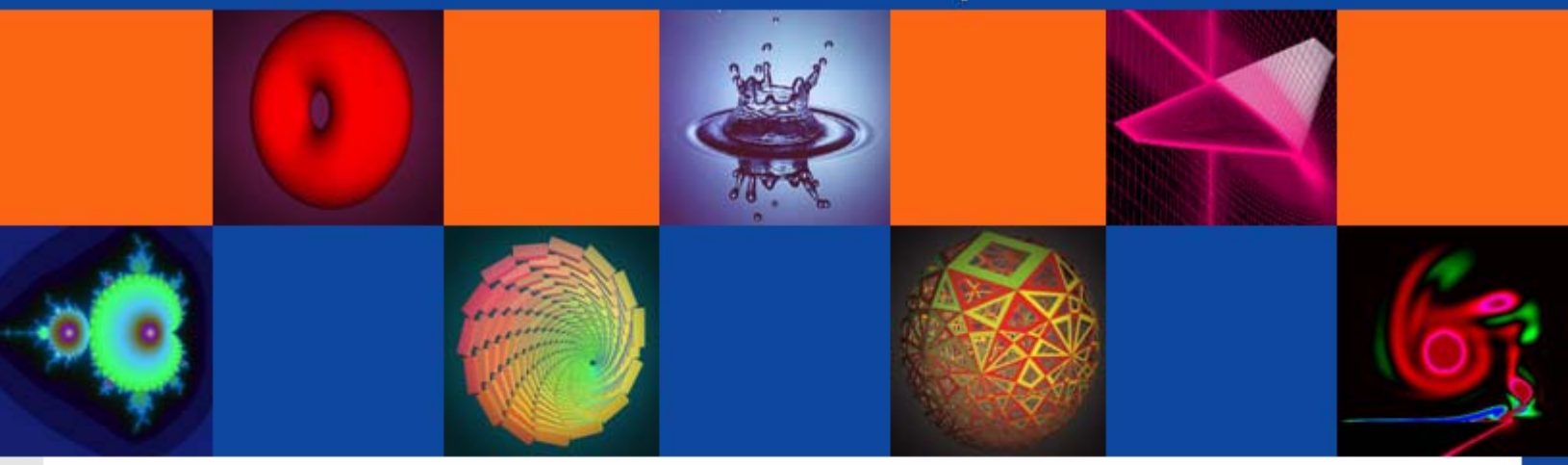
The University of Texas ARLINGTON

MATHEMATICS
Preprint Series



Reconstruction of Differentiable Transformations

Jie Liu
Guojun Liao
Jiaxing Xue

Technical Report 2007-21

http://www.uta.edu/math/preprint/

Reconstruction of Differentiable Transformations

Jie Liu
Department of Mathematics
Dixie State College
St George, UT 84770 USA

*Guojun Liao (Corresponding Author)
Department of Mathematics
University of Texas at Arlington
Arlington, TX 76019-0408 USA

Jiaxing Xue
Department of Computer Science Engineering
University of Texas at Arlington
Arlington, TX 76019 USA

Abstract: In the first part of this paper, various versions of the deformation method for grid adaptation and their applications are described. The main result of the paper is contained the second part. In particular, based on the deformation method, we propose a method of reconstructing any differentiable, invertible transformation on a square or a cube.

Key Words: Adaptive grids, Reconstruction of Transformations

1. Introduction

In order to solve partial differential equations (PDEs) numerically, we need to discretize the continuous differential equation into a system of algebraic equations for the nodal values of the field on a suitable grid (mesh) covering the physical domain.

Accuracy of the solution and computational efficiency are the two main concerns in computational grid generation. For problems with shock waves, boundary layers, etc. very fine grids over a small portion of the physical domain are required in order to resolve the large solution variations. The idea is to always put dense grids on the part which has large variation and coarse grids on the part which is smooth.

Various techniques ([8][9][10][11][12][14][19]) have been developed for generating moving grids, which relocate grid points to regions where higher resolution is needed. The total number of points and the connection between grid points are kept the same so that there is no need to change the data structure of the solver.

Different numerical methods are combined with moving grid techniques. Moving finite difference ([15],[16])and moving finite element ([17],[18])algorithms are developed. A moving mesh finite element method is designed to solve the incompressible Navier-Stokes equations in [20]. Some adaptive moving techniques for finite element and finite difference methods are reviewed in [21]. In [22], a moving mesh finite volume method is developed. Recently, the idea of meshfree adaptation is implemented in [38] and [39]. An overview of the meshfree methods can be found in [40].

In this work, we describe three versions of the deformation method. Then we study a reconstruction problem: Given a differentiable and invertible transformation, reconstruct the transformation by a differential system of linear, first order differential operators. Such a reconstruction method is proposed and some numerical examples are presented. The method is based on the deformation method developed in [27] and [28]. The least squares finite element method is used to solve the partial differential equations. This reconstruction approach has potential applications in image registration and computer vision simulation.

2. The Deformation Method

The deformation method is based on the idea of equivalent volume elements of a compact Riemannian manifold [26]. In 1992, this method is modified for grid adaptation in [28]. In this approach a new grid is constructed by moving the grid points such that specified cell volume distribution is achieved. A monitor function is defined according to the desired volume distribution. It is used to obtain a vector field by solving a linear Poisson equation. The grid points are moved according to a velocity field related to the vector field so obtained. The mathematical principles behind this method guarantee that grid lines of the same grid family will not cross each other. In [28], the Jacobian determinant, and consequently the cell volumes, was specified on the old grid before adaptation. In [31], the method is improved so that cell volumes can be specified as functions of the new grid after adaptation. In [30], this method is further extended into a real time moving grid method and used for solving one-dimensional unsteady problems. Some 1D and 2D applications and more analysis of adaptive moving grid by the deformation method were done in [29]. In [34], an adaptive deformation method is applied to solve the compressible Euler equations for field flows. A least-square finite element deformation method is developed in [36] and applied in [33] to a nonlinear problem. A 2D moving grid geometric deformable model using deformation method is developed in [35] for segmentation of image processing.

There are three versions of deformation method.

2.1 Version1

This is one of the steady versions of the deformation method where the Jacobian determinant is specified on the old grid ξ before adaptation.

Problem: Given a monitor function $f(\xi)$, find a transformation $\phi_1(\xi)$ such that

$$J(\phi_1) = \det \nabla \phi_1(\xi) = f(\xi) \tag{2.1}$$

We can use the following two steps to find such a transformation.

Step 1: Find a vector field $V(\xi)$ that satisfies:

$$\operatorname{div} V(\xi) = f(\xi) - 1 \tag{2.2}$$

Step 2: Define $V_t = \dfrac{V}{t + (1-t) f}$ and find transformations $\phi_t(\xi)$ by solving the ordinary differential equations

$$\frac{d\phi_t(\xi)}{dt} = V_t(\phi_t) \qquad t \in [0,1] \tag{2.3}$$

Here $\phi_t(\xi) = \phi(\xi, t)$, and let $\phi_1(\xi) = \phi(\xi, t=1)$.

Now, let us show that $\phi_1(\xi)$ satisfies (2.1).

In order to prove this, let

$$\begin{aligned} H(t,\xi) &= J(\phi_t(\xi))\left[t + (1-t) f(\phi_t(\xi))\right] \\ &= (\det \nabla \phi_t(\xi))\left[t + (1-t) f(\phi_t(\xi))\right] \end{aligned} \tag{2.4}$$

We now show

$$\frac{\partial H}{\partial t} = 0 \tag{2.5}$$

Since $\phi_0(\xi) = \phi(\xi, t=0)$ is the identity mapping, we have $\det \nabla \phi_0(\xi) = 1$ and $\phi_0(\xi) = \xi$. Assuming that (2.5) is true. Then

$$H(0,\xi) = \left(\det \nabla \phi_0(\xi)\right) f\left(\phi_0(\xi)\right) = f(\xi) \tag{2.6}$$

Also by (2.4) we have $$H(1,\xi) = \det \nabla \phi_1(\xi) \tag{2.7}$$

Thus (2.1) follows from (2.5), (2.6) and (2.7).
In order to prove (2.5), we need the Abel's Lemma.

Abel's Lemma:

Let M be a $n \times n$ matrix such that each element of the matrix is differentiable function of t. If $\frac{d}{dt}(M) = AM$ where A is a $n \times n$ matrix, then $\frac{d}{dt}(\det M) = (\operatorname{trace} A)(\det M)$.

This is a standard lemma, which can be found, for instance, in [25] or other standard ordinary differential equation textbooks.

Now we prove (2.5).

Proof:

$$\frac{\partial H}{\partial t} = \frac{\partial}{\partial t}\left[\left(\det \nabla \phi_t(\xi)\right)\left(t + (1-t) f\left(\phi_t(\xi)\right)\right)\right]$$

$$= \frac{\partial}{\partial t}\left[\left(\det \nabla \phi_t(\xi)\right)\right]\left[t + (1-t) f\left(\phi_t(\xi)\right)\right] \tag{2.8}$$

$$+ \left(\det \nabla \phi_t(\xi)\right)\frac{\partial}{\partial t}\left[t + (1-t) f\left(\phi_t(\xi)\right)\right]$$

Since $\frac{d}{dt}(\nabla \phi) = \nabla\left(\frac{d\phi}{dt}\right) = \nabla\left(V_t\left(\phi(\xi)\right)\right) = \left(\nabla_\phi V_t\right)(\nabla \phi)$, by Abel's Lemma we get:

$$\frac{\partial}{\partial t}\left(\det \nabla \phi\right) = \left(\operatorname{trace}\left(\nabla_\phi V_t\right)\right)\left(\det \nabla \phi\right), \tag{2.9}$$

where $\nabla_\phi V_t = \begin{vmatrix} \frac{\partial V_1}{\partial \phi_1} & \frac{\partial V_2}{\partial \phi_1} & \frac{\partial V_3}{\partial \phi_1} \\ \frac{\partial V_1}{\partial \phi_2} & \frac{\partial V_2}{\partial \phi_2} & \frac{\partial V_3}{\partial \phi_2} \\ \frac{\partial V_1}{\partial \phi_3} & \frac{\partial V_2}{\partial \phi_3} & \frac{\partial V_3}{\partial \phi_3} \end{vmatrix}$

Thus $$\operatorname{trace}\left(\nabla_\phi V_t\right) = \frac{\partial V_1}{\partial \phi_1} + \frac{\partial V_2}{\partial \phi_2} + \frac{\partial V_3}{\partial \phi_3} = \operatorname{div}_\phi V_t \tag{2.10}$$

Putting (2.10) into (2.9), we have

$$\frac{\partial}{\partial t}\left(\det \nabla \phi\right) = \left(\mathrm{div}_{\phi} V_t\right)\left(\det \nabla \phi\right) \tag{2.11}$$

Plugging (2.11) into (2.8), we have:

$$\frac{\partial H}{\partial t} = \left(\mathrm{div}_{\phi} V_t\right)\left(\det \nabla \phi\right)\left[t + (1-t) f\right] + \left(\det \nabla \phi\right)\frac{\partial}{\partial t}\left[t + (1-t) f\right]$$

$$= \left(\mathrm{div}_{\phi} V_t\right)\left(\det \nabla \phi\right)\left[t + (1-t) f\right] + \left(\det \nabla \phi\right)\left[1 - f + (1-t)(\nabla f) V_t\right] \tag{2.12}$$

$$= \left(\det \nabla \phi\right)\left\{\left(\mathrm{div}_{\phi} V_t\right)\left[t + (1-t) f\right] + \left[1 - f + (1-t)(\nabla f) V_t\right]\right\}$$

By step 2, we have

$$V = V_t\left[t + (1-t) f\right]$$

$$\Rightarrow divV = \left(divV_t\right)\left[t + (1-t) f\right] + V_t (1-t)\nabla f \tag{2.13}$$

$$\Rightarrow \left(divV_t\right)\left[t + (1-t) f\right] = divV - V_t (1-t)\nabla f$$

Plugging (2.13) into (2.12), we get:

$$\frac{\partial H}{\partial t} = \left(\det \nabla \phi\right)\left\{\left[divV - V_t (1-t)\nabla f\right] + \left[1 - f + (1-t)(\nabla f) V_t\right]\right\} \tag{2.14}$$

$$= \left(\det \nabla \phi\right)\left(divV + 1 - f\right)$$

By step 1, plugging (2.2) into (2.14), we get

$$\frac{\partial H}{\partial t} = \left(\det \nabla \phi\right)\left(f - 1 + 1 - f\right) = 0$$

Now, our remaining problem is how to find $V(\xi)$ such that $\mathrm{div} V(\xi) = f(\xi) - 1$ in Step 1. There are at least three different methods.

Method 1: Direct construction.

Method 2: Solve the Poisson equation $\Delta \omega = f - 1$ for ω, then let $V = \nabla \omega$.

The V found out by this way satisfies

$divV = div(\nabla \omega) = \Delta \omega = f - 1$.

Method 3: Solve the div-curl system $\begin{cases} divV = f - 1 \\ curlV = 0 \end{cases}$. Least-square finite element method is a good way to solve it.

We will discuss method 2 in Section 3. Here let us see some details about method 1.

In 2D, we need to find a vector field $V(V_1, V_2)$ on $\Omega = [0,1]\times[0,1]$ such that $\mathrm{div} V = f - 1 = g$ for a normalized monitor function $\iint_{\Omega} f = 1$

Let $G(x_1, x_2) = \int_0^{x_1} g(t, x_2)\, dt$

Define:
$$V:\begin{cases} V_1(x_1,x_2)=G(x_1,x_2)-h(x_1)G(1,x_2) \\ V_2(x_1,x_2)=h'(x_1)\int_0^{x_2}G(1,t)\,dt, \end{cases} \tag{2.15}$$

where $h(x_1)$ is a function satisfying $h(0)=0$, $h(1)=1$ and $h'(0)=h'(1)=0$.

For example, we can take $h(t)=\frac{1}{2}(1-\cos\pi t)$. Then

$$\begin{aligned} divV &= V_{1x_1}+V_{2x_2} \\ &= \frac{\partial}{\partial x_1}\left[G(x_1,x_2)-h(x_1)G(1,x_2)\right]+\frac{\partial}{\partial x_2}\left[h'(x_1)\int_0^{x_2}G(1,t)\,dt\right] \\ &= g(x_1,x_2)-h'(x_1)G(1,x_2)+h'(x_1)G(1,x_2) \\ &= g(x_1,x_2) \\ &= f(x_1,x_2)-1. \end{aligned}$$

So the vector constructed by (2.15) satisfies the divergence equation.

In 3D, a vector field $V(V_1,V_2,V_3)$ such that $divV = f-1$ can be defined by

$$V:\begin{cases} V_1(x_1,x_2,x_3)=\int_0^{x_1}g(t_1,x_2,x_3)\,dt_1-h(x_1)\int_0^1 g(t_1,x_2,x_3)\,dt_1 \\ V_2(x_1,x_2,x_3)=h'(x_1)\left(\int_0^{x_2}\int_0^1 g(t_1,t_2,x_3)\,dt_1dt_2-h(x_2)\int_0^1\int_0^1 g(t_1,t_2,x_3)\,dt_1dt_2\right) \\ V_3(x_1,x_2,x_3)=h'(x_1)h'(x_2)\int_0^{x_3}\int_0^1\int_0^1 g(t_1,t_2,x_3)\,dt_1dt_2dt_3, \end{cases}$$

where $h(t)=\frac{1}{2}(1-\cos\pi t)$. We can check directly that

$$\begin{aligned} divV &= V_{1x_1}+V_{2x_2}+V_{3x_3} \\ &= g(x_1,x_2,x_3)-h'(x_1)\int_0^1 g(t_1,x_2,x_3)\,dt_1+h'(x_1)\int_0^1 g(t_1,x_2,x_3)\,dt_1 \\ &\quad -h'(x_1)h'(x_2)\int_0^1\int_0^1 g(t_1,t_2,x_3)\,dt_1dt_2+h'(x_1)h'(x_2)\int_0^1\int_0^1 g(t_1,t_2,x_3)\,dt_1dt_2 \\ &= g(x_1,x_2,x_3) \\ &= f(x_1,x_2,x_3)-1 \end{aligned}$$

Another interesting direct construction method is worked out by in [27].

2.2 Version 2

This is another static version of the deformation method where the Jacobian determinant is specified on the new grid $\phi(\xi)$ after adaptation.

Problem: Given g and f (properly normalized), find a transformation $\phi: \partial\Omega \to \partial\Omega$ such that

$$g(\xi) J(\phi(\xi)) = f(\phi(\xi)), \quad \xi \in \Omega$$

where g and f satisfy $\int_\Omega \frac{1}{f} = \int_\Omega \frac{1}{g}$.

Note: The special case when *g=1* was treated in [32]. The general case is proposed in [43]. The material below is based on [43]. It is included here for self-completeness.

We can use the following three steps to find such a transformation.

Step 1 Compute V such that

$$div(V(\xi)) = \frac{1}{g(\xi)} - \frac{1}{f(\xi)} \text{ in } \Omega, \text{ and } V(\xi)\cdot\vec{n} = 0, \qquad \xi \in \partial\Omega.$$

Step 2 For each fixed node ξ, solve the ODE

$$\frac{\partial\varphi(\xi,t)}{\partial t} = \eta(\varphi(\xi,t),t) \qquad 0 \le t \le 1$$

with $\varphi(\xi,0) = \xi$, where $\eta(x,t) = \dfrac{V(x)}{t\frac{1}{f(x)} - (1-t)\frac{1}{g(x)}}$

Step 3 Define $\phi(\xi) = \varphi(\xi,1)$, then ϕ will be the solution.

Now, we show that ϕ satisfies (2.16).

Let
$$H(\xi,t) = \left(J(\varphi)(\xi,t)\right)\left(t\frac{1}{f(\varphi(\xi,t))} + (1-t)\frac{1}{g(\varphi(\xi,t))}\right) \tag{2.17}$$

If we can show (2.17) is independent of t, i.e.

$$\frac{\partial H}{\partial t} = 0 \tag{2.18}$$

then $H(\xi,0) = J(\varphi(\xi,0))\dfrac{1}{g(\varphi(\xi,0))} = 1/g(\xi)$ and

$$H(\xi,1) = \left(J(\varphi(\xi,1))\right)\frac{1}{f(\varphi(\xi,1))} = J/f(\phi(\xi))$$

$$\frac{\partial H}{\partial t} = 0 \Rightarrow H(\xi,0) = H(\xi,1) \Rightarrow 1/g(\xi) = J/f(\phi(\xi)) \Rightarrow gJ = f$$

The proof of (2.18) is very similar to the proof of the first case [42].

2.3 Version 3 [31]

This is the version for real time (or time-accurate) adaptation.

Problem: Given a monitor function *f(x,t)*, normalized so that

$\int \frac{1}{f} = |\Omega|$, where $|\Omega|$ is the volume of the domain , find a transformation ϕ such that

$$J\left(\phi(\xi,t)\right) = f\left(\phi(\xi,t),t\right) \text{ for } t>0, \tag{2.19}$$

assuming that (2.19) is true at $t=0$.

Such a transformation can be found by the following two steps.

Step 1: Find a vector field $V(\phi,t)$ such that:

$$\operatorname{div}_\phi V(\phi,t) = -\frac{\partial}{\partial t}\frac{1}{f(\phi,t)} = -\frac{\partial}{\partial t} g(\phi,t) \ ,$$

where $g\left(\phi(\xi,t),t\right) = \dfrac{1}{f\left(\phi(\xi,t),t\right)}$.

Step 2: Solve the following ordinary differential equation (ODE) for the transformation $\phi(\xi,t)$:

$$\frac{\partial \phi(\xi,t)}{\partial t} = f(\phi,t)V(\phi,t) = \eta(\phi,t).$$

We now show that $\phi(\xi,t)$ satisfies (2.19).

In order to prove this, define $H(\xi,t)$ by

$$H = \left(\det \nabla\phi(\xi,t)\right) g\left(\phi(\xi,t),t\right) = \frac{J(\phi,t)}{f(\phi,t)} = Jg \quad .$$

If we can show $\dfrac{\partial H}{\partial t} = 0$, then we have $H = \dfrac{J}{f} = \text{const}$.

Proof of $\dfrac{\partial H}{\partial t} = 0$:

$$\frac{\partial H}{\partial t} = \frac{\partial J}{\partial t} g\left(\phi(\xi,t),t\right) + J\frac{\partial g\left(\phi(\xi,t),t\right)}{\partial t} \tag{2.20}$$

By Abel's Lemma, since

$\dfrac{\partial}{\partial t}\left(\nabla_\xi \phi\right) = \nabla_\xi\left(\dfrac{\partial \phi}{\partial t}\right) = \left(\nabla_\phi\left(\dfrac{\partial \phi}{\partial t}\right)\right)\left(\nabla_\xi \phi\right) = \left(\nabla_\phi(\eta)\right)\left(\nabla_\xi \phi\right)$, we get

$$\frac{\partial J}{\partial t} = \frac{\partial}{\partial t}\left(\det \nabla\phi\right) = \left(\operatorname{trace}\left(\nabla_\phi \eta(\phi,t)\right)\right)\left(\det \nabla\phi\right)$$

$$= \left(\operatorname{div}_\phi \eta(\phi,t)\right)\left(\det \nabla\phi\right) = \left(\operatorname{div}_\phi\left(f(\phi,t)V(\phi,t)\right)\right)J$$

$$= \left[f(\phi,t)\operatorname{div}_\phi\left(V(\phi,t)\right) + \left\langle \nabla_\phi f(\phi,t), V(\phi,t)\right\rangle\right]J$$

$$= \left[-f\frac{\partial g}{\partial t} + \left\langle \nabla_\phi f, V\right\rangle\right]J$$

That is $$\frac{\partial J}{\partial t} = \left[-f \frac{\partial g}{\partial t} + \left\langle \nabla_\phi f, V \right\rangle \right] J \tag{2.21}$$

Also we have $$\frac{\partial g(\phi, t)}{\partial t} = \left\langle \nabla_\phi g, \frac{\partial \phi}{\partial t} \right\rangle + \frac{\partial g}{\partial t} \tag{2.22}$$

Plugging (2.21) and (2.22) into (2.20), we get

$$\frac{\partial H}{\partial t} = \left[-f \frac{\partial g}{\partial t} + \left\langle \nabla_\phi f, V \right\rangle \right] Jg + J \left(\left\langle \nabla_\phi g, \frac{\partial \phi}{\partial t} \right\rangle + \frac{\partial g}{\partial t} \right)$$

$$= J \left[-fg \frac{\partial g}{\partial t} + \left\langle \nabla_\phi f, V \right\rangle g + \left\langle \nabla_\phi g, fV \right\rangle + \frac{\partial g}{\partial t} \right] \quad (\text{note: } \frac{\partial \phi}{\partial t} = fV \text{ is used})$$

$$= J \left[-\frac{\partial g}{\partial t} + \left\langle \nabla_\phi f, V \right\rangle g + f \left\langle \nabla_\phi g, V \right\rangle + \frac{\partial g}{\partial t} \right] \quad (\text{note: } g = \frac{1}{f} \Rightarrow fg = 1 \text{ is used})$$

$$= J \left[\left\langle \left(g \nabla_\phi f + f \nabla_\phi g \right), V \right\rangle \right] \quad \left(\text{note: } fg = 1 \Rightarrow \nabla(fg) = g \nabla f + f \nabla g = 0 \text{ is used} \right)$$

$$= J \left[\left\langle 0, V \right\rangle \right]$$

$$= 0$$

The numerical implementations for all the three versions are similar. The method for version1 also works for other versions after the right hand side of the divergence equation is adjusted accordingly.

3. Reconstruction of Transformations

In this section we reconstruct differentiable transformations using a div-curl system. The idea comes from the implementation of deformation method for finding transformations. Notice that for all the three versions, the first step is to find a vector field by solving a divergence equation with different right hand sides. After we add an equation of the curl of the vector field, we can set up a div-curl system of equations. The least-square finite element method is a good way to solve it [41]. Now let's think in the opposite direction. For a transformation given on a uniform initial grid we can calculate its divergence and curl at each point. Thus we can set up a div-curl system of equations for each point. Solving this system on the grid, we can reconstruct the given transformation. This idea can be used to reconstruct any differentiable and invertible transformation.

This reconstruction method may apply to image registration, which is the process of establishing point-by-point correspondence between two images of a scene. Sets of data acquired by sampling the same scene or object at different times, or from different perspectives, are in different coordinate systems. Image registration is the process of transforming the different sets of data into one coordinate system. Registration is necessary in order to be able to compare or integrate the data obtained from different measurements. This process is also needed in various computer vision applications.

3.1 Div-curl System

We will first take a look at the div-curl system. Let D be an open bounded domain in R^3 with a piecewise smooth boundary $\Gamma = \Gamma_1 \bigcup \Gamma_2$. Let (x, y, z) denote a point in D. Let $F = P\vec{i} + Q\vec{j} + R\vec{k}$ be a vector field in D. Let $\vec{n}$ be the unit outward normal vector on the boundary. Then the 3D div-curl system of equations is:

$$\begin{cases} divF = \alpha & in\, D \\ curlF = \vec{\beta} & in\, D \\ \vec{n} \cdot F = 0 & on\, \Gamma_1 \\ \vec{n} \times F = 0 & on\, \Gamma_2 \end{cases} \tag{3.1}$$

where $\vec{\beta} = \beta_1\vec{i} + \beta_2\vec{j} + \beta_3\vec{k}$.

Our goal is to solve for P, Q, R, for a total of three unknowns. But we have four scalar equations in this system. So, it appears that this system is 'over-determined'. Let us reconsider this system by introducing a dummy variable θ as in [43], where $\theta \equiv 0$ in D and $\theta = 0$ on Γ_1 so that the system becomes:

$$\begin{cases} divF = \alpha & in\, D \\ \nabla\theta + curlF = \vec{\beta} & in\, D \\ \vec{n} \cdot F = 0 & on\, \Gamma_1 \\ \theta = 0 & on\, \Gamma_1 \\ \vec{n} \times F = 0 & on\, \Gamma_2 \end{cases} \tag{3.2}$$

It can be shown that system (3.2) is equivalent to system (3.1). Detailed proof can be found in [41]. Notice that system (3.2) is a system with four unknowns and four equations.

In Cartesian coordinates, we have:

$$curl\, F = \left(\frac{\partial R}{\partial y} - \frac{\partial Q}{\partial z}\right)\vec{i} + \left(\frac{\partial P}{\partial z} - \frac{\partial R}{\partial x}\right)\vec{j} + \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right)\vec{k}$$

$$\nabla\theta = \frac{\partial\theta}{\partial x}\vec{i} + \frac{\partial\theta}{\partial y}\vec{j} + \frac{\partial\theta}{\partial z}\vec{k}$$

$$div\, F = \frac{\partial P}{\partial x} + \frac{\partial Q}{\partial y} + \frac{\partial R}{\partial z}$$

So system (3.2) can be written as:

$$\begin{cases} \dfrac{\partial\theta}{\partial x} + \dfrac{\partial R}{\partial y} - \dfrac{\partial Q}{\partial z} = \beta_1 \\ \dfrac{\partial\theta}{\partial y} + \dfrac{\partial P}{\partial z} - \dfrac{\partial R}{\partial x} = \beta_2 \\ \dfrac{\partial\theta}{\partial z} + \dfrac{\partial Q}{\partial x} - \dfrac{\partial P}{\partial y} = \beta_3 \\ \dfrac{\partial P}{\partial x} + \dfrac{\partial Q}{\partial y} + \dfrac{\partial R}{\partial z} = \alpha \end{cases} \tag{3.3}$$

Define $\tilde{F} = \begin{pmatrix} P \\ Q \\ R \\ \theta \end{pmatrix}$ and $\tilde{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \alpha \end{pmatrix}$, then this system can be written in a matrix form:

$$A_0\tilde{F} + A_1\frac{\partial\tilde{F}}{\partial x} + A_2\frac{\partial\tilde{F}}{\partial y} + A_3\frac{\partial\tilde{F}}{\partial z} = \tilde{\beta} \text{ where}$$

$$A_0 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \qquad A_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

For any nonzero triplets (x, y, z), the characteristic polynomial for system (3.3) is

$$\det(A_1x + A_2y + A_3z) = \det\begin{pmatrix} 0 & -z & y & x \\ z & 0 & -x & y \\ -y & x & 0 & z \\ x & y & z & 0 \end{pmatrix} = \left(x^2 + y^2 + z^2\right)^2 \neq 0$$

Thus, system is elliptic and properly determined.
Least-square finite element method is a good way to solve the div-curl system. Detailed analysis can be found in [41]. The numerical implementation procedures can be found in [39].

3.2 Least Squares FEM

Let us consider the linear boundary-value problem:

$$\begin{aligned} Au &= f \ \text{ in } \Omega \\ Bu &= g \ \text{ on } \Gamma \end{aligned} \tag{3.4}$$

where $Au = \sum_{i=1}^{n_d} A_i \frac{\partial u}{\partial x_i} + A_0 u$ ($n_d = 2$ for 2D, $n_d = 3$ for 3D). B is a boundary operator. f and g are given vector-valued functions. u is a vector with m unknown functions of $\boldsymbol{x}\left(x_1, \cdots x_{n_d}\right)$.

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \cdots \\ u_m \end{pmatrix},\ f = \begin{pmatrix} f_1 \\ f_2 \\ \cdots \\ f_{n_d} \end{pmatrix},\ g = \begin{pmatrix} g_1 \\ g_2 \\ \cdots \\ g_{n_d} \end{pmatrix}$$

Define the residual as $R = Au - f$, then if $R = 0$ we get the exact solution for u . The least-square finite element method is to minimize R in a least-square sense, that is, to minimize the following functional:

$$I(v) = \|R\|_0^2 = \int_\Omega (Av - f)^2 d\omega .$$

A necessary condition for u to minimize $I(v)$ is:

$$\lim_{t\to 0} \frac{d}{dt} I(u + tv) = 0 \ .$$

Since

$$I(u+tv) = \int_\Omega \left[A(u+tv) - f \right]^2 d\omega$$

$$= \int_\Omega \left[(Au)^2 + (Av)^2 t^2 + f^2 + 2(Au)(Av)t - 2(Av) ft - 2(Au) f \right] d\omega,$$

we have

$$\lim_{t\to 0} \frac{d}{dt} I(u+tv) = \lim_{t\to 0} \int_\Omega 2\left[(Av)^2 t + (Au)(Av) - (Av) f \right] d\omega$$

$$= \int_\Omega 2\left[(Au)(Av) - (Av) f \right] d\omega$$

$$= 0.$$

Thus

$$\int_\Omega \left[(Au)(Av) \right] d\omega = \int_\Omega \left[(Av) f \right] d\omega .$$

That is

$$(Au, Av) = (f, Av). \tag{3.5}$$

This is the variational principle of equation (3.4).

In finite element, we discretize the domain into elements and then introduce finite element basis. Let φ_j be the element shape function, we write the expansion of the unknown variables in each element as

$$u_h^e(\boldsymbol{x}) = \sum_{j=1}^{N_n} \varphi_j(\boldsymbol{x}) \begin{pmatrix} u_1 \\ u_2 \\ \cdots \\ u_m \end{pmatrix}_j \tag{3.6}$$

where N_n is the number of nodes for one element.

Introducing (3.6) into (3.5) we get a linear system of algebraic equations:

$$KU = F \tag{3.7}$$

Here

$$K_e = \int_\Omega \left(A\varphi_1, A\varphi_2, \cdots, A\varphi_{N_n}\right)^T \left(A\varphi_1, A\varphi_2, \cdots, A\varphi_{N_n}\right) d\Omega$$

$$F_e = \int_\Omega \left(A\varphi_1, A\varphi_2, \cdots, A\varphi_{N_n}\right)^T f d\Omega$$

are the element matrices used to assemble the global matrix K and F.

3.3 Solving the Div-Curl System

Let's take a look at the definition of divergence and curl of a vector field first.

If $\boldsymbol{V} = P\vec{i} + Q\vec{j} + R\vec{k}$ is a vector field on $\square^{3\vec{3}}$ and the partial derivatives of $P(x, y, z)$, $Q(x, y, z)$ and $R(x, y, z)$ all exist, then

$$div\boldsymbol{V} = \frac{\partial P}{\partial x} + \frac{\partial Q}{\partial y} + \frac{\partial R}{\partial z}$$

$$curl\boldsymbol{V} = \left(\frac{\partial R}{\partial y} - \frac{\partial Q}{\partial z}\right)\vec{i} + \left(\frac{\partial P}{\partial z} - \frac{\partial R}{\partial x}\right)\vec{j} + \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right)\vec{k}$$

In 2D, all the terms related to R and z vanish. So we have

$$div\boldsymbol{V} = \frac{\partial P}{\partial x} + \frac{\partial Q}{\partial y}$$

$curl\boldsymbol{V} = \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right)\vec{k}$, where $\vec{k}$ is the unit outward normal vector, usually denoted as $\vec{n}$.

The matrix form of the div-curl system can be written as:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} \frac{\partial P}{\partial x} \\ \frac{\partial Q}{\partial x} \\ \frac{\partial R}{\partial x} \end{pmatrix}+\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}\begin{pmatrix} \frac{\partial P}{\partial y} \\ \frac{\partial Q}{\partial y} \\ \frac{\partial R}{\partial y} \end{pmatrix}+\begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} \frac{\partial P}{\partial z} \\ \frac{\partial Q}{\partial z} \\ \frac{\partial R}{\partial z} \end{pmatrix}=\begin{pmatrix} \frac{\partial P}{\partial x}+\frac{\partial Q}{\partial y}+\frac{\partial R}{\partial z} \\ \frac{\partial R}{\partial y}-\frac{\partial Q}{\partial z} \\ \frac{\partial P}{\partial z}-\frac{\partial R}{\partial x} \\ \frac{\partial Q}{\partial x}-\frac{\partial P}{\partial y} \end{pmatrix}$$

In 3D, linear hexahedral elements are used and the finite element approximation at each hexahedral is given by

$$V_h^e(x)=\sum_{i=1}^{8}\left[\varphi_i\begin{pmatrix} p_i \\ q_i \\ r_i \end{pmatrix}\right]$$

where p_i, q_i, r_i are the nodal values at the ith node of the hexahedral element and φ_i's are the shape functions.

The element matrices used to assemble the algebraic system $KV = F$ are

$$K_e=\int_{\Omega_e}\begin{pmatrix} (A\varphi_1)^T A\varphi_1 & \cdots & (A\varphi_1)^T A\varphi_8 \\ \vdots & \ddots & \vdots \\ (A\varphi_8)^T A\varphi_1 & \cdots & (A\varphi_8)^T A\varphi_8 \end{pmatrix}d\Omega$$

$$F_e=\int_{\Omega_e}\begin{pmatrix} (A\varphi_1)^T \boldsymbol{f} \\ \vdots \\ (A\varphi_8)^T \boldsymbol{f} \end{pmatrix}d\Omega$$

where $$A\varphi_i=\begin{pmatrix} \frac{\partial \varphi_i}{\partial x} & \frac{\partial \varphi_i}{\partial y} & \frac{\partial \varphi_i}{\partial z} \\ 0 & -\frac{\partial \varphi_i}{\partial z} & \frac{\partial \varphi_i}{\partial y} \\ \frac{\partial \varphi_i}{\partial z} & 0 & -\frac{\partial \varphi_i}{\partial x} \\ -\frac{\partial \varphi_i}{\partial y} & \frac{\partial \varphi_i}{\partial x} & 0 \end{pmatrix} \quad \text{for} \quad i=1,2,...,8.$$

and $\boldsymbol{f} = \begin{pmatrix} \frac{\partial P}{\partial x} + \frac{\partial Q}{\partial y} + \frac{\partial R}{\partial z} \\ \frac{\partial R}{\partial y} - \frac{\partial Q}{\partial z} \\ \frac{\partial P}{\partial z} - \frac{\partial R}{\partial x} \\ \frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} \end{pmatrix}$, calculated from the given transformation.

4. Numerical Examples

W set the position for the given transformation as $xn(i,j,k)$, $yn(i,j,k)$, $zn(i,j,k)$, and the array for our initial grid as $x(i,j,k)$, $y(i,j,k)$, $z(i,j,k)$, then

$$\frac{\partial P}{\partial x} = \frac{xn(i+1,j,k) - xn(i-1,j,k)}{x(i+1,j,k) - x(i-1,j,k)}$$

$$\frac{\partial P}{\partial y} = \frac{xn(i,j+1,k) - xn(i,j-1,k)}{y(i,j+1,k) - y(i,j-1,k)}$$

$$\frac{\partial P}{\partial z} = \frac{xn(i,j,k+1) - xn(i,j,k-1)}{z(i,j,k+1) - z(i,j,k-1)}$$

$$\frac{\partial Q}{\partial x} = \frac{yn(i+1,j,k) - yn(i-1,j,k)}{x(i+1,j,k) - x(i-1,j,k)}$$

$$\frac{\partial Q}{\partial y} = \frac{yn(i,j+1,k) - yn(i,j-1,k)}{y(i,j+1,k) - y(i,j-1,k)}$$

$$\frac{\partial Q}{\partial z} = \frac{yn(i,j,k+1) - yn(i,j,k-1)}{z(i,j,k+1) - z(i,j,k-1)}$$

$$\frac{\partial R}{\partial x} = \frac{zn(i+1,j,k) - zn(i-1,j,k)}{x(i+1,j,k) - x(i-1,j,k)}$$

$$\frac{\partial R}{\partial y} = \frac{zn(i,j+1,k) - zn(i,j-1,k)}{y(i,j+1,k) - y(i,j-1,k)}$$

$$\frac{\partial R}{\partial z} = \frac{zn(i,j,k+1) - zn(i,j,k-1)}{z(i,j,k+1) - z(i,j,k-1)}$$

Following are some of the numerical examples. Let the coordinates of the new position of node X_i be XN_i. We define Error = $\max\left|XN_i - X_i\right| = \sqrt{(xn_i - x_i)^2 + (yn_i - y_i)^2 + (zn_i - z_i)^2}$, i = 1,…, nmax. Error is the maximal distance between each pair of corresponding nodes of the given and the reconstructed transformations. nmax is the maximum number of nodes. Error is used to measure the accuracy of the reconstruction method.

The grid size of the following examples is 64×64 over the unit square $[0,1]\times[0,1]$ for 2D and $40\times 40\times 40$ over the unit cube $[0,1]\times[0,1]\times[0,1]$ for 3D. That means the distance between adjacent points in the uniform grid is $\frac{1}{64}=0.015625$ for 2D and $\frac{1}{40}=0.025$ for 3D.

Example 1: A transformation from the uniform Cartesian grid (Figure 1.1) to a grid stretched to a rectangle and refined around an arc is reconstructed. Figure 1.2 shows the intermediate step at half of the time steps. The final result is shown in Figure 1.3.

Example 2: A transformation on the 3D Cartesian grid from a unit cube is shown in Figure 2.1 by a grid adapted to a sphere. The reconstructed transformation is shown in Figure 2.2. The maximum error in is Error = 5.225×10^{-3}, which is compared favorably to the grid size $\frac{1}{40}=0.025 = 25\times10^{-3}$.

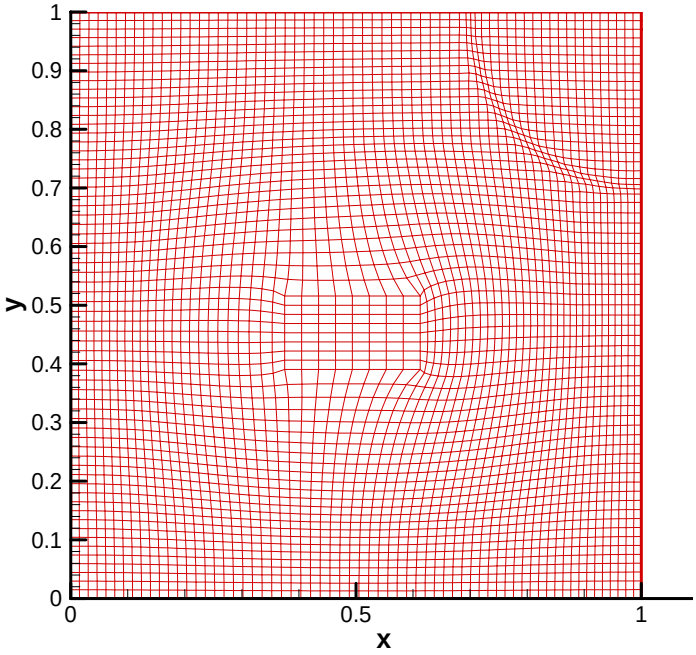


Figure 1.1 The Given Transformation in Example 1

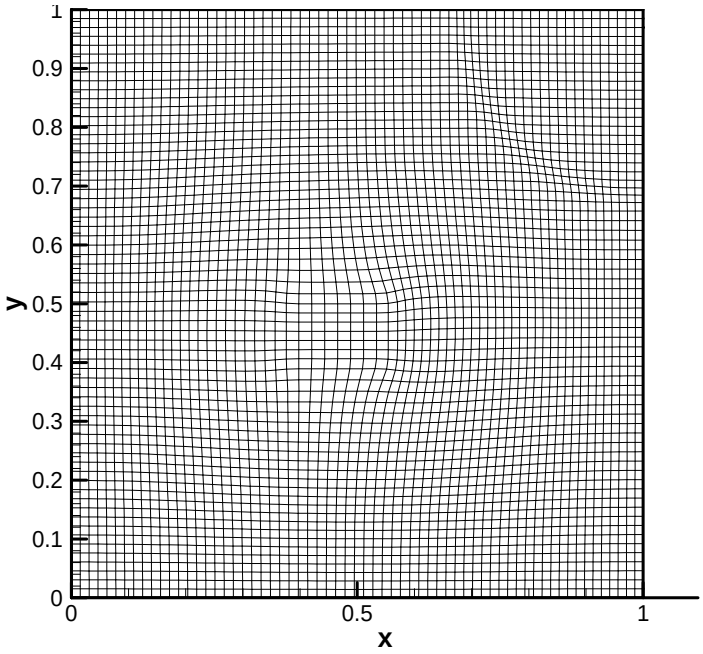


Figure 1.2 Reconstruction at time step $t = 5$

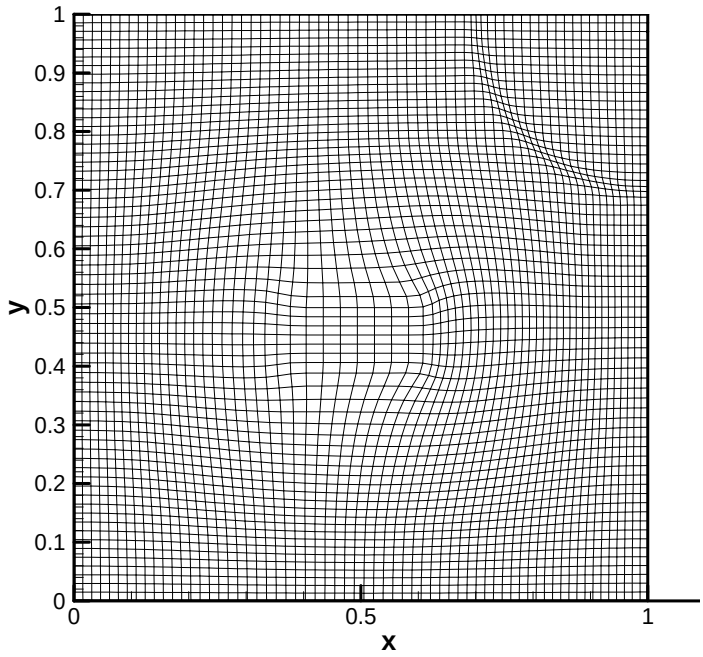


Figure 1.3 Reconstruction at time step $t = 10$

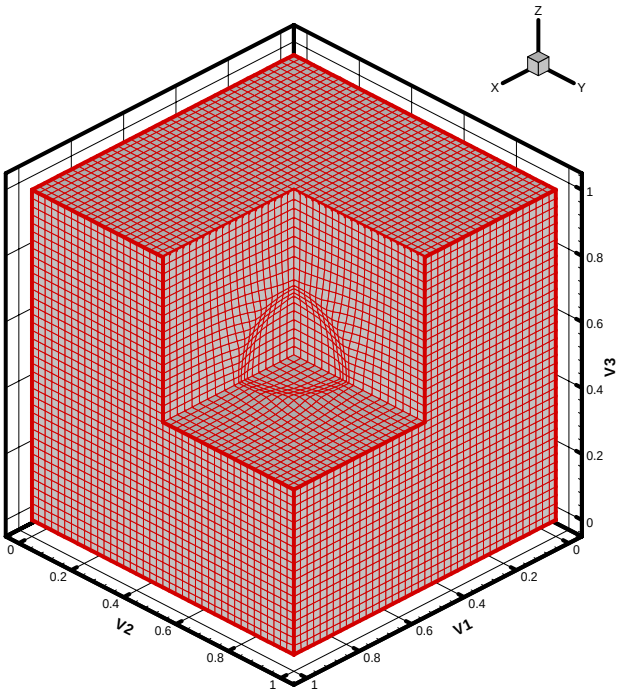


Figure 2.1 The Given Transformation in Example 2: A cube with a ball inside (cutaway plot)

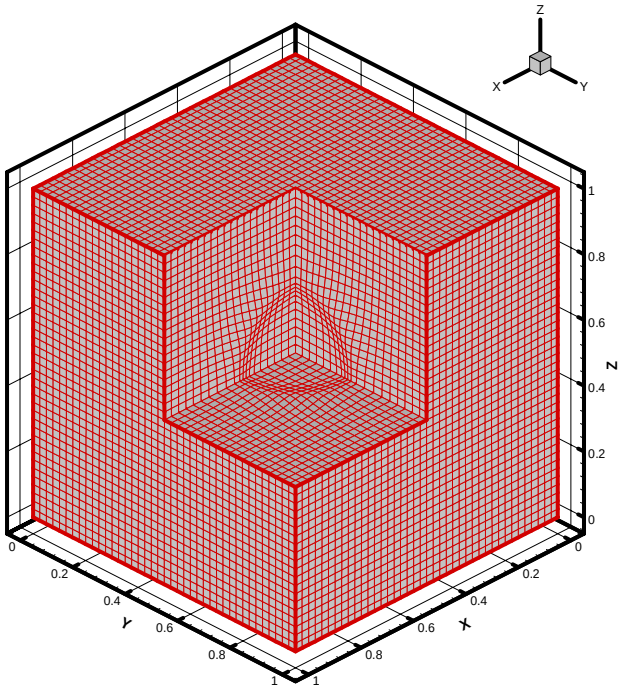


Figure 2.2 Reconstruction of Figure 2.1 for Example 2

5. Conclusions

In the first part of this paper, we describe all three versions of the deformation method for adaptive grid generation. For a monitor function f constructed according to desired grid size distribution, we can construct a transformation with Jacobian determinant $J = f$.

The main result of the paper is in the second part. We show that one can reconstruct any given differentiable, invertible transformation by its divergence and curl. The least squares finite element method is used to solve the div-curl system. Numerical examples in both two and three dimensions are presented. The examples show excellent accuracy of the method.

References

[1] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin, Numerical Grid Generation, North-Holland, Amsterdam, 1985.

[2] P. Knupp and S. Steinberg, The Fundamentals of Grid Generation, CRC Press, 1993.

[3] G. Carey, Computational Grid Generation, Adaptation and Solution Strategies, Taylor and Francis, 1997.

[4] J. F. Thompson, B. Soni, N. Weatherill, Handbook of Grid Generation, CRC Press, 1998.

[5] J. Thompson, A reflection on grid generation in the 90s: trends, needs and influences, 5th International Conference on Numerical Grid Generation in Computational Field Simulations, Mississippi State University, pp.1029-1110, 1996.

[6] B. Hamann, R. J. Moorhead, A survey of grid generation methodologies and scientific visualization efforts, *Chapter 3 in Scientific Visualization: Overviews, Methodologies, and Techniques*, pp. 59-101, 1997.

[7] D. Arney, J. Flaherty, An adaptive mesh-moving and local refinement method for time-dependent partial differential equations, ACM Transaction in Mathematical Software, 16, 1990.

[8] K. Miller, R. Miller, Moving Finite Elements I, *SIAM J. Numer.* Anal. 18, 1019-1032, 1981.

[9] K. Miller, Moving Finite Elements II, *SIAM J. Numer. Anal.* 18, 1033-1057, 1981.

[10] N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code, Part I, in 1D, *SIAM J. Sci. Comput.* 19, 728-765, 1998.

[11] N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code, Part II, in 2D, *SIAM J. Sci. Comput.* 19, 766-798, 1998.

[12] M. Baines, Moving finite elements, Oxford University Press, New York, 1994.

[13] J. Castillo, Mathematical Aspects of Numerical Grid Generation, Society for Industrial and Applied Mathematics, 1991.

[14] W. Cao, W. Huang and R. D. Russell, Approaches for Generating Moving Adaptive Meshes: Location versus Velocity, *Appl. Num. Math.*, 47 (2003), 121-138.

[15] E. A. Dorfi and L. Drury, Simple adaptive grids for 1D initial value problems, J. Comput. Phys. 69, 175-195, 1987.

[16] W. Huang, Y. Ren and R.D. Russell, Moving mesh partial differential equations (MMPDES) based on the equidistribution principle, *SIAM J. Numer. Anal.* 31, 709-730, 1994.

[17] W. Cao, W. Huang and R. D. Russell, An r-adaptive finite element method based upon moving mesh PDEs, Journal of Computational physics, 170, 871-892, 2001.

[18] R. Li, T. Tang, and P. Zhang, A moving mesh finite element algorithm for singular problems in two and three space dimensions, *J. Comput. Physics*, 177, 365-393, 2002.

[19] R. Li, T, Tang, and P. Zhang, Moving mesh methods in multiple dimensions base on harmonic maps, *J. Comput. Physics*, 170, 562-588, 2001.

[20] Y. Di, R. Li, T. Tang, and P. Zhang, Moving mesh finite element methods for the incompressible Navier-Stokes equations, *SIAM, J. Sci. Comput.*, 26, 1036-1056, 2005.

[21] D. Hawken, J. Gottlieb and J. Hansen, Review of some adaptive node-movement techniques in finite-element and finite-difference solutions of partial differential equations, *J. Comput. Physics*, 95, 254-302, 1991.

[22] A. van Dam, P. A. Zegeling, A robust moving mesh finite volume method applied to 1D Hyperbolic conservation laws from magnetohydrodynamics, *J. Comput. Physics*, 2006.

[23] W. Cao, W. Huang and R.D. Russell, A study of monitor functions for two-dimensional adaptive mesh generation, *SIAM J. Sci. Comput.* 20, 1978-1994, 1999.

[24] H. M. Tsai, A. S. F. Wong, J. Cai, Y. Zhu, and F. Liu, Unsteady flow calculations with a parallel multiblock moving mesh algorithm, *AIAA Journal*, 39, No. 6, 2001.
[25] Thomas J.R. Hughes and Jerrold E. Marsden, A short course in fluid mechanics: mathematics lectures series 6, Publish or Perish, Inc. 1976
[26] J. Moser, Volume elements of a Riemann Manifold, *Trans AMS*, 120, 1965
[27] G. Liao and J. Su, Grid generation via deformation, *Appl. Math. Lett.*, 5, 1992.
[28] G. Liao and D. Anderson, A new approach to grid generation, *Appl. Anal.*, 44, 1992.
[29] P. B. Bochev, G. Liao, and G. C. de la Pena. Analysis and computation of adaptive moving grids by deformation. *Numerical Methods for Partial Differential Equations*, 12, 1996.
[30] B. Semper and G. Liao, "A moving grid finite-element method using grid deformation", *Numer. Meth. PDE*, 11:603, 1995.
[31] G. Liao, T. Pan, and J. Su, "Numerical Grid Generator Based on Moser's Deformation Method", *Numer. Meth. Part. Diff. Eq.* 10, 21 (1994).
[32] G. Liao, G. de la Pena, "A deformation method for moving grid generation", *proceedings, 8th International Meshing Roundtable*, pp. 155-162. South Lake Tahoe, CA, October, (1999).
[33] D. Fleitas, J. Xue, J. Liu and G. Liao, Least-squares finite element adaptive grid deformation in a non-linear time dependent problem. In Advances in applied mathematics (2004 SIAM GATORS), Gainsville, Florida, 2004.
[34] F. Liu, S. Ji, and G. Liao, An adaptive grid method and its application to steady Euler flow calculations, SIAM J. Sci. Comput. 20, 811-825, 1998.
[35] X. Han, C. Xu, and J. L. Prince, A 2D Moving Grid Geometric Deformable Model, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (CVPR2003) , June, 2003, pp. I: 153-160.
[36] X. Cai, D. Fleitas, B. Jiang, and G. Liao, Adaptive grid generation based on least-squares finite-element method. *Computers and Mathematics with Applications*, 48, 2004.
[37] J. Xue, Moving grids by the deformation method, Dissertation, 2004.
[38] W. Morris, A meshfree adaptive numerical method, Dissertaion, 2004.
[39] D. Fleitas, The least-square finite element method for grid deformation and meshfree applications, Dissertation, 2005.
[40] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming and P. Krysl, Meshless Methods: an overview and recent developments, Computer methods in applied mechanics and engineering, 139, 3-47, 1996.
[41] B. Jiang, *The Least-Squares Finite Element Method: Theory and Applications in computational Fluid Dynamics and Electromagnetics*. Springer, Berlin, 1998.
[42] M. Grajewski, M. Koster, S. Kilian and S. Turek, Numerical Analysis and Practical Aspects of a Robust and Efficient Grid Deformation Method in the Finite Element Context, preprint, 2005.
[43] C. L. Chang and M. Gunzburger, A finite element method for first order elliptic systems in three dimensions. Appl. Math. Comput. 23, 135-146, 1987