Nationaal Lucht- en Ruimtevaartlaboratorium

National Aerospace Laboratory NLR

NLR-TP-2002-105

A simple, robust and fast algorithm to compute deformations of multi-block structured grids

S.P. Spekreijse, B.B. Prananta and J.C. Kok

NLR-TP-2002-105

A simple, robust and fast algorithm to compute deformations of multi-block structured grids

S.P. Spekreijse, B.B. Prananta and J.C. Kok

This investigation has been carried out under a contract awarded by the Netherlands Ministry of Defence, monitored by the Royal Netherlands Air Force, contract number (NTP) N98-04 and as part of NLR's Basic Research Programme, Working Plan number A.1.B.2.

The Netherlands Ministry of Defence and the Royal Netherlands Air Force have granted NLR permission to publish this report.

This report is published in the Proceedings of the 8th International Conference on Numerical Grid Generation in Computational Field Simulations, Honolulu, Hawaii, USA, June 3-6, 2002.

Customer:	National Aerospace Laboratory NLR
Working Plan number:	A.1.B.2
Owner:	National Aerospace Laboratory NLR
Division:	Information and Communication Technology/Fluid Dynamics
Distribution:	Limited
Classification title:	Unclassified
	February 2002

Summary

A grid deformation method has been developed for the movement of multiblock, structured grids due to surface deformation arising from aeroelastics or design optimisation. The method uses a volume spline interpolation technique to compute the deformation of block vertices and block edges. After that, the deformation of block-faces is computed by an arclength-based two-dimensional transfinite interpolation method. Finally, an arclength-based three-dimensional transfinite interpolation method is used to compute the deformation of grid points inside the blocks. The method is demonstrated for simple and complex aeroelastic aircraft applications using Navier-Stokes computational fluid dynamics.

Contents

12 Figures

(15 pages in total)

1 Introduction

Unsteady flow calculations for problems with deforming surface shapes, such as those found in aeroelastic simulations, require a fast and reliable method for deforming the computational grid during each time step. Like most authors, we consider grid deformation as a method of propagating geometric displacements into an existing initial grid while preserving the original character of the mesh (Refs. 5, 7,8). Several requirements are put on the grid deformation method. Since the grid deformation has to be carried out at each time step, it is of major importance to keep the method simple and efficient. Further, it should be robust enough to handle arbitrarily complex multi-block grids, including Navier-Stokes boundary layer grids and grid singularities. The method must produce deformed grids of acceptable quality to the flow solver. Preserving positive Jacobians (volumes) is a minimum requirement. Maintaining smoothness, orthogonality, and the overall quality and characteristics of the initial grid is also needed. Additionally, at block interfaces, the deformed grid must remain point matched (C^0-continuity). Finally, the only required input is the initial grid and the displacements of the grid points in those faces that define the configuration surface.

One of the advantages of structured multi-block grids compared to unstructured grids is that transfinite interpolation (TFI) can be used to compute the displacements of the interior grid points in a block from previously computed displacements of grid points in the six block-faces. Similarly, a two-dimensional TFI method can be easily used to computed the displacements of the interior grid points in a face from previously computed displacements of grid points in the four face-edges. Therefore, the main problem is the computation of the displacement of vertices and edges in a multi-block grid.

It is assumed that the displacement of the configuration surface is prescribed. From the connectivity information in a multi-block grid, it is possible to identify all vertices on the configuration surface. The prescribed displacements of those vertices are interpolated with the volume-spline method of Hounjet and Meijer (Ref. 10). The volume-spline method generates a function to compute the displacement everywhere as a function of location in space. The volume spline method is only used to compute the displacements of grid points in all vertices and all edges not lying on the configuration surface. The displacements of the remaining grid points are subsequently computed with a two-dimensional TFI method in the interior of faces and with a three-dimensional TFI method in the interior of the blocks, as mentioned above.

The volume-spline method requires a matrix inversion of a full square matrix of $N+1$ rows where N is the number of vertices on the configuration surface. In practice, it appears that for a complex fighter aircraft, the number of vertices on the configuration surface is typically of order 100.

The proposed grid deformation method is purely algebraic and non-iterative. The computation of a displacement by the volume-spline method in an arbitrary location in space is much more expensive than the computation of a displacement by a transfinite interpolation method. Fortunately, the volume spline method is only used to compute the displacements of grid points at vertices and edges which is a negligible amount of points compared to the remaining grid points in faces and blocks.

The grid deformation method as described here has been implemented in a stand-alone code to be used for static aeroelastic calculations and design optimisation. It also became part of NLR's time-dependent Reynolds-averaged Navier-Stokes flow solver ENSOLV for static and dynamic aeroelastic calculations (Ref. 2).

2 Volume Spline Interpolation

Volume spline interpolation is a very simple method to interpolate non-uniformly spaced data. The method belongs to the category of scattered data interpolation methods using radial basis functions (Ref. 1). The method is easily coded, robust and fully automatic. To introduce the method, it is first described in one dimension and after that extended to 3D.

Consider a set of unordered support points $\{x_i \in \mathbb{R}\}, i = 1 \dots N$, together with function values $\{f_i \in \mathbb{R}\}, i = 1 \dots N$. The volume spline interpolation function $f : \mathbb{R} \mapsto \mathbb{R}$ may then be defined as

$$f(x) = \sum_{i=1}^{N} \alpha_i R_i(x) \tag{1}$$

with $R_i(x) = |x - x_i|$. The unknown coefficients α_i can be computed from the interpolation conditions $f(x_i) = f_i$. For $x \neq x_i$, it is clear that $\partial^2 f/\partial x^2 = 0$. Thus, volume spline interpolation in one-dimension is nothing else then piecewise linear interpolation. It is clear that for $x \mapsto \pm\infty$ we have

$$f(x) = \pm x \sum_{i=1}^{N} \alpha_i \mp \sum_{i=1}^{N} \alpha_i x_i \tag{2}$$

Therefore, the function $f(x)$ is bounded if $\sum_{i=1}^{N} \alpha_i = 0$. In general, a bounded piecewise linear interpolation is obtained by

$$f(x) = \alpha_0 + \sum_{i=1}^{N} \alpha_i R_i(x) \tag{3}$$

where the coefficients $\alpha_i, i = 0 \dots N$, are computed from the interpolation conditions $f(x_i) = f_i$ together with side condition $\sum_{i=1}^{N} \alpha_i = 0$.

The extension of the volume spline method to three dimension is straightforward. In that case we have support points $\{\vec{x}_i \in \mathbb{R}^3\}, i = 1 \dots N$ with function values $\{f_i \in \mathbb{R}\}, i = 1 \dots N$, and the interpolation function is defined as

$$f(\vec{x}) = \alpha_0 + \sum_{i=1}^{N} \alpha_i R_i(\vec{x}) \tag{4}$$

with $R_i(\vec{x}) = \|\vec{x} - \vec{x}_i\|$. Again the coefficients $\alpha_i, i = 0 \dots N$, are computed from the interpolation conditions $f(\vec{x}_i) = f_i$ together with side condition $\sum_{i=1}^{N} \alpha_i = 0$.

The Laplacian of a scalar function $\phi(R)$ with $R = \|\vec{x}\|$ obeys in 3D the relation $\triangle\phi = 1/R^2 \partial/\partial R(R^2 \partial\phi/\partial R)$. For $\phi(R) = R$ we have $\triangle\phi = 2/R$ and thus $\triangle\triangle\phi = 0$.

Thus, the volume spline interpolation function $f(\vec{x})$ obeys the biharmonic equation $\triangle\triangle\phi = 0$ everywhere except at the support points. Therefore, like in 1D, the interpolation function $f(\vec{x})$ will not possess new local extrema outside the support points. This important property makes the volume spline interpolation method suitable for our purposes, because it will reduce the risk of the occurrence of grid folding in the deformed mesh. The boundedness of the volume spline is important to let the displacement decay towards the farfield boundaries. Notice that the volume spline interpolation has both rotational and translational invariance.

The volume spline method can also be used to interpolate a vectorfield. In that case we have support points $\{\vec{x}_i \in \mathbb{R}^3\}, i = 1 \ldots N$ with vectors $\{\vec{d}_i \in \mathbb{R}^3\}, i = 1 \ldots N$, and the interpolation is defined as

$$\vec{d}(\vec{x}) = \vec{\alpha}_0 + \sum_{i=1}^{N} \vec{\alpha}_i R_i(\vec{x}) \tag{5}$$

with $R_i(\vec{x}) = \|\vec{x} - \vec{x}_i\|$. Again the coefficients $\vec{\alpha}_i, i = 0 \ldots N$, are computed from the interpolation conditions $\vec{d}(\vec{x}_i) = \vec{d}_i$ together with side condition $\sum_{i=1}^{N} \vec{\alpha}_i = 0$. The coefficient matrix of the linear system of equations for the coefficients $\vec{\alpha}_i$ is full and symmetric and only depends on the position of the support points. The elements on the main diagonal are zero. Only one matrix inversion need to be calculated to compute simultaneously the three components of the coefficients $\vec{\alpha}_i$. For more than about 200 support points the matrix may become poorly conditioned (Ref. 1).

The volume spline method is used to interpolate the displacements at all vertices on the configuration surface. Thus the vertices on the configuration are used as support points $\vec{x}_i$ and the displacement vectors $\vec{d}_i$ define the righthandside of the interpolation conditions. Symmetry is easily handled by mirroring of the configuration vertices and corresponding displacement vectors. The solution of the matrix equation is obtained using the subroutine SGESV from the package LAPACK retrieved from NETLIB which can be found at the Internet address
`http://www.netlib.org/lapack/index.html`.

3 Transfinite Interpolation

After computing the displacement of the vertices and the edges of the blocks, the displacement of the remaining grid points of the multi-block grid is computed by transfinite interpolation as described below. Thus, the displacement of only a very small portion of grid points is computed by the expensive volume spline interpolation method, while the displacement of the remaining large part of grid points is computed by the inexpensive TFI method. The TFI method is only used to compute the displacements of grid points in the interior of faces not lying on the configuration surface and of grid points in the interior of blocks. The TFI-method is arclength-based in order to preserve the characteristics of the initial mesh.

A multi-block grid consists of a set of blocks $\{B\}$, faces $\{F\}$, edges $\{E\}$ and vertices $\{V\}$. The connectivity relations between the elements of these sets define the multi-block topology (Refs. 3,4).

Furthermore, each block $B \in \{B\}$ has its own structured three-dimensional volume grid defined as

$$X^B = \{\vec{x}_{i,j,k} \mid i = 0 \ldots NI^B, j = 0 \ldots NJ^B, k = 0 \ldots NK^B\}. \tag{6}$$

Additionally, normalized arclength variables (s, t, u) can be defined along the grid lines in respectively the i, j and k direction. For example, the normalized arclength in the i direction is defined as

$$s_{i,j,k} = \frac{\sum_{m=1}^{i} \parallel \vec{x}_{m,j,k} - \vec{x}_{m-1,j,k} \parallel}{\sum_{m=1}^{NI} \parallel \vec{x}_{m,j,k} - \vec{x}_{m-1,j,k} \parallel}. \tag{7}$$

Thus $s_{0,j,k} = 0$ and $s_{NI,j,k} = 1$. A similar formula is used to define t and u as normalized arclength in respectively the j and k direction.

Each face $F \in \{F\}$ has its own structured two-dimensional surface grid defined as

$$X^F = \{\vec{x}_{i,j} \mid i = 0 \ldots NI^F, j = 0 \ldots NJ^F\}. \tag{8}$$

Again, normalized arclength variables (s, t) can be defined along the grid lines of a surface grid in respectively the i and j direction. The s coordinates of a surface grid are defined by Eq.(7) whereby the k index is omitted.

Each edge $E \in \{E\}$ has its own structured 1D curve grid defined as

$$X^E = \{\vec{x}_i \mid i = 0 \ldots NI^E\}. \tag{9}$$

Each vertex $V \in \{V\}$ has its own position vector $\vec{x}_v$.

The TFI-method for the computation of the displacements in the interior of a surface grid of a face can be expressed most easily as a two-step recursionformula (Refs. 6). The first step computes the displacements in the interior by straight-line interpolation in the i-direction:

$$\vec{dx}^1(i,j) = (1-s_{i,j})\vec{dx}(0,j) + s_{i,j}\vec{dx}(NI,j). \tag{10}$$

The second step adds the mismatch of the displacements along the second pair of opposite edges by straight-line interpolation in the j-direction:

$$\begin{aligned} \vec{dx}^2(i,j) = \vec{dx}^1(i,j) \quad &+ \quad (1-t_{i,j})(\vec{dx}(i,0) - \vec{dx}^1(i,0)) \\ &+ \quad t_{i,j}(\vec{dx}(i,NJ) - \vec{dx}^1(i,NJ)). \end{aligned} \tag{11}$$

Finally, the displacement in the interior of the surface grid is defined as

$$\vec{dx}(i,j) = \vec{dx}^2(i,j). \tag{12}$$

After computing the displacement of the faces, the TFI-method is also used to compute the displacement in the interior of the blocks. For a block, the TFI method is a three step recursion formula. The first step computes the displacements in the interior by straight-line interpolation in the i-direction:

$$\vec{dx}^1(i,j,k) = (1-s_{i,j,k})\vec{dx}(0,j,k) + s_{i,j,k}\vec{dx}(NI,j,k). \tag{13}$$

The second step adds the mismatch of the displacements along the second pair of opposite faces by straight-line interpolation in the j-direction:

$$\begin{aligned} \vec{dx}^2(i,j,k) = \vec{dx}^1(i,j,k) \quad &+ \quad (1-t_{i,j,k})(\vec{dx}(i,0,k) - \vec{dx}^1(i,0,k)) \\ &+ \quad t_{i,j,k}(\vec{dx}(i,NJ,k) - \vec{dx}^1(i,NJ,k)). \end{aligned} \tag{14}$$

The third step adds the mismatch of the displacements along the third pair of opposite faces by straight-line interpolation in the k-direction:

$$\begin{aligned} \vec{dx}^3(i,j,k) = \vec{dx}^2(i,j,k) \quad &+ \quad (1-u_{i,j,k})(\vec{dx}(i,j,0) - \vec{dx}^2(i,j,0)) \\ &+ \quad u_{i,j,k}(\vec{dx}(i,j,NK) - \vec{dx}^2(i,j,NK)). \end{aligned} \tag{15}$$

Finally, the displacement in the interior of the volume grid is defined as

$$\vec{dx}(i,j,k) = \vec{dx}^{\,3}(i,j,k). \tag{16}$$

If the displacement of the configuration surface is zero, then the deformed and initial grid will be identical.

4 Results

Figure 1 shows an initial 12 blocks O-type grid for a RAE2822 airfoil. Notice that there are six vertices on the airfoil. The displacements of these six vertices are input for the volume spline method. Figures 2,3,4 show the deformed grid for a pitch down of 10^o. Figure 3 shows a closeup near the leading edge and figure 4 shows a closeup near the trailing edge. The original grid quality and smoothness are maintained. For illustration purposes, the results are shown for an Euler mesh without grid boundary layer. However, the method can be used equally well for a Navier-Stokes mesh with boundary layer grid, as shown in figure 5.

Figures 6,7,8 show the deformed grid for an extreme pitch down angle of 45^o. Figure 7 shows that near the leading edge the deformed grid is still not folded although the mesh is rather skew. At the trailing edge, grid folding occurs as shown in figure 8. This grid folding can be easily removed by not only prescribing the deformation of the airfoil but also prescribing of the wake-edge immediately behind the airfoil, as shown in figure 9. This example indicates that the method may fail for large deformations. Fortunately, such large deformations will not occur for realistic aeroelastic simulations or optimal design studies.

Figure 10 shows another type of application of the algorithm, namely the replacement of the RAE2822 airfoil by a NACA0012 airfoil without changing the characteristics of the initial grid.

The applicability for very complex geometries, typical for fighter-aircraft configurations, has also been demonstrated. Figure 11 shows an illustration. More details can be found in (Ref. 9).

The applicability of the algorithm for design optimisation has been demonstrated in (Ref. 11). Figure 12 shows an illustration.

5 Conclusions

The grid deformation as presented here is fully automatic and requires as input only an initial mesh and the displacements of the grid points lying on the configuration surface. The method uses the displacements of the vertices lying on the configuration surface as input for the volume spline interpolation method. The computed volume spline is used to compute the displacements of grid points at vertices and edges not lying on the configuration surface. Next, arclength-based transfinite interpolation is used to compute the displacements of the remaining grid points in the interior of faces and blocks. The method has been implemented in the NLR Reynolds-averaged Navier-Stokes multi-block flow solver ENSOLV (Ref. 2) and is now used on a routinely basis for aeroelastic simulations of military aircraft configurations (Ref. 9) and for aerodynamic shape optimisation studies (Ref. 11). It appears that during each time step, the computer time to recompute the deformed grid is less than 10% compared to the computer time spent by the flow solver. The method appears to be sufficiently robust and may only fail for very large deformations which fortunately do not occur for realistic aeroelastic simulations or optimal design studies.

6 References

1. J. Hoschek, D. Lasser (1993), Fundamentals of Computer Aided Geometric Design, A K Peters, LTD.
2. J.C. Kok, S.P. Spekreijse, Efficient and Accurate Implementation of the k-ω Turbulence Model in the NLR Multi-Block Navier-Stokes System. Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2000, Barcelona, Spain. NLR-TP-2000-144.
3. S.P.Spekreijse,J.W. Boerstoel,(1996), Multiblock Grid Generation. Part 1: Elliptic Grid Generation Methods for Structured Grids. Part 2: Multiblock Aspects. Von Karman Institute for Fluid Dynamics (VKI). Lecture Series 1996-06. 27th Computational Fluid Dynamics Course. NLR-TP-96338.
4. S.P. Spekreijse,J.W. Boerstoel, An Algorithm to Check the Topological Validity of Multiblock Domain Decompositions. In: Proceedings 6th International Conference on Numerical Grid Generation in Computational Field Simulations, Greenwich, 1998. NLR-TP-98198.
5. W.T. Jones, Efficient Smoothing and Deformation of Structured Volume Grids, In: Proceedings 6th International Conference on Numerical Grid Generation in Computational Field Simulations, Greenwich, 1998.
6. R.E. Smith. Transfinite Interpolation (TFI) Generation Systems. In: Handbook of Grid Generation, eds. N.P. Weatherill, J.F. Thompson, B.K. Soni. CRC Press,1999.
7. M.A. Potsdam, G.P. Guruswamy, A Parallel Multiblock Mesh Movement Scheme for Complex Aeroelastic Applications, AIAA-2001-0716, 39th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, 2001.
8. H.M. Tsai, A.S.F. Wong, J. Cai, Y. Zhu, F. Liu, Unsteady Flow Calculations with a Parallel Multiblock Moving Mesh Algorithm, AIAA Journal, vol 39, no 6, 2001.
9. B.B. Prananta, I.W. Tjatra, S.P. Spekreijse, J.C. Kok, J.J. Meijer, Static aeroelastic simulation of military aircraft configuration in transonic flow, In: Proceedings International Forum on Aeroelasticity and Structural Dynamics, Madrid, Spain, 2001. NLR-TP-2001-346.
10. M.H.L. Hounjet, J.J. Meijer, Evaluation of Elastomechanical and Aerodynamic Data Transfer Methods for Non-planar Configurations in Computational Aeroelastic Analysis. In: Proceedings International Forum on Aeroelasticity and Structural Dynamics, Manchester, UK, 1995. NLR-TP-95690.
11. M. Laban, MOB: A Computational Design Engine Incorporating Multi-Disciplinary Design an Optimisation for Blended Wing Body Configuration, EC5^{th} Framework Programme, Contract Number: GRDI-1999-11162, 2000.

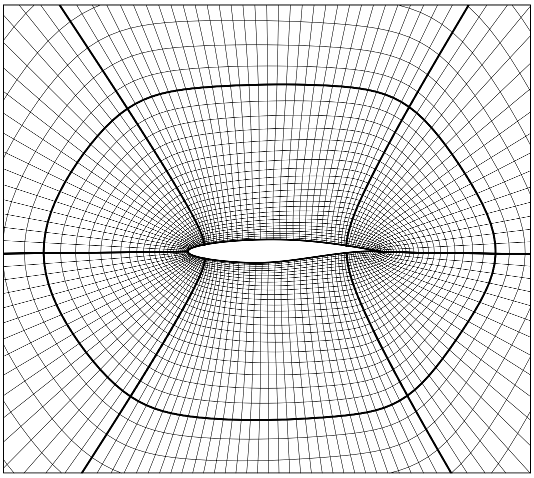

Fig. 1 *RAE2822 grid: 12 blocks.*

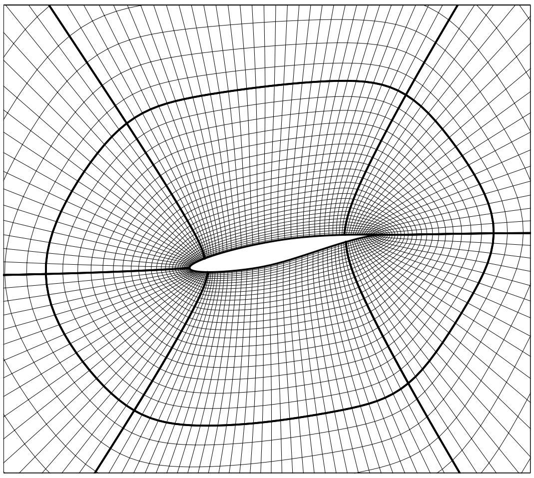

Fig. 2 *RAE2822 grid with* 10^o *pitch down.*

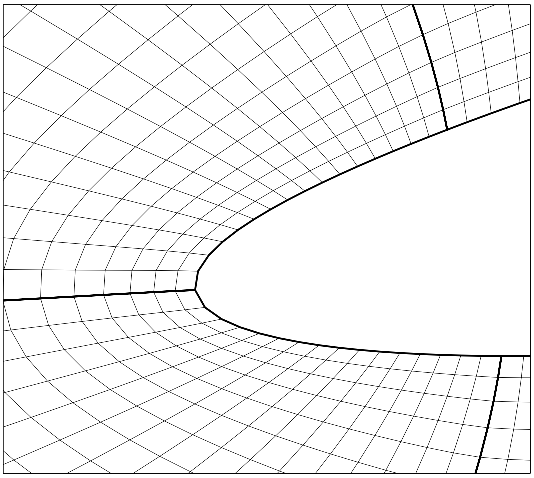

Fig. 3 *Detail at leading edge.*

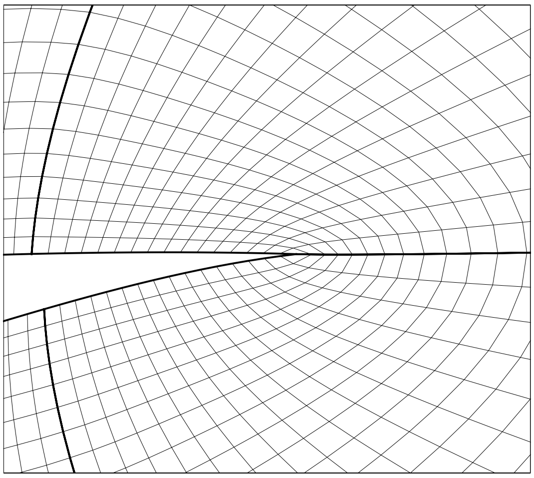

Fig. 4 *Detail at trailing edge.*

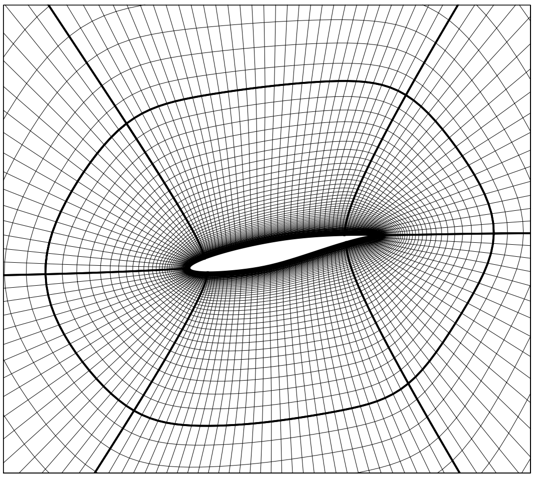

Fig. 5 *RAE2822 NAS grid with* 10^o *pitch down.*

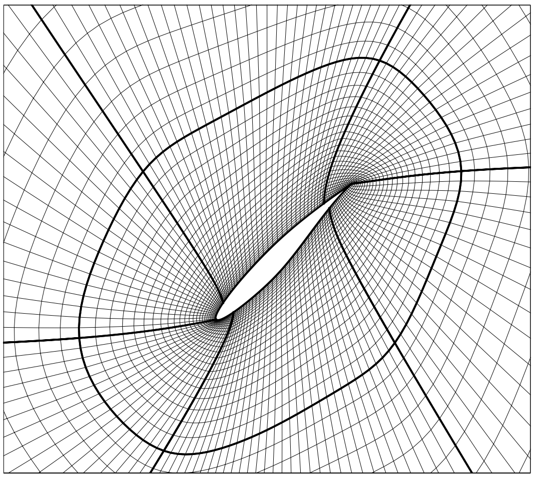

Fig. 6 *RAE2822 grid with* 45^o *pitch down.*

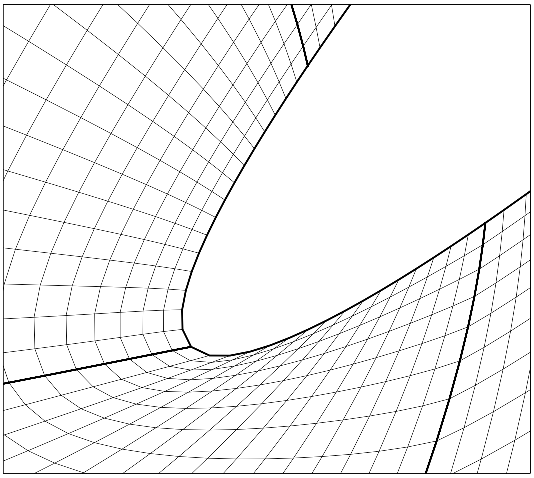

Fig. 7 Detail at leading edge.

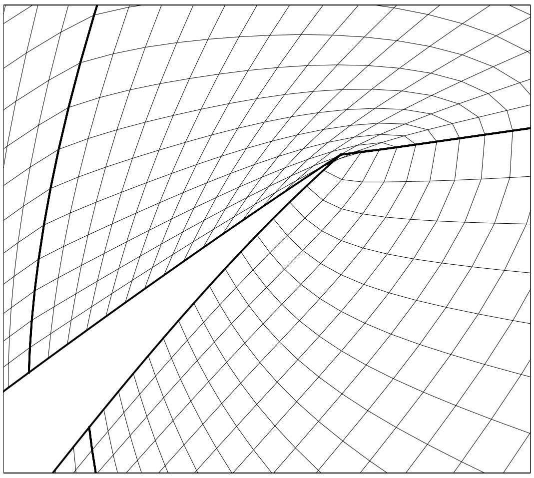

Fig. 8 Detail at trailing edge.

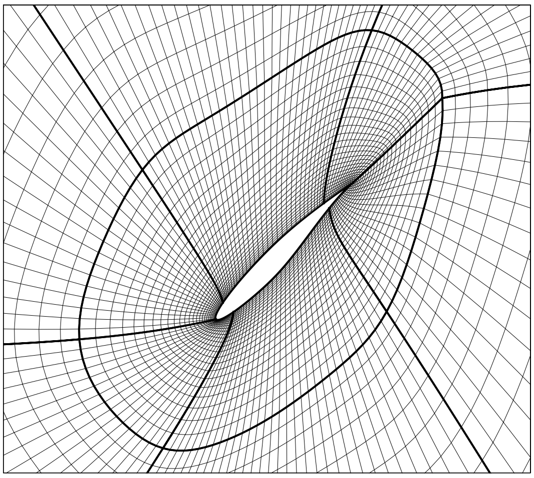

Fig. 9 RAE2822 grid with 45^o *pitch down and prescribed wake-edge deformation*

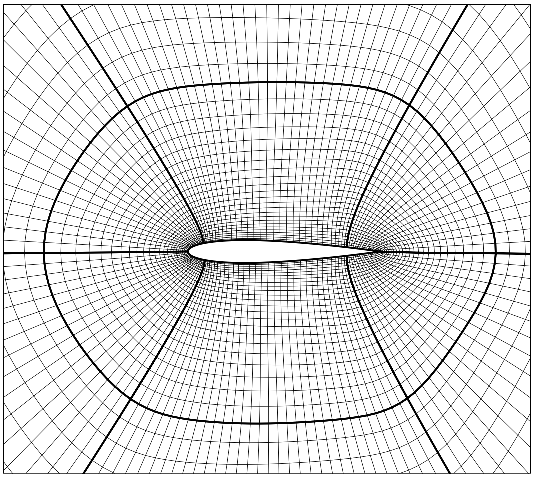

Fig. 10 RAE-airfoil replaced by NACA-airfoil.

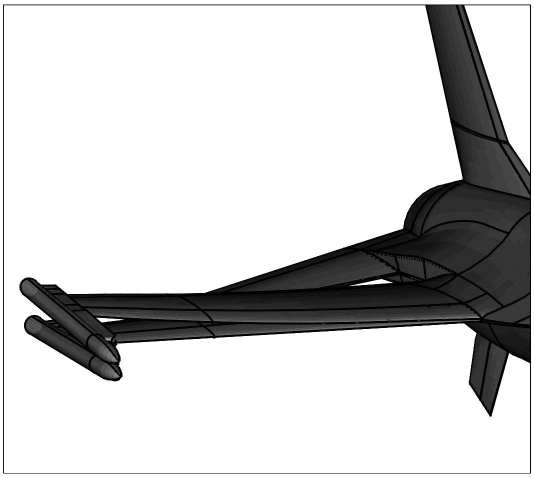

Fig. 11 Aeroelastic deformation of tip missile for fighter-aircraft.

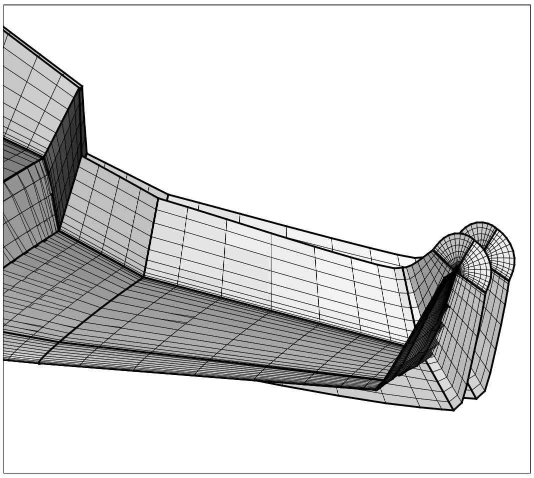

Fig. 12 Grid deformation of a winglet during optimal design process.