COMPUTER GRAPHICS
forum

DOI: 10.1111/j.1467-8659.2009.01584.x

COMPUTER GRAPHICS forum
Volume 29 (2010), number 1 pp. 160–174

Constrained Texture Mapping using Image Warping

H. Seo[1,2] and F. Cordier[3]

[1]Department of Computer Science and Engineering, Chungnam National University, Korea
[2]LSIIT (UMR 7005 CNRS), Université de Strasbourg, France
[3]LMIA, Université de Haute Alsace, France

Abstract

We introduce in this paper a new method for smooth foldover-free warping of images. It allows users to specify the constraints in two different ways: positional constraints to constrain the position of points in the image and gradient constraints to constrain the orientation and scaling of some parts of the image. We then show how our method is used for texture mapping with hard constraints. We start with an unconstrained planar embedding of the target mesh calculated with conventional methods. In order to obtain a mapping that satisfies the user-defined constraints, we use our warping method to align the features of the texture image with those of the unconstrained embedding. Compared to previous work, our method generates a smoother texture mapping and offers higher level of control for defining the constraints.

Keywords: foldover, positional constraints, texture mapping, warping

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling; I.3.6 [Computer Graphics]: Methodology and Techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

Texture mapping is a well-known technique for mapping an image onto the surface of a 3D model to enhance its visual appearance. This technique has been adopted for a broad range of applications such as special effects for the film industry that requires highly realistic models as well as the game industry for efficiently creating 3D models and virtual characters. The essential step of texture mapping is the surface parameterization of a 3D model, that is, finding a one-to-one correspondence between the entire surface of the model and a texture. It is often required that the mapping satisfies a set of constraints specified in the form of correspondence between points in the texture and on the surface.

1.1. Related work

A large body of work on texture mapping has been devoted to global parameterization of surfaces, that is finding a bijective function between the entire surface of a model and a planar texture space. See [HLS07] for a survey. One may think that some of these works could be extended for constrained texture mapping. For instance, constrained texture mapping could be achieved with the method [DMA02] by removing the degrees of freedom of the constrained vertices from the objective function of the parameterization. Such approach may not work for a large set of constraints, because it does not guarantee to generate a one-to-one mapping.

[Lev01] have proposed algorithms that satisfy the constraints using least-squares minimization. Although this method works well for a small number of constraints, it may fail when the number of constraints becomes large. [GDHZ06] have proposed a method that satisfies the constraints exactly; however, their method does not guarantee to compute a bijective mapping.

Other researchers [ESG01, KSG03, ZWT*05, LYY08] have proposed methods that exactly satisfy the positional constraints by adding Steiner vertices to the embedded mesh. Unlike these approaches, our method does not require modifying the structure of the embedded mesh. In addition, these previous methods only handle positional constraints whereas

ours allows constraining the gradient of the mapping as well.

[TT09] have presented a texture mapping method that overcomes the texture image distortions that result from the viewpoint and the object's 3D geometry. The aim of their work is different from ours. Their technique does not perform parameterization, but rather projection of the model according to the estimated local cameras. They do not consider the issues of constrained parameterization like bijectivity.

Several researchers have worked on the embedding of a surface with constraints into a simpler domain such as the sphere. Alexa [Ale00] has proposed a method to compute a constrained spherical parameterization; the constraints are defined in the form of feature-point correspondence between the two surfaces. This method may fail if too many feature points are given. [LLCY00, LL05] have proposed to solve this problem using edge flipping. However, the disadvantage of this method is that it may modify the geometry of the surface. [PSS01] have proposed a method to compute a consistent parameterization of a set of genus-0 surfaces. This method is based on the segmentation of each of the surfaces into a set of patches. In some cases, it may produce badly shaped patches, resulting into a parameterization with high distortion. The Praun's approach has been further extended for the consistent parameterization of surfaces with the same genus [SAPH04, KS04].

[BBT02] used image warping for the texture mapping. The main difference with our method is that their warping method does not guarantee to generate a bijective mapping and the purpose of their work was to reduce the space allocated for the texture image.

1.2. Overview

In this paper, we present a new method for image warping which produces a foldover-free and C^1-continuous mapping. This method allows the user to control the warping in two different ways: either with the positional constraints or the gradient constraints to rotate and scale parts of the image.

We then show how this warping method can be used for constrained texture mapping. We first compute a planar embedding of the 3D mesh (Figure 1(c)). Free-boundary parameterization methods are preferred since they generate planar embedding with much lower distortion [LPRM02, DMA02, SLMB05]. We then use our warping method to deform the texture image in order to align its features with those of the planar embedding (Figure 1(d)). Compared to other constrained texture mapping methods, our approach presents all the following advantages together:

- Our method produces a better visual result, especially for texture mapping with constraints that introduces large deformation (see Section 3.2). This is because our image warping method is C^1 continuous.
- Our method offers more flexibility; it allows the user to modify the position, orientation and scaling of the texture image on the surface of the 3D model. In addition, the user is allowed to place the positional constraints anywhere on the surface of the mesh unlike existing techniques which require the positional constraints to be located at the vertices [ESG01, KSG03].

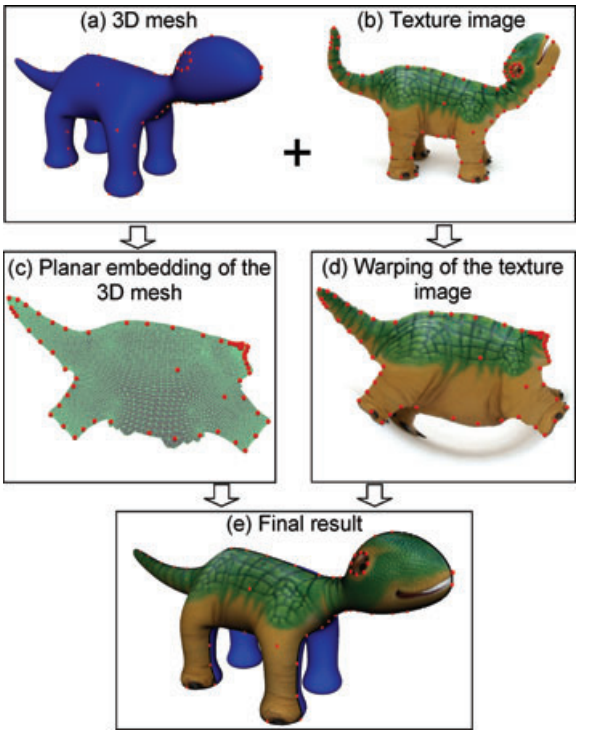

Figure 1: *Overview of the texturing method: 3D mesh (a) and texture (b), unconstrained planar embedding of the mesh (c), warping of the texture image to match the features with those of the unconstrained planar embedding (d), final textured model (e).*

The paper is organized as follows. Section 2 describes the foldover-free image warping technique. In Section 3, we give the implementation details, the results, and limitations of the method.

2. Foldover-Free Image Warping

2.1. Required properties of the image warping

The problem of image warping is to find a function f that maps points in the undeformed image to the deformed image. The mapping is constrained with a set of feature points

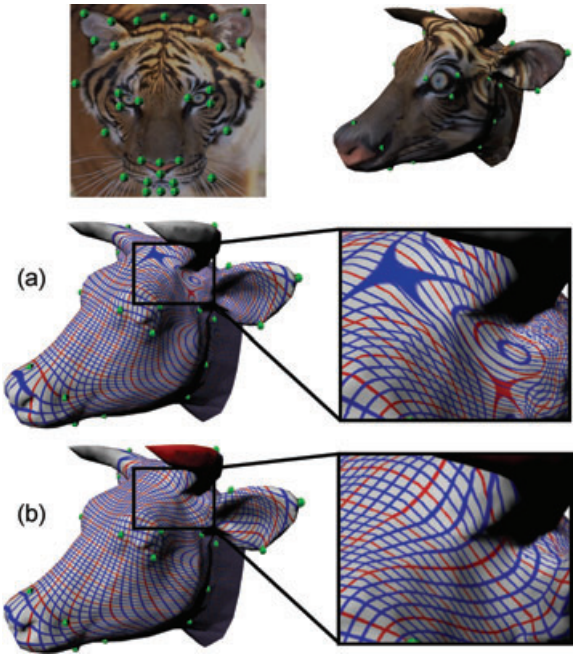

Figure 2: *Constrained texture mapping using the Radial Basis Function (a); the resulting mapping is not bijective and the texture shows a foldover. Constrained texture mapping using our foldover-free image warping (b).*

that are moved from their source position P_i to their target position Q_i. A warping method suitable for constrained texture mapping must satisfy the following four properties:

- **Smoothness:** f should produce smooth deformations; this property is desirable in order to generate visually pleasing texture mapping.
- **Bijectivity:** f should be bijective so that the constrained texture mapping is bijective. Figure 2 shows an example of texture mapping by a warping method that does not guarantee the bijectivity of the mapping.
- **Interpolation:** f should be an exact interpolant of the positional constraints, that is, the source positions should map to the target positions (i.e. $f(P_i) = Q_i$).
- **Identity:** f should satisfy the identity property. If the source and target positions of the feature points are coincident (i.e. $f(P_i) = P_i$), f should be the identity function.

2.2. Previous work in image warping

Techniques for image warping can be classified into two categories: techniques based on triangulation and those using a smooth function.

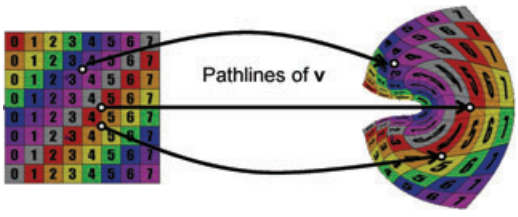

Figure 3: *Overview of the vector field based warping: the points of the image are moved along their respective pathline to reach their position in the deformed image.*

The main idea of triangulation-based warping [IMH05] is to compute the warping of the image by deforming a triangular mesh overlaid on the image; each triangle contains a small portion of the image. [FM98] have proposed a method based on a time-varying triangulation that provides a bijective mapping. [LYY08] have developed a similar approach for constrained texture mapping. Compared to our work, these approaches have two disadvantages. The mapping using triangulation is piecewise linear; it has only C^0 continuity. In addition, they only handle positional constraints; they do not provide a simple way to constrain the scaling and orientation of the image.

Several warping methods using a smooth function have been proposed [BN92, SMW06]. The most common method [Boo89] is based on the thin-plate splines; it attempts to minimize the amount of bending in the deformation. Most of them do not guarantee to generate a bijective mapping. Grid-based techniques such as free-form deformations [SP86, LCS95] use bivariate cubic splines to generate C^2 warping. These methods require the positional constraints to be arranged on a parallelepiped lattice, which makes them difficult to use for the constrained texture mapping; the positional constraints can have any position. [TDR01] have proposed a generic approach to transform a mapping with foldovers into a bijective mapping by analyzing the determinant of its Jacobian. The authors note that their method may not always work.

Compared to the previous work, our method offers the following features: it generates a mapping that is smooth and foldover free and which can be controlled with positional and gradient constraints. To the best of our knowledge, none of the existing techniques fulfills all these requirements together.

2.3. Computation of the warping

The driving idea is to compute the warping of the image using a time-dependant vector field. A similar technique has been used for 3D shape deformation [VTS06]. A C^1 continuous vector field $\mathbf{v}(x, y, t)$ is first constructed based on the user-specified constraints. We then compute the warped position of every point p_{src} of the image by applying a pathline integration of $\mathbf{v}(x, y, t)$ starting from p_{src} (Figure 3). The

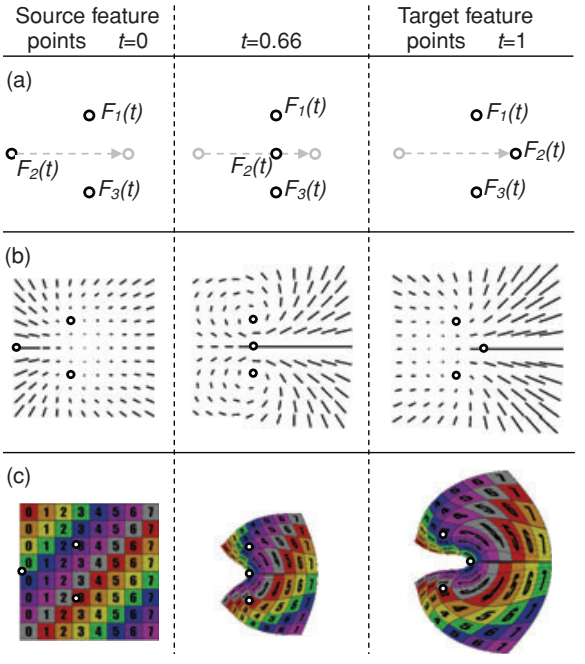

Figure 4: *Vector-field based warping: linear interpolation of the position of feature points from the source to the target positions (a), computation of the vector field of the velocity of the texture points (b), warping of the texture image (c).*

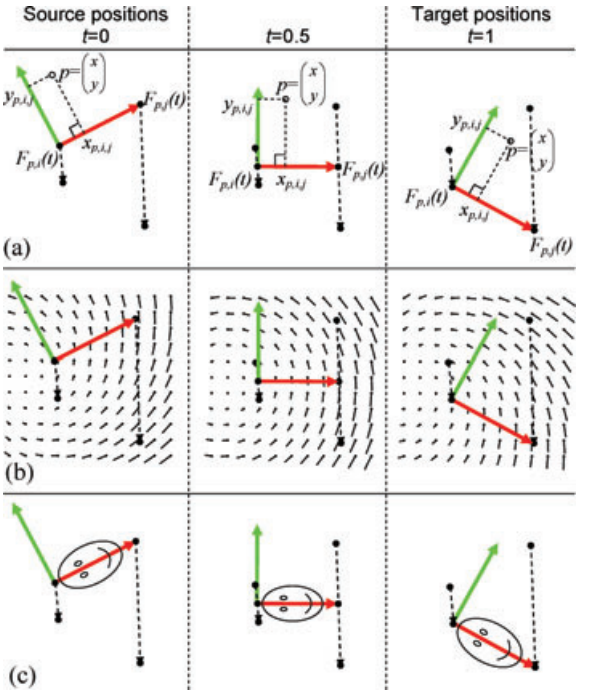

Figure 5: *Local coordinate frame defined by $F_i(t)$ and $F_j(t)$ (a), time-dependant vector field constructed from the local coordinate frame (b), warping of the image points computed with the vector field (c).*

pathline of the vector field starting from p_{src} is a 2D function $p(t)$ solution of the following set of equations:

$$\begin{cases} \dfrac{d}{dt}p(t) = \mathbf{v}(p(t), t) \quad \text{for } t \in [0, 1] \\ p(0) = p_{src} \end{cases} \tag{1}$$

The pathline starting from a point p_{src} is the trajectory that this point would make as it moves with the flow of the vector field; at any time, the velocity of the point is equal to the value of the vector field at the location of the point.

The user specifies the constraints with a set of feature points whose source position is P_i in the undeformed image and the target position Q_i in the image after warping. Our problem can be summarized as follow: given the source and target positions, we deform the image such that the source positions P_i map to the target positions Q_i.

The key idea is to smoothly move the feature points from their source position P_i to their target position Q_i (Figure 4(a)) and to relate the velocity of the points of the image to those of the feature points. Intuitively speaking, the 2D warping is equivalent to performing a sequence of 2D warping generated by incrementally moving the source points to the target positions.

The velocity of all the image points is defined as a vector field function $\mathbf{v}(x, y, t)$ (Figure 4(b)). Hereafter, we denote $F_{p,i}(t)$ and $\frac{dF_{p,i}(t)}{dt}$ the position and velocity of the feature point i respectively for $t \in [0, 1]$; $F_{p,i}(0)$ and $F_{p,i}(1)$ represent the source position P_i and target positions Q_i, respectively. The position $F_{p,i}(t)$ is obtained from a linear interpolation of $F_{p,i}(0)$ and $F_{p,i}(1)$. Since the mapping must be bijective, the feature points should not violate the bijectivity property as well, that is, the feature points should not share the same position at the same time. For all the feature points, we have $F_{p,i}(t) \neq F_{p,j}(t)$ for all i and j with $i \neq j$ and $t \in [0, 1]$.

2.3.1. *Vector field with positional constraints*

One important feature of image warping is the possibility to rotate and uniformly scale parts of the image using positional constraints. Similarly to [BN92], we compute the warping using pairs of positional constraints. Given two positional constraints with source and target positions, we compute a 2D similarity transformation (transformation that only includes translation, rotation, and uniform scaling) to warp the image.

Given two feature points i and j and a point p with coordinates x and y in the image, we compute the relative coordinates $x_{p,i,j}$ and $y_{p,i,j}$ of p in the local coordinate frame defined by $F_{p,i}(t)$ and $F_{p,j}(t)$ at time t (Figure 5):

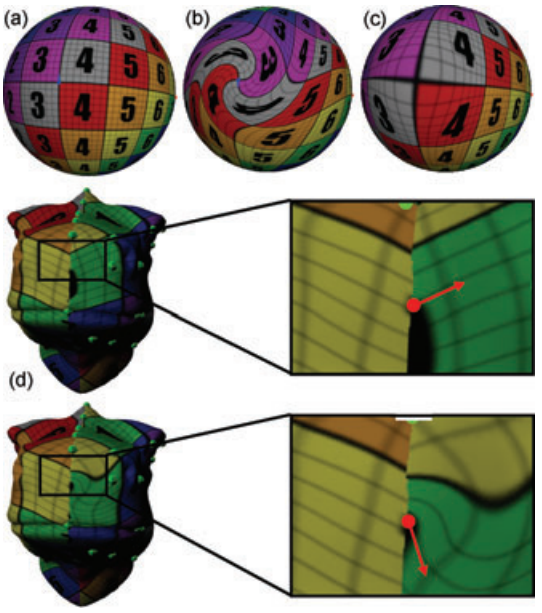

Figure 6: *Texture mapping with no constraint (a); constraining the orientation of the texture (b); constraining the scaling of the texture (c). Gradient constraints are useful to modify the orientation of the texture along the texture seam (d).*

$$p = F_{p,i}(t) + x_{p,i,j} \cdot (F_{p,j}(t) - F_{p,i}(t)) \text{ with } R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} + y_{p,i,j} \cdot R_{90}(F_{p,j}(t) - F_{p,i}(t)) \quad (2)$$

Note that the local coordinate frame is defined only if $F_{p,i}(t)$ and $F_{p,j}(t)$ do not have the same position. Given the velocity of the pair of feature points i and j, Equation (2) provides the corresponding velocity of the point $p = (x, y)^T$. The vector field function $\mathbf{v}_{i,j}(x, y, t)$ that relates the velocity of the point $p = (x, y)^T$ with those of the feature points i and j is given by

$$\mathbf{v}_{p,i,j}(x, y, t) = \frac{d}{dt}F_{p,i}(t) + x_{p,i,j} \cdot \frac{d}{dt}(F_{p,j}(t) - F_{p,i}(t)) + y_{p,i,j} \cdot R_{90}\frac{d}{dt}(F_{p,j}(t) - F_{p,i}(t)). \quad (3)$$

This vector field defines the velocity of all the points of the image in a way that these points are moved according to the similarity transformation defined by the velocity of $F_{p,i}(t)$ and $F_{p,j}(t)$. In other words, if the deformed position of a point p is computed with the vector field in Equation (3), the position of p with respect to the local coordinate frame defined by $F_{p,i}(t)$ and $F_{p,j}(t)$ remains unchanged (Figure 5(a)–(c)).

As stated in Section 2.1, our warping method should satisfy several properties in order to be suitable for constrained texture mapping. First, we can easily show that the warping method based on the vector field produces smooth deformations. The Equation (3) is twice continuous differentiable with respect to x and y; therefore the vector field is C^1 continuous.

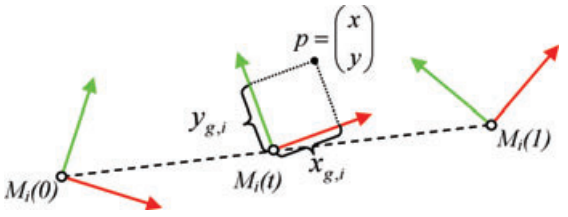

Figure 7: *p in the local coordinate frame defined by $M_i(t)$.*

Second, we show that the warping method maps the source positions to the target positions. If $p = (x, y)^T$ is coincident with $F_{p,i}(t)$, the relative coordinates $x_{p,i,j}$ and $y_{p,i,j}$ are both equal to 0. Equation (3) becomes $\mathbf{v}_{p,i,j}(x, y, t) = \frac{d}{dt}F_{p,i}(t)$. The velocity of p is the same as that of $F_{p,i}(t)$; this means that the point p maps to $F_{p,i}(1)$. Similarly, if p is coincident with $F_{p,j}(t)$, the vector field becomes $\mathbf{v}_{p,i,j}(x, y, t) = \frac{d}{dt}F_{p,j}(t)$ and the velocity of p is the same as that of $F_{p,j}(t)$.

Third, the warping method satisfies the identity property. If the source and target positions of the feature points i and j are coincident, $\frac{d}{dt}F_{p,i}(t)$ and $\frac{d}{dt}(F_{p,j}(t) - F_{p,i}(t))$ are both equal to 0 and the vector field is null. No deformation is applied to the image.

2.3.2. *Vector field with gradient constraints*

We provide an additional way to control the texture mapping process. As shown in Figure 6, the user can modify the orientation and size of the texture around a point on the surface.

Feature points for constraining the gradient are identical to other feature points, except that the users can modify the orientation and scaling of the texture image in addition to the position. These gradient constraints are useful to remove the texture discontinuity along the texture seam. Placing positional constraints along seam only eliminates the C^0 discontinuity. Using the gradient constraints, the artist is able to modify the orientation of the texture and thus make the texture appear C^1 continuous.

Each feature point i is then defined by a transformation matrix composed of a translation, a rotation and an uniform scaling components; $M_i(0)$ being the matrix corresponding to the texture image (source matrix) and $M_i(1)$ the matrix for the planar embedding of the mesh (target matrix). The idea is to interpolate the matrix $M_i(t)$ from the source to the target matrices and to compute a velocity on p such that p does not move in the local coordinate frame $M_i(t)$ (Figure 7). We first decompose the matrix $M_i(0)^{-1}\ M_i(1)$ into translation,

rotation and scaling:

$$M_i(0)^{-1}M_i(1) = \begin{pmatrix} 0 & 0 & T_x \\ 0 & 0 & T_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} S & 0 & 0 \\ 0 & S & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$M_i(t)$ is obtained by interpolating each component of the matrix: $M_i(t) = M_i(0) \cdot T(t)R(t)S(t)$ with

$$T(t) = \begin{pmatrix} 0 & 0 & tT_x \\ 0 & 0 & tT_y \\ 0 & 0 & 1 \end{pmatrix}, \; R(t) = \begin{pmatrix} \cos(t\phi) & -\sin(t\phi) & 0 \\ \sin(t\phi) & \cos(t\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and

$$S(t) = \begin{pmatrix} S^t & 0 & 0 \\ 0 & S^t & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The vector field corresponding to the gradient constraint is

$$\mathbf{v}_{g,i}(x, y, t) = \frac{d}{dt}\left(M_i(t) \cdot \begin{pmatrix} x_{g,i} \\ y_{g,i} \\ 1 \end{pmatrix}\right) \tag{4}$$

$(x_{g,i}, y_{g,i})^T$ is the position of the point $p = (x, y)^T$ in the local coordinate frame of $M_i(t)$. We denote $F_{g,i}(t) = M_i(t) \cdot (0, 0, 1)^T$, the origin of the coordinate system defined by $M_i(t)$ in the global coordinate system.

Similarly to the vector field in Equation (3), we check if the warping based on the vector field in Equation (4) has the properties listed in Section 2.1. First, we can easily see that the warping produces smooth deformation. Equation (4) is twice continuous differentiable with respect to x and y; therefore the vector field is C^1 continuous.

Second, the warping method maps the source positions to the target positions. If a point $p = (x, y)^T$ is coincident with the feature point $F_{g,i}(t)$, $x_{g,i}$ et $y_{g,i}$ are both equal to 0 in Equation (4). The vector field is equal to the translation part of the matrix $M_i(t)$; the point p maps to the target position of the feature point $F_{g,i}(t)$.

Third, the warping method satisfies the identity property. If the source matrix $M_i(0)$ and the target matrix $M_i(1)$ are equal, $M_i(t)$ is constant with respect to t and the vector field $\mathbf{v}_{g,i}(x, y, t)$ is null. The image does not undergo any deformation.

2.3.3. *Combination of vector fields*

Because the number of feature points in the texture image is usually more than two, the position of a point is affected by multiple feature points. The total velocity of the point $p = (x, y)^T$ is expressed as a weighted combination of the velocity associated with each feature point. We define a weight $\beta_{p,i,j}(p, t)$ for each pair of positional constraints i and j so that the influence of the feature point pair $F_{p,i}(t)$ and $F_{p,j}(t)$ increases as the point p becomes closer to either of the feature points.

$$\beta_{p,i,j}(p, t) = \frac{1}{d_{p,i}^{\alpha} d_{p,j}^{\alpha}} \text{ with } d_{p,i} = \|p - F_{p,i}(t)\| \text{ and } d_{p,j} = \|p - F_{p,j}(t)\|$$

The weight for a gradient constraint i is given by

$$\beta_{g,i}(p, t) = \frac{1}{d_{g,i}^{\alpha}} \text{ with } d_{g,i} = \|p - F_{g,i}(t)\|.$$

The exponent α is set by the user to control the smoothness of the warping (see Figure 8). These weights are normalized such that their sum is unity for each point $p = (x, y)^T$. Let n_{pos} be the number of positional feature points i and n_{grad} the number of gradient feature points, we compute $w_{p,i,j}(p, t)$ the normalized value of $\beta_{p,i,j}(p, t)$:

$$w_{p,i,j}(p, t) = \frac{d_{p,i}^{-\alpha} d_{p,j}^{-\alpha}}{\left(\sum_{\substack{l=1,m=1 \\ l\neq m}}^{n_{pos}} \left(d_{p,l}^{-\alpha} d_{p,m}^{-\alpha}\right) + \sum_{l=1}^{n_{grad}} d_{g,l}^{-\alpha}\right)}$$

$w_{g,i}(p, t)$, which is the normalized value of is $\beta_{g,i}(p, t)$ given by

$$\alpha_{g,i}(p, t) = \frac{d_{g,i}^{-\alpha}}{\left(\sum_{\substack{l=1,m=1 \\ l\neq m}}^{n_{pos}} \left(d_{p,l}^{-\alpha} d_{p,m}^{-\alpha}\right) + \sum_{l=1}^{n_{grad}} d_{g,l}^{-\alpha}\right)}.$$

These weights are rewritten as follows:

$$w_{p,i,j}(p, t) = \frac{\prod_{l=1,l\neq i,l\neq j}^{n_{pos}} d_{p,l}^{\alpha} \cdot \prod_{l=1}^{n_{grad}} d_{g,l}^{\alpha}}{\sum_{\substack{l=1 \\ m=1 \\ l\neq m}}^{n_{pos}} \left(\sum_{\substack{n=1 \\ n\neq l \\ n\neq m}}^{n_{pos}} d_{p,n}^{\alpha} \prod_{n=1}^{n_{grad}} d_{g,n}^{\alpha}\right) + \sum_{l=1}^{n_{grad}} \left(\sum_{\substack{n=1 \\ n\neq l}}^{n_{grad}} d_{g,n}^{\alpha} \prod_{n=1}^{n_{pos}} d_{p,n}^{\alpha}\right)}. \tag{5}$$

$$w_{g,i}(p, t) = \frac{\prod_{l=1,l\neq i}^{n_{grad}} d_{g,l}^{\alpha} \cdot \prod_{l=1}^{n_{pos}} d_{p,l}^{\alpha}}{\sum_{\substack{l=1 \\ m=1 \\ l\neq m}}^{n_{pos}} \left(\sum_{\substack{n=1 \\ n\neq l \\ n\neq m}}^{n_{pos}} d_{p,n}^{\alpha} \prod_{n=1}^{n_{grad}} d_{g,n}^{\alpha}\right) + \sum_{l=1}^{n_{grad}} \left(\sum_{\substack{n=1 \\ n\neq l}}^{n_{grad}} d_{g,n}^{\alpha} \prod_{n=1}^{n_{pos}} d_{p,n}^{\alpha}\right)}. \tag{6}$$

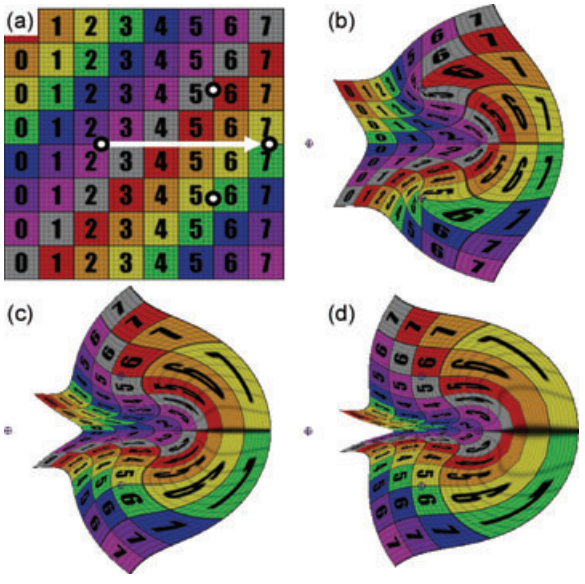

Figure 8: *Warping with different values of a equal to 1.0 (b), 2.0 (c) and 3.0 (d).*

As stated in Section 2.3, we assume that no feature points share the same position at the same time; we have $F_i(t) \neq F_j(t)$ for all feature points i and j with $i \neq j$ and $t \in [0, 1]$. This implies that the denominators of Equations (5) and (6) are always different from zero and the weights are defined for all coordinates (x, y) of p, even when p shares the same location as one of the feature points.

We provide a numerical example to show the values of $w_{p,i,j}(p, t)$ when the position of p is the same as one the feature points. Let be $F_{p,1}(t)$, $F_{p,2}(t)$ and $F_{p,3}(t)$ the position of three positional feature points and p having the same position as $F_{p,1}(t)$; the three weights for the three pairs of positional feature points are

$$w_{p,1,2} = \frac{d^{\alpha}_{p,3}}{d^{\alpha}_{p,1} + d^{\alpha}_{p,2} + d^{\alpha}_{p,3}} = \frac{d^{\alpha}_{p,3}}{d^{\alpha}_{p,2} + d^{\alpha}_{p,3}}$$

$$w_{p,1,3} = \frac{d^{\alpha}_{p,2}}{d^{\alpha}_{p,1} + d^{\alpha}_{p,2} + d^{\alpha}_{p,3}} = \frac{d^{\alpha}_{p,2}}{d^{\alpha}_{p,2} + d^{\alpha}_{p,3}}$$

$$w_{p,2,3} = \frac{d^{\alpha}_{p,1}}{d^{\alpha}_{p,1} + d^{\alpha}_{p,2} + d^{\alpha}_{p,3}} = 0.$$

These values shows that the position of p is solely defined by the two pairs of feature points $(F_{p,1}(t),\ F_{p,2}(t))$ and $(F_{p,1}(t),\ F_{p,3}(t))$. These two pairs are those connected to the feature point $F_{p,1}(t)$.

We also show that $w_{p,i,j}(p, t)$ and $w_{g,i}(p, t)$ are C^1 continuous. $d^{\alpha}_{p,i} = \|p - F_{p,i}(t)\|^{\alpha}$ and $d^{\alpha}_{p,j} = \|p - F_j(t)\|^{\alpha}$ are C^1 continuous with respect to the coordinates x and y of p for $\alpha > 1$. The denominator and numerators of Equations (5) and (6) are C^1 continuous with respect to x and y; it follows that $w_{p,i,j}(p, t)$ and $w_{g,i}(p, t)$ are C^1 continuous as well.

The total vector field is given by

$$\mathbf{v}(x, y, t) = \sum_{i,j=1 \text{ with } i \neq j}^{n_{pos}} \left(w_{p,i,j}(x, y, t) \cdot \mathbf{v}_{p,i,j}(x, y, t)\right) + \sum_{i=1}^{n_{grad}} \left(w_{g,i}(x, y, t) \cdot \mathbf{v}_{g,i}(x, y, t)\right). \quad (7)$$

The warped position of a point p_{src} is obtained by solving Equation (1) for t equal to 1 and $p(0)$ equal to p_{src}. In our implementation, we use the 4^{th} order Runge–Kutta integration with adaptive step-size (similarly to [VTS06]). One integration step involves computing the intermediate solution $p(t)$, updating the weights $w_{p,i,j}$ and $w_{g,i}$ and the local coordinates $x_{p,i,j}$, $y_{p,i,j}$, $x_{g,i}$ and $y_{g,i}$ in Equations (3) and (4) of the vector fields $\mathbf{v}_{p,i,j}$ and $\mathbf{v}_{g,i}$ by using $p(t)$. Note that the pathline integration is computed independently for each point of the texture image. Our algorithm is parallelized by assigning the computation of the pathline integration to different threads.

Each vector field $\mathbf{v}_{p,i,j}(x, y, t)$ and $\mathbf{v}_{g,i}(x, y, t)$ corresponds to a similarity transformation of the image. It follows that the deformation induced by the vector field $\mathbf{v}(x, y, t)$ is a weighted combination of the similarity transformations corresponding to each vector field $\mathbf{v}_{p,i,j}(x, y, t)$ and $\mathbf{v}_{g,i}(x, y, t)$.

2.3.4. *Properties of the vector-field based warping*

We now show that our warping method based on the vector field in Equation (7) has the four properties listed in Section 2.1.

Bijectivity: The mapping generated with our warping method is bijective (foldover-free). We demonstrate that the vector field is locally Lipschitz-continuous and show that this property is sufficient to guarantee the bijectivity [Lan95].

Theorem 1: The vector field $\mathbf{v} : \mathbb{R}^2 \to \mathbb{R}^2$ is called locally Lipschitz-continuous around each point of $\mathbb{R}^2$ if there exists $K > 0$ and $L > 0$ such that

$$\|\mathbf{v}(p_1) - \mathbf{v}(p_2)\| < K\|p_1 - p_2\|,\ \forall p_1 \in \mathbb{R}^2,$$
$$p_2 \in \mathbb{R}^2 : \|p_1 - p_2\| < L.$$

Because the vector field $\mathbf{v}(p, t)$ is C^1 continuous for $\forall p \in \mathbb{R}^2$, it is also locally Lipschitz-continuous.

Theorem 2: Let $\mathbf{v} : \mathbb{R}^2 \to \mathbb{R}^2$ be a continuous vector field satisfying the Lipschitz condition around each point of $\mathbb{R}^2$.

Then, there exists a unique pathline through any $p_0 \in \mathbb{R}^2$. Furthermore, every pathline is defined over $\mathbb{R}$.

This theorem implies that two pathlines cannot intersect in the 3D space-time domain. This property guarantees the bijectivity of the mapping.

Smoothness: Our warping method produces a smooth deformation. The vector fields defined in Equations (3) and (4) are both C^1 continuous with respect to x and y. It follows that $\mathbf{v}(x, y, t)$ in Equation (7), which is the weighted summation of these two vector fields is C^1 continuous in x and y. The warping based on the vector field $\mathbf{v}(x, y, t)$ is smooth.

Interpolation: Our warping method maps the source positions to the target positions; this is because the warping based on the two vector fields $\mathbf{v}_{p,i,j}(x, y, t)$ and $\mathbf{v}_{g,i}(x, y, t)$ map the source positions to the target positions (see Sections 2.3.1 and 2.3.2).

Identity: Our warping method satisfies the identity properties. If the source and target positions of the feature points are the same ($F_{p,i}(0) = F_{p,i}(1)$ for all feature points i), the vector fields in Equation (3) and (4) are null. No deformation is applied to the image.

3. Results and Limitations

Our method has been implemented with a Graphical User Interface that allows the user to add/remove feature points interactively. The texture mapping process includes the following steps: first, the user draws a simple closed curve on the triangular mesh to specify the region of the model to texture. A planar embedding of the region is then computed using the Least Squares Conformal Map method [LPRM02]. Next the user places the feature points in the texture image and the 3D model. During the placement of feature points, the constrained texture mapping can be calculated and displayed at any time. Our method is demonstrated with several examples of varying complexity, as shown in Figure 9 and the demonstration video. The complexity of the models ranges between 5 000 and 20 000 polygons and the number of feature points between 10 and 50.

3.1. Computation time

Given n positional constraints, the number of pairs of positional constraints is given by

$$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}.$$

Thus, the complexity of the algorithm is quadratic in the number of positional constraints. The computation time for mapping a texture image with 50 positional constraints is about few seconds on an Intel dual 2.6 GHz computer. This number of positional constraints is usually sufficient for most cases of texture mapping. We have measured the computation time for larger number of positional constraints. For 100 and 200 positional constraints, the algorithm takes about 10 seconds and 1 minute, respectively.

3.2. Discussion with related work

To the best of our knowledge, the methods proposed by [LYY08] and [KSG03] are the only ones that guarantee the bijectivity of the mapping. Compared to [KSG03] method, the method by [LYY08] is able to handle more challenging constraints and generate more pleasing results. Therefore, we compare our algorithm with that of [LYY08].

We have first compared the two methods with a 2D grid mesh composed of a small number of triangles. As can be seen in the Figure 10, the warping computed by [LYY08] has a better special distribution of the vertices; the variation of the edge lengths is smaller. This indicates that the method [LYY08] produces a mapping with a lower distortion for the stretch. This result is not surprising since their method is based on an optimization problem that minimizes the stretch of mapping. This is verified with a quantitative analysis. To measure the distortion of the parameterization, we used two metrics L^2 Stretch and L^2 Shear that measure the stretch and the conformality, respectively. These two metrics are computed for each triangle using the affine mapping defined with its 3D coordinates and its 2D coordinates in the texture space. The formula of the L^2 Stretch and the L^2 Shear are given by [SSGH01] and [SLMB05] respectively. The value of the L^2 stretch for [LYY08] is slightly better than the one for the vector-field based warping.

We have then compared the two methods with models of higher complexity. The Figure 9 shows the results of our approach and those obtained by [LYY08]. All the models have been textured with a checker image to show the difference of the mapping between the two methods. Our algorithm generates a smoother texture mapping; in particular, smoothness around the constraints is not ensured with [LYY08]. The difference between the two methods is particularly visible for the pig model (d) and the two cow models (f) and (g).

A quantitative analysis has been performed as well to compare our approach with that of [LYY08]. The values of the distortion metrics in Table 1 show that there is no significant difference in the quality of the parameterization between the two approaches, except for the models (d), (f) and (g).

The method by [LYY08] is based on the explicit minimization of a deformation energy in order to control the distortion. The associated optimization problem is nonlinear; the convergence is slow and usually the method finds an approximated solution, which is not the exact minimum of the optimization problem. This explains why the method of [LYY08] works well for meshes with small number of triangles (models (b) and (e) in Figure 9): the convergence is fast and a good approximation of the solution is found.

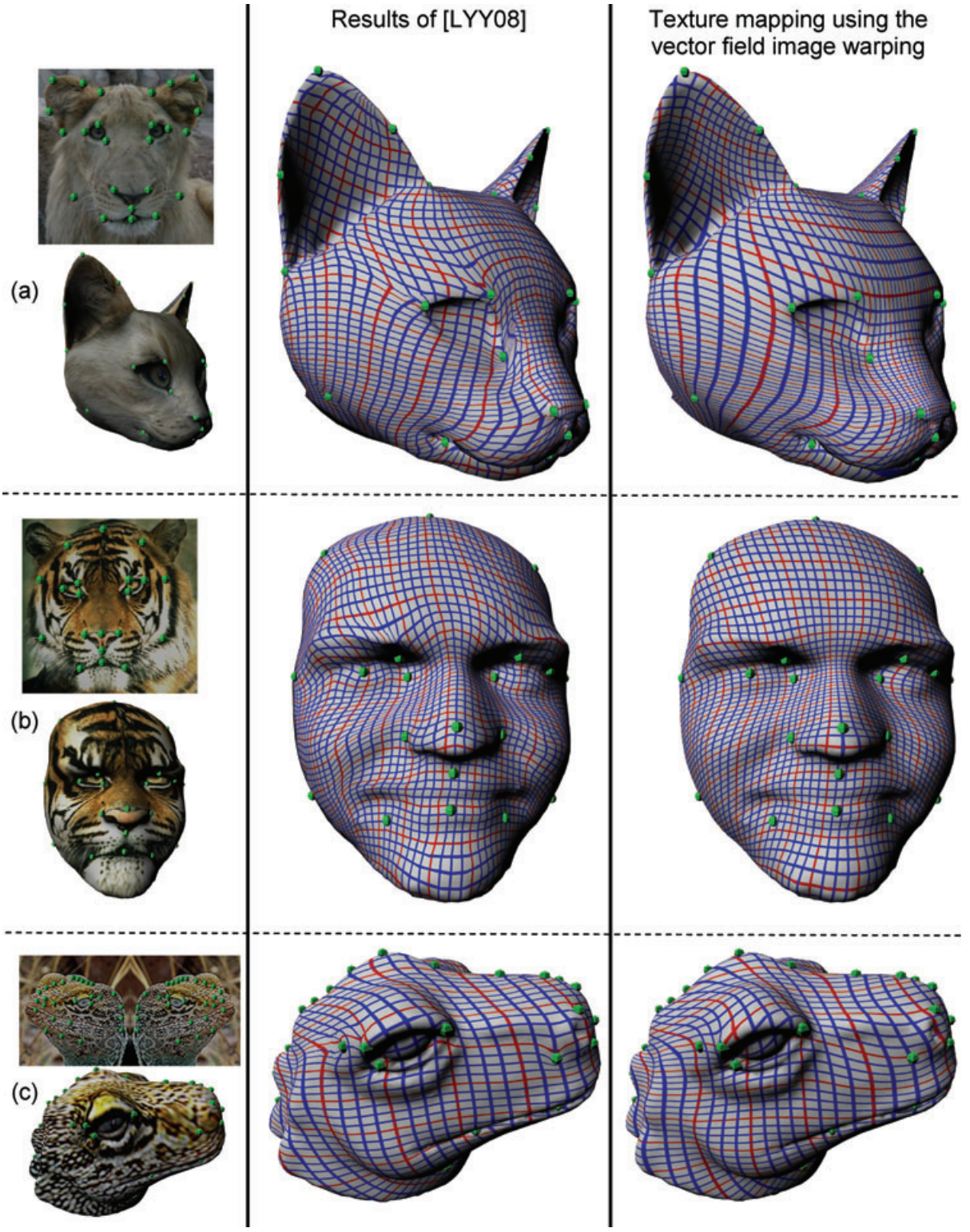

Figure 9a: *Comparison of our method with [LYY08] (zoom in for detail).*

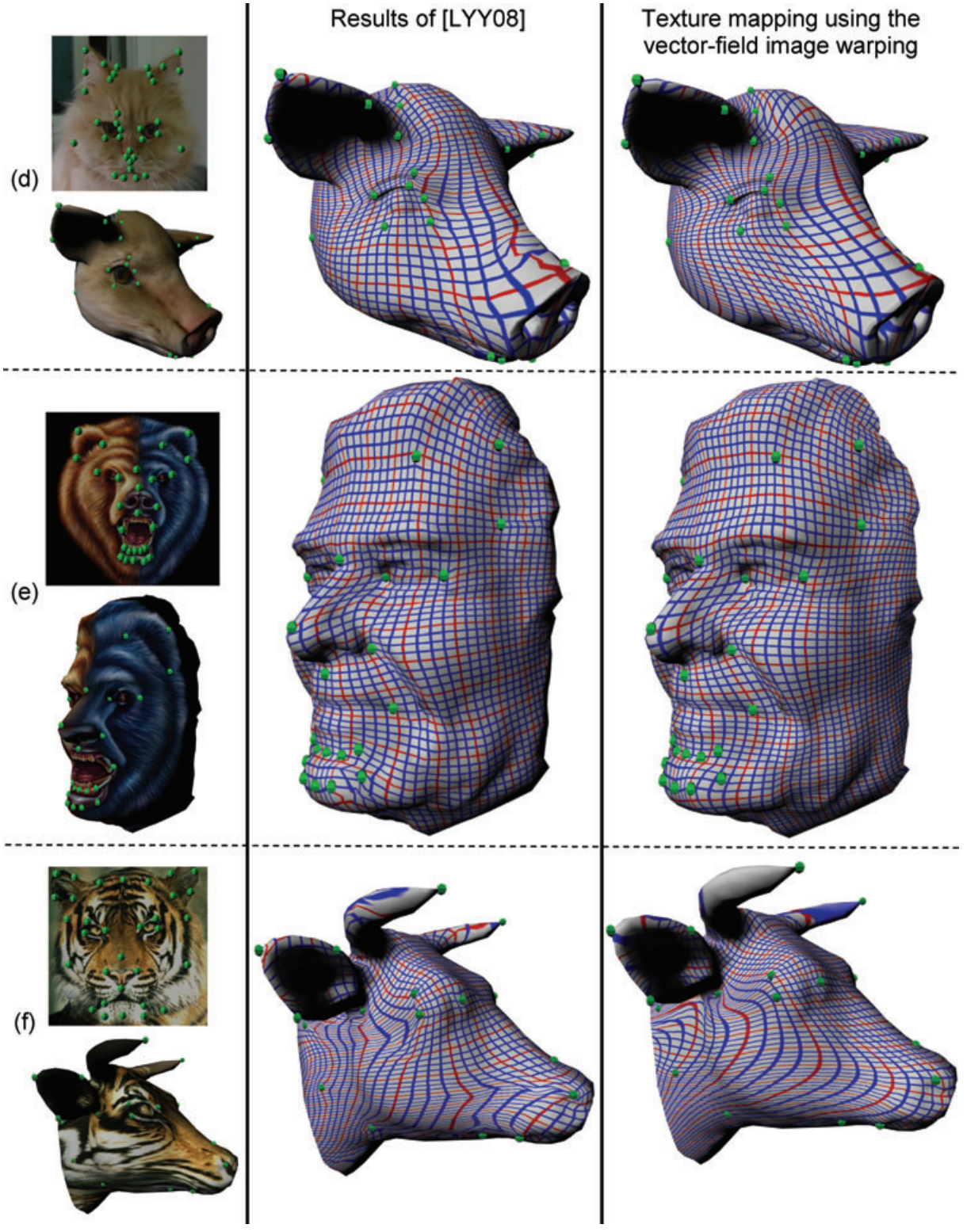

Figure 9b: *Comparison of our method with [LYY08] (zoom in for detail).*

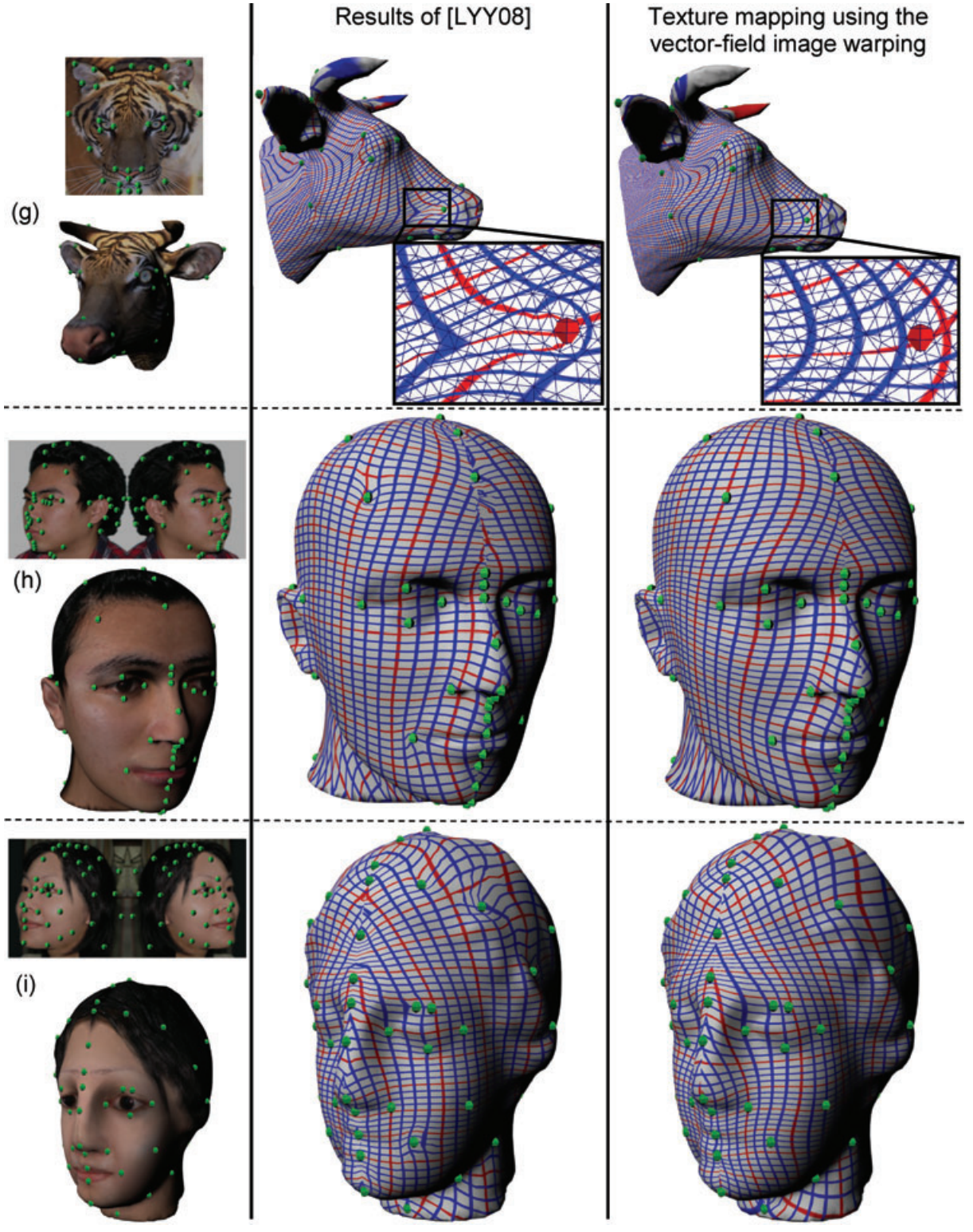

Figure 9c: *Comparison of our method with [LYY08] (zoom in for detail).*

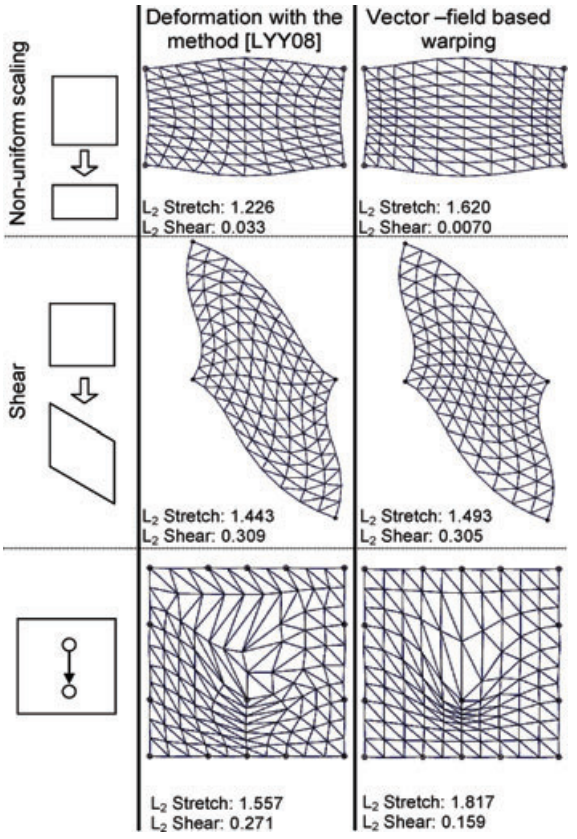

Figure 10: *Comparison of the vector-field based warping with the warping method by [LYY08].*

Table 1: *Quantitative comparison between quality of the texture mapping for the models shown in Figure 9.*

	[LYY08]		Our method	
	L^2 Stretch	L^2 Shear	L^2 Stretch	L^2 Shear
Model (a)	7.692	0.247	8.204	0.248
Model (b)	1.451	0.104	1.509	0.159
Model (c)	3.397	0.210	1.771	0.163
Model (d)	8.024	0.306	6.695	0.288
Model (e)	1.219	0.059	1.285	0.055
Model (f)	183.647	0.310	30.364	0.360
Model (g)	66.468	0.342	10.359	0.290
Model (h)	2.839	0.222	2.910	0.296
Model (i)	2.940	0.168	2.986	0.228

In the contrary, if the mesh has a large number of triangles with large distortion in the mapping (models (f) and (g)), the convergence is much slower and the remaining distortion is important. In contrary, our method does not require the post-processing step to lower the mapping distortion and therefore performs faster than the method of [LYY08].

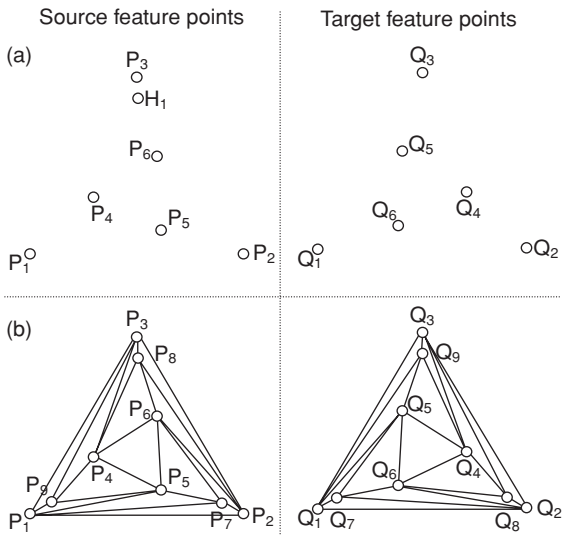

Figure 11: *Let two sets of feature points* $S_1 = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ *and* $S_2 = \{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6\}$ *(a); the construction of the compatible triangulation of* S_1 *and* S_2 *requires adding 3 Steiner vertices* $\{P_7, P_8, P_9\}$ *and* $\{Q_7, Q_8, Q_9\}$ *for each set.*

Another limitation of the method of [LYY08] is that it is sensitive to the quality of the mesh. The presence of badly shaped triangles introduces numerical errors in the optimization. This is illustrated by the pig model (d) in Figure 9 that contains many badly shaped triangles located on the nose. Since our vector-field warping does not depend on the quality of the triangulation, it produces better result.

3.3. Limitations

The warping method does not work when two feature points become coincident during the warping process. There are two reasons for this. First, the coordinate frame to compute the vector field in Equation (3) cannot be defined if $F_i(t)$ and $F_j(t)$ share the same position for the same value t (See Section 2.3.1). Second, we assume that $F_i(t) \neq F_j(t)$ for all i and j with $i \neq j$ and $t \in [0, 1]$. This assumption is required for the bijectivity property of the feature points.

The solution to this problem is to compute the feature-points trajectories such that they do not intersect in the 3D space-time domain. We have implemented a simple method that greatly reduces the possibility that feature points self-intersect. The idea is to compute a similarity transformation to align the source feature points as close as possible to the target feature points. [Ume91] proposes a method to compute this transformation with a least squares method that minimizes the square distances between the source and target feature points. Although this method does not guarantee the non-intersection of the feature-points trajectories, it has

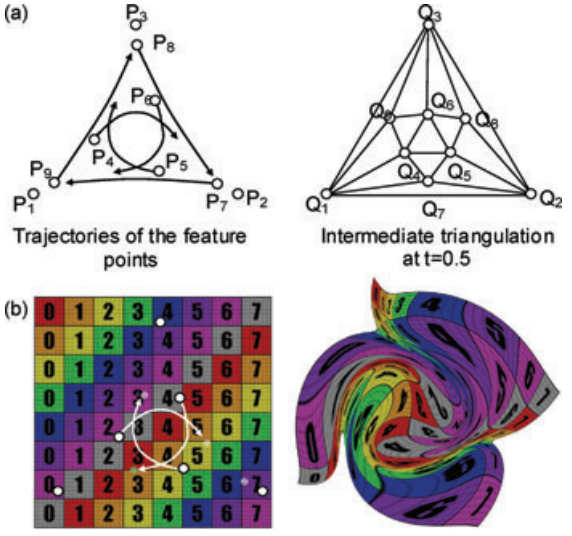

Figure 12: *Given two compatible triangulations of the source and target feature points, the method proposed by [SG01] computes the trajectories of the feature points (a); the trajectories are then used to compute the warping (b).*

been successfully used for all the textured models shown in this paper. In the next paragraph, we propose another method that always guarantees to generate non-intersecting trajectories for the feature points.

The problem of computing the trajectories of a set of points from their source position to their target position is a well-known problem in morphing and has been already solved. The computation of these trajectories is done two steps. We first compute a compatible triangulation of the source and target feature points. Let S_1 and S_2 be two finite sets of points in the Euclidean plane. Two triangulations T_1 of S_1 and T_2 of S_2 are called compatible if the face lattices formed by their triangles, edges, and points are isomorphic. The decision problem determining whether two sets of point are compatible is believed to be NP-hard [Saa87, SW94] demonstrated that two sequences of n points may be made compatible by adding $O(n^2)$ Steiner points. Figure 11 shows a compatible triangulation of two point sets with Steiner points.

Once a compatible triangulation of the two sets has been found, we compute the morphing between these two triangulated sets using the method proposed by [SG01] (Figure 12). Their technique, which is based on a convex representation of triangulations, takes as input a compatible triangulation with the same boundary and generates a valid morph in the form of a continuous sequence of valid triangulations, that is, the triangles do not intersect during the morph. The computed trajectories are then used to compute the vector field for the image warping. Note that the trajectories of the Steiner points that have been created for the compatible triangulation are not taken into account for the computation of the vector field.

Another limitation of our method is that the constrained mapping is not necessarily bijective. This is because we compute the unconstrained planar embedding of the 3D mesh using the Least Square Conformal Mapping [Lev01]; this method does not guarantee to generate a bijective mapping.

Like existing constrained texture mapping methods, our method may produce undesirable visual distortion that arises when the mapping is highly distorted and the resolution of the texture image is to low. The method proposed by [TBTS08] could be used to address this problem. Their approach is to expand image regions via texture synthesis to better fit the 3D geometry.

4. Conclusion

We have proposed a method for constrained texture-mapping of triangular meshes. Compared to previous work, our method allows more flexibility for manipulating the texture and provides a smoother mapping. As a future work, we plan to investigate other ways to construct the vector field using functions such as the Radial Basis Function.

Acknowledgment

We thank Tong-Yee Lee and Shao-Wei Yen for providing the models and helping us for the experimental comparison. We also thank Nicolas Chevallier for his helpful criticism and comments. This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) grant (No. R11-2007-028-02002-0) and in part by the Korea Research Foundation grant (D00401). The first author has been partly supported by the CNRS (Centre National de la Recherche Scientifique).

References

[Ale00] ALEXA M.: Merging Polyhedral Shapes with Scattered Features. *The Visual Computer 16*, (2000), 26–37.

[BBT02] BALMELLI L., BERNARDINI F., TAUBIN G.: Space-optimized texture maps. In *Computer Graphics Forum 21*, 3 (2002). G. Drettakis and H.-P. Seidel (Eds.), 411–420. (Proc. Eurographics'07).

[BN92] BEIER T., NEELY S.: Feature-based image metamorphosis. In *SIGGRAPH '92: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1992), ACM Press, pp. 35–42.

[Boo89] BOOKSTEIN F. L.: Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell. 11*, 6 (1989), 567–585.

[DMA02] Desbrun M., Meyer M., Alliez P.: Intrinsic Parameterizations of Surface Meshes. *Computer Graphics Forum 21*, 3 (2002), 209–218.

[ESG01] Eckstein I., Surazhsky V., Gotsman C.: Texture mapping with hard constraints. *Computer Graphics Forum (Proc. of Eurographics) 20*, 3 (2001), 95–104.

[FM98] Fujimura K., Makarov M.: Foldover-free image warping. *Graphical Models and Image Processing 60*, 2 (1998), 100–111.

[GDHZ06] Gingold Y. I., Davidson P. L., Han J. Y., Zorin D.: A Direct Texture Placement and Editing Interface. In *Proceedings of UIST*, Montreux, Switzerland, October 2006, pp. 23–32.

[HLS07] Hormann K., Lévy B., Sheffer A.: Mesh parameterization: theory and practice. In *ACM SIGGRAPH Courses* (San Diego, California, August 5–9, 2007). SIGGRAPH '07. ACM, New York, NY.

[IMH05] Igarashi T., Moscovich T., Hughes J. F.: As rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (2005), 1134–1141.

[KSG03] Kraevoy V., Sheffer A., Gotsman C.: Matchmaker: constructing constrained texture maps. In *Proceedings of SIGGRAPH* (2003), pp. 326–333.

[KS04] Kraevoy V., Sheffer A.: Cross-Parameterization and Compatible Remeshing of 3D Models. *ACM Trans. Graphics 23*, 3 (2004), 861–869.

[Lan95] Lang S.: *Differential and Riemannian manifolds, 3rd Edition*. Springer, New York, 1995.

[LCS95] Lee S.-Y., Chwa K.-Y., Shin S. Y.: Image metamorphosis using snakes and free-form deformations. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1995), ACM Press, pp. 439–448.

[LYY08] Lee T.-Y., Yen S.-W., Yeh I.-C.: Texture Mapping with Hard Constraints Using Warping Scheme. *IEEE Transactions on Visualization and Computer Graphics 14*, 2 (March–April 2008), 382–395.

[LPRM02] Levy B., Petitjean S., Ray N., Maillot J.-L.: Least squares conformal maps for automatic texture atlas generation. In *Proceedings of SIGGRAPH* (2002), pp. 362–371.

[Lev01] Levy B.: Constrained texture mapping for polygonal meshes. In *Proceedings of SIGGRAPH* (2001), pp. 417–424.

[LLCY00] Lin C.-H., Lee T.-Y., Chu H.-K., Yao Z.-Y.: Progressive Mesh Metamorphosis. *Journal of Computer Animation and Virtual Worlds 16*, 3–4 (2005), 487–498.

[LL05] Lin C.-H., Lee T.-Y.: Metamorphosis of 3D Polyhedral Models Using Progressive Connectivity Transformations. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (January/February 2005), 2–12.

[PSS01] Praun E., Sweldens W., Schorder P.: Consistent Mesh Parameterizations. In *Proc. ACM SIGGRAPH '01* (2001), pp. 179–184.

[Saa87] Saalfeld A.: Joint triangulations and triangulation maps. In *Proceedings of the 3rd Annual ACM Symposium on Computational Geometry* (Waterloo, Canada, 1987), pp. 195–204.

[SSGH01] Sander P. V., Snyder J., Gortler S. J., Hoppe H.: Texture Mapping Progressive Meshes. In *Proc. ACM SIGGRAPH '01* (2001), pp. 409–416.

[SMW06] Schaefer S., McPhail T., Warren J.: Image Deformation Using Moving Least Squares. In *Proceedings of ACM SIGGRAPH* (2006), pp. 533–540.

[SAPH04] Schreiner J., Asirvatham A., Praun E., Hoppe H.: Inter-Surface Mapping. *ACM Trans. Graphics 23*, 3 (2004), 870–877.

[SP86] Sederberg T. W., Parray S. R.: Free-form deformation of solid geometric models. In *Proceedings of ACM SIGGRAPH* (1986), ACM Press, pp. 151–160.

[SLMB05] Sheffer A., Levy B., Mogilnitsky M., Bogomyakov A.: ABF++: fast and robust angle based flattening. *ACM Transactions on Graphics 24*, 2 (2005), 311–330.

[SW94] Souvaine D. L., Wenger R.: Constructing Piecewise Linear Homeomorphisms. *Tech. Rep.* (Princeton, NJ, 1994), DIMACS, pp. 94–52.

[SG01] Surazhsky V., Gotsman C.: Controllable Morphing of Compatible Planar Triangulations. *ACM Transactions and Graphics 20*, 4 (2001), 203–231.

[TBTS08] Tai Y.-W., Brown M., Tang C.-K., Shum H.-Y.: Texture amendment: reducing texture distortion in constrained parameterization. *ACM Transactions on Graphics 27*, 5 (2008), 1–6.

[TDR01] Tiddeman B., Dufy N., Rabey G.: A general method for overlap control in image warping. *Computers and Graphics 25*, 1 (2001), 59–66.

[TT09] Tzur Y., Tal A.: FlexiStickers: photogrammetric texture mapping using casual images. *ACM Transactions on Graphics 28*, 3 (2009), 1–10.

[Ume91] Umeyama S.: Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 4 (April 1991), 376–380.

[VTS06] Von Funck W., Theisel H., Seidel H.-P.: Vector field based shape deformations. *ACM Trans. Graph. 25*, 3 (2006), 1118–1125.

[ZWT*05] Zhou K., Wang X., Tong Y., Desbrun M., Guo B., Shum H.-Y.: TextureMontage: Seamless Texturing of Arbitrary Surfaces From Multiple Images. In *Proceedings of SIGGRAPH* (2005), pp. 1148–1155.