Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Journal of Computational and Applied Mathematics 195 (2006) 83–92
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

Moving meshes by the deformation method

Guojun Liao*,[1], Jiaxing Xue

Department of Mathematics, University of Texas, Arlington, TX 76019-0408, USA

Received 15 August 2004; received in revised form 14 April 2005

Abstract

In this paper, we describe the deformation method for generation of moving meshes and its applications in computational fluid dynamics (CFD), and potential applications for image representation, segmentation, and landmark matching for image registration. We demonstrate that the deformation method can generate meshes of prescribed cell area. Numerical examples are presented to confirm that.

Keywords: Hypersonic flows; Image segmentation and registration; Landmark match

1. Introduction

A mesh consists of a set of nodes, a set of edges between nodes, a set of faces formed by edges, and a set of elements bounded by the faces. Computational meshes are needed in numerical simulation of partial differential equations by finite difference, finite volume, or finite element methods. The most common meshes on a 2D domain used for finite element methods are formed by mutually disjoint triangles or quadrilaterals whose union covers the domain. In 3D, tetrahedral or hexahedral elements are used instead.

Moving meshes are generated by repositioning the nodes according to boundary movement and the variations of physical fields. They are alternative to local refinement methods which add nodes to the regions where they are needed and remove nodes where they are not needed. Moving meshes are very useful in scientific computing such as fluid flow field simulation and fluid–structure interaction. They are becoming more and more popular in the emerging field of computational biology and computational medicine. They are especially suitable for problems in cardiology modeling and medical image registration with potential applications in image-guided surgical and radiological intervention systems.

In moving meshes on domains with moving boundary, the interior nodes are moved according to the movement of the boundary nodes with the connectivity of the mesh kept unchanged. The meshes may also be updated at each time step according to the features of the physical variables in order to provide sufficient resolution in regions where large variations occur.

* Corresponding author.
E-mail addresses: liao@uta.edu (G. Liao), joshjx@yahoo.com (J. Xue).
[1] Supported by NSF of United States under the grant of DMS 0310492 in computational mathematics. The opinions in this paper do not represent NSF.

0377-0427/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2005.07.022

A fundamental problem in moving mesh methods is to prevent element inversion. For instance, a triangular element is inverted if a vertex crosses its opposing side. This will occur if the vertex in question moves "faster" than the opposing edge. A quadrilateral element is inverted if any triangle that is contained in the element becomes inverted. Moreover, it is well known that poor quality elements affect the stability, convergence, and accuracy of numerical solvers. If the moving mesh method does not generate well-shaped elements, one may need to improve the mesh quality through topological or geometrical operations in a post-processing step.

This paper is inspired by the works [1,12] presented in the International Meshing Roundtable. Baker at Princeton [1] developed a three-step method for the metamorphosis (deforming, warping) of tetrahedral meshes. The first step is to move the interior nodes as far as possible using discrete Laplacian or elastic deformation while avoiding inversion. The second step is to remove the poorly shaped elements in the mesh. The final step is the addition of new elements to improve the mesh quality. Mesh examples of a beating heart are generated by the method. One disadvantage of this approach is that there is no theoretical guarantee on mesh convexity.

Vavasis et al. at Cornell [12] proposed a moving mesh method for problems where connectivity of the mesh is not allowed to change (for instance, electricity field coupled with mechanical movement of the heart). Their method is based upon linear weighted Laplacian smoothing. The first step in the algorithm is to generate a set of local weights for each interior node that represent the relative distances of the node to each of its neighbors. They use an interior point method from nonlinear programming in order to generate these weights. Next they apply a transformation to the boundary nodes. Using new positions of the boundary nodes and the weights from the original mesh, they solve a system of linear equations for the new positions of interior nodes. They proved a theorem, which gives general sufficient conditions for a mesh to avoid inversion by a transformation that is known to be injective. But they were unable to prove such a theorem specifically for their method. Instead, a conjecture is given. Also, the theorem assumes that the transformation is injective without proving that their method indeed guarantees injective property (one-to-one correspondence).

In this paper, we describe the deformation method for generation of moving meshes and its applications in computational fluid dynamics (CFD), and potential applications for image representation, segmentation, and landmark matching for image registration. The method is based on controlling the Jacobian determinant directly and precisely. It is proved mathematically that the Jacobian determinant of the transformation generated by the deformation method is equal to any (properly normalized, positive) function $f(x, y, z, t)$ prescribed by user. This fact implies that the continuum transformation is injective, and thus, meshes of sufficiently small elements would not be inverted. The deformation method is, to our best knowledge, the only method that can assure the one-to-one correspondence property in 3D.

A 2D version of our method is described in [2]. The current paper has included a 3D example for the first time (See Example 2).

2. The deformation method

The deformation method generates a time-dependent nodal transformation with prescribed Jacobian determinant. A positive monitor function $f(x, y, z, t)$ is used to obtain a vector field that moves the nodes to desired locations so that the element size of the resulting moving mesh is equal to f at each time. Such a monitor function is usually determined according to the solution error or the boundary curvature. The nodal transformation is calculated by the following steps:

Step 1: Calculate the node velocity field $\mathbf{v}$ from the following div–curl system:

$$\operatorname{div}\left(\frac{\mathbf{v}(x, y, z, t)}{f(x, y, z, t)}\right) = -\frac{\partial}{\partial t}\left(\frac{1}{f(x, y, z, t)}\right), \quad \operatorname{curl}\left(\frac{\mathbf{v}(x, y, z, t)}{f(x, y, z, t)}\right) = 0$$

with Neumann boundary condition on the fixed portions of the boundary, and the Dirichlet condition on the moving portions of the boundary.

Note: $\mathbf{v}$ can be specified at (interior) landmark points, curves, or surfaces in the interior of the domain for the purpose of landmark matching.

Step 2: Calculate the new location $T(x, y, z, t)$ of a node by integrating the velocity vector field $\mathbf{v}$: $\mathrm{d}T/\mathrm{d}t = \mathbf{v}(T, t)$ (referred to as the deformation ODE).

The main property of the transformation is provided in the following theorem, which can be proved either by showing $(\mathrm{d}/\mathrm{d}t)\ (J/f) = 0$ directly or by the well known Transport Formula from continuum mechanics as outlined below.

Theorem. *These two steps guarantee that* $J(T)(x, y, z, t) = f(T(x, y, z, t), t)$ *for each* $t > 0$, *provided that* $J(T)(x, y, z, 0) = f(T(x, y, z, 0), 0)$.

The Jacobian determinant has two significant geometric meanings: (1) It changes sign when the transformation maps a tetrahedron to another with reversed orientation (inversion). (2) Its absolute value indicates the change of a volume element by the transformation. More precisely, let $T(x, y, z, t) = (u(x, y, z, t), v(x, y, z, t), w(x, y, z, t))$. Let A be a small tetrahedron in the initial mesh and let $T(A)$ be its image under T. By the definition, we have $J(T) = D(u, v, w)/D(x, y, z) = \lim(\text{volume}(T(A))/\text{volume}(A))$. Thus, the theorem implies that $\text{volume}(T(A))/\text{volume}(A) \approx f$ (if A is small). This is why the deformation method can control cell area (or volume, in 3D) according to the monitor function f. Moreover, since the monitor function f is strictly positive, the fact that $J(T) = f$ implies that $T(A)$ and A have the same orientation, and thus A will not be inverted, provided that A is sufficiently small.

Outline of proof of the theorem. The main technical lemma used in the proof is the following well known formula for the time derivative of determinant (see [4,7]). For completeness and simplicity, we include a 2×2 version and work out some detail for non-experts. More precisely, in two dimensions, let $M(t) = (m_{ij}(t))$ be a 2×2 matrix with entry m_{ij}. Suppose $\mathrm{d}M/\mathrm{d}t = AM$, where $A = (a_{ij})$ is a 2×2 matrix. The lemma claims that $\mathrm{d}/\mathrm{d}t(\det M) = (a_{11} + a_{22})(\det M)$. To see this, we begin with

$$\begin{aligned}\frac{\mathrm{d}(\det M)}{\mathrm{d}t} &= \frac{\mathrm{d}}{\mathrm{d}t}(m_{11}m_{22} - m_{21}m_{12})\\ &= \frac{\mathrm{d}(m_{11})}{\mathrm{d}t}m_{22} + m_{11}\frac{\mathrm{d}(m_{22})}{\mathrm{d}t} - \frac{\mathrm{d}(m_{21})}{\mathrm{d}t}m_{12} - m_{21}\frac{\mathrm{d}(m_{12})}{\mathrm{d}t}.\end{aligned}$$

Next, from $\mathrm{d}M/\mathrm{d}t = AM$, we get

$$\begin{aligned}\frac{\mathrm{d}(m_{11})}{\mathrm{d}t} &= a_{11}m_{11} + a_{12}m_{21},\\ \frac{\mathrm{d}(m_{12})}{\mathrm{d}t} &= a_{11}m_{12} + a_{12}m_{22},\\ \frac{\mathrm{d}(m_{21})}{\mathrm{d}t} &= a_{21}m_{11} + a_{22}m_{21},\\ \frac{\mathrm{d}(m_{22})}{\mathrm{d}t} &= a_{21}m_{12} + a_{22}m_{22}.\end{aligned}$$

Using these formulas, we have

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}(\det(M)) &= m_{22}\frac{\mathrm{d}(m_{11})}{\mathrm{d}t} - m_{21}\frac{\mathrm{d}(m_{12})}{\mathrm{d}t} + m_{11}\frac{\mathrm{d}(m_{22})}{\mathrm{d}t} - m_{12}\frac{\mathrm{d}(m_{21})}{\mathrm{d}t}\\ &= m_{22}(a_{11}m_{11} + a_{12}m_{21}) - m_{21}(a_{11}m_{12} + a_{12}m_{22})\\ &\quad + m_{11}(a_{21}m_{12} + a_{22}m_{22}) - m_{12}(a_{21}m_{11} + a_{22}m_{21})\\ &= (a_{11} + a_{22})(m_{11}m_{22} - m_{21}m_{12}) = (a_{11} + a_{22})(\det M),\end{aligned}$$

as claimed. In general, the lemma can be written as

$$(\mathrm{d}/\mathrm{d}t)\det(M) = \mathrm{trace}(A)(\det M),$$

where $\mathrm{trace}(A) =$ the sum of the main diagonal entries of A. Below, we will present two approaches, both of which are based on the lemma.

Approach 1: Directly show $(\mathrm{d}/\mathrm{d}t)(J(T)/f(T, t)) = 0$ for each t.

To prove the theorem, we let $M = \nabla T$ in the lemma, where T is the transformation found by the deformation method. Note that, by interchanging the order of $\mathrm{d}/\mathrm{d}t$ and ∇, we have now $\mathrm{d}M/\mathrm{d}t = \nabla(\mathrm{d}T/\mathrm{d}t) = (\nabla \mathbf{v})(\nabla T) = AM$,

where $A = \nabla \mathbf{v}$, by the Chain Rule and the deformation ODE. Letting $J(T) = \det(\nabla T)$, the above lemma says that $\mathrm{d}J/\mathrm{d}t = (a_{11} + a_{22} + a_{33})J = (\nabla \cdot \mathbf{v})J$. By the Chain Rule and the equation $\nabla \cdot (v/f) = -(\partial/\partial t)(1/f)$, it follows, denoting $1/f = h$, that

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{J(T)}{f(T,t)}\right) &= \frac{\mathrm{d}}{\mathrm{d}t}(hJ) = h\frac{\mathrm{d}J}{\mathrm{d}t} + J\frac{\mathrm{d}h}{\mathrm{d}t} \\
&= hJ(\nabla \cdot \mathbf{v}) + J\left(\frac{\partial h}{\partial t} + (\nabla h) \cdot \frac{\mathrm{d}T}{\mathrm{d}t}\right) \\
&= hJ(\nabla \cdot \mathbf{v}) + J\left(\frac{\partial h}{\partial t} + (\nabla h) \cdot \mathbf{v}\right) \\
&= hJ(\nabla \cdot (f\mathbf{v}/f)) + J(-\nabla \cdot (\mathbf{v}/f) + (\nabla h) \cdot \mathbf{v}) \\
&= hJ(\nabla f)(\mathbf{v}h) + hJf(\nabla \cdot (\mathbf{v}/f)) \\
&\quad + hJ(-f\nabla \cdot (\mathbf{v}/f) + (\nabla h)f\mathbf{v}) \\
&= hJ((\nabla f)h + (\nabla h)f)\mathbf{v} \\
&= hJ\nabla(fh)\mathbf{v} \\
&= 0.
\end{aligned}$$

In the last step, we use the fact that $fh = 1$, and hence $\nabla(fh) = 0$. This assures that $J(T)/f(T,t) =$ constant, and thus $J/f = 1$ at each $t > 0$ if it is true at $t = 0$.

Approach 2: Use the Transport Formula from continuum mechanics.

Let $\Omega(t)$ be the image of an arbitrary domain $\Omega(0)$ in the space under a transformation $T(x, y, z, t)$. Let $\mathbf{v}(x, y, z, t)$ be the velocity of the particle which is at (x, y, z) at time t. Then, according to the Lagrange description, we have $\mathrm{d}T(x, y, z, t)/\mathrm{d}t = \mathbf{v}(T(x, y, z, t), t)$. Let h be any function defined in the space, then

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} h &= \int_{\Omega(0)}\left(\frac{\partial h}{\partial t} + (\nabla h) \cdot \mathbf{v} + h\nabla \cdot \mathbf{v}\right) \\
&= \int_{\Omega(0)}\left(\frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{v})\right).
\end{aligned}$$

This is the Transport Formula, which can be derived from the lemma shown above (see, for example, the textbook [7]). Now we choose $h = 1/f$. By the deformation method, we have

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{v}) = 0.$$

Thus $(\mathrm{d}/\mathrm{d}t)(\int_{\Omega(t)} h) = 0$ for each t, since the integrant in the right-hand side of the Transport Formula is zero. By a change of variable, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} h = \frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(0)} hJ(T) = \int_{\Omega(0)} \frac{\mathrm{d}}{\mathrm{d}t}(hJ(T)) = 0 \quad \text{for each } t.$$

Since $\Omega(0)$ is arbitrary, the integrant must be zero, i.e.

$$\frac{\mathrm{d}}{\mathrm{d}t}(hJ(T)) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{J(T)}{f}\right) = 0 \quad \text{for each } t, \text{ as desired.}$$

We want to point out that the shape of the elements is not directly controlled in the method. Taking $\mathbf{g} = 0$ in $\operatorname{curl} \mathbf{v} = \mathbf{g}$ will help in this regard, since now the velocity vector field $\mathbf{v}$ is irrotational (i.e. $\operatorname{curl} \mathbf{v} = 0$). In numerical

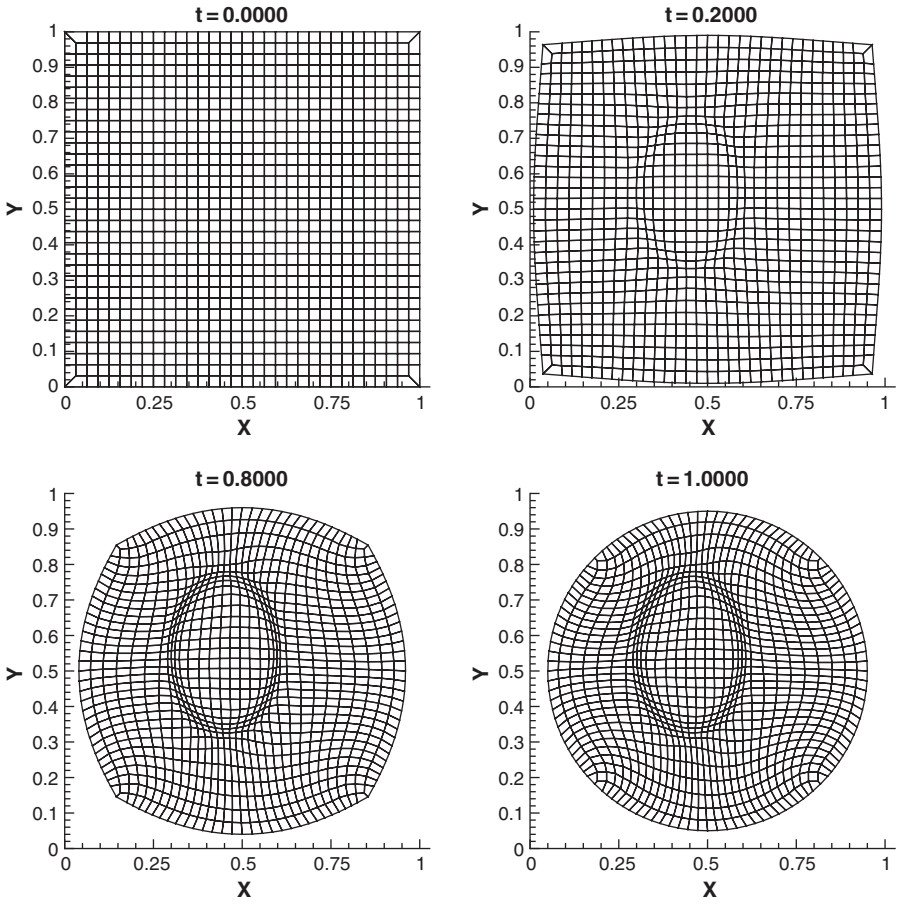

Fig. 1. From square to circle.

examples the method usually generates well-shaped elements with properly constructed monitor function f. In some cases, optimization steps may be necessary to improve the mesh quality.

Numerical implementation: The current version of the deformation method is implemented by the least square finite element method. The div–curl system is solved for the velocity vector field $\mathbf{v}$. On the fixed portions of the boundary, the Neumann condition is imposed to ensure that points on the boundary remain on it. On the moving portions of the boundary, the Dirichlet boundary condition is imposed, which is determined by the movement of the boundary nodes.

Since the method is based on a finite element method, it can be applied to structured, unstructured, and hybrid meshes. The least square finite element method is a powerful numerical tool, which can be used to solve different types of differential equations in a unified framework. Since it is based on minimizing the residuals, a natural monitor function can be readily constructed without complicated posteriori error estimates required in the traditional Galerkin finite element method. Thus the least square finite element method equipped with the moving mesh method has the potential to become a powerful numerical tool for scientific and engineering problems. We refer to [3,5,8,9] for more information about the least square finite element method.

We use the Euler method to solve for T from the ODE in step 2. More advanced methods, such as Runge–Kutta method, can also be used. For more details of the numerical implementation, see [2].

Example 1 (*From square to circle*). A square is deformed to a circle and a uniform mesh in the square is adapted according to the boundary movement and is refined around an ellipse (Fig. 1).

Example 2 (*Moving top boundary of a cube*). As seen in Fig. 2, the uniform mesh in a cube is deformed to follow the movement of the top surface.

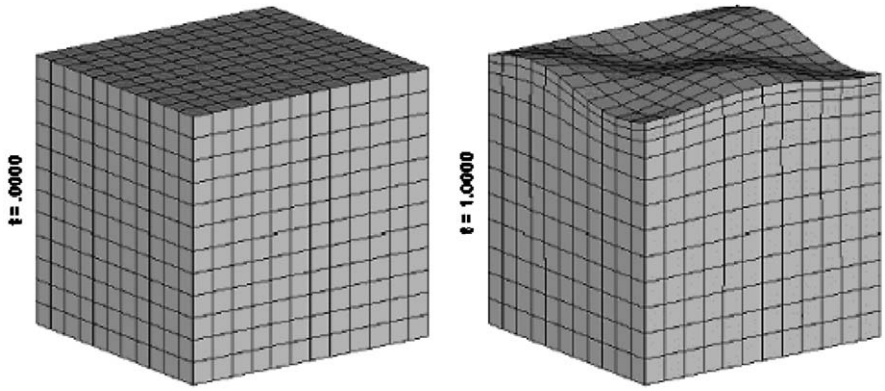

Fig. 2. The 3D mesh in the cube adapts to the movement of the top surface.

3. Applications in computational fluid dynamics

The deformation method has been applied to calculation of transonic and supersonic steady Euler flows around an airfoil with shock waves [11]. It is used also in unsteady flow problems. In [10], the method is used to generate a moving grid that remains refined around the propagating circular shocks in a 2D implosion problem.

Shock–shock interaction in hypersonic flow is also simulated on an adaptive mesh refined around multiple interacting shocks. This is a new result from joint work with Dr. Lei of Japan Aerospace Exploration Agency.

The region in the front of a hypersonic aircraft is covered with an initial uniform mesh of 41×81. After an initial calculation, the initial mesh is adapted according to the gradient of the Mach number. The adapted mesh and the Mach number contour based on the new mesh are shown in Fig. 3. Note that the mesh is dense in the regions where the shocks occur.

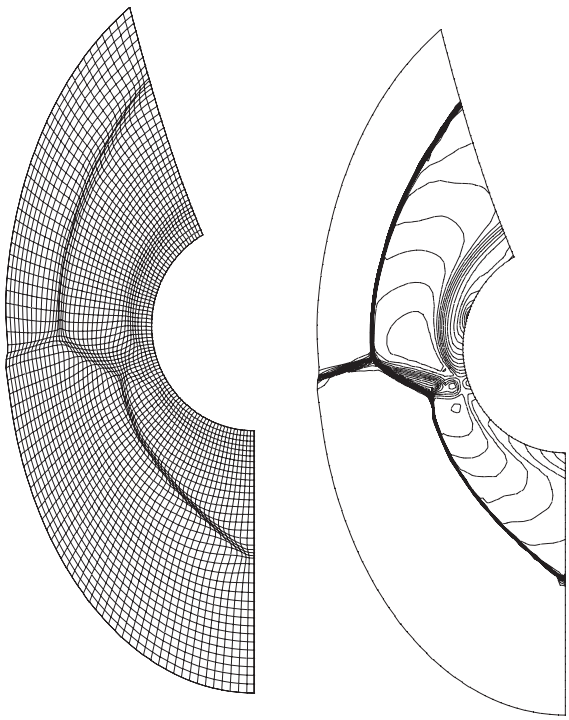

Fig. 3. Hypersonic flow with shock interactions. Left: adaptive mesh. Right: mach number contours.

4. Image processing

In this section, we show preliminary results that indicate potential applications of the deformation method in image representation, segmentation, and registration. *Image registration* is the process of establishing a common geometric reference frame between two image data. The process usually consists of two components: (1) Rigid (or affine) registration through global rotation, translation and scaling; (2) Non-rigid registration, which accounts for local deformation.

Non-rigid registration of biomedical image is an active research area with applications in computational anatomy, image-based diagnosis, image-guided surgical or radioactive systems, etc. Although significant progresses have been made as a result of multi-disciplinary efforts, it remains an outstanding open problem in 3D medical image analysis. The main scientific problem to be solved is to formulate models to calculate transformations, which (1) match corresponding landmarks representing biological and geometrical features; and (2), generate physically plausible deformation field in the whole domain.

The following example demonstrates the ability of the deformation method to generate meshes that represent images. The main idea is to let the monitor function f be proportional to the image intensity level.

A new trend in image registration is to formulate methods that are based on both the intensity levels and the landmarks. Intensity-based methods use only the intensity levels of the two images to drive the registration process. One shortcoming of these methods is their inaccuracy to match certain important anatomical structures. The landmark-based methods are developed to match landmark points, landmark curves, or landmark surfaces that identify important anatomical structures on the two images to be registered. Thus, it is natural to develop methods that combine the strengths of these two types of methods.

It is also straightforward to adopt the deformation method to the landmark matching problem. The main idea is to match correspondent landmark points of the two images by specifying the velocity vector field $\mathbf{v}$ at the landmark points. This can be accomplished by imposing the Dirichlet condition at (interior) landmark points when solving the div–curl system by the least square FEM. In the following example, we deform a square in a uniform Cartesian mesh (representing a uniform image) to a rectangle and at the same time, reduce the nodal intensity around an arc.

Example 3 (*Landmark matching and intensity levels*). Let $\{P(i)|i = 1, 2, \ldots, N\}$ be the landmark points in the fixed image, and let $\{Q(i)|I = 1, 2, \ldots, N\}$ be the corresponding landmark points in the moving image. We then impose the Dirichlet condition $\mathbf{v}(P(i)) = Q(i) - P(i)$. This condition is implemented in a least square finite element computer program which solves the div–curl system with specified velocity on the boundary and at the N landmarks $\{P(i)\}$. A preliminary example is presented in Fig. 4 to demonstrate the method. Certain landmark points on a square in the uniform Cartesian mesh are to be moved to corresponding points of a rectangle. As a result, the square is mapped to a rectangle. At the same time, nodes are concentrated on a piece of arc near the top-right corner.

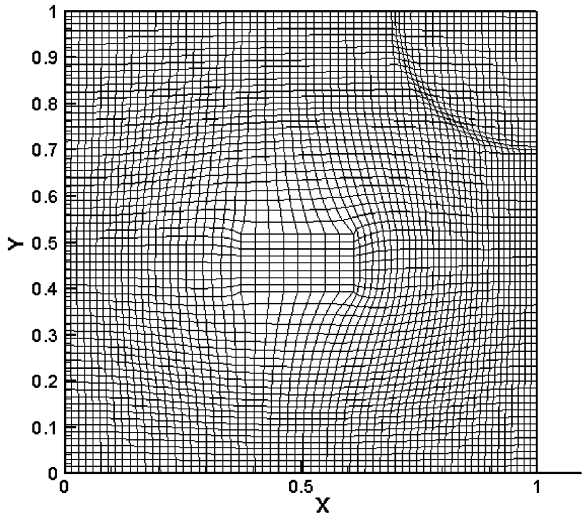

Fig. 4. Landmark matching from square to rectangle.

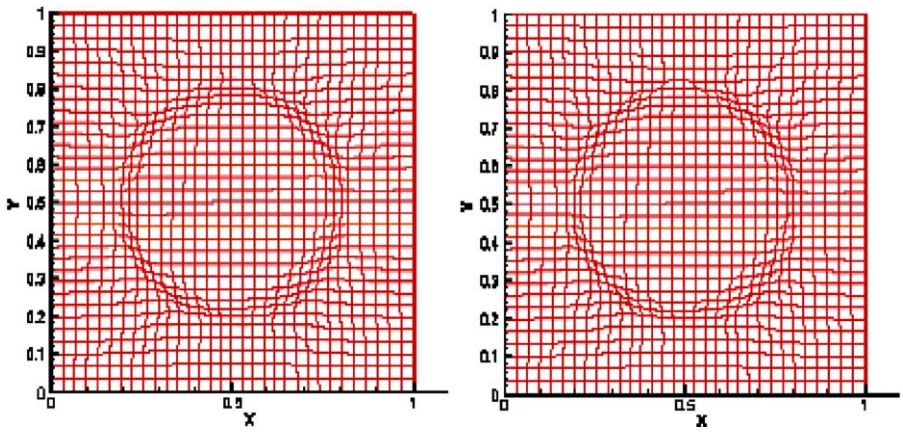

Fig. 5. Left: initial nodal transformation to be reconstructed. Right: reconstructed nodal transformation.

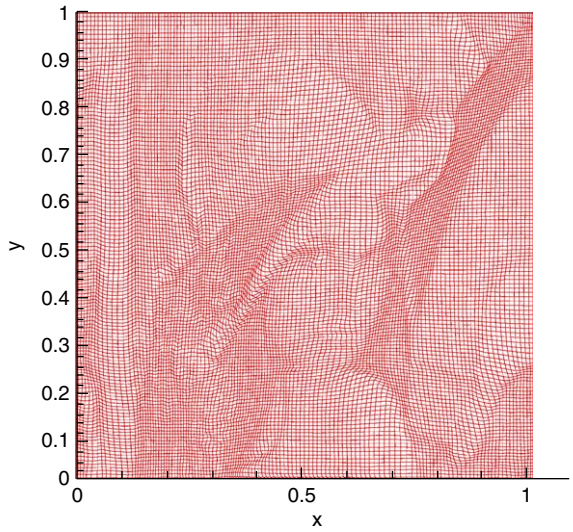

Fig. 6. 128 × 128 mesh adapted to image Lena.

Next two examples are designed to demonstrate that the deformation method can be used to reconstruct any given transformation.

Example 4 (*Reconstruction of any transformation by the div–curl method*). In this example, a mesh is first generated by deforming a Cartesian mesh to cluster around a circle through a mapping Φ (left picture of Fig. 5). We compute the divergence div Φ and curl Φ of the nodal mapping Φ. A new transformation (right picture of Fig. 5) is determined by solving

$$\operatorname{div} \Psi = \operatorname{div} \Phi, \quad \operatorname{curl} \Psi = \operatorname{curl} \Phi$$

with the Dirichlet condition $\Psi = \Phi$ on the boundary.

The transformation Ψ is remarkably similar to Φ by this simple computation. This example confirms the theoretical result that any transformation can be uniquely determined by its divergence and curl (see [8]).

Example 5 (*Reconstruction of image by the deformation method*). In this example, we use the deformation method to generate a 128 × 128 adaptive mesh (Fig. 6), whose cell area is proportional to the intensity level of the 256 × 256

Fig. 7. Image Lena.

image of Lena (commonly used for testing image processing algorithms). Compared to the image of Lena (Fig. 7), it is seen that the adaptive mesh in Fig. 6 indeed differentiates regions of different intensities.

5. Conclusions

We demonstrate in this paper that the deformation method can generate meshes of prescribed cell area. Specifically, a mathematical theorem shows that the Jacobian determinant of the nodal transformation is equal to a prescribed monitor function. Numerical examples are presented to confirm the theory. The least square finite element implementation shows promise of its potential applications in a variety of areas, ranging from computational fluid dynamics to image analysis. In the former, a monitor function is determined from an error indicator or from variations of physical variables such as the Mach number or density. The method moves nodes according to the monitor function so that the cell area is smaller where the error is greater. Here only qualitative correctness of cell area distribution is needed and the deformation method is one of several methods that provide qualitatively correct node movement. For the latter, a natural choice for the monitor function is to use the image's intensity level. This requires that the moving mesh method be able to control cell area (or volume, in 3D) precisely. The deformation method is, to our best knowledge, the only method appearing in the literature that has this ability.

The authors noticed the recent work [6] in which the deformation method is used in segmentation problems by the level set method.

Acknowledgements

The authors wish to thank the reviewers for valuable suggestions and comments on the manuscript. The first author thanks Professor Wang Renhong and Professor Luo Xiaonan for their invitation to the very fine conference in August 2004, held in Zhuhai, China.

References

[1] T.J. Baker, Mesh movement and metamorphosis, in: Proceedings of the Tenth International Meshing Roundtable, Sandia National Laboratories, Albuquerque, NM, 2001, pp. 387–396.

[2] X.X. Cai, D. Fleitas, B. Jiang, G. Liao, Moving meshes based on the least square finite element method, Comput. Math. Appl. 48 (2004) 1077–1085 (in memory of George Fix).
[3] Z.T.F. Chen, G.J. Fix, Least-squares finite element simulation of transonic flows, Appl. Numer. Math. 2 (1986) 399–408.
[4] E.A. Coddington, N. Levinson, Theory of Ordinary Differential Equations, McGraw-Hill, New York, 1955.
[5] G.J. Fix, M.D. Gunzburger, On least squares approximations to indefinite problems of the mixed type, Internat. J. Numer. Meth. Eng. 12 (1978) 453–469.
[6] X. Han, C.Y. Xu, L. Prince, A 2D moving grid geometric deformable model, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, 2003, pp. 153–160.
[7] T. Huges, J. Marsden, A Short Course in Fluid Mechanics, Mathematics Lecture Series, vol. 6, Publish or Perish, Inc., 1975.
[8] B. Jiang, Least Square Finite Element Method, Springer, Berlin, 1998.
[9] B.-N. Jiang, C.L. Chang, Least-squares finite element method for stokes problem, Comput. Meth. Appl. Mech. Eng. 78 (1994) 297–311.
[10] G. Liao, Z. Lei, G. de la Pena, Adaptive grids for resolution enhancement, Shock Waves 12 (2002) 153–156.
[11] F. Liu, S. Ji, G. Liao, An adaptive grid method and its application to steady Euler flow calculations, SIAM J. Sci. Comput. 20 (3) (1998) 811–825.
[12] S.M. Shontz, S.A. Vavasis, A mesh warping algorithm based on weighted Laplacian smoothing, in: Proceedings of the Tenth International Meshing Roundtable, Sandia National Laboratories, Santa Fe, NM, 2003, pp. 147–158.