Journal of Applied Fluid Mechanics, Vol. 4, No. 2, Special Issue, pp. 27-36, 2011.
Available online at www.jafmonline.net, ISSN 1735-3572, EISSN 1735-3645.
DOI: 10.36884/jafm.4.03.11931

Immersed Boundary Method for the Solution of 2D Inviscid Compressible Flow Using Finite Volume Approach on Moving Cartesian Grid

S.M.H. Karimian and M. Ardakani†

Center of Excellence in Computational Aerospace Engineering
Aerospace Engineering Department, Amirkabir University of Technology, Tehran, Iran

†*Corresponding author Email: m.ardakani@aut.ac.ir*

(Received April 25, 2010; accepted March 13, 2011)

ABSTRACT

In this study, two-dimensional inviscid compressible flow is solved around a moving solid body using Immersed Boundary Method (IBM) on a Cartesian grid. Translational motion is handled with a Cartesian grid generated around the body which moves with body on a background grid. In IBM, boundaries are immersed within the grid points. In this paper solution domain is discretized using finite volume approach. To implement boundary conditions on immersed boundaries, a set of Ghost finite volumes are defined along the wall boundaries. Boundary conditions are used to assign flow variables on these Ghost finite volumes. Governing equations are solved using dual time step method of Jameson. Finally, numerical results obtained from the present study are compared with the other numerical results to evaluate the correct performance of the present algorithm and its accuracy.

Keywords: Immersed boundary method; Cartesian grid; Moving mesh; Unsteady Euler; Finite volume; Jameson algorithm.

NOMENCLATURE

P	pressure
E	energy
u	velocity in x direction
v	velocity in y direction
U	x component of contravariant velocities
V	y component of contravariant velocities
x_t	x components of boundary velocity
y_t	y components of boundary velocity
$R_i(w)$	Convective flux force
$D_i(w)$	Numerical dissipation
t	real time
τ	pseudo time
ρ	Density
Ω	Cell's area
CFL	Currant number

1. INTRODUCTION

Immersed Boundary method (IBM) is a new approach introduced in the last decade for the simulation of fluid flow around bodies with complex stationary/moving boundaries. In contrast to conventional methods, IBM uses non-conformal Cartesian grid where surface of body do not necessarily pass through the grid points of the boundary. Therefore boundary conditions cannot be applied throughout the ordinary approaches. The technique by which boundary conditions are applied in IBM is the key point in these methods. Looking back to the original forms of IBM, it can be seen that boundary conditions have been applied by adding a forcing function to the governing equations. IBMs are categorized into two main groups. In the first group, known as "continues forcing function", the function responsible for implementing boundary conditions is added to the governing equations before discretization. Works of Peskin (1972), Lai *et al*. (2000), Goldstein *et al*. (1993), Beyer (1992), Fauci *et al*. (1994) and Unverdi *et al*. (1992) fall in this category.

In the second group, named "discrete forcing approach", forcing function is added to the discretized form of the governing equations. The significant advantage of this method is that it would be independent of the discretization method. Approaches in this group are subdivided into two categories "direct approaches" and "indirect approaches". In the indirect approaches, forcing function is well-selected to apply boundary conditions into discretized governing equations. Mohd-Yusof (1997) and Verzicco (2000) are good examples in this group. In their method, discretized form of the Navier-stokes equations is first solved and then corrections for boundary conditions are established. In this method, forcing function is determined in each time step using the latest velocity

field calculated in the solution domain. This method was used for modeling bodies with complex boundaries (Balaras 2004) and turbulence flow (Verzicco 2002). In some situation, implementation of forcing functions results in diffusing boundary effects over neighbor grid points. Following this, researchers introduced "Direct approaches" in which boundary conditions are implemented directly on the discrete boundary in the computational domain.

Direct approaches are subdivided into two groups, "cut-cell methods" and "ghost-node methods". In the cut-cell method Cartesian finite volumes on the boundary are tailored to conform to the boundary, as shown in Fig. 1a and Fig.1b. In this case conservation of conserved quantities can be satisfied on the boundary finite volumes (DeZeeuw *et al.* 1993; Quirk 1994; Yang *et al.* 1994; Udaykumar *et al.* 1996; Ye *et al.* 1999). In this case we do not deal with Cartesian grids on the boundary any more. In some cases, this tailoring or reshaping may result in very small grid cells with an adversely impact on the numerical stability. This can be overcome by a cell-merging strategy shown in Fig. 1c and Fig. 1d (Clarke *et al.* 1985).

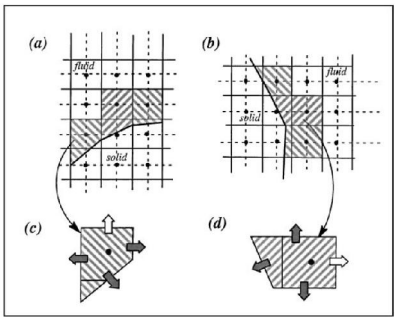

Fig. 1. Treatment procedure of boundary cells in cut-cell methods (Clarke *et al.* 1985)

Ghost-node methods are introduced to rebuild the solution at grid nodes in the vicinity of the immersed boundary using interpolation functions, to implement the boundary conditions. The main issue is that how the solution is rebuilt near the boundary. The choice of interpolation functions makes the difference between methods in this category.

One-dimensional interpolation is used by Fadlun *et al.* (2000) along the grid line intersecting solid boundary, but the choice of intersecting grid line seems to be arbitrary; see Fig. 2. Later on, Balaras (2004) presented an approach where the solution is reconstructed along a defined line normal to the body; similar approaches can be found in several existing studies (Gilmanov *et al.* 2003, 2005; Lai *et al.* 2000; Fadlun *et al.* 2000). Approaches using interpolation functions are simple and straight forward. However, in cases where boundary passes from a distance very close to the grid points, these approaches encounter some kind of instability.

Grid-point stencils which are normally used in IBM, are shown in Fig. 2. Depending on the location of boundary, some nodes in the solid section of domain would become a part of solution. These nodes are called 'ghost nodes'. Ghost nodes or ghost cells in finite volume methods, are solid nodes which at least have one fluid node in their neighbors.

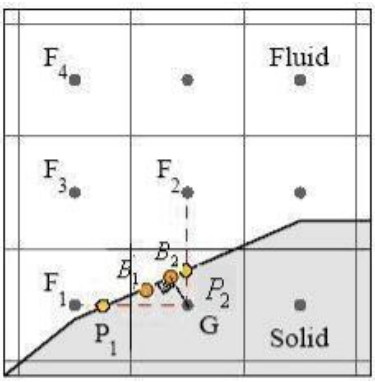

Fig. 2. Grid-point stencils for imposing boundary conditions (Lai *et al.* 2000)

Solid and fluid nodes are the nodes that are located within the solid and fluid regions of domain, respectively.

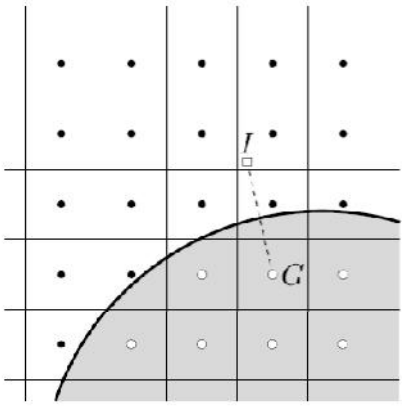

Fig. 3. Image point of a sample ghost node. G and I denote Ghost cell and Image point (Petter *et al.* 2008)

In IBM, variables at the ghost nodes are needed to close the governing equations at the fluid nodes. Boundary conditions along the solid boundary are implemented throughout the determination of these ghost-node variables. IBMs are mostly recognized by the method used to assign ghost-node variables. These variables are normally extrapolated from their values from the fluid part of the solution domain. The advantage of using ghost nodes is that the same discretized form of equations used inside the solution domain will be applied for the nodes in the vicinity of the solid boundary. In words, there is no need to reformulate the numerical algorithm for the nodes near to the boundary. There are numerous ways for assigning variables at ghost nodes, using interpolation schemes (Tseng *et al.* 2003). Although higher-order polynomials are more accurate, they are more sensitive to numerical instabilities. In this category, first order two-dimensional interpolation was used by Majumdar *et al.* (2001), and quadratic interpolation was used by Tseng *et al.* (2003, 2005). To resolve the instability problem, the concept of Image Point (denoted by I) was introduced and widely used by Mittal *et al.* (2003). As

shown in Fig. 3, an image point, located within the flow domain, is the mirror point of the ghost node with respect to the solid boundary. Determination of image points may raise difficulties. Some of these difficulties are shown in Fig. 4.

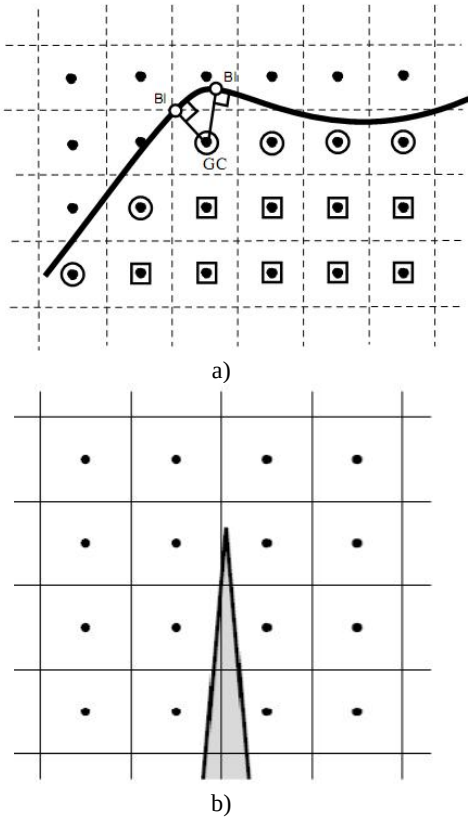

Fig. 4. Special cases in determination of ghost-node image points. a) Ghost node has no unique image point, b) Ghost node lies in fluid part of domain. GC and BI denote Ghost cell and boundary intercept [27]

In Fig. 4a, due to the position of ghost node with respect to the solid boundary a unique image point cannot be defined for it. On the other hand as shown in Fig. 4b, sometimes one cannot find a ghost node in solid region of domain. In this case, duplicate of the corresponding fluid node is considered as a fictitious ghost node. This requires separate memory locations for the fictitious ghost nodes. The corresponding image point is then determined.

Works of Mittal cover different Mach-number flows of viscid/inviscid conditions around stationary/moving bodies with complex geometries (Mittal *et al.* 2002, 2003, 2004, 2005, 2008). Simulations of moving boundaries are carried out using a qualified grid. This quality of grid must be preserved during body motion. This can be done using grid quality improvement procedure. Note that the motion of boundary causes continuous change of fluid, solid, and ghost nodes to each other. Therefore the algorithm should be able of handling these changes.

In the present paper, 2D inviscid compressible flow is simulated around moving bodies using IBM approach. Boundary conditions are implemented by direct method using ghost nodes. Solution domain is discretized into Cartesian finite volumes. To implement boundary conditions Ghost finite volumes (GFV) are established along the boundary. The main intention of this paper is to combine finite volume technique with IBM. As a result conservation laws will be satisfied within the domain. As will be discussed later flow variables at GFVs will be assigned based on the boundary conditions. For the simulation of flow around moving bodies there exist many approaches; see Mittal's works (Mittal *et al.* 2002, 2003, 2004, 2005, 2008). These include approaches in which the grid is regenerated after each time step of body motion, or approaches in which the grid is continuously adapted to the moving boundaries. Each of these methods has its own advantage and drawbacks. Methods in which the grid is generated repeatedly will be computationally costly. In addition to this flow variables should be interpolated from one grid to the other one. This will cause significant numerical error. Fully dynamic grids are also very complicated, and their coding will be cumbersome. Since Cartesian grid is used here, simulation of fluid flow around bodies with translational motion can be simply modeled. As mentioned before, Mittal *et al.* (2008) used finite difference to descretize governing equations. Boundary conditions are implemented in Mittal's work using ghost nodes for which corresponding image points in fluid domain have been defined, as mentioned before. Obviously with boundary motion ghost nodes and their related image points will be changed. Therefore, search would be needed for new ghost nodes and their related image points in each time step. To handle moving boundaries with translational motion in this paper, hybrid grid approach of Mirsajedi *et al.* (2006) is employed. Solution domain is divided into two zones. As shown in Fig. 5, first zone includes a Cartesian grid in which the moving body is located. This zone will move and the body is stationary with respect to it. Second zone is a background Cartesian grid in which the first zone can be moved. Any grid refinement can be performed in first zone to provide solution accuracy. Since the first zone has rectangular boundaries any translational motion can be simply modeled.

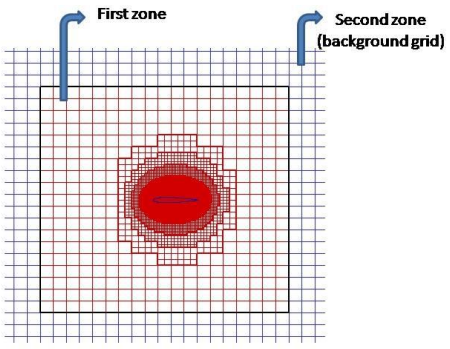

Fig. 5. Grid configuration; two zone approach

The advantage of this approach is that the number of node deletion/insertion process is minimized, and therefore very few data interpolation is required. On the other hand, since the moving body is stationary in the first zone, fluid, solid and ghost finite volumes remain unchanged during the solution procedure.

2. GRID GENERATION

In this study, moving body simulation is handled using two-grid zone approach of Mirsajedi *et al.* (2006). As shown in Fig. 5, immersed boundary of a moving body is located in the first zone which itself moves within the second zone, called back ground. Grids of both zones are Cartesian. As mentioned in the previous section this grid configuration allows simple handling of translational motion of a moving boundary. Although not employed in this paper, rotational motion of a moving body can be also modeled by this approach if one more zone is added to this grid configuration; see paper of Mirsajedi *et al.* (2006) for more details. As mentioned earlier, solid boundaries are defined as immersed boundaries in a Cartesian grid. The key task in IBM is to accurately impose boundary conditions on these immersed boundaries. For this purpose, high quality grids are required in the vicinity of immersed boundaries. This is carried out in this paper using a multi-layer refinement algorithm .There is other works such as the work of Mittal *et al.* (2007) in which the grid is refined around the body locally. However as shown in Fig. 6, these refinements spread out in x and y directions of the domain as banded refined zones, which is not desirable.

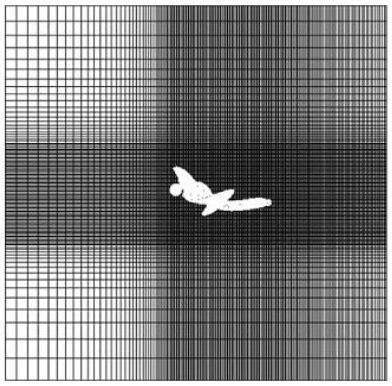

Fig. 6. Grid refinement around body in x and y directions (Ghias *et al.* 2007)

Grid generation procedure includes, 1) Cartesian grid generation in both zones as was shown in Fig. 5, 2) implementation of multi-layer refinement algorithm in the first zone to improve grid quality around the body. The following refinement procedure is only applied to the first zone. Refinement layers are shown in Fig. 7.

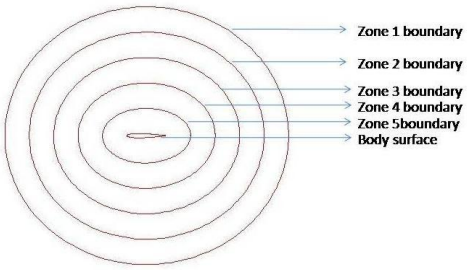

Fig. 7. Multi layers in refinement procedure

Refinement procedure is started from outer layer to the inner layer. One level of refinement is applied to Cartesian grids within all layers, as shown in Fig. 8b. In the next step one more level of refinement is applied to Cartesian grids within all layers except the most outer one, as shown in Fig. 8c. Having excluded the two most outer layers grid refinement is applied once again to the rest of Cartesian grids within the inner layers. Depending on the number of layers this procedure is continued up to the layer neighbor to the body, as shown in Fig. 8d.

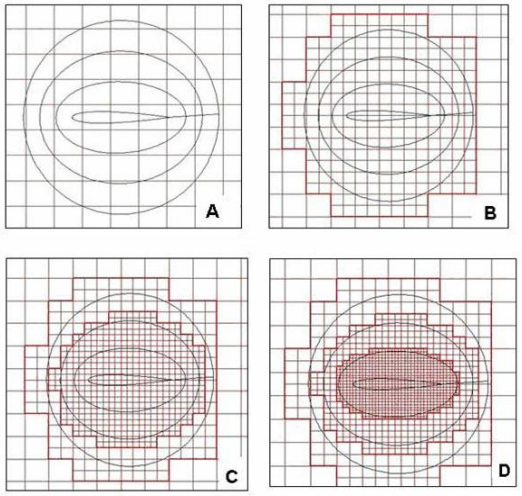

Fig. 8. grid refinement procedure in first zone around body surface

Since the solid body under consideration is fixed in the first zone, this grid zone will move with it. As shown in Fig. 9a, background grid lines become close to each other in front of the first zone, and become far from each other at the back of the first zone. To preserve grid quality, deformations are linearly distributed within the three or four rows of the background grid; this shown in Fig. 9b. In this algorithm, whenever the 1st grid line of background grid in the front of the first zone reaches to the old location of the 2nd grid line of background grid, these two grid lines will be merged with each other. In this case, the two grid lines of background grid in the rear of the first zone will be split into three lines; this is shown in Fig. 9b. In this procedure the total number of grid points in the domain will be constant.

As shown in Fig. 10, discretization of the solution domain produces three types of finite volumes when using IBM. A finite volume with its center located within the body is known as solid finite volume (SFV). These are the finite volumes on which flow equations are not solved. In contrast to this, if the center of finite volume is located in the fluid part of the solution domain the finite volume is named fluid finite volume (FFV). These are the finite volumes on which flow equations will be solved. In addition to these two types of finite volumes flow variables should be determined on a set of finite volumes called Ghost finite volume (GFV). GFVs are solid finite volumes which at least has one fluid finite volume in their neighbors. Flow variables on GFVs are needed for calculation of fluxes on the surface of FFVs. As will be discussed later, this

is where the boundary conditions will be applied on the solid boundaries. Note that since solid boundary is stationary with respect to first zone, solid, fluid, and ghost finite volumes remain unchanged during the body motion.

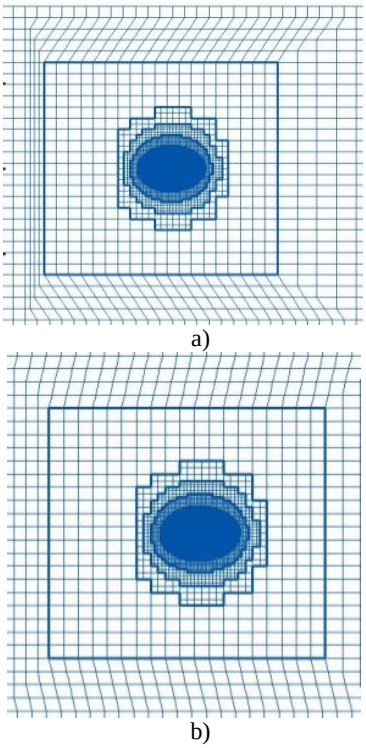

Fig. 9. First zone moving to left; A) Before line deletion/insertion, B) After line deletion/insertion

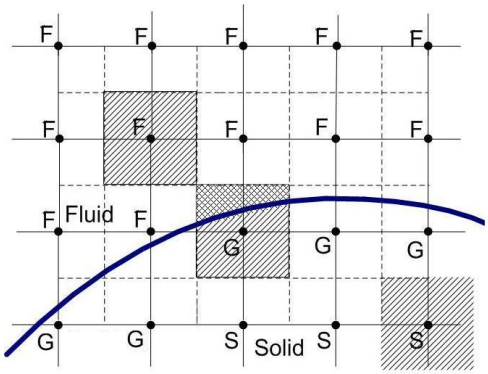

Fig. 10. Three types of finite volumes created by the immersed boundary; S: solid finite volume, F: Fluid finite volume, and G: Ghost finite volume

3. GOVERNING EQUATIONS

Integral form of the two dimensional unsteady Euler equations for compressible flow in the Cartesian coordinate are given by

$$\frac{d}{dt}\iint_{\Omega} W dA + \oint_{\partial\Omega}\left(\bar{F}dy - \bar{G}dx\right) = 0 \qquad (1)$$

where $W, \bar{F}$, and $\bar{G}$ are defined as,

$$W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix},$$

$$\bar{F} = \begin{pmatrix} \rho U \\ \rho u U + p \\ \rho v U \\ (\rho E + p)U + x_t p \end{pmatrix}, \quad \bar{G} = \begin{pmatrix} \rho V \\ \rho u V \\ \rho v V + p \\ (\rho E + p)V + y_t p \end{pmatrix} \qquad (2)$$

ρ, p, u, v and E are density, pressure, velocity components, and total energy, respectively. Contravariant velocities U and V are defined as

$$U = u - x_t \qquad V = v - y_t$$

Where x_t and y_t are velocity components of finite-volume boundary. In addition to these, equation of state for a perfect gas is used to complete the set of equations. Equation (1) is applied to each finite volume with area of Ω and boundary of ∂Ω. This results in the following equation

$$\frac{d}{dt}(w_i \Omega_i) + R_i(w) - D_i(w) = 0 \qquad (3)$$

where $R_i(w)$ is the convective flux over the surfaces of *i*th finite volume, and $D_i(w)$ is the numerical dissipation term which is introduced to prevent odd and even point oscillations, and oscillations in the vicinity of shock waves (Jameson *et al*. 1986).

Equation (3) is implicitly discretized in time; i.e. Jameson *et al*. (1996)

$$\frac{d}{dt}(w_i^{n+1}\Omega_i^{n+1}{}_i) + R_i(w_i^{n+1}) - D_i(w_i^{n+1}) = 0 \qquad (4)$$

Using second order accurate backward differencing (Jahangirian *et al*. 2004) Eq. (4) can be written as

$$\frac{3}{2\Delta t}(w_i^{n-1}\Omega_i^{n+1}) - \frac{2}{\Delta t}(w_i^{n}\Omega_i^{n}) + \frac{1}{2\Delta t}(w_i^{n-1}\Omega_i^{n-1}) + R_i(w_i^{n+1}) - D_i(w_i^{n+1}) = 0 \qquad (5)$$

The above equation is solved using dual time stepping scheme. This is done by solving the following equation at each time step.

$$\frac{\partial w}{\partial \tau} + R^*(w^n) = 0 \qquad (6)$$

where τ is pseudo-time in each time step, and $R^*(w^n)$ is the unsteady residual, defined as

$$R^*(w^n) = \frac{3}{2\Delta t}(w_i^{n-1}\Omega_i^{n+1}) - \frac{2}{\Delta t}(w_i^{n}\Omega_i^{n}) + \frac{1}{2\Delta t}(w_i^{n-1}\Omega_i^{n-1}) + R_i(w_i^{n+1}) - D_i(w_i^{n+1}) \qquad (7)$$

Equation (6) is a modified steady state problem in pseudo-time. This problem can be solved using explicit Runge-Kutta multistage scheme or any time marching method designed to solve steady state problems. To

accelerate convergence, local pseudo time stepping and implicit residual averaging are used. The four-stage Runge-Kutta scheme used in this paper is introduced by

$$
\begin{aligned}
w^{(0)} &= (w_i^{n+1})^m \\
w^{(1)} &= w^{(0)} - \frac{1}{4}\frac{\Delta\tau}{\Omega_i} R_i^*(w^{(0)}) \\
w^{(2)} &= w^{(0)} - \frac{1}{3}\frac{\Delta\tau}{\Omega_i} R_i^*(w^{(1)}) \\
w^{(3)} &= w^{(0)} - \frac{1}{2}\frac{\Delta\tau}{\Omega_i} R_i^*(w^{(2)}) \\
w^{(4)} &= w^{(0)} - \frac{1}{4}\frac{\Delta\tau}{\Omega_i} R_i^*(w^{(3)}) \\
(w_i^{n+1})^{m+1} &= (w)^{(4)}
\end{aligned}
\tag{8}
$$

Where

$$
R^*(w^{(l)}) = \frac{3}{2\Delta t}(w_i^{(l)}\Omega_i^{n+1}) - \frac{2}{\Delta t}(w_i^n\Omega_i^n) + \frac{1}{2\Delta t}(w_i^{n-1}\Omega_i^{n-1}) + R_i(w^{(l)}) - D_i(w_i^{n+1}) \tag{9}
$$

To increase computational efficiency, numerical dissipative term is only calculated in the first stage of Eq. (8). The allowable pseudo-time step for each cell is restricted by stability considerations and is given by

$$
\Delta\tau_i = \min\left[\frac{CFL\Omega_i}{\sum_{J=1}^{N}\lambda_i}, \frac{2\Delta t}{3}\right] \tag{10}
$$

Where j denotes edge of the corresponding cell, and

$$
\lambda = u\,\Delta y - v\,\Delta x \tag{11}
$$

4. BOUNDARY CONDITIONS

In the far field, non-reflecting boundary conditions are used based on the characteristic analysis. Solid boundaries are immersed within the grid. Therefore, boundary conditions cannot be applied by conventional methods. Instead, boundary conditions are implemented through the determination of the flow variables on the ghost finite volumes, which are defined along the solid boundaries. To determine flow variables on ghost finite volumes, an image point (I) should be defined for each ghost finite volume. Image point which is located within the flow domain is the mirror point of the center of ghost finite volume with respect to the solid boundary. Different image points are shown in Fig. 11. Flow variables on an image point are known since it is within the solution domain; this will be discussed later. Assuming zero normal-pressure gradient on the solid boundary, pressure of a ghost finite volume would be equal to its image-point pressure. For an adiabatic boundary, again temperature of a ghost finite volume would be equal to its image-point temperature. Fluid flow should have a velocity which its component normal to the solid boundary is equal to the normal velocity component of boundary. This is implemented using linear interpolation between normal velocities of GFV, image point, and solid boundary. For the case of stationary boundaries, normal velocity component at the GFV will be equal to the negative value of normal velocity component at the image point. For the velocity component tangent to the boundary, its value at the GFV is set equal to tangential component of velocity at the image point; these are shown in Fig. 12.

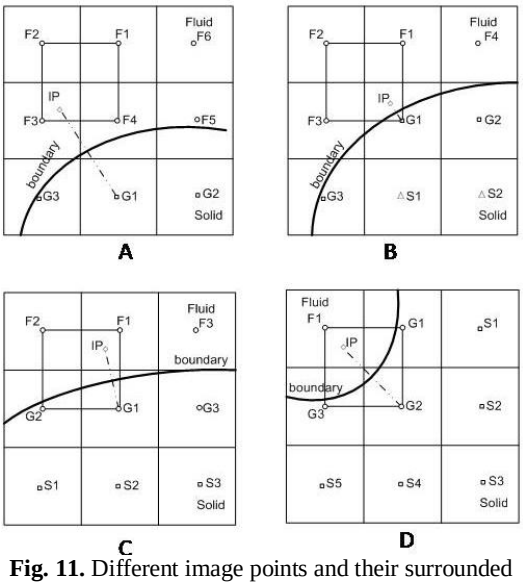

Fig. 11. Different image points and their surrounded finite volumes

At this stage determination of flow variables at the image points should be discussed. Although, it seems that flow variables can be easily determined on image point, in some cases this would be hard to do. As shown in Fig. 11a if the mage point are ghost finite volume, variables on image point are calculated using bilinear interpolation of the most recently updated variables at the GFVs; see Figs. 11b-d. In this case calculation of variables of the ghost finite volume is repeated until the converged values of variables at all of the GFVs are reached.

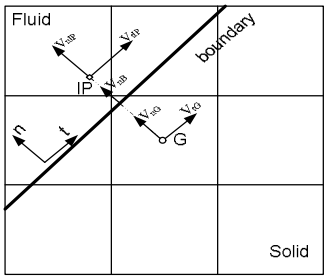

Fig. 12. Velocities on the Ghost Finite Volume

5. RESULT AND DISCUSSION

To validate present algorithm the following test cases are solved.

5.1 Stationary Airfoil

Subsonic, transonic and supersonic flows are simulated around NACA0012 airfoil at different angles of attack. First test case includes subsonic flow of Mach 0.5 passing the airfoil at 0 degree angle of attack. Solution

is carried out on three different grids coarse, middle and fine, whose specifications are given in Table 1. For each grid number of fluid finite volumes (FFV), ghost finite volumes (GFV), and solid finite volume (SFV) are given. Ghost finite volumes play the same role that boundary nodes play in conventional algorithms for the implementation of boundary conditions on body fitted grids. More accurate boundary conditions can be implemented with higher number of GFVs. As seen, the number of GFVs are doubled from coarse to middle, and from middle to fine grids.

Table 1 Specifications of coarse, middle and fine grids

Number of	Coarse grid With 5 layers	Middle grid with 6 layers	Fine grid with 7 layers
FFV	2221	4591	12094
GFV	48	104	216
SFV	30	200	998

In Fig. 13, Cp distributions of this subsonic flow on three grids are compared with each other and with the experimental results of AGARD (1985). As is obvious, results of fine grid are very close to the results of AGARD (1985). Convergence history of the fine grid solution is presented in Fig. 14. As seen after 130 real time steps error is decreased to the order of 1e -9.

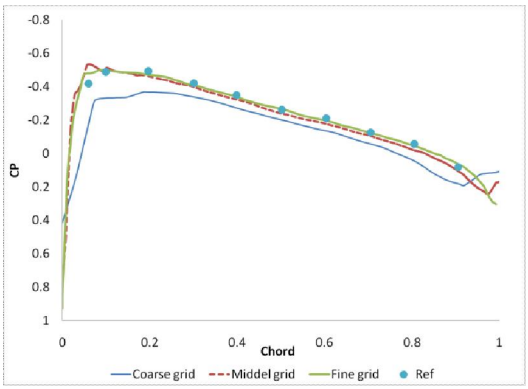

Fig. 13. Comparison between Cp distributions of NACA0012 airfoil on different grids with result of AGARD (1985); M∞=0.5, Incidence angle =0°

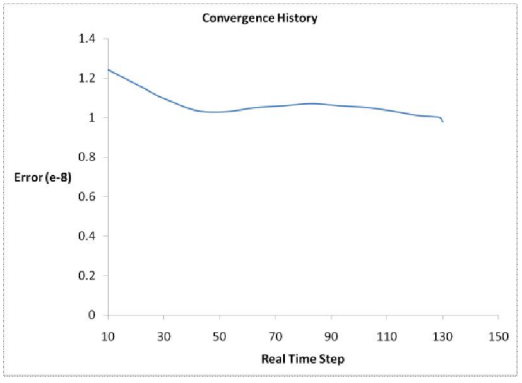

Fig. 14. Convergence history of the fine grid in the flow simulation around NACA 0012 airfoil at M∞=0.5 and Incidence angle =0°

Similarly, transonic flow of Mach 0.85 over a NACA0012 airfoil at 1 degree angle of attack is numerically solved by the present algorithm. Cp distributions of different grids are shown in Fig.15. Again results of fine grid have excellently matched the results of AGARD (1985). As seen, the captured shocks are well positioned and their strengths are correctly calculated.

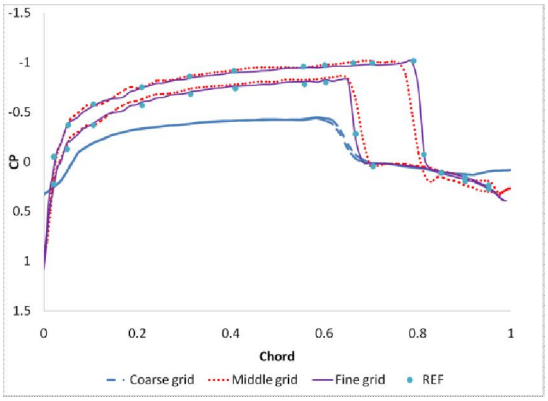

Fig. 15. Comparison between Cp distributions of NACA0012 airfoil on different grids with result of AGARD (1985); M∞=0.85, Incidence angle =1°

As mentioned earlier, flow variables of ghost finite volumes are determined in each time step based on the boundary conditions on the solid boundary. Therefore, convergence of variables at GFVs would be a good indication of solution convergence as well. Consider previous test case. For this steady state case, convergence history of pressure at GFV which is located at leading edge of airfoil, is shown in Fig. 16. As seen, after about 1000 iterations its value approaches to its converged value of 1.5.

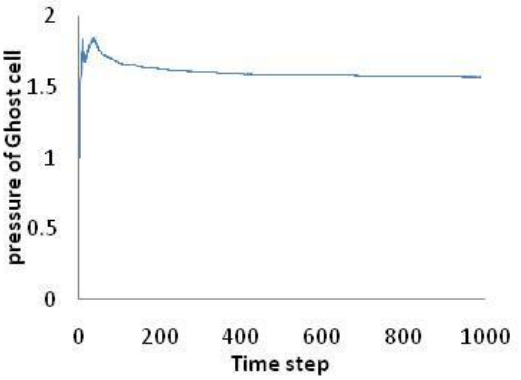

Fig. 16. Convergence history of GFV pressure in the steady state case of M∞=0.85, Incidence angle =1°

In the next case, Cp distribution of supersonic flow of Mach 1.2 over NACA0012 airfoil at 7 degrees angle of attack is plotted in Fig. 17 for the fine grid. Results of the present algorithm and that of AGARD (1985) match with each other very well. Note that the unsmooth result in the front of airfoil is due to high curvature of body geometry in this area that influence on determining GFVs' flow variables.

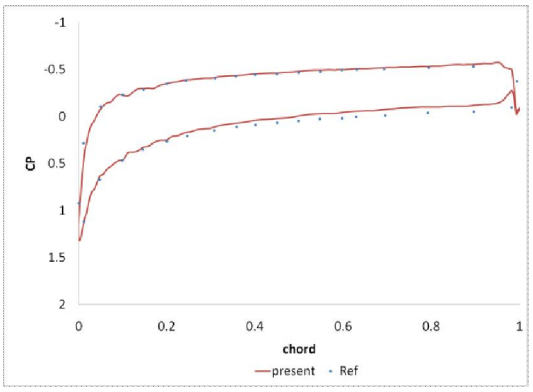

Fig. 17. Comparison between Cp distributions of NACA0012 airfoil with result of AGARD (1985); M∞=1.2, Incidence angle =7°

5.2 Airfoil with horizontal motion

Here we demonstrate some test cases to validate the capability of the present algorithm in solving unsteady flow fields with moving boundaries. First test case includes unsteady flow around NACA 0012 airfoil which moves with Mach 0.5 in a stationary fluid. After an initial transition, the flow field around the moving airfoil should become 'steady' with respect to the airfoil. For comparison purpose, this problem was also solved in the steady mode, i.e. the airfoil is stationary and the air flows with speed of Mach 0.5. Comparison of the predicted Cp distributions with the experimental data (AGARD 1985) is shown in Fig. 18. As seen, calculated Cp distributions match with each other and with the data of (AGARD 1985).
Note that the little difference between stationary and moving results is due to moving algorithm and determining boundary velocity on intersect points in order to determining GFVs' normal velocity that take effect on GFV's indirectly.

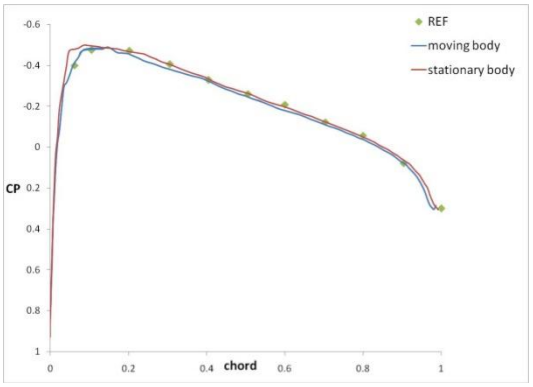

Fig. 18. Comparison of Cp distribution of NACA0012 airfoil with experimental data (AGARD 1985) in a horizontal translational movement

5.3 Airfoil with oblique motion

This test case is designed to examine the correct performance of the present algorithm in all directions. Air flow of Mach 0.5 passing NACA 0012 airfoil with incidence angle of 3 degrees will be simulated in the following three different setups.

In the first setup, airfoil is positioned parallel to the x-axes of the solution domain. Flow with Mach 0.5 and 3 degrees angle of attack then passes over the airfoil. In the second setup, an airfoil with incident angle of 3 degrees with respect to the x-axes is translated with speed of Mach 0.5 along the x axis in stationary fluid. Finally, in the third setup, a NACA0012 airfoil parallel to the x-axes, is translated with the speed of Mach 0.5 in a direction with 3 degrees incidence in a stationary fluid. The predicted Cp distributions obtained from these three cases are compared with each other in Fig. 19. All of the results match with each other. With this excellent test one can make sure that the method is perfectly implemented and is independent of grid. The little difference between stationary and moving results is same as previous test case that mention earlier.

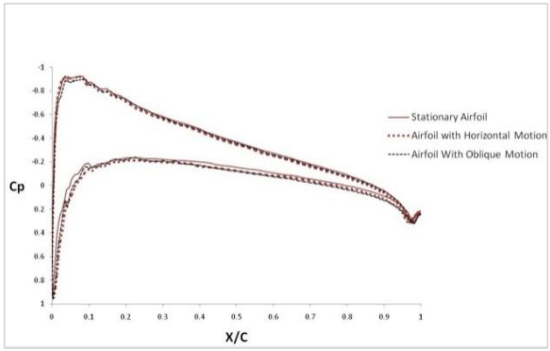

Fig. 19. Comparison of Cp distributions of flow of Mach 0.5 over NACA0012 airfoil at 3 degrees angle of attack for three different setups

6. CONCLUSIONS

In this paper, a moving-mesh algorithm is presented for the solution of two-dimensional compressible inviscid flow on Cartesian grid using Immersed Boundary Method (IBM). Solution domain is discretized to a number of finite volumes. Boundary conditions on solid boundaries are implemented throughout the determination of ghost finite volume variables in the solution domain. Grid refinement is performed in different layers around the body to prevent extra production of grid points. Flow equations are solved using dual time step method of Jameson. Numerical results obtained from the present study are compared very well with other numerical results.

REFERENCES

AGARD Fluid Dynamic Panel (1985). Test cases for Inviscid flow field methods. *AGARD Advisory Report*, AR-211.

Balaras, E. (2004). Modeling complex boundaries using an external force field on fixed Cartesian grids in large-eddy simulations. *Journal of computational fluids* 33, 375–404.

Berthelsen, P.A. and O.M. Faltinsen (2008). A local directional ghost cell approach for incompressible viscous flow problems with irregular boundaries. *Journal of computational physics* 227, 4354–4397.

Beyer, R. (1992). A computational model of the cochlea using the immersed boundary method. *Journal of computational physics* 98, 145-162.

Clarke, D.K., H.A. Hassan and M.D. Salas(1985). Euler calculations for multi-element airfoils using Cartesian grids. *AIAA Journal* 24, 353-358.

DeZeeuw, D. and K.G. Powell (1993). An adaptively refined Cartesian mesh solver for the Euler equations. *Journal of computational physics 104*, 56–68.

Fadlun, E.A., R. Verzicco, P. Orlandi and J. Mohd-Yusof (2000). Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulation. *Journal of Computational Physics* 161, 35–60.

Fauci, L.J. and A. Mcdonald (1994). Sperm motility in the presence of boundaries. *Bulletin of mathematical biology* 57, 679–699.

Goldstein, D., R. Handler and L. Sirovich(1993). Modeling a no-slip flow boundary with an external force field. *Journal of computational physics* 105, 354–366.

Ghias, R., R. Mittal and H. Dong (2007). A Sharp Interface Immersed Boundary Method for Compressible Viscous Flows. *Journal of Computational Physics* 225, 528–553.

Gilmanov, A. and F. Sotiropoulos(2005). A hybrid Cartesian/immersed boundary method for simulating flows with 3D geometrically complex moving bodies. *Journal of computational physics* 207, 457–492.

Gilmanov, A., F. Sotiropoulos and E. Balaras(2003). A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on Cartesian grids. *Journal of Computational Physics* 191, 660–669.

Jahangirian, A. and M. Hadidoolabi (2004). An implicit solution of the unsteady Navier-Stokes equations on unstructured moving grids. *24th International Congress of the Aeronautical Science*, Yokohama, Japan.

Jameson, A. and D. Mavriplis (1986). Finite Volume solution of the two dimensional Euler equation on a regular triangular mesh. *AIAA Journal* 24, 611-618.

Jameson, A. (1996). Time-dependent calculations using multi-grid with applications to unsteady flows past airfoils and wings. AIAA Paper 91-1596, *AIAA 10th Computational Fluid Dynamics Conference*, Honolulu.

Lai, M.C. and C.S. Peskin (2000). An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. *Journal of computational Physics* 160, 705-719.

Majumdar, S., G. Iaccarino and P. Durbin (2001). RANS solvers with adaptive structured boundary non-conforming grids. *Annual Research Briefs, Center of Turbulence Research*, 353–366.

Mirsajedi, S.M., S.M.H. Karimian and M. Mani (2006). A Multizone Moving Mesh Algorithm for Simulation of Flow around a Rigid Body With Arbitrary Motion. *ASME Journal of Fluids Engineering* 128, 297-304.

Mittal, R., V. Seshadri and H.S. Udaykumar(2004). Flutter, tumble and vortex induced autorotation. *Theoretical and Computational Fluid Dynamics* 17(3), 165-170.

Mittal, R., C. Bonilla and H.S. Udaykumar (2003). Cartesian grid methods for simulating flows with moving boundaries. *Journal of Computational Methods and Experimental Measurements* 4, 557-566.

Mittal, R. (2008). A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *Journal of Computational Physics* 227, 4825–4852.

Mittal, R., V. Seshadri and H.S. Udaykumar (2002). Computational modeling of fluidic micro-handling processes. *5th International Conference on Modeling and Simulation of Microsystems*, San Juan, Puerto Rico.

Mittal, R. (2005). Immersed Boundary Method. *Annual Rev Fluid Mech* 37, 239-261.

Mohd-Yusof, J. (1997). Combined immersed boundary B-spline methods for simulations of flows in complex geometries. *Annual Research Briefs*, NASA Ames Research Center/Stanford University Center for Turbulence Research, Stanford, CA, 317–327.

Peskin, C.S. (1972). Flow patterns around the heart valves. *Journal of computational physics* 10, 252–271.

Quirk, J.J. (1994). An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies. *Journal of computational fluids* 23, 125–142.

Tseng, Y.H. and J.H. Ferziger (2004). Large-eddy simulati,on of turbulent wavy boundary flow illustration of vortex dynamics. *Journal of turbulence* 5, 34.

Tseng, Y.H. and J.H. Ferziger (2003). A ghost-cell immersed boundary method for flow in complex geometry. *Journal of computational physics* 192, 593-623.

Udaykumar, H.S., W. Shyy and M.M. Rao (1996). A mixed Eulerian–Lagrangian method for fluid flows with complex and moving boundaries. *International journal of numerical methods fluids* 22, 691–712.

Unverdi, S. and G. Tryggvaso (1992). A fronttracking method for viscous, incompressible, multifluid flow. *Journal of computational physics* 100, 25-42.

Verzicco, R., J. Mohd-Yusof, P. Orlandi and D. Haworth (2000). LES in complex geometries using boundary body forces. *AIAA Journal* 38, 427–433.

Verzicco, R., M. Fatica, G. Iaccarino, B. Khalighi and P. Moin (2002). Large eddy simulation of a road-vehicle with drag reduction devices. *AIAA Journal* 40, 2447–2455.

Yang, G., D.M. Causon and D.M. Ingram (1999). Cartesian cut-cell method for axisymmetric separating body flows. *AIAA Journal* 37, 905–911.

Ye, T., R. Mittal, H.S. Udaykumar and W. Shyy (1999). An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries. *Journal of computational physics* 156, 209–240.