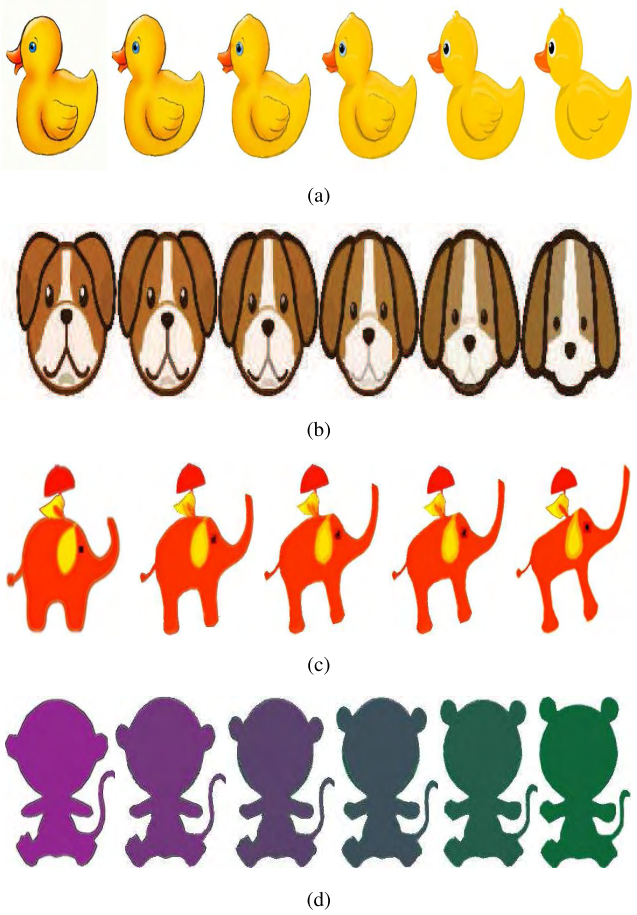

FIGURE 13. Animal deformation examples. In each subfigure, the first image and last image are the source image and target image, and others are intermediate frames. (a) Duck deformation. (b) Dog deformation. (c) Elephant deformation. (d) Monkey deformation.

In summary, various results displayed in the above mentioned figures show quite satisfying deformation. Each object in the in-between images looks good. During the deformation process, not only the inner features are well preserved but also the shape has a smooth transformation due to the existence of the contour features.

B. COMPARISON WITH CONVENTIONAL METHODS

We compare our image deformation method with existing image deformation methods such as [9] and [35]. We report the comparison results as follows.

1) AVOIDING FADE-IN AND FADE-OUT EFFECTS BY CONSIDERING CONTOUR CONSTRAINTS

Figure 15 shows two groups of deformation sequences. The top intermediate images are generated by another technique which only considers inner features while disregarding the contour. The bottom intermediate images are generated by our method, which can extract both the contour and inner features. In order to explore whether the contour feature has an important influence on the deformation effects, we let the users specify the inner features (e.g, the eyes, nose, mouth and eyebrow) when applying both of the

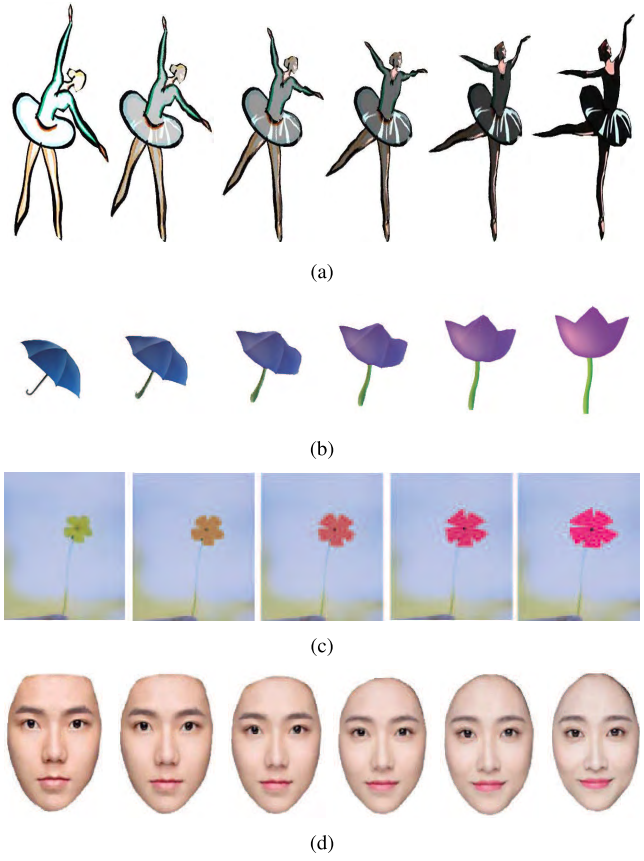

FIGURE 14. More deformation results. In each subfigure, the first image and last image are the source image and target image, and others are intermediate frames. (a) Dancer deformation. (b) Umbrella-flower deformation. (c) Leaf-flower deformation. (d) Students' face deformation.

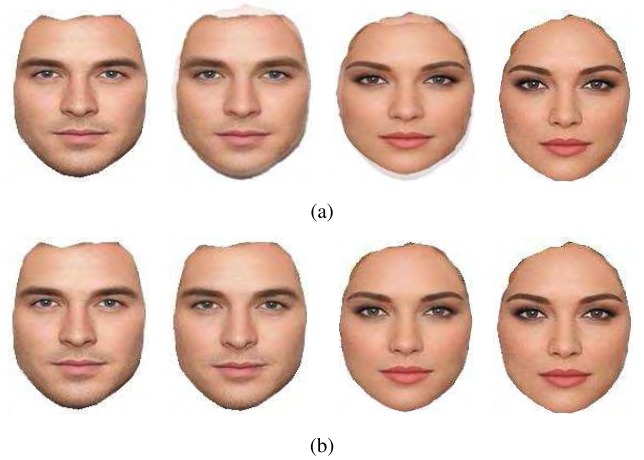

FIGURE 15. Comparison: results without considering contour constraints vs. results by our deformation system which considers contour constraints. From left to right: Frame #0, Frame #16, Frame #32, Frame #49. In each subfigure, the source image and target image are Frame #0 and Frame #49, and others are intermediate frames. (a) Fade-in and fade-out effects without contour constraints. (b) Smooth and natural effects with contour constraints.

two techniques. When only considering the inner features, the intermediate images have a fade-in and fade-out effect and a blur on the boundary, as shown in Fig. 15(a). When both the inner features and contour feature are considered,

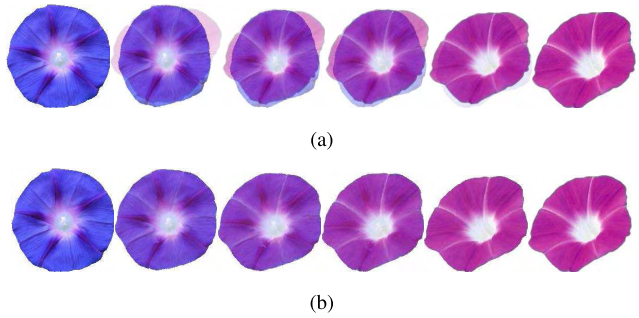

FIGURE 16. Comparison: results without considering contour constraints vs. results by our system which considers contour constraints. From left to right: Frame #0, Frame #10, Frame #20, Frame #30, Frame #40, Frame #49. In each subfigure, the source image and target image are Frame #0 and Frame #49, and others are intermediate frames. (a) Fade-in and fade-out effects without contour constraints. (b) Smooth and natural effects with contour constraints.

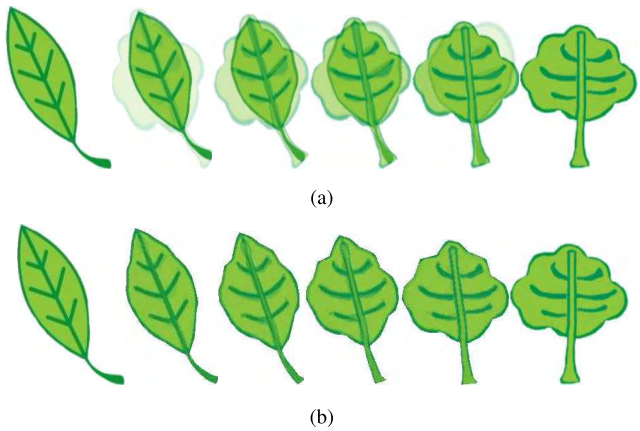

FIGURE 17. Comparison: results without considering contour constraints vs. results by our system which considers contour constraints. From left to right: Frame #0; Frame #10; Frame #20; Frame #30; Frame #40; Frame #49. In each subfigure, the source image and target image are Frame #0 and Frame #49, and others are intermediate frames. (a) Fade-in and fade-out effects without contour constraints. (b) Smooth and natural effects with contour constraints.

the intermediate images do not suffer from this problem, as shown in Fig. 15(b), where each intermediate image shows a considerably real and natural face.

When deforming two flowers with different shapes, the top flowers as shown in Fig. 16(a) generated by other techniques without considering contour constraints have a really worse appearance due to the lack of the contour feature, especially in Frame #20 and Frame #30. The cross dissolve problem is much severer in Fig. 17(a). In contrast, the intermediate flowers generated by our method are natural and smooth, as shown in Fig. 16(b). Similar deformation results generated by our method can be seen in Fig. 17(b), where each intermediate leaf does not have any cross dissolve problem and is deformed from the source image to the target image smoothly and naturally.

From the above examples, we find that the advantage of our method is considerably obvious since we take the contour constraints into consideration in addition to the interior feature constraints. Thus, we can conclude that due to the

contour constraints, our technique produces a smoother, more natural and more realistic image deformation.

2) RELIEVING TEDIOUS OPERATIONS FOR USERS

Our deformation system provides users with a fast and intuitive deformation experience. One important novelty of our method is we extract the contours automatically whereas existing techniques either ignore contour constraints [1] or sketch contours manually [9].

The significance of automatic contour extraction is considerably evident, especially when the features of two images are only the contours, like in Fig. 13(d). In this case, users do not need to sketch any curve, in that (i) our system automatically extracts the contour, and (ii) the source image and target image have no interior features. Thus, there is no tedious work for users and they can enjoy the entire natural automatic deformation process. In fact, even if there are some interior features inside the objects of interest, the improvement of our technique is still obvious as in such situations users only need to sketch several interior feature curves and do not need to manually sketch the contours.

In contrast, some existing techniques like [9] need to let users sketch the contour. Of course, such techniques can lead to the same result as our method. However, notice the fact that the user-sketching process has a complete control on the final result, which means the deformation completely follows the drawn outline. Hence, a successful deformation by such techniques requires users' skills and experiences, and also leads to tedious work for users. In principle, a good deformation tool should strive to minimize the manual effort.

TABLE 1. Average time comparison of the contour extraction step of our method with traditional methods.

type of contours	Our method (automatic extraction)	Traditional methods (by sketching)
simple shape	0.002 sec	1 min
complex shape	0.002 sec	2 min

Table 1 compares the average time used for automatic contour extraction and manually sketching contours. Observe that our automatic contour extraction saves a lot of time, no matter the contour is simple or complex. Hence, another advantage of our method is the automation of contour extraction relieves the cumbersome operations for users.

3) PROVIDING AN INTUITIVE USER-CONTROLLABLE INTERFACE

Some existing deformation methods like [31] lack a high-level control structure, making it inconvenient for users to deform images. To resolve this problem, we provide an intuitive user-controllable image deformation interface.

Our interface guides the user clearly and conveniently so that the user can control the deformation process. With our interface , users can load a source image and a target image, then automatically extract the contours of the objects of interest, subsequently sketch several interior features (if any) inside the contours and set corresponding points, then our system does triangulations and constructs a bijective mapping between the source object and the target object, and finally shows the morphing results.

This interface enables users to enjoy a convenient image deformation experience. It does not require tedious or complex operations and users just need to click two or three menu items. Through this interface, users can realize image deformation and watch the image deformation results immediately and intuitively.

4) ACHIEVING PLEASING MORPHING BETWEEN DIFFERENT OBJECTS

Traditional cage based methods such as [35]–[37] usually rely on a manually modeled cage. Some of these methods have the problem that the cage is hard to manipulate manually. However, our technique avoids such problems since our approach automatically constructs a mapping between the source object and the target object, which is invisible to users. Our deformation method can achieve pleasing morphing results between **different** objects (even if two objects have different interior features and colors) as shown in Fig. 11, Fig. 12, Fig. 13, and Fig. 14, in that it establishes a bijective mapping between the mesh of the source object and the mesh of the target object, whereas most existing cage-based deformation [36], [38] can only deform a source object into the very similar object with a somewhat different shape or pose but with the same color values. In other words, most existing cage-based deformation techniques can only somehow 'edit' a given image, but cannot morph an image into a new image with different colors as they cannot generate new colors. Therefore, given a source image and a target image, cage-based deformation techniques usually cannot deform the given source image into the given target image.

(a) cannot generate new color, so the target image cannot be deformed into

(b) the target image can be deformed into because new colors can be generated with our method

FIGURE 18. Comparison: results by traditional cage-based methods vs. results by our method. From left to right: Frame #0; Frame #10; Frame #20; Frame #30; Frame #40; Frame #49. In both (a) and (b), the source image and target image should be Frame #0 and Frame #49 of (b). Others are intermediate frames.

For example, a red object surely cannot be deformed into a green object if we use some existing traditional cage-based deformation techniques. Image deformation results such as the monkeys shown in Fig. 13(d) cannot be generated by traditional cage-based deformation. Another simple example is shown in Fig. 18, where a traditional cage-based method

can enlarge the image object but cannot generate new color, so that its deformation result, i.e., the last image shown in Fig. 18(a), has the same color as the source image, so the target image, i.e., the last image shown in Fig. 18(b), cannot be deformed into with this kind of traditional cage-based methods. In contrast, our deformation system can easily deform such kind of images.

Even if the source image and the target image have the same or similar colors, traditional cage-based deformation techniques still cannot transform the source image into the target image. Take the yellow duck shown in Fig. 13(a) as an example. The wing of the target yellow duck cannot be generated by traditional cage-based image deformation techniques although the source duck and the target duck have almost the same color. The reason is that traditional cage-based deformation can only change the shape, pose or size of the features but cannot introduce new pixel values. However, our deformation method can easily achieve such kind of morphing.

TABLE 2. Comparison of our method with other methods.

Our method	Traditional methods
extract contour **automatically**	extract contour by **sketch** [9] or **do not consider contours** [1]
more **natural** deformations	**fade-in and fade-out** results without regard to the shape [1]
relieve the cumbersome operation although automatic contour extraction cannot adapt to all images	**tedious** work for users although user-sketch can adapt to all images [9]
achieve **pleasing** morphing between different objects	**cannot** generate new colors; usually cannot deform a source image into a fully different target image [36], [38]
prove the bijectivity of our mesh mapping	do not provide a bijectivity proof [3], [17], [25]

C. SUMMARY AND DISCUSSION

After making comparison with existing deformation methods, we summarize the main characteristics of our method and other methods, and list the differences in Table 2. As our method automatically extracts contours of the object of interest, it reduces tedious work for users. As our method allows a user to specify inner features, it preserves inner feature details. As our method takes both the contours and inner features as constraints, it avoids fade-in and fade-out effects and thus can produce smooth deformation. As our method provides an intuitive interface, it provides a more convenient user experience. As our method establishes a mapping between the source and target meshes, it can achieve pleasing deformation between different objects.

VI. CONCLUSION

In this paper, we propose a 2D image deformation method based on guaranteed feature correspondence and mesh mapping. Our method first automatically extracts contours of two images and allows users to specify some inner features on both images, and then automatically constructs a guaranteed feature correspondence and applies triangulation on both images. Subsequently, our method automatically establishes a constrained mesh mapping between the source image mesh and target image mesh through harmonic maps. After calculating the intermediate triangular meshes and pixels of intermediate images, our method achieves a smooth and natural deformation process. Regarding our mesh mapping, we also prove its bijectivity and discuss its distortion metrics and other properties. Compared with traditional works, our method relieves the cumbersome operations for users by extracting image contours automatically, avoids face-in and fade-out effects by considering contour constraints, provides an intuitive user-controllable interface, and achieves pleasing morphing between different objects. Various deformation examples show that our image deformation method can deform images smoothly and naturally.

REFERENCES

[1] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 533–540, 2006.
[2] W.-C. Jhou and W.-H. Cheng, "Animating still landscape photographs through cloud motion creation," *IEEE Trans. Multimedia*, vol. 18, no. 1, pp. 4–13, Jan. 2016.
[3] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *Proc. ACM SIGGRAPH*, 2000, pp. 157–164.
[4] A. Tal and G. Elber, "Image morphing with feature preserving texture," *Comput. Graph. Forum*, vol. 18, no. 3, pp. 339–348, 1999.
[5] V. Surazhsky and C. Gotsman, "High quality compatible triangulations," *Eng. Comput.*, vol. 20, no. 2, pp. 147–156, 2004.
[6] H. Gupta and R. Wenger, "Constructing piecewise linear homeomorphisms of simple polygons," *J. Algorithms*, vol. 22, no. 1, pp. 142–157, 1997.
[7] W. V. Baxter, III, P. Barla, and K.-I. Anjyo, "Compatible embedding for 2D shape animation," *IEEE Trans. Vis. Comput. Graphics*, vol. 15, no. 5, pp. 867–879, Sep./Oct. 2009.
[8] M. Shapira and A. Rappoport, "Shape blending using the star-skeleton representation," *IEEE Comput. Graph. Appl.*, vol. 15, no. 2, pp. 44–50, Mar. 1995.
[9] Y. Liu, H. S. Seah, Y. He, J. Lin, and J. Xia, "Sketch based image deformation and editing with guaranteed feature correspondence," in *Proc. 10th Int. Conf. Virtual Reality Continuum Appl. Ind. (VRCAI)*, 2011, pp. 141–148.
[10] N. Arad and D. Reisfeld, "Image warping using few anchor points and radial functions," *Comput. Graph. Forum*, vol. 14, no. 1, pp. 35–46, 1995.
[11] T. Beier and S. Neely, "Feature-based image metamorphosis," in *Proc. ACM SIGGRAPH*, vol. 26, 1992, pp. 35–42.
[12] M. Eitz, O. Sorkine, and M. Alexa, "Sketch based image deformation," in *Proc. Vis., Modeling Vis.*, 2007, pp. 135–142.
[13] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin, "Synthesizing realistic facial expressions from photographs," in *Proc. ACM SIGGRAPH Courses*, 2006, Art. no. 19.
[14] J. Areeyapinan and P. Kanongchaiyos, "Face morphing using critical point filters," in *Proc. Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE)*, May 2012, pp. 283–288.
[15] S. G. Kong and R. O. Mbouna, "Head pose estimation from a 2D face image using 3D face morphing with depth parameters," *IEEE Trans. Image Process.*, vol. 24, no. 6, pp. 1801–1808, Jun. 2015.
[16] A. Verma, L. V. Subramaniam, N. Rajput, C. Neti, and T. A. Faruquie, "Animating expressive faces across languages," *IEEE Trans. Multimedia*, vol. 6, no. 6, pp. 791–800, Dec. 2004.
[17] S.-S. Lin, I.-C. Yeh, C.-H. Lin, and T.-Y. Lee, "Patch-based image warping for content-aware retargeting," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 359–368, Feb. 2013.
[18] K. H. Chan and R. W. H. Lau, "Contour-based warping," *Graph. Models Image Process.*, vol. 60, no. 5, pp. 331–348, 1998.
[19] D. Cohen-Or, D. Levin, and A. Solomovici, "Contour blending using warp-guided distance field interpolation," in *Proc. IEEE Vis.*, Oct./Nov. 1996, pp. 165–172.

[20] J. Jin, A. Wang, Y. Zhao, C. Lin, and B. Zeng, "Region-aware 3-D warping for DIBR," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 953–966, Jun. 2016.
[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2242–2251.
[22] A. Bansal, S. Ma, D. Ramanan, and Y. Sheikh. (2018). "Recycle-GAN: Unsupervised video retargeting." [Online]. Available: https://arxiv.org/abs/1808.05174
[23] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 6, pp. 567–585, Jun. 1989.
[24] A. Radford, L. Metz, and S. Chintala. (Jan. 2016). "Unsupervised representation learning with deep convolutional generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1511.06434
[25] G. Wolberg, "Image morphing: A survey," *Vis. Comput.*, vol. 14, nos. 8–9, pp. 360–372, Dec. 1998.
[26] A. A. Joshi, "Harmonic mappings between riemannian manifolds," M.S. thesis, Graduate School, Univ. Southern California, Los Angeles, CA, USA, 2006.
[27] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 257–266, 2002.
[28] J. R. Shewchuk. (2005). *Triangle: A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator*. [Online]. Available: http://www.cs.cmu.edu/%7Equake/triangle.html
[29] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler, "A local/global approach to mesh parameterization," *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1495–1504, 2008.
[30] O. Weber and C. Gotsman, "Controllable conformal maps for shape deformation and interpolation," *ACM Trans. Graph.*, vol. 29, no. 4, 2010, Art. no. 78.
[31] J. Smith and S. Schaefer, "Bijective parameterization with free boundaries," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 70:1–70:9, Jul. 2015.
[32] R. Shi *et al.*, "Hyperbolic harmonic mapping for surface registration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 5, pp. 965–980, May 2017.
[33] Y. Lipman, "Bounded distortion mapping spaces for triangular meshes," *ACM Trans. Graph.*, vol. 31, no. 4, 2012, Art. no. 108.
[34] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution analysis of arbitrary meshes," in *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Techn.*, 1995, pp. 173–182.
[35] B. Sheng, W. Meng, H. Sun, and E. Wu, "Sketch-based design for green geometry and image deformation," *Multimedia tools Appl.*, vol. 62, no. 3, pp. 581–599, 2013.
[36] Y. Lipman, D. Levin, and D. Cohen-Or, "Green coordinates," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 78:1–78:10, Aug. 2008.
[37] M. Ben-Chen, O. Weber, and C. Gotsman, "Variational harmonic maps for space deformation," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–11, 2009.
[38] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki, "Harmonic coordinates for character articulation," *ACM Trans. Graph.*, vol. 26, no. 3, Jul. 2007, Art. no. 71.

YAQIONG LIU received the bachelor's degree in computer science and technology and the bachelor's degree in financial management from Tianjin University, China, in 2009, and the Ph.D. degree in computer science and engineering from Nanyang Technological University, Singapore, in 2016. She is currently a Lecturer with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, China. Her research interests include networking and computing, spatial query processing, location-based services, mobile Internet and applications, and image animation. She is a member of the IEEE.

XIN LIN received the bachelor's degree in telecommunications management from the International School, Beijing University of Posts and Telecommunications, in 2017. She is currently pursuing the master's degree with The University of Manchester, U.K.

GUOCHU SHOU is currently a Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, China. His research interests include access network and edge computing, fiber and wireless network virtualization, network construction and routing, and mobile Internet and applications.

HOCK SOON SEAH is currently a Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include image animation and virtual/augmented reality.