Boundary Elements Method for Three-dimensional Potential Flow based on Unstructured Meshes

João Filipe Beja Pedro

Abstract

A low-order panel method is presented to calculate the perturbation potential of the flow around a ellipsoid using unstructured meshes is presented. The work is divided in three parts. The first consists in the code verification. This is done using an completely analytical unstructured mesh. When verifying the code the following geometric variables were evaluated: surface area, the distance between the collocation points of the panels and the corresponding points on the real ellipsoid surface and the normal vector to the panels surface. Proceeding with the code verification is then evaluated the potential for disturbance. After checking the code, the numerical solution for the potential is evaluated, where it is examined whether there are oscillations in the tip region of the ellipsoid. Finally, studies of mesh refinement were conducted for meshes without geometric similarity, where different relationships for $\textit{h}_i$ were tested in order to determine whether it is possible to extrapolate a numerical solution.

Key-words


Introduction

One of the main aspects of BEM is that, typically, it uses structured grids for the geometry discretization. This kind of grids is normally composed by stripes of elements that are the result of two families of lines that are defined by two perpendicular directions. For this reason it is common the use of quadrilateral elements in BEM.

This work will be based in the previous studies done at MARETEC and therefore will follow the Morino formulation (Morino et al. [1975]). The Morino formulation uses sources and doublets as panel singularities and it is the base for the code produced at MARETEC called Propan, that will be used in this work.

The conventional grids, for zones of the propeller blade with no "chord", like the tip region, generate ill-defined base vectors, and as consequence some oscillations of the potential solution were observed in propeller blades by Kinnas et al. [1994] and Baltazar [2008].

Some solutions were tried, namely, Falcão de Campos et al. [2006] the use of orthogonal grids on a ellipsoid that simulates the geometry of a propeller blade but has an analytical solution. Then Baltazar [2005] tried to implement hyperboloidal panels but the problems for the tip flow still persist.

Other solution tried, was the use of flow aligned grids in Pyo [1995] but it's not clear if a reliable pressure distribution is obtained in the tip region.

All the above solutions were done in structured grids, therefore it is proposed in this work to abandon the structured grids and explore the flexibility of the unstructured grids widely used in the finite element method. The geometry used in this study is the ellipsoid used by Falcão de Campos et al. [2006] and Baltazar [2005].

This work is divided in three parts. First a code verification is performed as described in Roache [1998] and it requires an error evaluation including the estimation of the order of convergence. For the convergence order determination grid refinement studies are required and therefore the ellipsoid parametrization described in Cruz and Falcão de Campos [2004] was used to generate geometrically similar grids.

After the code verification the potential solution near the tip region of the ellipsoid, obtained with this analytical grid, was checked. The objective was to verify if the oscillations present at Falcão de Campos et al. [2006] in tip of the ellipsoid still persist in the solution for the unstructured triangular meshes.

In general for real geometries as propeller blades in most cases it is not possible to define a parametrization, so a general mesh generator was implemented in order to discretize these domains. The method chosen was the Delaunay triangulation, but this method has the disadvantage of generate meshes that aren’t geometrically similar.

There is a wide discussion in how to perform refinement studies of these meshes, the problem is the definition of the typical element dimension used in the error equation, hence in these work it is tested a definition for the typical element dimension based in the norms of the elements area.

Mathematical Formulation

The boundary element method consists in solving the potential flow problem described by equation 1 and applying it to small boundary elements that discretize the domain of the problem described in this text as $S_B$. Then the general solution for $\phi(P)$ is given by the superposition of all the solutions obtained for each element. The discrete form of 1 is given by equation 2.
\[
2\pi \phi(P) - \iint_{S_n} \phi(Q) \frac{\partial}{\partial n_Q} \left( \frac{1}{R(P, Q)} \right) dS = 
\iint_{S_n} \left( \hat{n} \cdot \vec{U}_\infty \right) \frac{1}{R(P, Q)} dS \quad P \in S, \tag{1}
\]

\[
\frac{\phi(P)}{2} = - \sum_{i=1}^{n} \left[ \frac{1}{4\pi} \iint_{\Delta S_i} \mu(Q) \frac{\partial}{\partial n_Q} \left( \frac{1}{R(P, Q)} \right) dS \right] 
- \sum_{i=1}^{n} \left[ \frac{1}{4\pi} \iint_{\Delta S_i} \sigma(Q) \left( \frac{1}{R(P, Q)} \right) dS \right], \quad \tag{2}
\]

where \( R(P, Q) \) represents the distance between \( P \) and \( Q \), \( \mu = -\phi(Q) \) which is the doublet strength and \( \sigma = \hat{n} \cdot \vec{U}_\infty \) is the source strength and \( \phi(P) \) represents the perturbation potential at point \( P \).

In this work the domain will be discretized by linear triangular elements. The triangular master element has the following shape functions:

\[
\begin{align*}
\hat{\psi}_1 &= \xi \\
\hat{\psi}_2 &= \eta \\
\hat{\psi}_3 &= 1 - \xi - \eta
\end{align*} \tag{3}
\]

In this case for integration purposes the edge of the element has an length of 1. More details can be read in reference Reddy [2005].

\[
\vec{Q}(\xi, \eta) = \sum_{i=1}^{N} \vec{Q}_i \hat{\psi}_i. \tag{4}
\]

The tangent vectors along the any arbitrary direction \( \gamma \) can be obtained in equation 5.

\[
\vec{a}_\gamma = \frac{\partial \vec{Q}}{\partial \gamma}. \tag{5}
\]

Traditionally, in the boundary element method the tangent vectors are defined along two perpendicular directions, that usually define the stripes of elements seen in the panel method structured grids. These directions are represented in figure 2 by \( \xi \) and \( \eta \).

\[
Figure 1: Triangular master elements
\]

The transformation from the master element domain(\( \xi, \eta \)) is given by equation 4 and is illustrated in the figure 2.

\[
\vec{Q}(\xi, \eta) = \sum_{i=1}^{N} \vec{Q}_i \hat{\psi}_i. \tag{4}
\]

The tangent vectors along the any arbitrary direction \( \gamma \) can be obtained in equation 5.

\[
\vec{a}_\gamma = \frac{\partial \vec{Q}}{\partial \gamma}. \tag{5}
\]

Traditionally, in the boundary element method the tangent vectors are defined along two perpendicular directions, that usually define the stripes of elements seen in the panel method structured grids. These directions are represented in figure 2 by \( \xi \) and \( \eta \).

\[
\begin{align*}
\vec{a}_1 &= \frac{\partial \vec{Q}}{\partial \xi} \\
\vec{a}_2 &= \frac{\partial \vec{Q}}{\partial \eta}
\end{align*} \tag{6}
\]

The normal and tangent vectors are obtained in the same way for both types of elements.

To integrate equation 2 the source and dipole strengths must also be discretized. In this work this discretization is in accordance with the lower-order method. This method expands the dipole and source strengths in the form of a truncated Taylor series, using the collocation point as the expansion point, but only the first term of the expansion is used. The expansion yields:

\[
\begin{align*}
\mu_0 &= \mu(0, 0), \\
\sigma_0 &= \sigma(0, 0),
\end{align*} \tag{7}
\]

hence, along one panel the source and dipole distributions are constant. Based on the previous assumption, one can derive the following integrals:

\[
\begin{align*}
P_d &= \iint_{\Delta S} \frac{\hat{n} \cdot \vec{R}}{|\vec{R}|^3} dS, \\
P_s &= \iint_{\Delta S} \frac{1}{|\vec{R}|} dS,
\end{align*} \tag{8}
\]

where \( P_d \) and \( P_s \) are called influence coefficients and are determined from the computation of the previous expressions. In this case the integrals can be analytically determined, the formulation used was developed by Morino in 1975(Morino et al. [1975]).

The Morino formulation calculates the integrals by adding the contribution of each panel corner to the solid angle of panel subtended from the field point \( P \) for triangular elements it takes the form:

\[
\phi_D(P) = \beta_1 - \beta_2 + \beta_3, \tag{9}
\]

where
where I is the point calculated by equation 12:

\[ \beta_i = \arctan \left( \frac{\vec{R}_i \times \vec{d}_{\xi_i}}{|\vec{R}_i| \cdot |\vec{d}_{\xi_i}|} \right), \quad (10) \]

and where \( i = 1, 2, 3 \) and \( \vec{R}_i = \vec{R}(Q_i, P) \) is the vector from the point \( P \) to the corner of the panel \( Q_i \), and the vectors \( \vec{d}_{\xi_i} \) and \( \vec{d}_{\eta_i} \) are the tangent vectors in 11.

\[ \begin{align*}
\vec{d}_{\xi_1} &= \vec{d}_{\xi_2} = \vec{Q}_2 - \vec{Q}_1, \\
\vec{d}_{\xi_3} &= \vec{d}_{\eta_3} = \vec{Q}_3 - \vec{Q}_2, \\
\vec{d}_{\eta_2} &= \vec{d}_{\eta_3} = \vec{Q}_3 - \vec{Q}_2. 
\end{align*} \quad (11) \]

In the other hand the source influence coefficient is calculated by equation 12:

\[ \phi_s(P) = I_i^{(S)} - I_2^{(S)} + I_3^{(S)} + \left( \vec{R} \cdot \vec{n} \right)_0 \phi_D(P) \quad (12) \]

where \( I_i^{(S)} \) is defined as:

\[ \begin{align*}
I_i^{(S)} &= - \frac{\left( \vec{R}_i \times \vec{d}_{\xi_i} \right) \cdot \vec{n}_i}{|\vec{d}_{\xi_i}|} \sinh^{-1} \left( \frac{\vec{R}_i \times \vec{d}_{\xi_i}}{|\vec{R}_i| \times |\vec{d}_{\xi_i}|} \right) \\
+ \frac{(\vec{R}_i \times \vec{d}_{\eta_i}) \cdot \vec{n}_i}{|\vec{d}_{\eta_i}|} \sinh^{-1} \left( \frac{\vec{R}_i \times \vec{d}_{\eta_i}}{|\vec{R}_i| \times |\vec{d}_{\eta_i}|} \right). 
\end{align*} \quad (13) \]

In the previous equations \( (\vec{R} \cdot \vec{n})_0 \) is evaluated at the center of the panel, and \( \vec{n}_i \) is the normal vector to the panel surface at each corner. Because panels in this study are flat triangles, the unit normal vector can be evaluated at the center once it remains constant through out all panel.

The previous formulation, for points that are located far from the panel on analysis, becomes inefficient and a numerical integration method should be used. For that purpose, 3 zones were defined as function of the dimensionless distance between the point \( P \), where the influence is being computed, and the panel centroid. This definition was based on previous studies done in reference Baltazar [2005]. They are:

First zone \( \Rightarrow X/D \leq 3.0 \)
Second zone \( \Rightarrow 3.0 \leq X/D < 6.5 \)
Third zone \( \Rightarrow X/D \geq 6.5 \) \quad (14)

where \( D \) is the largest side of the triangular panel, and \( X \) is the distance between \( P \) and the panel centroid.

In the first zone the analytical Morino formulation (Morino et al. [1975]) is used. In the second zone a 4 point Gauss-Legendre integration rule is applied, and finally for the third zone the integration method is the 1 point Gauss-Legendre rule. For details about this method the reference Pina [1995] may be consulted.

This distinction in zones provides a computational advantage since the numerical integration is much faster than the analytical integration.

With the discretizations of the previous section equation 2 becomes:

\[ \sum_{j=1}^{N} (\delta_{ij} - D_{ij}) \phi_j = \sum_{j=1}^{N} S_{ij} \left( \pi \cdot \vec{U}^\infty \right)_j, \quad i = 1, 2, ..., N, \quad (15) \]

where \( \delta_{ij} \) is the Kronecker delta and \( D_{ij} \) and \( S_{ij} \) are the influence matrices and are defined by:

\[ \begin{align*}
D_{ij} &= \frac{1}{2\pi} \sum_{i=1}^{n} P_{di}, \\
S_{ij} &= \frac{1}{2\pi} \sum_{i=1}^{n} P_{ni}. 
\end{align*} \quad (16) \]

The expressions above are the in general form and \( n \) represents the order of the Taylor expansion previously mentioned. In the present case (LO method) \( n = 1 \).

So, equation 15 gives a simple system of algebraic equations that must be solved numerically. In this work the method used was already part of Propand and was the bi-conjugate gradient method.

To guarantee that the results obtained are reliable it is necessary to perform a code verification. As defined in the literature (Roache [1998]), verification is a purely mathematical exercise with the objective of showing that we are "solving the equations right".

Verification is divided in two parts: code verification and verification of the calculations.

Code verification is based in evaluations of the numeric error for which the analytical solution is known. The verification of the calculations is based in the evaluation of the uncertainty of the numeric solution of a problem without an analytical solution. Both are based on the observation of the numerical error.

It is possible to decompose the numeric error in three distinct parts. The first is the round-off error. This error comes from the finite precision of computers. And it may be consider negligible if double precision is used for a 32-bit machine. Because this error has its source in the finite representation of real numbers, it increases with the dimension of the system, i.e., it increases with the refinement of the mesh, and the maximum refinement of the mesh is determined by the machine precision.

The second part of the error is the iterative error. The source of this error is the iterative method used for the numerical solution of the system of equations defined in 15.

And finally, the third and the most important contribution for the numerical error, is the discretization error. This error comes from transformation (boundary elements, finite differences, finite-volume, finite-elements, etc.) of a continuous system into a discrete system.

The discretization error is represented through the expansion of a power series which only the first term is used:

\[ e(\phi) = \phi - \phi_{exact} = \alpha h_i^p. \quad (17) \]

Where \( \phi_{exact} \) represents the exact solution of the variable \( \phi \), \( \alpha \) is a constant, \( h_i \) is the typical dimension of the mesh and \( p \) is the order of convergence.

For the application of equation 17 it is necessary to define \( h_i \) which could be a difficult task for unstructured grids.
and where still exist a wide discussion about its definition for meshes without geometric similarity. It is necessary to consider:

1. If the results are in the asymptotic region of the convergence, i.e., the terms with higher order are neglected;

2. There are no guarantees that $p$ is constant through out all variables of the problem and for all regions of the domain.

Hence it is advisable that three or more meshes are used to assess the numerical error.

For the convergence studies, in literature typically, three norms of the error are used. These norms are presented in the following equations:

$$
\Lambda_{\infty} [e(g)] = \max (e(g), i)
$$

(18)

$$
\Lambda_{1} [e(g)] = \frac{\sum_{i=1}^{N} e(g,i)}{N}
$$

(19)

$$
\Lambda_{2} [e(g)] = \sqrt{\frac{\sum_{i=1}^{N} e(g)^{2}}{N}}
$$

(20)

To the error norms above applies the power series expansion (equation 17) and it yields:

$$
\Lambda_{n} [e(g)] = (\Lambda_{n} [e(g)])_{0} + \alpha h^{p},
$$

(21)

in the equation above the term $[\Lambda_{n} [e(g)])_{0}$ represents the error in the mesh with elements of size zero, and the other terms were previously explained.

The references Eça [2006], Eça [1996] and Eça and Hoekstra [2009] can be read for more details about the error analysis in computational fluid dynamics.

Mesh Generation

The aim of this work is to improve the potential flow solution near the tip region of the ellipsoid obtained with the BEM by introducing support for unstructured grids.

The advantage of these kind of grids comes from the higher flexibility to describe complex geometries. Hence, it was decided to tackle the problem in two phases. The first considers only the geometrical problem and the second handles the potential problem. Furthermore, because meshes representing real domains are typically difficult to refine with geometrical similarity, several refinement studies were done for these meshes to verify if the numerical solution can be extrapolated.

Any numerical method that tries to find the solution of differential equations needs a discrete domain. So, to represent in the code this discrete domain was decided that the connectivity matrix data structure should be applied. The connectivity matrix chosen can be represented by:

$$
\begin{bmatrix}
  t & n_1 & n_2 & \cdots & n_{t1} \\
  1 & t_1 & n_{11} & n_{12} & \cdots & n_{1t1} \\
  2 & t_2 & n_{21} & n_{22} & \cdots & n_{2t1} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  i & t_i & n_{i1} & n_{i2} & \cdots & n_{it1}
\end{bmatrix}
$$

Figure 3: Connectivity matrix model

The connectivity matrix also contains the order of the connection between the nodes. From the previous example, one can say that node $n_{11}$ is connected to node $n_{21}$ and so forth. The element is closed by the connection between the last node and the first one.

To complete the description of the geometry the coordinates of the nodes are also needed.

$$
\begin{bmatrix}
  k & x_k & y_k & z_k \\
  1 & x_1 & y_1 & z_1 \\
  2 & x_2 & y_2 & z_2 \\
  \vdots & \vdots & \vdots & \vdots \\
  k & x_k & y_k & z_k
\end{bmatrix}
$$

Figure 4: Matrix model for the nodes coordinates

In figure 4, $k$ represents the number of each grid node, and $x_k, y_k,$ and $z_k$ represent the coordinates of each node. With both matrices the domain geometry is now completely defined.

It was decided to generate two kind of meshes: first an analytical mesh was generated for easily obtain geometrically similar meshes required for the code verification, and then to represent real life geometries it was implemented an generic mesh generation method based on the Delaunay Triangulation.

Analytical Unstructured Mesh Generation

This discretization of the ellipsoid was generated using a parametrization of its surface that produces a completely orthogonal grid. This parametrization is fully documented in reference Cruz and Falcão de Campos [2004]. The use of a completely analytical grid is the fact that is easy to generate meshes with geometric similarity.

The equations used to generate these meshes in the representation of the surface in ellipsoidal coordinates:

$$
\begin{align*}
  x^{2}(\mu, \nu) &= \frac{\rho_{2}^{2}(\rho_{1}^{2} + \mu)(\rho_{1}^{2} + \nu)}{(\rho_{1}^{2} - \rho_{2}^{2})(\rho_{1}^{2} - 1)} \\
  y^{2}(\mu, \nu) &= \frac{(1 + \mu)(1 + \nu)}{(1 - \rho_{1}^{2})(1 - \rho_{2}^{2})} \\
  z^{2}(\mu, \nu) &= \frac{\rho_{2}^{2}(\rho_{1}^{2} + \mu)(\rho_{2}^{2} + \nu)}{(\rho_{2}^{2} - \rho_{1}^{2})(\rho_{2}^{2} - 1)}
\end{align*}
$$

(22)

where,
The properties of the meshes that were generated are presented in table 1.

<table>
<thead>
<tr>
<th>$N_\mu$</th>
<th>$N_\nu$</th>
<th>Number of elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>21</td>
<td>6400</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
<td>9216</td>
</tr>
<tr>
<td>29</td>
<td>29</td>
<td>12544</td>
</tr>
<tr>
<td>33</td>
<td>33</td>
<td>16384</td>
</tr>
<tr>
<td>37</td>
<td>37</td>
<td>20736</td>
</tr>
<tr>
<td>41</td>
<td>41</td>
<td>25600</td>
</tr>
<tr>
<td>49</td>
<td>49</td>
<td>36864</td>
</tr>
<tr>
<td>57</td>
<td>57</td>
<td>50176</td>
</tr>
<tr>
<td>65</td>
<td>65</td>
<td>65536</td>
</tr>
<tr>
<td>73</td>
<td>73</td>
<td>82944</td>
</tr>
<tr>
<td>81</td>
<td>81</td>
<td>102400</td>
</tr>
</tbody>
</table>

Table 1: Properties of the generated analytical meshes

Mesh Generation with Delaunay Triangulation

As one can easily imagine there are many shapes that can not be discretized by an analytical mesh, so an automatic mesh generator was implemented.

The mesh generation is divided in two distinct phases. The first is the generation of a set of points based on various criteria, to be explained later, and the second was the triangulation of that set, which is achieved using an algorithm called Delaunay triangulation.

The method used for the point generation is known in the literature (Georg and Borouchaki [1998]) as the classical method, or the method for creation along the edges. The method is characterized by the necessity of iterative interaction with the Delaunay kernel. The figure 6 describes the algorithm in general.

The point creation method divides the internal edges of the mesh $i$ to obtain the new points. This division is done according to a field function, which holds the desired element size. This function is a mathematical function which can represent physical properties of the problem such as the curvature of the domain or the pressure distribution of a flow. This function with the mesh $i$ form what is called in the literature by control space.

The other part of the mesh generation algorithm is the Delaunay triangulation. The Delaunay triangulation is an algorithm to construct triangles from a set of points. There are several methods to implement the Delaunay triangulation but in this study the method chosen was the reduced incremental method, which has in its core the expression called Delaunay kernel. This expression is given by:

$$
\tau_{i+1} = \tau_i - C_P + B_P,
$$

where $\tau_i$ represents the mesh at iteration $i$, $C_P$ represents what is called the cavity of the point $P$ and $B_P$ represents the ball of point $P$.

As seen before, the mesh is formed by a set of points and a connectivity matrix. The cavity of point $P$ is the set of elements that do not comply with the Voronoi diagram (the Voronoi Diagram are explained in George and Borouchaki [1998]) by the introduction of the point $P$, and, therefore, must be deleted from the connectivity matrix, and the ball of $P$ is the set of elements that are created by introducing $P$ to the mesh $\tau_i$. The result is the mesh $\tau_{i+1}$.

For the insertion of the point $P$ there are three possible situations: in the first $P$ is already inside the mesh $\tau_i$, and, therefore, $C_P$ and $B_P$ are not empty. Moreover $B_P$ can never be empty. The second possibility is that $P$ is outside the boundary of the mesh but is not far enough to $C_P$ to be an empty set. Finally, $P$ can be far away from the boundary of the mesh and then $C_P$ is an empty set.

It is not trivial to implement these three situations, so to simplify the algorithm the reduced method is introduced. The method starts to analyse the boundaries of the cloud of points and creates an artificial mesh that encloses all the domain guaranteeing that all points are inserted inside the mesh, reducing the triangulation to the first situation.

This artificial mesh is designated by $\tau_0$ and it is the first mesh in the triangulation. It can vary in number of elements and number of points, but the labelling of the artificial points that generate $\tau_0$ must be different from the
regular meshes because, in the end, it is necessary to remove all elements that have points from \( T_0 \).

The Delaunay triangulation is illustrated in the figure 7.

![Figure 7: Simplified representation of the triangulation algorithm](image)

The illustration of the algorithm for the generation of these meshes is in the figure 8.

Finally, because this method only generates a mesh in two dimensions it is necessary to project the mesh to the ellipsoid surface. This projection is done using the normal vector of the ellipsoid surface. The method is represented by the figure 9.

For the generated meshes the following field function was used in the control space:

\[
h_i = \frac{l_i}{2} ( -\frac{x_i^2}{2} + a ) + ( -\frac{y_i^2}{2} + b ),
\]

where \( l_i \) is the desired nominal size of the elements and \( a \) and \( b \) are parameters for controlling the stretching, i.e., the smaller its value the larger will be the stretching. To generate a family of grids, \( a \) and \( b \) must be constant.

The purpose behind the choice of this function for the control space is to try to describe as well as possible the large curvature that the ellipsoid presents at the edges. The average of the two quadratic functions is used to balance the effect of the two curvatures in the "chord-wise" direction and in the "span-wise" direction in the elements size. Finally, the coordinates \( x_i \) and \( y_i \) are both dimensionless to equalise the weight of the two curvatures in the elements size.

All generated meshes are given in tables 2 and 3.

The three first columns of both tables have the values of the parameters \( l_i \), \( a \) and \( b \) used in the respective control space. And the two last columns have the resulting number of elements and \( L_2 \) norm of elements area of the generated meshes.

**Results**

The main objective of this thesis is to verify if the use of unstructured grids in the Boundary Element Method improves the quality of the potential flow solution around the tip region of propeller blades. To accomplish this objective some modifications have been done to the MARETEC code, called Propan. Thus a careful code verification must be performed.

As previously said, code verification is achieved by doing convergence studies of the geometric properties and of the perturbation potential. The convergence studies for the verification purpose focus on the analytical grids. For triangular meshes it was studied the convergence of the unit normal vector, the surface area of the ellipsoid, the distance between the collocation point in the panel and the corresponding point in the surface of the ellipsoid, and the perturbation potential.

The collocation point of each element is located in its centroid, and to determine the corresponding point on the ellipsoid surface the normal vector of the element at the collocation point was used to create a straight line that intersects the ellipsoid at the desired point. This approximation introduces in the determination of the analytical normal an important perturbation which will be explained in detail further in the text.

The convergence study of the surface area was done di-
Table 2: Delaunay generated meshes without perturbations

<table>
<thead>
<tr>
<th>( l_i )</th>
<th>( a )</th>
<th>( b )</th>
<th>Number of elements</th>
<th>( L_2 ) Element area</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.12</td>
<td>1.1</td>
<td>1.3</td>
<td>7268</td>
<td>2.04E-003</td>
</tr>
<tr>
<td>0.11</td>
<td>1.1</td>
<td>1.3</td>
<td>8380</td>
<td>1.78E-003</td>
</tr>
<tr>
<td>0.10</td>
<td>1.1</td>
<td>1.3</td>
<td>10224</td>
<td>1.46E-003</td>
</tr>
<tr>
<td>0.09</td>
<td>1.1</td>
<td>1.3</td>
<td>12672</td>
<td>1.18E-003</td>
</tr>
<tr>
<td>0.08</td>
<td>1.1</td>
<td>1.3</td>
<td>15868</td>
<td>9.35E-004</td>
</tr>
<tr>
<td>0.07</td>
<td>1.1</td>
<td>1.3</td>
<td>20580</td>
<td>7.15E-004</td>
</tr>
<tr>
<td>0.06</td>
<td>1.1</td>
<td>1.3</td>
<td>28152</td>
<td>5.24E-004</td>
</tr>
</tbody>
</table>

Table 3: Delaunay generated meshes with perturbations

<table>
<thead>
<tr>
<th>( l_i )</th>
<th>( a )</th>
<th>( b )</th>
<th>Number of elements</th>
<th>( L_2 ) Element area</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.12</td>
<td>1.1</td>
<td>1.2</td>
<td>7984</td>
<td>1.90E-003</td>
</tr>
<tr>
<td>0.10</td>
<td>1.01</td>
<td>1.4</td>
<td>11388</td>
<td>1.33E-003</td>
</tr>
<tr>
<td>0.11</td>
<td>1.01</td>
<td>1.2</td>
<td>12020</td>
<td>1.29E-003</td>
</tr>
<tr>
<td>0.09</td>
<td>1.01</td>
<td>1.3</td>
<td>16132</td>
<td>9.55E-004</td>
</tr>
<tr>
<td>0.07</td>
<td>1.2</td>
<td>1.3</td>
<td>20564</td>
<td>7.19E-004</td>
</tr>
<tr>
<td>0.06</td>
<td>1.1</td>
<td>1.3</td>
<td>28152</td>
<td>5.24E-004</td>
</tr>
</tbody>
</table>

Because the analytical formula to obtain the value of the ellipsoid surface area is hard to implement, the error could not be evaluated and therefore code validation could not be completed. So it was decided to analyse the distance between the panels collocation points and the corresponding point on the real ellipsoid surface. The determination of this point was previously discussed in the present text.

The distance between these two points could be interpreted as the error between the positions of the panel collocation point and the real surface point. Therefore it is possible to extrapolate the error value for the mesh of size 0, i.e., for the continuous domain. So, with this variable is possible to accomplish a complete code verification.

For these refinement studies the \( \Lambda_2 \) norm was evaluated. The results are presented in the figure 11.

Figure 11: Convergence of the \( \Lambda_2 \) of distance between the collocation point and the corresponding point on the ellipsoid for the orthogonal grid

Again, the obtained order of convergence is in accordance with the expected value, and the extrapolated value for the mesh with size 0 is 0. Because the convergence curve yielded by the Richardson extrapolation in logarithmic scale is a straight line, this means that the error of the mesh with \( h_i = 0 \) is zero. This statement is demonstrated by applying a logarithm to both members of equation 21. The obtained relation only represents a straight line when the difference between the continuous system and the mesh size 0((\( \Lambda_{h_i} \) [\( e(q) \])\( _h \)) is 0).

Thus with the order of convergence with a value of 2 and the error of zero for the mesh size 0, for all norms of error, the code verification certifies that for the geometric approximations the code is correct.

Because for the potential determination the unit normal is used, this geometric property was also evaluated for convergence and error extrapolation. In this case only the \( \Lambda_2 \) were used. The obtained results for the \( z \) component of the normal vector are presented in figure 12. The results for the other components exhibit the same behaviour.

As it can be seen, the extrapolation of the error indicates a convergence to 0.

Despite the convergence to 0 the order of convergence was lower than would be expected. The explanation for this fact is related with the unbalanced nature, from the geometrical point of view, that the triangular elements hold in the unstructured analytical mesh.

By doing a triangulation of a cloud of points yielded by
a parametrization and a discretization that were optimized for the rectangular elements, the resulting elements tend to be rectangular triangles with one of the cathetus much larger than the other. This property acting with the effect of the great curvature of the surface introduces to the method used to determine the analytical normal vector of the ellipsoid an important perturbation. To better understand this effect the following figure is presented. Note that this effect is larger when the curvature is also larger.

As one can see in figure 13 if the element was a rectangular one(dashed vectors at point $C_q$) the numeric unit normal would be practicably coincident with the analytical normal vector. This fact, obviously, depends on the curvature of the surface, but typically this coincidence will occur for balanced elements. But, as said before, for the triangular elements, which, in this case are very unbalanced, the curvature effect is quite intense.

With the code verification done for the geometrical part of the problem, it remains to verify the code that computes the potential solution. Once again, these verification was done according to Eça [2006], and the analysed variables were the $\Lambda_\infty$, $\Lambda_1$ and $\Lambda_2$ error norms.

The obtained results for the $\Lambda_2$ norm are presented in figure 14.

For the perturbation potential convergence mixed results were obtained:

The statement done for the expected order of convergence of the geometric properties also applies to the potential. However in table 4 the results present a different order of convergence from two. This happens because the unit normal is used in the determination of the potential and propagates to the solution some of the effects of the curvature of the ellipsoid. This is particularly true for the $\Lambda_\infty$ norm because it corresponds to the maximum of the error which is a local variable and is located in the zone of the ellipsoid with larger curvature. The order of convergence recovers in the other error norms because by definition they are global variables and therefore the larger error at the borders of the ellipsoid is dispersed through out all domain. The recovery is minor in $\Lambda_2$ norm because it is based the root mean square that was engineered to reduce the influence of minor terms(normally "noise") in the solution. Once again because the curves resulting from the Richardson extrapolation are straight lines, the error norms converge to 0.

The convergence to 0 with an order that is approximately 2 shows that code produces correct computations and therefore the unstructured meshes does not influence results.

The main objective of this work is to verify if with the use of unstructured meshes in the panel method, the oscillations verified in Falcão de Campos et al. [2006] still persist in the solution.

This verification was done first in the analytical mesh and because in real domains it is often impossible to found an analytical discretization of the domain, it was decided to verify if in the meshes generated by the Delaunay triangulation the potential solution has any oscillation near the tip region.

The results for both meshes are presented in figures 15 and 16. As it can be observed in the figures above there is no oscillation in both meshes for the potential solution.

As expected, the potential is proportional to the stretching function, through a unstructured grid is possible to obtain a smooth potential solution.

As said before, for grids with no geometrical similarity it

---

**Table 4: Perturbation potential convergence results**

<table>
<thead>
<tr>
<th>Error Norm</th>
<th>Convergence Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Lambda_\infty$</td>
<td>$p = 1.5$</td>
</tr>
<tr>
<td>$\Lambda_1$</td>
<td>$p = 1.9$</td>
</tr>
<tr>
<td>$\Lambda_2$</td>
<td>$p = 1.8$</td>
</tr>
</tbody>
</table>

---

Figure 12: Convergence of $\Lambda_2$ of $UN_z$ for the orthogonal grid

Figure 13: Ellipsoid curvature effect demonstration

Figure 14: Convergence of the $\Lambda_2$ of the perturbation potential for the orthogonal grid
The definitions presented in equation 26 were tested:

\[
\begin{align*}
    h_i1 & = \frac{1}{\sqrt{N_e}} \\
    h_i2 & = \sqrt{\Lambda_1} \\
    h_i3 & = \sqrt{\Lambda_2}
\end{align*}
\]  

(26)

where \(N_e\) is the total number of elements, \(\Lambda_1\) is the elements area \(\Lambda_1\) norm and \(\Lambda_2\) is the elements area \(\Lambda_2\) norm.

The reason to the choice of these properties is the fact that they all can describe the mesh with geometric similarity in the same way, i.e., for meshes that are geometrically similar \(h_i1 = h_i2 = h_i3\).

This work tried to define several \(h_i\) based on different geometrical properties of the mesh such as the relations of the norms of the elements area. The different \(h_i\) obtained are presented in tables 5 and 6.

Table 5: Different mesh typical sizes for Delaunay meshes without perturbations

<table>
<thead>
<tr>
<th>Mesh i</th>
<th>(h_i) N.(^o) elements</th>
<th>(h_i) (\Lambda_\infty)</th>
<th>(h_i) (\Lambda_1)</th>
<th>(h_i) (\Lambda_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.06</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.07</td>
<td>1.17</td>
<td>1.16</td>
<td>1.17</td>
<td>1.17</td>
</tr>
<tr>
<td>0.08</td>
<td>1.33</td>
<td>1.76</td>
<td>1.33</td>
<td>1.34</td>
</tr>
<tr>
<td>0.09</td>
<td>1.49</td>
<td>1.38</td>
<td>1.49</td>
<td>1.50</td>
</tr>
<tr>
<td>0.10</td>
<td>1.66</td>
<td>1.58</td>
<td>1.66</td>
<td>1.68</td>
</tr>
<tr>
<td>0.11</td>
<td>1.83</td>
<td>2.25</td>
<td>1.83</td>
<td>1.84</td>
</tr>
<tr>
<td>0.12</td>
<td>1.97</td>
<td>2.18</td>
<td>1.97</td>
<td>1.97</td>
</tr>
</tbody>
</table>

Table 6: Different mesh typical sizes for Delaunay meshes with perturbations

<table>
<thead>
<tr>
<th>Mesh i</th>
<th>(h_i) N.(^o) elements</th>
<th>(h_i) (\Lambda_\infty)</th>
<th>(h_i) (\Lambda_1)</th>
<th>(h_i) (\Lambda_2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.06</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0.07</td>
<td>1.17</td>
<td>1.49</td>
<td>1.17</td>
<td>1.17</td>
</tr>
<tr>
<td>0.09</td>
<td>1.32</td>
<td>1.32</td>
<td>1.32</td>
<td>1.35</td>
</tr>
<tr>
<td>0.11</td>
<td>1.53</td>
<td>1.42</td>
<td>1.53</td>
<td>1.57</td>
</tr>
<tr>
<td>0.10</td>
<td>1.57</td>
<td>1.98</td>
<td>1.57</td>
<td>1.59</td>
</tr>
<tr>
<td>0.12</td>
<td>1.88</td>
<td>1.58</td>
<td>1.88</td>
<td>1.90</td>
</tr>
</tbody>
</table>

in the verification of the code of this work and the objective of refinement studies of geometric non-similar meshes is to find a way to better define \(h_i\). It is observed that only \(h_i\) based in the \(\Lambda_2\) norm of the elements area are slightly different from the typical \(h_i\). Therefore, it was decided to conduct the studies using \(h_i\) defined by the \(\Lambda_\infty\).

Furthermore, the other definition of \(h_i\) that differs from the typical one is based on the \(\Lambda_\infty\) of the element’s area. But as can be seen, the variation \(h_i\) are not monotonous, i.e., the \(h_i\) does not constantly increase with the decrease of the number of elements of the mesh. This fact is due to local nature of the \(\Lambda_\infty\) norm. Because this norm is defined the maximum value of the area is only represents one element, and because these meshes are randomly generated it is not guaranteed that the maximum and minimum values of the elements area respect the mesh refinement.

To see the influence of the numerical “noise” induced by the non-similarity of these meshes, that some perturbations should be introduced in the mesh creation process in the Delaunay triangulation. So, three kinds of meshes were compared, the analytical and geometrically similar and the Delaunay generated meshes with and without perturbations.

The results obtained for the refinement studies using the \(\Lambda_\infty\) are presented in figures 17 through 18.

From the results above one can conclude that numerical solutions can be extrapolated from refinement studies of meshes without geometric similarity, but some aspects have to be considered.

First, all the meshes have to belong to the same family of meshes, i.e., have to be generated with the same parameters, this is observed in the figures 17 and 18, where the convergence order obtained for the meshes without perturbations(same family of meshes) is higher than the order obtained for the meshes with perturbations.

Second, one must guarantee that the meshes are fine enough to assure that the error equation is represented in the asymptotic region of the convergence, because as seen
Distance \( L \) and Area \( \Lambda \) dimension is defined as the relation of extrapolated from mesh refinement studies if the grid typical can observe a higher order of convergence if more refined convergence order is still 1, this means that probably one \( \Lambda \) for the distance meshes approaches the theoretical convergence order of 2. For the distance \( \Lambda_2 \) norm one can observe that the global convergence order is still 1, this means that probably one can observe a higher order of convergence if more refined meshes were studied.

Despite the limitations, the numerical solution can be extrapolated from mesh refinement studies if the grid typical dimension is defined as the relation of \( \Lambda_2 \) norm of the elements area.

Conclusions

It is possible to observe some improvement in the potential solution by improving the geometry discretization of complex shapes with unstructured grids. This is well presented in the figures 15 and 16 were the oscillations near the tip region of the ellipsoid observed in Kinnas et al. [1994] and Falcão de Campos et al. [2006] disappear. But despite the success some precautions have to be taken when using the unstructured meshes.

The problem has to be well discretized and the precautions for the structured meshes, such as, a well defined stretching function and a refined enough mesh to guarantee that the solution is in the asymptotic region are crucial to the quality of the solution. The previous results show that when the problem is well represented the solutions have quality.

From the refinement studies of the meshes with no geometric similarity one can conclude that, again with some precautions, the numerical solution can be extrapolated. Despite the fact that the order of convergence has not been the theoretical one, the extrapolated values from the meshes without perturbations are good approximations of the values obtained with geometrically similar meshes.

The fact that the order of convergence could not be 2, shows that these refinement studies are not suited for verification of the code. The code verification still have to be done in the traditional way described in Eça [2006].

For future work it is recommended that the application of unstructured meshes to the hydrodynamic problem of the propeller continues to the application in lifting surfaces, with Kutta condition and vortex wake representations. It is possible that the representation of the trailing edge connection with the wake can be improved by the use of these kind of meshes.

For the refinement studies it is necessary to go deeper and try different kinds of meshes, and try meshes with greater refinement. Apply these studies to theoretical and real problems, such as, a discretization of a circle or the discretization of a propeller blade.

References


J. A. C. Falcão de Campos, P Ferreira de Sousa, and J Bosschers. A verification study on low-order three-dimensional potential-based panel codes. Computers &


