INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2004; **45**:53–78 (DOI: 10.1002/fld.645)

A projection method for solving incompressible viscous flows on domains with moving boundaries

Hua Pan[*,1,†], Lun Sheng Pan[2], Diao Xu[2], Teng Yong Ng[3] and Gui Rong Liu[4]

[1] *University of Chicago, Chicago, IL, U.S.A*
[2] *Institute of High Performance Computing, Singapore 117528, Singapore*
[3] *Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore*
[4] *National University of Singapore, Singapore 119260, Singapore*

SUMMARY

In this paper, a projection method is presented for solving the flow problems in domains with moving boundaries. In order to track the movement of the domain boundaries, arbitrary-Lagrangian–Eulerian (ALE) co-ordinates are used. The unsteady incompressible Navier–Stokes equations on the ALE co-ordinates are solved by using a projection method developed in this paper. This projection method is based on the Bell's Godunov-projection method. However, substantial changes are made so that this algorithm is capable of solving the ALE form of incompressible Navier–Stokes equations. Multi-block structured grids are used to discretize the flow domains. The grid velocity is not explicitly computed; instead the volume change is used to account for the effect of grid movement. A new method is also proposed to compute the freestream capturing metrics so that the geometric conservation law (GCL) can be satisfied exactly in this algorithm. This projection method is also parallelized so that the state of the art high performance computers can be used to match the computation cost associated with the moving grid calculations. Several test cases are solved to verify the performance of this moving-grid projection method. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: ALE co-ordinate; projection method; moving boundary; moving grid; unsteady incompressible flow; parallel computation

1. INTRODUCTION

Flow problems with moving boundaries can be encountered in a variety of applications. Typical ones include free surface induced flow, phase change and fluid–structure (or solid) interactions. In these problems, the flow domain is unsteady and so are the applied boundary conditions. It is well known that the solution of fluid flow can be best accomplished by using boundary-conforming co-ordinate transformation. Generally, it is also desired to use boundary-conforming co-ordinate transformation to facilitate the interface tracking. Therefore, the non-Eulerian co-ordinate system has to be applied to exactly represent the moving boundaries of the fluid domain so that accurate solutions can be obtained. The non-Eulerian co-ordinate

*Correspondence to: Hua Pan, University of Chicago, 5640 S. Ellis Avenue, RI-473, Chicago, IL 60637, U.S.A.
†E-mail: huapan@flash.uchicago.edu

Received 26 May 2002
Revised 20 March 2003

system can also be applied on the adaptive grid calculations, where the accuracy of the flow solution can be improved without increase of grid density. However, it is not necessary to use moving grid during grid adapting for steady-flow calculation, where the iterative solution procedure can continue without taking into account the change of grid position. For unsteady flow, the movement of grid points has to be properly represented in the numerical algorithms.

Owing to the movement of co-ordinate system, an additional conservation equation results relating the change of elementary control volume to the co-ordinate frame velocity. This relation was named by Thomas *et al.* [1] as the geometric conservation law (GCL). This conservation equation must be satisfied simultaneously with other conservation equations, such as mass, momentum and other quantities of fluid flow in solving the moving boundary problem. A good review regarding the moving grid can be found in Reference [2], where the GCL for both static and dynamic meshes was identified in detail as the surface and volume conservation law, respectively. The surface conservation law ensures that the control volume is closed by its surfaces and volume conservation law guarantees the conservation of volume change for the moving grid. It has been demonstrated in Reference [3] that an artificial mass source would be introduced if the system fails to satisfy the GCL.

Some work has been presented in the literature to simulate flows with moving boundaries. Shyy *et al.* [4] solved a two-dimensional phase change problem on moving structured grid where the GCL was enforced by solving the transport equation of the Jacobian. Demirdžić *et al.* [5] used an SIMPLE-like method to simulate the flow in a channel with moving indentation. In his method, the grid velocity was carefully designed in order to satisfy the GCL. Zhang *et al.* [2] followed this approach to develop the formula for evaluating the grid velocity (volume change) for both the two- and three-dimensional problems. They also presented and analysed the general finite difference and finite volume formulations for the moving grid calculation. However, only the tetrahedron cell was used in their formulation for three-dimensional problem. Obayashi [6] analysed freestream capturing metrics on moving co-ordinates for both the finite volume and finite difference formulation, where the Vinokur's formula [7] was used to calculate the value of the metrics. However, the author found that this formula did not satisfy the GCL exactly.

Bell *et al.* [8] proposed a Godunov-projection method for solving the unsteady incompressible flows. The cell Reynolds number restriction was removed or enlarged in this method. Subsequently, this method was developed and studied by other researchers. Lai *et al.* [9] used this method to simulate zero Mach number combustion and Bell *et al.* [10] extended it to solve incompressible variable density problems. Brown *et al.* [10] studied the effect of coarse grids on this method. Almgren *et al.* [12] suggested using the approximate projection instead of exact projection to allow use of a fast linear solver. Pan *et al.* [13] applied this method on overlapping grids and parallelized this algorithm. Trebotich *et al.* [14] made an attempt to extend the Godunov-projection method for solving a specified two-dimensional incompressible flow with moving boundaries. In their method, the velocity is divided into two components, potential and vortical components, in order to eliminate the inhomogeneities in the incompressibility constraint introduced by the presence of moving boundaries. However, the Poisson equation has to be solved five times in a single time step.

In this work, an attempt is made to solve the moving boundary problem on a general three-dimensional computational domain by extending the Bell's Godunov-projection method so that the advantages of the Godunov-projection method can be retained. The general formulation of the unsteady viscous incompressible Navier–Stokes equations on the arbitrary-Langrangian–

Eulerian (ALE) co-ordinate system is presented first. Then the freestream capturing metrics are examined and a formula for computing the Jacobian is proposed so that the volume conservation law can be guaranteed. The moving grid projection method is presented to solve the ALE formulation of incompressible flow. The parallel algorithm based on the grid partition is also presented and implemented in a general three-dimensional code. Several test cases are solved to demonstrate the capability of this method.

2. GOVERNING EQUATIONS

The governing equations for the unsteady incompressible flow on the ALE co-ordinate system can be categorized into conservation laws for several quantities, such as surface, volume, mass, momentum and so on. After mapping the physical space into the computational space, the governing equations in the computational space can be written as follows:

- surface conservation law:

$$\frac{\partial}{\partial \xi^j}(\mathbf{S}^j \cdot \mathbf{a}) = 0 \tag{1}$$

- volume conservation law:

$$\left.\frac{\partial J}{\partial t}\right|_{\xi} - \frac{\partial}{\partial \xi^j}(\mathbf{S}^j \cdot \mathbf{u}_b) = 0 \tag{2}$$

- mass conservation law:

$$\frac{\partial}{\partial \xi^j}(\mathbf{S}^j \cdot \mathbf{u}) = 0 \tag{3}$$

- momentum conservation law:

$$\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\xi} + \frac{1}{J}[\mathbf{S}^j \cdot (\mathbf{u} - \mathbf{u}_b)]\frac{\partial \mathbf{u}}{\partial \xi^j} = \frac{\varepsilon}{J}\frac{\partial}{\partial \xi^j}\left(\mathbf{S}^j \cdot \frac{1}{J}\mathbf{S}^k \frac{\partial \mathbf{u}}{\partial \xi^k}\right) - \frac{1}{J}\frac{\partial}{\partial \xi^j}(\mathbf{S}^j p) + \mathbf{F} \tag{4}$$

Here $\boldsymbol{\xi} = (\xi^0, \xi^1, \xi^2)^{\mathrm{T}}$ denotes the computational co-ordinates and $\mathbf{x} = (x^0, x^1, x^2)^{\mathrm{T}}$ denotes the Cartesian co-ordinates in the physical space. $\mathbf{S}^j = \nabla \xi^j$ and $J = \det(\Phi)$, where Φ is the mapping between the physical and computational space. $\mathbf{a}$ is the arbitrary constant vector and $\mathbf{u}$ is the flow velocity. $\mathbf{u}_b$ is the co-ordinate velocity or grid velocity. The pressure is represented by p and $\mathbf{F}$ denotes the source term. ε is the kinematic viscosity. The conservation laws for other fluid quantities, such as temperature and concentration, take similar forms. In the ALE formulations, the convection velocity should be the relative velocity $(\mathbf{u} - \mathbf{u}_b)$ instead of flow velocity $\mathbf{u}$ and the time derivative is computed on the computational space, such as $\partial J/\partial t|_{\xi}$ and $\partial \mathbf{u}/\partial t|_{\xi}$. It can be easily observed that the conventional Navier–Stokes equation can be recovered in case of $\mathbf{u}_b = 0$. It is verified that after applying Equation (2) in Equation (4), the conventional Navier–Stokes equation can also be recovered. In order to simplify the formulation, we use Δ_{ξ} and ∇_{ξ} denote the Laplacian and gradient operator in the computational space, respectively, from now on.

The initial and boundary conditions are application-specific. For different flow calculations, the appropriate initial and boundary conditions have to be determined so that the numerical calculation can be initiated. It should be pointed out that it is not straightforward to specify the divergence-free initial velocity field and it is even harder to decide the initial pressure distribution. For the boundary conditions, either Dirichlet or Neumann-type condition, can be prescribed on the domain boundaries.

Bell *et al.* [8, 15] proposed the Godunov-projection method, which has been successfully applied to solve the unsteady incompressible Navier–Stokes equations on a Eulerian co-ordinate system. Various kinds of flow problems have been addressed by using this method, such as variable density flow and zero Mach number combustion. In this paper, an attempt is made to extend this method so that the Navier–Stokes equations on the non-Eulerian co-ordinate system, Equations (3) and (4), can also be solved. However, the geometric conservation laws, equation Equations (1) and (2), must be satisfied in the numerical scheme in order to avoid the existence of artificial source term as stated by Demirdžić *et al.* [3]. Before the numerical procedure is presented, the calculation of transformation metrics is reviewed in the next section so that the geometric conservation law can be satisfied in terms of freestream capturing metrics.

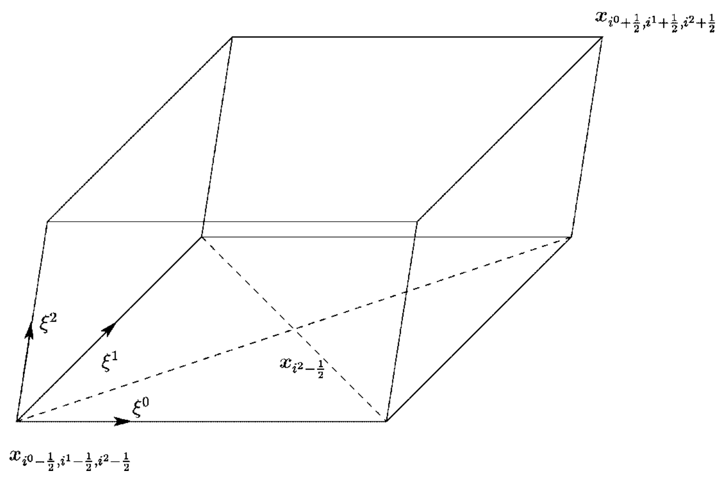
3. FREESTREAM CAPTURING METRICS

In this section, the transformation metrics $\mathbf{S}^j$ and the Jacobian J are approximated in terms of finite volume discretization. In structured grid, $\Delta\xi^0$, $\Delta\xi^1$ and $\Delta\xi^2$ represent the constant step lengths of the unit computational domain. In the finite volume context, the cell surface area vector can be evaluated as $\mathbf{S}^j\Delta\xi^m\Delta\xi^n$, where $j \neq m \neq n$, and the cell volume can be calculated as $J\Delta\xi^0\Delta\xi^1\Delta\xi^2$. Therefore, the surface area vector and the cell volume must be evaluated so that the geometric conservation law can be satisfied.

In order to simplify the notation of the equations, the following rules are applied hereafter:

- The subscript index (i^0, i^1, i^2) is used to refer cell and the superscript n is used to refer the time level.
- The surface has one half indexed, such as $(i^0 + \frac{1}{2}, i^1, i^2)$.
- The vertex is labelled as $(i^0 + \frac{1}{2}, i^1 + \frac{1}{2}, i^2 + \frac{1}{2})$.
- The implied indexing scheme is applied, where the subscripts and superscripts take the default values if they are missing in the expressions. For example, $\mathbf{u}_{i^0+1/2}$ refers to $\mathbf{u}^n_{i^0+1/2,i^1,i^2}$ and $\mathbf{x}_{i^0+1/2,i^1+1/2,i^2+1/2}$ is the co-ordinates of the vertex $(i^0 + \frac{1}{2}, i^1 + \frac{1}{2}, i^2 + \frac{1}{2})$.

For the hexahedral cell in three-dimensional space, the surface is not unique since it has four vertices and only three vertices's are sufficient to define a plane. Therefore, there are various ways to approximate the cell surface by using the different triangle divisions and various formula can be derived consequently. Figure 1 shows a hexahedral cell of the grid. In order to satisfy the surface conservation law as presented in Equation (1), the triangulation presented below is applied. First, a surface centre point $\mathbf{x}_{i^j+\frac{1}{2}}$ is introduced by averaging of co-ordinates of the four vertexes and the surface area vector is computed by summing the area vectors of four triangles, which are constructed by connecting the central point to the



$x_{i^0+\frac{1}{2},i^1+\frac{1}{2},i^2+\frac{1}{2}}$
ξ^2
ξ^1
ξ^0
$x_{i^2-\frac{1}{2}}$
$x_{i^0-\frac{1}{2},i^1-\frac{1}{2},i^2-\frac{1}{2}}$

Figure 1. Geometry of hexahedral cell.

four vertices. Therefore, the surface area vector can be computed as below:

$$\mathbf{S}_{i^j+1/2} = \frac{1}{2}(\mathbf{x}_{i^m+1/2,i^n+1/2} - \mathbf{x}_{i^m-1/2,i^n-1/2})|_{i^j+1/2} \times (\mathbf{x}_{i^m-1/2,i^n+1/2} - \mathbf{x}_{i^m+1/2,i^n-1/2})|_{i^j+1/2} \tag{5}$$

where $m = (j+1)\%3$ and $n = (j+2)\%3$. Similar formulas can be derived for other surfaces. In fact, the surface area vector computed by using this triangulation is the average of surface area vectors by using other possible triangulations. Actually, this equation is commonly used in the finite volume formulation and it can be easily verified that this formula does satisfy the surface conservation law (SCL) as shown in Equation (1).

Several formulae have been proposed for estimating the volume of hexahedral cell based on different approximations. However, the volume conservation law (VCL) as in Equation (2) must be satisfied. Figure 2 shows the positions of a hexahedral cell before and after its movement.

Therefore, the VCL can be expressed as below:

$$\Delta V = \sum_{j=0,1,2} (\mathrm{d}V_{i^j+1/2} - \mathrm{d}V_{i^j-1/2}) \tag{6}$$

where the volume change of hexahedral cell, ΔV, must equal to the sum of the volumes scanned by its surfaces during the movement, $\mathrm{d}V_{i^j\pm1/2}$.

The following equation was derived by using the concept of the equivalent plane:

$$V = \frac{1}{3}\sum\nolimits_{j=0,1,2} (\mathbf{S}_{i^j-1/2})\cdot(\mathbf{x}_{i^0+1/2,i^1+1/2,i^2+1/2} - \mathbf{x}_{i^0-1/2,i^1-1/2,i^2-1/2}) \tag{7}$$

Obayashi [6] applied this formula to calculate both the cell volume and the volume changes and claimed that this formula satisfied the VCL as stated in Equation (6). However, after carefully examination, it is found that this formula fails to ensure the VCL exactly. The author believes that the volume of hexahedral cell must be computed enough accurately in order to guarantee the VCL. The following way of dividing hexahedral cell is used. Besides the six surface centre points, a cell centre point $\mathbf{x}_{i^0,i^1,i^2}$ is also introduced where its co-ordinates

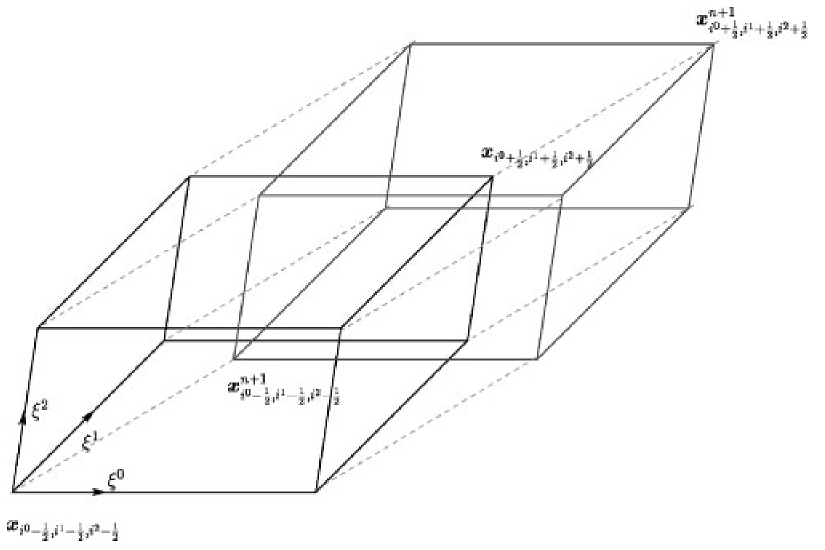

$x^{n+1}_{i^0+\frac{1}{2},i^1+\frac{1}{2},i^2+\frac{1}{2}}$
$x_{i^0+\frac{1}{2},i^1+\frac{1}{2},i^2+\frac{1}{2}}$
$x^{n+1}_{i^0-\frac{1}{2},i^1-\frac{1}{2},i^2-\frac{1}{2}}$
ξ^2
ξ^1
ξ^0
$x_{i^0-\frac{1}{2},i^1-\frac{1}{2},i^2-\frac{1}{2}}$

Figure 2. Hexahedral cell movement.

is average of the cell's eight vertexes. Therefore, the hexahedral cell can be divided into 24 tetrahedrons by connecting the cell centre point to the vertexes and the surface centre points, where each tetrahedron is constructed by using the cell centre point, a surface centre point and two vertexes. This cell division is compatible with the surface division presented above. Since the tetrahedron is the elementary shape in the three-dimensional space, the computed volume of the hexahedral cell is unique and exact. After simplification of the summation of the volume of the 24 tetrahedrons, the following formula is obtained for computing the volume of the hexahedral cell:

$$V = \frac{1}{6}\sum_{j=0,1,2}[(\mathbf{S}_{i^j+1/2} + \mathbf{S}_{i^j-1/2})\cdot(\mathbf{x}_{i^j+1/2} - \mathbf{x}_{i^j-1/2})] \tag{8}$$

It has been verified that this formula guarantees the VCL exactly when both the cell volumes and the volume changes are evaluated by using this formula. Its disadvantage may be that this formula is more costly compared to Equation (7). However, considering the fact that all the surface area vectors have to be computed in the numerical procedure, the cost for computing the volume is not too high since the surface area vectors are already available. In the first numerical test case, it is demonstrated that the freestream flow can be captured by using this formula to calculate the volume and volume change.

4. NUMERICAL PROCEDURE

Projection methods, originally developed by Chorin [16], are fractional step methods based on the decomposition theory, which says that any vector **V** can be uniquely decomposed into a divergence-free component $\mathbf{V}_\mathrm{d}$ and the gradient of a scalar field $\nabla\phi$. In mapped grid, the decomposition is defined in terms of the computational space, i.e.

$$\mathbf{V} = \mathbf{V}_\mathrm{d} + \frac{1}{J}\frac{\partial}{\partial \xi^j}(\mathbf{S}^j \phi)$$

Therefore, the projection operator $\mathbf{P}$ can be defined such that $\mathbf{V}_\mathrm{d} = \mathbf{PV}$ and $(1/J)\partial/\partial\xi^j(\mathbf{S}^j\phi) = (\mathbf{I} - \mathbf{P})\mathbf{V}$. The Godunov-projection method was proposed by Bell *et al.* [15], where the Godunov procedure was applied in order to provide a robust discretization for the convection term so that the cell Reynolds number restriction can be removed. This method is extended in this paper to solve the ALE formulation of incompressible Navier–Stokes equations.

Using the projection operator defined above, the Navier–Stokes equations (3) and (4) can be written in the equivalent form

$$\mathbf{u}_t = \mathbf{P}\left[\frac{\varepsilon}{J}\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial \mathbf{u}}{\partial \xi^k}\right) - \frac{1}{J}[\mathbf{S}^j\cdot(\mathbf{u} - \mathbf{u}_b)]\frac{\partial \mathbf{u}}{\partial \xi^j} + \mathbf{F}\right] \tag{9}$$

and the pressure can be evaluated as the gradient component of the projected vector field, i.e.

$$\frac{1}{J}\frac{\partial}{\partial\xi^j}(\mathbf{S}^j p) = (\mathbf{I} - \mathbf{P})\left[\frac{\varepsilon}{J}\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial \mathbf{u}}{\partial \xi^k}\right) - \frac{1}{J}[\mathbf{S}^j\cdot(\mathbf{u} - \mathbf{u}_b)]\frac{\partial \mathbf{u}}{\partial \xi^j} + \mathbf{F}\right] \tag{10}$$

In this section, the temporal discretization is presented first followed by the spatial discretization. Finally, the projection is briefly introduced.

4.1. Temporal discretization

In the Godunov-projection method, the second-order Crank–Nicolson approximation is used:

$$\begin{aligned}\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \mathbf{P}\Bigg[&\left(\frac{\varepsilon}{2J}\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial \mathbf{u}}{\partial \xi^k}\right)\right)^{n+1/2} \\ &- \left(\frac{1}{J}[\mathbf{S}^j\cdot(\mathbf{u} - \mathbf{u}_b)]\frac{\partial \mathbf{u}}{\partial \xi^j}\right)^{n+1/2} + \mathbf{F}^{n+1/2}\Bigg]\end{aligned} \tag{11}$$

However, the linear algebra problem associated with this equation could be extremely costly because of the non-local behaviour of the projection. Bell gave an alternative where the fractional step method is used. An intermediate velocity field is computed by solving the momentum equation along with the currently available pressure field. Then the new divergence-free velocity field is obtained by projecting the intermediate velocity field. There are two options for computing the intermediate velocity field associated with two projection methods, the incremental form and pressure form projection, respectively. According to Rider's analysis [17], the pressure form projection is more robust. Therefore, the intermediate velocity $\mathbf{u}^*$ is computed by using following equations:

$$\left[1-\frac{\varepsilon\Delta t}{2J}\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial}{\partial\xi^k}\right)\right]^{n+1}\hat{\mathbf{u}}^*=\left[1+\frac{\varepsilon\Delta t}{2J}\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial}{\partial\xi^k}\right)\right]^{n}\mathbf{u}^n$$
$$-\Delta t\left(\frac{1}{J}[\mathbf{S}^j\cdot(\mathbf{u}-\mathbf{u}_b)]\frac{\partial\mathbf{u}}{\partial\xi^j}\right)^{n+1/2}$$
$$-\Delta t\left(\frac{1}{J}\frac{\partial}{\partial\xi^j}(\mathbf{S}^j p)\right)^{n-1/2}+\frac{\Delta t}{2}\mathbf{F}^{n+1/2} \qquad (12)$$

$$\mathbf{u}^*=\hat{\mathbf{u}}^*+\left(\frac{\Delta t}{J}\frac{\partial}{\partial\xi^j}(\mathbf{S}^j p)\right)^{n-1/2} \qquad (13)$$

The pressure field obtained from the previous time step is substituted here as the rough approximation of the current pressure field since the current pressure field is still not available. The reason to use Equation (13) is to put back the pressure gradient into the intermediate velocity field so that the error resulting from previous time step is combined into the vector field, which would be projected afterwards. Therefore, accumulation of numerical error can be avoided. Once the intermediate velocity field is obtained, the projection can be performed on the intermediate velocity to get the new flow field

$$\mathbf{u}^{n+1}=\mathbf{P}(\mathbf{u}^*) \qquad (14)$$

$$\left(\frac{1}{J}\frac{\partial}{\partial\xi^j}(\mathbf{S}^j p)\right)^{n+1/2}=\frac{1}{\Delta t}(\mathbf{I}-\mathbf{P})\mathbf{u}^* \qquad (15)$$

After substituting $\mathbf{u}^*$ in Equations (14) and (15), it can be found that the combination of Equations (9) and (10) can be constructed. More detailed analysis regarding this fractional scheme can be found in References [8, 10].

It should be pointed out this pressure form of projection is equivalent to the pressure Poisson equation method as indicated by Rider [17]. The pressure Poisson equation is defined by taking the divergence of the momentum equation and evoking the solenoidal condition. Hirt [18] suggested a modification to control the growth of error of pressure Poisson equation, where the current velocity field is put back in the right-hand side. The pressure Poisson equation obtained is as following:

$$\frac{\partial}{\partial\xi^i}\left(\mathbf{S}^i\cdot\frac{1}{J}\frac{\partial}{\partial\xi^j}(\mathbf{S}^j p)\right)=\frac{\partial}{\partial\xi^i}\left(\mathbf{S}^i\cdot\left(\frac{1}{J}\varepsilon\frac{\partial}{\partial\xi^j}\left(\mathbf{S}^j\cdot\frac{1}{J}\mathbf{S}^k\frac{\partial\mathbf{u}}{\partial\xi^k}\right)\right.\right.$$
$$\left.\left.-\frac{1}{J}[\mathbf{S}^j\cdot(\mathbf{u}-\mathbf{u}_b)]\frac{\partial\mathbf{u}}{\partial\xi^j}+\mathbf{F}\right)\right) \qquad (16)$$

In fact, the Possion equation associated with the pressure form of projection, $\Delta\phi=\nabla\cdot(\mathbf{u}^*/\Delta t)$ takes the same form as this equation. In conclusion, the pressure form of projection is equivalent to the pressure Poisson equation. The boundary condition applied for this pressure form of projection must be selected carefully. Gresho [19] gave excellent comments

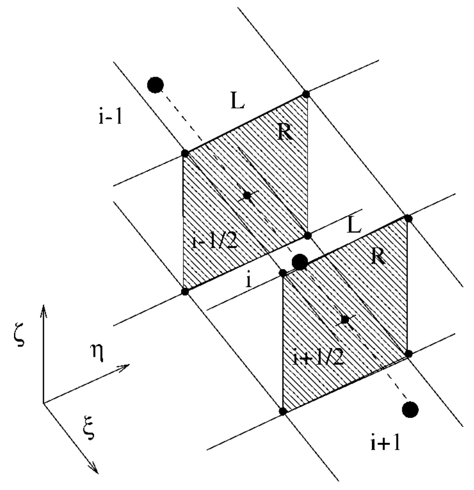


Figure 3. Extrapolated surface velocity.

regarding this issue. They recommended deriving the boundary condition from the normal motion equation at the boundary. This idea is followed in the work to derive the boundary condition at the moving wall boundary.

4.2. Convection discretization

The convection term used in Equation (12) is evaluated at the half time level in order to make the scheme second-order accurate in time. Therefore, the velocity field at $t^{n+1/2}$ must be approximated. The extrapolation procedure used by Bell [15] is followed in this paper to estimate the velocity field at half time level, where the velocity at the cell surface is extrapolated. The MAC projection is performed to make sure the extrapolated velocity field satisfies the continuity equation. It should be pointed out that the grid at the half time level must be calculated since the grid is also time dependent. A simple assumption is made here that every grid vertex moves with constant velocity during the time step from t^n to t^{n+1}. Therefore, the middle grid can be easily obtained once the grid at next time level is generated. The extrapolation procedure applied in this method is presented in detail.

Using the Taylor-series expansion, the surface-based velocity variables can be extrapolated from cells at both sides of the surface as below:

$$\mathbf{u}_{i^j+1/2}^{L,n+1/2} = \mathbf{u} + \frac{\Delta\xi^j}{2}\frac{\partial \mathbf{u}}{\partial \xi^j} + \frac{\Delta t}{2}\frac{\partial \mathbf{u}}{\partial t}\bigg|_{\xi} \tag{17}$$

$$\mathbf{u}_{i^j+1/2}^{R,n+1/2} = \mathbf{u}_{i^j+1} - \frac{\Delta\xi^j}{2}\left(\frac{\partial \mathbf{u}}{\partial \xi^j}\right)_{i^j+1} + \frac{\Delta t}{2}\left(\frac{\partial \mathbf{u}}{\partial t}\bigg|_{\xi}\right)_{i^j+1} \tag{18}$$

Figure 3 shows the location of left and right surface velocity vectors. The time derivative is replaced by using the momentum equation (4) and the following equations can be obtained:

$$\mathbf{u}_{i^j+1/2}^{L,n+1/2} = \mathbf{u} + \left[\frac{\Delta\xi^j}{2} - \frac{\Delta t}{2J}\mathbf{S}^j\cdot(\mathbf{u}-\mathbf{u}_b)\right]\frac{\partial \mathbf{u}}{\partial \xi^j}$$

$$-\frac{\Delta t}{2J}\sum_{k\neq j}(\mathbf{S}^k\cdot(\mathbf{u}-\mathbf{u}_b))\frac{\partial \mathbf{u}}{\partial \xi^k}$$

$$+\frac{\varepsilon\Delta t}{2}\Delta_\xi \mathbf{u} - \frac{\Delta t}{2}\nabla_\xi p + \frac{\Delta t}{2}\mathbf{F} \tag{19}$$

$$\mathbf{u}_{i^j+1/2}^{R,n+1/2} = \mathbf{u}_{i^j+1} - \left[\frac{\Delta\xi^j}{2} + \left(\frac{\Delta t}{2J}\mathbf{S}^j\cdot(\mathbf{u}-\mathbf{u}_b)\right)_{i^j+1}\right]\left(\frac{\partial \mathbf{u}}{\partial \xi^j}\right)_{i^j+1}$$

$$-\frac{\Delta t}{2J}\left(\sum_{k\neq j}(\mathbf{S}^k\cdot(\mathbf{u}-\mathbf{u}_b))\frac{\partial \mathbf{u}}{\partial \xi^k}\right)_{i^j+1}$$

$$+\frac{\varepsilon\Delta t}{2}(\Delta_\xi \mathbf{u})_{i^j+1} - \frac{\Delta t}{2}(\nabla_\xi p)_{i^j+1} + \frac{\Delta t}{2}(\mathbf{F})_{i^j+1} \tag{20}$$

where Δ_ξ refers to the Laplacian operator and ∇_ξ the gradient operator in the computational space. In these equations, the normal and transverse convection terms are separately expressed. The pressure term in these equation should be removed. According to Bell *et al.* [20], the usage of lagged pressure term would introduce a non-linear instability when the CFL number is larger than 0.5. Therefore, the pressure term is suppressed in the following equations and the MAC projection is performed instead to recover the extrapolated solenoidal velocity field, which is introduced later in this section.

Colella [21] developed multi-dimensional upwind approximation where the normal and transverse derivative terms are differentiated separately in order to correctly capture the discontinuity in the multi-dimensional space. The extrapolation of surface velocities as shown in Equations (19) and (20) is accomplished in a series of steps. The approximation of the space derivative $\partial\mathbf{u}/\partial\xi^j$, denoted as $\Delta_{\xi^j}\mathbf{u}$, can be computed by using various limiters, such as minmod and super-bee limiters, etc. We choose the following limiter:

$$Q(r) = \max\left[0, \min\left(2, 2r, \frac{1+r}{2}\right)\right] \tag{21}$$

where r is the ratio between the left difference $\delta^{\mathrm{L}} = \mathbf{u} - \mathbf{u}_{i^j-1}$ and right difference $\delta^{\mathrm{R}} = \mathbf{u}_{i^j+1} - \mathbf{u}$, i.e. $r = \delta^{\mathrm{L}}/\delta^{\mathrm{R}}$. Therefore, $\Delta_{\xi^j}\mathbf{u} = Q(r)\delta^{\mathrm{R}}$. The grid velocity in the convection velocity is combined into the volume change of the cell in the ξ^j direction, ΔV^j. After the normal convection is approximated in an upwind fashion, the surface velocity extrapolated from both sides can be estimated as below:

$$\hat{\mathbf{u}}^{L,n+1/2}_{i^j+1/2} = \mathbf{u} + \left[\frac{\Delta\xi^j}{2} - \max\left(0, \frac{\Delta t}{2J}\bar{u}^j - \frac{1}{J}\Delta V^j\right)\right]\Delta_{\xi^j}\mathbf{u}$$
$$- \sum_{k\neq j}\left(\frac{\Delta t}{2J}\bar{u}^k - \frac{1}{J}\Delta V^k\right)\widehat{\Delta_{\xi^k}\mathbf{u}}$$
$$+ \frac{\varepsilon\Delta t}{2}\Delta_\xi\mathbf{u} + \frac{\Delta t}{2}\mathbf{F} \tag{22}$$

$$\hat{\mathbf{u}}^{R,n+1/2}_{i^j+1/2} = \mathbf{u}_{i^j+1} - \left[\frac{\Delta\xi^j}{2} + \min\left(0, \left(\frac{\Delta t}{2J}\bar{u}^j - \frac{1}{J}\Delta V^j\right)_{i^j+1}\right)\right](\Delta_{\xi^j}\mathbf{u})_{i^j+1}$$
$$-\sum_{k\neq j}\left(\frac{\Delta t}{2J}\bar{u}^k - \frac{1}{J}\Delta V^k\right)_{i^j+1}(\widehat{\Delta_{\xi^k}\mathbf{u}})_{i^j+1}$$
$$+ \frac{\varepsilon\Delta t}{2}(\Delta_\xi\mathbf{u})_{i^j+1}\frac{\Delta t}{2}\mathbf{F}_{i^j+1} \tag{23}$$

where $\bar{u}^j = \mathbf{S}^j\cdot\mathbf{u}$. The volume change ΔV^j is evaluated by averaging the volumes scanned by the surface $\mathbf{S}^j_{i^j+1/2}$ and $\mathbf{S}^j_{i^j-1/2}$, respectively, i.e. $\Delta V^j = 1/2(\Delta V^j_{i^j+1/2} + \Delta V^j_{i^j-1/2})$. Here the scanned volume is computed by using the proposed formula (8) presented in last section. $\widehat{\Delta_{\xi^k}\mathbf{u}}$ represents the transverse convection derivative, which is approximated by using the following upwind scheme:

$$\widehat{\Delta_{\xi^k}\mathbf{u}} = \mathbf{u} - \mathbf{u}_{i^k-1} + \left[\frac{\Delta\xi^k}{2} - \frac{1}{J}\left(\frac{\Delta t}{2}\bar{u}^k - \Delta V^k\right)\right](\Delta_{\xi^k}\mathbf{u} - \Delta_{\xi^k}\mathbf{u}_{i^k-1})$$
$$\text{if } \left(\frac{\Delta t}{2}\bar{u}^k - \Delta V^k\right) \geqslant 0 \tag{24}$$

$$\widehat{\Delta_{\xi^k}\mathbf{u}} = \mathbf{u}_{i^k+1} - \mathbf{u} - \left[\frac{\Delta\xi^k}{2} + \frac{1}{J}\left(\frac{\Delta t}{2}\bar{u}^k - \Delta V^k\right)\right](\Delta_{\xi^k}\mathbf{u}_{i^k+1} - \Delta_{\xi^k}\mathbf{u})$$
$$\text{if } \left(\frac{\Delta t}{2}\bar{u}^k - \Delta V^k\right) < 0 \tag{25}$$

The Laplacian derivative term can be evaluated using a standard scheme. After the extrapolation, the ambiguities of the surface velocity are solved by using an upwind averaging procedure based on the Roe's Riemann solver. From now on, we suppress the superscript $n+\frac{1}{2}$ in the favour of using simple equations. First the advection velocity is computed:

$$\tilde{u}^{\text{adv}}_{i^j+1/2} = \tfrac{1}{2}[\tilde{u}^{\text{L}}_{i^j+1/2} + \tilde{u}^{\text{R}}_{i^j+1/2} - \text{sign}(\tilde{u}^j_{\text{avg}})(\tilde{u}^{\text{R}}_{i^j+1/2} - \tilde{u}^{\text{L}}_{i^j+1/2})] \tag{26}$$

where

$$\tilde{u}^{\mathrm{L}}_{i^j+1/2} = \frac{\Delta t}{2}(\mathbf{S}^j \cdot \hat{\mathbf{u}}^{\mathrm{L}})_{i^j+1/2} - \Delta V^j_{i^j+1/2}$$

$$\tilde{u}^{\mathrm{R}}_{l^j+1/2} = \frac{\Delta t}{2}(\mathbf{S}^j \cdot \hat{\mathbf{u}}^{\mathrm{R}})_{i^j+1/2} - \Delta V^j_{i^j+1/2}$$

and $\tilde{u}^j_{\mathrm{avg}} = \frac{1}{2}(\tilde{u}^{\mathrm{L}}_{l^j+1/2} + \tilde{u}^{\mathrm{R}}_{l^j+1/2})$. Then, the value of the surface velocity is evaluated in favour of the upwind direction:

$$\hat{\mathbf{u}}_{i^j+1/2} = \tfrac{1}{2}[\hat{\mathbf{u}}^{\mathrm{L}}_{i^j+1/2} + \hat{\mathbf{u}}^{\mathrm{R}}_{i^j+1/2} - \mathrm{sign}(\tilde{u}^{\mathrm{adv}}_{i^j+1/2})(\hat{\mathbf{u}}^{\mathrm{R}}_{i^j+1/2} - \hat{\mathbf{u}}^{\mathrm{L}}_{i^j+1/2})]$$

Since the pressure term is missing in the extrapolation, the surface velocity field obtained so far is not divergence free. The MAC projection is performed on this field to recover the solenoidal velocity field

$$\mathbf{u}_{i^j+1/2} = \mathbf{P}^{\mathrm{MAC}}(\hat{\mathbf{u}}_{i^j+1/2})$$

Finally, the convection term in Equation (12) is calculated by using the difference scheme as follows:

$$\Delta t\left(\frac{1}{J}[\mathbf{S}^j \cdot (\mathbf{u} - \mathbf{u}_b)]\frac{\partial \mathbf{u}}{\partial \xi^j}\right) = \frac{1}{J}\sum_j \left[\tilde{u}^{\mathrm{adv}}\left(\frac{\mathbf{u}_{i^j+1/2} - \mathbf{u}_{i^j-1/2}}{\Delta \xi^j}\right)\right] \tag{27}$$

where the advection velocity takes the value of the average of advection velocities on both $j \pm 1/2$ surface of the cell, i.e.

$$\tilde{u}^{\mathrm{adv}} = 1/2(\tilde{u}^{\mathrm{adv}}_{i^j+1/2} + \tilde{u}^{\mathrm{adv}}_{i^j-1/2})$$

and the advection velocity on the $i^j + 1/2$ surface is computed:

$$\tilde{u}^{\mathrm{adv}}_{i^j+1/2} = \Delta t(\mathbf{S}^j \cdot \mathbf{u})_{i^j+1/2} - \Delta V_{i^j+1/2}$$

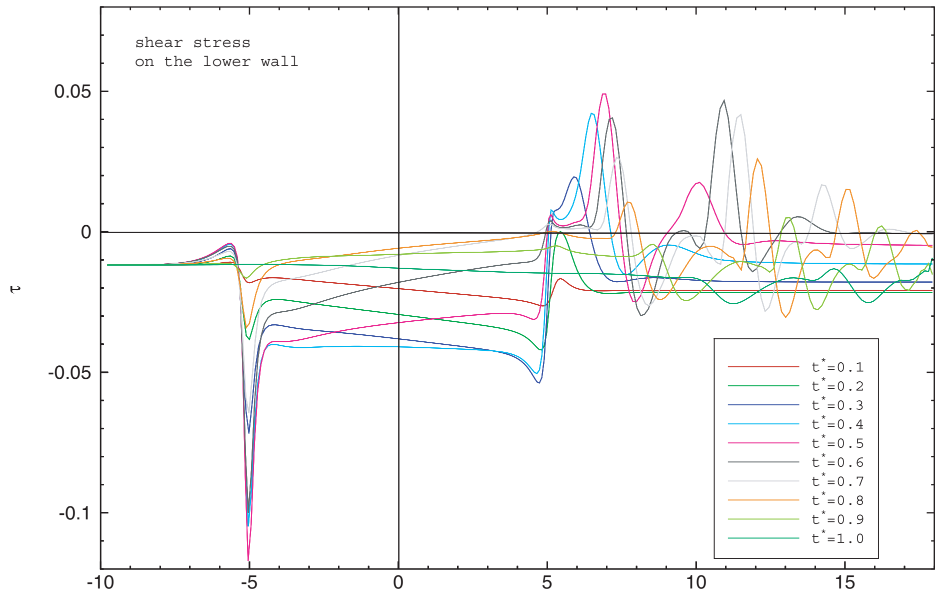
where the volume change $\Delta V_{i^j+1/2}$ is between the current grid position and the grid at the next time level.

The time step of this second-order Godunov method is restricted by the CFL condition for the sake of stability. The criteria for the time step is

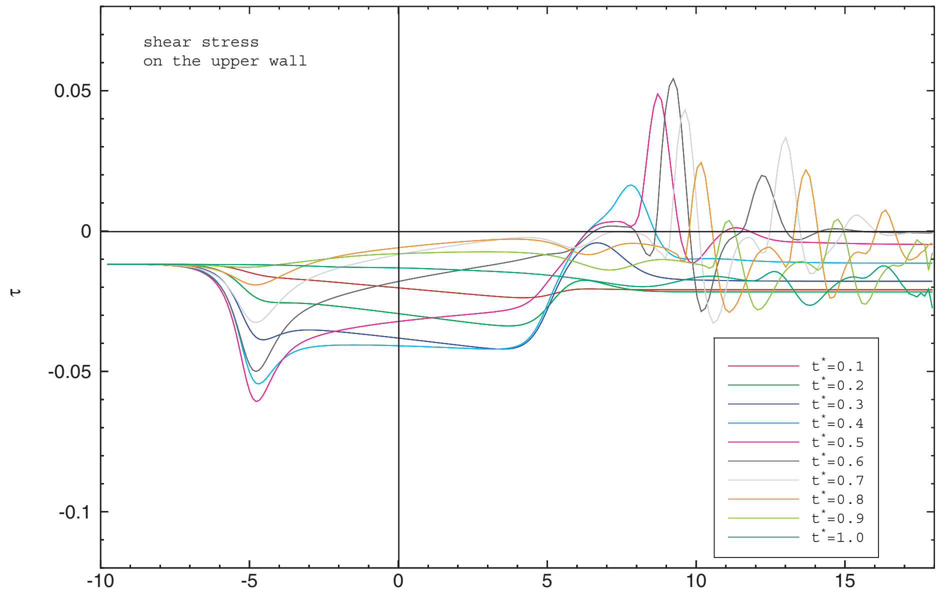
$$\max_{i^0,i^1,i^2}\left(\frac{\Delta t \mathbf{S}^j \cdot \mathbf{u} - \Delta V_{i^j}}{J}\right) \leqslant \mathrm{CFL} \tag{28}$$

4.3. Projection

The exact projection method uncouples the grid and deteriorates the solution where volumetric sources exist as outlined in Reference [9]. Additionally, the local decoupling renders the implementation of efficient linear algebra techniques cumbersome as presented in Reference [12]. In order to overcome these problems, approximate projection proposed by Almgren [12, 22] using a standard stencil for discretizing the Laplacian operator is used. Rider [17] showed that the dimension of null space of the standard discrete Laplacian operator is only one instead of four for the discrete operator of the exact projection in two-dimensional space.



(a)



(b)

Plate 1. Shear stress on the lower and upper walls at various instants of time, $t^* = 0.1$–1.0: (a) stress on the lower wall; and (b) stress on the upper wall.

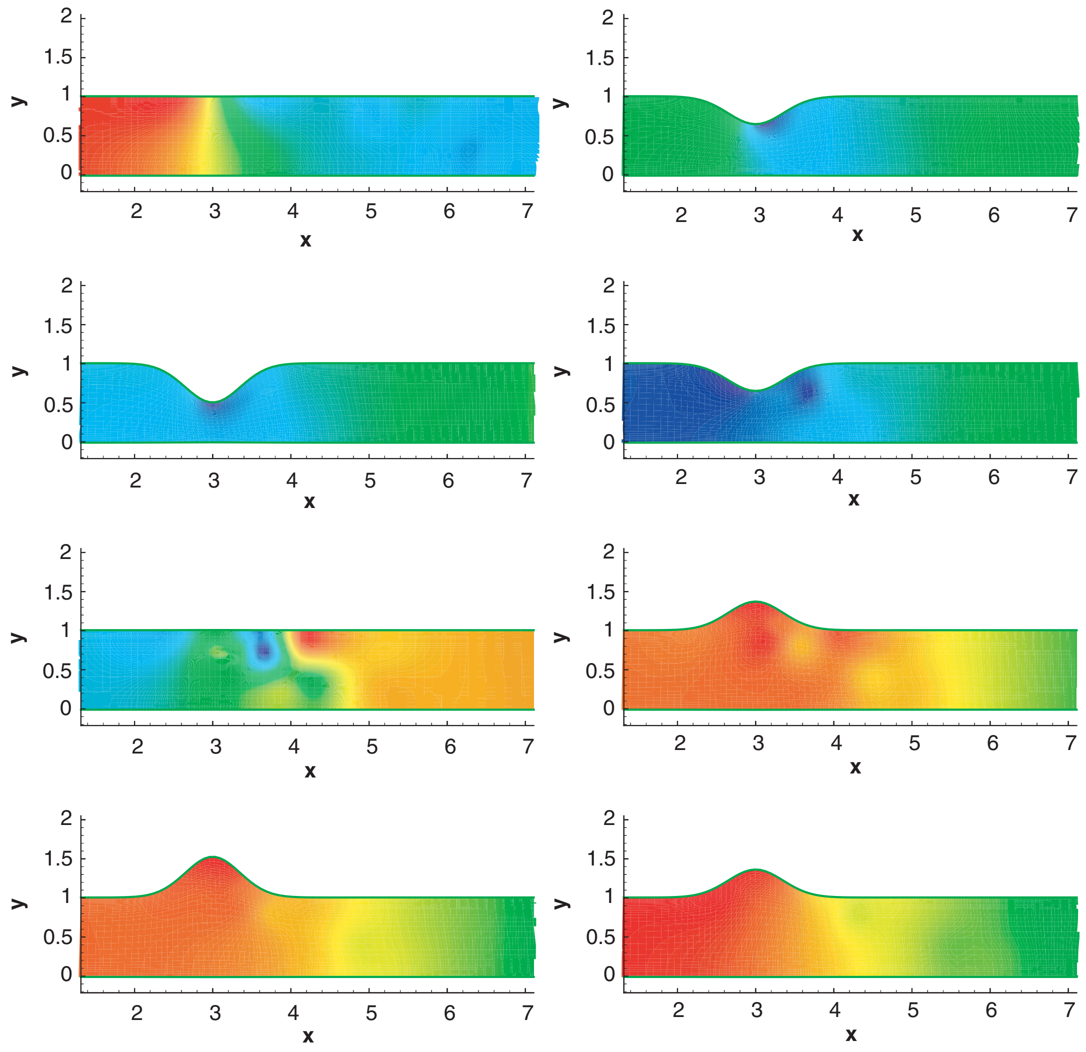


Plate 2. Pressure contour plots at different instants of time for the flow inside deforming tube: (a) flat wall moving inward; (b) partly pinched wall moving inward; (c) fully pinched wall moving outward; (d) partly pinched wall moving outward; (e) flat wall moving outward; (f) partly bulged wall moving outward; (g) fully bulged wall moving inward; and (h) partly bulged wall moving inward.

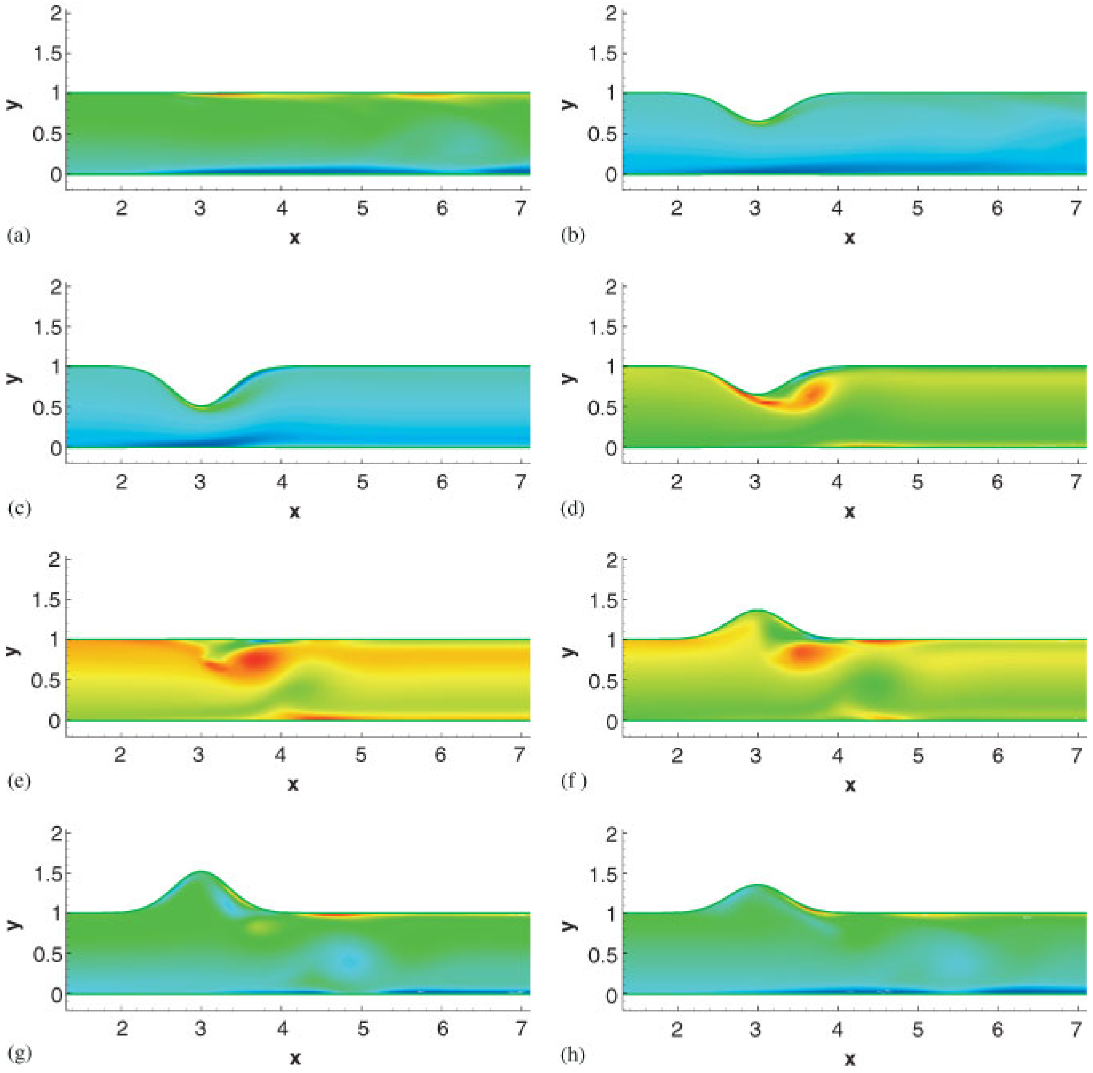


Plate 3. Vorticity contour plots at different instants of time for the flow inside deforming tube: (a) flat wall moving inward; (b) partly pinched wall moving inward; (c) fully pinched wall moving outward; (d) partly pinched wall moving outward; (e) flat wall moving outward; (f) partly bulged wall moving outward; (g) fully bulged wall moving inward; and (h) partly bulged wall moving inward.

Hence, the grid is coupled this way and the discrete divergence is a function of truncation error. Denoting the approximate projection as $\tilde{\mathbf{P}}$, it should be pointed out that the approximate projection is not idempotent, i.e. $\tilde{\mathbf{P}}^2 \neq \tilde{\mathbf{P}}$. The Poisson equation associated with the projection is given by

$$\mathbf{L}\psi = \mathbf{D}\left(\frac{\mathbf{u}^*}{\Delta t}\right) \tag{29}$$

where $\mathbf{L}$ is the discrete Laplacian operator and $\mathbf{D}$ denotes the discrete divergence operator. Here the intermediate velocity $\mathbf{u}^*$ is projected directly since the pressure form of projection is more robust according to Rider's analysis [17]. Appropriate boundary conditions should by applied. Gresho [19] gave excellent comments regarding this issue. Here the normal motion equation at the moving wall boundary is used as the boundary condition. Once the linear system of equations is solved, the velocity and the gradient of pressure term can be determined as follows:

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \mathbf{G}(\psi) \tag{30}$$

$$\frac{1}{J}\frac{\partial}{\partial \xi^j}(\mathbf{S}^j p)^{n+1/2} = \mathbf{G}(\psi) \tag{31}$$

where $\mathbf{G}$ is the discrete gradient operator. Both the velocity and the pressure field are updated.

5. PARALLEL STRATEGY

In order to overcome the computational overhead of unsteady three-dimensional numerical flow simulations, the algorithm presented above is parallelized on multi-block structured grids. Generally, the parallelization is accomplished by partitioning the computational grid and distributing the grid partitions between the processors. Special consideration must be paid to the load balancing and the parallel efficiency in terms of the ratio of time cost between the computation and communication.

In this work, the partition is achieved by partitioning the multi-block grids to ensure good load balancing. Each block grid is partitioned into a number of parts and distributed between the processors. Therefore, each processor has a collection of grid partitions which belong to different block grids. Each block grid is partitioned in a scalable manner where the grid is divided as equally as possible and the interface between grid partitions is minimized so that the amount of data exchanged during the computation is minimized as well and the efficiency of the parallel algorithm can be enhanced. In this work, a boxwise partition is employed to partition the block grid where the component grid can be divided in all three co-ordinate directions and the number of the grid partitions is equal to the number of the employed processors.

In order to make the produced parallel code portable, the message passing standard, MPI [23], has been used to implement the communication in the parallel numerical algorithm so that any platform supporting MPI can be used to do the calculation. The master/slave model shown in Figure 4 has been used as the model for implementing the algorithm in parallel by exploiting the scalable parallel computer architecture of the SGI Origin 2000 HPC Platform, where one processor (master) controls the execution of the overall computation including I/O

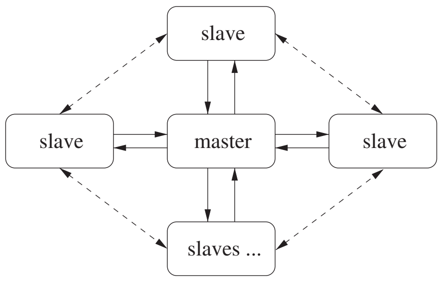


Figure 4. Structure of master/slave program model.

and other processors (slaves) execute the flow computation on their partitioned computational domain simultaneously. Basically, the slaves run in a loop awaiting the commands sent by the master. Upon completion of the job, the slaves signal the master so that subsequent actions can be initiated by the master. The slave is also allowed to communicate with other slaves directly under the supervision of the master so that unnecessary message passing is eliminated. The synchronization of processes is fulfilled by using blocked message sending and receiving of MPI.

6. NUMERICAL RESULTS

A parallel flow solver based on the algorithm presented above has been developed. Several test cases are solved to show its applicability. Although the cases presented here are all two-dimensional problems, the flow solver is three dimensional, where the symmetric boundary condition is applied on both boundaries of the third co-ordinate. The freestream on the moving domain is calculated by using this method to verify whether the GCL is satisfied in this method. The metrics formula presented in Section 3 is applied here. A benchmark problem, the flow inside a channel with moving indentation, which had be studied experimentally and numerically by other researchers, is solved here to examine the performance of this projection method. The flow inside a tube, which has an oscillating wall, is also simulated to demonstrate the performance of this method for solving cases with large Strouhal number.

6.1. Freestream capturing

The freestream on a moving domain is simulated here by using the method described in this paper. The purpose is to examine whether the GCL is satisfied exactly. If so, the freestream flow remain unchanged after the domain changes. Otherwise, the pseudosource exists and the flow field is deteriorated. The initial and final physical domains are shown in Figure 5(a) and 5(b), which are a rectangle and rhombus, respectively. The uniform grids used in this case are also shown in these figures. The inflow boundary condition is specified on all the boundaries of the domain, where the velocity of the freestream is specified as $\mathbf{u}_\infty = (1, 0, 0)$, where only u component has value and v and w are zero. Therefore, the value of v and w of the velocity field on the rhombus domain can be easily identified as error. Since the flow field should have no change, there is no limit on the time step. Several test cases are calculated,

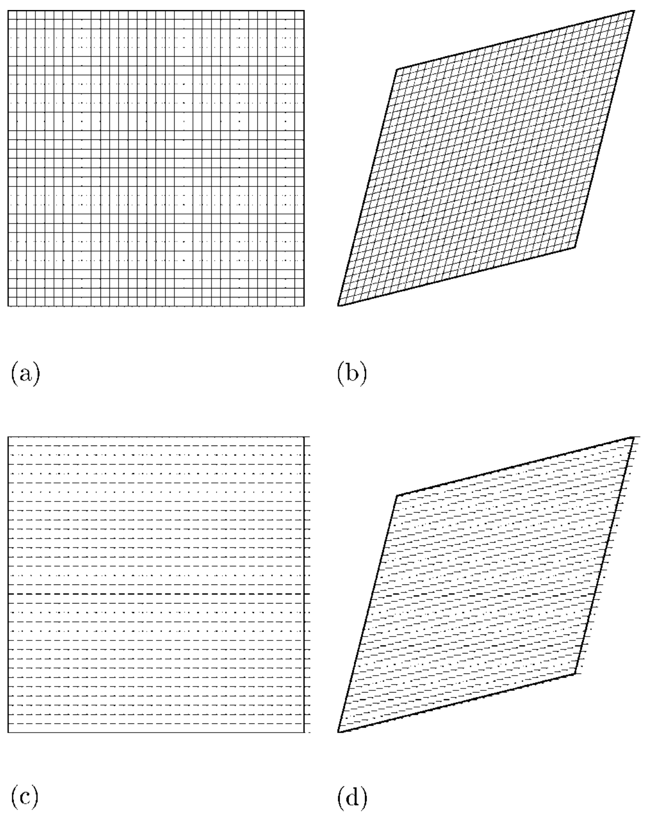


Figure 5. Domains and vector plots for freestream capturing: (a) initial domains; (b) final domain; (c) initial flow vector; and (d) final flow vector.

where the domain completes the change in either one time step or a series of time steps. Flows at different Reynolds number are also simulated. For all calculations, similar results are obtained, which are displayed in Figures 5(c) and (d) showing the initial and final velocity field. The velocity vectors show no change except the origin positions are changed due to the change of the grid points. The errors for all flow variables, u, v, w and p are very small (approximately $1.0e-8$), which can be considered as the truncation error of floating-point numbers expressed in the computer. Therefore, it can be concluded that the freestream flow on the moving domain can be accurately captured and there is no pseudosource, which would occur once the GCL is violated in the numerical procedure.

6.2. Flow inside a channel with a moving indentation

The flow inside a channel with a moving indentation is simulated here to verify the projection method presented in this paper. This problem had been studied experimentally by Pedley [24] and numerically by Ralph [25] and Demirdžić [5]. The geometry of the flow domain is shown in Figure 6 and the indentation is defined by using an analytic function, which was presented

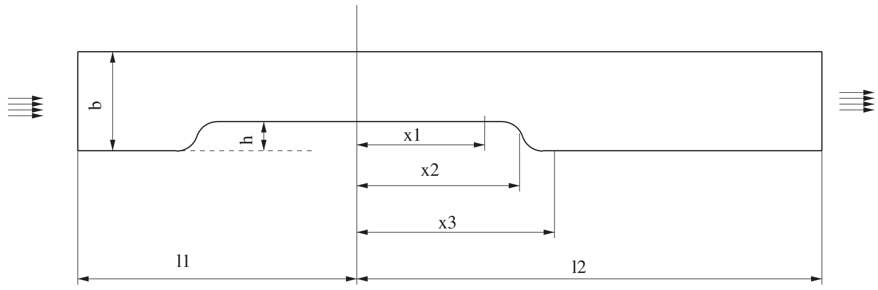


Figure 6. Geometry of the channel with moving indentation.

in Reference [5]:

$$y(x) = \begin{cases} \frac{1}{2}h(1 - \tanh[a(x - x_2)]) & \text{for } 0 \leqslant x < x_3 \\ y(-x) & \text{for } x < 0 \end{cases} \tag{32}$$

where $a = 4.14$, $x_1 = 4b$, $x_3 = 6.5b$ and $x_2 = 1/2(x_1 + x_3)$. Here b represents the channel height. The height of the indentation is the function of time:

$$h = 0.5h_{\max}[1 - \cos(2\pi t^*)]$$

where $t^* = (t - t_0)/T$. t_0 is the starting time and T is the period of the moving indentation. $h_{\max} = 0.38b$ is the maximum height of the indentation. In summary, all parameters specified here take the same values as in Reference [5].

The grids used in this calculation have $221 \times 41 \times 4$ grid points, which is similar to the fine grids used in Reference [5], and the grid points are clustered around the downstream connection part between the indentation and the channel. At each time step, the grid is regenerated by using an elliptic grid generator, which is based on the algorithm presented by Spekreijse and Boerstoel [26]. Figure 7 shows the grids when no indentation exists and the indentation reaches its maximum height. The uniform step length is used in the z direction, where the z co-ordinate is between -1.0 and 1.0.

The same Strouhal number and Reynolds number are used in this calculation, where $St = 0.037$ and $Re = 507$. The inflow velocity is specified by a parabolic distribution, where v and w is zero and the mean value of u is 1.0. The computed results at different instants of time are shown in Figures 8–17, where the vector, the pressure contour and the streamline plots are presented at $t^* = 0.1, 0.2, \ldots, 1.0$ respectively. The flow field at the plane where $z = 0.0$ is used in these plots. Only the section behind the indentation is presented since the vortices are produced in this part and the flow ahead the indentation is not much affected by the movement of the indentation. The vector plots are drawn by using the same reference vector so that the velocity magnitude at the different instants of time can be visually compared.

The flow rate is higher than the inlet rate when $t^* < 0.5$ since the indentation rises and more fluid is driven away. This can be observed in Figures 8–12. The first vortex behind the indentation is produced at $t^* = 0.3$ and then increases its range as the indentation keeps rising. At $t^* = 0.4$, the first upper wall vortex is also induced, which in turn induces the second lower wall vortex at $t^* = 0.5$. After $t^* > 0.5$, the flow rate is less than the inlet rate

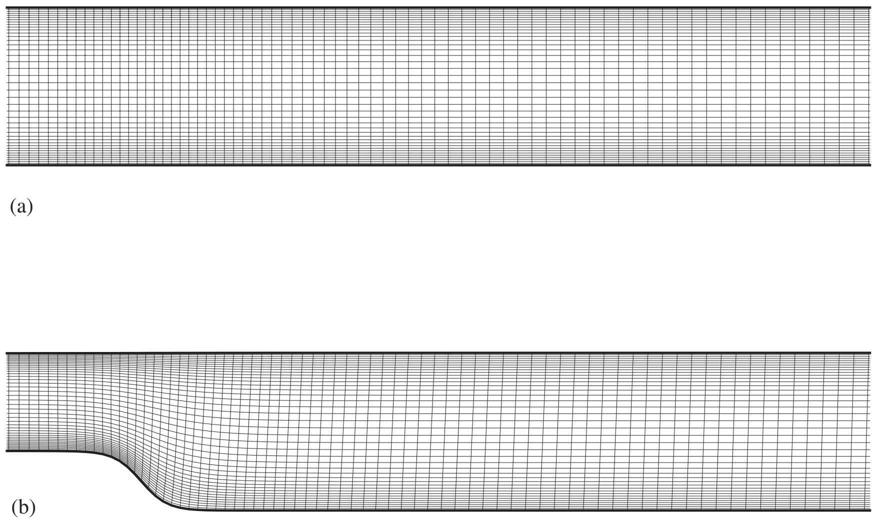


Figure 7. Section of the grids at: (a) $t^* = 0$; and (b) $t^* = 0.5$.

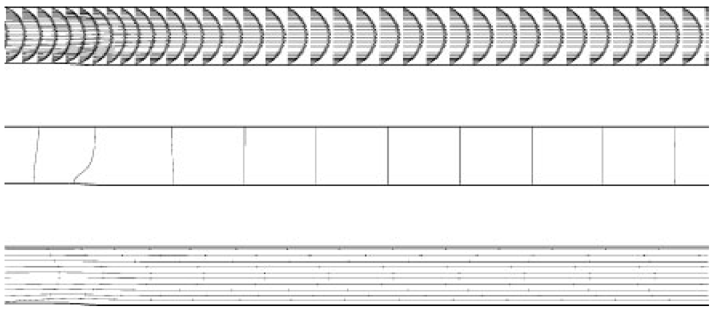


Figure 8. Flow with moving indentation at $t^* = 0.1$.

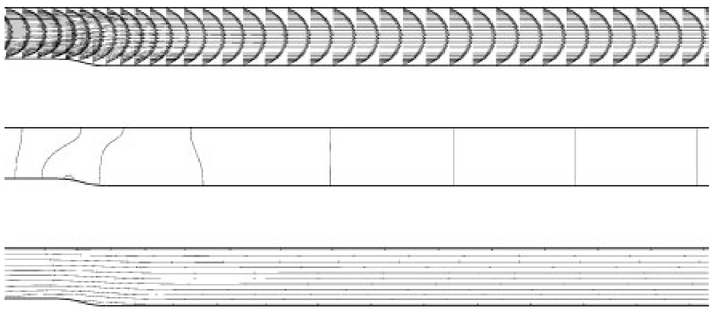


Figure 9. Flow with moving indentation at $t^* = 0.2$.

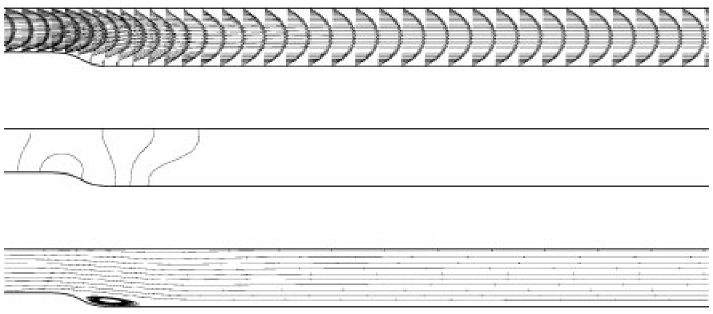


Figure 10. Flow with moving indentation at $t^* = 0.3$.

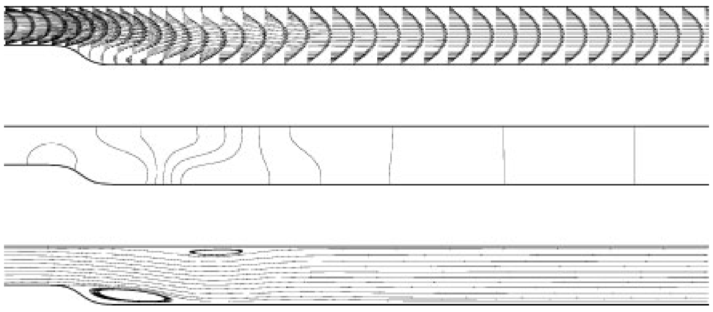


Figure 11. Flow with moving indentation at $t^* = 0.4$.

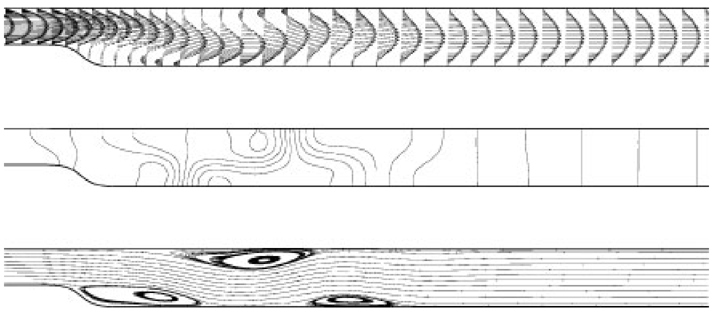


Figure 12. Flow with moving indentation at $t^* = 0.5$.

since the indentation is retracting and part of the oncoming fluid is used to fill the gap. More vortices are developed, where the second upper wall vortex appears and the first upper vortex is stretched as shown in Figure 13. The two lower wall vortices are also enhanced. A few more vortices are induced afterwords. As the lower wall further retracts, these vortices are shifted downstream and become weaker as shown in Figures 14–16. At $t^* = 1.0$, only slightly wavy streamlines remain since the flow channel is now flat. After one cycle, the pressure distribution was also recovered from the very complex structure at $t^* = 0.8$ to the nearly uniform distribution as shown in Figure 17.

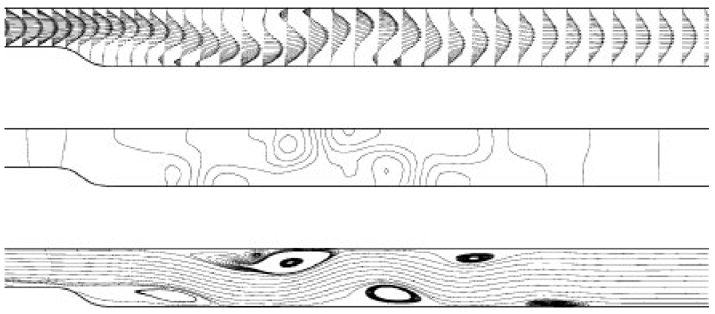


Figure 13. Flow with moving indentation at $t^* = 0.6$.

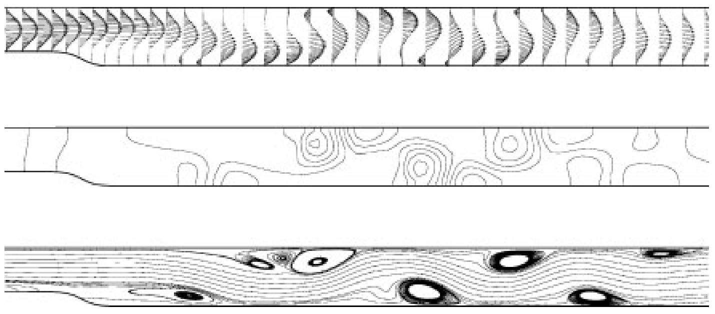


Figure 14. Flow with moving indentation at $t^* = 0.7$.

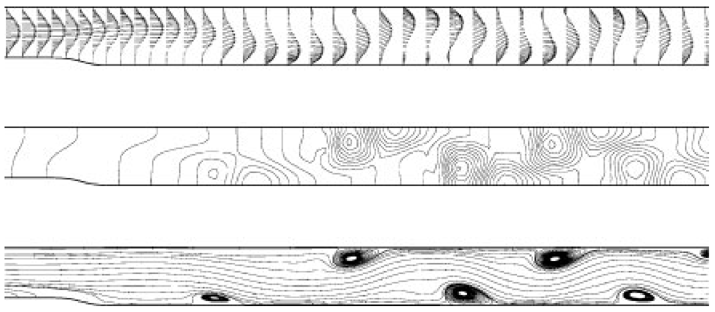


Figure 15. Flow with moving indentation at $t^* = 0.8$.

Demirdžić [5] also presented the vector, pressure contour and streamline plots at the same instants of times except $t^* = 0.1$. After comparison, we concluded that the results we obtained are very similar with Demirdžić's results. The shear stress on the lower and upper wall at these same instants of time are also presented in Plate 1. These figures indicate the strength of the eddies and the position of the separation and reattachment. After comparison of these shear stress distribution with Demirdžić's results, it can be concluded that similar solutions are obtained in this calculation by using the proposed projection method.

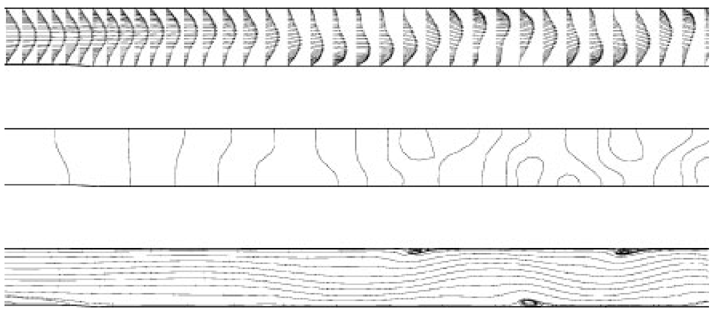


Figure 16. Flow with moving indentation at $t^* = 0.9$.

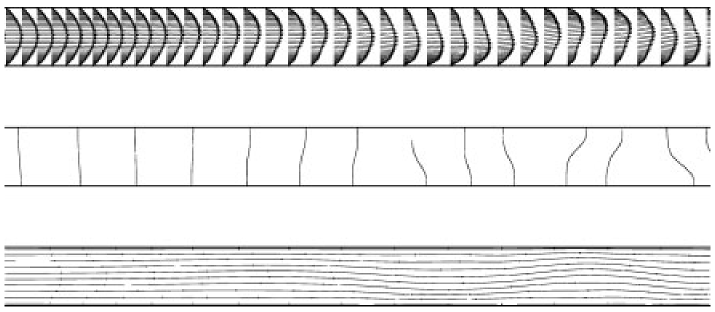


Figure 17. Flow with moving indentation at $t^* = 1.0$.

The performance of the three-dimensional parallel code was studied on the Origin 2000 mainframe computer using this case. The speed-up and the efficiency are shown in Figure 18. A reasonable speed-up and efficiency are achieved. When using 8 processors, the speed-up can reach 5.45. However, the efficiency is not very high, only 0.682 for 8 processors. The main reason is that the elliptic grid generation code is a sequential code, which is executed only on the master process and all slave processes are idle in this duration. It can be expected that the parallel performance of the algorithm can be improved greatly once the grid generation code is also parallelized.

6.3. Flow inside deforming tube

The Strouhal number used in the Demirdžić case is small, only 0.037, which means the indentation moves relatively slow. In this case, the flow inside a deforming tube, which has a rapidly oscillating wall, is simulated here to demonstrate the applicability of the proposed projection method for large Strouhal number.

Figure 19 shows the configuration of this flow. The lower wall of the tube is flat and fixed and its upper wall oscillates according to the function

$$y(x,t) = y_0 \left[1 - a_m \sin\left(\frac{\pi}{2} t\right) \exp(-4(x - X_c)^2)\right] \tag{33}$$

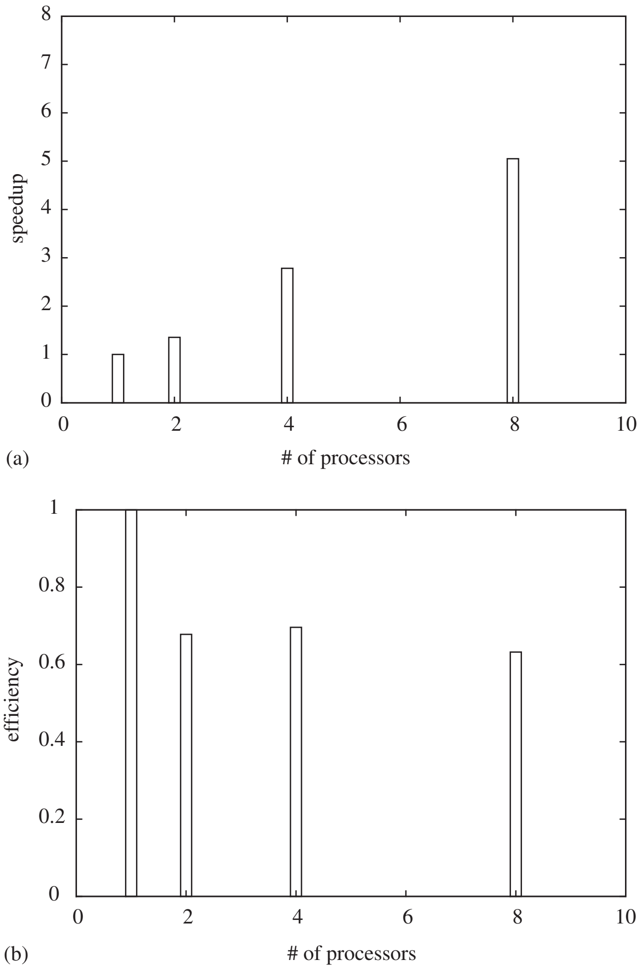


Figure 18. Speed-up and efficiency of the developed parallel code: (a) speed-up; and (b) efficiency.

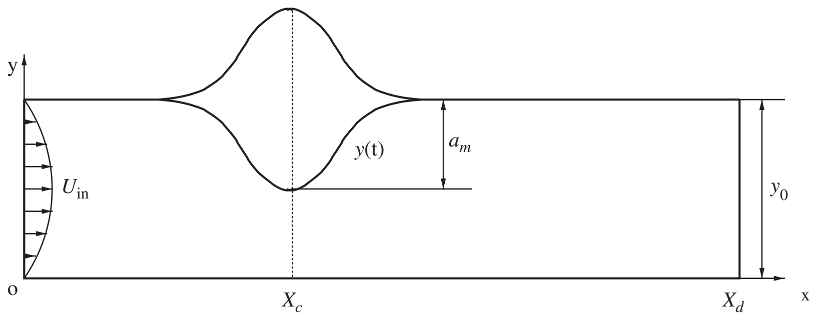


Figure 19. Configuration of flow inside moving tube.

where y_0 is the height of tube and a_m is the magnitude of the oscillation. X_c is the location of extrema for the Gaussian movement. The tube wall is initially flat and then it moves inward to fully pinched position at $t=1$ and back out through the flat position at $t=2$ to a fully bulged position at $t=3$ and flat again at $t=4$. Maximum wall speeds occur at the flat position. The parabolic distribution of velocity is specified at the inlet, where its mean velocity is 1.0. In this case, the following parameters are used:

$$y_0 = 1, \quad a_m = 0.5, \quad X_c = 3, \quad X_d = 12$$

Therefore, the Strouhal number for this case is 0.25. The Reynolds number is set to 200, which is based on the mean inflow velocity and the tube height. The tube upper wall is assumed to be a flexible so that the tangent velocity is set to zero always.

The grid points are clustered along the tube wall and the hump at $x = X_c$. The elliptic grid generation code is used to generate the grid at each time step. The grid dimensions are $121 \times 31 \times 4$, where the uniform step is used on the z direction. The symmetrical boundary condition is applied on both z boundaries. The sections of the grids at three different instants of time are displayed in Figure 20, where the tube wall is flat, fully pinched and fully bulged.

The flow fields computed at various instants of time are shown in Figure 21 and Plates 2 and 3, which consists of a period of the tube wall oscillation. Figure 21 show the velocity vector plots, the pressure contour plots and vorticity contour plots, respectively, at the different instants of time when the tube wall is at flat position with inward movement as shown in (a), partly pinched position with inward movement in (b), fully pinched position with outward movement in (c), partly pinched position with outward movement in (d), flat position with outward movement in (e), partly bulged position with outward movement in (f), fully bulged position with inward movement in (g) and partly bulged position with inward movement in (h).

When the upper wall protrudes inward from its flat position, the velocity behind the hump is larger than the inflow velocity since the volume of the tube is being reduced and more fluid is driven out by the movement of the tube wall. As the slope of the hump increases, the flow behind the hump begins to separate and the vortex is formed in the wake of the hump as shown in Figure 21(b). The vortex is enhanced until the upper wall is fully pinched as presented in Figure 21(c). The vortex is pushed away from the upper wall in this process because of the fact that the upper wall moves inwards. In Figure 21(c), it also can be found that there is a vortex near the lower wall behind the opposite location of the hump, which is induced by the upper wall vortex. Plate 3(c) clearly shows the lower wall vortex.

After the fully pinched position is reached, the upper wall begins to move outwards until it reaches the fully bulged position. An interesting phenomenon observed is that the upper wall vortex is kept at its position instead of being flushed away. Figures 21(d)–(f) and Plate 3(d)–(f) illustrate this phenomenon. Plate 2(c)–2(e) shows that the pressure gradient is reversed as long as the upper wall begins to move outwards. The outflow velocity is reduced and part of the incoming fluid is used to fill up the increasing tube volume. The lower wall vortex is enhanced under the reversed pressure gradient. It also raises by the sucking of the upper wall. The second upper wall vortex is developed behind the hump under the reversed pressure gradient, which further reduces the outflow rate to match the requirement of flow rate to fill the expanding tube volume.

Figure 21(g) and Plate 3(g) show the flow field where the full protrusion is reached. At this moment, the upper wall begins to move inwards and the tube volume starts to shrink.

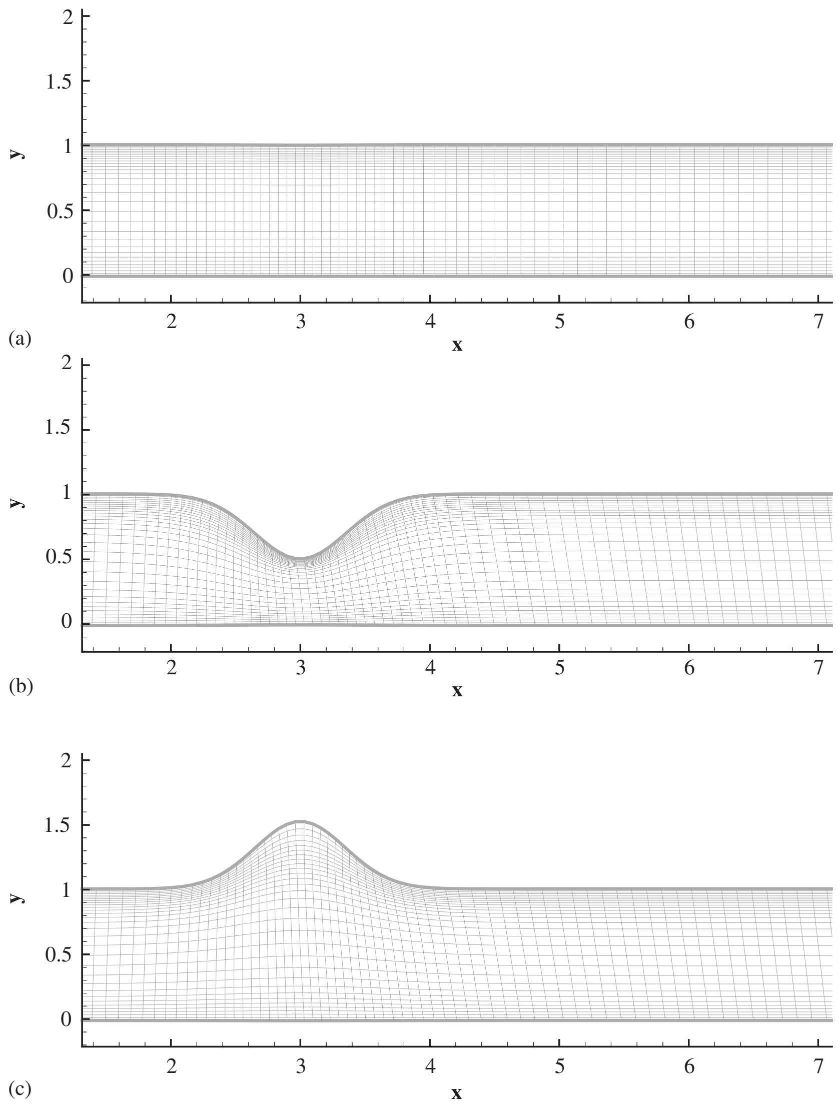


Figure 20. Grid sections at three different instants of time: (a) flat wall; (b) fully pinched wall; and (c) fully bulged wall.

Accumulated fluid during the bulging process is driven out and the outflow velocity is larger than the inflow velocity. The positive pressure gradient is recovered as shown in Plate 2(g). As a consequence, the vortices are weakened and are flushed out of tube as shown in Figure 21(h) and 21(a). It can be seen in these figures that there is a vortex inside the protrusion just behind the first curve and it disappears quickly as the curvature of the protrusion is reducing and the pressure gradient is positive.

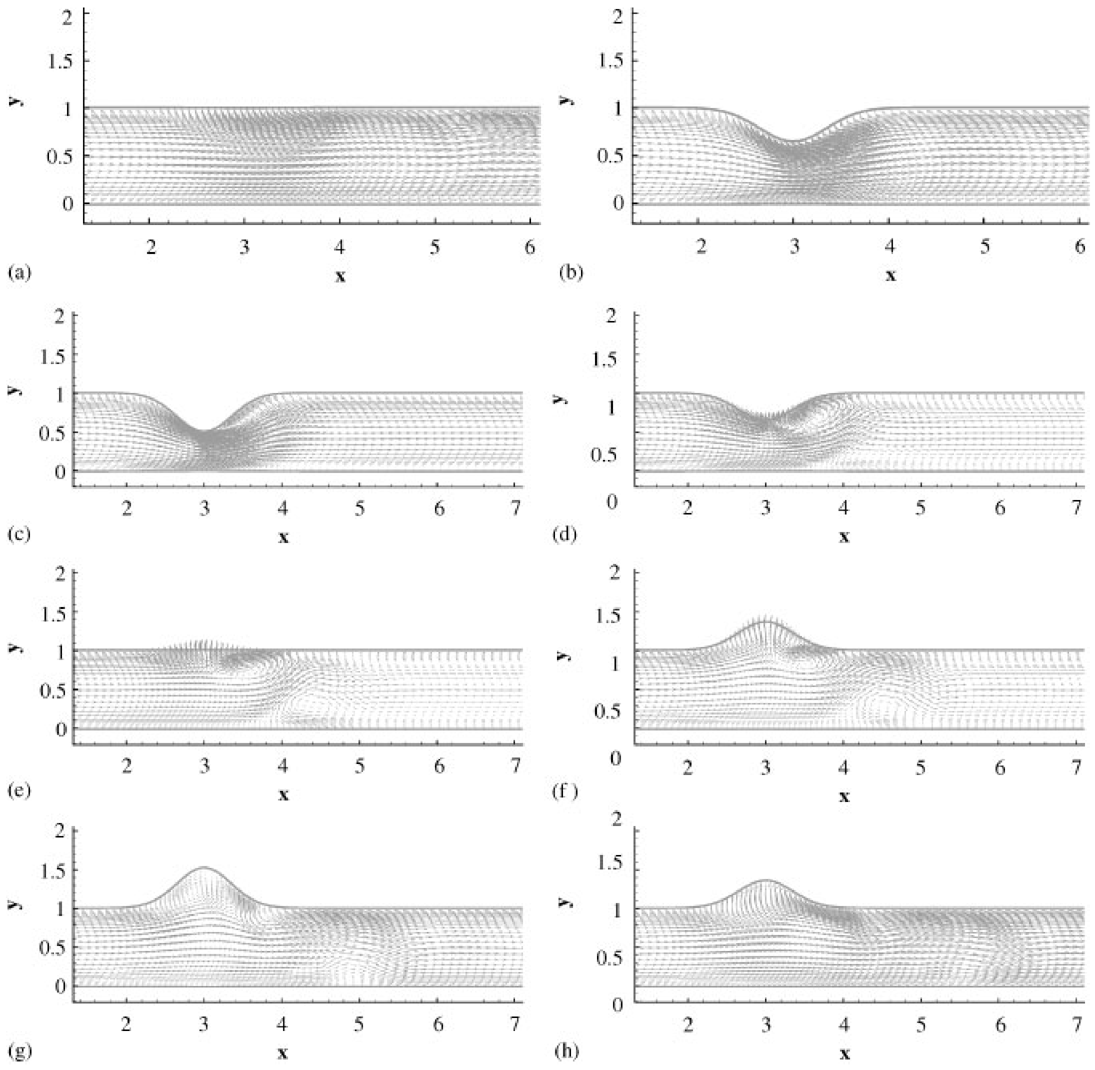


Figure 21. Vector plots at different instants of time for the flow inside deforming tube: (a) flat wall moving inward; (b) partly pinched wall moving inward; (c) fully pinched wall moving outward; (d) partly pinched wall moving outward; (e) flat wall moving outward; (f) partly bulged wall moving outward; (g) fully bulged wall moving inward; and (h) partly bulged wall moving inward.

In summary, the flow field computed by using the proposed projection method is reasonable and interesting results are obtained for the flow inside the fast deforming tube.

7. CONCLUSIONS

In this paper, a projection method is presented to solve the unsteady incompressible Navier–Stokes equations on the ALE co-ordinate system so that flow problems with moving boundaries can be solved accurately. This projection method is based on the Godunov-projection (BCG) method proposed by Bell [15]. Substantial changes are made to solve the Navier–Stokes equations on the ALE co-ordinates and to let the geometrical conservation law be satisfied. The freestream capturing metrics for general three-dimensional structured grids are reviewed and a new formula is proposed to exactly compute the volume of a hexahedral cell. It is also verified that the new formula does guarantee the GCL in this projection method. The numerical algorithm is also parallelized based on the grid partitioning. Several numerical test cases are used to demonstrate the applicability of this projection method for solving the moving boundary problems, where large and rapid domain deformation exists. It can be concluded that this method is capable of solving the ALE formulation of unsteady incompressible Navier–Stokes equations. The parallel performance of this algorithm is also presented and reasonable speedup and efficiency are achieved.

REFERENCES

1. Thomas PD, Lombard CK. Geometric conservation law and its application to flow computations on moving grids. *AIAA Journal* 1979; **17**(10):1030–1037.
2. Zhang H, Reggio M, Trépanier JY, Camarero R. Discrete form of the gcl for moving meshes and its implementation in cfd scheme. *Computers and Fluids* 1993; **22**:9–23.
3. Demiržić I, Perić M. Space conservation law in finite volume calculation of fluid flow. *International Journal for Numerical Methods in Fluids* 1988; **8**:1037–1050.
4. Shyy W, Pal S, Udaykumar HS, Choi D. Structured moving grid and geometric conservation laws for fluid flow computation. *Numerical Heat Transfer Part A: Applications* 1998; **34**:369–397.
5. Demiržić I, Perić M. Finite volume method for prediction of fluid flow in arbitraily shaped domains with moving boundaries. *International Journal for Numerical Methods in Fluids* 1990; **10**:771–790.
6. Obayashi S. Freestream capturing for moving coordinates in three dimensions. *AIAA Journal* 1991; **30**(4): 1125–1128.
7. Vinokur M. An analysis of finite-difference and finite volume formulations of conservation laws. *Journal of Computational Physics* 1989; **81**(1):1–52.
8. Bell JB, Colella P, Glaz HM. A second-order projection method for viscous incompressible flow. In *Proceedings of 8th AIAA Computational Fluid Dynamics Conference*, Honolulu, Hawaii, 1987; pp. 789–794.
9. Lai MF, Bell JB, Colella P. A projection method for combustion in the zero mach number limit. In *Proceedings of 11th AIAA Computational Fluid Dynamics Conference*, Orlando, FL, USA, vol. 7, 1993, pp. 776–783.
10. Bell JB, Solomon JM, Szymczak WG. A projection method for viscous incompressible flow on quadrilateral grids. *AIAA Journal* 1994; **32**:1961–1969.
11. Brown DL, Minion ML. Performance of under-resolved two-dimensional incompressible flow simulations—1. *Technical Report* Los Alamos National Laboratory (published in internet) 1995.
12. Almgren AS, Bell JB, Szymczak WG. A numerical method for the incompressible Navier–Stokes equations based on an approximate projection. *SIAM Journal on Scientific Computing* 1996; **17**(2):358–369.
13. Pan H, Damodaran M. Parallel computation of viscous incompressible flows using godunov-projection method on overlapping grids. *International Journal for Numerical Methods in Fluids* 2002; **39**(5):441–463.
14. Trebotich DP, Colella P. A projection method for incompressible viscous flow on moving quadrilateral grids. *Journal of Computational Physics* 2001; **166**:191–217.
15. Bell JB, Colella P. A second-order projection method for the incompressible Navier–Stokes equations. *Journal of Computational Physics* 1989; **85**(2):258–283.
16. Chorin AJ. On the convergence of discrete approximations to the Navier–Stokes equations. *Mathematics of Computation* 1969; **23**:341–353.
17. Rider WJ. The robust formulation of approximate projection methods for incompressible flows. *Technical Report* LA-UR-94-3015, Los Alamos National Laboratory, 1994.
18. Hirt CW, Harlow FW. A general corrective procedure for the numerical solution of initial-value problems. *Journal of Computational Physics* 1967; **2**:114–119.
19. Gresho PM, Sani RL. On pressure boundary conditions for the incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluid* 1987; **7**:1111–1145.
20. Bell JB, Colella P, Howell LH. An efficient second-order projection method for viscous incompressible flow. In *Proceedings of 10th AIAA Computational Fluid Dynamics Conference*, Honolulu, Hawaii, 1991, pp. 91–1560.
21. Colella P. Multidimensional upwind methods for hyperbolic laws. *Journal of Computational Physics* 1990; **87**:171–200.

22. Almgren AS, Bell JB, Colella P, Howell LH. An adaptive projection method for the incompressible euler equations. In *Proceedings of 11th AIAA Computational Fluid Dynamics Conference*, Orlando, Florida, 1993, pp. 530.
23. MPI Forum. Mpi:a message-passing interface standard. *International Journal of Supercomputer Application* 1994; **8**:165–416.
24. Pedley TJ, Stephanoff KD. Flow along a channel with a time-dependent indentation in one wall: the generation of vorticity waves. *Journal of Fluid Mechanics* 1985; **160**:337–367.
25. Ralph ME, Pedley TE. Flow in a channel with a moving indentation. *Journal of Fluid Mechanics* 1988; **190**:87–112.
26. Spekreijse SP, Boerstoel JW. Multiblock grid generation part 1: elliptic grid generation methods for structured grids. *Lecture Series 1996-06 von Karman Institute for Fluid Dynamics* 1996.