Practical Grid Generation Tools with Applications to Ship Hydrodynamics

Eça L.
Instituto Superior Técnico
ea@marine.ist.utl.pt

Hoekstra M.
Maritime Research Institute Netherlands
M.Hoekstra@marin.nl

Windt J.
Maritime Research Institute Netherlands
J.Windt@marin.nl

Abstract

This paper presents 2-D, surface and 3-D grid generation tools for building structured grids. Algebraic, orthogonal and elliptical grid generation techniques are applied to standard test cases of Numerical Ship Hydrodynamics: a surface grid on the P4119 propeller blade for a boundary element method and volume grids for the viscous flow calculation around the KVLCC2 tanker at model and full scale Reynolds numbers. Special emphasis is given to the possibility of blending the different grid generation techniques to optimize the deviations from orthogonality and the grid line spacing.

Introduction

The efficient generation of grids around complex geometries remains one of the great challenges of Computational Fluid Dynamics. In industrial applications grid generation is often the most time-consuming task. Although for that reason the popularity of unstructured grids has increased, there are several fields where the use of structured grids is the rule rather than the exception. Ship hydrodynamics is an example and this paper will be concerned with structured grids only.

The viscous flow around a ship stern is characterized by very large Reynolds numbers that may reach the order of $10^9$ at full scale. The numerical simulation of such flows requires grids with the following properties: orthogonality at the ship surface where the no-slip condition is applied and high stretching of the grid towards that surface to resolve the flow in the near-wall region. It is not always easy to meet these requirements, because grid orthogonality is often in conflict with grid line distance control.

In recent years, practical grid generation tools for the efficient generation of grids around ship hulls have been developed at IST and MARIN. These tools include algebraic, orthogonal and elliptic grid generators for 2-D, surface and 3-D grids.
The orthogonal grid generators have been described in [1] and [2] and the elliptical grid generators are based on the so-called Grape approach, [3]. In this paper we give a brief description of these tools, focusing on the main advantages and drawbacks of each technique. A special emphasis is given to the surface grid generation methods and to the way in which the surface definition is incorporated in the grid generation.

The efficacy of the available tools is demonstrated on the generation of a surface grid on the P4119 propeller blade for a boundary element method and on a grid around the KVLCC2 tanker for a viscous flow calculation at model and full scale Reynolds number. Both are standard test cases in Numerical Ship Hydrodynamics. It is shown that by blending of the different techniques the best compromise between grid line distances and orthogonality is obtained. In the second application we also present some of the main features of the flow calculation to highlight the advantages of a ‘good quality’ grid.

The present paper is organized in the following way: section 2 gives a brief description of the 2-D, 3-D and surface grid generation tools; section 3 presents two examples of the application of the present tools and section 4 summarizes the main conclusions.

**Grid Generation Tools**

**2-D**

Two techniques are available for the generation of 2-D grids: an orthogonal grid generator, [1], and an elliptical grid generator based on the Grape approach, [3]. The generation of 2-D orthogonal grids in practical applications may be hard, because the orthogonality condition often leads to unacceptable interior grid line spacings. However, when there is no strict need to specify the grid coordinates on all the boundaries, the orthogonal grid generator may be used to find the “optimal” boundary point distribution allowing the grid nodes to move along the boundary. This possibility makes the 2-D orthogonal grid generator very useful for external flow calculations, where the boundary point distribution at the outer boundary is quite flexible.

In 2D elliptical grid generation, the use of ‘control functions’ directly calculated from the boundary point distribution no longer represents the state of the art, [4]. Nevertheless, the Grape approach is still a good option for the generation of grids for viscous flow calculations, because it allows the control of the orthogonality at the boundary and of the near-wall grid line distances.

The ‘control functions’ are calculated at the boundaries of the domain and then interpolated to the interior of the domain. The so-called Non-linear terms of the ‘control functions’ contain second derivatives at the boundaries which require information from the interior grid nodes, whereas the Linear terms may be directly calculated from the boundary point distribution, the orthogonality condition and
the distance of the first grid node to the boundary.

One of the main difficulties of the Grape approach is the calculation of the Non-linear terms of the ‘control functions’ in regions with small grid line spacings. These terms are inversely proportional to the distance of the first grid node to the boundary, hence may become extremely large in grids for viscous flow calculations. Therefore, we have introduced the possibility to generate the first layers of grid nodes close to the boundary with a straightforward algebraic technique based on the orthogonality to the boundary and on the desired grid line distance. The calculation of the Non-linear terms, including second derivatives with respect to the direction perpendicular to the boundary, is shifted to the border of the region defined with the algebraic procedure. Thus the region of smallest grid line spacing is avoided and a simple discretization of the second derivatives is permitted.

**3-D**

The two 2-D techniques mentioned above have also 3-D versions. The 3-D orthogonal grid generation technique, which is discussed in [2], has an highly questionable usefulness in complex geometries. However, the possibility to tune boundary point distributions with the orthogonal grid generator is carried over to 3-D. Therefore, the orthogonal technique may again be utilized as a preprocessor to “optimize” the boundary node locations for the Grape approach.

The ‘control functions’ of the Grape approach in 3-D have the same kind of properties as in 2-D, but are more difficult to handle, see for example [5]. The Non-linear terms are equivalent and the algebraic technique mentioned above for the 2-D code can also readily be applied to 3-D. However, the greatest difficulty with the 3-D Grape ‘control functions’ is to decide what to do with all the Linear terms in the interior of a given domain once they have been calculated at the six boundaries. It is our experience that the Non-linear terms are not sufficient to guarantee a smooth interior grid line spacing in the typical grids of viscous flow calculations with high stretching. Therefore, in our approach one may select from various types of interpolation of the Linear terms. At the interior grid nodes, the Linear terms contribution may be interpolated from the values calculated at the six boundaries of the domain or from the values at only four boundaries of the domain. When only four boundary values are selected, we have even considered the possibility to compute the Linear terms using the 2-D equations.

**Surface grids**

One of the difficulties of surface grid generation is to ensure that all the grid coordinates belong to the true surface. Therefore we have chosen to use the surface definition based on two parametric coordinates, \( u \) and \( v \), and a set of transformations \( x = x(u, v) \), \( y = y(u, v) \) and \( z = z(u, v) \), which may be analytical expressions or surface-spline interpolations. By doing all grid manipulation with the parametric coordinates \( u, v \) as dependent variables, grid nodes are necessarily on the
Two different methods have been developed for surface grid generation: an orthogonal generating system based on a system of partial differential equations, [6], and a recently developed algebraic grid generator based on grid line distance control along families of grid lines. The orthogonal grid generator is described in detail in [6]. It optimizes the grid orthogonality but it has no control of the grid line spacing. On the other hand, the algebraic method gives a perfect control of the interior grid line spacing, but it controls the grid orthogonality only at the boundaries. The objective of the algebraic method is to generate a surface grid with a specified grid node stretching along individual grid lines. The stretching is usually defined at the boundaries and then interpolated to the interior of the domain. However, the domain may be divided into several sub-domains to obtain the desired stretching distribution

1. The iterative algorithm to determine the surface grid nodes that satisfy the specified stretching is very simple. Starting from an initial approximation, which may be generated by a simple transfinite interpolation in the \((u, v)\) plane, each iteration is composed of two sweeps, one in each family of grid lines. For example, for a grid defined in the \(\xi, \eta\) computational domain with \(n_x\) nodes in the \(\xi\) direction and \(n_y\) nodes in the \(\eta\) direction, a sweep in the \(\xi\) grid lines\(^2\) is composed of the following steps:

   - From \(j = 1\) to \(j = n_y\):
     - Calculate the distance along the grid line. At the boundaries \(j = 1\) and \(j = n_y\) this is straightforward because these lines belong to the surface definition and so it is easy to determine the corresponding values of \(u\) and \(v\). However, the interior grid lines \(\eta = \text{constant}\) may have an arbitrary shape and so the following procedure has to be performed:
       1. Generate a spline representation of \(x, y\) and \(z\) in the \(\xi\) grid line with a local independent variable\(^3\), \(s\).
       2. At each point defined by this spline, \((x_s, y_s, z_s)\), invert a 2-D system to obtain \(u\) and \(v\), which are the independent variables of the surface definition.
     - Knowing the total length of the grid line, distribute the nodes \((x_s, y_s, z_s)\) along the line according to the desired stretching distribution.
     - Find the corresponding \(u, v\) values by interpolation.
     - Obtain \(x, y\) and \(z\) from the values of \(u\) and \(v\).

---

1In the limit, the stretching may be given as an input parameter to every grid line.
2The sweep in the \(\eta\) grid lines is perfectly equivalent.
3Currently, it is just the grid node counter, which in this case varies from 1 to \(n_x\).
The most delicate phase of the algorithm is the calculation of $u$ and $v$ from the values of $(x_i, y_i, z_i)$. This physical location may not belong to the true surface, because the spline is calculated with a local independent variable, $s$. The local spline is essential to ensure that we have smooth grid lines to perform the distance based interpolation. To ensure that the grid nodes are on the true surface, two coordinates must be selected to obtain $u$ and $v$ from $(x_s, y_s, z_s)$.

In general, it is not simple to eliminate $x$, $y$ or $z$, because the surface orientation may change. Therefore, a local 2-D Cartesian coordinate system is introduced with the origin at $(x_o, y_o, z_o)$ computed from $(u(i_o, j_o), v(i_o, j_o))$, where $(i_o, j_o)$ are the indexes of the grid node of the left corner of the grid line segment where the distance is being calculated. The two local axes are on a plane tangent to the surface at the coordinate system origin. The projection of the $(x_t - x_o, y_t - y_o, z_t - z_o)$ position vector to the local coordinate system defines the point $(x_j, y_j)$. These coordinates are the basis for the determination of the point of the parametric space, $(u, v)$, that has exactly the same projection for $(x(u, v) - x_o, y(u, v) - y_o, z(u, v) - z_o)$. This procedure leads to a system of two non-linear equations that is solved by Newton’s iteration. In general, the convergence is very fast because $(u(i_o, j_o), v(i_o, j_o))$ is not too far from the solution. However, one has to be careful close to grid singularities, like for example at the tip of a propeller blade. The system of non-linear equations might have no solution if the surface geometry is poorly defined.

The procedure described above generates grids with an almost perfect control of the grid line spacing, but it lacks control of the grid line orthogonality. Therefore, we have included the possibility to have grid line orthogonality at the boundaries. The number of grid lines affected by this control is selected from the input and it may be applied also at the guide lines selected to define the grid line distances.

For the sake of simplicity, we describe the procedure to the boundary $\eta = 0, j = 1$. The application of the algorithm described above generates a new set of values of the parametric coordinates, $u_o, v_o$, that satisfy the required stretching in each $\xi$ line. Alternatively, at the selected number of grid lines above the boundary, that we will designate by INODE, we compute a different set of values for $(u, v)$ that satisfies the orthogonality condition in the following discrete form:

\[
\begin{align*}
(x(i + 1, j_s) - x(i - 1, j_s))(x(s) - x(i, j_s)) & + \\
(y(i + 1, j_s) - y(i - 1, j_s))(y(s) - y(i, j_s)) & + \\
(z(i + 1, j_s) - z(i - 1, j_s))(z(s) - z(i, j_s)) & = 0, 
\end{align*}
\]

where $j_s$ may be equal to 1 or to $j - 1$. In the first case the line remains orthogonal to the boundary and in the second one it gives orthogonality to the line below. This choice is made by an input parameter. The solution of equation (1) is performed along the selected $\xi$ grid lines. Therefore, the procedure described above to ensure that the grid nodes remain on the true surface is also applied. The or-
The orthogonality condition generates a new set of values for \((u,v)\) that we designate by \(u_o, v_o\). The final value of \((u,v)\) to obtain the new coordinates \(x, y\) and \(z\) is calculated by Hermite interpolation along the \(\eta\) direction with \((u,v) = (u_o, v_o)\) at \(j = 1\), \((u,v) = (u_a, v_a)\) at \(j = 1 + \text{INODE}\) and zero derivatives at \(j = 1\) and \(j = 1 + \text{INODE}\). This procedure guarantees a smooth blending between the solution from the orthogonal condition and the grid lines obtained from the required stretching.

### Applications to Ship Hydrodynamics

#### P4119 Propeller Blade

The P4119 Propeller has been extensively used as a reference test case for Boundary Element Methods, BEM, [7]. The conventional construction of grids on propeller blades for BEM methods uses a family of grid lines coinciding with blade sections at constant radius. The solution at the tip of the blade is then troublesome, because the section chord goes to zero and the two families of grid lines become almost parallel. Therefore, the use of nearly-orthogonal grids on the propeller surface is a valid alternative to the conventional grids, [8].

In the present examples, the surface of the propeller is represented by a 2-D cubic spline interpolation on a conventional grid with 81 grid nodes along each blade section and 41 grid nodes in the radial direction.

Figure 1 presents a grid of \(65 \times 17\) grid nodes obtained with the algebraic surface method using cosine stretching distributions on the two families of grid lines. The maximum deviation from orthogonality is 28.3° and the mean deviation from orthogonality is 8.8°. The generation of an orthogonal grid on the propeller blade surface with the same boundary point distribution is troublesome, due to the grid singularity at the tip of the blade. The application of the orthogonal grid generator to the complete domain leads to an unacceptable maximum deviation from orthogonality of 86.2° close to the tip. However, it is possible to blend the two

![Figure 1: 65 \times 17 algebraic grid on the blade of the P4119 propeller.](image-url)
techniques to reduce the deviations from orthogonality obtained with the algebraic method.

The grid depicted in figure 2 was obtained with the orthogonal grid generator using the algebraic grid as the initial guess and fixed grid nodes at the first $\eta$ line close to the tip. In this grid, the maximum deviation from orthogonality is 26.4° degrees and the mean deviation from orthogonality is 1.5°, which is significantly smaller than in the algebraic grid.

**KVLCC2 Tanker**

The KVLCC2 tanker is one of the test cases of the latest Workshop in Numerical Ship Hydrodynamics held in 2000, [9]. Of course a variety of volume grids for a viscous flow calculation can be conceived for such a case. Here we consider a two-block grid with the H-O topology typically applied in MARIN’s flow solver PARNASSOS, [10].

In PARNASSOS, the flow in the near-wall region is solved without the use of wall functions. Therefore, at large Reynolds numbers extremely small grid line spacings are needed in the direction normal to the wall. The direct generation of such a grid with the Grape approach would lead most certainly to a nightmare in the convergence of the Non-linear terms of the ‘control functions’. Therefore, we have adopted a different procedure to obtain the required grid. A basis grid is generated with a reasonable stretching in the normal direction, $j$, and the desired grid node distribution in the streamwise, $i$, and girthwise, $k$ directions. The grid nodes along the normal direction are then redistributed with a 1-D coordinate transformation applied to all the $j$ lines. The final grid retains the deviations from orthogonality of the basis grid and satisfies the desired grid node density in the near-wall region. The generation of the basis grid is performed for two sub-domains: the bow and the stern regions. Each sub-domain has six boundaries: the inlet and outlet planes,
the ship surface, the symmetry plane of the ship, the free surface and the external boundary. The inlet, outlet, symmetry plane and free surface are 2-D planes and the ship surface and external boundaries are curved surfaces.

In each sub-domain, a $121 \times 51 \times 41$ grid is generated with the following procedure: The first boundary grid to be generated is on the ship surface, where we applied the orthogonal grid generator in the stern region and the algebraic method in the bow region. Next, the grids on the four plane domain faces are all generated as follows. The face edges on the symmetry plane and free surface are straight lines were a typical boundary-layer grid node distribution is specified. At the edge belonging to the external surface, which in the present case is an elliptical cylinder, the grid nodes are initially placed in equidistant locations. A preliminary grid is then generated with the 2-D orthogonal grid generator allowing the grid nodes to move along the external boundary. The final grids are obtained using the preliminary grid as the initial guess for the 2-D Grape approach, which is applied with fixed nodes on all the boundaries. The maximum and mean deviations, $\theta_M$ and $\theta_m$, of these boundary grids are given in table 1. The maximum and mean deviations from orthogonality are small in all cases.

<table>
<thead>
<tr>
<th>Boundary Grid</th>
<th>Bow $\theta_m$</th>
<th>Bow $\theta_M$</th>
<th>Stern $\theta_m$</th>
<th>Stern $\theta_M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inlet</td>
<td>0.1</td>
<td>8.0</td>
<td>4.5</td>
<td>25.1</td>
</tr>
<tr>
<td>Outlet</td>
<td>4.5</td>
<td>25.1</td>
<td>0.6</td>
<td>18.5</td>
</tr>
<tr>
<td>Ship surface</td>
<td>4.0</td>
<td>30.4</td>
<td>1.4</td>
<td>33.4</td>
</tr>
<tr>
<td>Free surface</td>
<td>3.6</td>
<td>37.0</td>
<td>2.8</td>
<td>30.7</td>
</tr>
<tr>
<td>Symmetry plane</td>
<td>0.8</td>
<td>13.9</td>
<td>1.6</td>
<td>29.5</td>
</tr>
</tbody>
</table>

Table 1: Maximum and mean deviations from orthogonality of the grid on the boundaries of the two sub-domains of the basis grid around the KVLCC2 tanker.

At the external boundary, an initial guess of the boundary grid is obtained with a simple algebraic interpolation on the predefined boundary shape, using the known grid node distributions at the four edges. Subsequently, the 3-D orthogonal grid generator is applied with moving nodes on the external boundary to produce the final result.

The boundary grids obtained as described above are illustrated in figure 3 for the two sub-domains. To improve the graphic quality only every second grid line is plotted. Figure 3 presents also several views of the most difficult regions of the boundary grids (all grid lines displayed).

Finally the interior grid is generated with the 3-D Grape approach using fixed grid node coordinates on the boundary faces. The maximum and mean deviations from

---

$^4$In the present application gravity waves are neglected and so the free surface is a flat surface.
orthogonality of the basis grid are given in table 2.

The largest deviations from orthogonality occur in the $i = constant$ surfaces, away from the ship surface. Figure 4 presents four views of the grid in the regions with the largest deviations from orthogonality. The plots show that the near-wall grid has the desired properties: orthogonality at the boundary and smooth grid line distances.

By applying 1-D stretching to this basis grid, we have generated two grids for the

<table>
<thead>
<tr>
<th>Region</th>
<th>$(\theta_{mi}^{ij})^o$</th>
<th>$(\theta_{mi}^{ij})^o$</th>
<th>$(\theta_{mi}^{ik})^o$</th>
<th>$(\theta_{mi}^{jk})^o$</th>
<th>$(\theta_{M}^{ik})^o$</th>
<th>$(\theta_{M}^{jk})^o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bow</td>
<td>2.9</td>
<td>42.0</td>
<td>3.9</td>
<td>39.3</td>
<td>8.5</td>
<td>51.7</td>
</tr>
<tr>
<td>Stern</td>
<td>2.5</td>
<td>32.3</td>
<td>2.2</td>
<td>48.1</td>
<td>6.8</td>
<td>45.7</td>
</tr>
</tbody>
</table>

Table 2: Maximum and mean deviations from orthogonality of the grid on the boundaries of the two sub-domains of the basis grid around the KVLCC2 tanker.
calculation of the flow around the KVLCC2 tanker at model scale, $R_n = 4.6 \times 10^6$, and full scale, $R_n = 2.03 \times 10^9$, respectively, where $R_n$ is the Reynolds number. The model scale flow is calculated in a $321 \times 81 \times 41$ grid and for full scale $R_n$ the grid has $321 \times 121 \times 41$ grid nodes. The stretching applied in the normal direction ensures that the maximum $y^+$ of the first grid node away from the ship surface in the two calculations is approximately 0.5. In both cases, the grid includes 40 sections upstream of the ship and in the wake which are obtained by translating the inlet and outlet planes of the basis grid along the $x$ direction.

The flow is simulated with PARNASSOS, [10], which solves the RANS equations using an iterative streamwise marching procedure. The present calculations were performed with Menter’s one-equation turbulence model, proposed in [11].

Figure 5 shows the convergence histories of the two calculations, here represented by the maximum norm of the changes between successive iterations of the five dependent variables. The plots demonstrate that in the present grids it is possible to converge the flow field, including the turbulent quantities, to machine accuracy at
both Reynolds numbers. This is an important property, if only to allow the proper execution of Verification studies, [12], which are becoming mandatory nowadays.

Conclusions

This paper presents the application of 2-D, surface and 3-D tools for structured grid generation to standard test cases of Numerical Ship Hydrodynamics. The methods developed are based on algebraic techniques and systems of partial differential equations, including 2-D and 3-D versions of the Grape approach, techniques for the generation of orthogonal grids and surface grids with control of the grid line stretching.

The examples included in this paper show that the efficient use of the different techniques available leads to surface and volume grids which satisfy the exigent requirements of Boundary Element and RANS methods in complex geometries. The present exercise suggests that the ‘optimal’ grid may be obtained with a blending of different grid generation techniques.

References


