
Implementation of error indicators for automatic grid refinement in 3D

Erasmus Mundus Master of Science in Computational Mechanics
Master Thesis report

Student: Khalid Ait Said
Supervisor: Dr. Jeroen Wackers
Advisor: Dr. Michel Visonneau

Abstract

This work deals with the numerical treatment of solutions of the Reynolds-averaged Navier-Stokes equations computed by the ISIS-CFD flow solver. The objective is to analyze a local adaptive mesh refinement strategy to reach acceptable accuracy with the lowest human and computational costs.

To achieve this, the behavior of three a-posteriori error indicators is examined in order to derive effective criteria for controlling both accuracy and computational effort at once. The ability of the methods to predict the error correctly is analyzed and evaluated.

The three estimates are based on the Hessian matrix of field variables. Two of them are based on a scalar approach, while the last one is based on a tensor approach. The whole procedure is applied to steady turbulent flows over relatively complex geometries. The efficiency of these methods is evaluated by the features flow visualization, distribution and number of cells.

In the case of scalar criteria, the norm of the Hessian matrix was found to be better than the square root of the Hessian matrix norm. The tensorial criterion that has been tested performed even better than the two scalar ones.

Acknowledgement

First of all, I would like to express my deep gratitude to my beloved parents, their continuing support made this master thesis possible.

I would like to thank Prof. Michel Visonneau, for his wonderful course displayed for the Erasmus Mundus students in Computational Mechanics, but more important, for his kindness and his good listening.

In addition, many thanks for Dr. Jeroen Wackers, for directing this research and being very helpful in all the ways possible to make this task easy and help to complete this work. His attention as adviser and his amazing researcher spirit inspired and guided me throughout the way.

Of course, I would like to thank the CFD team for their continuous help. Dr. Patrick Queutey, Dr. Emmanuel Guilmineau, Dr. Kunihide Ohashi, Dr. Ganbo Deng and Dr. Alban Leroyer, for the vital help they provided. Also, special thanks to Prof. Jean Piquet for providing help and insight whenever asked.

Table of contents

Acknowledgement.....	3
CHAPTER 1	DTSEE method
1.1. Introduction	5
1.2. Error Estimation by DTSEE	7
1.3. Discussion about DTSEE	8
1.4. Hessian Matrix.....	11
CHAPTER 2	Implementation of the Hessian matrix
2.1. Adaptive grid refinement in ISIS-CFD.....	13
2.2. Method of implementation.....	15
2.3. Comparison	18
CHAPTER 3	Test of refinement criteria
3.1. Scalar criterion	24
3.2. Tensorial criterion	33
3.3. Cell generation behavior	39
CHAPTER 4	Conclusion
Reference	43

CHAPTER 1

DTSEE method

In this chapter, a simple introduction to the adaptive mesh refinement method is presented, along with the error indicators that are used to control this method. Then the DTSEE error estimate presented by Jasak [1] is discussed, an error in his work is highlighted.

1.1. Introduction

The method of adaptive mesh refinement employs successively the use of a flow solver and mesh generation tools. After a first run of the solver on an original grid, it determines the size of local mesh necessary anywhere in the field of calculation to ensure the desired accuracy. This information is exploited by the mesh to generate an adapted grid to be used for another simulation. Both steps are repeated until the final solution is reached.

In general, an adaptive mesh refinement strategy identifies the regions in the computation domain in which certain criteria are satisfied and then, increases the nodal density in those regions. Broadly speaking, most of these techniques can be described as feature based [7]. This is a standard approach that is employed in adaptive mesh refinement and is not discriminating in the sense that this strategy equally distributes the error across the mesh. Therefore the need of a measure of error is essential to the efficiency and success of such methods.

A numerical solution is obtained following a set of rules that provide a discrete description of the governing equations and the solution domain. Its accuracy is determined from the correspondence between the exact solution and its numerical approximation.

The judgement on the solution accuracy should therefore be done by comparing it with the exact solution, which is usually unavailable. Error estimation is therefore an important integral part of numerical solution procedures. Determining a reliable error measure, on the other hand is a separate and difficult issue. Finite elements techniques provide a direct mechanism for error estimation such residual type methods; Finite volume techniques do not.

Early efforts in the field of error detection have been directed towards problems with complex flows. The solution has some distinct features that need to be resolved accurately. It is known in advance that those features can be recognized by large gradients in flow variables [7]. Adaptive refinement is therefore directed towards the regions in which gradients are high. In the exact solution, discontinuities are infinitely thin, refinement is therefore stopped when the thickness of discontinuities in the numerical solution is considered to be small enough, or when the maximum number of computational points is reached.

Combinations of gradients of flow variables used to control the adaptive refinement procedure are called error indicators. An error indicator highlights the regions of the domain where a better resolution is needed. In general, it does not provide information about the absolute error level. Error indicators are cheap to compute and in many situations, can give reliable information about the solution.

The error indicator will be used to control an adaptive refinement procedure. It should therefore provide reliable information about the distribution of the error through the computational domain. Accurate error distribution is critical in calculations with highly localized refinement, where the successive levels of refinement are embedded into each other. First levels of refinement should be located accurately in order to reduce unnecessary overheads. Even if the error level is not estimated accurately, a correct error distribution guarantees that refinement regions are properly selected.

The study of Jasak [1] concerns the error estimation in the field of Computational Fluid Dynamics. The issue of a-posteriori error estimation is analyzed in general terms, and three approaches to error estimation are presented. Following each of these approaches, he proposed three new error estimates:

- The Direct Taylor Series Error estimate is based on the Taylor series truncation error analysis. Unlike the widely accepted Richardson extrapolation, this new error estimate can be used on a single-mesh solution. In spite of its asymptotic exactness, the accuracy of this error estimate is not considered satisfactory.
- The Moment Error estimate derives the solution error from the imbalance in higher moments of the variable. In order to provide the error magnitude, the imbalance is normalized in an appropriate way.
- The Residual Error estimate measures the error from the inconsistency between the prescribed variation of the function over the control volume and the face interpolation practice, enabling the estimation of the error in the convection-diffusion part of the discretization.

In the work of Hay [3], different a posteriori error estimation are examined in order to derive effective criteria for controlling both accuracy and computational effort at once for a wide range of problems.

The work carried by Ali [5] presents a simple and efficient way of implementing an error criterion. Based on the gradient of flow variables, it has been shown that the gradient of pressure performs better than velocity or vorticity gradient as error indicator, especially in problems with boundary layer computation.

The aim of this study is to consider the Hessian matrix and implement refinement criteria based on the second derivatives of field variables. Two types of criteria will be tested, a scalar and a tensorial refinement criterion.

In the first part of this work, we analyze the theoretical issues behind the DTSEE method which was presented by Jasak and further developed by Hay, and discuss about its implementation. The second part deals with the implementation of three refinement criteria (chapter 2), the tests are performed for the KVLCC2 case and the results are discussed (chapter 3).

1.2. Error Estimation by DTSEE

In order to simplify matters, we examine the problem of error estimation in steady-state calculations, where the whole error comes from the discretization of spatial terms.

The DTSEE method or the Direct Taylor Series Error estimate derives the error from the known order of accuracy and the local variation of the solution. It is based on the Taylor series truncation error analysis. As developed in the work of Jasak [1, 2], it is possible to obtain the Taylor series error estimate from a single-mesh result.

This error estimation method is based on the analysis of the numerical solution in terms of the Taylor series expansion. Every smooth function can be written as an expansion in its derivatives around a given point in space.

The discretization process can be considered as a truncation of the infinite series. The truncated form of the expansion at the computational point is used to describe the variation of the solution over the control volume surrounding it.

A p-th order accurate discretization method describes the local variation of a given function Φ with, ϕ the first p terms of the Taylor series:

$$\phi(\mathbf{x}) = \phi_P + (\mathbf{x} - \mathbf{x}_P) \cdot (\nabla \phi)_P + \dots + \frac{1}{(p-1)!} (\mathbf{x} - \mathbf{x}_P)^{p-1} \underbrace{\dots}_{p-1} \underbrace{(\nabla \nabla \dots \nabla \phi)_P}_{p-1}. \quad (1)$$

The discretization error e can also be expressed as an (infinite) series in higher derivatives of Φ :

$$e(\mathbf{x}) = \Phi(\mathbf{x}) - \phi(\mathbf{x}) = \sum_{n=p}^{\infty} \frac{1}{n!} (\mathbf{x} - \mathbf{x}_P)^n \underbrace{\dots}_n \underbrace{(\nabla \nabla \dots \nabla \phi)_P}_n.$$

If the mesh is too coarse, the contribution of higher-order terms can be significant, particularly if higher derivatives are large. In the case of second order accurate Finite Volume discretization, the prescribed spatial variation of ϕ over the control volume is linear:

$$\phi(\mathbf{x}) = \phi_P + (\mathbf{x} - \mathbf{x}_P) \cdot (\nabla \phi)_P.$$

The leading term of the error is:

$$e(\mathbf{x}) = \frac{1}{2} (\mathbf{x} - \mathbf{x}_P)^2 : (\nabla \nabla \phi)_P. \quad (2)$$

The Taylor series error estimate in the control volume surrounding the point P is then calculated as:

$$\begin{aligned} e_t(\phi) &= \frac{1}{V_P} \frac{1}{2} \left| \left[\int_{V_P} (\mathbf{x} - \mathbf{x}_P)^2 dV \right] : (\nabla \nabla \phi)_P \right| \\ &= \frac{1}{2V_P} |\mathbf{M} : (\nabla \nabla \phi)_P|. \end{aligned} \quad (3)$$

Where M is the second geometric moment of the control volume:

$$\mathbf{M} = \int_{V_P} (\mathbf{x} - \mathbf{x}_P)^2 dV. \quad (4)$$

The terms will be approximated from the available solution and mesh geometry. The error estimate obtained in such a way is called the Direct Taylor Series Error estimate. The task of creating a single-mesh Taylor series error estimate can be divided into two parts: estimating the second gradient of ϕ in P and evaluating the geometric moment tensor M for the control volume.

1.3. Discussion about DTSEE

Two errors in the DTSEE method are discussed; one is found earlier, the other one in this work.

Error equation

As pointed out by Dr. Wackers, the equation of the error is not exactly as presented in the work of Jasak [1] and Hay [3]. The fact is, even with a second order accurate discretization, the term Φ_P , which represents the approximated value of Φ at cell center, has no reason to be the exact solution, and therefore, this term will not disappear in the equation (2) stated above.

The equation becomes:

$$e(x) = \underbrace{(\Phi_{P_{exact}} - \phi_P)} + \frac{1}{2}(x - x_p)^2: (\nabla\nabla\phi)_P . \quad (5)$$

Hence, we have an additional term that cannot be evaluated directly, more discussion need to be considered for such a method. However, considering only the term containing the Hessian operator, it could be used as an indicator of the error, which is widely used in the computational field.

Geometric moment M

As defined previously in equation (4), the geometric moment over the control volume is written as

$$M = \int_{V_p} (\mathbf{x} - \mathbf{x}_p) \otimes (\mathbf{x} - \mathbf{x}_p) dV. \quad (6)$$

Calculation of M on an arbitrary unstructured mesh is more complex. The approach to be used here follows the work of Helf and Küster [9]. Using Gauss' theorem, the volume integral is reduced to a set of geometric moment integrals over the faces.

The origin considered here is in the centre of the control volume. We consider then the variable $\mathbf{X} = \mathbf{x} - \mathbf{x}_p$.

As developed in the work of Jasak [1], equation (6) can be written as follows:

$$\begin{aligned} M &= \int_{V_p} \mathbf{X} \otimes \mathbf{X} dV \\ &= \frac{1}{3} \int_{V_p} \nabla \cdot [\mathbf{X} \otimes (\mathbf{X} \otimes \mathbf{X})] dV \\ &= \frac{1}{3} \int_{\partial V_p} (d\mathbf{S} \cdot \mathbf{X})(\mathbf{X} \otimes \mathbf{X}) \\ &= \frac{1}{3} \sum_f \int_{S_f} (d\mathbf{S} \cdot \mathbf{X})(\mathbf{X} \otimes \mathbf{X}) \end{aligned}$$

The surface integrals can be further simplified if we consider that all faces are flat; for every point on the face, the dot-product $\langle d\mathbf{S}, \mathbf{x} \rangle$ then reduces to a constant. It follows that:

$$\int_{S_f} (d\mathbf{S} \cdot \mathbf{X})(\mathbf{X} \otimes \mathbf{X}) = \tilde{S} \cdot \mathbf{X}_f \int_{S_f} (\mathbf{X} \otimes \mathbf{X}) dA .$$

The volume integral is reduced to a sum of surface integrals over flat faces. Point x_f has been selected as a sample point; any other point from the face plane will give the same product.

$$M = \frac{1}{3} \sum_f \tilde{S} \cdot \mathbf{X}_f \int_{S_f} (\mathbf{X} \otimes \mathbf{X}) dA . \tag{7}$$

However, a simple test of this result will show that the development is not done rigorously, and the coefficient (marked in red above) is false, the first step needs more development and precision.

Example:

We consider the simplest geometric form possible: a hexahedral control volume aligned with the coordinate system (figure 1).

In this case, the analytical geometric moment is obtained in a straightforward manner:

$$\begin{aligned} M &= \int \mathbf{x} \otimes \mathbf{x} dv \\ &= \frac{V_P}{12} \begin{pmatrix} \Delta x^2 & 0 & 0 \\ 0 & \Delta y^2 & 0 \\ 0 & 0 & \Delta z^2 \end{pmatrix} . \end{aligned}$$

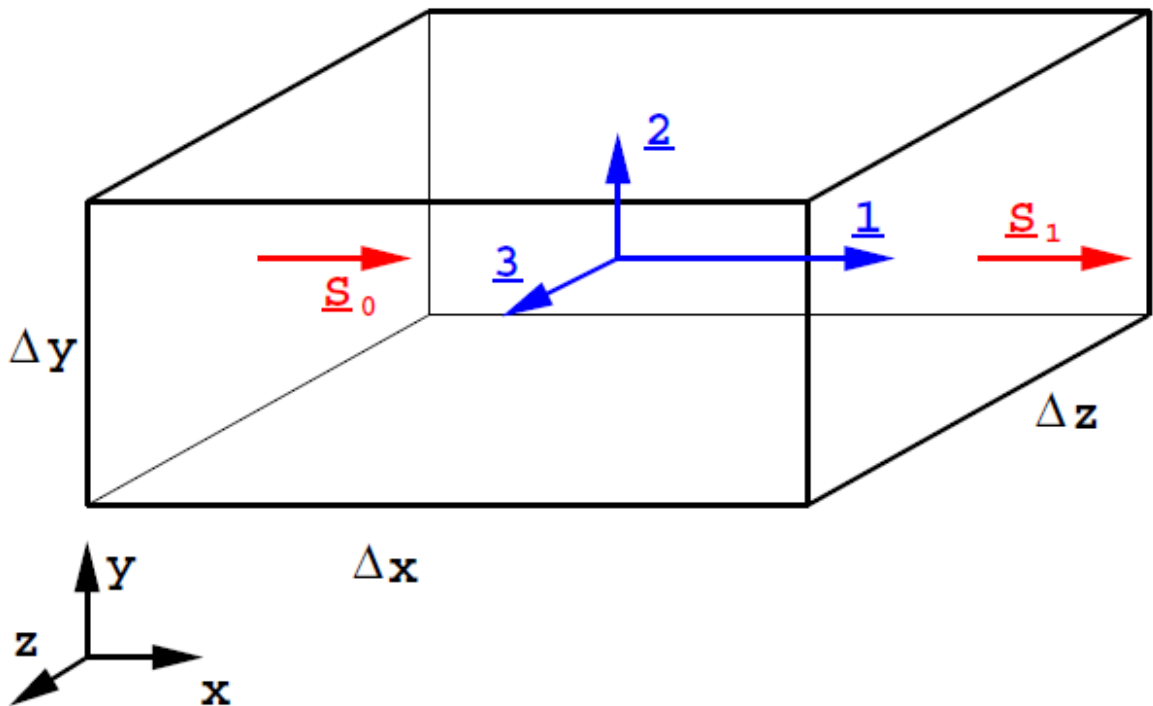


Figure 1: A hexahedral control volume; the coordinate system is at the centre of the control volume.

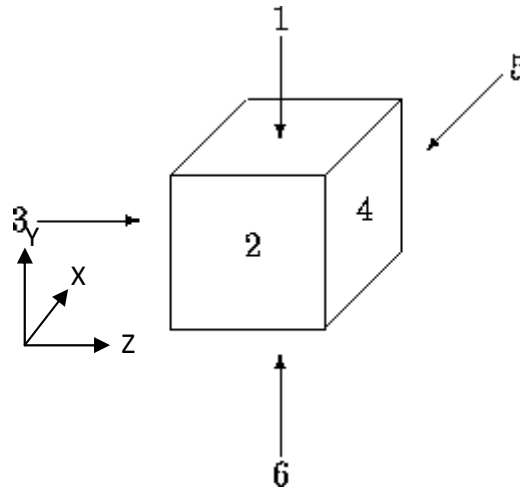


Figure 2: The numbering of faces for a control volume.

Now, let us use equation (7), we compute each term separately.

The numbering of the faces is as given in figure 2.

Let us start with M_{12} for example; the origin is the center of the control volume:

$$M_{12} = \frac{1}{3} \left[\sum_1^6 \left(\int xy \, dS (\mathbf{n} \cdot \mathbf{x}) \right) \right] = 0.$$

In fact, for all the integrals over faces, we end up with either $[cst1 * \int_{-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}} x \, dx]$, or $[cst2 * \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} y \, dy]$, or $[\int_{-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}} x \, dx * \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} y \, dy]$, which all give zero.

The same result is obtained for $M_{ij} = 0 \quad [\forall (i, j) \quad i \neq j]$ (8)

Now, let us focus on M_{11} :

$$\begin{aligned}
 M_{11} &= \frac{1}{3} \left[\sum_1^6 \left(\int x^2 \, dS (\mathbf{n} \cdot \mathbf{x}) \right) \right] \\
 M_{11} &= \frac{1}{3} \left[\int_1 x^2 \, dx \cdot dz (\mathbf{e}_y \cdot \mathbf{x}) + \int_6 x^2 \, dx dz (-\mathbf{e}_y \cdot \mathbf{x}) + \right. \\
 &\quad \left. \int_2 x^2 \, dy \cdot dz (-\mathbf{e}_x \cdot \mathbf{x}) + \int_5 x^2 \, dy \cdot dz (\mathbf{e}_x \cdot \mathbf{x}) + \right. \\
 &\quad \left. \int_3 x^2 \, dx \cdot dy (-\mathbf{e}_z \cdot \mathbf{x}) + \int_4 x^2 \, dx \cdot dy (\mathbf{e}_z \cdot \mathbf{x}) \right]
 \end{aligned}$$

$$M_{11} = \frac{1}{3} \left[\int_1 x^2 dx \cdot dz \left(\frac{\Delta y}{2} \right) + \int_6 x^2 dx \cdot dz \left(\frac{\Delta y}{2} \right) + \int_2 x^2 dy \cdot dz \left(\frac{\Delta x}{2} \right) + \int_5 x^2 dy \cdot dz \left(\frac{\Delta x}{2} \right) + \int_3 x^2 dx \cdot dy \left(\frac{\Delta z}{2} \right) + \int_4 x^2 dx \cdot dy \left(\frac{\Delta z}{2} \right) \right]$$

$$M_{11} = \frac{1}{3} \left[\int_{-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}} x^2 dx (\Delta z \Delta y) + \iint_{\frac{-\Delta y}{2} \frac{-\Delta z}{2}}^{\frac{\Delta y}{2} \frac{\Delta z}{2}} x^2 dy \cdot dz (\Delta x) + \int_{-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}} x^2 dx (\Delta y \Delta z) \right]$$

$$M_{11} = \frac{1}{3} \left[2 \frac{\Delta x^3}{3 * 8} (\Delta z \Delta y) + \frac{\Delta x^2}{4} (\Delta y \Delta z \Delta x) + 2 \frac{\Delta x^3}{3 * 8} (\Delta y \Delta z) \right]$$

$$M_{11} = \frac{1}{3} \Delta V \Delta x^2 \left[2 \frac{1}{3 * 8} + \frac{1}{4} + 2 \frac{1}{3 * 8} \right]$$

$$M_{11} = \frac{1}{3} \Delta V \Delta x^2 \left[\frac{5}{12} \right] \quad (9)$$

To recover the analytical solution, the coefficient in red in equation (9) should be $\frac{1}{5}$ and not as given $\frac{1}{3}$.

The problem is in fact coming from the following step: $\mathbf{x} \otimes \mathbf{x} = \frac{1}{3} \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x})$?

Let us develop in an indicial form the previous expression:

$$\begin{aligned} \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) &= (x_i * x_j * x_k)_{,k} \\ \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) &= (x_i * x_j) * x_{k,k} + (x_i * x_k) * x_{j,k} + (x_j * x_k) * x_{i,k} \\ \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) &= (x_i * x_j) * 3 + (x_i * x_j) + (x_j * x_i) \\ \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) &= 5 (x_i * x_j) \end{aligned}$$

Though, the right coefficient is:

$$\mathbf{x} \otimes \mathbf{x} = \frac{1}{5} \nabla \cdot (\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) \quad (10)$$

Injecting equation (10) into equation (7), the correct expression should be:

$$M = \frac{1}{5} \sum_f \tilde{S} \cdot \mathbf{X}_f \int_{S_f} (\mathbf{X} \otimes \mathbf{X}) dA \quad (11)$$

1.4 Hessian Matrix

Considering a field quantity ϕ , the aim is to compute the following matrix:

$$H = \begin{pmatrix} \phi_{,xx} & \phi_{,xy} & \phi_{,xz} \\ \phi_{,xy} & \phi_{,yy} & \phi_{,yz} \\ \phi_{,xz} & \phi_{,yz} & \phi_{,zz} \end{pmatrix} \quad (12)$$

Here the matrix H is considered symmetric, the assumption that $\phi_{,ij} = \phi_{,ji}$ is made. This condition requires a certain degree of the smoothness of the field variable considered, otherwise, the assumption is not valid anymore.

CHAPTER 2

Implementation of the Hessian matrix

In this chapter, the way ISIS-CFD deals with adaptive refinement is presented; the method of implementing the geometric moment is developed, and two ways of implementing the Hessian matrix are discussed.

2.1. Adaptive grid refinement in ISIS-CFD

The ISIS-CFD flow solver, available as a part of the FINETM/Marine computing suite, uses the incompressible unsteady Reynolds-averaged Navier Stokes equations (RANS).

The solver is based on the finite volume method to build the spatial discretization of the transport equations. The face-based method is generalized to two-dimensional, rotationally-symmetric, or three-dimensional unstructured meshes for which non overlapping control volumes are bounded by an arbitrary number of constitutive faces.

The velocity field is obtained from the momentum conservation equations and the pressure field is extracted from the mass conservation constraint, or continuity equation, transformed into a pressure-equation. In the case of turbulent flows, additional transport equations for modeled variables are solved in a form similar to the momentum equations and they can be discretized and solved using the same principles. Incompressible and non-miscible flow phases are modeled through the use of conservation equations for each volume fraction of phase. The whole discretization is fully implicit in space and time and is formally second order accurate. Several near-wall low-Reynolds number turbulence models, ranging from one-equation Spalart–Allmaras model, two-equation $k-\omega$ closures, to a full Reynolds stress transport $Rij-\omega$ model are implemented in the code.

With adaptive grid refinement, the solver works as follows: the flow solver is run on the initial grid for a limited number of time steps. Then the refinement process is called to adapt the grid. The refinement procedure has a specific refinement criterion; if the criterion, based on the current solution indicates that certain parts of the grids are not fine enough and need to be refined, the grid is refined and the solution of on the initial grid is copied to the new refined grid. The flow solver is run again and the refinement procedure is called and the refinement criteria decide what changes are to be made to the grid i.e. to refine or derefine the grid. The process is repeated until convergence is obtained (for steady flows).

In order to make the process more flexible the refinement procedure is divided into three distinct parts. Each part can be dealt with separately as these parts exchange minimal information between them. The parts are:

1) Refinement criterion: the refinement criterion or error indicator is an essential part of the adaptive process. The refinement criterion decides which parts of the grids are to be refined or de-refined based on a certain threshold value. The important point is that an error indicator can be

developed which may not depend on the type and orientation of the cells and thus avoiding the need for developing separate refinement criterion for different cell types.

2) Refinement decision: next step is the refinement decision based upon the refinement criterion in which a list of flags is created indicating which cell will be refined and in case of directional refinement, the direction is also specified. This decision depends upon the cell type but not on the way the refinement criterion was computed. It is simply an evaluation of the criterion field. The refinement decision has two steps. First, the refinement criterion is evaluated in each cell, based on a certain threshold value the refinement decision is taken. If the value of the refinement criteria exceeds the threshold value, the cell is refined; and if it is below (the threshold \div 2.5), the cell will be de-refined. Similar methodology applies to the directional refinement, if the refinement criterion exceeds the threshold in a certain direction, the cell will be refined in that direction and vice versa.

In the second step the decision in each cell is adapted to its neighbor cell. As a result, in an iterative procedure, refinement decisions are added and de-refinement decisions are removed. Completion of the refinement decisions is a great advantage before the start of the refinement.

3) Refinement: The final step is the actual refinement of the grid. First, all cells selected for de-refinement are de-refined, and then refinement of all cells to be refined is performed. During refinement, new small size cells are created, faces and nodes are added between them, and the cell family ties are adjusted; for de-refinement, small size cells are merged into their original large cells, unnecessary faces are removed, and the original family ties are restored. In parallel, once the refinement decisions are taken, the grid in each block can be refined without any communication with the other blocks. Both refinement and de-refinement are done cell by cell to ensure maximum generality and robustness of the code. After the treatment of each single cell, a correct grid with all its pointers is left, even if some pointers have to be changed again later when a neighbor cell is refined. In this way, a cell to be refined can treat all its faces and neighbors the same manner; no distinction is needed between cells that are already refined, cells that still have to be refined, and cells that are not refined at all. To further increase the generality of the code, the parts that refine cells and faces are completely decoupled.

For more details about mesh adaptation procedure in ISIS-CFD, as it is based strongly on the work of Hay, Wackers and Visonneau, [3, 4, 8], one can refer directly to their work.

2.2. Method of implementation

Geometric moment

As discussed in the previous chapter, the integration over the control volume will be transformed to an integration over the faces of the same control volume. First, the control volume is split to a sum of small control volumes, that are the cells; then for each cell, the divergence theorem is applied.

In ISIS-CFD, the unstructured mesh used is generated by HEXPRESS and is hexahedral. An analytical integration over an arbitrary domain, delimited by four points at least, which are not a priori in the same plane, is though impossible. Therefore, an approximation is needed.

Another subdivision of the domain is considered. Every face is cut to a set of triangles based on the centre of the face, and the integration is performed over each triangle (figure 3).

The integration over a triangle is widely used in the computational community; many methods are available in the literature [6]. For simplicity, this integration is performed by means of a mapping to the unit triangle (figure 4), then using Gauss Quadrature formulas to get the result.

The Gauss Quadrature method used is based on 3 points, which means that the method is exact for 2nd order polynomials, ie the geometric moment is integrated exactly on each triangle.

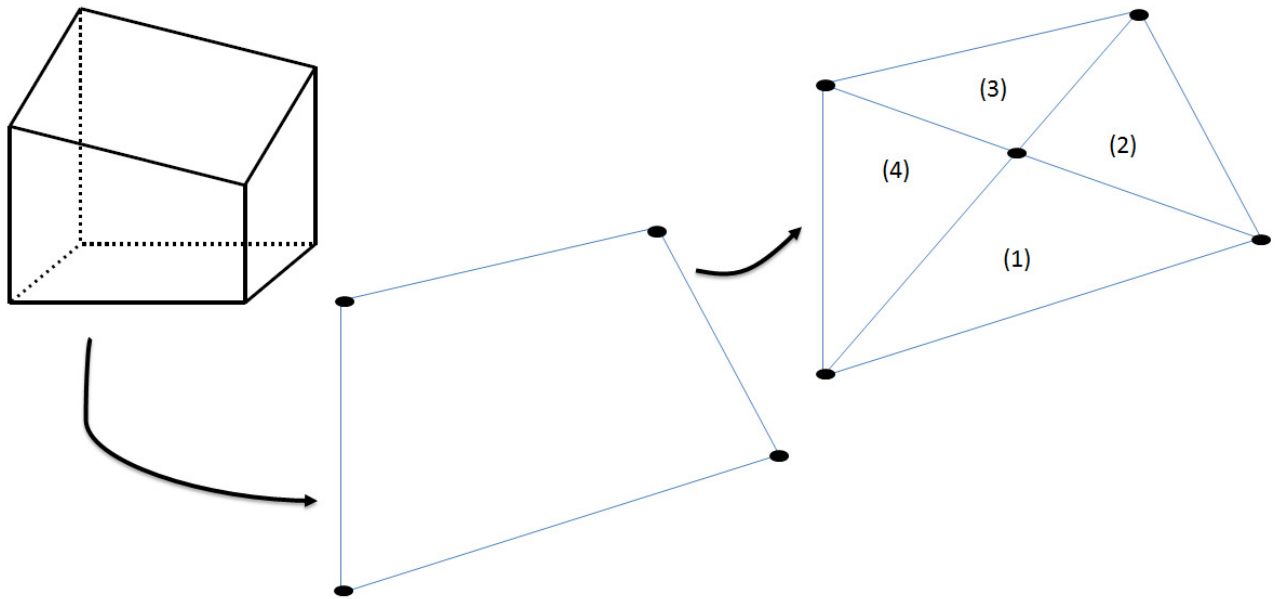


Figure 3: The general method followed to divide each set of faces into triangles.

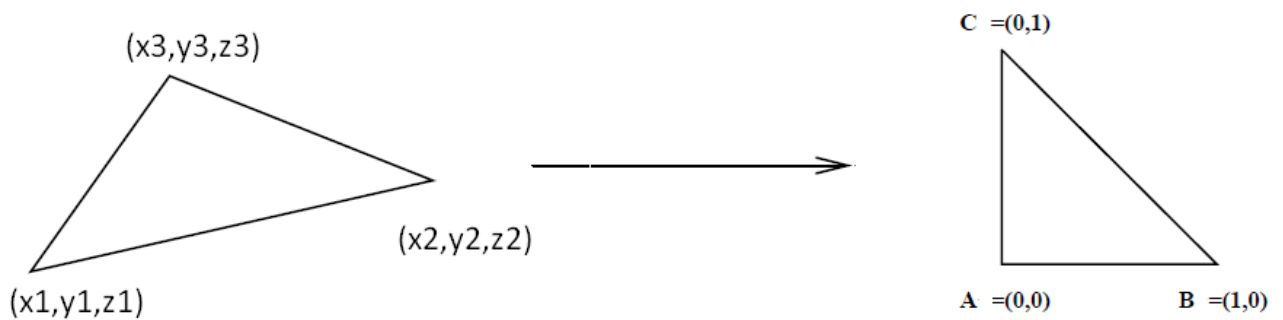


Figure 4: Mapping of a given triangle to the unit triangle used for the Gauss Quadrature rule.

Hessian matrix

The computation of an approximation to the Hessian matrix H (equation (12)) is based on the least squares method. It is computed in the centre of the cell, and the computation involves not only the neighboring cells, but also the neighbors of the neighbors.

The field chosen for this computation is the pressure, because of the performance observed of the gradient of pressure as error indicator, especially with boundary layer computations [5]. As the velocity and vorticity gradients are high in the boundary layer region, boundary layer refinement results in an unnecessary and costly refinement there and little refinement in the other regions. The pressure gradient criterion avoids such refinement in the boundary layer because the pressure varies little in that zone in the normal direction.

With the assumptions already made on the Hessian matrix, we have only 6 unknowns to be determined, while the standard number of neighbor cells and neighbor of neighbors are 24 at least.

Taylor series development of the pressure at the centre of the cell c up to the second order gives:

$$p(x) = \beta_1 + \beta_2(x - x_c) + \beta_3(y - y_c) + \beta_4(z - z_c) + \beta_5(x - x_c)^2 + \beta_6(y - y_c)^2 + \beta_7(z - z_c)^2 + \beta_8(x - x_c)(y - y_c) + \beta_9(x - x_c)(z - z_c) + \beta_{10}(y - y_c)(z - z_c). \quad (13)$$

Then, for each neighbor i , $i \in [1, m]$, where m is the number of neighbors, we have the following equation:

$$\beta_1 + \beta_2(x_i - x_c) + \beta_3(y_i - y_c) + \beta_4(z_i - z_c) + \beta_5(x_i - x_c)^2 + \beta_6(y_i - y_c)^2 + \beta_7(z_i - z_c)^2 + \beta_8(x_i - x_c)(y_i - y_c) + \beta_9(x_i - x_c)(z_i - z_c) + \beta_{10}(y_i - y_c)(z_i - z_c) = p_i. \quad (14)$$

We end up with an over-determined system of m linear equations in n unknown coefficients, n being 10 here, $\beta_1, \beta_2, \dots, \beta_n$, with $m > n$, written in matrix form as

$$\mathbf{X}\boldsymbol{\beta} = \mathbf{y},$$

where

$$\mathbf{X} = \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{21} & X_{22} & \cdots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mn} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}.$$

The coefficients of the Hessian matrix are extracted from β_i , $i \in [5, 10]$, while the other coefficients represent the gradient of pressure and are not used.

The goal is then to find the coefficients β which fit the equations "best", in the sense of solving the quadratic minimization problem [11] of the form:

$$\arg \min_{\beta} \sum_{i=1}^m \left| y_i - \sum_{j=1}^n X_{ij} \beta_j \right|^2 = \arg \min_{\beta} \| \mathbf{y} - \mathbf{X} \beta \|^2.$$

This minimization problem has a unique solution $\tilde{\beta}$ (provided that the n columns of the matrix \mathbf{X} are linearly independent), given by solving the normal equations

$$(\mathbf{X}^T \mathbf{X}) \hat{\beta} = \mathbf{X}^T \mathbf{y}. \quad (15)$$

The equations may also be weighted, based on the distance of the neighbors to the cell. In this case, one can minimize the weighted sum of squares:

$$\arg \min_{\beta} \sum_{i=1}^m w_i \left| y_i - \sum_{j=1}^n X_{ij} \beta_j \right|^2 = \arg \min_{\beta} \| W^{1/2} (\mathbf{y} - \mathbf{X} \beta) \|^2,$$

where $w_i > 0$ is the weight of the i th equation, and W is the diagonal matrix of such weights.

The normal equations are then:

$$(\mathbf{X}^T \mathbf{W} \mathbf{X}) \hat{\beta} = \mathbf{X}^T \mathbf{W} \mathbf{y}. \quad (16)$$

The weights used to solve the system are chosen to be the inverse of the distance between the considered cell (c) and the neighbors (i), such that: $w_i = 1/(d(\vec{x}_i - \vec{x}_c))^2$; considering such weights, it is reasonable to exclude the equation of the cell itself from the system of equations, and consider only its neighbors.

The Hessian matrix obtained will be represented by $H(P)_{10}$, where P is for pressure, and 10 for the number of variables used in the computation.

In order to gain in efficiency and computational cost, we can reduce the number of unknowns by using the gradient of pressure already computed by the solver at that particular iteration when the refinement is taking place. So rather than having 10 unknowns, we will be seeking only 6 unknowns.

Equation (14) becomes:

$$\begin{aligned} & \beta_5(x_i - x_c)^2 + \beta_6(y_i - y_c)^2 + \beta_7(z_i - z_c)^2 + \\ & \beta_8(x_i - x_c)(y_i - y_c) + \beta_9(x_i - x_c)(z_i - z_c) + \beta_{10}(y_i - y_c)(z_i - z_c) \\ & = p_i - p_c - \left(\frac{dp}{dx}\right)_c(x_i - x_c) - \left(\frac{dp}{dy}\right)_c(y_i - y_c) - \left(\frac{dp}{dz}\right)_c(z_i - z_c). \end{aligned} \quad (17)$$

This method has the advantage of reducing the computational cost, as solving the system occurs for each cell; it is known that the method of Gauss elimination has the asymptotic complexity of order $(system\ dimension)^3$, which means that we reduce the complexity, even though the reduction is not significant for a problem with coarse mesh, this may be important for more complex problem with large numbers of cells, as this is repeated for each cell for dozens of iterations.

When it comes to the cells on the boundary, we have less equations because of the neighbors, and the number of neighbors may happen to be less than 10; therefore, we do not have an over determined system, and the first method used gives a wrong result. This problem may occur also while dividing the domain into sub domains for parallel computation.

The Hessian matrix obtained, as in the previous case, will be represented by $H(P)_6$, where 6 stands for the number of variables used in the computation.

2.3. Comparison

The two methods were compared, and the results are displayed below. First we check the convergence of $H(P)_{10}$ and $H(P)_6$ to the same result, using real case computation. After that, each method will be used to compute analytical pressure fields where the solution is known.

In the following, the KVLCC2 problem is considered; more details about this study case are given in the introduction of chapter 3. First, we compare the convergence of the methods versus the grid size.

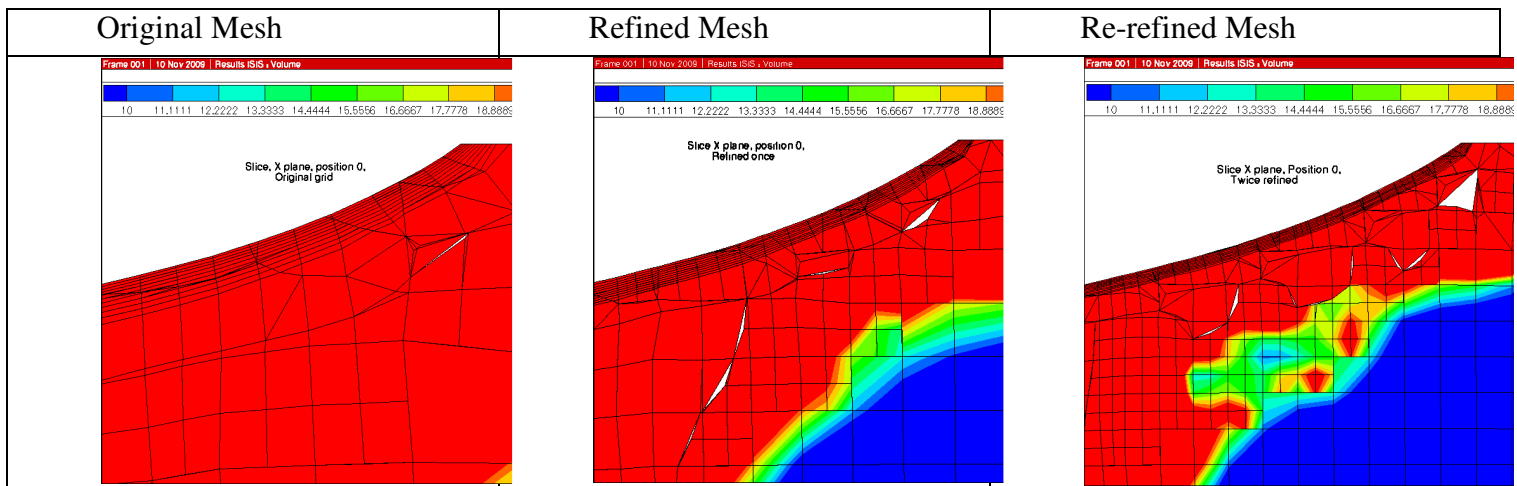


Figure 5: The distribution of the difference between $H(P)_{10}$ and $H(P)_6$, for the position $x=0$, on the coarse grid (left); then for refined grid (centre); then for a twice refined grid (right).

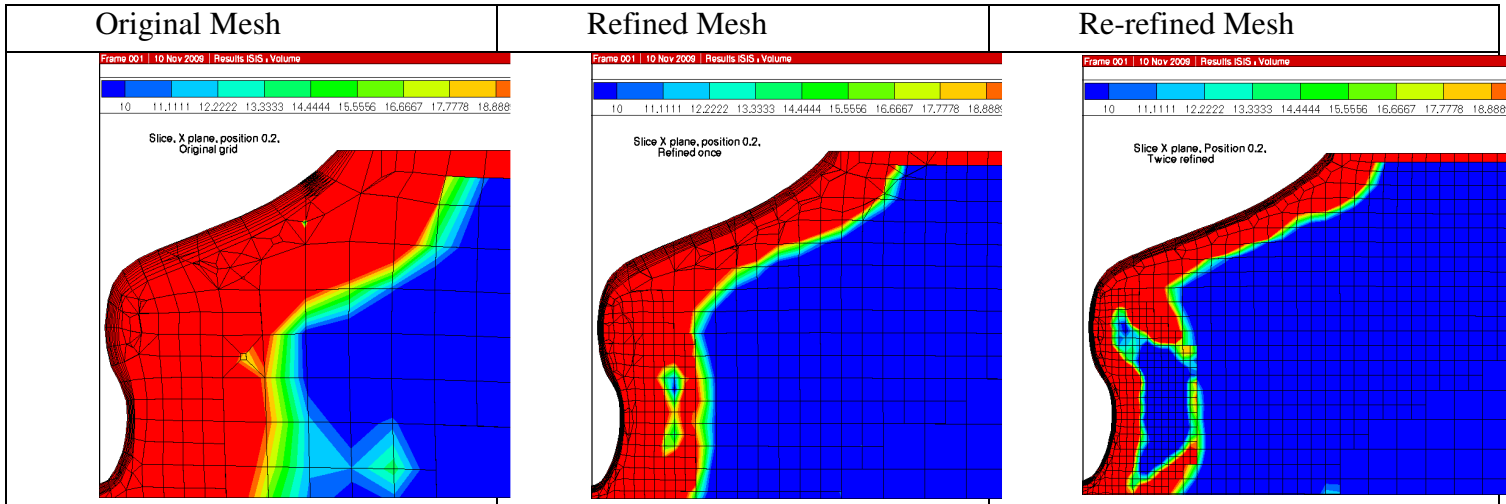


Figure 6: The distribution of the difference between $H(P)_{10}$ and $H(P)_6$, for the position $x=0.2$, on the coarse grid (left); then for refined grid (centre); then for a twice refined grid (right).

The mesh refinement in figure 5 and figure 6 was generated by the mesh generator Hexpress; after each refinement, the grid size is divided by two. The solution is converging with the grid. The field displayed in these images represents the difference between $H(P)_{10}$ and $H(P)_6$, or more precisely, the norm of the difference between the two operators. The Frobenius norm is used:

$$\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2} = \sqrt{\text{trace}(A^* A)}$$

One can observe that, as the mesh grows finer, the difference is becoming smaller, which means that both of the operators converge to the same result.

Now, having in mind the difference between these operators, we will examine the performance of such methods, for some analytical fields, where the analytical solution is available. The gradients of pressure used to compute the second operator are those given by the solver, and not the exact analytical gradient.

Polynomial field

The pressure field is set to be a polynomial in x , y and z . The equation used for pressure field is:

$$p(x, y, z) = x^2 + y^2.$$

The two computational ways were tested. Such a field is very easy to compute; both $H(P)_{10}$ and $H(P)_6$ are very close, and are almost equal to the analytical solution up to machine precision. The results obtained are not displayed in any figure, as the plots do not show significant difference.

Sinusoidal field

The pressure field used here is sinusoidal. The equation is:

$$p(x, y, z) = \sin(y).$$

The results are given in figure 7 and figure 8.

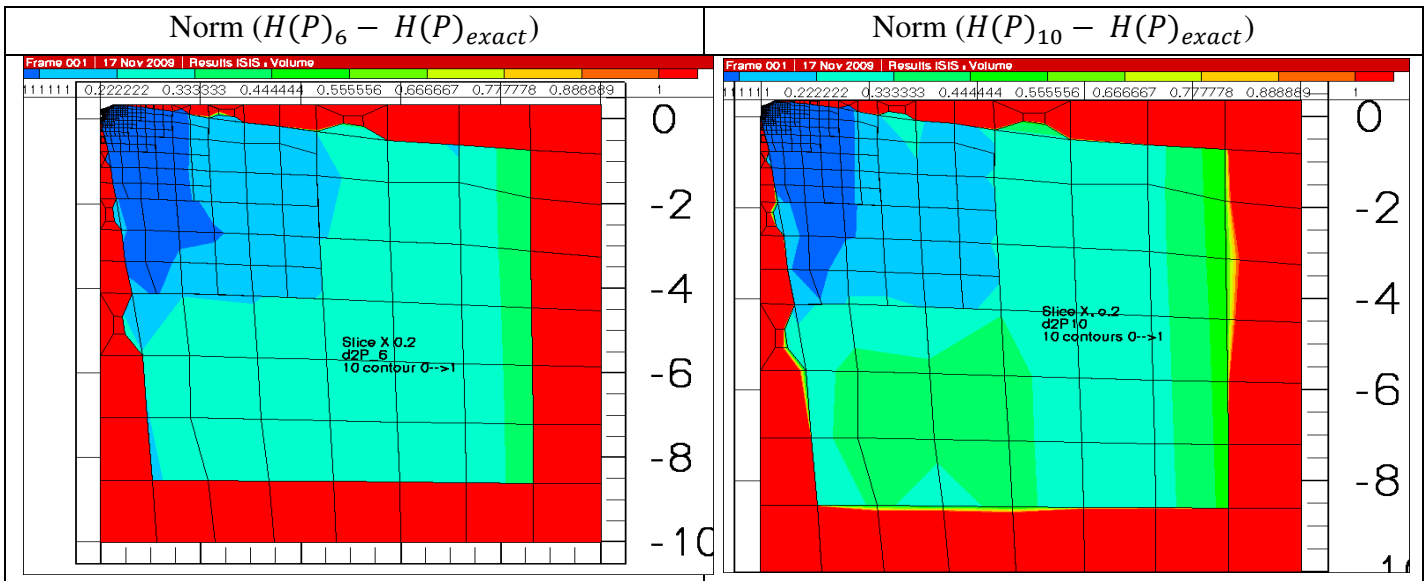


Figure 7: In the position $x=0.2$, the norm of error between $H(P)_6$ and the analytical solution of the sinusoidal field (left), and error between $H(P)_{10}$ and the analytical solution (right).

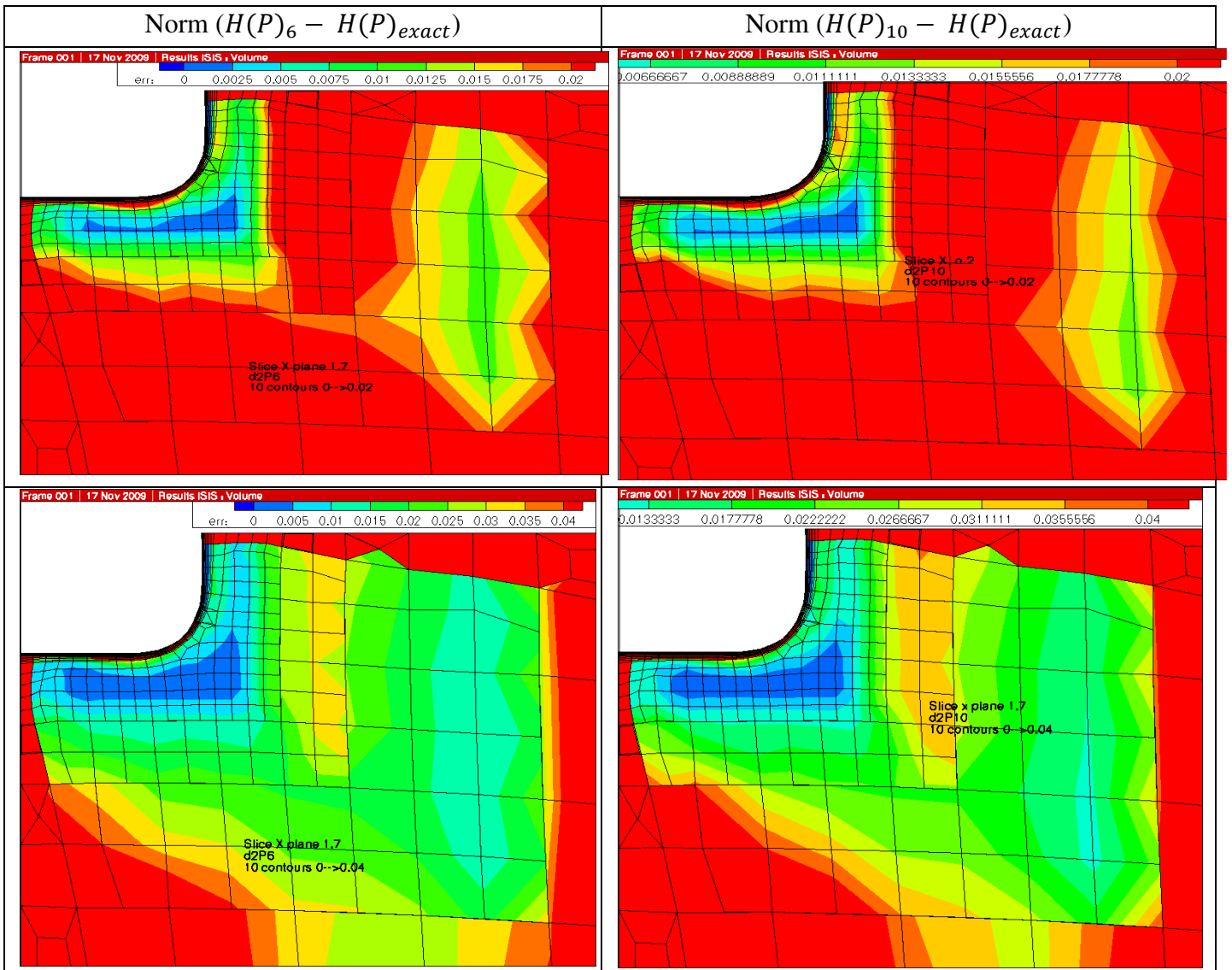


Figure 8: In the position $x=1.7$, the norm of error between $H(P)_6$ and the analytical solution of the sinusoidal field for different contours limit (left, top and bottom), and error between $H(P)_{10}$ and the analytical solution (right, top and bottom).

Fractional field

The pressure field used here is fractional. The equation is:

$$p(x, y, z) = 1 / (x + y).$$

The results are given in figure 9.

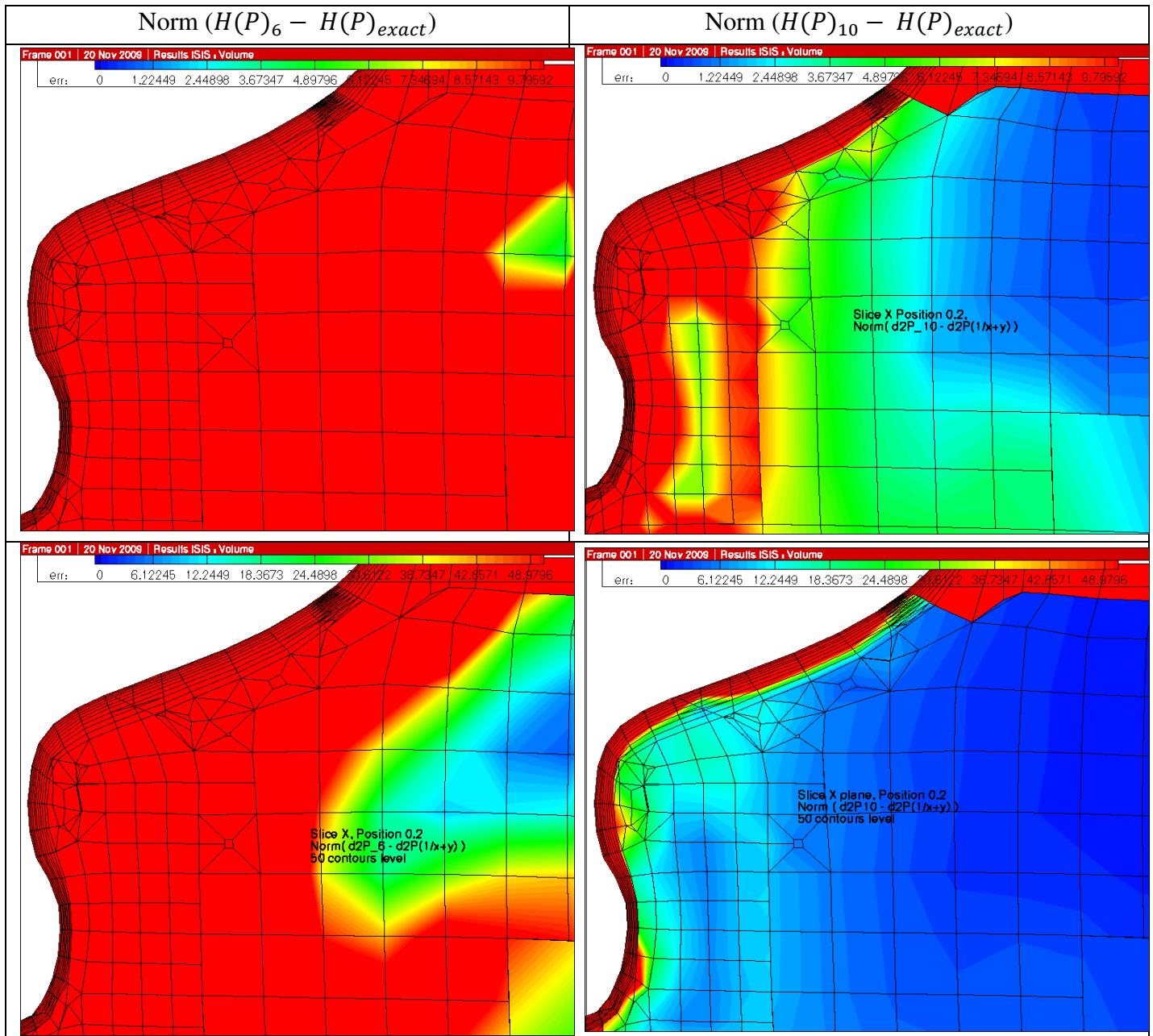


Figure 9: In the position $x=0.2$, the norm of error between $H(P)_6$ and the analytical solution of the fractional field for different contours limit (left, top and bottom), and error between $H(P)_{10}$ and the analytical solution (right, top and bottom).

The pictures show the performance of the two methods on some simple flow fields. The $H(P)_{10}$ method performs better than the $H(P)_6$ method in the fraction case (figure 9); and both give approximately the same result for a sinusoidal field (figure 7 and figure 8).

Both the methods give poor results in the boundary. Such problem is due to the lack of information around the boundary cells, which leads to a loss of accuracy in the computation of the gradients and the Hessian matrix terms. A problem of visualization increases the magnitude of the error observed, as the values at the domain frontier nodes are interpolated from the neighboring cells.

The $H(P)_6$ in theory, gives better result in the boundary, because it needs less equations to be computed; but as in general case we have sufficient number of equations, $H(P)_{10}$ performs well too.

For the rest of the work, the method adopted is $H(P)_{10}$; it will be used to check the efficiency of the refinement criteria.

Chapter 3

Test of refinement criteria

In ISIS-CFD, the refinement procedure is actually performed in a way that the refinement criterion doesn't depend on the cell size, which is not the case of the DTSEE criteria. In fact, the refinement is done in such a way that the quantity: [(Criterion Value) x (Cell Size)] is homogeneous, constant all over the domain and equal to the threshold value. To be able to use the refinement procedure, we shall only consider a criterion based on the norm of the Hessian matrix over each cell. In this case, a suitable threshold value is to be sought, in order to have a good quality of the grid and then of the fields computed.

As the focus in marine applications is on the capturing of vortices, we analyze the bilge vortices behind a ship and check the ability of the criteria to predict such vortices accurately. Traditionally, the interest in the wake flows has been focused on the so-called "hooks" in the propeller plane which are zones of low axial velocity. The flow in this area affects directly the propeller and its efficiency, that is why it is always preferable to have a uniform flow in this zone. However, the presence of strong vorticity is responsible for the creation of such hooks.

For this purpose the KVLCC2 (figure 10) test case will be used. This test case has been extensively analyzed in the past and very detailed data are now available, and secondly, this case was also used to test the refinement for the gradient criteria [5].

The hooks discussed above are particularly present as shown by the experimental data. So predicting such hooks along with the other flow features with the refinement criteria is of particular importance.



Figure 10: KVLCC2 at model scale.

The KVLCC2 was conceived by the Korean Institute of Ships and Ocean Engineering (KRISO) to provide data for both explication of flow physics and CFD validation for a modern tanker ship with bulbous bow [12]. The characteristics for the double body model are presented:

X= 0 at aft perpendicular

Length between perpendiculars, LPP=5.517m

Draft, d=0.3586m

Wetted surface area, S0=8.0838m²

Speed, U=1.047m/s

Reynolds number, Re=4.6 E06

Drag coefficient, with D the effort on the ship

$$C_D = \frac{D}{\frac{1}{2}\rho S U^2} \quad (18)$$

After computing the field variables, we display the results as non-dimensional variables; in this particular situation, the propeller position is obtained by the mean of the operation

$$X_{Propeller} / L_{pp} = 0.0175$$

Beside the validation of the refinement criterion by the capturing of the hooks, we will consider the drag coefficient computed after refinement, and it will be compared to the experimental one.

3.1. Scalar criterion

In this part, many parameters of the refinement procedure are to be changed. The number of generations, basically speaking, refers to how many refinements a given cell can be subject to. In ISIS-CFD, there is the possibility to prevent the refinement in the boundary layer, so, this option will be used to check the effect on the performance of the criterion and the generation of cells.

These computations are performed with 5000 non-linear iterations. The refinement procedure is started after 500 iterations, in order to allow the field variables to be established and start to converge; then it will be called every 50 iterations.

First criterion: $\| H(P)_{10} \|$

	Threshold value	Boundary layer refinement	Turbulence model	Number of generations	Number of cells
CASE 1.1	50	Yes	k- ω SST	2	931 10^3
CASE 1.2	200	Yes	k- ω SST	2	585 10^3
CASE 1.3	300	Yes	k- ω SST	2	501 10^3
CASE 1.4	50	No	k- ω SST	2	705 10^3
CASE 1.5	200	Yes	EASM	2	610 10^3
CASE 1.6	300	Yes	EASM	3	2 205 10^3

Table 1: The tests carried out with the refinement criterion based on the norm of the Hessian matrix, to check the effect of many parameters.

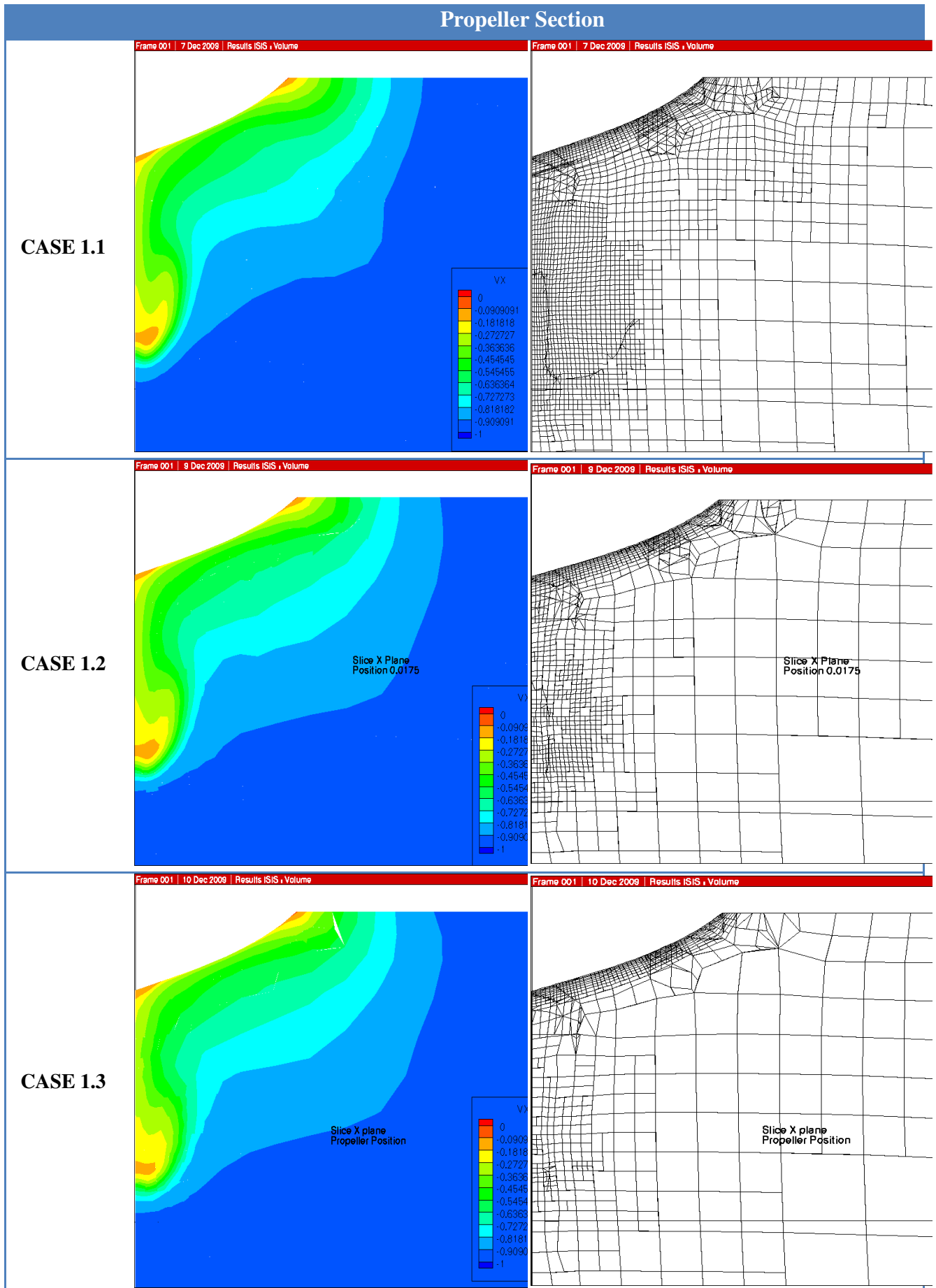


Figure 11: Axial velocity in the propeller plane for different threshold values, the number of generations is set to 2 and the boundary layer refinement is allowed.

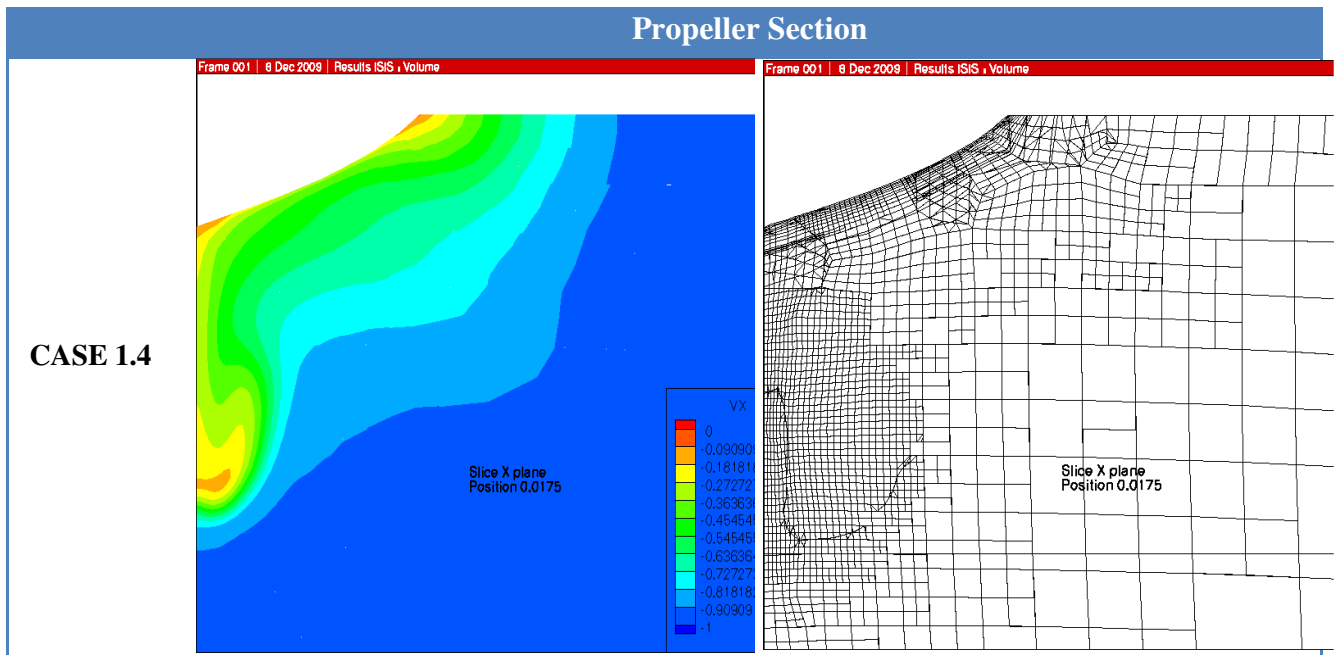


Figure 12: Axial velocity in the propeller plane for threshold value=50, the number of generations is set to 2 and the boundary layer refinement is not allowed.

In case 1.1 (figure 11 on the top) and case 1.4 (figure 12), the threshold value is too small, which means that many cells will be refined. The hook shape is well captured, but the cost is high, even without a boundary layer refinement; the distribution of the cells is uniform and dense in the whole area around the propeller, which is costly and not necessary to capture the hook shape.

In case 1.2 and case 1.3, the number of cells generated is within the range of 500E3, the hook shape is captured; the distribution of the cells is less dense in the propeller area.

If we increase the threshold value, less than 500E3 cells are generated and the hook shape fades, we do not capture anymore the interesting phenomena in that zone.

To check the efficiency of this criterion, we observe its behavior in zones with uniform flow, for example in the middle of the tanker (figure 13). We observe that the flow is well established, there is no variation in the way refinement is taking place in that zone.

The tests carried in the previous studies about the gradient criterion [5] show no variation of pressure in the boundary layer, while unexpectedly, we observe here a slight gradient which causes refinement in the boundary layer. This may be due to changes in the ISIS-CFD computation parameters, since it has been updated many times (figure 14).

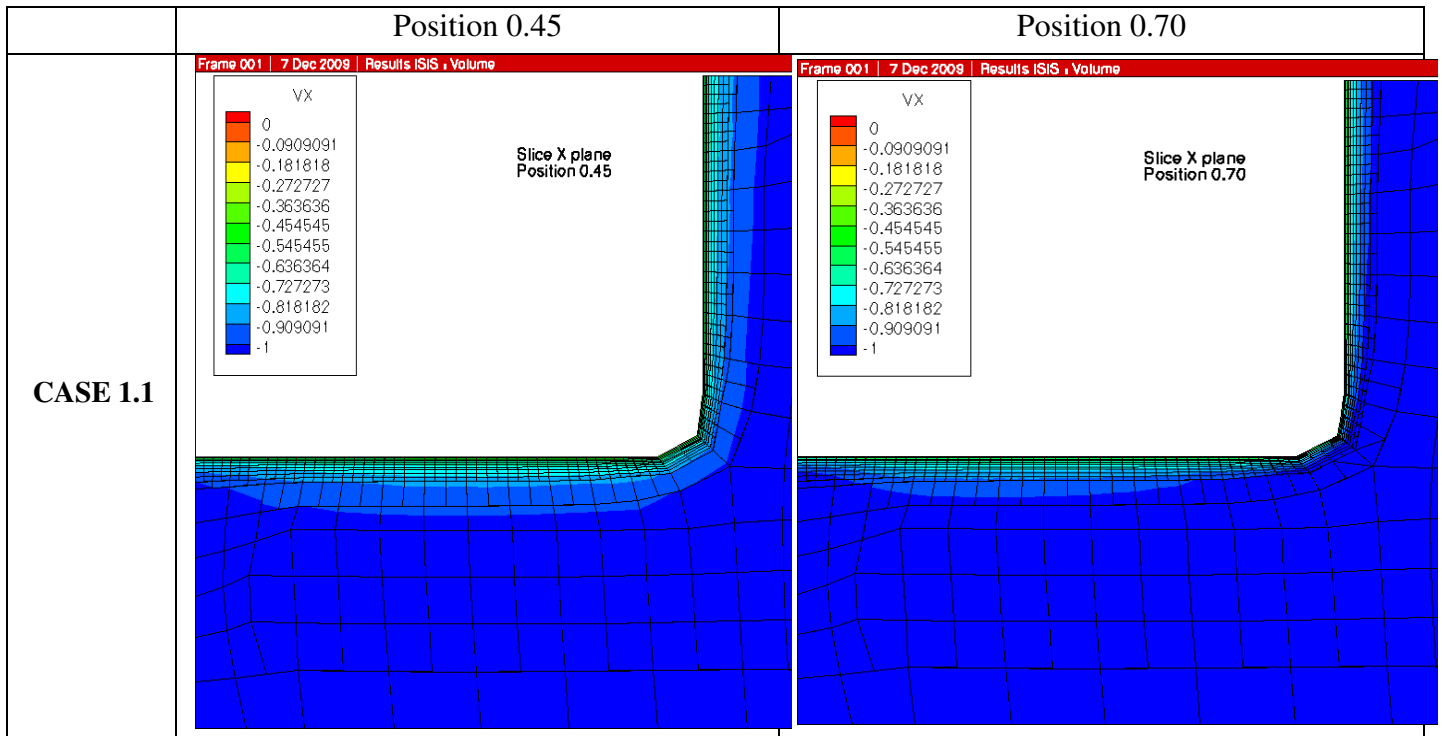


Figure 13: Axial velocity in different sections in the middle of the tanker, $x=0.45$ and $x=0.70$, for a threshold value=50, the number of generations is set to 2 and the boundary layer refinement is allowed.

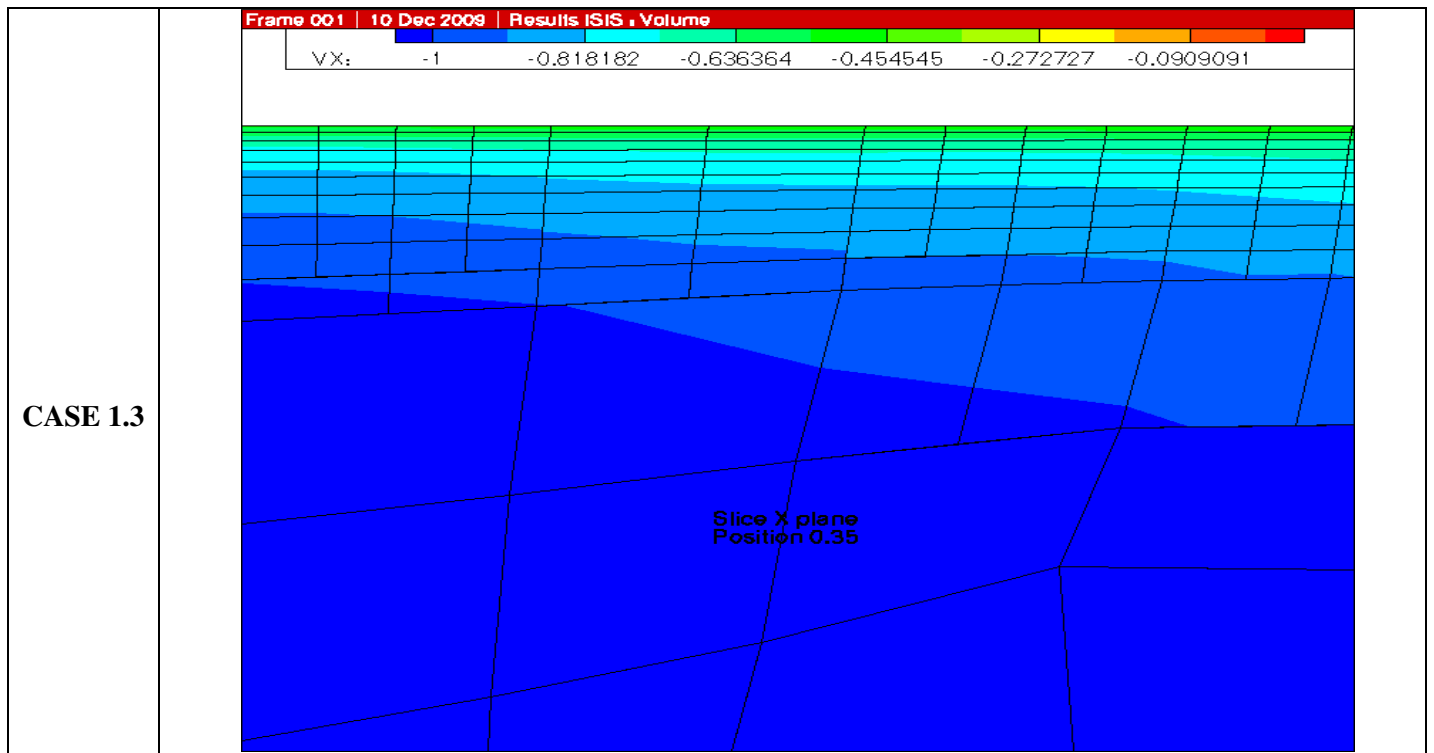


Figure 14: Zoom into a boundary layer region, $x=0.35$, for a threshold value=300, the number of generations is set to 2 and the boundary layer refinement is allowed.

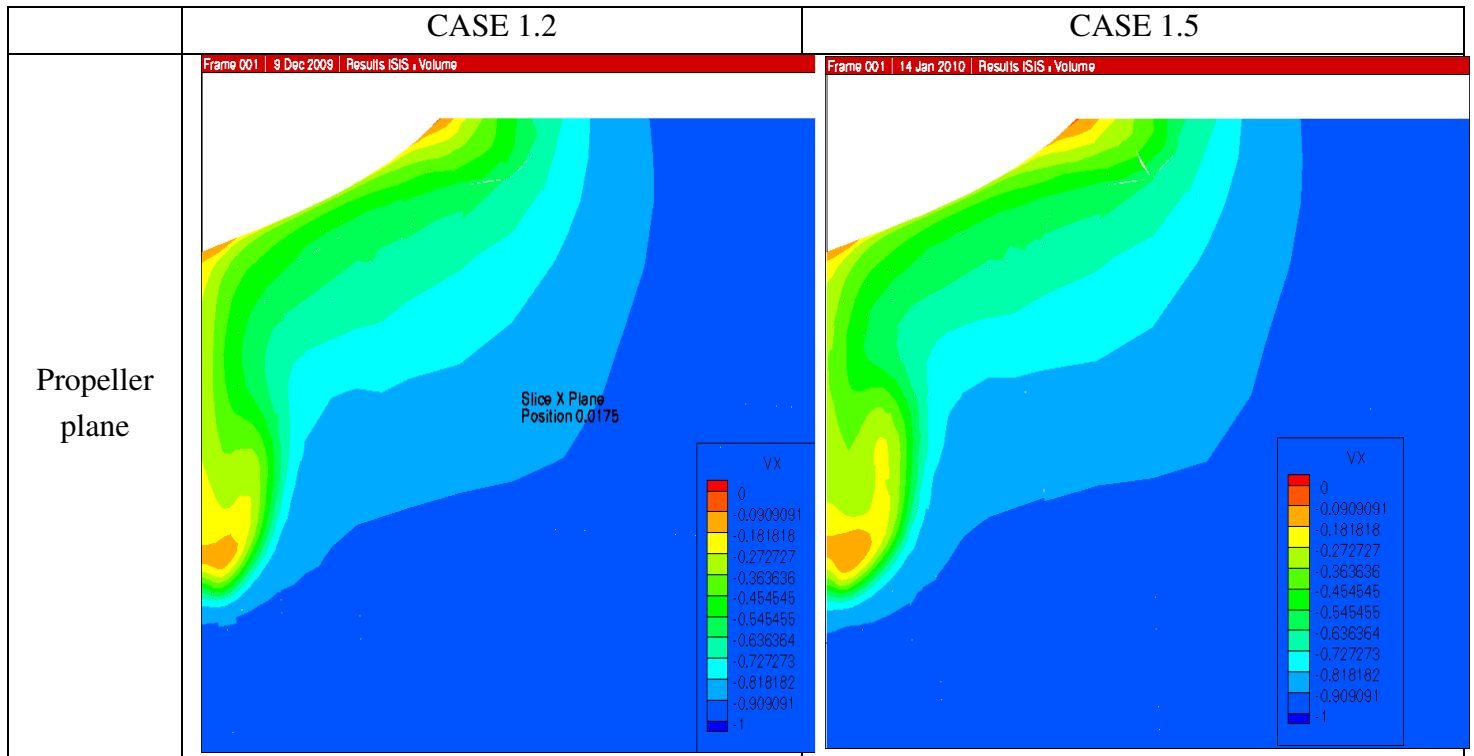


Figure 15: Axial velocity in the propeller plane for threshold value=200, the number of generations is set to 2 and the boundary layer refinement is allowed, k- ω turbulence model (left) and EASM model (right).

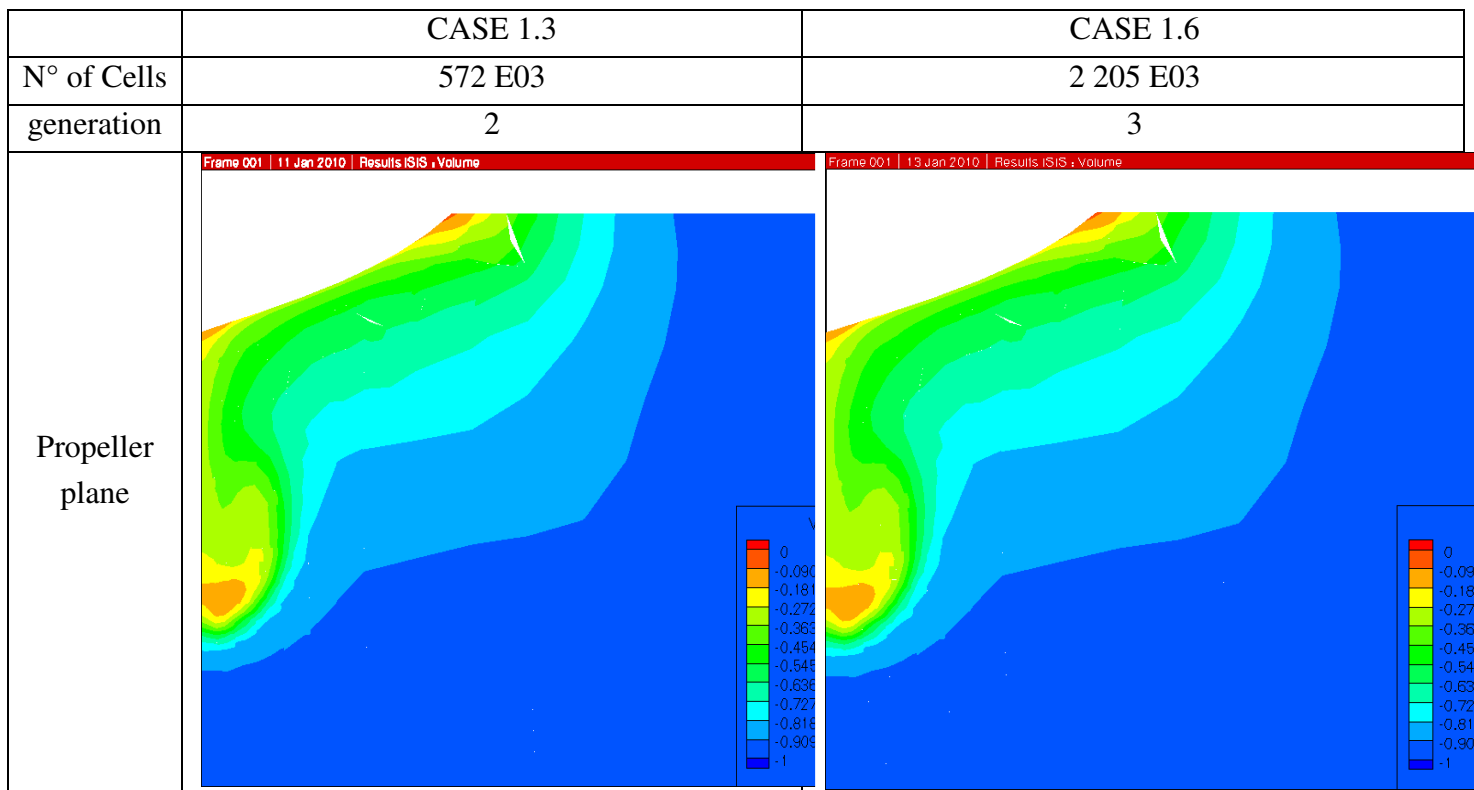


Figure 16: Axial velocity in the propeller plane for threshold value=300, the boundary layer refinement is allowed, EASM model is used, 2 generations (left) and 3 generations (right).

We can observe (figure 15) that the hook shape is better captured using the EASM model than with the k- ω model; this confirms the superiority of the EASM model for the capturing of the flow features, this result has already been stated in the previous studies [5].

By increasing the number of generations (figure 16), the number of cells is multiplied by 4, but still the hook shape is not well captured. Increasing the number of generations results in a costly and inefficient mesh distribution over the domain, since the refinement is taking place mostly in the boundary layer. To gain accuracy, we better decrease the threshold value, rather than increasing the number of generations.

	CASE 1.1	CASE 1.2	CASE 1.4
Drag Coefficient C_D	4.046 E-03	4.065 E-03	4.085 E-03
Experimental	4.110 E-03		
Error %	1.56	1.09	0.61

Table 2: The result obtained for the drag coefficient for a refinement criterion based on the norm of the Hessian matrix, compared to the experimental one [12].

Another way of checking the efficiency of the method is to consider the drag coefficient. The output of ISIS-CFD gives the effort D, which is transformed to the drag coefficient C_D (equation (18)).

We observe that the drag coefficient is converging; it is close to the experimental result, the error is probably coming from the fact that the turbulence model does not capture the entire phenomenon that occurs on the flow, adding to this the error coming from the integration performed to compute the drag coefficient; besides, the experimental value is subject to measurement error too.

Second criterion: $\sqrt{\|H(P)_{10}\|}$

Since $[(\text{Criterion Value}) \times (\text{Cell Size})] = \text{constant}$ is the way refinement is implemented, and because the second derivatives in the Taylor expansion are related to the square of the cell size, we consider here the square root of the Hessian norm matrix.

	Threshold value	Number of generations	Boundary layer refinement	Turbulence model	Number of cells
CASE 2.1	25	2	Yes	EASM	141 10 ³
CASE 2.2	5	2	Yes	EASM	487 10 ³
CASE 2.3	3	2	Yes	EASM	531 10 ³
CASE 2.4	1.5	2	Yes	EASM	717 10 ³
CASE 2.5	1	2	Yes	EASM	973 10 ³
CASE 2.6	3	3	Yes	EASM	1 816 10 ³

Table 3: The tests carried with the refinement criterion based on the square root norm of the Hessian matrix, to check the effect of many parameters.

Propeller Section

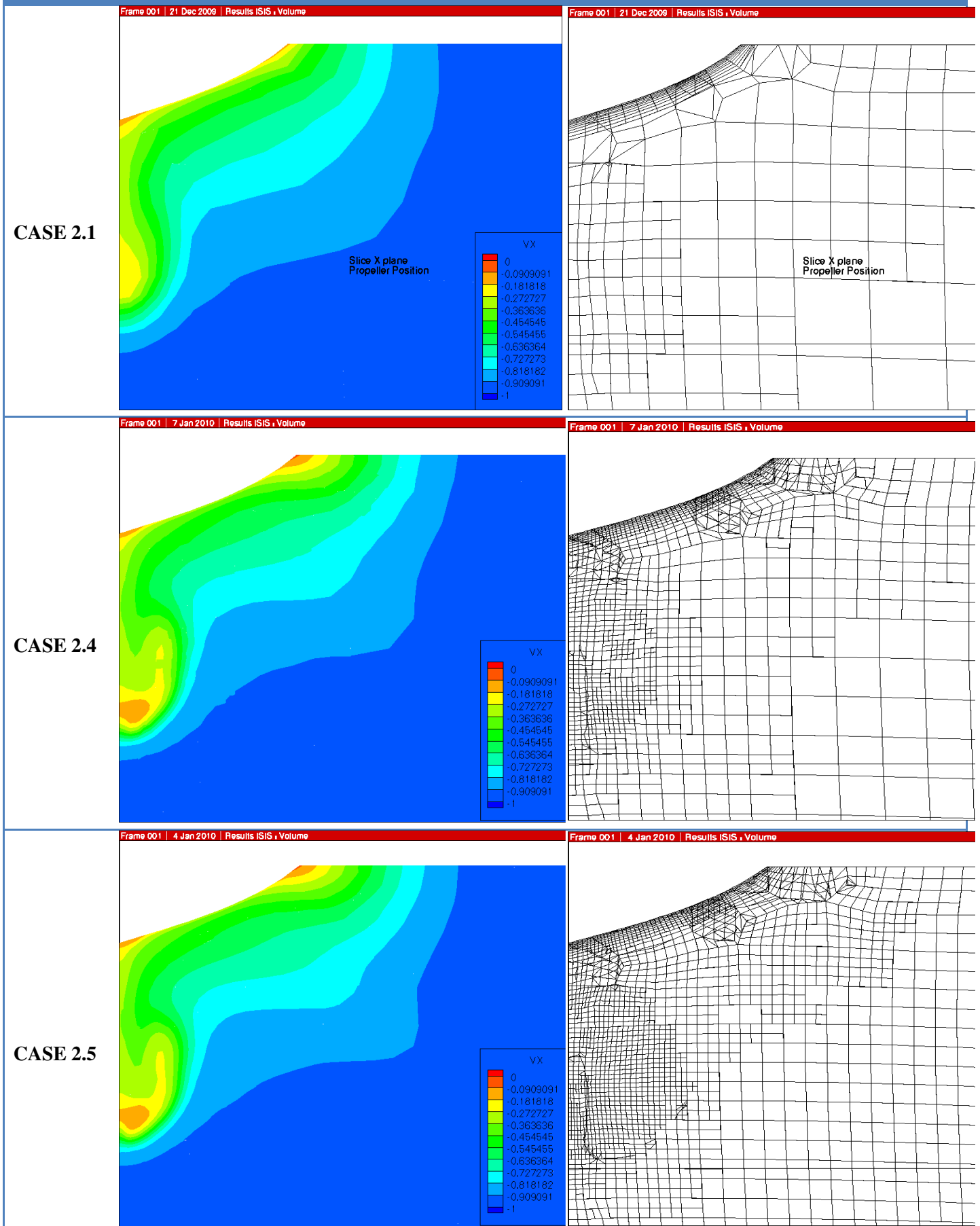


Figure 17: Axial velocity in the propeller plane for different threshold values, the number of generations is set to 2 and the boundary layer refinement is allowed.

These computations are performed with the Explicit Algebraic Stress Model for turbulence, with 5000 non linear iterations. The refinement procedure is started after 500 iterations, in order to allow the field variables to be established and start to converge.

We can observe that in case 2.1, 2. 2 and 2.3, refinement took place in the propeller zone, but this was not enough to capture the features in that area. For a smaller threshold, the hook is captured. For a threshold value of 1.5, the shape of the hook is clear and the mesh is refined almost everywhere; but for a threshold value of 1, the hook is well captured because all the cells are refined around the propeller.

Even though this criterion is highlighting the zones where refinement should take place, it is very expensive. The fact is that this criterion does not give the same distribution of the error as the norm of the Hessian matrix, which causes lot of refinement in the boundary layer.

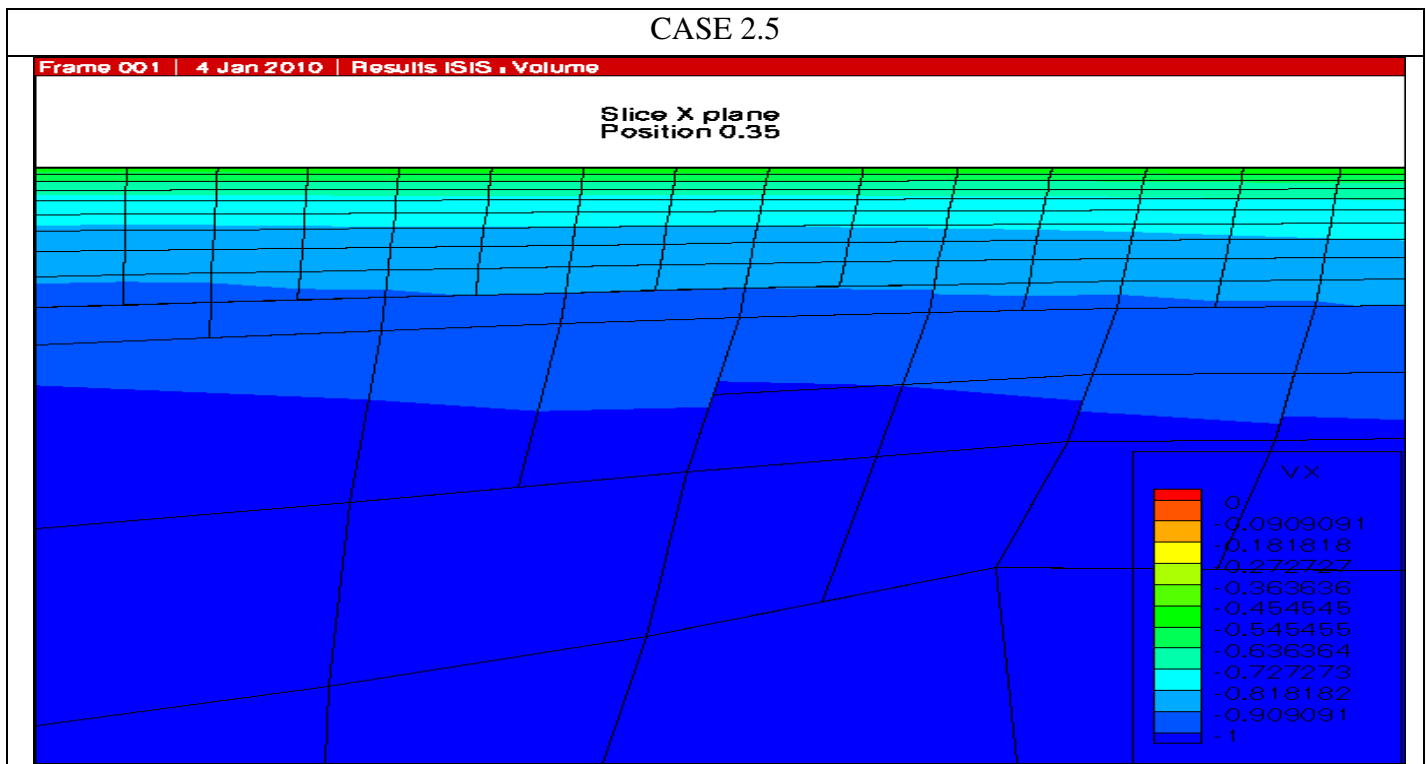


Figure 18: Axial velocity in the middle of the tanker for threshold values equal to 1, the number of generations is set to 2 and the boundary layer refinement is allowed.

This picture shows that refinement takes place even in areas where the flow is changing insignificantly, especially in the boundary layer, right in the middle of the ship. Hence, we do not start to observe the hook shape until 700E3 cells are generated, while with the first criterion, only 600E3 cells is enough.

The increasing of the generation number improves slightly the hook shape, but with a very high number of cells; again, as for the previous criterion, this is an inefficient way of gaining accuracy (figure 19).

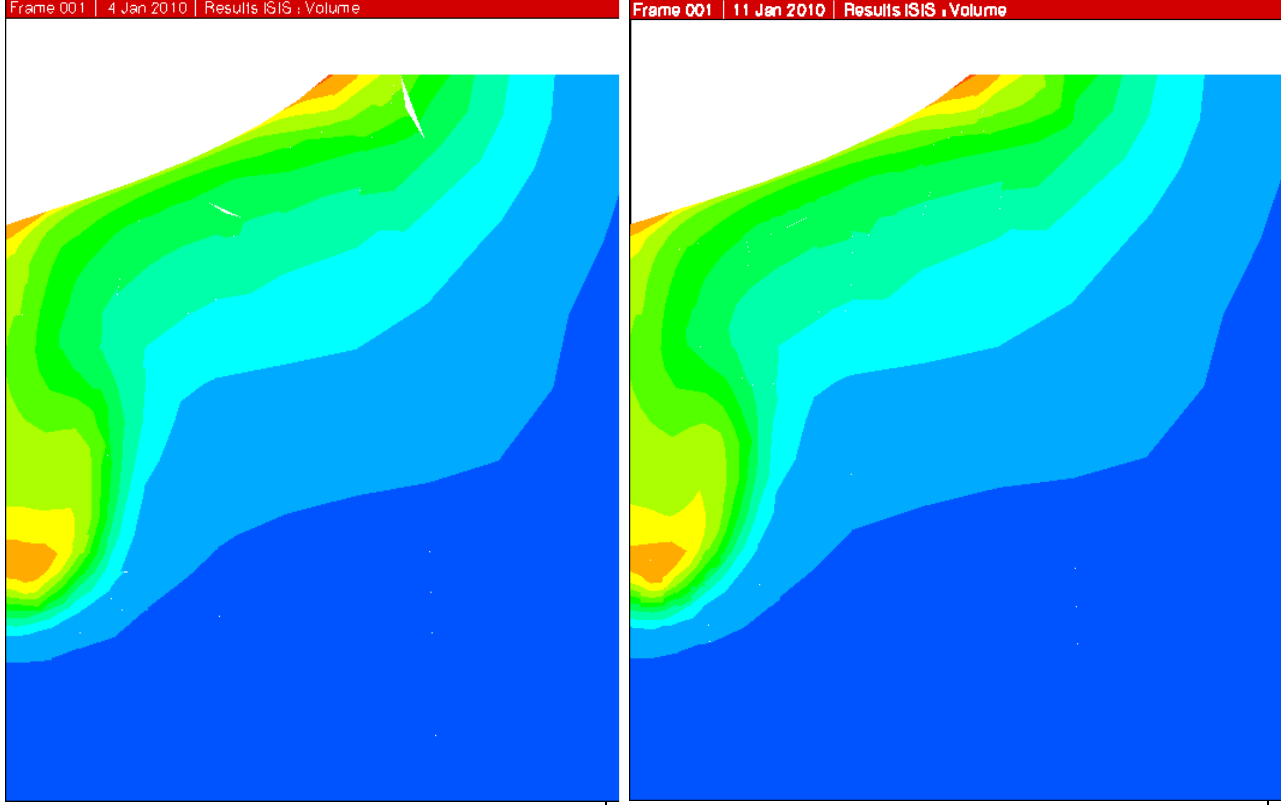
	CASE 2.3	CASE 2.6
Cell number	531 E03	1 816 E03
Number of generation	2	3
result		

Figure 19: Axial velocity in the propeller plane for threshold values equal to 3, the number of generations is 2 (left) and 3 (right), and the boundary layer refinement is allowed.

	CASE 2.1	CASE 2.3	CASE 2.4	CASE 2.5
Drag Coefficient C_D	4.188 E-03	4.167 E-03	4.126 E-03	4.088 E-03
Experimental	4.110 E-03			
Error %	1.90	1.39	0.39	0.53

Table 4: The result obtained for the drag coefficient for a refinement criterion based on the square root norm of the Hessian matrix, compared to the experimental one.

As the mesh is becoming finer, the drag coefficient is getting smaller, and it oscillates around the experimental value, with a small relative error.

3.2. Tensorial criterion

The tensorial approach deals with the anisotropic adaptation, which depends on the specific definition of the cell spacing [10]. The idea behind this approach is to make the refinement decision in another vectorial space, this transformation consists on obtaining the image of actual cells by the mean of a linear transformation; when the image of the space is obtained, the refinement decisions will be taken to have a uniform mesh in that space. For instance, the linear transformation used here is obtained from the Hessian matrix of a field variable.

Every linear transformation could be represented by a matrix in a given space, and as the matrix chosen for the transformation should be definite positive [10], the Hessian matrix H (equation (12)) is transformed using the absolute value of its eigenvalues, the existence of real eigenvalues is ensured since H is a symmetric matrix. The results that are presented below are obtained using the Hessian of the pressure field.

	Threshold value	Number of generations	Boundary layer refinement	Turbulence model	Number of cells
CASE 3.1	300	2	Yes	EASM	268 10^3
CASE 3.2	200	2	Yes	EASM	319 10^3
CASE 3.3	100	2	Yes	EASM	431 10^3
CASE 3.4	75	2	Yes	EASM	491 10^3
CASE 3.5	50	2	Yes	EASM	596 10^3
CASE 3.6	300	3	Yes	EASM	752 10^3

Table 5: The tests carried with the refinement criterion based on the tensorial transformation using the Hessian matrix, to check the effect of two parameters.

The results obtained (figure 20) show that the hook shape is well captured with a good precision using less than $500 \cdot 10^3$ cells, while it requires at least $600 \cdot 10^3$ for the scalar criteria. This method is efficient since for an adequate threshold value, there is no refinement in the boundary layer even though it is allowed (figure 21), and the variation of pressure in the boundary layer is not observed everywhere any more.

Propeller Section

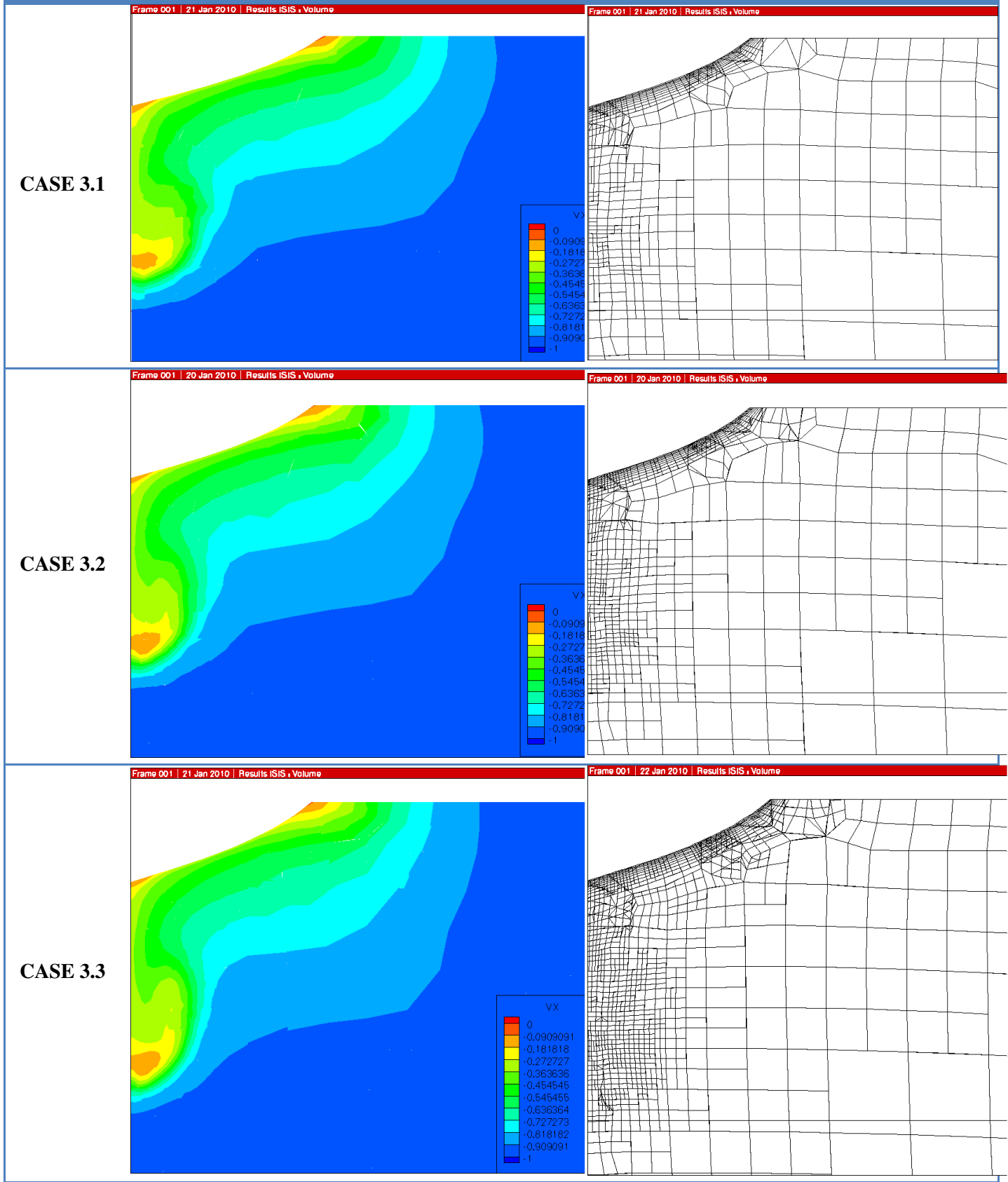


Figure 20: Axial velocity in the propeller plane for different threshold values, the tensorial criterion based on pressure is used, the number of generations is set to 2 and the boundary layer refinement is allowed.

Propeller Section

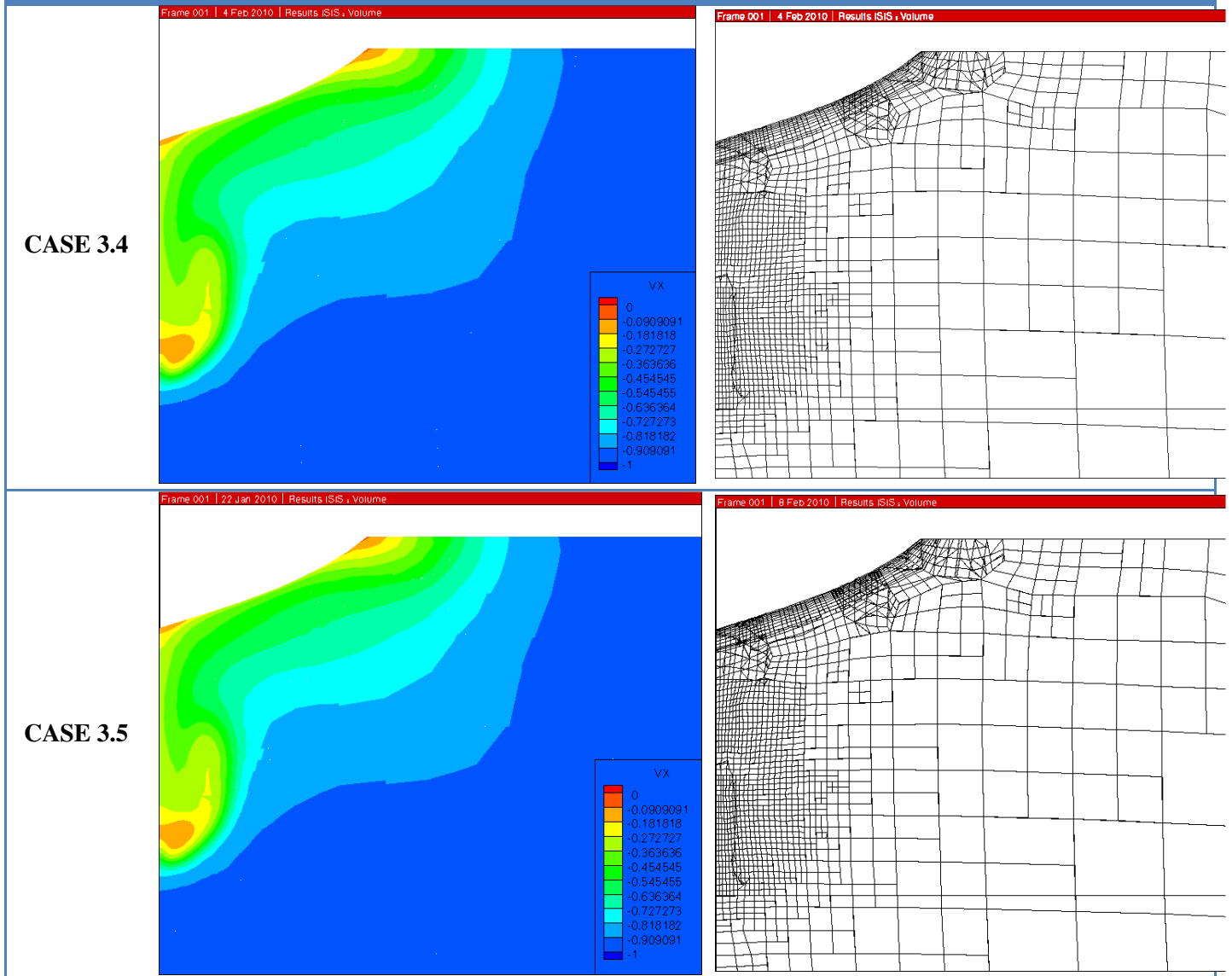


Figure 20: (Continued).

CASE 3.4

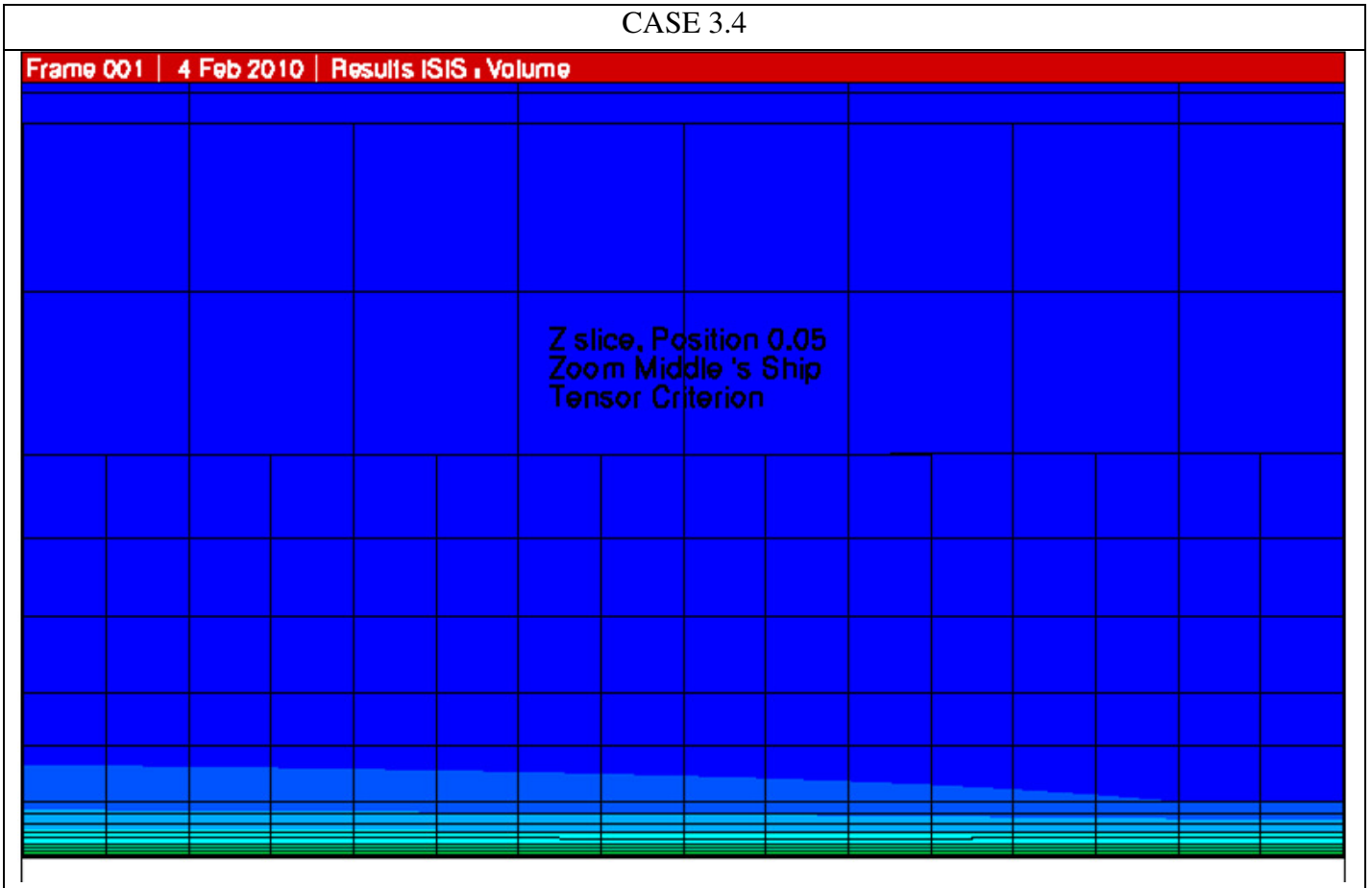


Figure 21: Axial velocity in the middle of the tanker for threshold values equal to 75, the number of generations is set to 2 and the boundary layer refinement is allowed.

CASE 3.4

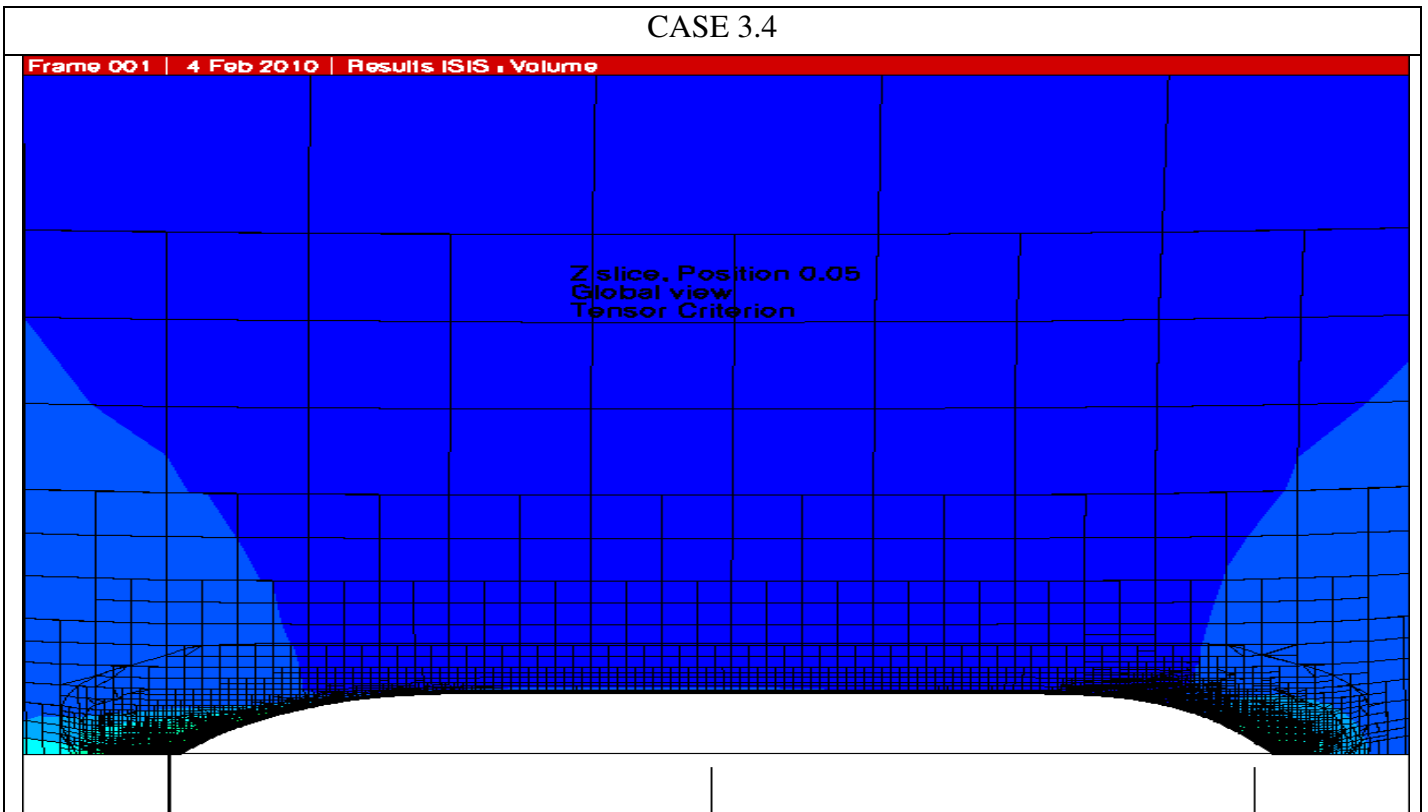


Figure 22: General view of the mesh around the KVLCC2, for a threshold value equal to 75, the number of generations is set to 2 and the boundary layer refinement is allowed.

CASE 3.4

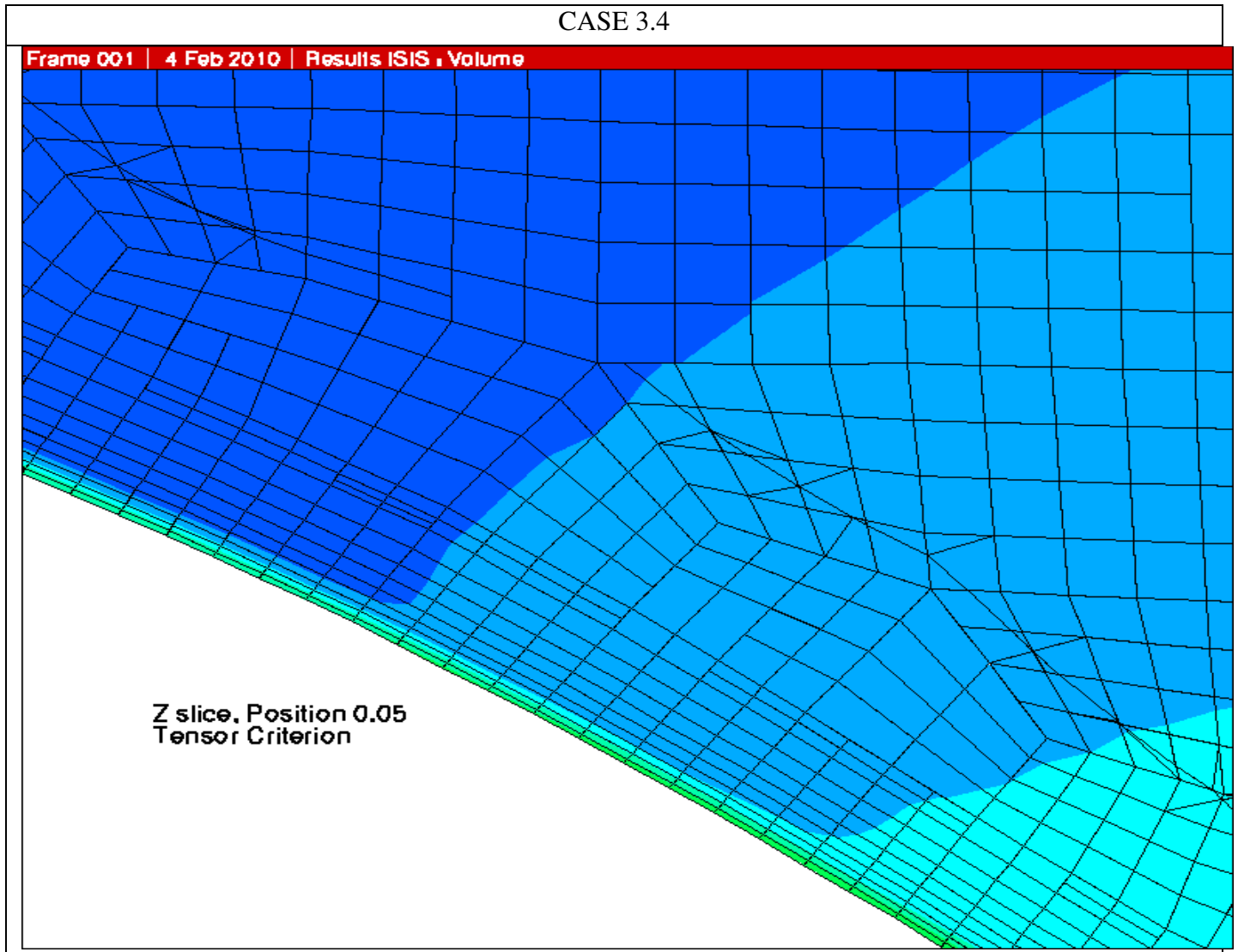


Figure 23: View of the mesh in the front of the KVLCC2, for a threshold value equal to 75, the number of generations is set to 2 and the boundary layer refinement is allowed.

The general view (figure 22) highlights that the refinement takes place in the zones where there is a variation of the flow, as the mesh density is higher in the bow and the stern of the ship. However, it was observed that in some of the areas where the refinement takes place, there is an arbitrary distribution of the mesh (figure 23), since some cells are refined and others not, some are refined in one direction while the surrounding cells are refined in both directions.

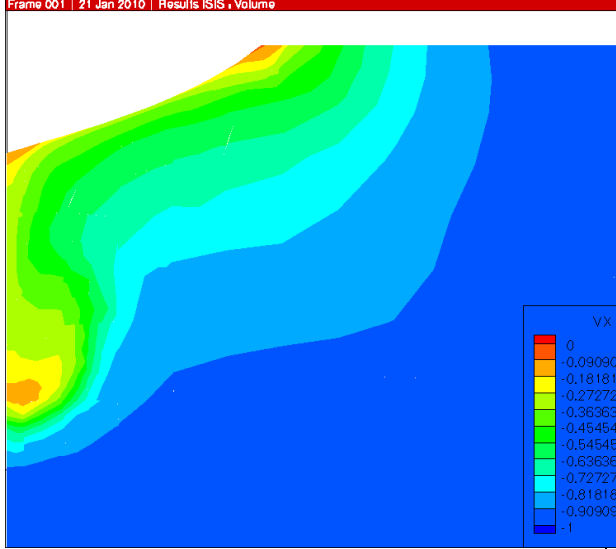
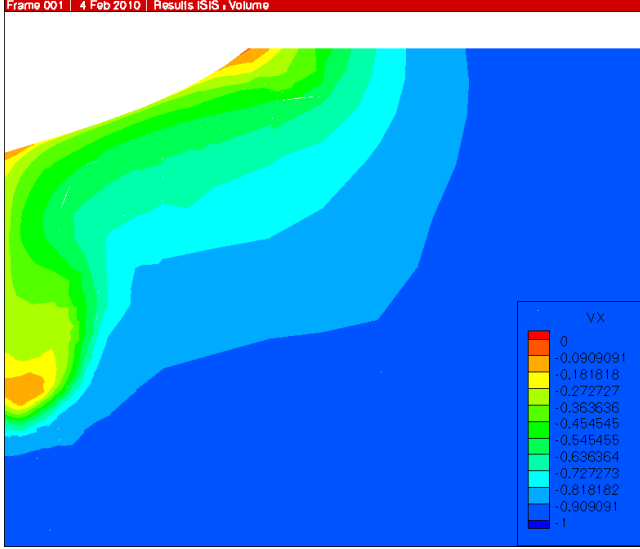
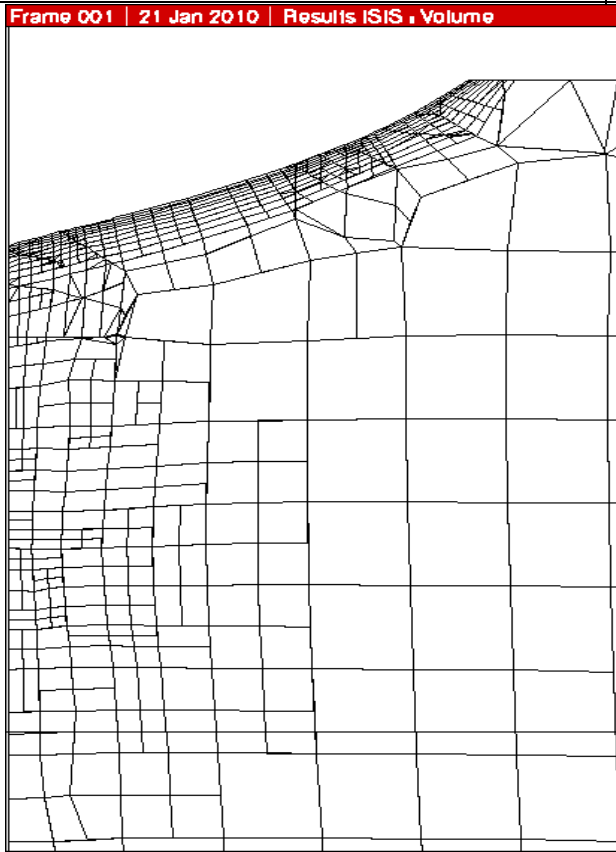
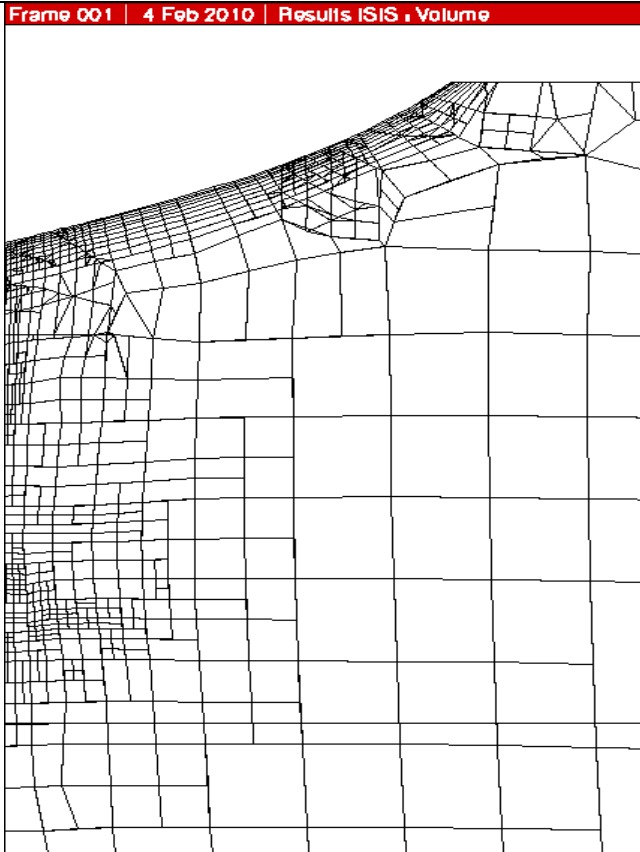
	CASE 3.1	CASE 3.6
Cell number	268 E03	752 E03
Number of generation	2	3
result		
		

Figure 24: Axial velocity (top) and mesh distribution (bottom) in the propeller plane for threshold values equal to 300, the number of generations is 2 (left) and 3 (right), and the boundary layer refinement is allowed.

Concerning the number of generations, the tensorial criterion behaves as the scalar one, the increasing of the number of generations leads to an unnecessary increase of the number of cells (figure 24) ; while the same accuracy could be retrieved by simply decreasing the threshold value.

The drag coefficient (table 6) is converging to a constant value as the threshold number is increasing. As mentioned before, many sources of the error can be identified, for example the turbulence model, the integration scheme and the uncertainty lying in the experimental result.

	CASE 3.1	CASE 3.2	CASE 3.3	CASE 3.4	CASE 3.5
Drag Coefficient C_D	4.220 E-03	4.217 E-03	4.198 E-03	4.195 E-03	4.194 E-03
Experimental	4.110 E-03				
Error %	2.68	2.60	2.14	2.07	2.04

Table 6: The result obtained for the drag coefficient for a refinement criterion based on the tensorial approach with the Hessian matrix of pressure, compared to the experimental result.

3.3. Cell generation behavior

In ISIS-CFD, the procedure of refinement should be called after a certain number of iterations, in order to have a relatively established flow before using the flow variables to compute a refinement criterion; after that, the refinement procedure may be called repeatedly by setting a step based on the number of iterations desired between two refinements.

One can observe that after many calls of refinement procedure, the number of cells generated is not changing significantly; an example is given in figure 25.

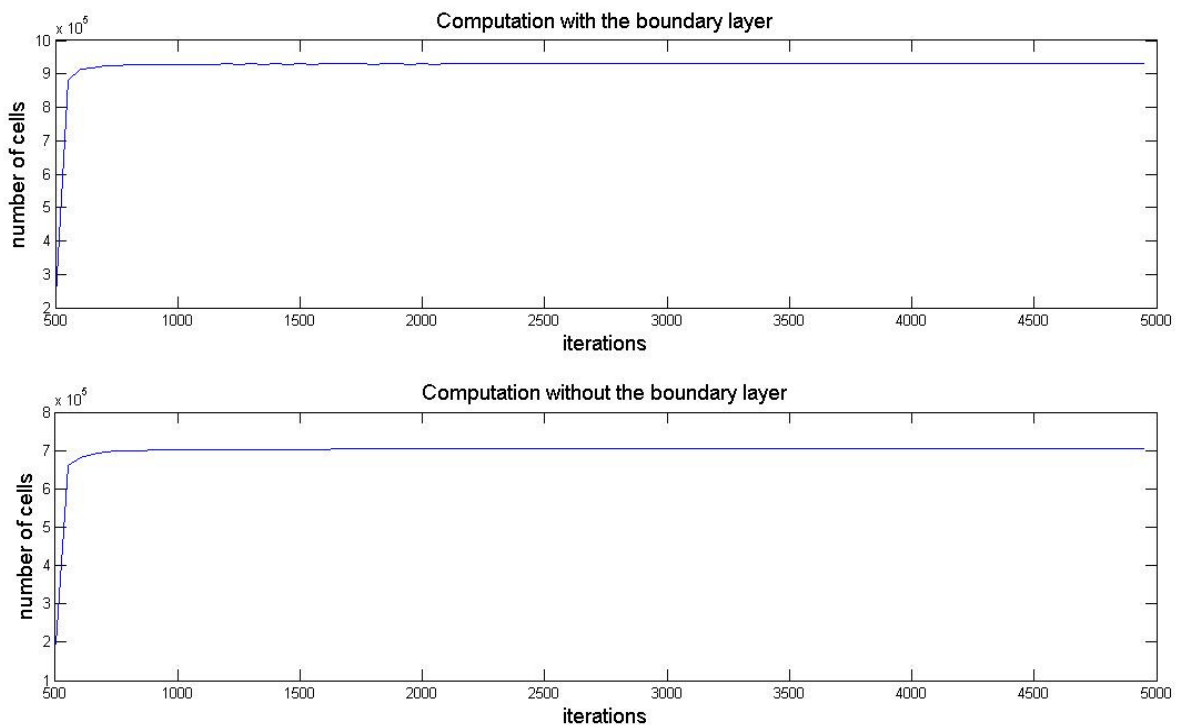


Figure 25: The evolution of the number of cells with iterations; in the case 1.1 where refinement is allowed in the boundary layer (top), and case 1.4 without refinement in the boundary layer (bottom).

After a few refinement steps only, the number of cells generated is converging to a constant value, but the refinement is still called after each 50 iterations, which results in an unnecessary and costly treatment coming from the use of a complex criterion such as tensorial approach.

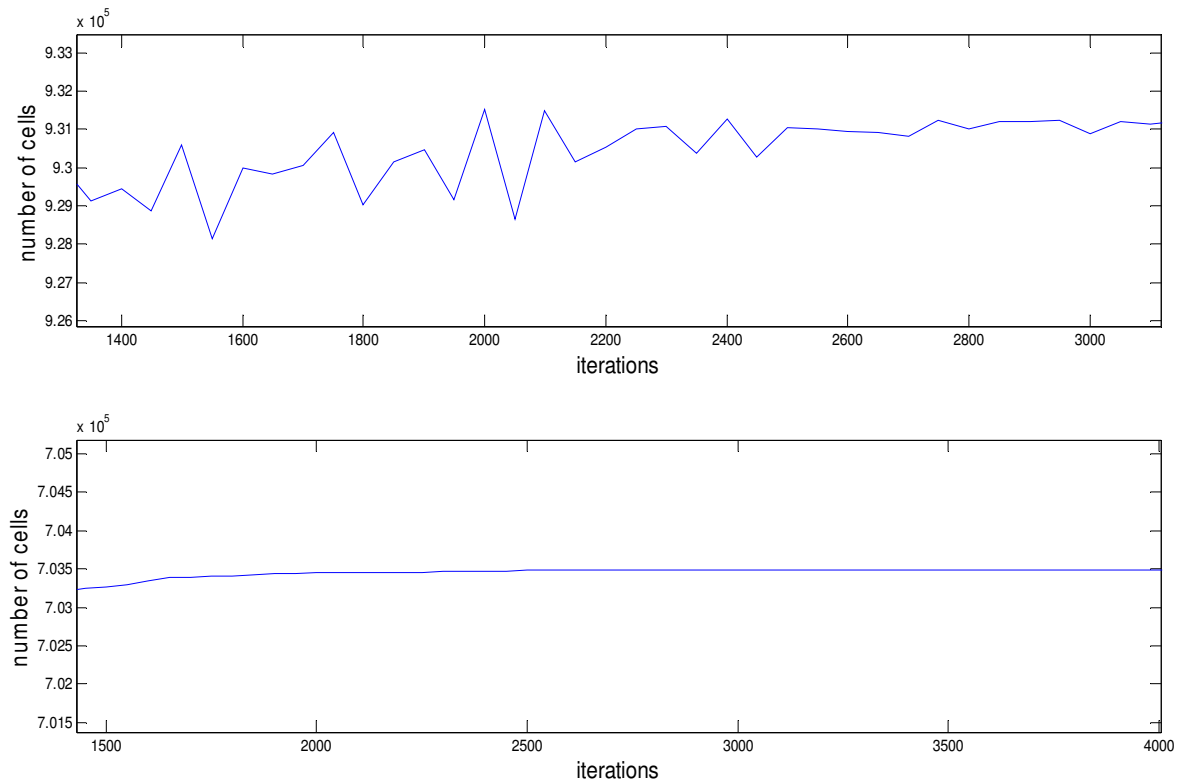


Figure 26: A zoom in the curve representing the evolution of the number of cells with iterations; in the case 1.1 where refinement is allowed in the boundary layer (top), and case 1.4 without refinement in the boundary layer (bottom).

A zoom in the graphics shows that the cells number is stabilized within a range (figure 26); for a computation with boundary layer, this range is of the order of 2000 cells for 900E3 cells, ie 0.22 %; while without boundary condition, this range is of the order of 200 only, for 700E3 cells approximately, ie 0.03 %.

Chapitre 4

Conclusion

This study helped to compare two approaches for adaptive grid refinement using the pressure as the field of reference. The first approach used classically scalar criteria; two scalar criteria were implemented and tested on a real field computation of flow around the KVLCC2 tanker. The first criterion is based on the Frobenius norm of the Hessian matrix of the pressure field, while the second one is based on the square root of the same norm. Among the two scalar criteria used, the one based on the Frobenius norm of the Hessian matrix performs better, considering the number of cell needed to reach certain accuracy and the distribution of the cells in the domain. The square root of the norm of the Hessian matrix tends to be very expensive compared to the accuracy achieved.

The second approach based on second order tensors was used. The Hessian matrix of the pressure field is computed, then, used to map the actual mesh to another space where the decision of refinement or not is taken, based on the threshold value adopted. The tensorial approach gives even better results; the number and distribution of cells are far more interesting. For the entire criteria, the drag coefficient computed is within an acceptable range around the experimental coefficient.

This work helped to develop a numerical implementation of the DTSEE method, and highlighted an error developed in the previous studies [1, 3]. It was also found that using 10 degrees of freedom to compute the Hessian matrix gives reasonably better results than using only 6 degrees of freedom; and also, these two methods converge to the same result with the grid.

In the following, some suggestions for further development are presented. The idea, for now, is to focus in the tensorial criteria, and in order to validate this approach, it will be interesting to check the efficiency of the method using other field variables than pressure. All the computations are done based on the KVLCC2 test case, it is also necessary to try this method using other test cases to check the behavior of the adaptive grid refinement.

It is important to mention that, to be able to present significant results, many other computations than those presented were performed, which is a time consuming task. As mentioned in the last chapter, there are many possibilities to optimize the adaptive grid refinement and mesh generation process; for example, a tool that can be useful is to set a parameter that detect the change in cells number after each refinement performed, and if this change is less than 0.1 % for 5 consecutive refinements, it means that the refinement is no more useful, therefore we can stop the refinement and carry on with the non linear iterations to compute the field variables without any mesh adaptation. This tool will be very useful, especially if the computation of the refinement criteria is performed using tensorial approach or even adjoints methods.

Also, for sake of homogeneity, the whole study was conducted using the least squares method to compute the Hessian matrix; however, the weighted least squares method has been implemented and tested for simple analytical fields. The first numerical results show the supremacy of the

weighted least squares method as expected, since the error on the Hessian matrix is reduced considerably; but it is still necessary to test this method for real flow fields.

Another approach for the adaptive grid refinement is to be used, rather than using a threshold value to have a homogeneous distribution of the cells in the domain, a new approach is based on the maximum number of cells desired; for that, the threshold value will be adjusted to reach the number of cells desired at the end of the computation.

References

- [1] H. Jasak. *Error Analysis and Estimation for the Finite Volume Method, with Applications to Fluid Flows*, Ph.D. thesis, Imperial College of Science, Technology and Medicine, UK, June 1996.
- [2] H. Jasak and A.D. Gosman, *Element residual error estimate for the finite volume method*, Computers & Fluids 32 - 223–248 (2003).
- [3] A. Hay. *Etude des stratégies d'estimation d'erreur numérique et d'adaptation locale de maillages non-structurés pour les équations de Navier-Stokes en moyenne de Reynolds*, Ph.D. thesis, University of Nantes, France, February 2004.
- [4] J. Wackers and M. Visonneau, *Adaptive grid refinement for ISIS-CFD*, 11th Numerical Towing Tank Symposium (NuTTS 2008), Landeda, France.
- [5] A.Zaib, *Error indicators for controlling automatic grid refinement in ISIS-CFD*, Master of Science Thesis, Ecole Centrale de Nantes, France.
- [6] H.T. Rathod , K.V. Nagaraja and B. Venkatesudu ,*Numerical integration of some functions over an arbitrary linear tetrahedra in Euclidean three-dimensional space*, Applied Mathematics and Computation 191- 397–409 (2007).
- [7] N. Talon Kasmai, *Solution adaptive mesh strategies for flow with vortices*, Master of science thesis, Department of Aerospace Engineering, Mississippi state, Mississippi, August 2008.
- [8] A. Hay, M. Visonneau , *High-order methods for the numerical simulation of vortical and turbulent flows, adaptive mesh strategy applied to turbulent flows*, C. R. Mécanique 333 - 103–110 (2005).
- [9] C. Helf, U. Küster, *A Finite Volume Method With Arbitrary Polygonal Control Volumes and High Order Reconstruction for the Euler Equations*, Computational Fluid Dynamics, John Wiley and Sons, September 1994.
- [10] J.Majewski, *Anisotropic adaptation for flow simulations in complex geometries*, in 36th CFD /ADIGMA course on HP-Adaptive and HP-Multigrid methods, Von Karman Institute for Fluid Dynamics, October 2009.
- [11] en.wikipedia.org/wiki/Least_squares.
- [12] www.nmri.go.jp/cfd/cfdws05.