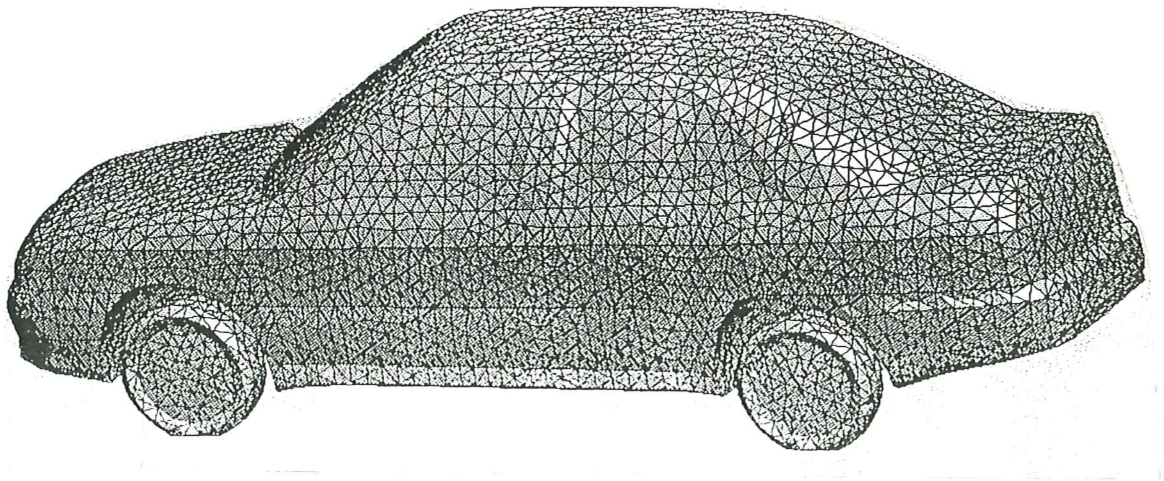
3D finite element computations for viscous aerodynamic flows around automobiles

T. Fischer
J. Miquel
O. Fruitós
E. Oñate



Publication CIMNE Nº 50, May 1994

3D finite element computations for viscous aerodynamic flows around automobiles

T. Fischer
J. Miquel
O. Fruitós
E. Oñate

Publication CIMNE Nº 50, May 1994

International Center for Numerical Methods in Engineering
Gran Capitán s/n, 08034 Barcelona, Spain

Contents

1. INTRODUCTION

The aim of this report is to document the research carried out at CIMNE concerning the numerical simulation of viscous fluid flow around moving vehicles using the finite element method .

The difficulty with the representation of 3D viscous fluid flow has always been the necessary vast amount of points and the many iteration steps to converge to the steady state solution. Using adaptive methods and local timestepping a more efficient use of internal capacities is possible. In addition, the recent development at CIMNE of required tools such as a mesh generator, post processing and visualization software for three dimensions has made it at all possible to obtain and to critically analyze results.

The flow solver is an explicit two-step Taylor-Galerkin scheme [1] that requires little memory per node and element. In addition, cputime is linearly related to the number of nodes and elements. The power of our approach is that both complex geometries and viscous flow simulation can be performed at a resonable cost of stored memory. Complex geometries can be discretized by means of finite elements and the two-step scheme is most efficient by reducing memory needs without sacrificing accuracy [1]. Hence, the bottleneck of finite element calculations is not the solver anymore, but the capabilities of generating an adequate mesh, which is also subject of this article.

This study reports first successful results and excellent approximation of aerodynamical values around the geometry of a commercial automobile, the SEAT Toledo. The geometry of the SEAT Toledo, was obtained from a CAD-file which was gently supplied by SEAT. Using Patran [11] an input file for the mesh generation process was obtained. This was used to obtain the final mesh of tetrahedra discretizing the three dimensional domain of analysis using the mesh generator developed at CIMNE which was subsequently introduced as the mesh for the numerical simulation process. The mesh is the discretization of the flow domain with tetrahedral elements.

The study was performed involving many unknown parameters such as computation speed, memory requirements, mesh quality and mesh size. We first had to gain experience in order to find the right combinations of these parameters to be able to reach basic results. The first part of this report documents the features of the fluid flow solver, then a description of the mesh generation process is given and finally the results of the aerodynamic study are presented.

2. BASIC THEORY

The solution of the compressible Navier-Stokes equations using the finite element method and a Taylor-Galerkin scheme is described first. As recently shown [2,3], this algorithm allows to deal successfully with quasi-incompressible flow situations typical in the aerodynamics of automobiles. The numerical algorithm is capable of simulating laminar viscous flow conditions past geometries in two and three dimensions. The spatial discretization is based on linear finite elements (triangles and tetrahedra in two and three dimensions, respectively). The algorithm is extended in a two-step fashion using a lumped mass matrix formulation to allow a reduction of stored memory and the use of an explicit time integration scheme [1]. The implementation involves the use of local timestepping to reach the quasi steady state solution.

The finite element method used is based on the well known two-step Taylor-Galerkin algorithm documented by Peraire [1], Zienkiewicz [5] and Oñate et al [16]. Various results obtained with this approach are reported by Quintana [6] and Fischer et al [7]. The following is a brief outline of this method.

2.1 Navier-Stokes Equations

The Navier-Stokes set of equations for a compressible fluid in conservative (or divergence) form for a laminar flux within a three dimensional domain reads, in the absence of source terms:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_i}{\partial x_i} = \frac{\partial \mathbf{g}_i}{\partial x_i} \qquad i = 1,3 \tag{1}$$

where the nodal unkowns $\mathbf{u}$, the advective fluxes $\mathbf{f}_i$ and the viscous fluxes $\mathbf{g}_i$ are:

$$\mathbf{u} = \begin{Bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho \epsilon \end{Bmatrix} \quad \mathbf{f}_i = \begin{Bmatrix} \rho u_i \\ \rho u_i u_1 + p\delta_{1i} \\ \rho u_i u_2 + p\delta_{2i} \\ \rho u_i u_3 + p\delta_{3i} \\ (\rho\epsilon + p)u_i \end{Bmatrix} \quad \mathbf{g}_i = \begin{Bmatrix} 0 \\ \sigma_{1i} \\ \sigma_{2i} \\ \sigma_{3i} \\ k\,\frac{\partial T}{\partial x_i} + u_j\,\sigma_{ji} \end{Bmatrix} \tag{2}$$

Assuming a bulk viscosity of $(3\lambda + 2\mu) = 0$, the Navier-Poisson law for a Newtonian fluid leads to the components of the viscous stress tensor as

$$\sigma_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{3}$$

In above: ρ is the density, p is the pressure, T is the temperature, u_i Cartesian components of the velocity, ϵ the total energy per unit mass, δ_{ij} the Kronecker delta, k

is the material heat conductivity, and $\mu = \mu(T)$ is the dynamic coefficient of viscosity which is calculated from Sutherland's equation:

$$\mu(T) = \mu_0 \left(\frac{T_0 + S_0}{T + S_0}\right) \left(\frac{T}{T_0}\right)^{3/2} \tag{4}$$

where subscript 0 denotes reference values and S_0 is taken as 198.6 for air flow.

The pressure is obtained from the equation of state for a perfect gas:

$$p \;=\; (\gamma - 1)\rho(\epsilon - 0.5 u_j u_j) \tag{5}$$

where $\gamma = C_p/C_v$ is the ratio of specific heats . In the present computations we have taken $\gamma = 1.4$ which comes closest to air.

2.2 Taylor-Galerkin algorithm

As previously mentioned, the solution scheme chosen is the well known Taylor-Galerkin method, successfully developed and used by different authors [1,5].

Non structured meshes of linear triangular and tetrahedral finite elements are used for 2D and 3D computations, repectively [1].

2.2.1 Outline of the algorithm

The solution vector $\mathbf{u}$ is expanded in Taylor series as:

$$\mathbf{\Delta u} \;=\; \mathbf{u}^{n+1} \;-\; \mathbf{u}^n \;=\; \Delta t \frac{\partial \mathbf{u}^n}{\partial t} \;+\; \frac{\Delta t^2}{2}\frac{\partial^2 \mathbf{u}^n}{\partial t^2} \;+\; O(\Delta t^3) \tag{6}$$

where superscript n indicates solution at time $n\Delta t$, etc. Hence using equation (1) and neglecting the $O(\Delta t^3)$ term the above expansion reads:

$$\mathbf{\Delta u} \;=\; \Delta t \left(\frac{\partial \mathbf{g}_j}{\partial x_j} \;-\; \frac{\partial \mathbf{f}_i}{\partial x_i}\right)^n \;-\; \frac{\Delta t^2}{2}\left[\frac{\partial}{\partial x_i}\left(\mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right) \;-\; \frac{\partial}{\partial x_i}\left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right)\right]^n \tag{7}$$

where $\mathbf{A}_i = \partial \mathbf{f}_i / \partial \mathbf{u}$ and $\mathbf{G}_i = \partial \mathbf{g}_i / \partial \mathbf{u}$ are matrices of vector gradients, and where derivatives or products of order higher than two have been neglected.

Equation (7) can be now discretized using C_0 finite elements [5] by making

$$\mathbf{u}^n \;=\; \sum_i N_i \mathbf{a}_i^n \;=\; \mathbf{N}\mathbf{a}^n \tag{8}$$

Where $\mathbf{N}_i$ is the interpolation function of node i and $\mathbf{a}_i^n$ the nodal values of the unknown $\mathbf{u}$ at time n. In the following only linear interpolations will be considered.

Inserting (8) into (7), the integral form of the discretized equation, where standard Galerkin weighting functions are used, may be written by:

$$\int_\Omega \mathbf{N}^T \Delta \mathbf{u} d\Omega \; =$$

$$\Delta t \int_\Omega \mathbf{N}^T \left[\frac{\partial \mathbf{g}_j}{\partial x_j} - \frac{\partial \mathbf{f}_i}{\partial x_i}\right]^n d\Omega - \frac{\Delta t^2}{2} \int_\Omega \mathbf{N}^T \left[\frac{\partial}{\partial x_i}\left(\mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right) - \frac{\partial}{\partial x_i}\left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right)\right]^n d\Omega \tag{9}$$

At this point it should be noted that $\mathbf{A}_i$ and $\mathbf{G}_i$ are matrices of vector gradients that have to be computed and stored (time and memory consuming task).

Integration by parts of (9) using Greens theorem gives:

$$\int_\Omega \mathbf{N}^T \Delta \mathbf{u} d\Omega \; =$$

$$\Delta t \left\{ \int_\Omega \mathbf{N}^T \left(-\frac{\partial \mathbf{f}_i}{\partial x_i}\right) d\Omega - \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_j} \mathbf{g}_j d\Omega + \int_\Gamma \left(\mathbf{N}^T \mathbf{g}_j\right)_p d\Gamma \right\}^n - \tag{10}$$

$$-\frac{\Delta t^2}{2} \left\{ \int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i} \left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k} - \mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right) d\Omega - \int_\Gamma \mathbf{N}^T \left(\mathbf{A}_i \frac{\partial \mathbf{f}_k}{\partial x_k} - \mathbf{G}_i \frac{\partial \mathbf{f}_k}{\partial x_k}\right)_p d\Gamma \right\}^n$$

Subscript p indicates normal projections and superscript n denotes values at time n. The finite element interpolation leads to the following system of algebraic equations:

$$\mathbf{M} \, \Delta \mathbf{u}^n = \mathbf{RHS}^n \tag{11}$$

$$\text{where} \quad \mathbf{M}_{ij} = \int_\Omega \mathbf{N}_i \mathbf{N}_j d\Omega \tag{12}$$

To preserve the explicitness of the algorithm when a steady state solution is to be obtained, equation (11) can be solved using a lumped mass matrix. In other cases, a Jacobi-type iteration procedure may be performed to recover the properties of the full mass matrix [1].

2.2.2 Two-step algorithm

To avoid the computation and storage of $\mathbf{A}_i$ and $\mathbf{G}_i$, an alternative algorithm has been used. First, the products containing $\mathbf{G}_i$ include products of derivatives higher than second order and are therefore neglected. Then, in the first step, a Taylor expansion at time $n + 1/2$ is performed to first order (neglecting at this stage the viscous terms):

$$\mathbf{u}^{n+1/2} \simeq \mathbf{u}^n + \frac{\Delta t}{2}\frac{\partial \mathbf{u}^n}{\partial t} = \mathbf{u}^n - \frac{\Delta t}{2}\frac{\partial \mathbf{f}_i^n}{\partial x_i} \tag{13}$$

From this step an approximation of $\mathbf{u}^{n+1/2}$ is obtained which allows us to estimate $\mathbf{f}_i^{n+1/2}$ to be used in the second step. The use of Taylor expansion for $\mathbf{f}_i^{n+1/2}$ and equation (1) as well as neglecting second order derivatives, provides the tools to replace $\mathbf{A}_i$ by values of advective fluxes at time $n + 1/2$ [1]:

$$\mathbf{f}_i^{n+1/2} \;-\; \mathbf{f}_i^n \;=\; -\frac{\Delta t}{2}\left(\mathbf{A}_i \frac{\partial \mathbf{f}_i}{\partial x_i}\right) \tag{14}$$

Replacing in equation (8) $\mathbf{A}_i$ by the values obtained in equation (14) and integrating by parts the following expression

$$\int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_i}\mathbf{f}_i d\Omega \;=\; -\int_\Omega \mathbf{N}^T \frac{\partial \mathbf{f}_i}{\partial x_i} d\Omega \;+\; \int_\Gamma (\mathbf{N}^T \mathbf{f}_\mathrm{i})_p d\Gamma \tag{15}$$

the term reduces to:

$$\left(\int_\Omega \mathbf{N}^T \mathbf{N} d\Omega\right) \left(\mathbf{u}^{n+1} \;-\; \mathbf{u}^n\right) \;=\; \mathbf{M}\,\Delta \mathbf{u} \;=$$

$$\Delta t \left[\int_\Omega \frac{\partial \mathbf{N}^T}{\partial x_k}\left(\mathbf{f}_i^{n+1/2} - \mathbf{g}_j^n\right) d\Omega - \int_\Gamma \mathbf{N}^T \left(\mathbf{f}_i^{n+1/2} \;-\; \mathbf{g}_j^n\right)_p d\Gamma\right] \tag{16}$$

Subscript i and j indicate mean nodal values and elemental values respectively.

2.2.3 Stability

The stability of the algorithm is preserved if the timestep satisfies the following criterion (for the viscous case) [1]

$$\Delta t_{crit} \;=\; \frac{C h_e}{(|\mathrm{v}| + c) + \frac{4\mu}{\rho_{min} \mathrm{Pr} \mathrm{Re} h_e}} \tag{17}$$

where c the local speed of sound, C is the Courant number, μ is the dynamic viscosity, ρ_{min} is the minimum element density, $\mathbf{v}$ is the velocity vector, and h_e is a mesh spacing value taken to be the minimum of the element. The Courant number is chosen to be less than 1 to ensure the stability of the explicit Taylor-Galerkin algorithm.

In order to enhance the performance of the program and to accelerate the solution towards steady state, local timestepping can be used. This means that the critical timestep for every node is used to advance the solution, instead of applying the smallest time increment all over the domain.

2.2.4 Artificial dissipation

To account for discontinuities in the solution of fluid flow using a high order scheme (e.g. Taylor expansion) a certain kind of artificial viscosity or diffusion must be introduced. First an algorithm of the Lapidus kind [2,3] and then an approach similar to Jameson [4] are described. In the case where pressure differences (especially in the incompressible flow range) are expected to be small, the application of the velocities-based Lapidus-diffusion is recommended and this has been chosen in the example presented in this paper.

a) *Lapidus algorithm*

It can be shown that for the ith node of an unidimensional solution, a second derivative of the unknowns can be obtained as:

$$C\left[\frac{\partial}{\partial x}(h^2\frac{\partial \phi}{\partial x})\right]_i = \left[\mathbf{M}_l^{-1}(\mathbf{M}_c - \mathbf{M}_l)\phi\right]_i \tag{18}$$

$\mathbf{M}_l$ is the diagonalized or lumped mass matrix, $\mathbf{M}_c$ is the consistent mass matrix, C is a constant, and ϕ is the unknown variable.

An extension of this idea to a multidimensional problem gives:

$$\Delta \mathbf{u}_s = \mathbf{u}_s^{n+1} - \mathbf{u}^{n+1} = \Delta t\,\frac{\partial}{\partial \mathbf{l}}\left(k^\star \frac{\partial(\mathbf{v}^{n+1})}{\partial \mathbf{l}}\right) \tag{19}$$

$k^\star$ is a variable which determines the amount of additional viscosity necessary, $\mathbf{l}$ is the normalized vector of velocity gradients, $\Delta \mathbf{u}_s$ are incremental smoothed values, $\mathbf{u}_s^{n+1}$ is the smoothed solution at time n+1, and $\mathbf{v}$ is the velocity vector.

$k^\star$ and $\mathbf{l}$ are obtained as follows:

$$k^\star = C_k(h^{el})^2\left|\frac{\partial(\mathbf{v}\mathbf{l})}{\partial \mathbf{l}}\right| \quad ; \quad C_k : \textit{Lapidus constant} \tag{20}$$

$$\mathbf{l} = \begin{Bmatrix} l_1 \\ l_2 \\ l_3 \end{Bmatrix} = \frac{1}{|\text{grad } \mathbf{v}^2|} \begin{Bmatrix} \text{grad}_1 \mathbf{v}^2 \\ \text{grad}_2 \mathbf{v}^2 \\ \text{grad}_3 \mathbf{v}^2 \end{Bmatrix} \tag{21}$$

b) *2nd and 4th order artificial dissipation*

An alternative to the Lapidus approach is to introduce an artificial dissipation obtained as a blend of second and fourth order terms [4]. This formulation is better suited if stronger shocks and shocks of different strengths appear.

The general expression for the added dissipation flux (which is taken from a finite difference formulation) is as follows:

$$f_d = \alpha_i^{(2)} \Delta \mathbf{u}_i - \alpha_i^{(4)} \Delta^3 \mathbf{u}_i \tag{22}$$

This is the additional term to be added to all the equations with the modification of the energy equation where the enthalpy is constant at steady state. Hence, the last component of $\mathbf{u}$, $\rho\epsilon$, is replaced by the expression ρH.

The two coefficients, $\alpha_i^{(2)}$ and $\alpha_i^{(4)}$, are obtained from the expressions

$$\alpha_i^{(2)} = c^{(2)}(|\mathbf{v}| + c) S_e \tag{23}$$

$$\alpha_i^{(4)} = \max\left[0, c^{(4)} - c^{(2)} S_e\right] \tag{24}$$

and a pressure switched diffusion coefficient for each node which is calculated according to the following term:

$$S_e = \max_{i \epsilon el} \frac{|(\mathbf{M}_c - \mathbf{M}_l)\, p_i|}{\mathbf{M}_c\, p_i} \tag{25}$$

The terms $c^{(2)}$ and $c^{(4)}$ are user defined constants that can be tuned according to the desired amount of diffusion to be added.

2.3 Adaptive Solution

In order to increase the economy of a mesh, adaptive remeshing techniques can be applied. The effect is the addition of points and elements in areas where strong gradients appear and the removal of them in smoother parts of the flow domain. Moreover, the elements can be streched according to a dimensionality criteria which reduces its size in the direction of strong gradients wheras it stretches its length along the least local gradient. The required tool to determine local gradients in the present solution, which consequently provides the input parameters for the next mesh generation process, is the *a posteriori* error estimator [8].

2.3.1 Error estimator

The estimation of the error is based on the following element mean quadratic error:

$$E^{el} = C(h^{el})^2 \left| \frac{\partial^2 \phi}{\partial x^2} \right|_{el} \tag{26}$$

h^{el} is the 1D element length, ϕ is the variable chosen, and C is a constant($C = 1/11$ for linear elements). Note that eq. (26) is only valid for elliptic problems [5], however it has been shown to produce accurate enough results for the kind of problems considered in this paper.

If the criterion of equidistribution of the error among the elements is adopted [5], then:

$$(h^{el})^2 \left| \frac{\partial^2 \mathrm{v}}{\partial x^2} \right|_{el} = \text{constant} \tag{27}$$

For 2D and 3D problems equation (27) may be extended as follows:

$$(\delta_i)^2 |\lambda_i| = \text{constant} \qquad i = 1,3 \tag{28}$$

where δ_i is the corresponding element size and λ_i are the eigenvalues of the Hessian matrix

$$C_{ij} = \frac{\partial^2 \phi}{\partial x_i \partial x_j} \tag{29}$$

This has the advantage of providing the postulated *directionality* to the error estimator. It is a desirable property, since shocks or boundary layers present a strong variation in one direction and smooth behavior in the direction normal to the first. It is then useful, to take advantage of this to produce meshes with *stretched* elements in the mentioned regions. The new element sizes δ_1 and δ_2 in the principal directions are defined according to:

$$(\delta_1)^2 \; |\lambda_1| = (\delta_2)^2 \; |\lambda_2| = \delta_r \; \lambda_{\max} \tag{30}$$

λ_1 and λ_2 being the eigenvalues of matrix $\mathbf{C}$, δ_r is a user specified constant (size) and λ_{max} is the maximum eigenvalue over the whole mesh.

In order to perform the error estimation a scalar vairable of refinement must be chosen by the user which is introduced as ϕ. Due to the vector character of $\mathbf{u}$, only the modulus of $\mathbf{u}$ is suited for the error estimation. The other implemented refinement variables are the Mach number, pressure, density and total energy. In the test cases presented in this study the Mach number, has been used as the error variable for our supersonic studies and the velocity modulus for the subsonic simulations. These variables are reasonable because a refinement in the region of shock or near the boundary is desired where the gradient of the Mach number and velocity modulus is obvious.

2.3.2 Mesh generation

A finite element mesh of linear tetrahedres of 4 nodes is generated using a Delauney technique [9]. The element sizes are decided in accordance with the sizes specified by a background grid. The initial background grid may be a very simple mesh, and the subsequent grids are those used for the latest or seemingly best computation, where the sizes are decided as specified in the previous paragraph.

For supersonic computations, three or four remeshings are normally enough to obtain a good solution (for subsonic simulations even less remeshings are necessary) and the process of remeshing is stopped when the user reaches a level of definition that is considered appropriate.

3. GEOMETRY DEFINITION

In the particular case of the SEAT Toledo modelization (courtesy of SEAT S.A.), the following systems were involved:

- PATRAN V2.5, PDA Engineering, PATRAN Division [11]
- CIMNE-CatAr, CIMNE Barcelona, España [12]

The original definition of the surfaces was not yet adequate for the meshing process (see Figure 1). This circumstance made a "mapping process" necessary which defined additional entities (lines and surfaces) in order to get a definition of the geometrical model and of adequate topology. The general criteria employed fixed grade entities (of cubic order), while the accuracy was controlled by changing the amount of necessary entities to approximate each original entity. The new geometrical entities were obtained from PATRAN and from the design and mesh generation process performed by the program CIMNE-CatAr. The transformation of the information between the two systems was done by means of ASCII files in VDA format (V2.5 [15]). In the following section, general aspects of the analytical definition of the implemented geometrical entities are described.

3.1 Analytical definition of the geometrical entities

The representation of the geometry of a design in the context of CAD is accomplished using geometrical entities which are of polinomial nature with the exception of the single point. The order of these polinomials is only limited by practical considerations (commonly not superior to 8 or 9 in industrial applications). Some CAD packages fix the order, ie. 3, thus obtaining greater precision by increasing the number of geometrical entities.

In the case of curves in the cartesian space, a biyective transformation can be defined by a single parameter, for instance, the common expression for a cubic parametric curve would be

$$\mathbf{Z}(\xi) = \mathbf{S}_1\xi^3 + \mathbf{S}_2\xi^2 + \mathbf{S}_3\xi + \mathbf{S}_4 \qquad 0 \le \xi \le 1 \tag{31}$$

or in matrix form:

$$\mathbf{Z}(\xi) = [\xi^3 \; \xi^2 \; \xi \; 1] \begin{bmatrix} \mathbf{S}_1 \\ \mathbf{S}_2 \\ \mathbf{S}_3 \\ \mathbf{S}_4 \end{bmatrix} \tag{32}$$

where $\mathbf{Z}$ includes coordinates of an n-dimensional space; in the case n=3, $\mathbf{Z}$ represents the cartesian coordinates x, y, z. In addition, it is common to choose the variable ξ between 0 and 1.

The derivative of equation (32) with respect to ξ reads:

$$\frac{dZ}{d\xi} = 3S_1\xi^2 + 2S_2\xi + S_3 \tag{33}$$

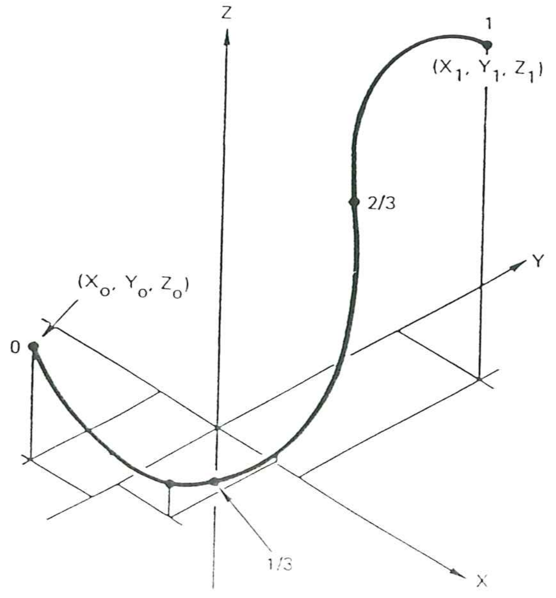
or:

$$\frac{dZ}{d\xi} = [3\xi^2 \; 2\xi \; 1 \; 0] \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \tag{34}$$

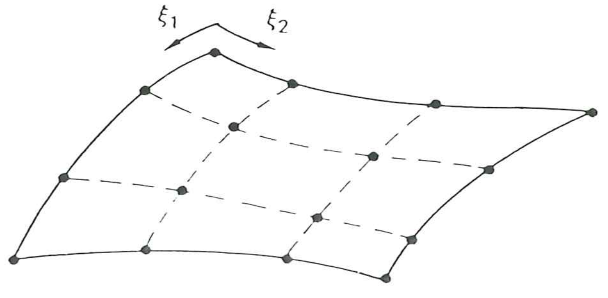
From these equations, we obtain [12]

$$\mathbf{Z}(\xi) = [F_1(\xi) \; F_2(\xi) \; F_3(\xi) \; F_4(\xi)] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{11}{2} & 9 & -\frac{9}{2} & 1 \\ -1 & \frac{9}{2} & -9 & \frac{11}{2} \end{bmatrix} \begin{bmatrix} Z(0) \\ Z(\frac{1}{3}) \\ Z(\frac{2}{3}) \\ Z(1) \end{bmatrix} \tag{35}$$

Once the coordinates of the four points are given, the unique definition of the curve becomes evident, shown in drawing a). In the same manner arbitrary surfaces are defined, even though two parameters are now necessary, shown in drawing b).



drawing a)



drawing b)

The F_i's are given by:

$$F_1(\xi) = 2\xi^3 - 3\xi^2 + 1 \qquad F_2(\xi) = -2\xi^3 + 3\xi^2$$

$$F_3(\xi) = \xi^3 - 2\xi^2 + \xi \qquad F_4(\xi) = \xi^3 - \xi^2$$

If we continue that approach it is possible to reproduce the corresponding transformation of the target surface in the **Z** space. Consider, for example, a bicubic surface; we can express it by:

$$\mathbf{Z}(\xi_1,\xi_2) = [\xi_1^3 \ \xi_1^2 \ \xi_1 \ 1] \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} & \mathbf{S}_{13} & \mathbf{S}_{14} \\ \mathbf{S}_{21} & \mathbf{S}_{22} & \mathbf{S}_{23} & \mathbf{S}_{24} \\ \mathbf{S}_{31} & \mathbf{S}_{32} & \mathbf{S}_{33} & \mathbf{S}_{34} \\ \mathbf{S}_{41} & \mathbf{S}_{42} & \mathbf{S}_{43} & \mathbf{S}_{44} \end{bmatrix} \begin{bmatrix} \xi_2^3 \\ \xi_2^2 \\ \xi_2 \\ 1 \end{bmatrix} \tag{36}$$

or:

$$\mathbf{Z}(\xi_1,\xi_2) = \sum_{i=1}^{4}\sum_{j=1}^{4} \mathbf{S}_{ij}\, \xi_1^{4-i}\, \xi_2^{4-j} \tag{37}$$

Similar considerations prove that a bicubic surface is uniquely defined by 16 points or four lines in the direction of ξ_1 or four lines in the direction of ξ_2.

In summary, in order to reproduce each entity of a CAD-definition, it is enough to transfer the order and coefficients of its analytic representation. The above representation uses cubic lines and surfaces to approximate the original geometrical model which are used as the input for the mesh generation process.

3.2 Design and Mesh generation

As mentioned earlier, the mesh generation process produces an unstructured tetrahedral mesh of linear 4 noded elements. The unstructured characteristic leads to the fact that the number of elements per node is not equal at every node, which makes it possible to discretize complex geometries at higher precision and economy. The following is a brief description of the process followed to generate the used meshes [14].

The control of the size and form of the elements is performed via a background mesh. The geometric entities which are to be triangulated are of the same nature as those defining the domain. In addition to the lines and patches, hyperpatches are used. Hyperpatches are a cluster of many patches (which can be observed in Figure 2), that define the background mesh (in this case the back spoiler of the car). A more complete description of the meshing process can be obtained from reference [13].

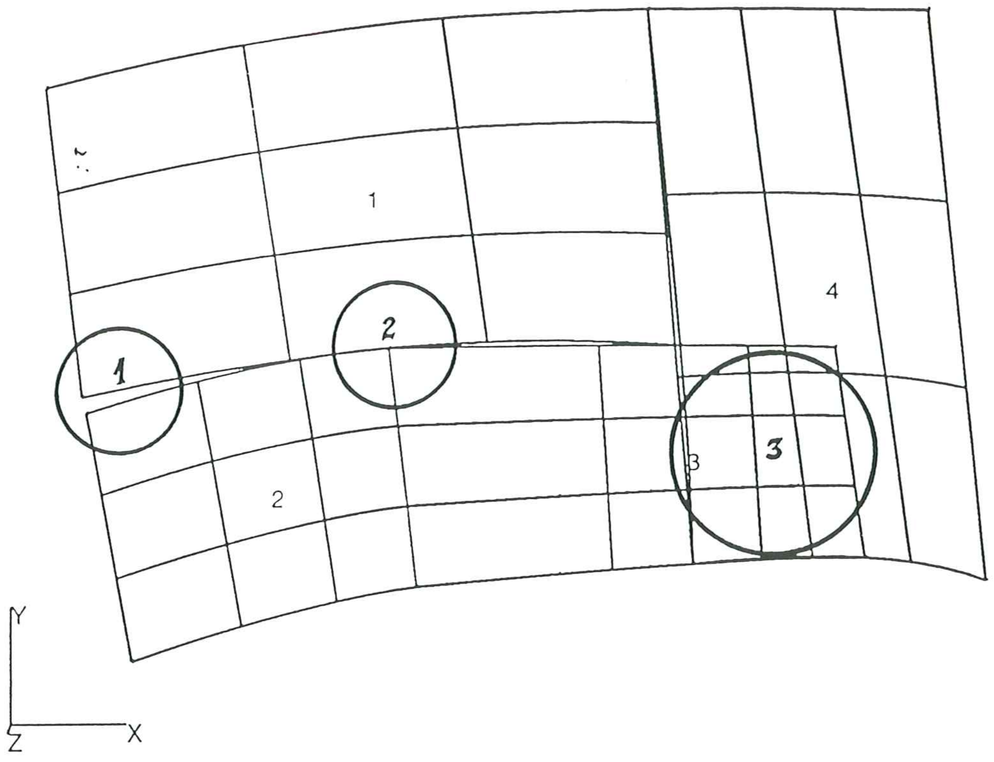


Figure 1: Typical singularities in the representation of CAD-definitions.

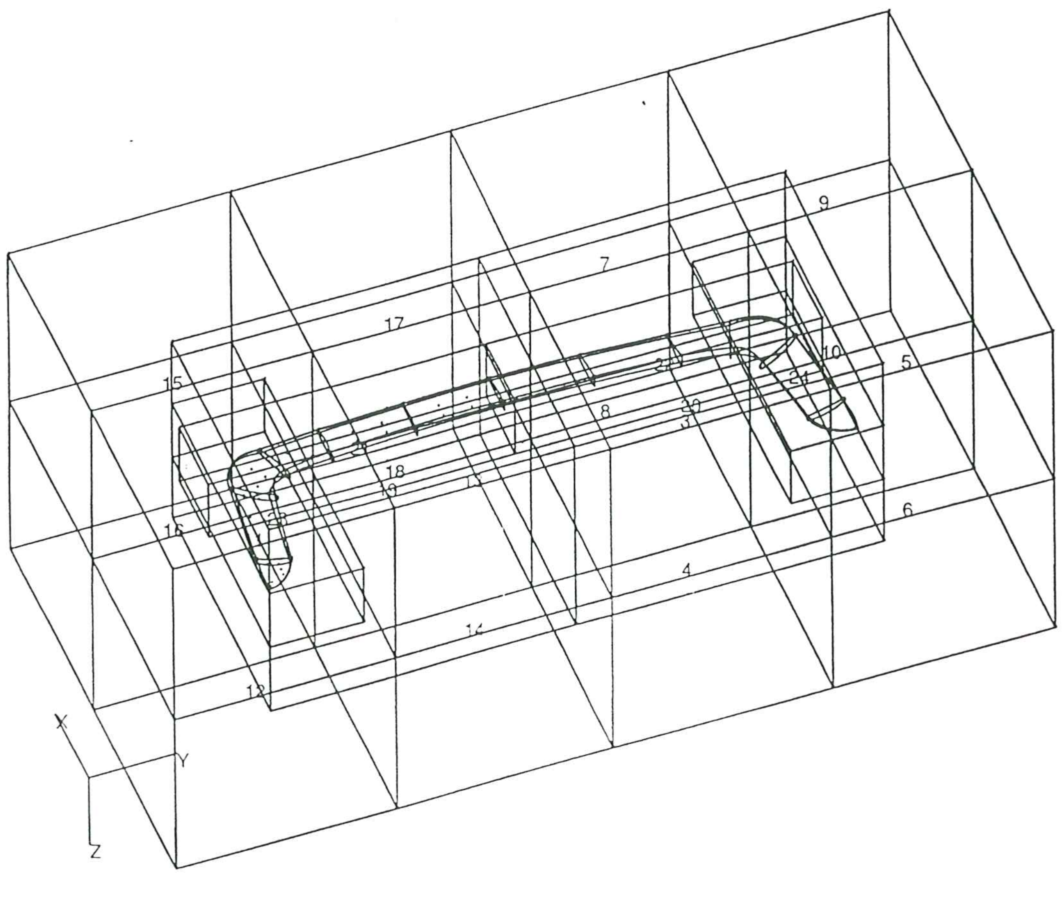


Figure 2: Example of hyperpatches used to generate the background mesh of the rear spoiler belonging to the SEAT Toledo.

4. RESULTS

Basically, the scheme of aerodynamical computations followed can be divided into four steps:

a) pre-processing (mesh generation),

b) numerical finite element solution,

c) post-processing and visualization,

The current results were obtained on a Silicon Graphics workstation of 34 Megabytes main storage.

Mesh Generation

Mesh Quality

The mesh generation process is an essential and crucial step in the course of numerical fluid flow simulation. The quality of the results depends highly on the quality of the mesh. The structure and shape of the elements determines the quality of the mesh, in particular flat elements tend to distort mesh quality characteristics. To guarantee a better mesh quality, we have adopted several means to measure and improve the mesh. For instance, near the boundary flat elements seem to occur with a length-height ratio of 1000 and greater. Since the timestep of each iteration is proportional to the minimal height of the element, very small timesteps result compared to neighboring elements. This leads to unbalanced and ineffective iterations and even in wrong results. We avoid these shortcomings by redefinition of elements, movement of points and smoothing so that the mesh quality is improved but the geometry is not affected.

Final Mesh

The final mesh consisted of ca. 215 109 linear tetrahedral elements and 40 638 points. It is divided into a symmetrical part where the wheels and tires are rotating artificially on a moving ground simulation. This provides a most realistic simulation and avoids numerical uncertainties for connecting boundaries. The inflow and outflow boundaries are placed far away (greater than 10 chords of the vehicle) in order to avoid circulative lift influences on the object. The infinity inflow velocity is taken to be Mach 0.1 (approximately 120 km/h), the inflow temperature is 20 degrees celsius and the pressure is 1013 mbar.

Figure 3 shows the basic geometry provided by SEAT as a CAD file. Various patches can be determined forming the outer shell of the car domain. As mentioned earlier, this information is transformed by means of an intermediate program such that a tetrahedral mesh generator can read its data. Through subsequent visualization, with the help of Flavia 3D [10], the regenerated geometry of the vehicle can be seen in Figures 4, 5 and 6. In Figure 4 the final surface mesh of the SEAT Toledo is shown. In Figures 5 and 6, a light direction vector highlights the three dimensional aspects of the boundaries, so that its recognition is easier and its shape can be appreciated.

From the CAD patches, a boundary grid is generated according to the information given in the data files. Figures 7 and 8 show some of the boundaries as a result of this. The different faces are distinguished by different colors. Figure 7 demonstrates the extent of the complete fluid flow domain chosen to avoid the influence of circulation. Figure 8 shows details of the surface near the car. It can be seen that the wheels are in different color than the rest of the car, so as to simulate their rotation. The same is true for the ground, with prescribed velocity values so as to simulate its movement.

Finally, the postprocessing capabilitites allow to capture details of the flow domain by "cutting" the mesh and plotting contours across the cut. An example of a mesh cut can be appreciated in Figure 9.

Solution

Having performed the discretization process, the results of the solver are documented in form of variables at the nodes, as well as convergence history of the residuals, the drag and lift coefficients. Figure 10 shows that the reductions in the error were greater than three orders of magnitude, thus indicating that a global steady state has been reached. Note, however, that this problem is of transitory nature because of expected turbulence and other instationary phonomena that were not captured due to the relative coarse mesh.

The following plot (Figure 11) presents the evolution of the drag coefficient, reflecting a target around 0.28 after 17000 iterations. The history of the lift coefficient (Figure 12) reflects a value of cl=0.07 after 17000 iterations. These values will naturally improve with a finer remeshing. The size of the elements near the boundary is generally greater or equal to the expected boundary layer. In order to better estimate the viscous effects the size of boundary layer elements should at least be smaller than 1/10 of the expected boundary layer. Thus, these values can only give a global order of magnitude.

Visualization

The process of visualization is an important feature in the world of numerical simulation. It provides interactive access to the obtained results. The function of postprocessing is to give the engineer the necessary access to both global and detailed analysis. For this purpose a visualizer for 3D flow computations named FLAVIA has been developed at CIMNE [10]. Here we can only give a few examples for demonstration. Figure 13 displays a general picture of the distribution of velocities showing the viscous behaviour (recirculation) behind the vehicle; Figure 14 shows streamlines together with the car shape. The recirculation phenomenon can be observed in more detail in Figure 15 using velocity vectors for this purpose. Also the pressure distribution is shown in a detail shot in front of the Toledo (Figure 16). The highest pressures are observed in front of the vehicle, at the tires and on the windshield.

5. CONCLUSIONS

We have presented an efficient and versatile fluid flow solver which can meet the demands of 3D aerodynamical simulation in automobile and similar moving vehicles. Complex geometries should not be limited by the enormous time and memory requirements of other popular approaches to the solution of viscous flow. Our explicit scheme does not need to store huge matrices nor does it require their setup and calculation, without sacrificing accuracy. The scheme remains second order accurate.

The presented results are more than satisfactory considering the simplicity of the mesh used. Currently, we are aiming to improve both the mesh quality and efficiency, using adaptability and increasing the points close to the surface. In addition, our solver is being further developed, so that it is can efficiently run in a parallel computing environment.

6. ACKNOWLEDGMENTS

The authors thankfully acknowledge the support received from SEAT in providing the automobile geometry for the example presented in this report.

7. REFERENCES

1.-Peraire, J. *A finite element method for convective dominated flows.* PhD Thesis. University College of Swansea, 1986

2.-Lapidus, A. *A Detached Shock Calculation by Second Order Finite Differences.* J. of Computational Physics 2, 154-177, 1967

3.-Morgan, K., Löhner R. and Peraire J. *A simple extension to multidimensional problems of the artificial viscosity due to Lapidus*, Communications in Applied Numerical Methods, 1985

4.-Jameson A. *Transonic aerofoil calculations using the Euler equations*, Numerical Methods in Aeronautical Fluid Dynamics, 1982

5.-Zienkiewicz O.C. and Taylor R.L. *The Finite Element Method*, 4th Edition, Volume 2, Mc. Graw Hill, 1991

6.-Quintana F. *Finite element solution of high speed compressible flows*, PhD Thesis, Univ. Politécnica de Cataluña, 1992

7.-Fischer T. *Finite element analysis of compressible supersonic and subsonic flow problems in two dimensions*, Diploma Thesis, CIMNE, Barcelona, 1991

8.-Oñate, E. *Error estimations and adaptive refinement techniques for structural and fluid flow problems*, Mecánica Computational Vol. 11, S. Idelsohn, Asociación Argentina de Mecánica Computational, 1991

9.-BM3D, *A 3D Pre-Processor, Version 2*, Reference Manual, Computational Dynamics Research Innovation Centre; University College Swansea

10.-Bugeda, G. *FLAVIA. Graphic program for flow visualization using FEM*, CIMNE Report, 1992

11.-PDA engineering, PATRAN Plus Release 2.4, User manual, 1992

12.-CIMNE, *CatAr*, Versión 1.0, CIMNE Barcelona, 1992

13.-Peiro, J., *A Finite Element Procedure for the Solution of the Euler Equations on Unstructured Meshes*, PhD Thesis, University College of Swansea, 1989

14.- Fruitós, O., Bugeda, G., Kreiner, R., Oñate, E., *Sistema de adquisición/ tratamiento de datos y generación de mallas de elementos finitos a partir diseños CAD*, CIMNE Report, 1993

15.-Guideline VDA Surface Interface Version 2.0, VDA CAD/CAM Committee, VDA Working Group "Geometrical Interface"

16.-Oñate, E., Quintana, F., Codina, R., Miquel, J. *Finite element formulations for incompressible and compressible flows*, CIMNE Report, 1991

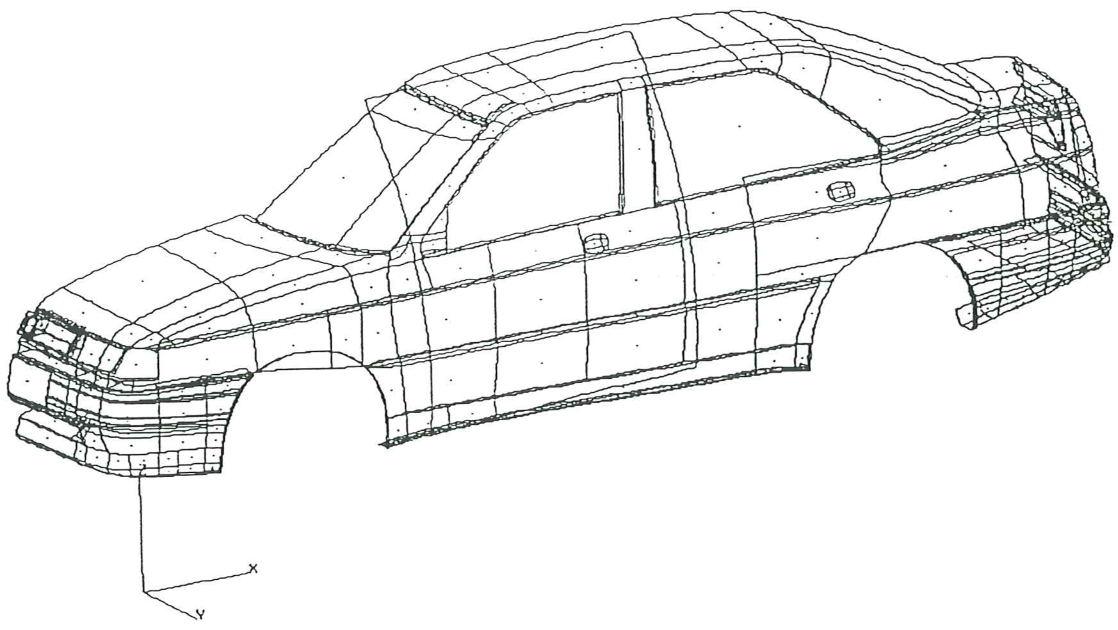


Figure 3: CAD geometry of the Toledo kindly provided by SEAT.

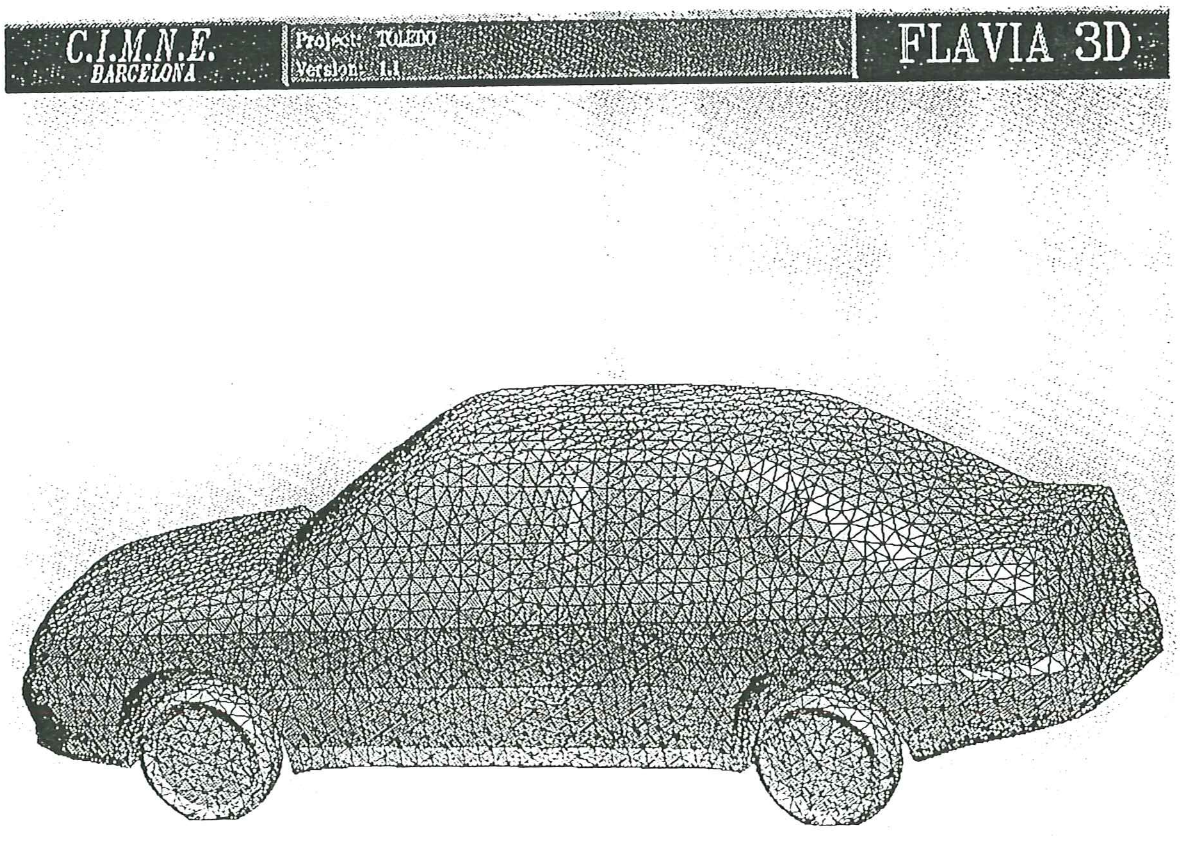


Figure 4: Finite element surface mesh of the car geometry, generated from the CAD-definition.

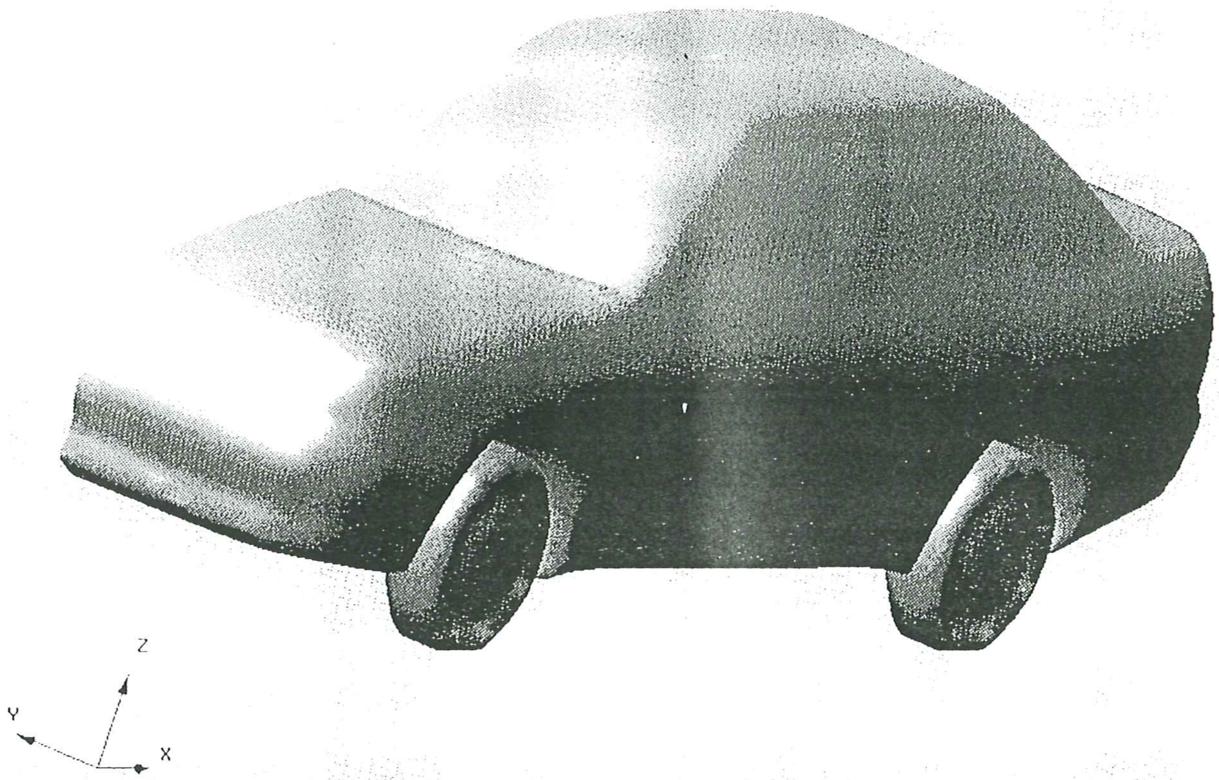


Figure 5: Visualization through FLAVIA postprocessing software [10] showing the front part of the car by means of different gray shadings.

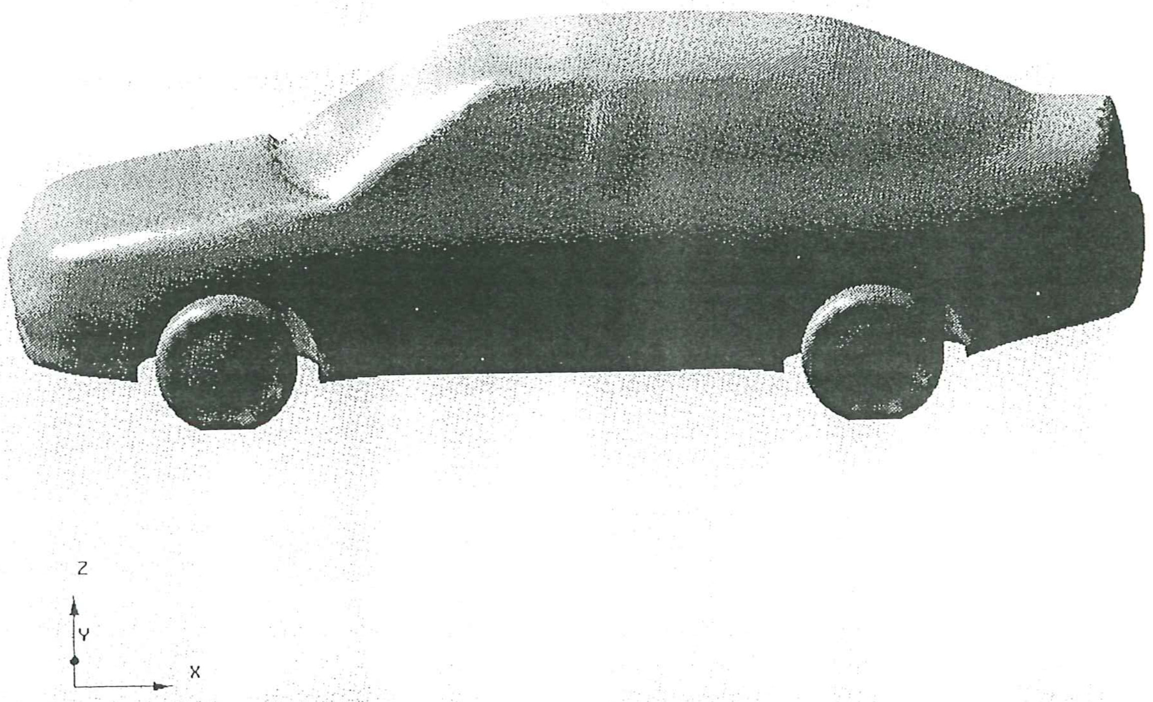


Figure 6: Visualization through FLAVIA postprocessing software [10] in a side view of the car by means of different gray shadings.

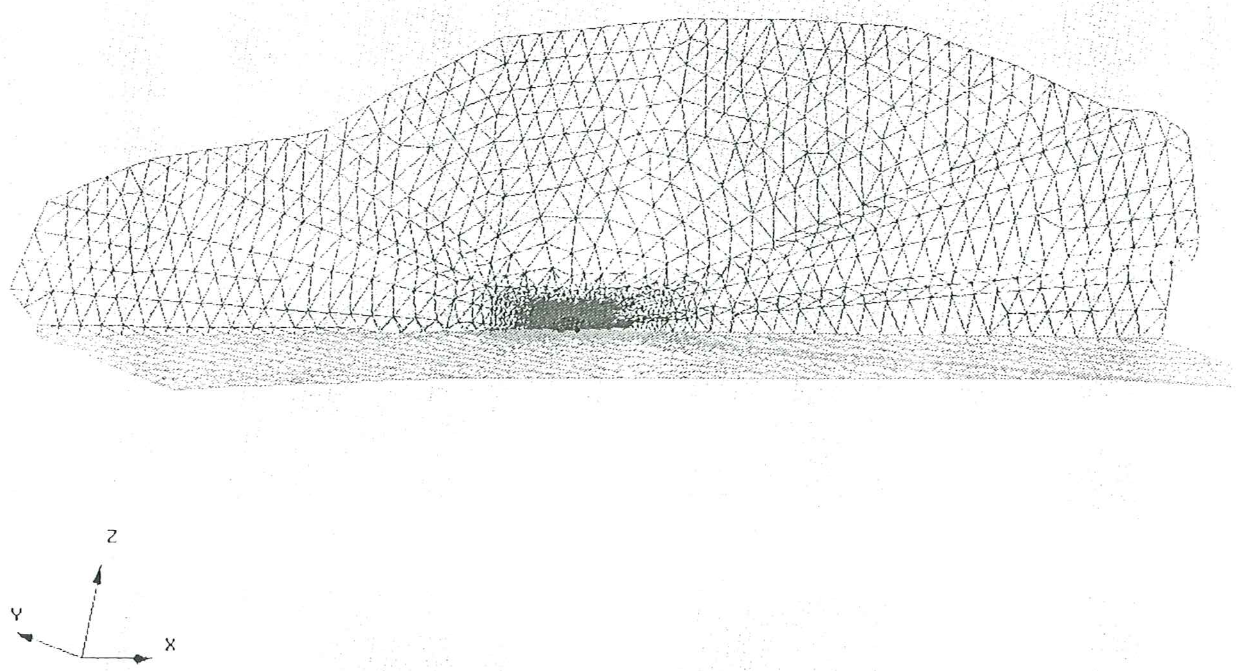


Figure 7: Total dicretized flow domain.

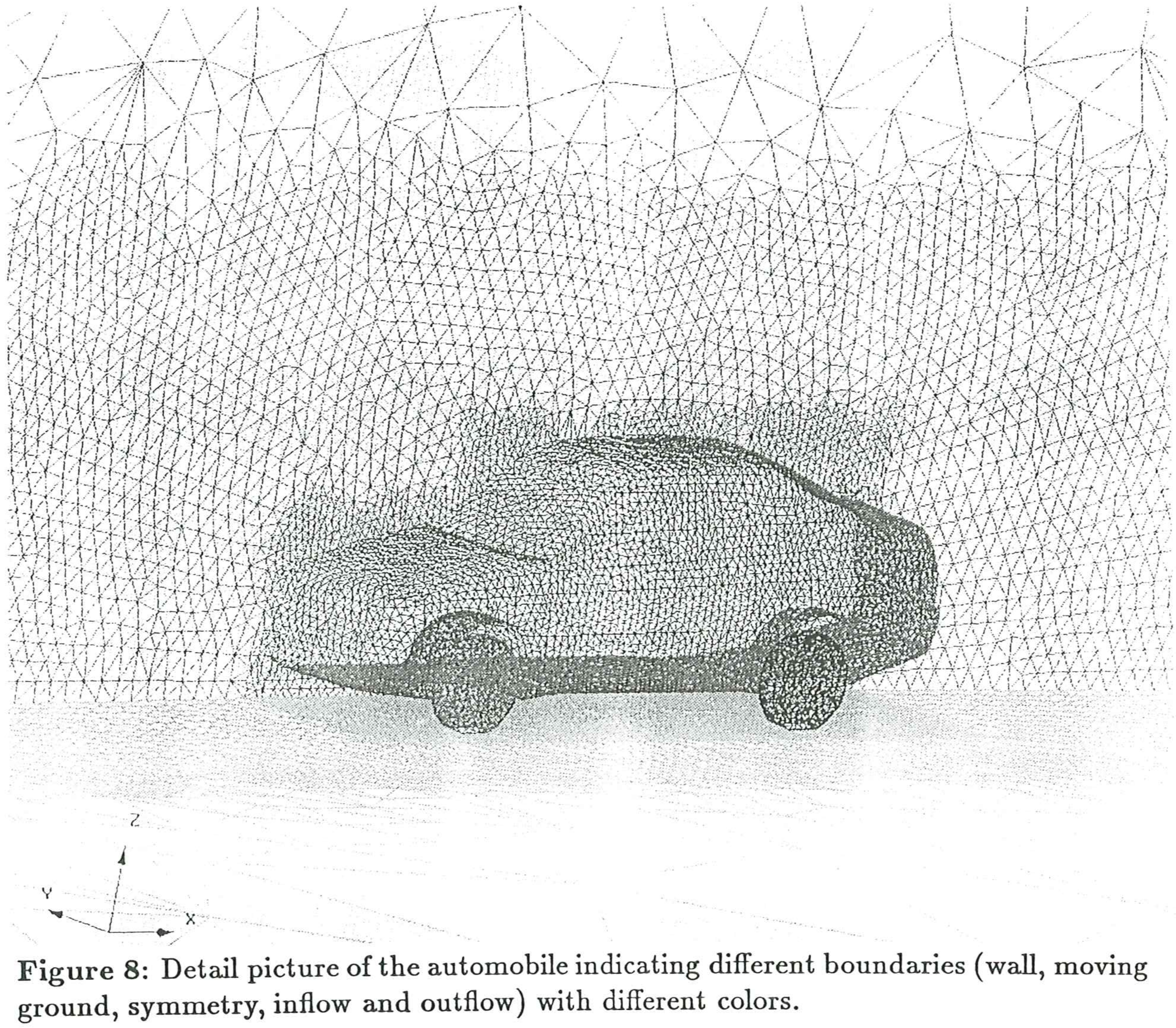


Figure 8: Detail picture of the automobile indicating different boundaries (wall, moving ground, symmetry, inflow and outflow) with different colors.

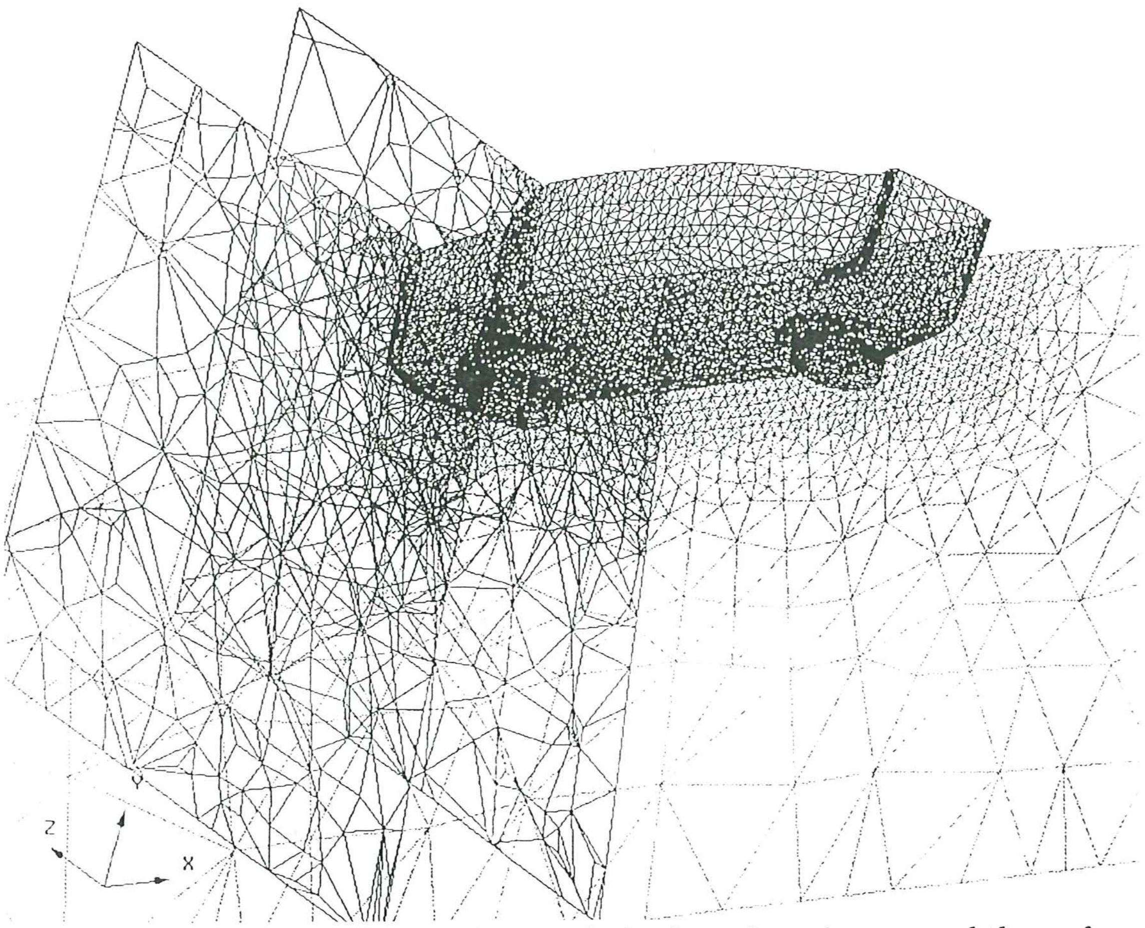


Figure 9: **3D mesh cuts allowing closer analysis of certain regions around the surface.**

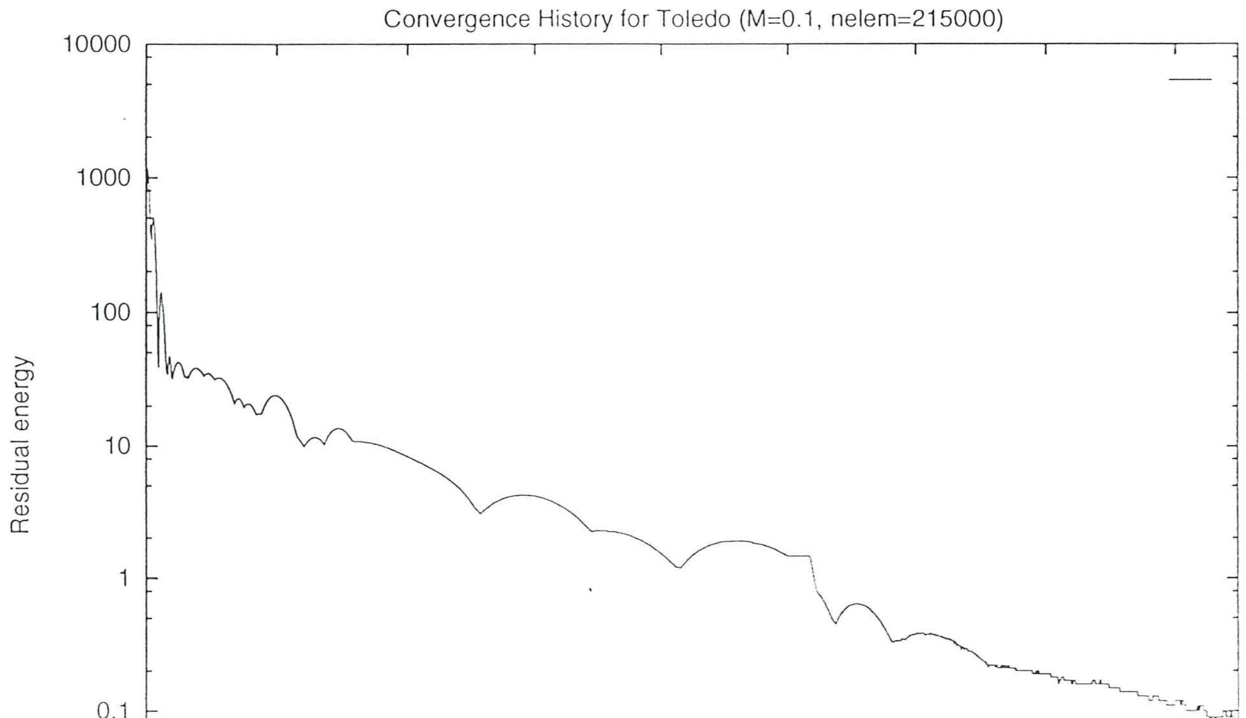


Figure 10: Convergence history to three orders of magnitude.

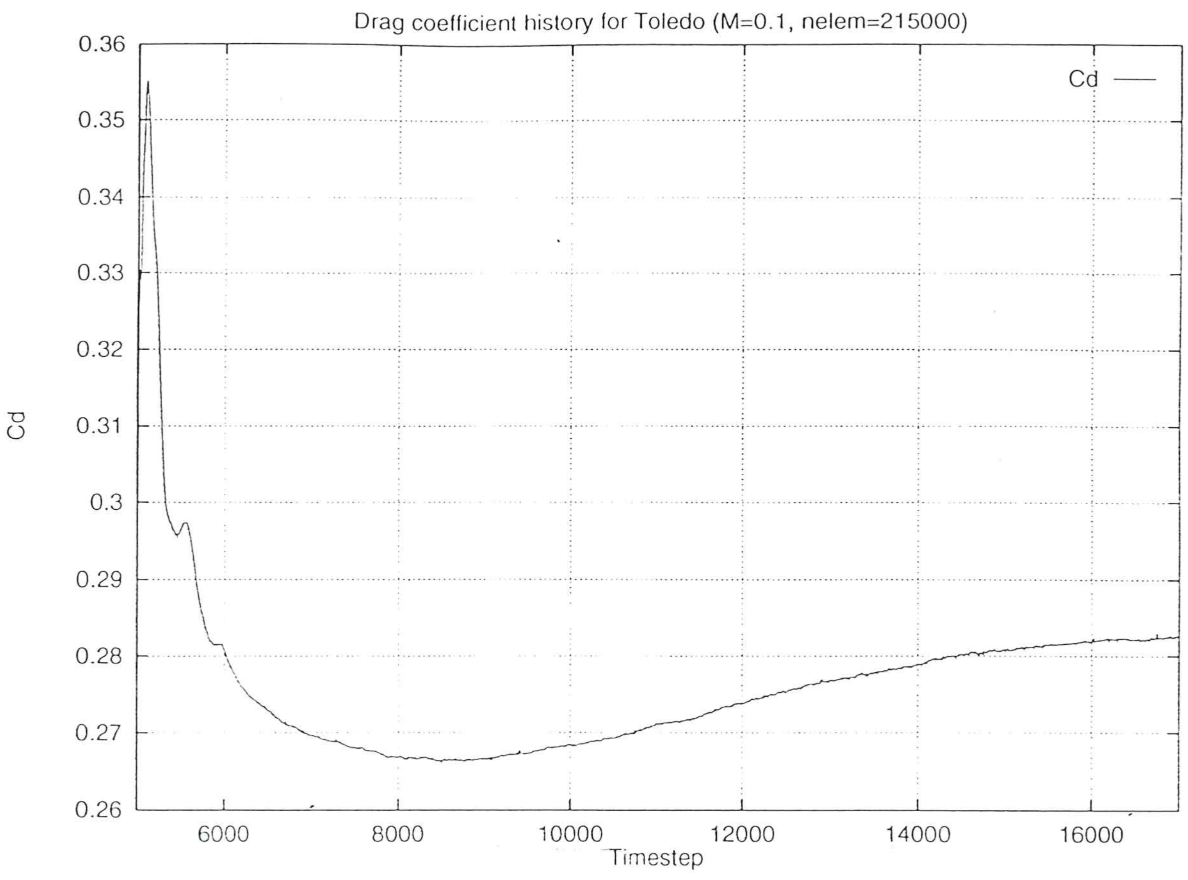


Figure 11: Drag coeffient history with a target c_d of about 0.28.

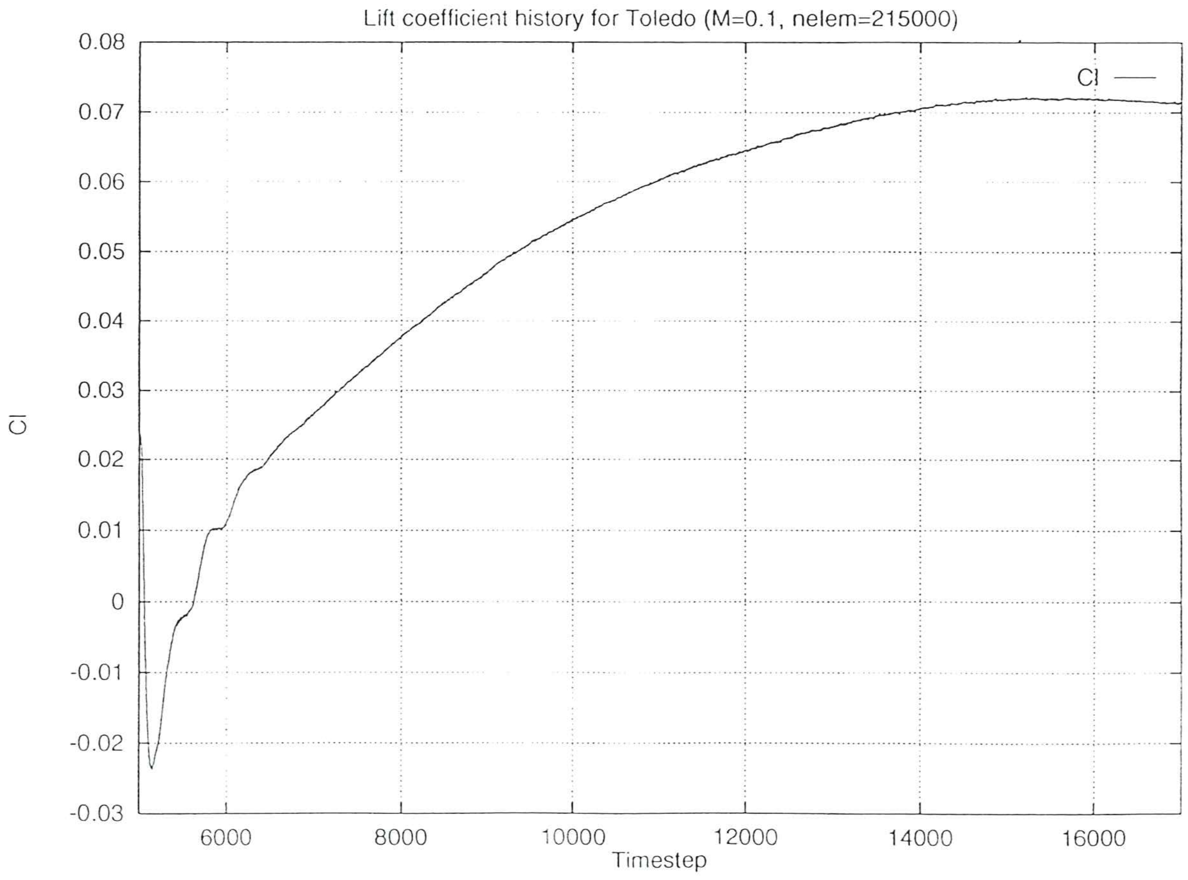


Figure 12: Lift coeffient history with a target c_l of about 0.07.

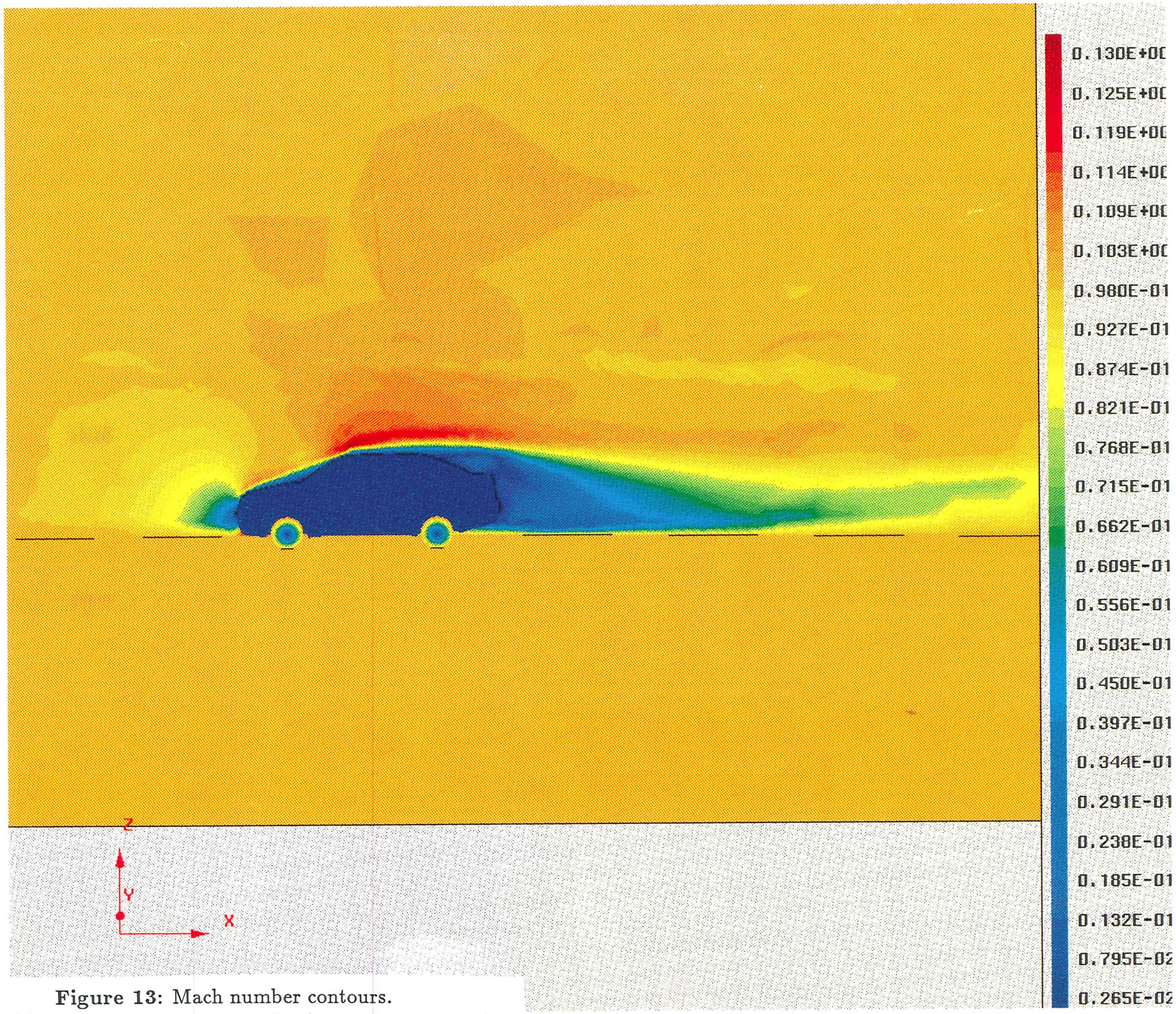


Figure 13: Mach number contours.

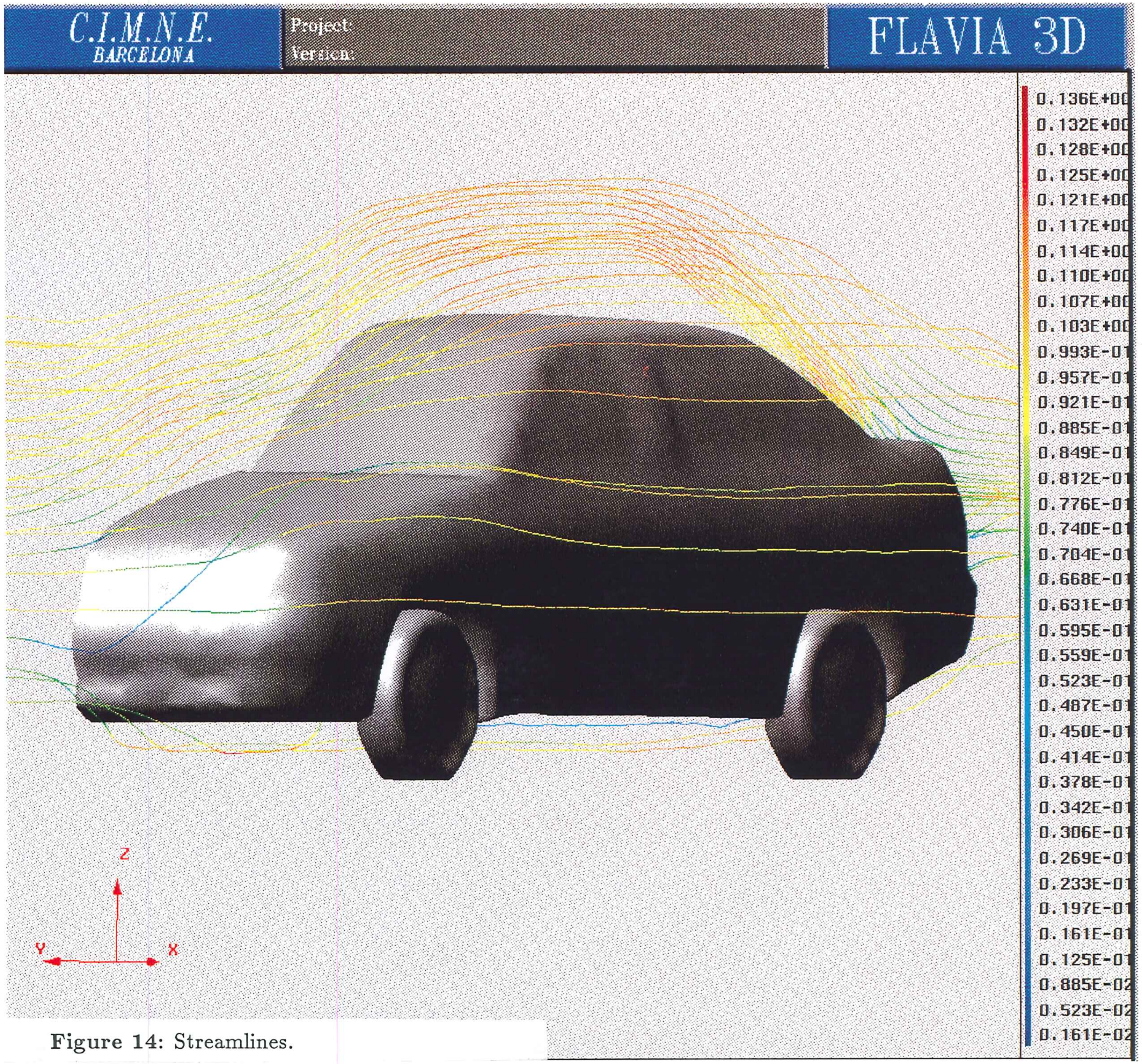


Figure 14: Streamlines.

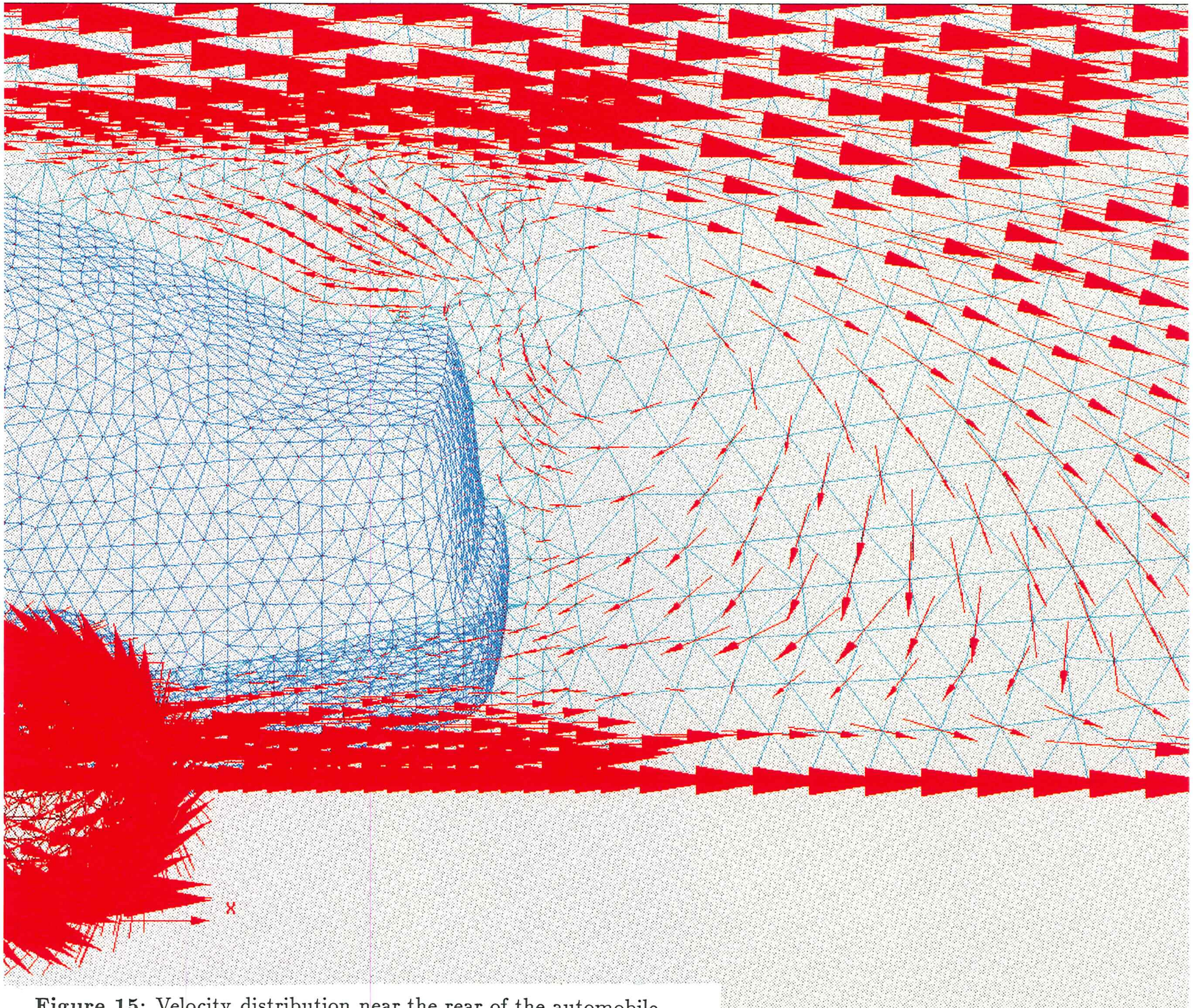


Figure 15: Velocity distribution near the rear of the automobile.

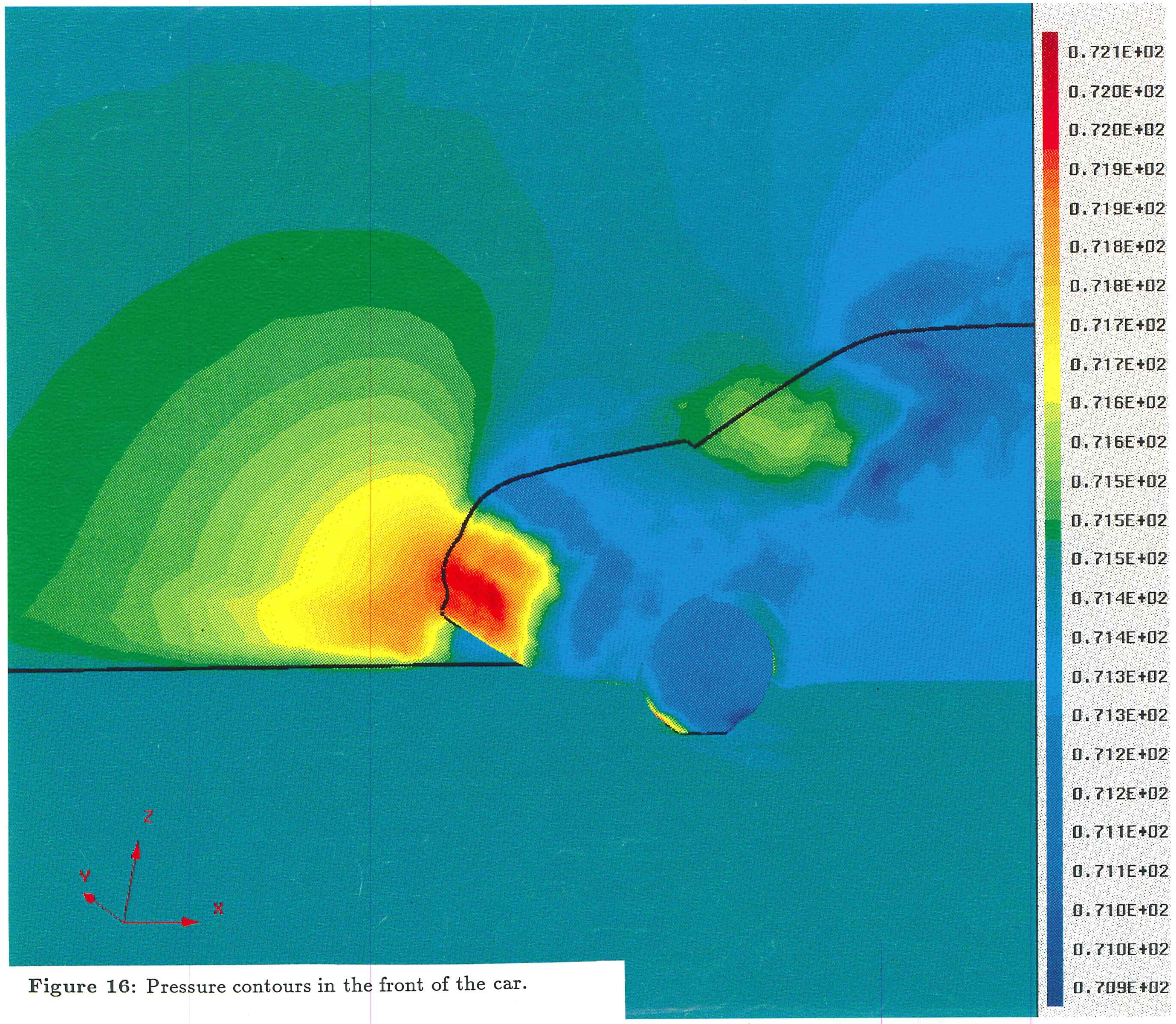


Figure 16: Pressure contours in the front of the car.