Treatment of Moving 3D Body on Cartesian Grid

Paulus Lahur, Graduate School of Engineering, Dept. of Aerospace Eng., E-mail: lahur@aero4.nuae.nagoya-u.ac.jp
Yoshiaki Nakamura, Dept. of Aerospace Eng., Nagoya University, E-mail: nakamura@nuae.nagoya-u.ac.jp
Furo-cho, Chikusa-ku, Nagoya 464-8603

Simulation of the flow around a moving body is a challenging problem, especially in 3D. The type of grid employed can have a significant effect on computational efficiency. Body-fitted type of grids such as structured and tetrahedral-based unstructured grids move with the body, so that global grid modification is necessary during the movement. On the other hand, non-body-fitted type of grids such as Cartesian stay stationary while the body moves across it. As a result, it requires only local modification in the vicinity of body surface, which saves significant computational effort. This paper discusses the method proposed in the presently on-going research to treat moving body problem. The current approach seems to offer a significant improvement over an existing method.

1. Introduction

Cartesian grids have been used to treat increasingly complicated geometry in 3D flow problems. So far various methods have been proposed to extend the capability of the grid to perform more realistic flow simulation. These methods make use of Cartesian grid properties that offer advantage over other types of grids. One challenging frontier to Cartesian grid researches is flow simulation around a moving body. Cartesian grid remains stationary as the body moves across it, so that only local grid information around the body surface needs to be recomputed, whereas the rest of the grid is left untouched. To the contrary, in body-fitted grids such as structured and tetrahedral-based unstructured grids, modification is required in a considerably larger grid region, and in some cases, where the movement is large, global modification is necessary.

Several studies have been carried out for 2D moving body by Bayyuk et al.[1] and Yang et al.[2] Bayyuk et al. proposed and successfully demonstrated that merging the cells intersected by body surface can eliminate the problems of appearing and disappearing cells due to body movement.

A Cartesian grid method for the moving 3D body problem is attempted in the present research, as an extension of our previous researches regarding Cartesian grid for stationary 3D body.[2] Important issues of the method are discussed in this paper.

2. Computational Scheme

The flow computation is based on Euler equations, which can be written as follows.

$$\frac{\partial}{\partial t}\int \vec{U}\, d\Omega + \int \vec{F} \bullet \hat{n}\, dS = 0 \tag{1}$$

where $\vec{U}$ is the state vector, $\vec{F}$ the flux tensor, Ω the cell volume, $\hat{n}$ the surface normal vector, and S the surface area.

The discretized form of the equations takes into account the change of cell volume in time, as shown in Eq. (2). Thus, for example, the conserved variable is the mass instead of the density.

$$\frac{\left(\Omega\vec{U}\right)^{n+1} - \left(\Omega\vec{U}\right)^{n}}{\Delta t} = -\sum_{k=1}^{m}\left(\vec{F} \bullet \hat{n}\, S\right)_k^n \tag{2}$$

where Δt is the time step. Superscripts n and n+1 indicate the time level. m is the total number of face in the cell under consideration, k the face number, $\hat{n}$ the face normal vector, and S the face area.

A 3-stage Runge-Kutta method is employed to advance the solution to the next time step. Flux computation on a cell surface is based on Hännel flux vector splitting. The flow solution on each side of the surface is extrapolated from the corresponding cell center to the surface center in order to obtain second order spatial accuracy. The solution gradient is computed using a least square method. To retain the upwind property, the contribution of the neighbor that shares the surface is not taken into account. A kind of minmod limiter is then applied to the extrapolated values in order to avoid numerical oscillation.

The maximum allowable time step is computed from the maximum velocity of fluid flow and that of solid body.

$$\Delta t = \frac{CFL \cdot \Delta L_{min}}{\max\left(\left|\vec{V}_s\right|, \left|\vec{V}\right|\right) + C} \tag{3}$$

where ΔL_{min} is the smallest length of free cells, or a fraction of it, $\vec{V}_s$ the solid body velocity, and $\vec{V}$ the flow velocity.

The flux computation at solid interface also needs to be modified to take into account the energy contribution due to body movement in the direction normal to body surface (see Eq. (4)).

$$\vec{F}_n = \left(0 \quad P \quad 0 \quad 0 \quad P\left(V_s\right)_n\right)^T \tag{4}$$

3. Cut-Cell Merging

3.1 Treatment of Moving Body Problem

Due to body movement across stationary Cartesian grid, a cut cell may “disappear”, or become completely inside the body. The reverse is also possible, i.e., a cell inside body may “appear”, or enter a fluid region outside the body. Since the cell inside body has no volume and flow solution, direct application of Eq. (2) will violate conservation, and will result in errors. Bayyuk et al. suggested merging cut cells into their neighbors such that the merged cells remains as cut cells during one time step.[1] After the step is completed, the cell is separated back to its constituent cells (see Fig. (1)).

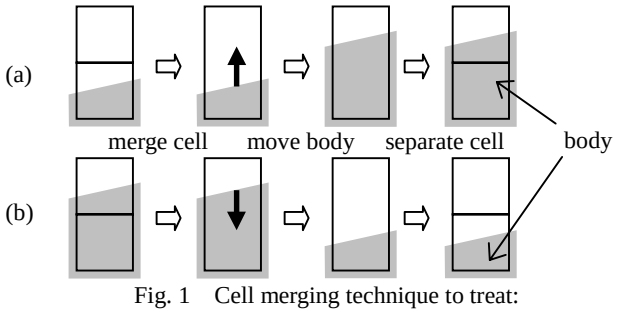


Fig. 1 Cell merging technique to treat:
(a) “disappearing” and (b) “appearing” cells.

Note that the time step is limited by Eq. (3) so as to prevent a solid cell from becoming a fluid cell in just one time step, and vice versa, which will greatly simplifiy the algorithm, as shown in Fig. (2).

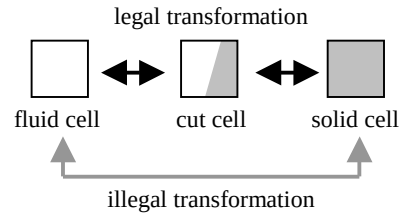


Fig. 2 Possible cell transformation in one time step as a solid body moves across a stationary Cartesian grid.

3.2 Implementation of Cut Cell Merging

The preceding discussion suggests the following computation strategy:

(1) compute the total flux of all cells at time level n,
(2) merge cut cells,
(3) move the body to the next time level, n+1,
(4) compute the cell geometrical properties at time level n+1,
(5) compute flow solution at time level n+1,
(6) separate merged cut cells.

Merging two cells can be carried out by two methods: (1) removing the physical interface between the cells, or (2) by summing the total flux and the volume of the two. As far as the computational effort and results are concerned, there is no strong reason to prefer one of the approaches. So, the latter approach is taken because it's implementation is simpler in the present code.

The next issue that has to be resolved regarding cell merging is how to determine which cells to merge. This is a very important issue in treatment of moving body in Cartesian grid, since the procedure determines the effectiveness of the treatment. In the current research, the following procedure is proposed:

(1) Cut cells at time level n are merged with neighboring cut cells at time level n or n+1. Those neighbors are in the direction perpendicular to the body surface inside the cut cells.
(2) Cut cells at time level n with volume less than a threshold are to be merged with their neighboring fluid cells. The neighbor is defined as in (1).
(3) Cut cells at time level n+1 that do not belong to any group for merging are merged to neighboring cut cells at time level n.

The only requirement during the merging process is that the basic shape of Cartesian cell is preserved. Thus, for example, when there is a merging group with an L shape, another cell is included into the group in order to satisfy the requirement. This type of complication normally occurs in the region where the body surface is diagonal to the Cartesian grid. In such case the merging group includes typically four cells in 2D, although there are rare occasions when the group includes nine cells. An illustration of cell merging is shown in Fig. (3).

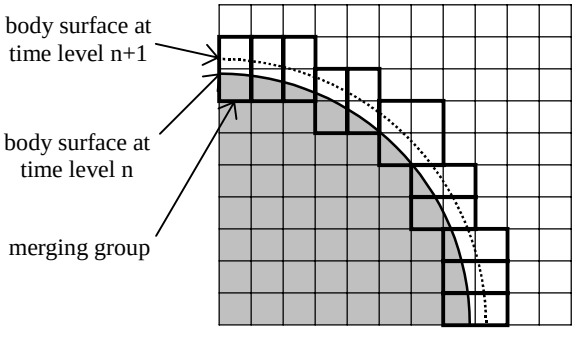


Fig. 3 Cell merging.

The cell-merging method proposed here has the following advantageous features compared to the method described in [1]:

(1) Cell merging process can start from any cut cell on body surface, whereas the method in [1] requires a suitable starting point.
(2) Whereas the method in [1] traverses neighboring cut cells along the body surface in a specified direction, so it is difficult to extend its implementation to 3D. On the other hand, the present method can proceed with no particular order, so it allows direct extension to 3D geometry.

4. Data Structure

In the simulation of moving body, reducing the computation task per time step is crucial, so that this becomes one of the main arguments in choosing the type of data structure. In unstructured approaches, cell connectivity is stored explicitly, so that finding a neighboring cell is very simple and fast. By contrast, in tree-based approaches, it is obtained by traversing up and down the tree. Although the latter approach takes less storage, more computation is required. Thus, an unstructured approach is adopted in the present research.

However, the current implementation of the method has not yet fully exploited the advantage of unstructured approach, namely solution-based grid adaptation.

5. Test Cases

5.1 Moving Piston

This test case is intended to validate the 3D method in 1D computation. The test is adapted from the Sod's test problem for shock tube, whose length is 20 m, and is coarsely discretized into 50 cells. The stationary fluid inside the tube is initially separated in the middle. On the left side the density is 1 kg/m^3 and the pressure is 100 kPa, while on the right side the density is 0.125 kg/m^3 and the pressure is 10 kPa. The resulting distributions of pressure, density, and velocity at t=0.01 second are shown in Fig. (4). Comparison with the analytical (exact) results show that the basic computational scheme is capable of capturing the flow features such as shock, expansion, and contact discontinuity.

For the piston case, the left half of the fluid is replaced with a solid piston moving to the right at the same speed as the contact of discontinuity in the Sod's test problem. It is shown in Fig. (4) that the results are quite close. The only significant difference is in the contact discontinuity; that is, in the Sod's test problem the result appears to be smeared. This is understandable, because the contact discontinuity is computed in Sod's test case, whereas in the piston case it is imposed as boundary condition.

The results of the piston case mentioned above are also compared with the case where the piston is stationary and the fluid flows toward the piston with the same velocity (see Fig. (5)). The shock in the case of moving piston is slightly more smeared than that of stationary piston, which seems to be due to the effect of cell merging.

5.2 Moving Cylinder

Cylinder is used to test the 3D algorithm outlined above in 2D flow. Two cases are computed and compared: (1) moving cylinder in stationary fluid, and (2) stationary cylinder in moving fluid (see Fig. 6). The relative velocity between the fluid and the cylinder is M=3. A uniform Cartesian grid is employed, with the length of cell equals the cylinder diameter divided by 32.

The distributions of pressure, density, and velocity at t=2 are shown in Figs. (8) and (9). It is found that both cases give quite similar results. Note that the velocity distribution shows the absolute fluid velocity, so the magnitudes of the distribution are different.

As in the piston case, the shock for the case of moving cylinder is more smeared than that for the stationary cylinder.

6. Concluding Remarks

A new cell merging method to treat moving body in Cartesian grid has been proposed. The existing method merge cells by starting from a suitable point on body surface and traverse its neighboring cut cells along the body surface. On the other hand, with the present method, cell merging starts from and progresses to any cut cell on body surface. This flexibility is considered to be important in extending the treatment to 3D body.

From the results in treatment of moving piston and cylinder, it is found that the cell merging method proposed here is effective.

References

(1) Bayyuk, S.A., Powell, K.G., and van Leer, B., "A Simulation Technique for 2-D Unsteady Inviscid Flows around Arbitrarily Moving and Deforming Bodies of Arbitrary Geometry," AIAA-93-2291-CP, 1993.

(2) Yang, G., Causon, D.M., and Ingram, D.M., "Cartesian Cut-Cell Method for Axisymmetric Separating Body Flows," AIAA Journal, Vol. 37, No. 8, August 1999.

(3) Lahur, P.R. and Nakamura, Y., "Anisotropic Cartesian Grid Generation," AIAA 2000-2243, 2000.

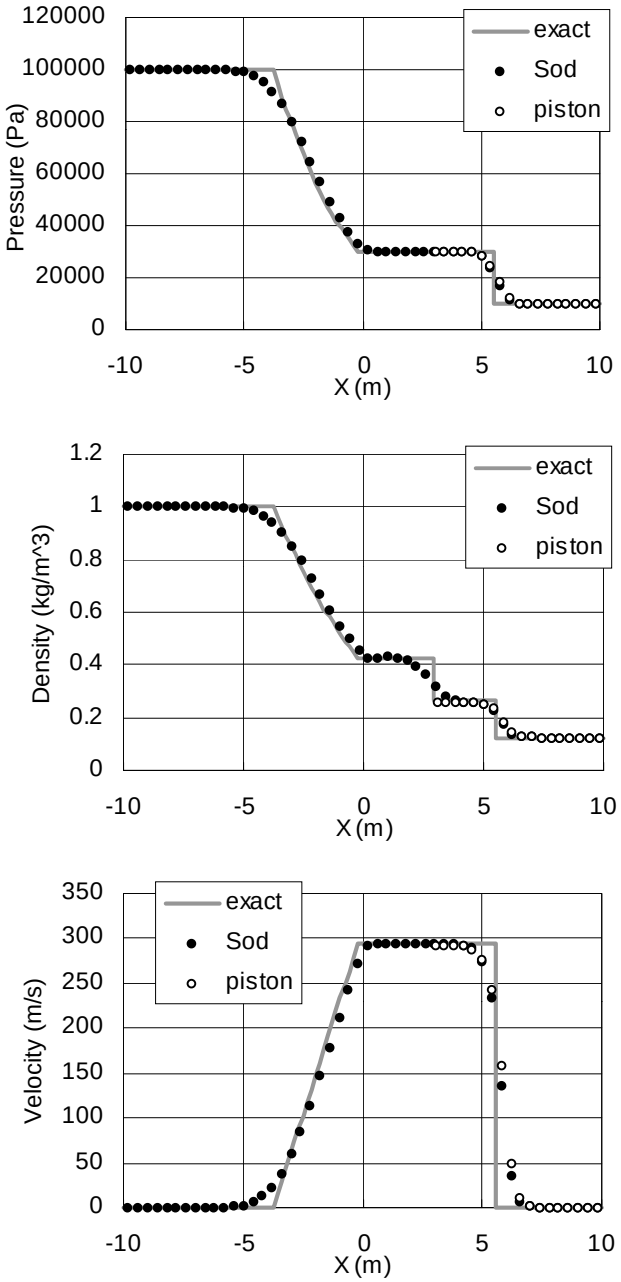


Fig. 4 Comparison between piston and the computational results of Sod's test problem.

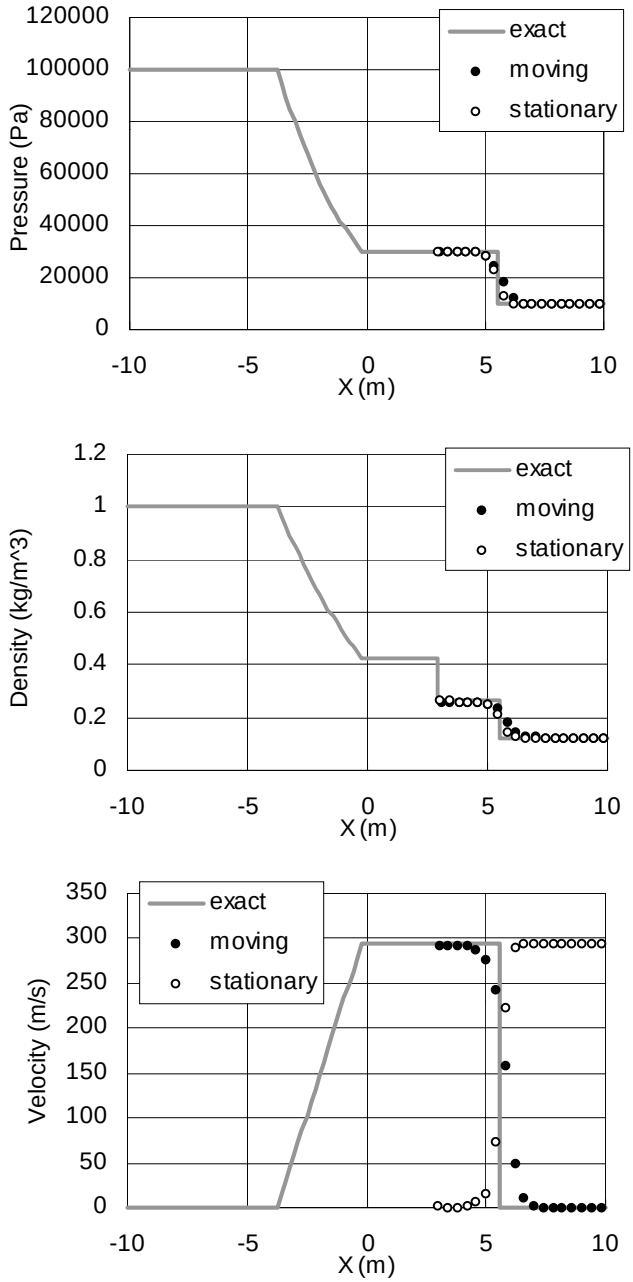
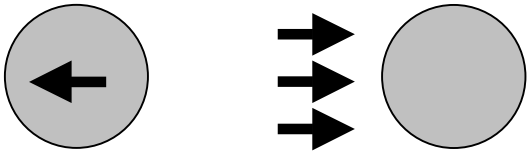


Fig. 5 Comparison between moving piston in stationary fluid and stationary piston in moving fluid.



(a) (b)

Fig. 6 Two cases of identical relative motion:
(a) moving cylinder in stationary fluid, and
(b) stationary cylinder in moving fluid.

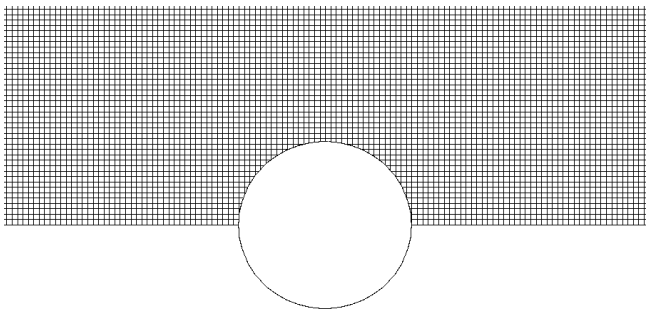
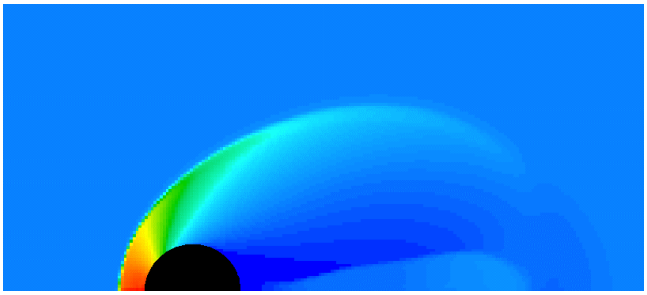
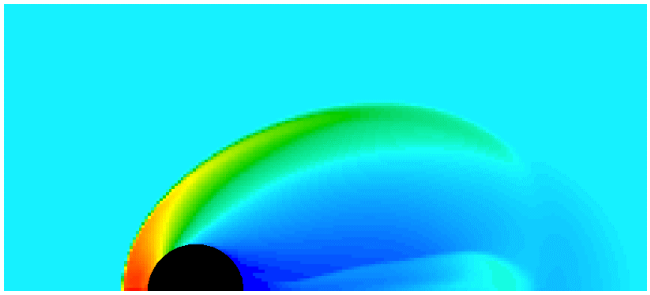


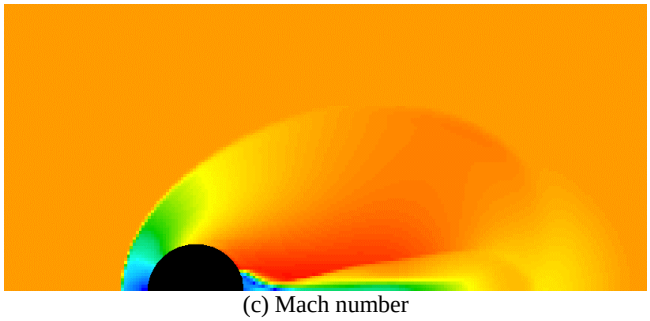
Fig. 7 Uniform Cartesian grid for the cylinder case.



(a) Pressure

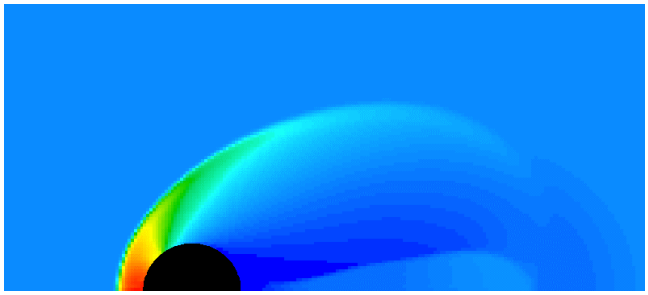


(b) Density

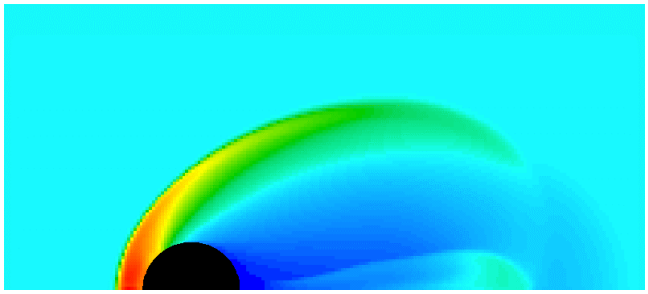


(c) Mach number

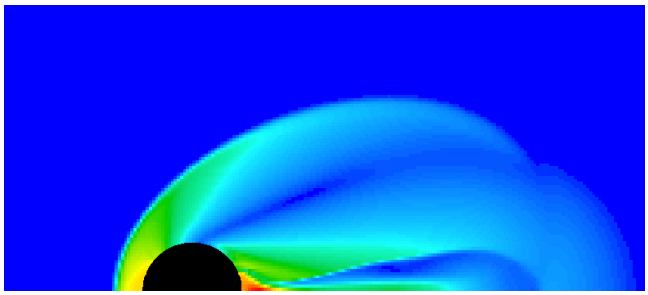
Fig. 8 Flow moving at M=3 to the right around stationary cylinder.



(a) Pressure



(b) Density



(c) Mach Number

Fig. 9 Flow due to cylinder moving at M=3 to the left in stationary fluid.