Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 195 (2006) 3168–3180
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Fast and accurate simulations of air-cooled structures

Cécile Dobrzynski [a], Pascal J. Frey [a], Bijan Mohammadi [b], Olivier Pironneau [a,*]

[a] *Laboratoire Jacques-Louis Lions, Université Paris VI, Place Jussieu, F-75006 Paris, France*
[b] *Université Montpellier II, Math. Dept., CC51, 34095 Montpellier Cedex 5, France*

Received 3 February 2004; received in revised form 17 February 2005; accepted 22 March 2005

Abstract

There are fields of engineering for which the CAD-based Navier–Stokes solvers are too expensive; architecture and medicine for blood flows are two such examples. Mesh generation and adaptation is also a bottleneck because the users are not expected to have the know-how.

We report here on a Navier–Stokes solver for incompressible temperature and time dependent flows dedicated to architectural applications. The building blocks are not new: a finite element method with time implicit pressure projection steps and mesh adaptativity; but putting them together in an easy to use and efficient 3D code is the challenge which motivates this paper.

For non-engineering applications the user interface is a big problem. In an earlier attempt we designed `freefem3D` based on a fictitious domain discretization, thus avoiding boundary fitted mesh. However it turned out that the display of the solutions requires a boundary fitted mesh; it is possible to generate a feasible surface mesh for graphics but it is much more difficult to generate a feasible surface mesh for FEM.

In this project the user interface is taken from `freefem3d`; then, with a marching cube algorithm we produce a graphic only feasible mesh; finally a surface mesh, adapted to a FEM computation is constructed with an adaptation module and the result is used as input to a Delaunay volumic mesh generator. The solver is optimized and parallelized, all modules are the authors' work.

Three applications are presented, for which the data preparation takes less than a day and results are obtained overnight on a PC cluster. One of the application is presented in details; it is an air cooling system for a canister containing radio-nucleides.

Keywords: Air-cooled structures; Navier–Stokes solvers

* Corresponding author.
E-mail addresses: dobrzyns@ann.jussieu.fr (C. Dobrzynski), pironneau@ann.jussieu.fr (O. Pironneau).

0045-7825/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2005.03.008

1. Introduction

Air flow studies for buildings and chambers require, in principle, the numerical simulation of the Navier–Stokes equations with varying temperature and turbulence models. Some of these problems are relevant to architecture for instance and it is rare that architects have the means to make computer intensive simulations. For instance the sizing of an air conditioner is done by an empirical formula which gives a rough estimate of all the energy loss as a function of the size of the windows, size of the room, wall material etc. Conversely numerical simulations subcontracted by architects do not encourage them to generalize the practice because they require CAD systems and a steep learning curve for the software. We are aware of two such cases, one for the Millau bridge where the aerodynamics (the bridge is exposed to the strong "Mistral" wind) was sub-contracted to us by a company which does not normally perform such air flow simulations and in nuclear engineering for a prototype building for the cooling of radioactive waste. The situation is similar in medicine where cardiologists do not want to become experts in grid generation etc.

The problem considered here involves air at temperatures between $-20°$ and $100°$; ambient air is an incompressible Newtonian fluid with temperature dependent density. So the flow is governed by the Navier–Stokes equations for incompressible viscous flows coupled with the temperature equation (see [18]) by the Boussinesq approximation for the buoyancy terms. The k–ϵ model with wall laws is used for turbulence [17] but we do not report on it here because the meshes compatible with overnight calculations are still too coarse for meaningful 3D time dependent turbulent results and it is effective only in 2D at the present time [28].

A semi-implicit fractional time-step method is used with some upwinding with SUPG. The finite element method is used with P^1 approximations in space for all variables (see [6,9] for instance). Most of the computing time is spent on the Poisson equation for the pressure at each time step. Parallel computing (via MPI) is used for speed up, at the level of linear algebra for the matrix-vector multiplications occurring in the conjugate gradient steps for the linear system solvers.

For these incompressible flows finite volume methods are not superior to finite element methods. Even though no sophistication is built in the solver for pressure/velocity compatibility, the real problem is one of accuracy and multiple scales and so mesh adaptivity preferably non-isotropic is essential [7].

The originality of this study lies in the careful computer implementation into a single package containing a small CSG CAD equivalent system for the geometry using the data structures of VR [29], an automatic surface mesh generator from a background fictitious domain grid, a surface mesh adaptation module [23] and a Delaunay automatic triangulation program by George [13,4,14].

Because every step was optimized, the result is a state of the art package capable of challenging the best CAD integrated commercial package in terms of user comfort and computing speed. The figures show what can be done with an inexpensive computer equipment overnight (less than 16 pc with 2 Gig RAM on each board and linked by 1 Gbit ethernet).

2. Problem statement

Let $\Omega \subset \mathbb{R}^3$ be the region occupied by air. The flow velocity u, density ρ, temperature θ and pressure p are governed by Navier–Stokes equations for compressible viscous flow. The law of perfect gas $p = \rho R\theta$ gives an inverse law in terms of the temperature θ at constant pressure. The Boussinesq approximation to Navier–Stokes equations assumes that θ varies little around a mean temperature θ_0 and its effect is felt only in the term $g\rho$. Hence one may divide the momentum equation for u by ρ and obtain

$$\frac{\partial u}{\partial t} + u \cdot \nabla u + \nabla p - \nabla \cdot (v(\nabla u + \nabla u^{\mathrm{T}})) = -g\frac{\rho}{\rho_0}, \quad \nabla \cdot u = 0 \text{ in } \Omega \times (0, T), \tag{1}$$

where v is the kinematic viscosity and p the reduced pressure.

To evaluate the buoyancy term $g\rho$ the Boussinesq approximation gives

$$\theta = \theta_0 + \theta', |\theta'| \ll |\theta_0| \Rightarrow \rho = \frac{p_0}{R(\theta_0 + \theta')} \sim \frac{p_0}{R\theta_0}\left(1 - \frac{\theta'}{\theta_0}\right).$$

This can be re-written equivalently as $\rho - \rho_0 = -\alpha\rho_0(\theta - \theta_0)$ where α is the coefficient of volume expansion. Therefore the variation in density is neglected everywhere except in the buoyancy term which is a linear function of temperature θ. Notice that $-g(\rho_0 + \alpha\rho_0\theta_0)/\rho_0$ is the gradient of $g(1 + \alpha\theta_0)z$ when θ_0 is constant, so that this term can be absorbed into the pressure and finally the term is functionally equivalent to $-\rho g/\rho_0 \sim g\alpha\theta$.

The temperature equation is

$$\frac{\partial\theta}{\partial t} + u\nabla\theta - \kappa\Delta\theta = 0 \quad \text{in } \Omega, \tag{2}$$

where κ is given with initial conditions (θ_0) and either Dirichlet boundary conditions $\theta = \theta_0$ on Γ_1 or Fourier conditions:

$$\frac{\partial\theta}{\partial n} + a\theta + b(\theta^4 - \theta_e^4) = 0 \quad \text{on } \Gamma_2, \tag{3}$$

where $\Gamma_1 \cup \Gamma_2 = \partial\Omega$, θ_e the external temperature, a an absorption coefficient and b a radiation parameter.

For the velocity we prescribe also the initial state $u_0(x)$ and use either a no slip (Dirichlet) condition $u = w$ on Γ_1, or a no stress (Neumann) condition

$$-pn + \nu(\nabla u + \nabla u^{\mathrm{T}}) \cdot n = 0 \quad \text{on } \Gamma_2, \tag{4}$$

where n denotes the outward unit normal at the boundary. In most cases Γ_1 represents the walls and the inflow boundaries, while Γ_2 represents the outflow boundaries.

3. Numerical discretization

3.1. Time scheme

We used a semi-implicit Euler finite difference scheme with a pressure correction to impose incompressibility. If δt is the time step and $u^n = u(n\delta t)$; Eqs. (1) and (2) lead to

$$\begin{cases} \dfrac{\tilde{u}^n - u^{n-1}}{\delta t} + u^{n-1} \cdot \nabla u^{n-1} + \nabla p^{n-1} - \nu\Delta u^{n-1} = g\alpha\theta^{n-1}, \\ \dfrac{\theta^n - \theta^{n-1}}{\delta t} + u^n\nabla\theta^n - \kappa\Delta\theta^n = 0. \end{cases} \tag{5}$$

To recover incompressibility for u^n we use a projection (see [10] or [19]) and solve for ϕ

$$-\Delta\phi = -\nabla \cdot \tilde{u}^n \text{ in } \Omega, \quad \frac{\partial\phi}{\partial n} = 0 \text{ on } \Gamma_1, \qquad \phi = 0 \text{ on } \Gamma_2. \tag{6}$$

And then set

$$u^n = \tilde{u}^n - \nabla\phi, \qquad p^n = p^{n-1} - \frac{\phi}{\delta t}.$$

3.2. Discretization in space

Let $\Omega_h = \cup_j K_j$ be a discretization by non-overlapping tetraedra.

Let V_h be the space of continuous affine functions on this triangulation and V_{0h} a subspace containing the Dirichlet boundary conditions. Let $J_{0h} = V_{0h}^3$. The Petrov–Galerkin approximation finds $\tilde{u}_h - w_h \in J_{0h}$ such that $\forall v_h \in J_{0h}$

$$\int_{\Omega_h} \frac{\tilde{u}_h^n - \tilde{u}_h^{n-1}}{\delta t} v_h + \int_{\Omega_h} \tilde{u}_h^{n-1} \nabla \tilde{u}_h^{n-1} v_h + \int_{\Omega_h} \nu \nabla \tilde{u}_h^{n-1} \nabla v_h + \int_{\Omega_h} \nabla p^{n-1} \cdot v_h + \int_{\Omega_h} (\nabla p^{n-1} + \tilde{u}_h^{n-1} \nabla \tilde{u}_h^{n-1}) \cdot G_1^K(\tilde{u}_h^{n-1}, v_h) = \int_{\Omega_h} \tilde{f} v_h, \tag{7}$$

where w_h is an extension in $(V_h)^3$ of the boundary conditions and G_1^K are the stabilizing functions given by the PSI scheme [20]; Notice that the time derivative and the viscous terms were removed from the stabilization factor. To solve these equations, we used the mass lumping for the time term.

The discretization of (6) is done in the usual way on V_{0h} and for the temperature one seeks for a $\theta_h^n - \theta_{0h} \in \Theta_{0h}$ with $\forall \tau_h \in \Theta_{0h} = \{\tau_h \in V_h,\ \tau_h|_{\Gamma_1} = 0\}$:

$$\int_{\Omega_h} \frac{\theta_h^n - \theta_h^{n-1}}{\delta t} \tau_h + \int_{\Omega_h} \kappa \nabla \theta_h^{n-1} \nabla \tau_h + \int_{\Omega_h} u_h^{n-1} \nabla \theta_h^{n-1} \tau_h + \int_{\Gamma_{2h}} \kappa (a\theta_h^n + b(\theta_h^n - \theta_e)((\theta_h^{n-1})^2 + \theta_e^2)(\theta_h^{n-1} + \theta_e))\tau_h + \int_{\Omega_h} u_h^{n-1} \nabla \theta_h^{n-1} \cdot G_1^K(\theta_h^{n-1}, v_h) = 0. \tag{8}$$

3.3. Iterative solvers and preconditioners

The Poisson problems are solved using a conjugate gradient method. To solve the non-symmetric matrices we use a specific version of the GMRES algorithm which allows a variable preconditioner implemented with a reverse communication protocol for more flexibility on parallel machines [27].

The solver is parallelized by a Domain Decomposition algorithm and implemented with MPI to run on a PC cluster. The explicit steps achieve perfect parallelism of course. It is only the pressure projection which creates problems. The parallelization is implemented at the matrix-vector multiply level, so that even with a preconditioner and with a PC cluster with ethernet links the cost of communications is small.

The methods just described are now part of `NSIKE` [8].

4. Mesh adaptation

In this context, it is well known that the quality of the numerical solutions is strongly related to the underlying (shape and size) mesh quality [5]. However, it is desirable to find a compromise between accuracy and computing time, thus to reduce as much as possible the number of degrees of freedom. Mesh adaptation is the key point of this strategy. In this section, we will briefly recall the (anisotropic) error estimate of the interpolation error based on the second derivatives of the solutions presented in [22,24].

Assume that the error is bounded by the interpolation error e_i on mesh $\mathscr{H}_i$. The goal is to equi-distribute the interpolation error over the mesh elements. Recall that on a mesh element K (see [25])

$$\|u - \Pi_h u\|_{\infty,K} \leqslant c \max_{x \in K} \max_{\vec{e} \in E_K} \langle \vec{e}, |H_u(x)| \vec{e} \rangle, \tag{9}$$

where c is a constant, E_K is the set of the edges of K and $|H_u|$ is the absolute value of the Hessian of the variable u: $|H_u| = \mathscr{R}|\Lambda|\mathscr{R}^{-1}$, with $|\Lambda| = \mathrm{diag}(|\lambda_i|)$, where $\mathscr{R}$ is the eigenvector matrix and $\Lambda = \mathrm{diag}(\lambda_i)$ is the eigenvalue matrix. For other strategies see [30–32].

Thus the Hessian of u provides anisotropic (directional) information. More generally, given a positive definite matrix field $\widetilde{\mathscr{M}}(K)$ on K, let

$$\varepsilon_K = c \max_{\vec{e} \in E_K} \langle \vec{e}, \widetilde{\mathscr{M}}(K)\vec{e} \rangle. \tag{10}$$

Then we constrain the Delaunay mesh generator to give for each edge e

$$\langle \vec{e}, \mathscr{M}(K)\vec{e} \rangle = 1, \tag{11}$$

where $\mathscr{M}(K) = c\varepsilon^{-1}\widetilde{\mathscr{M}}(K)$ (see [3]).

In practice, to avoid unrealistic metrics, we have to bound the eigenvalues λ_i of H_u as follows:

$$\tilde{\lambda}_i = \min_i(\max_i(c\varepsilon^{-1}|\lambda_i|, h_{\max}^{-2}), h_{\min}^{-2}). \tag{12}$$

Moreover, in the context of numerical simulations (especially in CFD), it is often desirable to combine various metrics together (e.g. pressure, temperature and velocity Hessians). For instance, given two metric tensors $\mathscr{M}_1$ and $\mathscr{M}_2$, the intersection metric $\mathscr{M} = \mathscr{M}_1 \cap \mathscr{M}_2$ is represented by the ellipsoid and can be computed using the simultaneous reduction of the two underlying quadratic forms: $\mathscr{E}_{\mathscr{M}} = \sup_{\mathscr{M}_i \in \mathbf{M}_d} \mathscr{E}_{\mathscr{M}_i} \subset \mathscr{E}_{\mathscr{M}_1} \cap \mathscr{E}_{\mathscr{M}_2}$.

4.1. Unit mesh adaptation

The discrete anisotropic metric field $\mathscr{M}$ is used to prescribe the size and stretching of the mesh elements in order to equidistribute the interpolation error. Practically, this can be achieved by adjusting the edge lengths to 1

$$l_{\mathscr{M}}(\overrightarrow{PX}) = \int_0^1 ({}^t\overrightarrow{PX}\,\mathscr{M}(P + t\overrightarrow{PX})\overrightarrow{PX})^{1/2} = 1. \tag{13}$$

The generation of an adapted mesh can be obtained in two ways, either by modifying iteratively the current mesh [16] or by re-creating a new surface and a new volume mesh. Both approaches have been implemented and involve a constrained anisotropic Delaunay mesh generator to create or modify the mesh topology. It consists in replacing all distances checks by edge lengths computations when inserting a new vertex.

In the classical problem, the constrained Delaunay kernel based on cavity remeshing can be written as: $\mathscr{T} = \mathscr{T} - \mathscr{C} + \mathscr{B}$, where $\mathscr{T}$ denotes the current mesh, $\mathscr{C}$ is the cavity associated with the point P and $\mathscr{B}$ is the remeshing of $\mathscr{C}$ (a star-shaped polyhedron with respect to P). Here, the construction of $\mathscr{C}(P) = \{K \in \mathscr{T}, P \in \mathrm{Ball}(P)\}$ (Ball(P) is the open circumsphere of K) is obtained using a proximity criterion [15].

In the adapted mesh the local density of the mesh vertices is strictly related to the metric specifications. However, in order to comply with finite element requirements, this adapted mesh must be optimized. The size and the shape of the elements are improved using local topological and geometrical mesh modifications (edge flipping and node relocation).

Mesh generation is carried out sequentially because it is not greedy in terms of computer resources (e.g. 10^7 elements with 512 Meg RAM and 10^6 elements per minute).

5. A case study

The French atomic energy center at Cadarache is building a test site to evaluate the cooling properties of their design for hot nuclear spent fuel [2]. The problem is not in the radioactivity but in the number of years necessary to bring down the system from hot to warm conditions. So the geometry is a hall open at the top

with an air intake at the bottom; the cold air flows around the hot spent fuel container and cools it. Normally hot air could flow naturally by a chimney effect but in the present testing facilities the hall is too small and so air has to be brought into the air intake by a ventilator.

5.1. User interface

The domain is defined by constructive solid geometry (CSG) using the language of POV-Ray [21], an image synthesis software that is in the public domain. For example, the top part of the chimney is defined by five planes identified by colors; these will be used later for boundary conditions.

```
plane {⟨−1,0,0⟩, 0 pigment {color rgb ⟨1,0.,0.⟩}}
plane {⟨1,0,0⟩, 3.7 pigment {color rgb ⟨1,1,0.⟩}}
plane {⟨0,−1,0⟩, 0 pigment {color rgb ⟨1,0.,1⟩}}
plane {⟨0,1,0⟩, 4.3 pigment {color rgb ⟨1,1,1⟩}}
plane {⟨0,0,1⟩, 18 pigment {color rgb ⟨0,0,0⟩}}
```

The scene is defined by set operations on each object.

POV-Ray must be adapted to PDE solvers. So we enriched the language and add keywords such as inside to define the computational domain. Here the chimney is on the left or right side of each plane and it is defined by writing

```
inside (⟨1,0,0.⟩) and inside (⟨1,0,1⟩) and inside (⟨1,1,1⟩) and
inside (⟨1,1,0.⟩) and inside (⟨0,0,0⟩)
```

This language is compiled and produces a C-function which tells whether a given point is in the computational domain. Then with this a modified Marching Cube algorithm [1] intersects the CSG with a background uniform finite difference mesh and creates an admissible, but not optimal, surface mesh (see Fig. 1). This strategy is used now in freefem3D [1] but only for graphics. Then an optimized computational surface mesh from the marching cube mesh with the software YAMS, a fully automatic adaptative isotropic surface remeshing tool described in [11,12]. Here it produced an initial mesh with 10,675 vertices and 21,342 triangles (see Fig. 2). Then the Delaunay-based software GHS3D (see [13]) produced a volume mesh with 54,183 vertices and 293,862 tetrahedra.

5.2. Air flow computation

The challenge we had to face is to link in an automatic fashion POV-Ray, freefem3D, YAMS, GHS3D and the new NSIKE described in Section 3. A script in python has been written for this purpose and it works because each module is robust in all cases, except for the time step in NSIKE which is problem specific and must be chosen separately for each case. The results are reported on Figs. 3–4.

The inflow velocity, under the canister is 2 m s^{-1}, there is free flow at the top of the chimney and all walls are considered to be viscous. The Reynolds number is 18,000. The flow is displayed at time $t = 52.5$ s.

The cylinder which represents the nuclear waste canister has been considered as a heat source (the temperature of the waste is about 500°) and the input air represented a cold source (the temperature of this

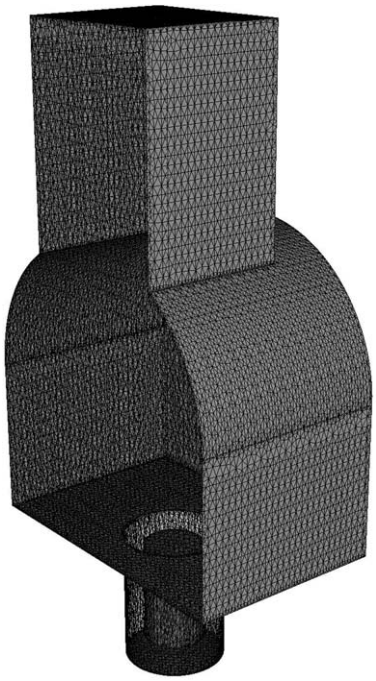

Fig. 1. Mesh generated by FreeFEM3d.

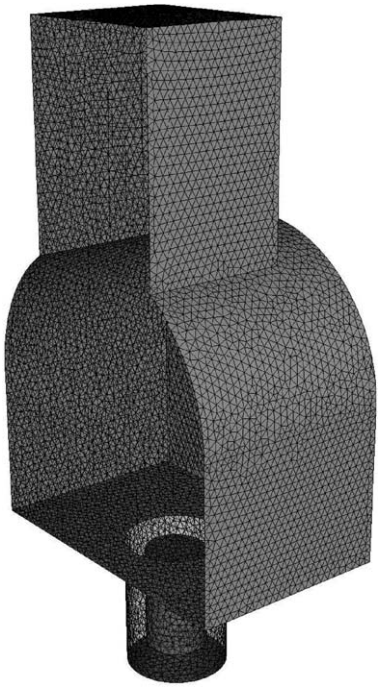

Fig. 2. Computational mesh: mesh generated by YAMS.

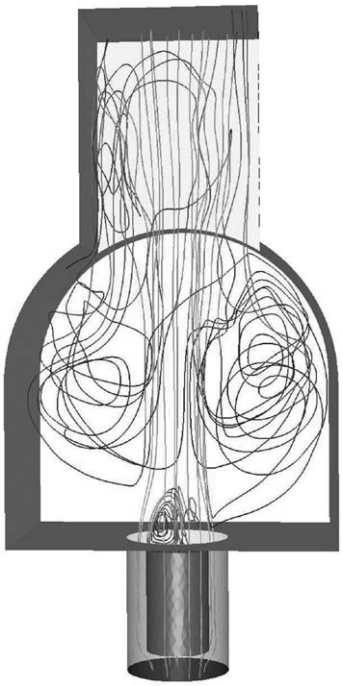

Fig. 3. Streamlines of the velocity field at time $t = 52.5$ s.

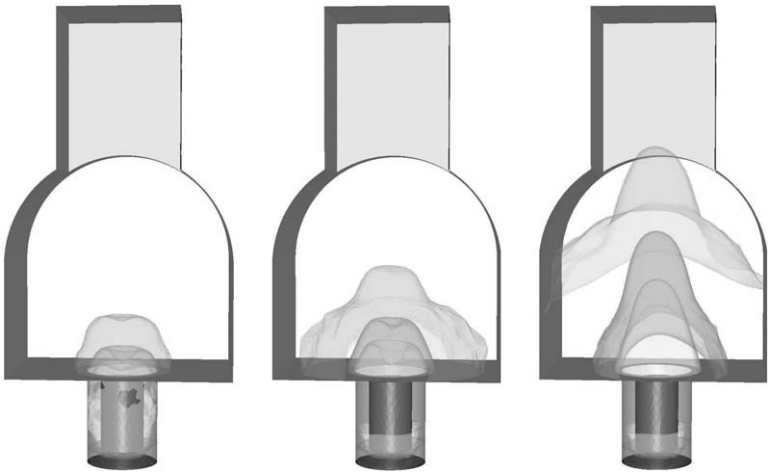

Fig. 4. Isosurfaces of the temperature distribution within the chimney at times 2″, 11″ and 31″.

source is the external temperature that is about 20°). Neumann conditions have been prescribed on the remaining part of the domain. Fig. 4 shows the isosurfaces of the temperature distribution within the chimney.

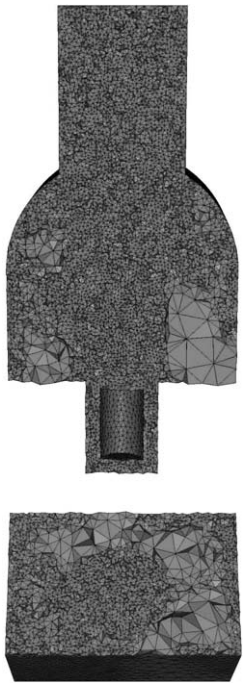

Fig. 5. Adapted meshes: cut through the tetrahedral mesh.

Fig. 6. Adapted meshes: cut through the tetrahedra and associated velocity field (on the top) or curl field (at the bottom).

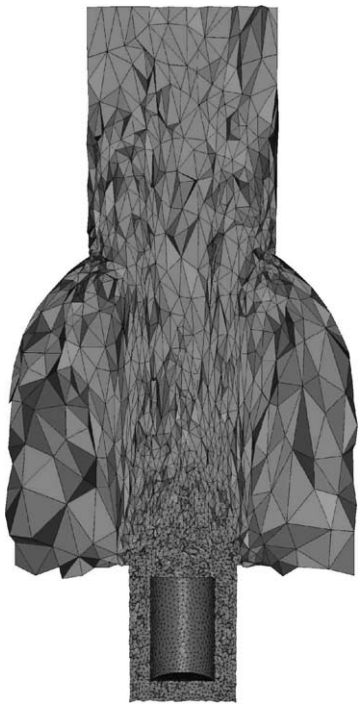

Fig. 7. Anisotropic meshes: cut through the tetrahedral mesh.

Fig. 8. Adapted meshes: cut through the tetrahedra and associated velocity field (on the top).

5.3. Mesh adaptation

To create the metric, we consider two variables: L^2 norm of velocity field and L^2 norm of curl fields. Indeed, if the adaptation is only based on L^2 norm of velocity field, the vortex is not captured. In the example, the mesh has been adapted 30 times and at each iteration 100 time steps have been computed within the solvers.

5.4. Performance

The total time to solve the Navier–Stokes equations is close to 7 h on 12 PC workstation. The last mesh contains about 121,000 vertices and 696,000 tetrahedra. Figs. 5 and 6 show various cuts through the adapted volume meshes at the final iteration. Notice that the flow is not symmetric because it is not perfectly stationary. All graphics are displayed with the public domain `medit` (written by Frey) [26].

Figs. 7 and 8 illustrate the anisotropy achieved using the mesh adaptation method (`MMG3d` software) based on local mesh modifications.

6. Others examples

To illustrate the flexibility of the approach we present briefly two other examples.

The first one is for the air conditioning in a house. The geometry we consider is the last floor of a furnished house. The hot air inflows are on the left, from an open door. Fig. 9 shows the isosurfaces of the temperature distribution in the rooms. In this example Fourier radiating conditions have been used at the windows. The Reynolds number is 3000. The inflow temperature is 0° while the room temperature is initially 20.

The second application is a system of ventilation in a subway station. The aim of this system is to extract the heat created by the train traffic. The ventilators at the bottom right extract the air with a speed of $60\,\mathrm{m\,s^{-1}}$. Because of train traffic, a mild flow in the central tube is created by imposing a Dirichlet condition on the left while the right boundary has free outflow. The temperature is constant. Fig. 10 represents the streamlines of the velocity field within the subway for the Reynolds number is 3000.

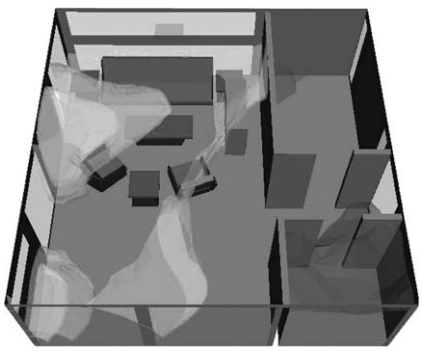

Fig. 9. Isosurfaces of the temperature within a house.

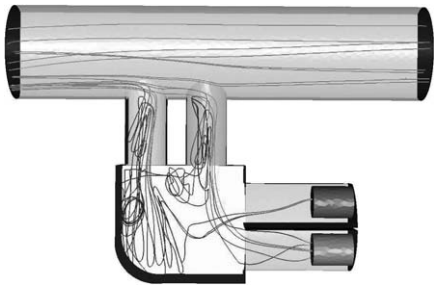

Fig. 10. Streamlines of the velocity field within the subway ventilation system.

7. Conclusion

In order to open new fields of applications for CFD we have built a software by linking some state of the art tools developed by the authors. To our knowledge this is the first complete package including a transparent mesh adaptation module with anisotropic options. Four modules were linked and special developments were made for the buoyancy term and the surface mesh generation from the image synthesis software `POV-Ray` so as to bypass expensive CAD. The results presented in this paper have demonstrated the feasibility and the efficiency of the approach especially with respect to 3D non-isotropic mesh adaptivity.

Data preparation never took more than a day and the performances are sufficient to allow in house runs even in small labs.

However more computing power is needed still to run a full turbulence model such as k–ϵ within this approach in 3D. Similarly on the case study one would have liked to obtain the chimney effect but this is still not within the reach of such overnight calculations on a PC cluster.

References

[1] S. Del Pino, Une méthode d'éléments finis pour la résolution d'EDP dans des domaines décrits par géométrie constructive. Thèse. Universitée Pierre et Marie Curie, 2002. Available from: <http://www.freefem.org/ff3d/>.
[2] Dossier de presse, Recherche sur les déchets radioactifs: avancées dans les domaines du conditionnement et de l'entreposage, Décembre 2003. Available from: <www.cea.fr/fr/presse/dossiers/Dechets2003.pdf>.
[3] F. Alauzet, P.J. Frey, Estimateur d'erreur géométrique et métriques anisotropes pour l'adaptation de maillage. Partie I: aspects théoriques, RR-4759, INRIA Rocquencourt, 2003.
[4] H. Borouchaki, P.L. George, Quality mesh generation, CR Acad. Sci. Paris t 328 (Series IIb) (2000) 505–518.
[5] P.G. Ciarlet, Basic error estimates for elliptic problems, in: Finite Element methods (Part 1), in: P.G. Ciarlet, J.L. Lions (Eds.), Handbook of Numerical Analysis, vol. II, North Holland, 1991, pp. 17–352.
[6] R. Glowinski, Numerical Methods for Nonlinear Variational Problems, Springer-Verlag, New York, 1984.
[7] F. Guibault, P. Labbé, J. Dompierre, Adaptivity works! controlling the interpolation error in 3D, in: Proceedings of the Fifth World Congress on Computational Mechanics, Vienna, Austria, July, 2002.
[8] G. Medić, B. Mohammadi, NSIKE: an incompressible Navier–Stokes solver for unstructured meshes, RT-3644, INRIA-Rocquencourt, 1999.
[9] O. Pironneau, The Finite Element Method for Fluids, Wiley, Chicester, 1989.
[10] J.A. Chorin, A numerical method for solving incompressible viscous flow problems, J. Comput. Phys. 2 (1967) 12–26.
[11] P.J. Frey, YAMS: a fully automatic adaptive isotropic surface remeshing procedure, RT-0252 INRIA-Rocquencourt, 2001.
[12] P.J. Frey, About surface remeshing, in: Proceedings of the 9th International Meshing Roundtable, New Orleans, LO, USA, 2000, pp. 123–136.
[13] P.L. George, Premières expériences de maillage automatique par une méthode de Delaunay anisotrope en trois dimensions, RT-0272, INRIA Rocquencourt, 2002.
[14] P.L. George, H. Borouchaki, Génération automatique de maillages tridimensionnels respectant une carte de taille, Revue européenne des éléments finis 7 (4) (1998) 339–363.

[15] P.L George, H. Borouchaki, P. Frey, P. Laug, E. Saltel, Mesh generation and mesh adaptativity: theory and techniques, in: E. Stein, R. De Borst, T.J.R. Hughes (Eds.), Encyclopedia of Computational Mechanics, Wiley, 2004.
[16] R. Löhner, Regridding surface triangulations, J. Comput. Phys. 126 (1996) 1–10.
[17] B. Mohammadi, O. Pironneau, Analysis of the K-Epsilon Turbulence Model, Wiley, 1994.
[18] L. Landau, F. Lifschitz, Mécanique des fluides, Mir, 1964.
[19] W. Dettmer, D. Perić, Comput. Methods Appl. Mech. Engrg. 192 (9–10) (2003) 1177–1226.
[20] H. Deconink, R. Struijs, G. Bourgois, P.L. Roe, Compact advection schemes on unstructured grids, VKI Lecture Series, 1993–2004.
[21] A. Wardley, Persistence of vision, `POV-Ray`. Available from: <http://www.povray.org>.
[22] M. Fortin, Estimation d'erreur a posteriori et adaptation de maillages, Revue européenne des éléments finis 9 (4) (2000).
[23] P. Frey, Description of the software `YAMS` for automatic surface mesh adaptation. Available from: <http://www-rocq1.inria.fr/gamma/yams/yams.html>.
[24] H. Borouchaki, D. Chapelle, P.L. George, P. Laug, P.J. Frey, Estimateurs d'erreur géométriques et adaptation de maillage, in: P.L. George (Ed.), Maillage et adaptation, Série Mécanique et Ingénierie des Matériaux, Méthodes Numériques, Hermès Science, Paris, 2001.
[25] F. Alauzet, P.J. Frey, Estimateur d'erreur géométrique et métrique anisotropes pour l'adaptation de maillage. Partie I: aspects théoriques, Rapport de Recherche INRIA 2003, RR-4759.
[26] P. Frey, The public domain visualisation software `medit`. Available from: <http://www.ann.jussieu.fr/~frey/logiciels/medit.html>.
[27] G. Medic, B. Mohammadi, Injection/suction boundary conditions for fluid/structure interaction simulation in incompressible flows, Int. J. Numer. Meth. Fluids 40 (7) (2001) 275–290.
[28] C. Dobrzinsky, Numerical Coupling for air flow computations in complex architectures. Congrès d'Analyse Numérique, Obernay, May 2004, and ECCOMAS Conference Jyvaskyla, July 2004.
[29] S. Del Pino, O. Pironneau, A fictitious domain based general PDE solver, in: E. Heikkola et al. (Eds.), Numerical Methods for Scientific Computing, CIMNE, Barcelona, 2003.
[30] M.J. Marchant, N.P. Weatherill, O. Hassan, The adaptation of unstructured grids for transonic viscous flow simulation, Finite Elem. Anal. Des. 25 (3–4) (1997) 199–217.
[31] J.F. Thompson, B.K. Soni, N.P. Weatherill, Handbook of Grid Generation, CRC Press, 1999.
[32] N.P. Weatherill, O. Hassan, Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints, Int. J. Numer. Meth. Engng. 37 (1994) 2005–2039.