Three-Dimensional Grid Generation around a Ship Hull Using the Geometrical Method

by Yoshiaki Kodama*, Member

Summary

A grid generation method called "geometrical method" was used to generate three-dimensional grid around a Series 60 (Cb=0.7) ship hull.

The surface grid of the ship hull was generated by representing the hull surface using spline curves which connect the given offset points. The outer boundary was defined as a cylinder of a given radius. Intermediate points of the initial grid were given on a straight line connecting inner and outer boundary points.

The initial grid was modified iteratively to satisfy four requirement criteria, i.e., orthogonality, smoothing, clustering, and minimum spacing. They were quantified based on geometrical configurations of the grid. In the procedure, the grid point locations were changed gradually to satisfy those requirements, or balance in case they conflicted. Finally, the grid was reclustered in a single sweep to be applicable to high Reynolds-number flow computation. Therefore, it is possible to generate coarse grid initially with small number of grid points and then increase the number of points in a simple manner afterwards, thus minimizing CPU time and man-hour for generating the grid.

1. Introduction

Before computing flows using a finite-difference method, it is first necessary to generate grid in a flow region where the computation is to be made. To generate grid around a body of three-dimensional complex geometry is a difficult task, and it is not rare that the grid generation requires more effort and time than computing flows.

The grid generation method proposed here is called a geometrical method, in contrast to algebraic methods or partial differential equation methods. In the geometrical method, the initial grid is modified iteratively based on the geometrical information of the grid. The method was previously devised by the author, and used for the 2-D grid generation. Hino extended the method to 3-D grid around a Wigley model. Ohishi and Himeno modified the method and generated a grid around ship models in a pseudo-3 D manner. The present work is an extension and generalization of the previous work.

2. Requirement for Grid Modification

The grid modification is made based on several requirements. In the present chapter, those requirements are explained in detail. The explanations are given mainly for 2-D grid, for simplicity. Extension to 3-D grid is given wherever necessary. The grid numbering is given as (i, j) in 2-D grid, and (i, j, k) in 3-D grid. Followings are four requirements criteria adopted here.

(1) Orthogonality

When a line intersects with two parallel lines as shown in Fig. 1 (a), (b), (c), it is possible, by setting up a normal vector \( \mathbf{n} \) for the segment, to define how much the points of intersection should move so that the line segment \( (j, j+1) \) becomes orthogonal with intersecting lines. The way the points move to realize orthogonality is not unique, and can be specified by a distribution parameter \( s \), which defines the distribution ratio of the displacement vector between the \( j \) and \( j+1 \) points. Fig. 1 (a) shows the case \( s=1.0 \), in which

\[
\begin{align*}
\text{(a)} & \quad s=1.0 \\
\text{(b)} & \quad s=0.0 \\
\text{(c)} & \quad s=0.5
\end{align*}
\]

(a) \( s=1.0 \) 
(b) \( s=0.0 \) 
(c) \( s=0.5 \)

Fig. 1 Orthogonality

* Ship Research Institute
only the upper (i.e., \( j+1 \)) point moves. Fig. 1 (b) shows the case \( s=0.0 \), where only the lower point moves. Intermediate states between the two are possible by varying \( s \) in the range \( 0 \leq s \leq 1.0 \), and Fig. 1 (c) shows such an example, i.e., the case \( s=0.5 \).

Except for the boundary points, the contribution at a point comes from both upper and lower segments. Therefore, it is necessary to define the actual displacement vector \( \Delta \hat{p}_j \) as a function of upper and lower contributions. Here \( \Delta \hat{p}_j \) is defined as an average of the upper contribution \( \Delta \hat{p}^+ \) and the lower contribution \( \Delta \hat{p}^- \) weighted inversely proportional to the distances \( l^+ \) and \( l^- \) as shown in Fig. 1 (d). That is,

\[
\Delta \hat{p}_j = \frac{1}{l^+ + l^-} (l^- \Delta \hat{p}^- + l^+ \Delta \hat{p}^+). \tag{1}
\]

By giving various distribution of the \( s \) values between bottom and top ends, it is possible to obtain lines showing various orthogonality behaviors, as shown in Fig. 1 (e), where the lines connecting bottom and top ends intersect parallel lines. Both the end points are assumed to be fixed, so that the displacement is simply neglected there. In the case i), where \( s \) ranges from 1.0 at the bottom end to 0.5 at the top end, the orthogonality is exactly satisfied at the bottom end. At the top end the slope is nearly constant, which has close analogy with a simple support in the beam theory. The case ii) is the opposite of the case i). In the case iii), where \( s=1.0 \sim 0.0 \), orthogonality is satisfied at both ends. In the case iv) where \( s=0.5 \sim 0.5 \), the result is a simple straight line connecting the end points. That is, at each point, upper and lower contributions exactly cancel out. The weighting according to the distances as shown in Eq. (1) is necessary to obtain such a straight line, in case the intersecting parallel lines are not equally spaced. When \( s = 0.5 \), orthogonality acts to damp high-frequency wiggles.

Fig. 2 (a), (b) shows C-grids around a NACA 0012 wing section, where the inner boundary consisting of the wing surface and the cut in the wake represents the bottom end, and the outer boundary represents the top end. Fig. 2 (a) shows the case \( s=1.0 \sim 0.5 \) and Fig. 2 (b) shows the case \( s=1.0 \sim 0.0 \). The difference in the orthogonality behavior is clearly seen.

As shown in Fig. 3 (a), (b), a normal vector \( \mathbf{n} \) for the \((j\sim j+1)\) segment is defined as an average of \( n^*_j \) and \( n^*_j+1 \), where \( n^*_j \) is a normal vector at the point \( p^*_{j,j} \) which is defined as normal to the line connecting two points at equal distance from \( p_{j,j} \) to \( i \pm 1 \) directions respectively. This way of defining a normal vector for a segment gives symmetry to the algorithm, and, as shown in Fig. 3 (c), a circumferencial line of an \( n \)-sided regular polygon is judged as orthogonal with respect to the intersecting lines focusing at the center, and remains unchanged, even though each intersection angle is smaller than 90 degrees.

In case the grid element has very low aspect ratio, i.e. the element is very tall and thin, the grid point displacement due to this requirement
can exceed the distance between the adjacent points. Therefore a certain type of limiter which is consistent with the minimum spacing requirement described later, has been applied for this displacement, in order to avoid possible grid intersection.

In 3-D case, the normal vector in $k$-direction, for example, is generated by first making up vectors connecting two equally spaced points from the point $P_{i,j,k}$ in $i_{+}$ or $j_{-}$ directions, and then taking an outer product of the two vectors.

(2) Smoothing (Averaging)

Smoothing adds an elliptic nature to the grid. It is done by taking the average of coordinates of the neighboring points. Suppose the point $C$ in 2-D grid shown in Fig. 4 (a) shifts by the smoothing criterion. Then the new location of $C$ is defined as

$$
\vec{C} = \frac{1}{1 + 1} \left[ \frac{1}{e + w} \frac{\vec{E} + \vec{W}}{2} + \frac{1}{n + s} \frac{\vec{N} + \vec{S}}{2} \right].
$$

which shows that the new location is obtained by averaging the coordinates of the four neighboring points with weighting inversely proportional to the average distance in each direction. By adopting this weighting, the smoothing acts selectively in the high-frequency direction. A simple averaging without taking into account the distance often generates unplausible grid especially near boundaries, where the aspect ratio of a grid cell can be very high due to high-Reynolds-number computation requirement, and/or there may be a certain singularity such as a kink due to body geometry requirements. Two such schematic examples are shown in Fig. 4 (b), (c), where a simple averaging would result in the new point location $\vec{x}$. In Fig. 4 (b), where there is a kink, the new grid intersects with body surface. In Fig. 4 (c), where the grid spacing changes rapidly along the body surface, orthogonality is significantly damaged. By the use of Eq. (2.2), the above deficiencies are greatly reduced.

Extension to 3-D is straightforward.

(3) Clustering

In computing high Reynolds-number flows, it is necessary to cluster grid points near a body surface, where there is a boundary-layer and the velocity changes rapidly in a very small distance. Clustering is shown schematically in Fig. 5 (a), (b). Suppose the grid points need to be clustered toward bottom end. The distance $t_{\text{original}}$ is computed along the line by summing up the segment length, resulting in $0 \leq t \leq t_{\text{max}}$. Then the point coordinates $x, y, z$ can be viewed as discrete functions of $t_{\text{original}}$ as shown in Fig. 5 (b). Clustering is imposed by giving a minimum distance $t_{\text{min}}$ adjacent to the bottom wall. Then the points may be re-distributed by geometrical progression along $t_{\text{new}}$ axis, requiring the spacing adjacent to the wall being equal to $t_{\text{min}}$ and keeping the total length and number of points unchanged. Finally, the coordinates of the new points are obtained by interpolation using the $t_{\text{original}}-(x, y, z)$ relations.

(4) Minimum spacing

This criterion demands that each grid spacing be greater than a certain value, so that the grid line intersections (or negative Jacobian) are avoided.

First the $t_{\text{original}}$ axis is defined in the same way as in clustering. If there are segments whose length is smaller than a given value denoted as $t_{\text{min}}$, as marked with $x$-signs in Fig. 6, those segment length is made larger to be equal to $t_{\text{min}}$ by making neighboring segments that much shorter and keeping the total length $t_{\text{max}}$ unchanged. The new $t$-distribution called $t_{\text{new}}$ is then used to obtain new $(x, y, z)$ coordinates by interpolation. It should be noted that grid line intersections can occur due to excessive skewness, even when this requirement is rigorously satisfied. Therefore it is important, in some cases, to apply the orthogonality requirement together with the present one.

In actual modification procedures, the grid point displacements due to the above requirements are executed by multiplying relaxation factors smaller than unity. Generally speaking, the values of the relaxation factors should be determined depending on grid topology. A relaxation factor for the clustering requirement should be made smaller than those for other requirements, because clustering has far-reaching effect, while others affect only the adjacent points in one iteration.

Though it is possible to tune up the relaxation factors by changing the values space-dependently, that would force one to go through a lot of
trial and errors, each time one deals with a new type of grid. Therefore, constant relaxation values are used in all the grids shown in this paper. In other words, the algorithms for the above criteria have been devised to adjust automatically as much as possible to various grid topologies.

3. Surface Grid of a Ship Hull

Before constructing grid around a ship hull, a surface grid must be generated. This chapter describes how to generate a surface grid on a ship hull from the offset informations.

(1) Splines

Cubic parametric splines are used throughout this chapter, though other forms of splines such as exponential splines can also be used. In the segment \( (P_i, P_{i+1}) \), they are defined by the following equations.

\[
\ddot{q}(s) = \ddot{p}_1, p_0(s) + \ddot{p}_{i+1} - q_0(s) + \dot{p}_1, p_1(s) + \dot{q}_1, q_1(s), \quad (0 \leq s \leq 1) \quad (3)
\]

where

\[
J_i = |\ddot{p}_{i+1} - \ddot{p}_i|, \quad \dot{v} = \frac{d\dot{p}}{dt} = \frac{\ddot{p}}{ds}, \quad \dot{p}_0(s) = 2s^3 - 3s^2 + 1, \quad \dot{q}_0(s) = -2s^3 + 3s^2
\]

\[
\dot{p}_1(s) = s^3 - 2s^2 + s, \quad \dot{q}_1(s) = s^3 - s^2 \quad (4)
\]

The parameter \( s \) ranges \( 0 \leq s \leq 1 \) in the segment. The vector \( \dot{v}_j \) represents slope at the point \( P_j \). The variable \( t \) is actual length along the segment. In case a slope at the point \( P_j \) is specified as a boundary condition, the value of \( \dot{v}_j \) at the point is explicitly given. \( \dot{p}_0(s) \), \( \dot{q}_0(s) \), \( \dot{p}_1(s) \), and \( \dot{q}_1(s) \) are given functions whose forms are specified to be consistent with boundary conditions.

At the point of contact of two adjacent segments, the values of coordinates and their 1st and 2nd derivatives with respect to \( t \) are made equal, respectively. At end points, either the 2nd derivative is set zero (simple support), or the slope value is given (fixed support). Thus a scalar tridiagonal matrix system of equations results, and can be efficiently solved.

(2) Offsets

Offsets of a ship hull are given as a sequence of points along vertical lines, which consist of a stem line, a stern line, and frame lines, as shown in Fig.7 (a), where a ship hull is placed with bottom up. The \( x \)-axis is in bow-stern direction, and the \( z \)-axis is pointing upward in the depth direction. Each line can be straight or curved, or the combination of the two, as shown in Fig.7 (b). In a straight line segment, the slopes \( v \) at the both ends are explicitly given, and the slope value is used as a boundary condition in a curved line region next to it.

(3) Waterplane

The trim at F. P. and A. P. is given to define a waterplane. The coordinate transformation from original \( (x, z) \) to new \( (x^*, z^*) \) is made as shown in Fig.8, cutting off the parts above (or below in the figure) the waterplane and adding points on the plane by interpolation using splines.

(4) Re-distribution on vertical lines

The points are then re-defined on each vertical line, keeping the newly given number of points constant. The points may be clustered toward keel line and waterline, the degree of which depends on the requirements by the NS solver.

(5) Re-distribution on horizontal lines

Finally, the points are re-distributed along each horizontal line, which is constructed by connecting points at equal ordering in each vertical line. The surface grid thus obtained is shown in Fig.9.

4. 3-D Grid Generation

In this chapter the grid around a Series 60 (\( C_b = 0.7 \)) hull is shown as an example of 3-D grid generation.

(1) Initial grid

The grid has H-O topology (Fig.10), as has been used previously in the NS computation.

Fig.7 Offsets on vertical lines

Fig.8 Waterplane

Fig.9 Surface grid

Fig.10 Grid of H-O topology
The physical space \((x, y, z)\) is transformed to the computational space \((\xi, \eta, \zeta)\), whose numbering is \((i, j, k)\). The \(\xi\), \(\eta\), and \(\zeta\) axes are approximately in streamwise, girthwise, and normal-to-wall directions, respectively. The left boundary is \(x-z\) symmetry plane. The right boundary, the water-plane, is \(x-y\) symmetry plane. The bottom boundary consists of either the hull surface or the \(x-z\) symmetry plane, extending upstream from F.P. or downstream from A.P. There are points of mapping singularity on the \(x-z\) symmetry plane. The top boundary is a cylindrical surface with a given radius. The grid points on the bottom and top boundary surfaces are fixed in the subsequent grid modification procedures. The part of the bottom surface which extends upstream from F.P. or downstream from A.P., are therefore fixed in the subsequent modification. The points on the left or right boundary are treated as intermediate points, by adding points beyond the plane of symmetry, and therefore move in the same way as other intermediate points.

The generated grid around a Series 60 \((C_b=0.7)\) ship hull is shown in Fig. 11. The related parameters are given below.

Parameters for Series 60 \((C_b=0.7)\) grid
- Number of points \(i=61\), \(j=15\), \(k=21\)
- Radius of outer boundary \(O.5\)
- \(\zeta_{min}\) \(0.005\)

(2) Grid modification

The requirement criteria used for grid modification are:

- (a) Orthogonality in \(\zeta\)-direction \((\varepsilon=1.0-0.5)\)
- (b) Smoothing
- (c) Clustering
- (d) Minimum spacing in \(\xi\)-direction
- (e) Minimum spacing in \(\eta\)-direction
- (f) Minimum spacing in \(\zeta\)-direction

In the following two figures, comparison of initial and modified grid is shown. Fig. 12 (a), (b) shows the grid at nearly-midship section, viewed from downstream. The grid is made orthogonal at the hull surface in the modified grid. Fig. 13 (a), (b) shows the sectional grid at AP viewed from behind with 30 degrees looking-down angle. The modified grid is smooth everywhere, in spite of the presence of the point of mapping singularity.

(3) Re-clustering

In order to compute high-Reynolds number flows, it is necessary to cluster further the grid points toward a body surface. A simple criterion widely used for the minimum spacing adjacent to the wall as a function of the Reynolds number is

\[
\zeta_{min} = 0.05 \sqrt{Re},
\]

where the body length is normalized as unity. Therefore, if a flow with \(Re=1,000,000\) is computed, \(\zeta_{min}\) should be \(5.0 \times 10^{-5}\), and the grid is forced to have extremely high aspect ratio near the body surface.

A straightforward procedure of first generating initial grid by satisfying such clustering criterion and then modifying the grid by the modification procedures described above would result in failure.
because even the slightest movement of the grid point would easily cause intersection in such a densely-packed grid.

The re-clustering procedure described below has been devised to circumvent this difficulty. Suppose a grid is generated by requiring clustering with a given $D_{\text{min}}$ together with orthogonality, smoothing, etc. One such example is shown in Fig. 14 (a), where it is clearly seen that the minimum spacing adjacent to the bottom (inner) boundary is not uniform but has a greater value at the concave side of the kink at the trailing edge. This has occurred due to the smoothing criterion, etc. If the clustering requirement alone is applied, the kink propagates far into the grid zone without damping, as shown in Fig. 14 (b). Therefore, it may be stated that deviations of the grid point distribution from the clustering requirement occurs as a compromise with other requirements.

The key to successfully re-distributing points without propagating kinks depends on how well one can reflect the information inherent in the original grid to the re-distributed grid. The method is now described (Fig. 15). First, $t^{\text{original}}$ axis is defined by summing up segment length, as in clustering. Second, $\bar{t}^{\text{original}}$ axis is defined by re-distributing points by exactly satisfying $D_{\text{min}}$ and keeping the total length constant. Usually the $t^{\text{original}}$ and $\bar{t}^{\text{original}}$ distributions differ, due to the requirements stated above.

To take advantage of the information inherent in the original grid, a new parameter $r_d$, which is called a distance ratio, is introduced:

$$r_d = \frac{t^{\text{original}}}{\bar{t}^{\text{original}}}$$

Near the concave kink the grid point tends to move away from the surface, and $r_d$ becomes greater than unity, as shown in Fig. 15. At the top end ($t=t_{\text{max}}$), $r_d$ becomes unity by definition. At the bottom end ($t=0$) $r_d$ is set to unity. A new $\bar{t}^{\text{new}}$ distribution is defined by giving new $D_{\text{min}}$ and new number of points (possibly greater than the original number), corresponding to the high Reynolds-number computation. Then the $t^{\text{new}}$ distribution is defined by multiplying $r_d$ value obtained at each $\bar{t}^{\text{new}}$ point by interpolation, with
Finally, \((x, y, z)\) coordinates are interpolated at each \(t^{\text{new}}\) point using \(t^{\text{original}} - (x, y, z)\) relation initially defined. The final grid distribution made through the re-distribution procedure stated above is shown in Fig. 16, where the number of grid points has remained unchanged and \(\Delta_{\text{min}}\) has been changed from 0.005 to 0.0005. It is clear that the information inherent in the original grid (Fig. 14(a)) is well reflected in the new distribution.

Using this technique, the grid shown in Fig. 11 has been re-clustered. The number of points in \(k\)-direction has changed from 21 to 31, and \(\Delta_{\text{min}}\) from 0.005 to 0.0005. The result is shown in Figs. 17 and 18. Fig. 18 shows near-midship section, corresponding to Fig. 12(b). Thus the validity of the reclustering procedure has been confirmed.

5. Conclusions

In constructing a system, there is always a trade-off between perfectness and efficiency. Suppose one wishes to make a grid generation system which can be applied to various type of grids. One way is to exhaust all the cases that can happen and construct a system which can handle them all. The problem with this type of system is that it can be very complex and heavy, demanding a lot of debugging effort and CPU time, in addition to the fact that exhausting all possible cases is practically impossible. The other is to make up a system by combining small separate parts. In this way one can handle various type of grids by changing the combination, depending on the type of grid. In order to make this type of system work well, a user must have knowledge about each component to such an extent that he or she can judge which component should be used with what degree of weighting, depending on the type of grid. Considering the fact that a computer is not good at pattern recognition which is essential to the grid generation, the author believes that the latter, in which human intelligence is combined with computing power, is more realistic than the former.

The present grid generation system belongs naturally to the latter category. The geometrical grid generation method previously proposed by the author for 2-D grid has been generalized and
extended to 3-D grid, and is applied to a Series 60 hull form. The criteria for grid modification are, orthogonality, smoothing, clustering, and minimum spacing, all of which are valid both to 2-D and 3-D grids.

None of the algorithms is perfect, which means that there are chances when it fails in generating "good grid". For example, though the minimum spacing requirement is introduced to avoid grid intersection, the grid can intersect due to excessive skewness even when the minimum spacing requirement is rigorously satisfied. However, the failure can be avoided by applying orthogonality requirement together with the minimum spacing. Then the grid will become less skewed and less liable to intersect.

In 3-D grid generation, it is practically very difficult to "see" all the grid distributions, even with the use of high-speed graphics on a high-end graphic workstation. The algorithms presented in this paper can be used to quantify the grid quality. For example, the orthogonality algorithm can measure the angle between a line segment and a normal vector. Development along this line should be made in the future to bring intelligence into the grid generation system.

Finally, the present grid generation system can be used to "tune-up" existing grids. That is, an existing grid can be modified to improve certain qualities such as orthogonality or smoothness, globally or locally, using the present system. Examples will be given elsewhere6).

Computed results of flows using the grid thus generated will be presented in the near future.

References