11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
E. Oñate, J. Oliver and A. Huerta (Eds)

ADAPTATION OF THE COMPUTATIONAL GRID TO A MOVING WING-FUSSELAGE INTERSECTION VIA NURBS AND RADIAL BASIS

M. J. MARTIN-BURGOS[†], M. CORDERO-GRACIA[*] AND M. GÓMEZ[†]

[*]School of Aeronautics
Universidad Politécnica de Madrid
Pza. Cardenal Cisneros 3, 28040 Madrid, Spain
e-mail: marta.cordero@upm.es

[†]School of Aeronautics
Universidad Politécnica de Madrid
Pza. Cardenal Cisneros 3, 28040 Madrid, Spain
e-mail: mariojmartin@telefonica.net

Key words: Intersection, Wing Fuselage, Grid Adaptation, NURBS, Radial Basis

Abstract. This paper proposes two approaches to automatically adapt the computational grid at the presence of moving intersections between surface components, such as wing-fuselage assemblies. The first method is based on the underlying CAD definition, which is represented by Non-Uniform Rational B-Splines (NURBS). The second method employs Radial Basis Functions (RBFs). Both methods have been tested to fix the surface grid of an DLR F6 wing-body configuration, where the wing geometry has been deformed at the intersection with the fuselage. The proposed strategies are suitable for both structured and unstructured computational grids and can be deployed in an industrial context to adapt the computational grid upon deformations of surface components

1 INTRODUCTION

In the context of aircraft design, the grid generation of complex configurations involving several components, is usually an expensive and time consuming task that requires great expertise. In order to avoid the regeneration of the computational grid, automatic grid adaptation techniques are considered a fast reliable approach for small deformations; commonly employed in automatic optimization loops and aeroelastic deformations. However, grid deformation algorithms face limitations at the presence of moving intersections between surface components, such as wing-fuselage and wing-pylon-nacelle assemblies. Without a proper treatment of the intersection between surface patches, the use of automatic optimization methods for aircraft design are limited to individual components,

while the interaction between components can be quite relevant.

In this work, two grid adaptation algorithms are proposed and tested to fix the surface grid at the presence of moving intersections between components. Once the surface grid is properly adapted to the new configuration, a suitable volumetric adaptation is employed. The first one is based on Non-Uniform Rational B-Splines (NURBS) and the second one is based on Radial Basis functions (RBFs).

This paper is organized as follows. In section 2, NURBS are briefly introduced and the algorithm to adapt the surface grid is explained. In section 3, the method based on RBFs interpolation is presented. In section 4, both strategies are employed and compared to adapt the surface grid of a wing-body configuration, where the wing geometry is deformed at the intersection with the fuselage, while the fuselage is kept fixed. Finally, in section 5, some conclusions and future work are suggested.

2 NON-UNIFORM RATIONAL B-SPLINES (NURBS)

Non-Uniform Rational B-Splines (NURBS) is a flexible parameterization extensively employed by Computer Aided Design (CAD) tools to represent the aerodynamic surface of an aircraft. Furthermore, they are also the standard of geometry format definition, such as in International Graphics Exchange Specification (IGES) interchange files. Using the underlying CAD definition significantly reduces the integration effort necessary to carry out multidisciplinary design. Usually it is no possible to define the aerodynamic surface of an aircraft as a continuous surface from the geometry and several NURBS patches are employed to assemble different sections.

The link of the computational grid, employed for simulations, and the CAD geometry, defined by NURBS patches, requires the knowledge of the parametric coordinates $\{\xi, \eta\}$ of each computational grid surface vertex. Given the displacements of the control points, it is possible to calculate the deformed geometry spatial coordinates from the parametric ones. These parametric coordinates remain constant, except at intersections between two NURBS surface patches, where further treatment is required.

The proposed methodology comprises the following steps [1]: First, the geometric definition is extracted from the IGES file, as a collection of several NURBS patches. Then, the parametric coordinates of the surface vertex involved in the deformation are calculated; surface vertex at intersections should have two pair of parametric coordinates, one for each intersecting NURBS. Once deformations of the geometry are applied to the NURBS patches, Cartesian coordinates of surface vertex are calculated and detect if there is a moving intersection, which requires to rearrange the parametric coordinates in order to match the new intersection. Then, a surface deformation algorithm is applied using the new intersection as boundary. Finally, the deformation is propagated to the volumetric grid.

2.1 Mathematical background

From the mathematical point of view, NURBS surfaces are parametric representations defined as:

$$S(\xi,\eta) = \frac{\sum_{i}^{I}\sum_{j}^{J} U_{i,p}(\xi)V_{j,q}(\eta)\,\omega_{ij}\,C_{ij}}{\sum_{i}^{I}\sum_{j}^{J} U_{i,p}(\xi)V_{j,q}(\eta)\,\omega_{ij}} \tag{1}$$

where $\{\xi,\eta\}$ are the parametric coordinates, U and V are the basis functions of orders p and q respectively, while C_{ij} are the control points and ω_{ij} the weights. One of the most effective expression to calculate the basis functions is through a recursive algorithm, which in some literature is referred to as the De Boor's algorithm:

$$U_{i,1}(\xi) = \begin{cases} 1 & \text{if } u_i \le \xi < u_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$U_{i,k}(\xi) = \frac{(\xi - u_i)U_{i,k-1}}{u_{i+k-1} - u_i} + \frac{(u_{i+k} - \xi)U_{i+1,k-1}}{u_{i+k-1} - u_{i+1}}$$

The terms u_i are coefficients from the so called knot vector, which is a sequence of real numbers that frequently have the form $\{\underbrace{0,\ldots,0}_{p+1},\ldots,u_i,\ldots,\underbrace{1\ldots,1}_{p+1}\}$.

2.2 Treatment of the intersection

Surface vertices of the computational grid at joints and intersections may belong to several NURBS surface patches, S_a and S_b, and are represented by two pairs of parametric coordinates $\{\xi_a,\eta_a\}$ and $\{\xi_b,\eta_b\}$, one for each NURBS, although they correspond to the same spatial coordinates, verifying:

$$S_a(\xi_a,\eta_a) - S_b(\xi_b,\eta_b) = 0 \tag{3}$$

To simplify the notation, let us call P and Q the spatial coordinates of the vertex calculated from the first and second pair of parametric coordinates respectively, as is shown in figure 1. After a movement of the intersection, these spatial coordinates no longer represent the same location; it can be seen as the vertex has been virtually splitted into two definitions. The goal is to recalculate the parametric coordinates to match the new intersection.

For relatively small deformations, the new parametric coordinates can be efficiently computed with a Newton-Raphson algorithm:

$$\{\xi_a,\eta_a\}^{n+1} = \{\xi_a,\eta_a\}^{n} - \frac{f}{f'} \tag{4}$$

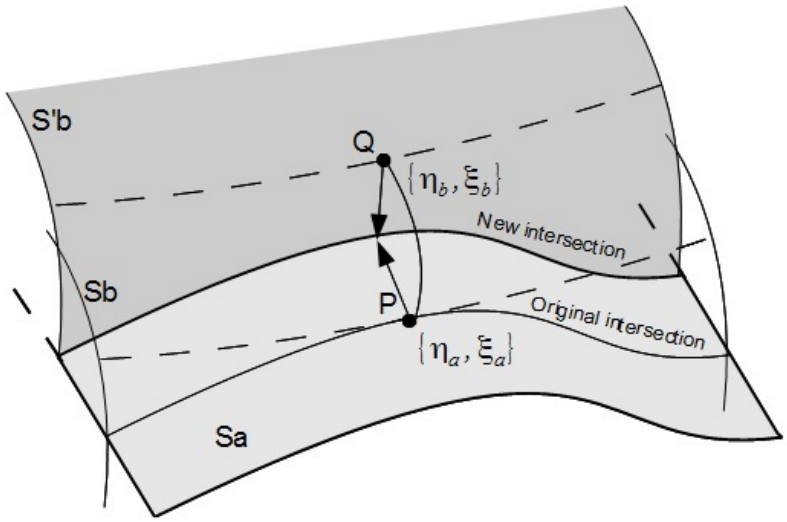

Figure 1: A vertex that originally was at the intersection is represented by two pairs of parametric coordinates $\{\xi_a, \eta_a\}$ and $\{\xi_b, \eta_b\}$, which correspond the spatial coordinates P and Q that initially represent the same location.

In order to improve the robustness, a two step algorithm is employed, where alternatively one pair of parametric coordinates are fixed.

$$f = ||S_a(\xi,\eta) - S_b(\xi,\eta)||^2 = ||P - Q||^2 \tag{5}$$

$$f' = \begin{pmatrix} f'_\xi \\ f'_\eta \end{pmatrix} = 2 \begin{pmatrix} \dfrac{\partial S(\xi,\eta)}{\partial \xi} * (P-Q) \\ \\ \dfrac{\partial S(\xi,\eta)}{\partial \eta} * (P-Q) \end{pmatrix}$$

where we have defined the operator $\bar{u} * \bar{v} = (u_1 v_1, u_2 v_2, u_3 v_3)$ which leads f' a 2×3 matrix. Multiplying f' by its transpose and simplifying in (4) leads to the expression

$$\Delta\xi = \frac{\left[(P-Q)\cdot\dfrac{\partial S(\xi,\eta)}{\partial \xi}\right]\left|\left|\dfrac{\partial S(\xi,\eta)}{\partial \xi}\right|\right|^2 - \left[(P-Q)\cdot\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right]\left[\dfrac{\partial S(\xi,\eta)}{\partial \xi}\cdot\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right]}{2d} \tag{6}$$

$$\Delta\eta = \frac{\left[(P-Q)\cdot\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right]\left|\left|\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right|\right|^2 - \left[(P-Q)\cdot\dfrac{\partial S(\xi,\eta)}{\partial \xi}\right]\left[\dfrac{\partial S(\xi,\eta)}{\partial \xi}\cdot\dfrac{\partial S(\xi,\eta)}{\partial \eta}\right]}{2d}$$

$$d = \left|\left|\frac{\partial S(\xi,\eta)}{\partial \xi}\right|\right|^2 \left|\left|\frac{\partial S(\xi,\eta)}{\partial \eta}\right|\right|^2 - \left[\frac{\partial S(\xi,\eta)}{\partial \xi}\cdot\frac{\partial S(\xi,\eta)}{\partial \eta}\right]^2$$

In the above expression, the symbol $\cdot$ is the dot product and $||\bar{a}||^2 = \bar{a} \cdot \bar{a}$.

There are two pairs of parametric coordinates $\{\xi_a, \eta_a\}$ and $\{\xi_b, \eta_b\}$ to be calculated, while there are only three equations available, one for each Cartesian coordinate. Each iteration, the above algorithm projects the line PQ onto the derivatives of each NURBS panel, until it finds the intersection. The initial values are the initial parametric coordinates before the deformation, which has been either provided by the grid generator or previously calculated using an inversion point algorithm.

2.3 Inversion point

Parametric coordinates from the computational grid are calculated as the projection of the vertex P to the NURBS surface S that minimizes the distance [2]:

$$\min \left\{ \left|\left| S(\xi, \eta) - P \right|\right|^2 \right\} \tag{7}$$

which leads to a similar expression obtained in equation (7). The essential difference is to provide a suitable initial value, which is critical for the robustness of the algorithm, because the presence of kinks and discontinuities, e.g. sharp edges, may cause the algorithm to fail leading to collapsed computational surface elements. One approach consists on calculating the projection of the vertex to the control polyhedra; this is the surface formed by the control points or the equivalent second order NURBS surface. Alternatively, the projection can be calculated as the intersection of the vertex surface normal, if a reliable normal value is available. Finally, a computationally expensive, but reliable method, consists on raster all the mid values of the knot vector.

2.4 Surface deformation

Once the parametric coordinates of the surface vertex at the intersection are recalculated, the surface grid should be adapted. The approach implemented is based on minimization of a linear elasticity analogy; seeing the elements of the grid connected with elastic edges

$$\bar{a} = \frac{\sum_i (\bar{x}_i - \bar{x}_i^0)\phi(\bar{x}_i, \bar{a})}{\sum_i \phi(\bar{x}_i, \bar{a})} \tag{8}$$

In the above expression, $\bar{a}$ is the vertex parametric coordinates, which is connected to $\bar{x}_i$ nodes. The notation $\bar{x}^0$ indicates the original position of the node, while the term $\phi(\bar{x}_i, \bar{a})$ is an arbitrary function that represents the elasticity module. For the present test case we have used the following well known value that works well

$$\phi(\bar{x}_i, \bar{a}) = \left|\left| \bar{x}_i^0 - \bar{a}^0) \right|\right|^{-2} \tag{9}$$

More elaborated expressions can be suggested to preserve the shape and/or the volume of the finite elements. The system of equations generated, can be solved iteratively with a Jacobi algorithm, until the residual converge to zero $(\bar{x}_i^{t+1} - \bar{x}_i^t) \to 0$, as follows:

$$\bar{a}^{t+1} = \frac{\sum_i (\bar{x}_i^{t+1} - \bar{x}_i^0)\phi(\bar{x}_i, \bar{a})}{\sum_i \phi(\bar{x}_i, \bar{a})} + \bar{a}^0 \tag{10}$$

In this approach, by performing the adaptation on the parametric coordinates, the surface vertex are guaranteed to be on the geometric CAD definition defined by the NURBS patches. This method is robust, easy to implement and to parallelize; on the downside, it has only first order convergence.

3 INTERPOLATION BASED ON RADIAL BASIS FUNCTIONS

Once the geometry has been modified we can apply mesh move techniques to transfer the deformation from the geometry to the whole surface grid. Although, many different techniques exist to interpolate a finite set of values, any particular choice will be strongly influenced by the requirements imposed by the underlying physical or mathematical problem. Common desirable properties of interpolation techniques are robustness, efficiency and monotonicity, where the last property is of special importance in order to avoid spurious oscillations in the continuous function to be reconstructed, and hence in the interpolated values. Radial basis functions (RBFs) have become a well-established method for interpolation of both scattered and gridded data [[3], [4], [5]].

The interpolation problem can be formulated in the following way. Given a finite set of d-dimensional centers $\{\bar{x}_1^s, \bar{x}_2^s, \ldots, \bar{x}_{N_s}^s\}$ and its displacement field $\{h_1^s, h_2^s, \ldots, h_{N_s}^s\}$ we aim to obtain a new displacement field $\{h_1^a, h_2^a, \ldots, h_{N_a}^a\}$ to transform in a smooth and regular way the set of evaluation nodes $\{\bar{x}_1^a, \bar{x}_2^a, \ldots, \bar{x}_{N_a}^a\}$. The method consists in building a continuous spatial distribution $h(\bar{x})$ using the discrete values $\bar{x}_i^s$

$$h(\bar{x}) = \sum_{i=1}^{N_s} \omega_i \Phi(||\bar{x} - \bar{x}_i^s||) + \Pi(\bar{x}) \tag{11}$$

where the function Φ is a fixed basis function which is radial respect to the Euclidean distance and Π is a low degree d-variate polynomial. The coefficients ω_i are calculated by requiring exact recovery of the original function over the centers

$$h_i^s \equiv h(\bar{x}_i^s) \qquad \forall i = 1, \ldots, N_s \tag{12}$$

If the polynomial term is included, the system must be completed by the additional zero condition

$$\sum_{i=1}^{N_s} \omega_i q(\bar{x}_i^s) = 0 \tag{13}$$

for all polynomials with a degree $\deg(q) \leq \deg(\Pi)$.

By imposing conditions (12) and (13) in (11) the following linear system is obtained

$$\boldsymbol{U}_s = \boldsymbol{C}_{ss}\boldsymbol{\omega} \tag{14}$$

where $\boldsymbol{C}_{ss}$ is the interpolation matrix.

Then, the displacement of evaluation nodes is given by the following expression

$$h_j^a = h(\bar{x}_j^a) = \sum_{i=1}^{N_s} \omega_i \Phi(||\bar{x}_j^a - \bar{x}_i^s||) + \Pi(\bar{x}_j^a) \qquad \forall j = 1, \ldots, N_a \tag{15}$$

or, using matrix notation

$$\boldsymbol{U}_a = \boldsymbol{A}_{as}\boldsymbol{\omega} \tag{16}$$

From a computational point of view, the interpolation evaluation can be made following two different approaches: (a) Calculation of the coupling matrix $\boldsymbol{G} = \boldsymbol{A}_{as}\boldsymbol{C}_{ss}^{-1}$ and (b) Solving the linear system given by (14) and computing system (16). Because of the nature of the problem a one order polynomial has been used and the volume spline as radial basis function

$$\Phi(||\bar{x}||) = ||\bar{x}|| \tag{17}$$

which leads to very good results when it is applied to global problems [5].

4 TEST CASE

The test case selected is the DLR F6 wing body configuration, extracted from the 2$^{\text{nd}}$ AIAA CFD Drag Prediction Workshop [6], and is shown in figure 2. To illustrate the capabilities to adapt the surface grid with both approaches, the upper side of the wing is deformed at the intersection, while the fuselage remains fixed. The whole surface grid is composed of 56.322 vertices, and 112.644 triangle elements. The wing is defined by seven NURBS patches and the central fuselage is defined by one NURBS. The patch that represents the upper side of the wing section that intersects with the fuselage is modified by an arbitrary bump deformation.

$$C_y = a\,\frac{C_x - C_x^0}{C_x^1 - C_x^0} \tag{18}$$

where C^1 and C^0 are the control points at the leading and trailing edge respectively. That parameter a is adjusted so the maximum deformation approximately doubles the original distance to the chord.

Grid adaptation is applied to both fuselage and wing surfaces, although the deformation is more severe at the fuselage and requires more extensive rearrange, while the wing surface grid is slightly deformed. So, for illustration purposes, only the fuselage grid is shown. The baseline configuration is shown in figure 3.

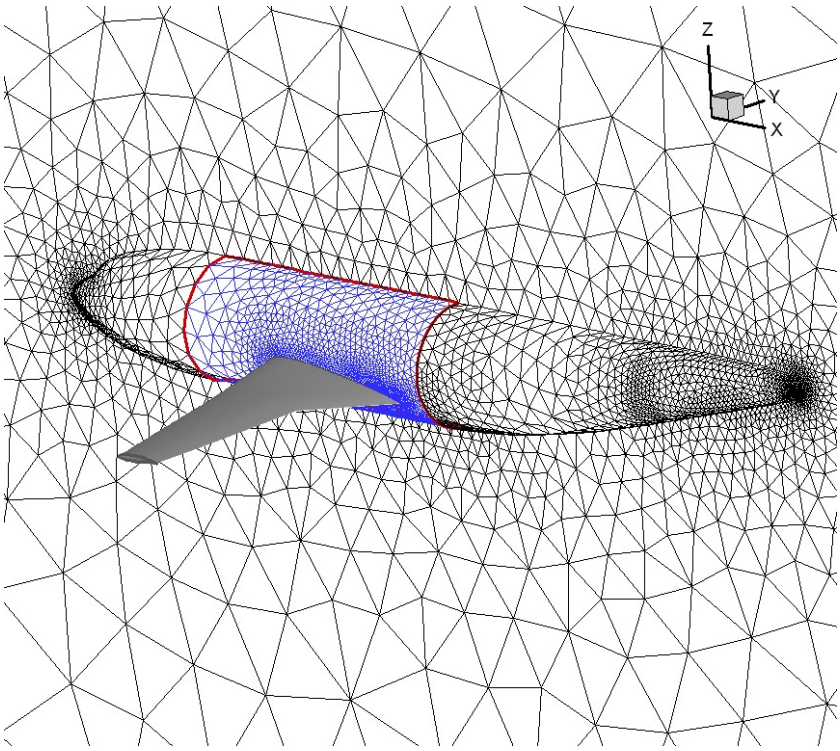

Figure 2: F6 wing-body computational grid.

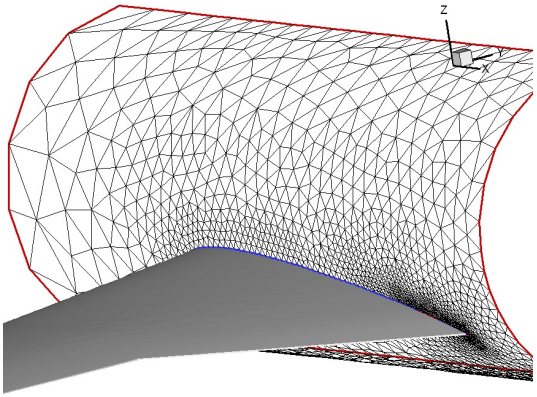

Figure 3: Detail of the wing-body grid, before the deformation.

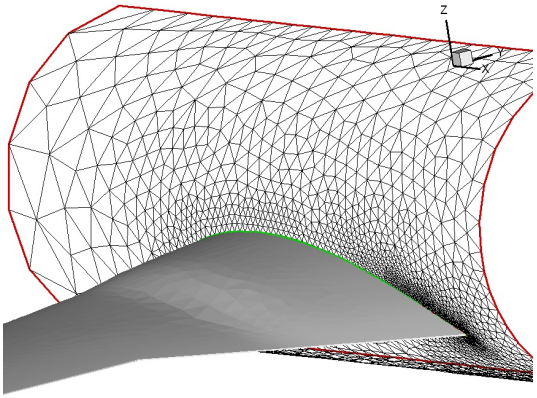

Figure 4: Surface mesh adaptation using NURBS.

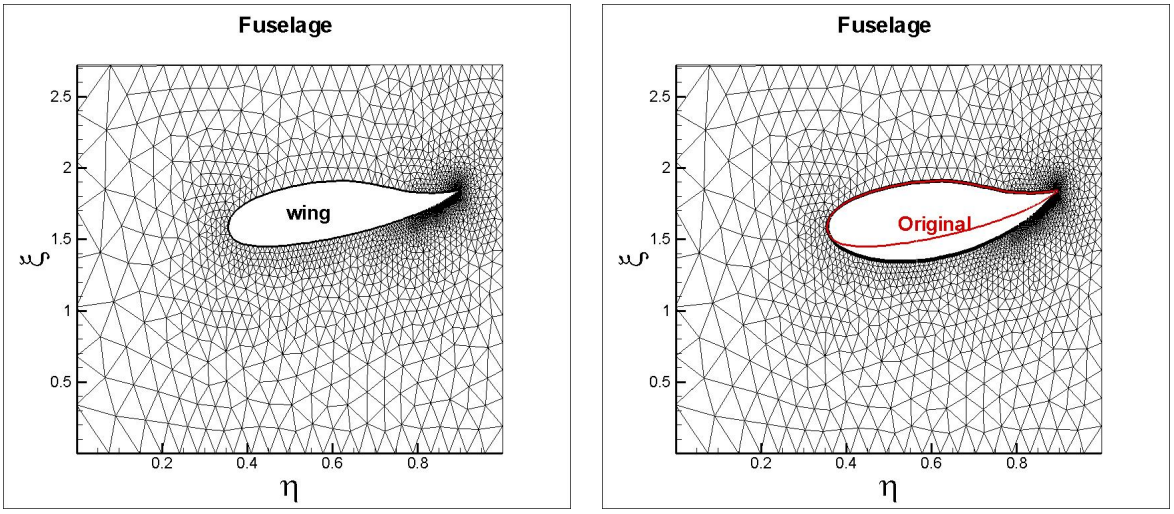

Figure 5: Surface grid in parametric coordinates of the fuselage, where the baseline configuration is shown (left) and the adaptation (right).

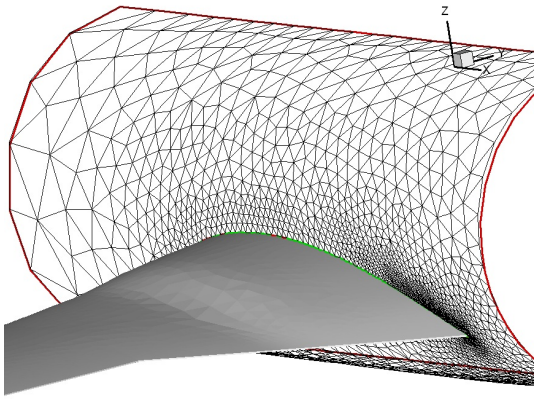

Figure 6: Surface mesh adaptation using Radial Basis Functions.

5 CONCLUSIONS AND FURTHER WORK

The ability to adapt the computational grid upon deformations of the geometry is a very useful technique that avoids the need of remesh, which is also suitable for automatic design tools. Two approaches have been tested: based on NURBS and Radial Basis Functions interpolation. Both methods provide suitable computational grids, where the shape of the elements are well preserve.

The strong point of the NURBS is that preserves the original CAD geometry and it is

not computationally expensive. On the downside, is that deformations have to be applied to the CAD definition and requires the parametric coordinates of the surface vertex. Additionally, it is not clear how to propagate the surface deformation through different NURBS patches.

On the other hand, Radial Basis Functions interpolation provides the ability to adapt directly the computational grid, without any knowledge of the original CAD file. On the downside, its main disadvantage is that it may deform other components; in this example, the fuselage was slightly deformed from its original geometry.

REFERENCES

[1] Martin, M. J., Valero, E., Andres, E. and Lozano, C. Treatment of surface intersections for gradient-based aerodynamic shape optimization. *European Conference for Aeronautics & Space Sciences*. July, 2013 Germany.

[2] Martin, M. J., Andres, E., Widhalm, M., Bitrian, P. and Lozano, C. Non-Uniform rational B-splines-based aerodynamic shape design optimization with the DLR TAU code. *Journal of Aerospace Engineering* (2012) **226**(10):1225-1242.

[3] Beckert, A., Wendland, H. Multivariate interpolation for fluid-structure-interaction problems using radial basis functions. *Aerospace Science and Technology* (2001), **5**:125–134.

[4] Rendall T.C.S., Allen C.B. Unified fluid-structure interpolation and mesh motion using radial basis functions. *Int. J. Numer. Meth. Engineering* (2008), **74**:1519–1559.

[5] Cordero-Gracia, M., Gómez M., Ponsin, J. and Valero E. An interpolation tool for aerodynamic mesh deformation problems based on octree decomposition. *Aerospace Science and Technology* (2012), **23**:93 – 107.

[6] Brodersen, O. and Stumer, A. Drag Prediction of Enine-Airframe Inerference Effects Using Unstructured Navier-Stokes Calculations. *AIAA Applied Aerodynamics Conference* 11-14 June 2001, Anaheim, California.