V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal, 14–17 June 2010

FAST AND STABLE TREATMENT OF NON-WATERTIGHT GEOMETRY FOR INCOMPRESSIBLE FLOW SIMULATION ON CARTESIAN GRID

Kei Akasaka*, Kenji Ono

Functionality Simulation and Information Team, VCAD System Research Program, Riken
2-1, Hirosawa, Wako-shi, Saitama 351-0198, Japan
E-mail: {kakasaka, keno}@riken.jp

Keywords: Computational Fluid Dynamics, Embedded Boundary Condition, Grid Generation, Arbitrary Geometry, Watertight, Polygon

Abstract.

Computational fluid dynamics (CFD) is now widely used as an essential tool in the development of industrial products. However, the time required for repairing non-watertight geometries has recently become a serious problem in current CFD processes. In the case of industrial products such as automobiles and electronic devices, the geometry is composed of several million polygons. In particular, a geometry including incomplete polygons is called a non-watertight geometry. In general CFD processes, all incomplete polygons must be repaired before grid generation. This repair work is time-consuming when a geometry has many incomplete polygons.

Therefore, we developed an efficient simulation method. Notably, the proposed method allows the flow around a non-watertight geometry to be simulated. The method can considerably reduce the turnaround time and effort required for implementing CFD processes because the repair work can be eliminated.

In the proposed approach, the Cartesian grid method is used. The Cartesian grid enables rapid and automatic grid generation, even if an object has a complex shape and non-watertight geometry. Moreover, this method is combined with an embedded boundary condition technique in order to capture arbitrary shapes on the background Cartesian grid with higher accuracy. In addition, a local mesh refinement technique is adopted to realize more efficient calculation, and large-eddy simulation (LES) is used to reproduce high-Reynolds-number turbulent flow.

Preliminary tests were performed using a non-watertight object and an inclined thin plate; as a result, the proposed method was found to enable rapid and stable simulation of the flow around the non-watertight geometry. In addition, this approach could be used to approximate shapes more accurately than the voxel method that is one of conventional approaches, and good agreement with experimental data was observed.

1 INTRODUCTION

1.1 Background

At present, computational fluid dynamics (CFD) is widely used as an essential tool in the development of industrial products. However, the time required for repairing non-watertight geometries has recently become a serious problem in current CFD processes. In the case of industrial products such as automobiles and electronic devices, the geometry is composed of several million polygons. In particular, as shown in Figure 1, a geometry including incomplete polygons is called a non-watertight geometry. Since most general CFD software requires a watertight geometry, all incomplete polygons must be repaired before grid generation, as shown in Figure 2. With existing technology, repairing polygons automatically remains difficult, in spite of intensive research[1]. The repair task is time-consuming and can require several days to weeks when the geometry includes many incomplete polygons. This task often accounts for the greater part of the turnaround time of the simulation. If an efficient CFD approach for avoiding the repair of polygons can be developed, the turnaround time is expected to be reduced substantially.

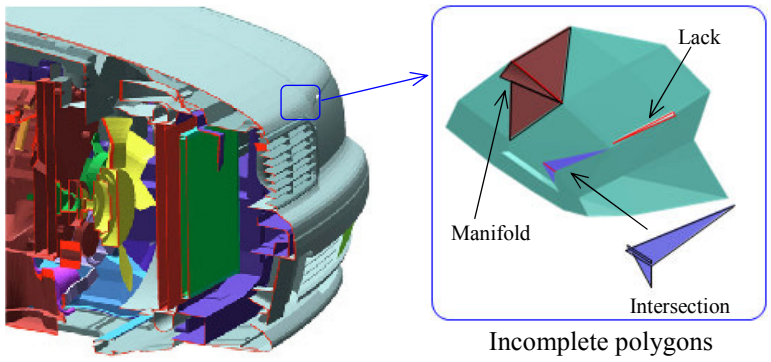


Figure 1: Example of non-watertight geometry including incomplete polygons

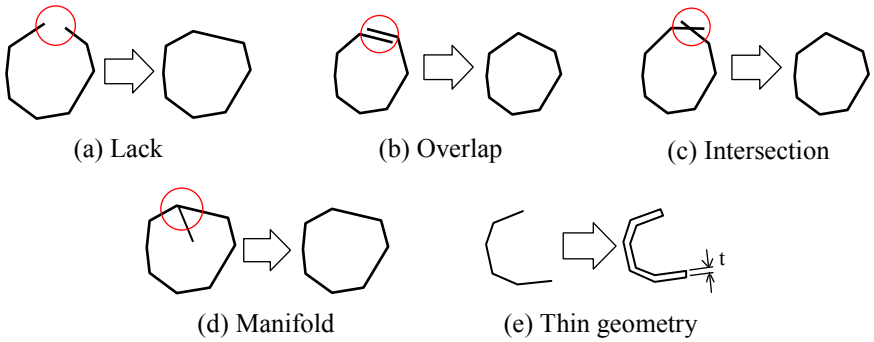


Figure 2: Repair of incomplete polygons

Here, we propose a CFD approach that tolerates non-watertight geometry (Figure 3). Since the repair of polygons is not required in this approach, the turnaround time can be shortened considerably. Although the flows around the incomplete polygons might be slightly irrelevant (see Figure 3), the proposed method is expected to be useful when an approximate flow (not detailed) around an object is needed without delay. This approach

will also reduce the burden of repair work because only incomplete polygons that interfere with the design study need to be repaired.

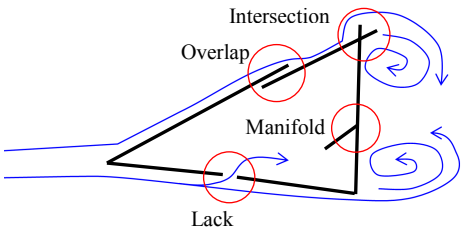


Figure 3: Flow around a non-watertight geometry

1.2 Conventional approach and proposed method

The voxel method[2,3] is a conventional approach to the flow simulation around a non-watertight geometry and is among the Cartesian grid methods. Figure 4 shows examples of shape approximation of non-watertight geometries by the voxel method. The voxel method allows uniform grids to be generated automatically in a short time. This feature is highly useful in actual product design. However, in the voxel method, the shape is approximated by cubic elements. In some cases, this shape approximation might be insufficient in terms of accuracy. Although the cut-cell technique[4] and the immersed boundary method[5] have been proposed to improve the accuracy of shape approximation on a Cartesian grid, these approaches are not applicable to non-watertight geometry.

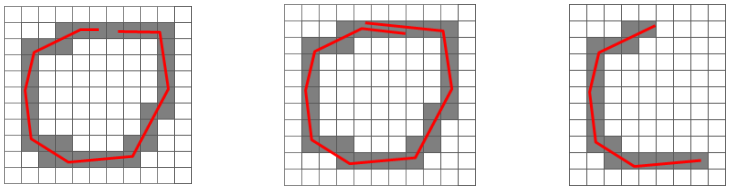


Figure 4: Shape approximation of non-watertight geometries by the voxel method

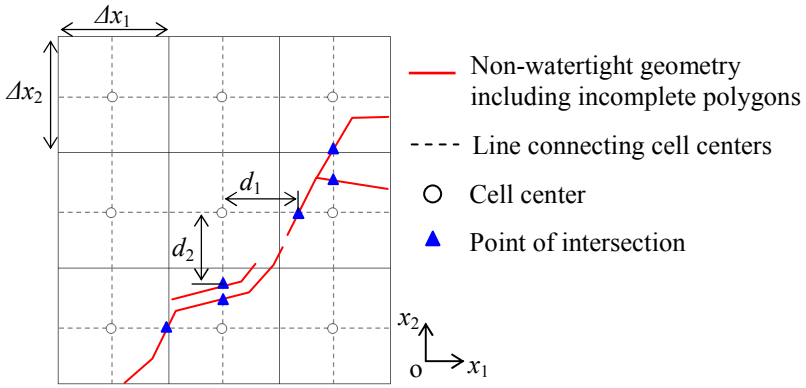


Figure 5: Definition of distance from cell center to geometric boundary in two dimensions

To address these issues, we developed a practical simulation method to improve the accuracy of shape approximation by the Cartesian grid method; furthermore, this approach can be applied to non-watertight geometries. In the proposed method, at the cell nearest to a geometric boundary, the governing equation of the fluid is discretized using the distance d_i. As shown in Figure 5, d_i represents the axial distance (d_1, d_2) from a cell center to the geometric boundary. Since the distance information is taken into

account in the discretization of the governing equation, the accuracy of shape approximation by this approach is better than that by the voxel method. Moreover, by using the computational technique[6] of the ray tracing in computer graphics, the distance can be calculated with ease. As shown in Figure 5, the distance is also acquired from non-watertight geometries.

2 GOVERNING EQUATIONS AND NUMERICAL METHOD

2.1 Governing equations

The governing equations of the incompressible fluid used in this study are the spatially filtered Navier-Stokes equation and a continuity equation:

$$\frac{\partial \tilde{u}_i}{\partial t} + \frac{\partial \tilde{U}_j \tilde{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \tilde{p}}{\partial x_i} + \nu \frac{\partial}{\partial x_j}\left(\frac{\partial \tilde{u}_i}{\partial x_j}\right) - \frac{\partial \tau_{ij}}{\partial x_j}, \tag{1}$$

$$\frac{\partial \tilde{U}_j}{\partial x_j} = 0, \tag{2}$$

where the tilde symbol (~) denotes the spatial filtering operation, p is the pressure, u_i is the velocity vector at the cell center, U_j is the velocity vector on the cell face, t is the time, ν is the kinematic viscosity, and τ_{ij} is the subgrid scale (SGS) Reynolds stress, respectively. Variables are located as shown in Figure 6.

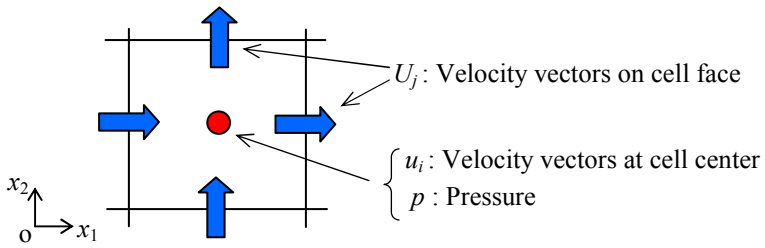


Figure 6: Location of variables on collocated grid in two dimensions

2.2 Numerical method

Both convective and viscous terms of the Navier-Stokes equation are discretized using a central difference scheme with second-order accuracy. In some cases, an upwind scheme with second-order accuracy is also used for the convective term. For the time integration, the Adams-Bashforth method with second-order accuracy is adopted. The fractional step algorithm[7] is used for the pressure-velocity coupling. The SGS Reynolds stress must be modeled; in this study, the Smagorinsky model[8] is used. In addition, the Cartesian grid is adopted and combined with the local mesh refinement technique using a nested grid method[9]. This technique allows efficient deployment of finer grids close to the geometry in order to improve computational accuracy.

2.3 Discretization of Navier-Stokes equation

The discretization method is described for the one-dimensional case for ease of presentation. The viscous term is discretized by using a central difference scheme with second-order accuracy:

$$\nu \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right)\bigg|_i = \nu\left(\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2}\right), \tag{3}$$

where the index i denotes position on the x-axis, and the tilde symbol (~) is omitted. In the proposed method, when the geometric boundary is located in the right side of cell i as shown in Figure 7, the linearly extrapolated velocity $\hat{u}_{i+1}$ [10] is used in place of u_{i+1} of equation (3):

$$\hat{u}_{i+1} = \left(1 - \frac{3\Delta x}{\Delta x + 2d}\right)\bar{u}_{i-1/2}. \tag{4}$$

Here, hat (^) and bar (¯) symbols denote extrapolating and interpolating operations, respectively; Δx is the grid size, and d is the distance shown in Figure 5. This extrapolated velocity $\hat{u}_{i+1}$ is calculated under the following two assumptions: firstly, the gradient of velocity in the vicinity of the geometric boundary is linear; secondly, the velocity at the geometric boundary satisfies the no-slip condition. Meanwhile, $\bar{u}_{i-1/2}$ denotes the linearly interpolated velocity on a cell face position calculated from u_i and $u_{i\text{-}1}$.

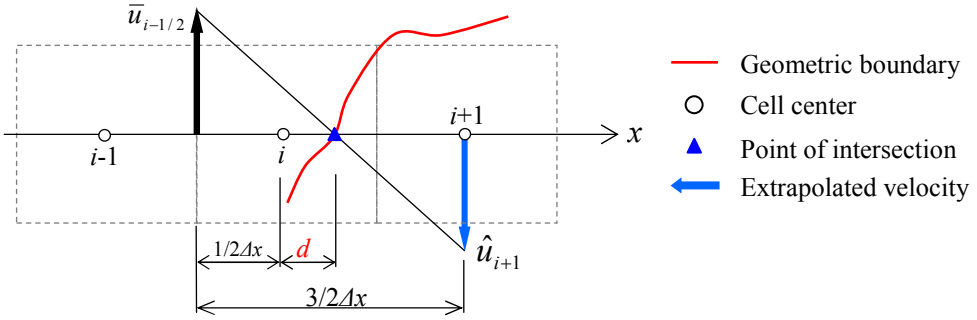


Figure 7: Schematic of velocity extrapolation assuming a linear profile in one dimension

The convective term is discretized by using a central difference scheme with second-order accuracy:

$$\begin{aligned} \frac{\partial U\, u}{\partial x}\bigg|_i &= \frac{f_{i+1/2} - f_{i-1/2}}{\Delta x}, \\ f_{i-1/2} &= \frac{1}{2}U_{i-1/2}(u_i + u_{i-1}), \\ f_{i+1/2} &= \frac{1}{2}U_{i+1/2}(u_{i+1} + u_i). \end{aligned} \tag{5}$$

Similar to the viscous term, when the geometric boundary is located in the right side of cell i as shown in Figure 7, the extrapolated velocity $\hat{u}_{i+1}$ is used in place of u_{i+1} of equation (5). The convective and viscous terms of the other axes (x_2 and x_3) are also discretized in the same manner.

It should be noted that the extrapolated velocity can be calculated by other methods. For example, it can be also determined by using u_i as shown in equation (6)[11]:

$$\hat{u}_{i+1} = \left(1 - \frac{\Delta x}{d}\right) u_i \,. \tag{6}$$

However, $1/d$ is included in equation (6); the $1/d$ term might result in computational instability if the distance d is extremely small. On the other hand, equation (4) is able to maintain computational stability when the distance d is almost zero.

2.4 Discretization of Poisson equation

The Poisson equation of pressure, which is given in equation (7), is discretized by using a central difference scheme with second-order accuracy. The discretized Poisson equation is solved iteratively by the successive over-relaxation (SOR) method:

$$\frac{\partial}{\partial x_j}\left(\frac{\partial p^{n+1}}{\partial x_j}\right) = \frac{1}{\Delta t}\frac{\partial u_j^*}{\partial x_j}, \tag{7}$$

where u_j^* is a pseudo-vector and n is the time step. In this study, equation (8) is taken as the boundary condition of pressure at the geometric boundary, except for extremely small Reynolds numbers:

$$\frac{\partial p}{\partial n} = 0 \,. \tag{8}$$

Equation (8) shows that the pressure gradient in the normal direction $\boldsymbol{n}$ is zero at the geometric boundary. In this study, the difference in the pressure gradients between the normal and axial directions is assumed to be very small since the grid resolution is extremely fine. Under this assumption, the pressure gradient in the normal direction is approximated by the pressure gradient in the axial direction, as shown in Figure 8 and equation (9):

$$\begin{aligned} &\frac{\partial p}{\partial n} \approx \frac{\partial p}{\partial x_1} = 0 \quad \text{(at point of intersection A in Figure 8),} \\ &\frac{\partial p}{\partial n} \approx \frac{\partial p}{\partial x_2} = 0 \quad \text{(at point of intersection B in Figure 8).} \end{aligned} \tag{9}$$

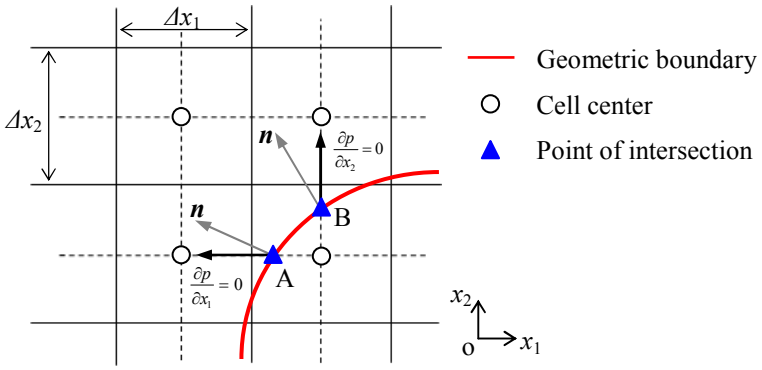


Figure 8: Schematic of pressure boundary condition at geometric boundary in two dimensions

Efficiently solving the Poisson equation is critical, taking into account the boundary condition given in equation (9). In this study, the left-hand side of the Poisson equation is discretized as shown in equation (10):

$$\frac{\partial^2 p^{n+1}}{\partial x_1^{\,2}} + \frac{\partial^2 p^{n+1}}{\partial x_2^{\,2}} = \frac{1}{\Delta x_1}\left(\psi_{i+1/2,j} \left.\frac{\partial p^{n+1}}{\partial x_1}\right|_{i+1/2} - \psi_{i-1/2,j} \left.\frac{\partial p^{n+1}}{\partial x_1}\right|_{i-1/2} \right) + \frac{1}{\Delta x_2}\left(\psi_{i,j+1/2} \left.\frac{\partial p^{n+1}}{\partial x_2}\right|_{j+1/2} - \psi_{i,j-1/2} \left.\frac{\partial p^{n+1}}{\partial x_2}\right|_{j-1/2} \right), \tag{10}$$

where

$$\psi_{i\pm1/2,j\pm1/2} = \begin{cases} 0\text{: Geometric boundary} \\ 1\text{: Fluid.} \end{cases} \tag{11}$$

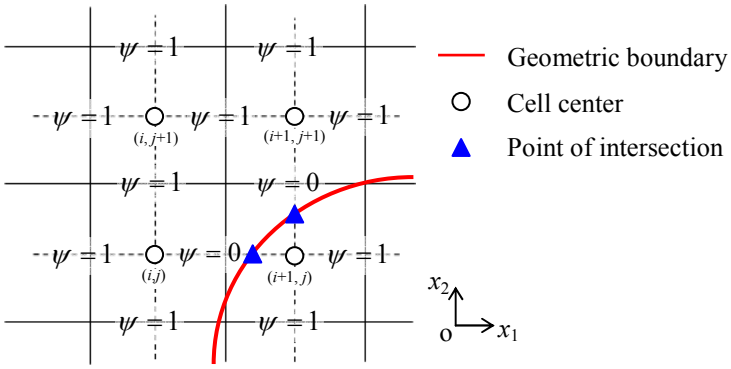


Figure 9: Function for installing pressure boundary condition in two dimensions

Here, i and j represent positions in the x_1 and x_2 directions, respectively, and ψ is a function denoting the existence of the geometric boundary. The function ψ is defined on the cell face as shown in Figure 9 and equation (11). The function can be calculated at the same time as the distance d_i. Discretization using this function allows efficient implementation[12] of the boundary condition when programming.

3 RESULTS AND DISCUSSION

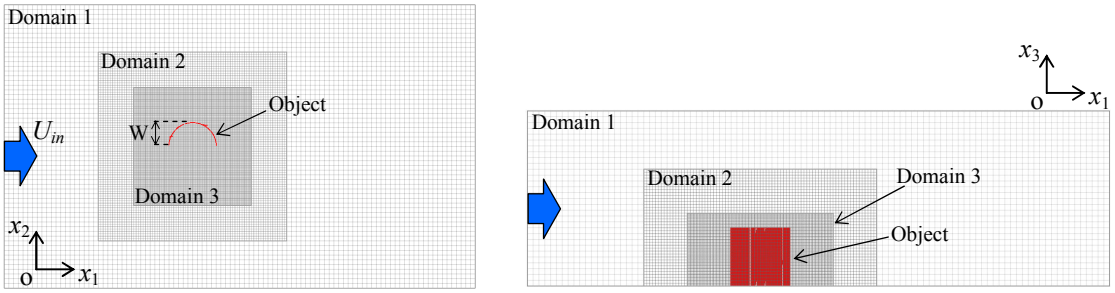
As validation and verification of the proposed method, two cases were calculated. First, the flow around a watertight object was calculated as verification. Second, as validation, the flow around an inclined thin plate was calculated and then compared with experimental data and the results of the voxel method.

3.1 Case 1: Flow simulation around non-watertight object

The computational domains and grid of Case 1 are shown in Figure 10 and Table 1. The nested grid technique was adopted for the computational grid. This nested grid was composed of three domains. The total number of grids was 0.93 million. A non-watertight object was located at the center of the computational domain. The object had a shape similar to a half-circular cylinder, and included incomplete polygons, as shown in Figure 1. At the inlet of Domain 1, a uniform velocity condition was imposed, while at the exit, free-outlet condition was adopted. The no-slip condition was imposed on the

ground, ceiling, and sidewall of Domain 1. In this case, the Reynolds number was 5.0×10^2, based on the width W of the object and the inlet velocity U_{in}. For the convective term, an upwind scheme with second-order accuracy was adopted.

Figure 11 shows velocity vectors around the object at x_3=0.5W. As shown in the figure, the proposed method enables grid generation and stable simulation even for the non-watertight geometry that includes incomplete polygons featuring lack, overlap, manifold, and intersection.

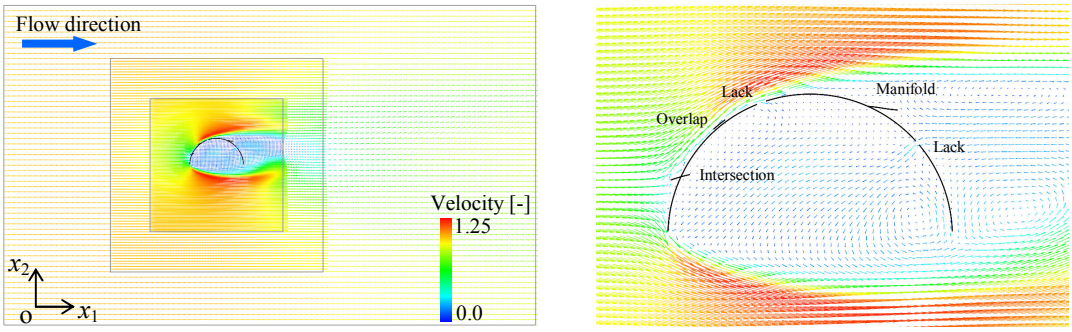


(a) Top view (b) Side view

Figure 10: Computational domain and grid for Case 1

	Grid size (Δx_1, Δx_2, Δx_3)	Dimensions (Length × Width × Height)
Domain 1	0.20W	20W × 12W × 6W
Domain 2	0.10W	8W × 8W × 4W
Domain 3	0.05W	5W × 5W × 2.5W

Table 1: Grid size and dimensions of each domain for Case 1



(a) Top view (b) Close-up view around object

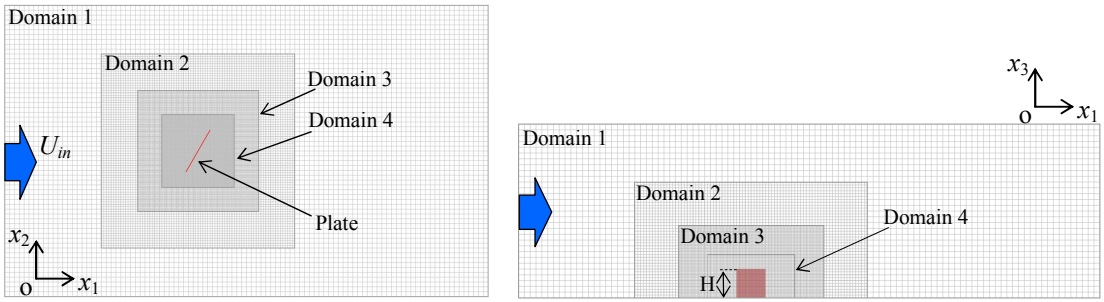
Figure 11: Distribution of velocity vectors around non-watertight object

3.2 Case 2: Flow simulation around inclined thin plate

The computational domains and grid for Case 2 are shown in Figure 12 and Table 2. An inclined thin plate was located at an angle of 30 degrees with respect to the grid lines of the background Cartesian grid. The height and width of the plate were H and 2H, respectively. In this case, up to four computational domains were used for the nested grid. The total number of grids was 1.2 million. The boundary conditions of the inlet, exit, ground, ceiling, and side wall were the same as in Case 1. In Case 2, two different flows were computed at Reynolds numbers of 5.0×10^2 and 5.0×10^4, based on the height H of the object and the inlet velocity U_{in}.

Figure 13 shows velocity vectors around the plate for the Reynolds number of 5.0×10^2. For the convective term, an upwind scheme with second-order accuracy was adopted. Three computational domains were used for the nested grid. Figure 13(b) shows a comparison of velocity vectors between the proposed method and the voxel method. This figure shows a close-up view around the center of the plate. It was confirmed that the vectors of the proposed method are along the inclined plate, whereas the vectors of the voxel method appear similar to flow along a bumpy surface. This comparison indicates that the proposed method more accurately approximates the shape than the voxel method.

Figure 14 shows that pressure distribution on the surface of the upstream side of the plate. In this case, the Reynolds number was 5.0×10^4. For the convective term, a central difference scheme with second-order accuracy was adopted. The Smagorinsky model was used for the turbulence model. The coefficient of the Smagorinsky model was taken as Cs=0.10 in this study. For the nested grid, four computational domains were used. As shown in Figure 14, the present results exhibit less fluctuation and better agreement with experimental data[13] than the results of the voxel method.

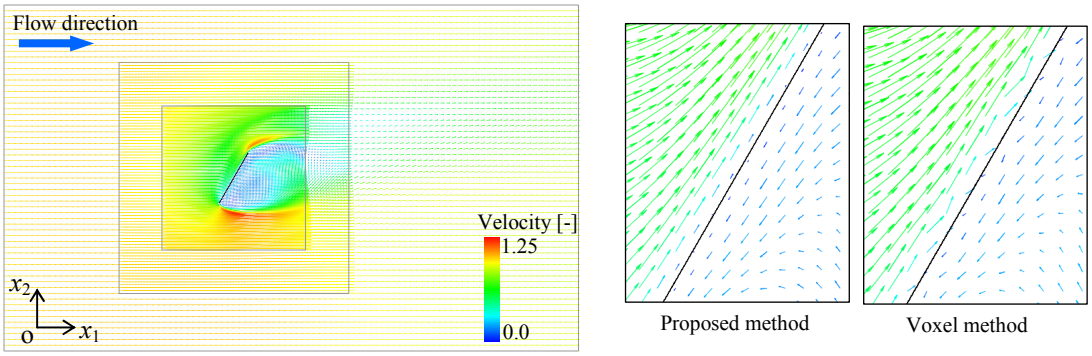


(a) Top view (b) Side view

Figure 12: Computational domains and grid for Case 2

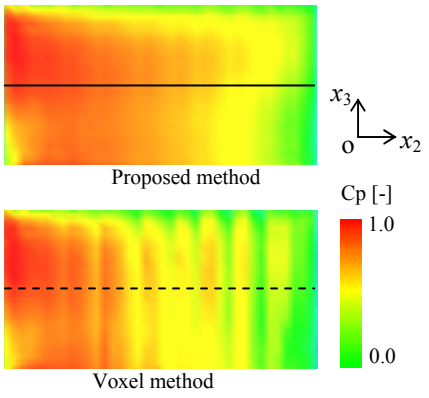
	Grid size (Δx_1, Δx_2, Δx_3)	Dimensions (Length × Width × Height)
Domain 1	$0.200H$	$20H \times 12H \times 6H$
Domain 2	$0.100H$	$8H \times 8H \times 4H$
Domain 3	$0.050H$	$5H \times 5H \times 2.5H$
Domain 4	$0.025H$	$3H \times 3H \times 1.5H$

Table 2: Grid size and dimensions of each domain for Case 2

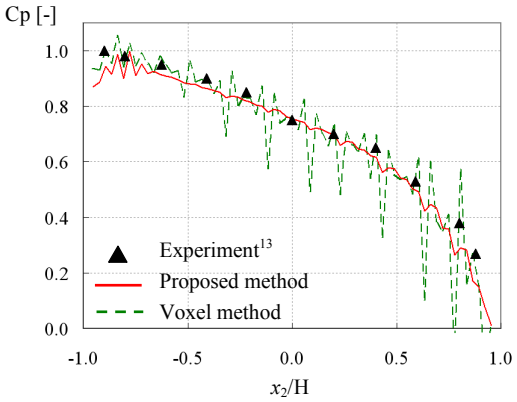


(a) Top view (b) Close-up view around plate

Figure 13: Distribution of velocity vectors around plate at Re=5.0×10^2 for $x_3=0.5H$



(a) Pressure distribution on surface of upstream side



(b) Pressure coefficient on surface of upstream side at x_3=0.5H

Figure 14: Comparison of pressure on plate for proposed method, voxel method, and experiment at Re=5.0×10^4

4 CONCLUSIONS

- The proposed method allows grid generation and stable simulation for a non-watertight object. This method is expected to contribute greatly to the reduction of time and effort required for repairing incomplete polygons, which is the most time-consuming task in CFD processes.
- The proposed method more accurately approximates shape than the conventional voxel method.
- The present results more closely agreed with experimental data than results obtained by the voxel method.

REFERENCES

[1] Z. J. Wang and K. Srinivasan, An Adaptive Cartesian Grid Generation Method for Dirty Geometry, *International Journal for Numerical Methods in Fluids*, Vol.39, pp. 703-717 (2002).

[2] K. Tsuboi, K. Miyakoshi and K. Kuwahara, Incompressible Flow Simulation of Complicated Boundary Problems with Rectangular Grid System, *Theoretical and Applied Mech*., Vol. 40, pp.297-309 (1991).

[3] K. Ono, K. Fujitani and H. Fujita, Applications of CFD Using Voxel Modeling to Vehicle Development, *In Proceedings of the 3rd ASME/JSME Joint Fluids Engineering Conference*, FEDSM99-7323 (1999).

[4] J. J. Quirk, An Alternative to Unstructured Grids for Computing Gas Dynamic Flows around Arbitrarily Complex Two-Dimensional Bodies, Computers and Fluids, vol.23, pp.125–142 (1994).

[5] E. A. Fadlun, R. Verizicco, P. Orlandi and J. Mohd-Yusof, Combined Immersed - Boundary Finite -Difference Methods for Three-Dimensional Complex Flow Simulations, Journal of Computational Physics, Vol.161, pp.35-60 (2000).

[6] T. Möller and B. Trumbore, Fast, Minimum Storage Ray-Triangle Intersection, *Journal of Graphics Tools*, No.2, pp.21-28 (1997).

[7] A. J. Chorin, Numerical Solution of the Navier Stokes Equations, *Mathematics of Computation*, No.22, pp.745-762 (1968).

[8] J. Smagorinsky, General circulation experiments with the primitive equations I. The basic experiment, *Monthly Weather Review*, Vol.91-3, pp.99-165 (1963).

[9] K. Ono, K. Akabane, H. Shiozawa and K. Fujitani, Prediction of cooling flow rate through the front grille using flow analysis with a multi-level mesh, *Soul 2000 FISITA world automotive congress*, Paper No. F2000H201 (2000).

[10] K.Akasaka and K. Ono, Simulation of Incompressible Viscous Flow on Cartesian Grid for Arbitrary Geometries Composed of Non-Watertight Polygon Elements, *Transactions of the Japan Society of Mechanical Engineers. B*, in press (2010).

[11] O. Ichikawa and K. Fujii, Study of Boundary Conditions on Solving Flow Field around Arbitrary Shape by Using Cartesian Grid, *Proceeding of 13th CFD symposium*, F03-3, p.248 (1999).

[12] K. Akasaka and K. Ono, An Implementation of Boundary Condition of Incompressible Flow Solver for Complex Geometries on Voxel Method, *Transactions of JSCES*, Vol. 2006, 20060024 (2006).

[13] S. Okamoto, M. Kobayashi, T. Kadono, H. Kagaya and A. Shimane, Influence of Attack Angle on Flow Past a Flat Plate of Finite Width Placed Vertically on a Ground Plane, *Transactions of the Japan Society of Mechanical Engineers. B*, Vol.72, No.714, pp. 322-330 (2008).