Integrated geometry parametrization and grid movement using B-spline meshes

Jason E. Hicken* and David W. Zingg †

Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, M3H 5T6, Canada

We propose an algorithm that integrates geometry parametrization and mesh movement using the control points of a B-spline mesh. An initial mesh is created using B-spline volumes in such a way that the control points mimic a coarse grid. The control points corresponding to the surface nodes are adopted as the design variables. Mesh movement is achieved by applying a standard movement algorithm to the coarse B-spline grid. For example, we have developed a semi-algebraic scheme in which the B-spline control points are updated using the equations of linear elasticity, and the new mesh is regenerated algebraically. We illustrate the approach with a few examples, including a flat plate morphing into a blended-wing-body configuration.

I. Introduction

Climate change and increased oil prices have renewed interest in unconventional aircraft designs that dramatically improve fuel efficiency — the blended-wing body is an example. Multi-disciplinary optimizations of these concepts typically start from a baseline design and make small shape and structural changes; see, for example, Liebeck,[1] Campbell et al.,[2] and Le Moigne and Qin.[3] Modifying a baseline design with small changes is a sensible approach for refining a design; however, for unconventional designs it would be preferable to make no assumptions regarding the shape and structure of the aircraft. In this case, both the shape parametrization and mesh movement need to be highly flexible and robust.

Shape parametrizations fall into two broad categories: free-form and feature-based. Examples of free-form parametrizations include the surface mesh points and Bézier or B-spline control points. Feature-based parametrizations use conventional aircraft parameters such as span, twist, sweep, and taper. Many authors combine the two approaches; for example, Le Moigne and Qin[3] use a free-form Bézier parametrization for the airfoil sections together with twist and sweep variables. However, for investigations of unconventional designs, a strictly free-form parametrization is most flexible.

During the optimization process, the design variables are updated and change the shape of the aerodynamic surface. Consequently, the computational mesh must be adapted to conform to this new shape. Several algebraic and computational approaches have been proposed to move the mesh in response to the surface shape changes.

Algebraic grid movement can be applied to multi-block structured grids,[4,5] but this approach is limited to relatively small shape changes and a particular type of grid. Liu et al.[6] have proposed an algebraic mesh movement based on mapping the mesh to a Delaunay graph. They demonstrate that the Delaunay graph approach is robust to large shape changes, provided multiple increments are used; however, analysis of their method suggests that, in general, using multiple increments produces a discontinuous objective function gradient. This may limit the Delaunay graph approach to gradient-free optimizations.

Jakobsson and Amoignon[7] developed a promising mesh movement algorithm based on radial basis functions (RBFs); see also reference 8 where a similar approach is applied in the context of aeroelastic problems. In RBF mesh movement, the surface geometry displacements are interpolated into the interior. The interpolation coefficients are obtained by inverting a relatively large, dense system for each mesh movement. After the coefficients are determined, applying the interpolation is equivalent to a sparse matrix-vector multiplication. Depending on whether this sparse matrix is stored or recomputed, the RBF approach can be

*PhD Candidate, AIAA Student Member

†Professor and Director, Tier 1 Canada Research Chair in Computational Aerodynamics, Associate Fellow AIAA

computationally or memory intensive. In addition, the mesh may not interpolate the geometry unless the geometry is also parametrized using radial basis functions.[7]

Batina[9] introduced the spring analogy method, which models the mesh edges as springs; this method was later modified to include torsional springs.[10] While more robust than most algebraic algorithms, the spring analogy method can produce negative cell volumes during large, 3-dimensional shape changes.[11] Johnson and Tezduyar[12] demonstrated that the equations of linear elasticity can be used to achieve robust mesh movements, even for large shape changes. This approach has been used successfully for aeroelastic problems[13] and aerodynamic optimization.[11] Unfortunately, the equations of linear elasticity are typically less diagonally dominant than the spring analogy equations, so the elasticity approach tends to be more computationally expensive.

Experience with algebraic and differential-equation based mesh movement algorithms suggests that efficiency and robustness are mutually exclusive. We propose a compromise: apply a robust mesh movement algorithm to a coarse grid and regenerate the fine grid algebraically from the coarse one. In addition, the points on the surface of the coarse grid can be chosen as the design variables, integrating mesh movement with geometry parametrization. We demonstrate this idea by using B-spline volume grids[14,15] with the control points defining the coarse mesh.

Section II reviews B-spline concepts and their extension to volume grids. We apply linear elasticity mesh movement to the coarse grid; this algorithm is outlined in Section III. The parametrization and mesh movement are demonstrated in Section IV with several examples. Our conclusions are summarized in Section V.

II. B-spline Mesh Concepts

A. B-Spline Curves and Basis Functions

A B-spline curve of order p is a weighted sum of basis functions;

$$\mathbf{C}(\xi) = \sum_{i=1}^{N} \mathbf{B}_i \mathcal{N}_i^{(p)}(\xi), \qquad 0 \le \xi \le 1. \tag{1}$$

The points $\{\mathbf{B}_i \in \mathbb{R}^d, i = 1, \ldots, N\}$ are called de Boor control points, or simply control points. The functions $\mathcal{N}_i^{(p)}(\xi)$ are the B-spline basis functions of order p; they are $p-1$ degree spline polynomials. The polynomials making up the splines are joined at non-decreasing knot locations, $\{T_i, i = 1, \ldots, p+N\}$, such that the basis is C^{p-2} continuous at the knots. The parameter range for the curve is given by $[0,1] = [T_1, T_{p+N}]$; other ranges can be scaled to this domain. The basis functions can be defined recursively using the parameter value and the knots:

$$\begin{aligned} \mathcal{N}_i^{(1)}(\xi) &= \begin{cases} 1 & \text{if } T_i \le \xi < T_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \\ \mathcal{N}_i^{(p)}(\xi) &= \left(\frac{\xi - T_i}{T_{i+p-1} - T_i} \right) \mathcal{N}_i^{(p-1)}(\xi) + \left(\frac{T_{i+p} - \xi}{T_{i+p} - T_{i+1}} \right) \mathcal{N}_{i+1}^{(p-1)}(\xi). \end{aligned} \tag{2}$$

In general, a B-spline curve can be controlled using the order of its basis, the number and position of its de Boor points, and its knot values and their multiplicity. Unless stated otherwise, we will use fourth-order basis functions (cubic splines) and drop the superscript: $\mathcal{N}_i(\xi) \equiv \mathcal{N}_i^{(4)}(\xi)$. This choice of basis is common, and provides a good compromise between continuity and computational speed. We will also use open knot vectors of the form $\{0,0,0,0,T_5,T_6,\ldots,T_{N-1},1,1,1,1\}$. The multiplicity of $p = 4$ at the ends of the knot vector ensures that the curve passes through $\mathbf{B}_1$ and $\mathbf{B}_N$. Figure 1 plots the fourth order B-spline basis functions for the knot vector $\{0,0,0,0,0.1,0.2,0.5,1,1,1,1\}$.

B. 1-Dimensional B-Spline Meshes

The B-spline curve equation (1) is typically used to define a mapping of the form $\mathbf{C} : [0,1] \to \mathcal{P}$ where $\mathcal{P}$ is a subset of $\mathbb{R}^2$ or $\mathbb{R}^3$. However, we can also use (1) to map one interval of the real line to another. Such 1-dimensional B-spline curves provide a suitable analogy and introduction to our ultimate application: volume mesh generation and movement.

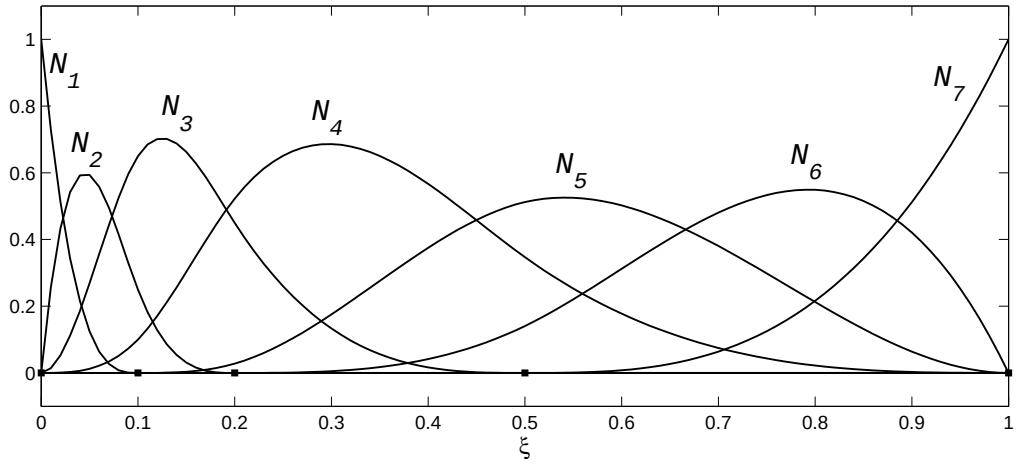
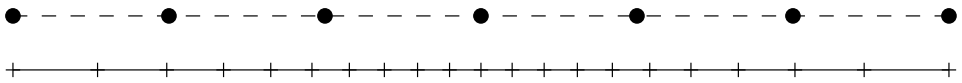
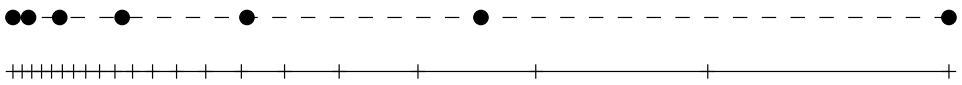


Figure 1. Example B-spline basis for the knot vector $\{0, 0, 0, 0, 0.1, 0.2, 0.5, 1, 1, 1, 1\}$; the knot locations are denoted by the square symbols.



(a) initial B-spline mesh and control points (solid circles).



(b) mesh perturbed using control points.

Figure 2. Example showing influence of control points on the grid spacing of a 1-dimensional B-spline mesh.

Consider a set of uniformly spaced 1-dimensional control points $\{B_i, i = 1, N\}$ and a uniform open knot vector, i.e. a knot vector where the internal knots are evenly spaced between the end knots. The B-spline "curve" constructed using these control points and knots produces a mapping $C : [0, 1] \rightarrow [B_0, B_N]$. If we discretize the parameter space into L points, $\{\xi_1, \xi_2, \ldots, \xi_L\}$, then the image of these points is a 1-dimensional B-spline mesh. Figure 2(a) provides an example of a B-spline mesh and its control points. In this particular example, the control points, knots, and parameter values are uniformly spaced. The resulting mesh itself is not uniform because of the asymmetry of the basis functions. However, a uniform mesh can be obtained from uniformly spaced control points by changing the parameter values.[16]

We can adapt the B-spline mesh by altering the positions of the control points. Such an adaptation is shown in Figure 2(b) for illustrative purposes. Notice that the control points act like a coarse mesh; when they move closer together, the spacing on the actual mesh also decreases.

The coarse-grid behaviour of the control points applies not only to mesh spacing but to space curves as well. B-spline curves of order p are constrained to lie within the convex hull of p neighbouring control points, and they also obey a variation diminishing property.[17] Therefore, the B-spline control points approximate the 2-dimensional or 3-dimensional space curves they define. Indeed, as more control points are added, using knot insertion for example, the control points converge to the geometry they define. This coarse grid property of the control points is particularly useful when moving B-spline volume meshes.

C. B-Spline Volume Meshes and Modified Basis Functions

B-spline surfaces and volumes can be created using tensor products. We will focus on B-spline volumes since surfaces can be considered a special case. Tensor product B-spline volumes are well suited to structured grid generation;[14,15] however, B-spline volumes can also be generated by generalizing triangular Bézier patches, which may be of interest to unstructured grid users.

A B-spline tensor product volume is a mapping from the cubic domain $\mathcal{D} = \{\boldsymbol{\xi} = (\xi, \eta, \zeta) \in \mathbb{R}^3 | \xi, \eta, \zeta \in [0,1]\}$ to $\mathcal{P} \in \mathbb{R}^3$ and is defined by

$$\mathbf{x}(\boldsymbol{\xi}) = \sum_{i=1}^{N_i} \sum_{j=1}^{N_j} \sum_{k=1}^{N_k} \mathbf{B}_{ijk} \mathcal{N}_i(\xi) \mathcal{N}_j(\eta) \mathcal{N}_k(\zeta). \tag{3}$$

A B-spline volume mesh is produced by simply discretizing the domain $\mathcal{D}$. The parameters ξ, η, and ζ do not need to be discretized in a uniform way. Indeed, nonuniform parameter spacing is usually necessary for precise control of mesh spacing in physical space. If a flow solver requires uniform meshing spacing in parameter space, an intermediate mapping is implied.[a]

The basis functions appearing in (3) can be generalized to permit spatially varying knot functions, also called curved knot lines.[18] The usefulness of such knot functions will be explained below. Consider the basis functions in the ξ-direction, and suppose the knots are functions of η and ζ. Then, the modified basis functions are given by

$$\begin{aligned} \mathcal{N}_i^{(1)}(\xi; \eta, \zeta) &= \begin{cases} 1 & \text{if } T_i(\eta,\zeta) \le \xi < T_{i+1}(\eta,\zeta), \\ 0 & \text{otherwise.} \end{cases} \\ \mathcal{N}_i^{(p)}(\xi; \eta, \zeta) &= \left(\frac{\xi - T_i(\eta,\zeta)}{T_{i+p-1}(\eta,\zeta) - T_i(\eta,\zeta)} \right) \mathcal{N}_i^{(p-1)}(\xi; \eta, \zeta) \\ &\quad + \left(\frac{T_{i+p}(\eta,\zeta) - \xi}{T_{i+p}(\eta,\zeta) - T_{i+1}(\eta,\zeta)} \right) \mathcal{N}_{i+1}^{(p-1)}(\xi; \eta, \zeta). \end{aligned} \tag{4}$$

Analogous definitions apply to the basis functions $\mathcal{N}_j^{(p)}(\eta; \zeta, \xi)$ and $\mathcal{N}_k^{(p)}(\zeta; \xi, \eta)$.

For fixed (η, ζ), the modified basis function (4) reduces to the standard definition (2); hence, the modified basis retains C^{p-2} continuity at the knots in the ξ direction. Although less obvious, the modified basis is also C^{p-2} in the η- and ζ-directions, provided $T_i(\eta, \zeta) \in C^{p-2}$ and the internal knots have a multiplicity at most one. This result is a consequence of the chain rule and the smoothness of the derivative of a B-spline with respect to its knots.[19–21]

The internal knots of the modified basis functions must be strictly increasing and sufficiently smooth, but the user is otherwise free to choose the functional form. For simplicity, we use bilinear knots of the form

$$T_i(\eta,\zeta) = T_{i,(0,0)}(1-\eta)(1-\zeta) + T_{i,(1,0)}\eta(1-\zeta) + T_{i,(0,1)}(1-\eta)\zeta + T_{i,(1,1)}\eta\zeta. \tag{5}$$

The four constants $T_{i,(0,0)}$, $T_{i,(1,0)}$, $T_{i,(0,1)}$, and $T_{i,(1,1)}$ denote the knot values at the (η, ζ) edges of the parameter space. Again, similar knot definitions are used for $T_i(\zeta, \xi)$ and $T_i(\xi, \eta)$. Figure 3 shows an example modified basis function. The salient feature is the changing basis location.

D. Approximating Grids Using B-spline Volume Meshes

We have described how B-spline volumes can be used, in theory, to define mappings from computational space to physical space. Many practical issues remain unanswered. For example, how do we choose the surface control points such that the mappings conform to the geometry of interest? This is certainly an important question when a mesh is generated for analysis, but it is less important in preliminary design where the geometry of interest is the optimized one. Nevertheless, an initial geometry can be approximated to engineering tolerances using B-spline meshes.

[a] In practice, finite-difference and other mapping-based solvers use the grid coordinates only, so the details of the B-spline parameters are not important; however, the smoothness of the mapping must be consistent with the order of the scheme being used.

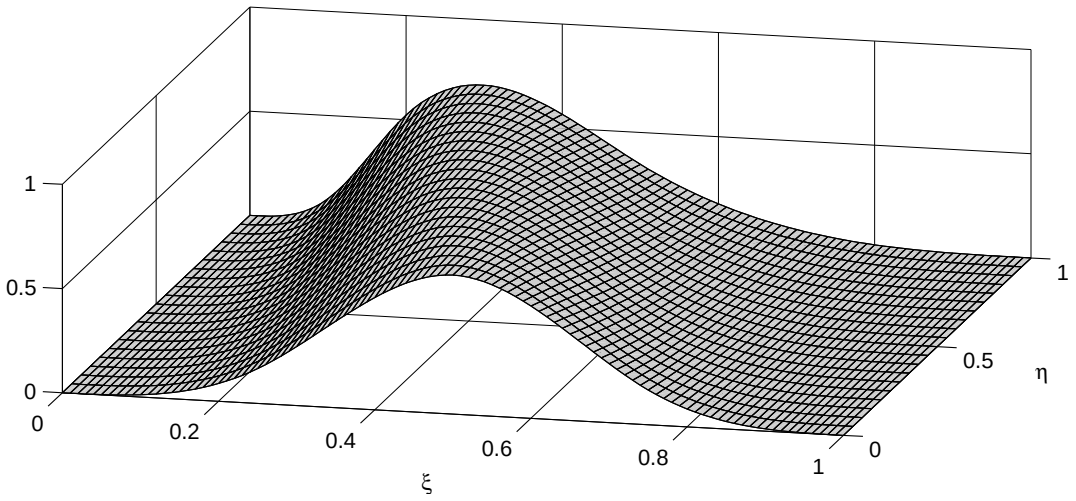


Figure 3. Example 2-dimensional modified basis, or, equivalently, a 3-dimensional modified basis for fixed ζ.

Issues related to mesh generation also need to be addressed. How should the volume control points and parameter values be determined to obtain a high quality mesh? Should the domain be broken into multiple mappings using a semi-structured approach? These are open-ended questions, and to provide thorough and exhaustive answers would go beyond the scope of this paper. For the purposes of demonstrating the B-spline parametrization and mesh movement, we propose fitting existing, canonical grids to determine the initial control point positions and the (ξ, η, ζ) parameter values.

A least-squares fit with parameter correction[16] is often used to find spline curve and surface approximations to data points. This algorithm can easily be extended to B-spline volume approximations of a grid, as we now demonstrate.

Each block of the structured grid is associated with a B-spline volume. For each B-spline volume, the user chooses the number of control points and the B-spline order in the parameter directions (ξ, η, ζ); at interfaces where blocks meet, the number of control points and order must be consistent to ensure continuity.

First, parameter values $\boldsymbol{\xi}_{q,r,s}$ are calculated for each point in the grid $G = \{\mathbf{x}_{q,r,s} | q = 1, \ldots, L_q, r = 1, \ldots, L_r, s = 1, \ldots, L_s\}$. We use a chord-length parametrization to be consistent with the choice of knot vectors (see below), but other parametrizations are possible.[22] To illustrate the chord length parametrization, consider the parameter $\xi_{q,r,s}$ along the grid line $\{q = 1, \ldots, L_q, r = r_0, s = s_0\}$:

$$\begin{aligned} \xi_{1,r_0,s_0} &= 0 \\ \xi_{q,r_0,s_0} &= \frac{1}{\Xi} \sum_{t=1}^{q-1} \|\mathbf{x}_{q+1,r_0,s_0} - \mathbf{x}_{q,r_0,s_0}\|, \qquad q = 2, \ldots, L_q, \end{aligned}$$

where the total arc length is approximated by

$$\Xi = \sum_{t=1}^{L_q - 1} \|\mathbf{x}_{q+1,r_0,s_0} - \mathbf{x}_{q,r_0,s_0}\|.$$

Next, the spatially varying knot vectors must be determined. The importance of the knot vector in fitting B-spline curves is well documented.[22,23] When fitting surface or volume data points with conventional algorithms, there are an infinite number of curves being fit with the same knot vector. By allowing the knots to vary, it should be possible to improve the B-spline approximation of surface and volume data. We use the spatially varying bilinear knots defined by (5), with the knots on each parameter edge defined by a chord length parametrization. We have found that chord-length-based knots produce control meshes that approximate the mesh spacing of a coarse grid, a feature that is important for the mesh movement algorithm.

The specific formula for the edge knots will be demonstrated using the edge $\{(\xi, 0, 0)|\xi \in [0, 1]\}$. For this edge we have

$$T_{i,(0,0)} = (\xi_{q+1,1,1} - \xi_{q,1,1})[\bar{q}_i - (q-1)] + \xi_{q,1,1}, \quad q - 1 \leq \bar{q}_i \leq q$$

where

$$\bar{q}_i = \left(\frac{L_q - 1}{N_i + 1 - p}\right)(i - p).$$

These edge knots are constructed to have approximately the same number of parameter values in each knot interval, $[T_{i+1}, T_i], i = p, N_i - 1$. Consequently, the knots obtained using the above formula have a chord length parametrization because the parameters use a chord length parametrization.

Finally, least-squares systems are solved to determine the control points that best approximate the grid; see reference 22, for example. Least-squares systems are solved sequentially for the edge, face, and internal grid points. This ordering ensures that adjacent blocks have consistent grid point locations. If necessary, the iterative parameter correction algorithm of Hoschek[16] can be applied to improve the fit by adjusting the parameter values.

E. Applications of B-Spline Meshes

The control mesh is designed to mimic a coarse grid, so the surface control points provide a low dimensional approximation of the surface. This makes the surface control points a suitable choice for the design variables in shape optimization. The use of Bézier or B-spline control points in design optimization is not new;[24] the difference here is that the design variables are a subset of a volume control mesh.

In this paper, our focus is on B-spline parametrization and mesh movement as it pertains to design optimization; however, we would like to mention other possible uses of B-spline meshes:

- the surface control points can be used to control the shape of the wing in studies of unsteady aerodynamics;
- the volume control points can be used for mesh adaptation, rather than the individual grid points;
- the parameter space can be refined in a smooth way to achieve rigorous mesh convergence studies;
- hierarchical grids can easily be constructed for multigrid, and;
- grids and geometries suitable for high-order discretizations can be obtained with a suitable choice of B-spline basis order.

III. B-spline Grid Movement

A. Possible Grid Movement Algorithms

Once fitted using B-spline volumes, the grid can be manipulated using the control mesh of points $\{\mathbf{B}_{ijk}\}$. Suppose that a subset of control points on the surface, denoted by $\{\mathbf{B}_{ijk}|(i, j, k) \in S\}$, have been chosen as design variables. A very simple mesh movement algorithm is obtained by fixing the control points that are *not* in this subset. Although this fixes the internal control points, the grid points near the surface still move because the boundary B-spline basis functions extend into the interior. Such a mesh movement strategy may be useful if only small shape changes are necessary.

If larger shape changes are expected, then an algorithm is needed that moves the internal control points based on the surface control points. Essentially, any grid movement algorithm can be applied to the control points; algebraic, spring analogy, linear elasticity, radial basis functions, etc. For B-spline volume meshes, there are typically three orders of magnitude fewer control points than grid points, so the CPU time of the mesh movement becomes insignificant relative to the flow solution. For this reason we have chosen to use a robust, linear elasticity-based mesh movement.

B. Control Point Movement Based on Linear Elasticity

We model the control mesh as a linear elastic solid with stiffness controlled using a non-constant Young's modulus, E. The algorithm is very similar to the one used in Truong et al.;[25] only a brief outline of the method is given here.

The control points are updated according to the equilibrium equation

$$\frac{\partial\left(\tau_{ij}\right)}{\partial x_j} + f_i = 0, \tag{6}$$

where the stress tensor, τ_{ij}, is assumed to be linearly related to the Cauchy strain tensor, e_{ij}. We also assume the solid is isotropic and symmetric, and that displacements are small. Under these assumptions we have

$$\tau_{ij} = \frac{E}{1+\nu}\left(e_{ij} + \frac{\nu e_{kk}\delta_{ij}}{1-2\nu}\right)$$
$$e_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right).$$

Note, when the small displacement assumption is not valid, the problem can be broken into a sequence of mesh movement problems by moving the surface in increments. We elaborate on this below.

The governing equation (6) is discretized on the control mesh using the finite element method with trilinear elements. The force vector f_i is defined implicitly based on the displacement of known degrees of freedom, such as the surface and far-field control points. The resulting system of algebraic equations is of the form

$$\boldsymbol{\mathcal{M}} = K(\mathbf{b} - \mathbf{b}^{(0)}) - \mathbf{f} = 0 \tag{7}$$

where $\mathbf{b}$ is a block column vector of control point coordinates, $\mathbf{f}$ is the discrete force, and the stiffness matrix K is sparse, symmetric, and positive-definite. The initial control point coordinates are denoted by $\mathbf{b}^{(0)}$. We solve the linear system (7) using the conjugate gradient method preconditioned with ILU(p).[26] The system is solved to a relative tolerance of 10^{-12} with respect to the L^2 norm of the residual.

As noted above, when large shape changes are expected, the mesh movement can be applied in increments. Suppose m increments are used, with $\mathbf{b}_{\text{srf}}^{(0)}$ and $\mathbf{b}_{\text{srf}}^{(m)}$ denoting the initial and final locations of the surface control points, respectively. Then the intermediate values of the surface control points are determined by the linear interpolation

$$\mathbf{b}_{\text{srf}}^{(i)} = \frac{i}{m}\left(\mathbf{b}_{\text{srf}}^{(m)} - \mathbf{b}_{\text{srf}}^{(0)}\right) + \mathbf{b}_{\text{srf}}^{(0)}, \qquad i = 1, \ldots, m.$$

In addition, when increments are used, the stiffness matrix becomes a function of the control point coordinates at the previous level through the spatially varying Young's modulus (explained below). Thus, the linear equation for the control points at increment i has the form

$$\boldsymbol{\mathcal{M}}^{(i)} = K^{(i)}\left(\mathbf{b}^{(i-1)}\right)\left[\mathbf{b}^{(i)} - \mathbf{b}^{(i-1)}\right] - \mathbf{f}^{(i)} = 0, \qquad i = 1, \ldots, m. \tag{8}$$

Using increments helps improve the robustness of the mesh movement while maintaining linearity; note, however, this approach does have implications for gradient-based optimization,[25,27] since $K^{(i)}$ is a function of $\mathbf{b}^{(i-1)}$.

Element stiffness is controlled using a spatially varying Young's modulus. The Young's modulus is calculated at the beginning of the mesh movement, or the beginning of each increment if the movement is broken into smaller steps. The goal is to vary the element stiffness in such a way that mesh quality is maintained in critical regions of the grid, e.g. the boundary layer. First, we assume that $E_{\mathcal{E}} \propto V_{\mathcal{E}}^{-1}$, where $V_{\mathcal{E}}$ is the element volume; this ensures that small elements are stiffer than large elements. In regions where element volumes have similar magnitudes, we need an additional stiffening mechanism to maintain quality. Here we have chosen to use an orthogonality measure; hence, elements that are at risk of becoming more skewed have their stiffness increased. The orthogonality measure for an arbitrary element $\mathcal{E}$ is given by

$$\Phi_{\mathcal{E}} = \left(\prod_{\nu=1}^{8} \frac{1}{(\mathbf{u}_\nu \times \mathbf{v}_\nu)\cdot \mathbf{w}_\nu}\right)^2$$

where the index ν denotes the vertices of the element. The vectors $\mathbf{u}_\nu$, $\mathbf{v}_\nu$, and $\mathbf{w}_\nu$ are unit vectors parallel to the element edges that meet at vertex ν, and they form a right-handed system. Hence, the triple product $(\mathbf{u}_\nu \times \mathbf{v}_\nu) \cdot \mathbf{w}_\nu$ is positive for valid elements, equal to 1 for orthogonal vectors, and tends to zero as the vectors become coplanar. Following Bar-Yoseph,[28] the orthogonality measure is normalized by its value on the initial mesh. In summary, the Young's modulus for element $\mathcal{E}$ at increment i is given by

$$E_{\mathcal{E}}^{(i)} = \frac{\Phi_{\mathcal{E}}^{(i)}}{\Phi_{\mathcal{E}}^{(0)} V_{\mathcal{E}}^{(i)}}$$

IV. Examples

In this section, we demonstrate the proposed mesh movement and parametrization scheme with two examples. In the first example, we parametrize an existing shape — the ONERA M6 wing[29] — to test the algorithm in a typical design optimization setting. The second example is intended to mimic a conceptual design optimization and involves morphing a flat plate into a blended-wing body with winglets. In both cases the proposed mesh movement is compared with a node-based-linear-elasticity approach, i.e. the algorithm described in Section III.B is applied to individual nodes rather than control points. In all cases, the mesh movement is broken into $m = 5$ increments, and Poisson's ratio is fixed at -0.2.

To assess the mesh movement algorithms, we adopt an orthogonality measure similar to the one used to stiffen elements in the linear elasticity algorithm. Specifically, the quality for an element $\mathcal{E}$ is given by

$$Q_{\mathcal{E}} = \sqrt{\frac{1}{\Phi_{\mathcal{E}}}} = \prod_{\nu=1}^{8} (\mathbf{u}_\nu \times \mathbf{v}_\nu) \cdot \mathbf{w}_\nu. \tag{9}$$

It follows from this definition that elements with perfect orthogonality have $Q_{\mathcal{E}} = 1$ while highly skewed elements have $Q_{\mathcal{E}} \approx 0$. In the case of the B-spline mesh movement algorithm, $Q_{\mathcal{E}}$ is measured for elements on the interpolated mesh and not the control mesh.

The fitting process used to determine the B-spline control mesh and parameter spacing produces an approximation to the initial mesh. To quantify the quality of the fit of the initial mesh, we evaluate the root-mean-square (RMS) and L^∞ norm errors. The RMS error is based on the distances between corresponding nodes in the initial and fitted grids:

$$\mathrm{RMS}_{\mathrm{error}} = \sqrt{\frac{\sum_{q,r,s} \|\mathbf{x}_{q,r,s} - \mathbf{x}(\boldsymbol{\xi}_{q,r,s})\|_2^2}{\sum_{q,r,s} 1}}.$$

To be clear, the sum in the RMS error is over all nodes and all blocks. For the infinity norm errors, we report the maximum absolute error for each coordinate. The RMS and L^∞ norm errors for the two cases are listed in Table 1. These errors are acceptable for most engineering analyses. If tighter tolerances are required, more control points can be added or parameter correction — which was not used here — can be invoked.

Table 1. Errors between initial and B-spline grids

case	$\mathrm{RMS}_{\mathrm{error}}$	$\|\Delta x\|_\infty$	$\|\Delta y\|_\infty$	$\|\Delta z\|_\infty$
ONERA M6	4.04×10^{-4}	2.01×10^{-3}	1.52×10^{-3}	1.54×10^{-3}
blended-wing body	7.11×10^{-5}	1.35×10^{-4}	5.21×10^{-4}	2.41×10^{-4}

A. ONERA M6 wing

In this example, we parametrize the ONERA M6 wing and transform it into a rectangular wing with NACA 0012 airfoil sections. The root chord of the M6 wing is normalized to 1.0, and the rectangular wing has a root chord of 0.49664. The shape transformation involves sweep, scaling, and section modifications.

The parametrization is obtained from the surface control points of a B-spline mesh fitted to an initial 12 block, HH-topology grid. Each block in the grid consists of $L_q \times L_r \times L_s = 45 \times 65 \times 33$ nodes, producing

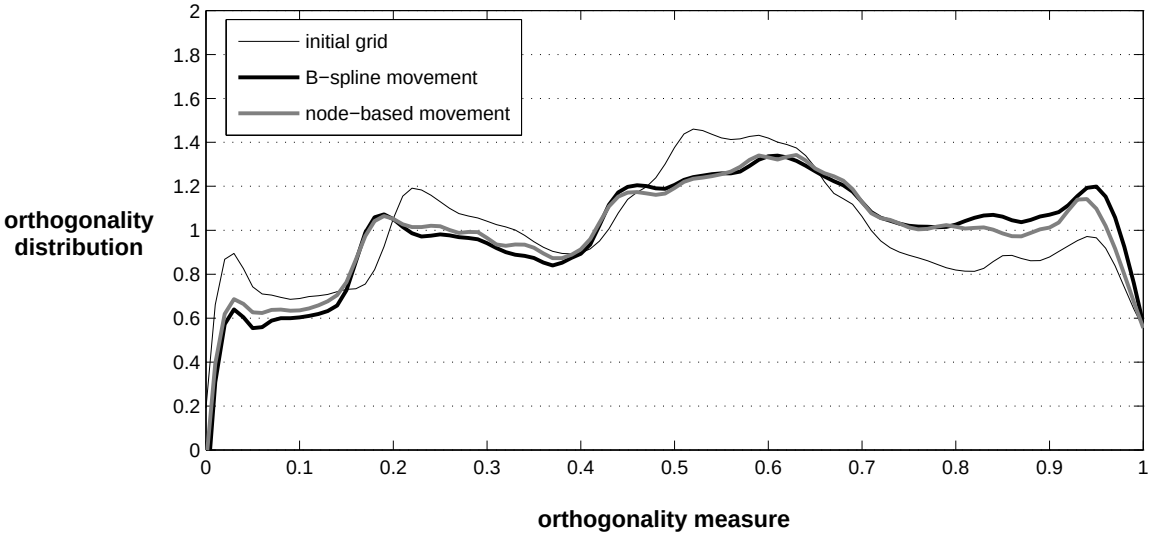


Figure 4. Orthogonality distribution of cells for the initial ONERA M6 grid and grids for the rectangular wing obtained using the B-spline mesh movement and node-based mesh movement.

approximately 1.158×10^6 nodes in total. The mesh spacing is typical for an inviscid flow analysis with the off-wall, leading edge, trailing edge, and tip spacing set at 0.001. The B-spline volumes for each block use $N_i \times N_j \times N_k = 13 \times 13 \times 9$ control points: the B-spline grid is approximately 60 times smaller than the computational grid. Figure 7, in Appendix A, provides visualizations of the initial and final B-spline control meshes and their corresponding computational grids.

The final computational grid is compared to the initial grid using the orthogonality measure (9). The measure is calculated for each element, and then grouped into bins which uniformly divide the range of possible values, namely $[0, 1]$. These bins are then used to produce a distribution of orthogonality. Integrating the distribution over the orthogonality range $[a, b]$ gives the ratio of elements that lie in this range. In particular, integrating the distribution over $[0, 1]$ gives 1.

Figure 4 plots the orthogonality distribution for the initial ONERA M6 grid and the grid for the rectangular wing. For comparison, the figure also includes the distribution for the grid obtained using the node-based-linear-elasticity approach. The final grids have distributions which are qualitatively similar to the initial distribution, which we would expect for a mesh movement based on linear elasticity. More notable are the similarities between the B-spline and node-based mesh-quality distributions. Indeed, in some cases the B-spline distribution is better; consider the first peak, near the low end of the quality range, which is smaller for B-spline mesh. This can be attributed to the smoothing properties of the B-spline volumes.

For this problem, the B-spline and node-based mesh movement required 227 seconds and 27.79 hours, respectively, on a single 1500 MHz Itanium 2 processor. For the B-spline mesh movement we found that a fill level of 1 was optimal in the ILU(p) preconditioner, while a fill level of 2 was better for the larger node-based problem. While better preconditioners may exist, the relative CPU time between the two approaches is likely to remain the same.

B. Blended-Wing Body

The initial shape is a flat plate with unit chord and a semi-span of 2. As in the previous example, the mesh consists of 12 blocks in an H-H topology. Each block has $45 \times 45 \times 45$ nodes. The off-wall spacing is 0.001, while the leading edge, trailing edge, and tip spacing are 0.005.

The initial mesh is fit using 12 B-spline volumes, with $9 \times 9 \times 9$ control points per volume; the control mesh reduces the number of degrees of freedom by a factor of 125 relative to the fine mesh. The fitted mesh for the flat plate is shown in Figure 8(b), together with its control mesh in Figure 8(a).

The control points on the surface of the plate are chosen as design variables. In this example, we set the

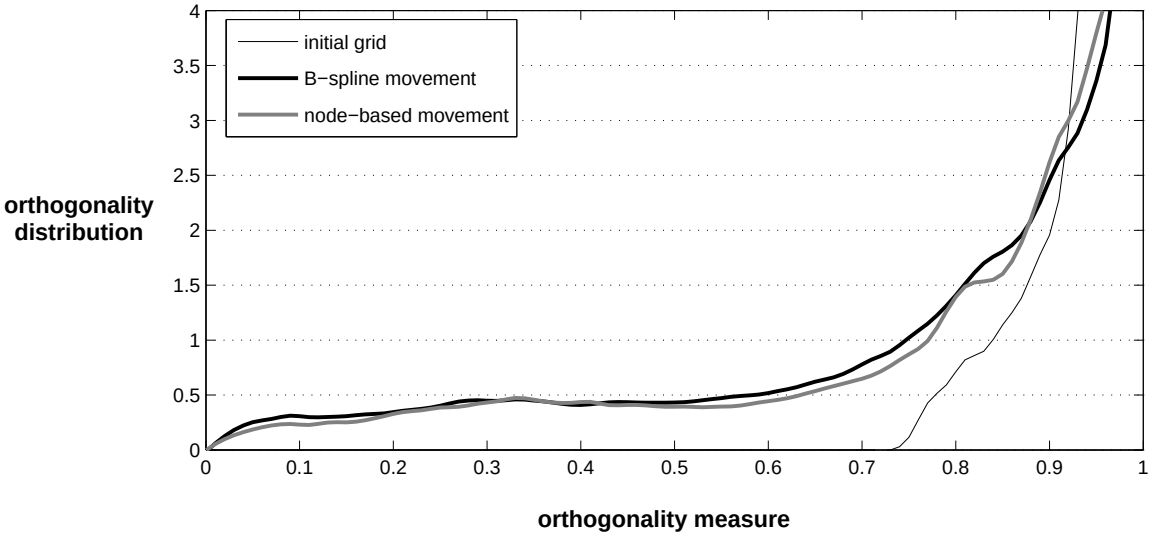


Figure 5. Orthogonality distribution of cells for blended-wing-body grids obtained using the B-spline mesh movement and node-based mesh movement. Note the use of a log-scale for the ordinates.

design variables to obtain a generic blended-wing geometry with winglets. The perturbed surface control points and control mesh are shown in Figure 8(c). The resulting mesh for the blended-wing-body shape is shown in figure 8(d).

The orthogonality measure distribution for the initial grid is plotted in Figure 5, together with the distributions for the grids obtained using the B-spline and node-based mesh movement algorithms. The initial grid is almost Cartesian and its distribution reflects this: all the elements have orthogonality measures greater than 0.74. In transforming from the flat plate to the blended-wing body, some loss of element orthogonality is unavoidable. As in the ONERA M6 example, the important observation is that the B-spline and node-based mesh movements produce very similar quality distributions, despite the significant difference in CPU time: the B-spline mesh movement required 128 seconds while the node-based mesh movement required 32.4 hours.

Finally, we solve for the flow around the blended-wing body to demonstrate the B-spline mesh movement in an analysis type application. Figure 6 shows the surface mesh and C_p distribution for the blended-wing-body grid obtained using the B-spline algorithm. The inviscid flow is defined by an angle of attack of 4 degrees and Mach number of 0.3. The flow solution was obtained using the algorithm described in reference 30.

V. Conclusions

A B-spline mesh can be used to integrate a CAD-free geometry parametrization with a semi-algebraic mesh movement algorithm. As a result, mesh movement can be made more efficient by reducing the number of degrees of freedom associated with the grid. Specifically, a coarse grid is perturbed using a robust mesh movement algorithm, and, subsequently, a fine mesh is obtained algebraically from the coarse one. We have demonstrated that B-spline mesh movement produces grids with quality comparable to those obtained using a node-based mesh movement while requiring two to three order of magnitude less CPU time.

B-spline volume mappings have many potential applications beyond parametrization and mesh movement. For example, they may be of use in mesh adaptation and refinement. The control points can be moved according to error estimates; this approach should be faster and more robust than moving individual points, given the properties of B-splines. When applied to convergence studies, these mappings provide an explicit transformation from parameter space to computational space, permitting a smooth refinement strategy. Using the same strategy, hierarchical grids can be constructed for multigrid. Finally, by adjusting the order of the B-spline basis functions, we can produce grids with sufficient smoothness for high-order discretizations.

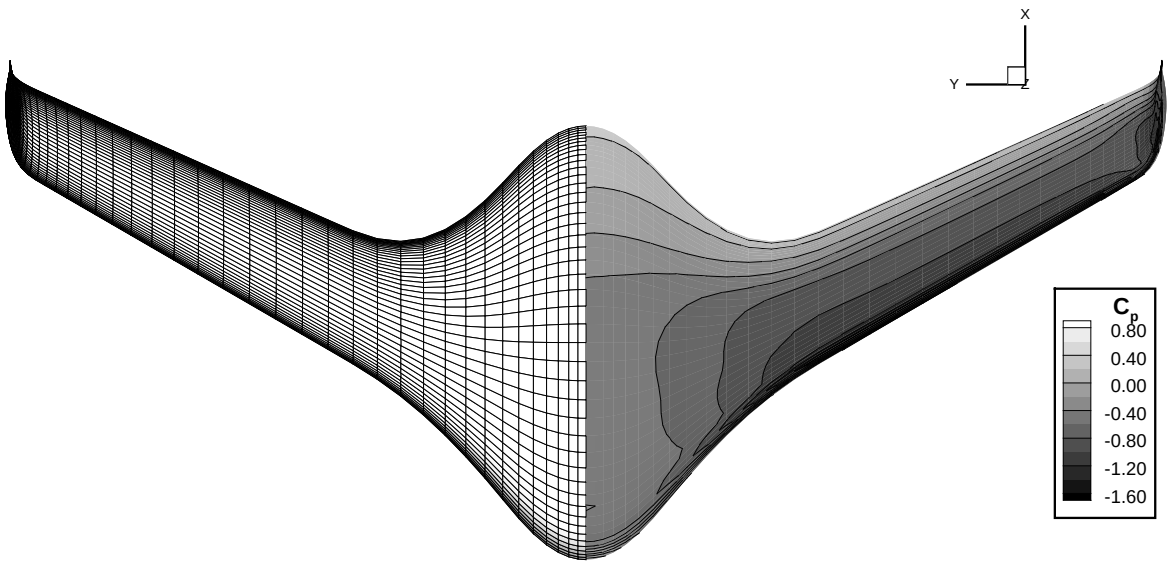


Figure 6. Euler flow solution on the blended-wing-body grid produced using the B-spline mesh movement algorithm.

Acknowledgments

The authors gratefully acknowledge financial assistance from the Natural Sciences and Engineering Research Council (NSERC), the Canada Research Chairs program, Mathematics of Information Technology and Complex systems (MITACS), and the University of Toronto.

References

[1]Liebeck, R., "Design of the blended wing body subsonic transport," *Journal of Aircraft*, Vol. 41, No. 1, 2004, pp. 10–25.

[2]Campbell, R. L., Carter, M. B., Odis C. Pendergraft, J., Friedman, D. M., and Serrano, L., "Design and testing of a blended wing body with boundary layer ingestion nacelles at high Reynolds numbers (invited)," *The 43rd AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–2005–0459, Reno, Nevada, 2005.

[3]Moigne, A. L. and Qin, N., "Aerofoil profile and sweep optimisation for a blended wing-body aircraft using a discrete adjoint method," *The Aeronautical Journal*, Sept. 2006, pp. 589–604.

[4]Jones, W. T., "A grid generation system for multi-disciplinary design optimization," *12th AIAA Computational Fluid Dynamics Conference*, No. AIAA–1995–1689, San Diego, California, United States, June 1995.

[5]Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D., "Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation," *The 34rd AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–1996–0094, Reno, Nevada, 1996.

[6]Liu, X., Qin, N., and Xia, H., "Fast dynamic grid deformation based on Delaunay graph mapping," *Journal of Computational Physics*, Vol. 211, 2006, pp. 405–423.

[7]Jakobsson, S. and Amoignon, O., "Mesh deformation using radial basis functions for gradient- based aerodynamic shape optimization," *Computers and Fluids*, Vol. 36, 2007, pp. 1119–1136.

[8]Allen, C. B. and Rendall, T. C. S., "Unified approach to CFD-CSC interpolation and mesh motion using radial basis functions," *25th AIAA Applied Aerodynamics Conference*, No. AIAA–2007–3804, Miami, Florida, United States, June 2007.

[9]Batina, J. T., "Unsteady Euler airfoil solutions using unstructured dynamic meshes," *AIAA Journal*, Vol. 28, No. 8, Aug. 1990, pp. 1381–1388.

[10]Farhat, C., Degand, C., Koobus, B., and Lesoinne, M., "Torsional springs for two-dimensional dynamic unstructured fluid meshes," *Computer Methods in Applied Mechanics and Engineering*, Vol. 163, 1998, pp. 231–245.

[11]Nielsen, E. J. and Anderson, W. K., "Recent improvements in aerodynamic design optimization on unstructured meshes," *AIAA Journal*, Vol. 40, No. 6, June 2002, pp. 1155–1163.

[12]Johnson, A. A. and Tezduyar, T. E., "Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces," *Computer Methods in Applied Mechanics and Engineering*, Vol. 119, 1994, pp. 73–94.

[13]Yang, Z. and Mavriplis, D. J., "Unstructured dynamic meshes with higher-order time integration schemes for the unsteady Navier-Stokes equations," *41st AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA–2005–1222, Reno, Nevada, 2005.

[14]Yu, T.-Y. and Soni, B. K., "Application of NURBS in numerical grid generation," *Computer-Aided Design*, Vol. 27, No. 2, Feb. 1995, pp. 147–157.

[15]Yu, T.-Y. and Soni, B. K., "NURBS in structured grid generation," *Handbook of grid generation*, edited by J. F. Thompson, B. K. Soni, and N. P. Weatherill, chap. 30, CRC Press Inc., 1999.

[16]Hoschek, J., "Intrinsic parametrization for approximation," *Computer Aided Geometric Design*, Vol. 5, No. 1, 1988, pp. 27–31.

[17]Rogers, D. F. and Adams, J. A., *Mathematical Elements for Computer Graphics*, McGraw–Hill, Inc., New York, NY, 1990.

[18]Hayes, J. G., *Numerical Analysis*, chap. Curved knot lines and surfaces with ruled segments, Springer Berlin/ Heidelberg, 1982, pp. 140–156.

[19]Schumaker, L. L., *Spline functions*, John Wiley & Sons, Inc., New York, NY, 1981.

[20]Walz, C., "A unified approach to B-spline recursions and knot insertion, with application to new recursion formulas," *Advances in Computational Mathematics*, Vol. 3, No. 1, 1995, pp. 89–100.

[21]Piegl, L. A. and Tiller, W., "Computing the derivative of NURBS with respect to a knot," *Computer Aided Geometric Design*, Vol. 15, 1998, pp. 925–934.

[22]Farin, G. E., *Curves and surfaces for computed aided geometric design: a practical guide*, Academic Press, Inc., London, UK, 4th ed., 1997.

[23]Hoschek, J. and Lasser, D., *Fundamentals of Computer Aided Geometric Design*, A. K. Peters Ltd., Wellesley, Massachusetts, 1993, translated by Larry L. Schumaker.

[24]Burgreen, G. W. and Baysal, O., "Three-dimensional aerodynamic shape optimization using discrete sensitivity analysis," *AIAA Journal*, Vol. 34, No. 9, Sept. 1996, pp. 1761–1770.

[25]Truong, A. H., Oldfield, C. A., and Zingg, D. W., "Mesh movement for a discrete-adjoint Newton-Krylov algorithm for aerodynamic optimization," *AIAA Journal*, Vol. 46, No. 7, July 2008, pp. 1695–1704.

[26]Meijerink, J. A. and van der Vorst, H. A., "An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix," *Mathematics of Computation*, Vol. 31, No. 137, Jan. 1977, pp. 148–162.

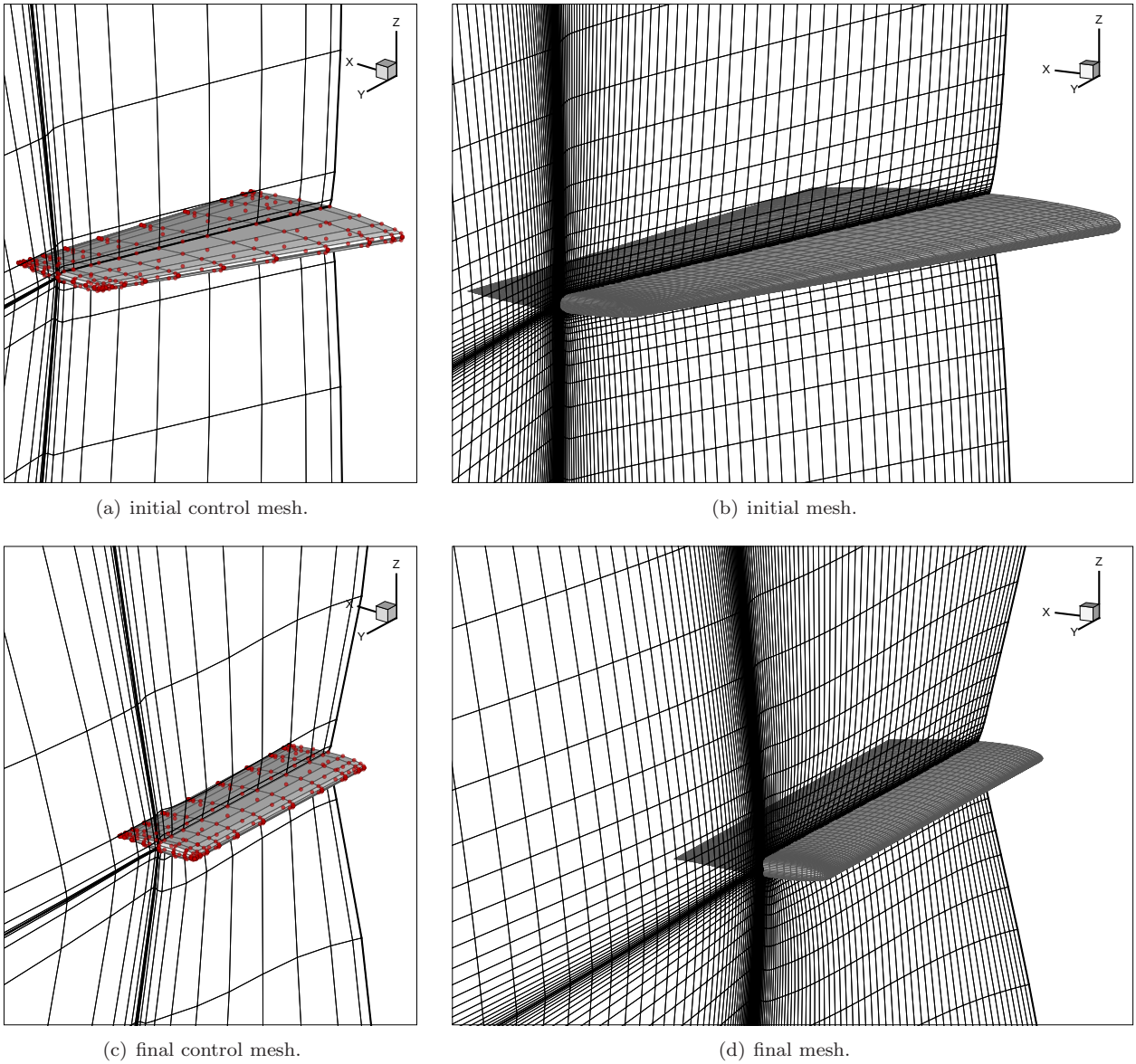
[27]Hicken, J. E. and Zingg, D. W., "An investigation of induced drag minimization using a parallel Newton-Krylov algorithm," *The 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, No. AIAA-2008-5807, Victoria, British Columbia, Canada, 2008.

[28]Bar-Yoseph, P. Z., Mereu, S., Chippada, S., and Kalro, V. J., "Automatic monitoring of element shape quality in 2-D and 3-D computational mesh dynamics," *Computational Mechanics*, Vol. 27, 2001, pp. 378–395.

[29]Schmitt, V. and Charpin, F., "Pressure distributions on the ONERA-M6-wing at transonic mach numbers," Tech. rep., Office National d'Etudes et Recherches Aerospatiales, 92320, Chatillon, France, 1979.

[30]Hicken, J. E. and Zingg, D. W., "A parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms," *AIAA Journal*, accepted July 2008.

A. Visualization of control and computational grids



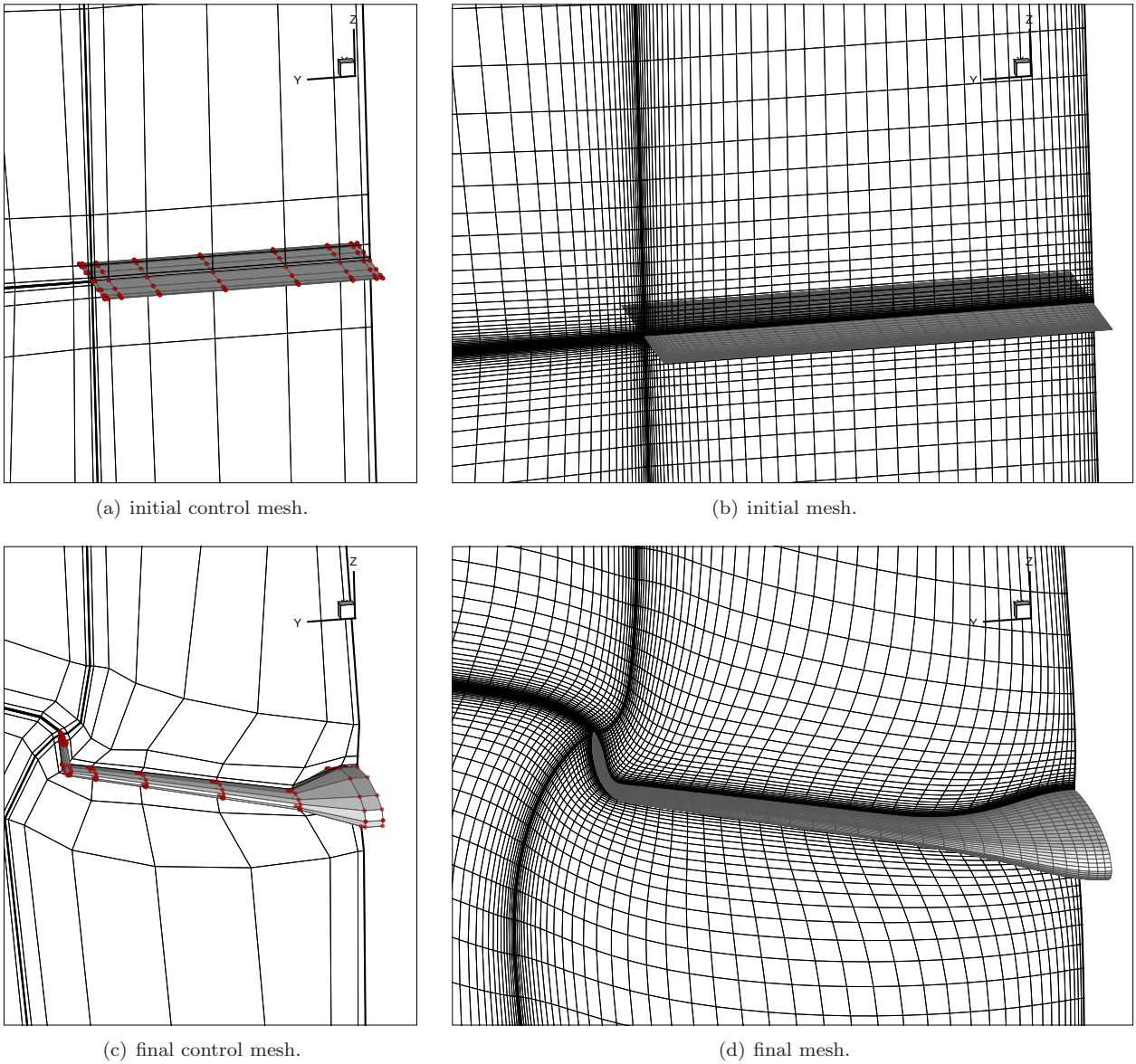
(a) initial control mesh.

(b) initial mesh.

(c) final control mesh.

(d) final mesh.

Figure 7. Control mesh (left) for the initial shape (top) and final shape (bottom). The surface control points are marked with red spheres. The figures on the right show the grid corresponding to the control mesh on the left.



(a) initial control mesh.

(b) initial mesh.

(c) final control mesh.

(d) final mesh.

Figure 8. Control mesh (left) for the initial shape (top) and final blended-wing-body shape (bottom). The surface control points are marked with red spheres. The figures on the right show the grid corresponding to the control mesh on the left.