Surface Mesh Movement Algorithm for CAD-Based Aerodynamic Shape Optimization

Anh H. Truong*, David W. Zingg†

Institute for Aerospace Studies, University of Toronto

4925 Dufferin St., Toronto, Ontario, Canada, M3H 5T6,

Robert Haimes‡

Massachusetts Institute of Technology, Cambridge, MA, USA

This paper focuses on the development of a surface mesh movement algorithm suitable for Computer-Aided-Design (CAD)-based aerodynamic shape optimization. The algorithm interrogates the CAD system via the vendor-neutral API Computational Analysis PRogramming Interface (CAPRI) and uses CAPRI's watertight triangulation of a modified CAD geometry to guide the movement of the structured surface mesh as the geometry changes during the optimization process. A mapping procedure is introduced which not only preserves the characteristics of the original surface mesh, but also guarantees that the new mesh points are on the CAD geometry. The deformed surface mesh is then smoothed in the parametric space before it is transformed back into three-dimensional space. The procedure is efficient in that all the processing is done in the parametric space, incurring minimal computational cost. The mesh movement tool is integrated into a three-dimensional shape optimization framework, with a linear-elasticity volume-mesh movement algorithm, a Newton-Krylov flow solver for the Euler equations, and a discrete-adjoint gradient-based optimizer. The accuracy of the computed gradients is verified through a number of examples.

*Graduate Student

†Professor, J.A. Bombardier Foundation Chair in Aerospace Flight, Senior Canada Research Chair in Computational Aerodynamics and Environmentally Friendly Aircraft Designs, Associate Fellow AIAA.

‡Principal Research Engineer, Member AIAA.

Nomenclature

Alphanumeric

a	Translation of a coordinate system in the x direction
A, B, C	Coefficients in the general equation of a line
b	Translation of a coordinate system in the y direction
C_D	Drag coefficient
C_L	Lift coefficient
$\mathcal{X}$	Design variables
f	Winslow smoothing function for either x or y coordinate direction
G	Grid variables
G_{surf}	Surface mesh
$\mathcal{J}$	Objective function
$\mathcal{L}$	Lagrangian
M	Mach number
Q	Flow variables
$\mathcal{C}$	Constraints
$\mathcal{R}$	Flow residual
S	Wetted planform area of the wing
s	Scale change between coordinate systems
X, Y	Transformed coordinates in physical space

Greek

ϵ	Finite-difference step size
ε	Nonorthorgonality angle between the axes of the two coordinate systems
ξ, η	Coordinates in computational space
Ω	Feasible region of the design space
ψ, λ	Lagrangian multipliers for the flow and grid variables
φ	Rotation of the axes of one coordinate system with respect to the other
ω	Angle of attack
τ	Linear twist
θ	Rotation angle

Abbreviations

Adj	Adjoint
CAD	Computer-Aided Design
CFD	Computational Fluid Dynamic
EGADS	Engineering Geometry Aerospace Design System

FD	Finite-Difference
FFD	Free-Form Deformation
Ganimede	Geometry ANd Inherent MEsh DEformation
NIST	National Institute for Standards and Technology
NURBS	Non-Uniform Rational B-Splines
OpenCASCADE	Open-source Computer Aided Software for Computer Aided Design and Engineering
OpenCSM	Open-source Constructive Solid Modeler
MDO	Multidisciplinary Design Optimization
SNOPT	Sparse Nonlinear OPTimizer

Subscripts

0	Origin
i, j	Node indices

I. Introduction

CAD systems have robust modeling capability and are able to capture design intent. They allow designers to create and modify shapes effortlessly in 3D simply by altering the dimensions of the features from which they were created. The designer is able to better understand how the product will look and function before making physical prototypes and therefore able to avoid many design errors. Such functionality and versatility provided by the CAD system makes it an essential tool for product design and development. Ideally, the CAD representation of the geometry should be used directly in numerical simulations to maintain the accuracy and integrity of the model. However, there are several issues that prevent CAD from being widely used as a geometry parameterization tool in aerodynamic shape optimization algorithms. First, CAD entities are usually inaccessible and thus non-modifiable by a third-party application. Second, users have no control over the mathematical representation or topological outcome of the model. This lack of control makes it difficult to relate the surface mesh of the computational domain to the CAD face as it evolves during the optimization process. Third, the complexity and unavailability of the source code makes it difficult to obtain analytical sensitivity information required by gradient-based optimizers. Consequently, CAD models have traditionally been translated to commonly used formats such as IGES[1] or STEP.[2,3] The translated models often contain gaps and overlaps and may need extensive repair. This laborious, manual-intervention process typically consumes up to 80% of the total time required for Computational Fluid Dynamic (CFD) analyses[4] and is a major bottleneck in the use of CFD as a practical design tool.

In 1991, the AIAA Technical Committee on Multidisciplinary Design Optimization (MDO)

identified that one of the key areas needing development is to establish unified numerical modeling parmeterized in terms of the design variables – a consistent vehicle geometry to be used as a basis for all mathematical models, such that changes to the geometry are centrally coordinated.[5] To address this problem, most CAD systems began introducing toolkits such as Autocad ObjectARX, Parasolids, the SolidWorks API, CATIA CAA, and Pro/Toolkit[6,7] for accessing geometric data, database structures, graphics systems, and native command definition in the CAD system. This marks a major milestone, as CAD entities are now accessible and modifiable and thus directly usable in shape optimization algorithms.

In particular, a CAD parameterization is ideal for use in multidisciplinary design optimization (MDO) where the geometry representation has to be consistent across different engineering disciplines because it permits simultaneous changes in the outer skin and the internal structural components, i.e. a modification in the outer mold shape will cause a corresponding change in the underlying structure that is connected to it or vice versa. However, its use has been very limited thus far for two important reasons. First, CAD software is very complex and users have no control on the mathematical definition or the number of entities generated by a CAD program. This means that the designer cannot relate the geometry definition to the body-fitted computational mesh, particularly the surface mesh, and is limited to optimization problems in which the geometry can undergo only relatively small movements.[8,9,10] In these cases, the geometry definition does not change and the movement of the surface mesh points can be "tracked" by their CAD parametric coordinates,[8,10] or regenerated entirely based on the initial topology.[9] Alternatively, Nemec *et al.*[11] used Cartesian meshes so that mesh movement or regeneration can be avoided, but they are limited to optimization problems where the flow is inviscid. Second, CAD systems do not provide derivatives of surface displacements with respect to the design variables, which are needed in the chain rule to compute the sensitivities of the objective/constraint function with respect to design parameters. The surface sensitivity derivatives can be obtained by differentiating the CAD modeler if the source code is available, such as in the case of the open-source CAD engine OpenCASCADE.[12] However, differentiating the source code is not an easy task due to its size and complexity.[13] At present, OpenCASCADE contains over $14,000$ classes, making differentiation a difficult undertaking. Thus far, only Kleinveld *et al.*[14] were successful in differentiating their in-house CAD modeler, Ganimede (Geometry ANd Inherent MEsh DEformation), to obtain analytical sensitivity derivatives.

Yu *et al.*[15] proposed an alternate representation of CAD models using NURBS (Non-uniform rational B-splines), a form which CAD systems use to export their geometries. The derivatives of the surface with respect to the design variables at any given location on the surface are obtained by applying automatic differentiation (AD) in reverse mode to a generic NURBS implementation. Xu *et al.*[16] generalized this approach to 3D geometries consisting

of multiple NURBS patches. They introduced constraints for geometric continuity across NURBS patch interfaces to maintain a desired level of continuity of tangency and curvature between adjacent NURBS patches when a control point on or near a patch interface is displaced. Jones *et al.*[13] infer the parameter sensitivity by tracing the evolution of the hierarchical associativity of CAD features. Robinson *et al.*[17] use the design velocity field approach to evaluate the surface displacement over a faceted coarse surface mesh and then interpolate on the boundary of the fine computational mesh. The use of design velocity for optimization is well established. It can be computed for meshes of the boundary,[18,19] or directly from the geometric CAD model.[20,21] The design velocity is a measure of the normal displacement of the model boundary caused by a modification of the design parameter. The change in objective function caused by the perturbation of the design parameter can be predicted using the boundary method expressed in terms of design sensitivity and is described by Choi *et al.*[22,23] This method assumes that the change in performance is of first order, which is valid for small boundary movements that are continuous over the boundary.[17] If the relationship between the design variables and nodal coordinates is linear, then the velocity field needs to be calculated once, whereas if the relation is nonlinear, then the velocity field must be updated at each design iteration.[24]

More recently, Haimes and Dannenhoffer[25] created a browser-based geometry construction and manipulation tool called Engineering Sketch Pad, which, in many ways, mirrors the functionality of modern parametric commercial CAD systems, as it is built upon OpenCSM (which is the open-source constructive solid modeler that is in turn built upon EGADS - the Engineering Geometry Aerospace Design System, and OpenCASCADE) that is able to provide analytic parameter sensitivity.

Often, the effect of each design parameter is determined using the finite-difference approach.[8,11,26,27,28,9] Using this approach, the number of required geometry and surface mesh deformations scales proportionally to the number of design variables. Furthermore, in a Sequential Quadratic Programming optimizer such as SNOPT, during a line search, the geometry is generally perturbed a few more times to construct a quadratic fit along the direction of the gradient. The computational cost can become excessive for problems with large numbers of design variables. Moreover, choosing an optimal step size to avoid truncation or round-off errors can be challenging as it varies with each design variable and with each design cycle.[29] The finite-difference approach also requires the topology of the model to remain constant, which can be hard to achieve in some cases. Despite these issues, this approach is general and relatively straightforward to implement.

In this paper, we develop a robust surface mesh movement method that deforms the initial structured surface mesh to fit the new CAD geometry for aerodynamic shape optimization. As in our previous work,[30] CATIA v5 is used to generate the CAD geometry

and communication with the CAD software is done using a vendor-neutral Computational Analysis PRogramming Interface called CAPRI.[31] Rather than relying on the mathetical representation of the CAD face, the mesh movement algorithm uses the discrete form provided by CAPRI to drive the algorithm and guarantee that the surface mesh lies on the CAD face. The objectives of the algorithm are to maintain the mesh quality and characteristics of the initial surface mesh as it adapts to the new shape during the optimization process, to maintain geometric fidelity by ensuring that the mesh is on the CAD face, and to provide accurate surface sensitivity derivatives of the mesh node with respect to the geometric design variables. The background and methodology regarding surface parameterization used by the surface mesh movement algorithm are described in the following sections. A finite-difference method for the computation of the surface sensitivity[11] is implemented. The surface mesh movement algorithm is integrated into a three-dimensional shape optimization framework, with a linear-elasticity volume-mesh movement algorithm,[32] a Newton-Krylov flow solver for the Euler equations,[33] and a gradient-based optimizer, SNOPT.[34] The validity and accuracy of the CAD-based optimization algorithm are demonstrated through a number of verification and optimization cases.

II. Surface Mesh Movement Algorithm

The deformation process involves the following steps:

1. Cluster the CAD faces and the corresponding CAPRI tessellation into sizable patches, compared to the size of the mesh, to maintain consistent surface topology after each geometry regeneration.

2. Parameterize the surface mesh and the CAPRI tessellation for each of the patches onto the same region in the 2D plane, aligning corresponding boundaries.

3. Search for a triangle in the CAPRI tessellation that contains each of the surface nodes. The position of the node inside the triangle, i.e. its barycentric coordinates, is calculated.

4. Determine the CAD parametric coordinates for the surface node based on the barycentric coordinates and the CAD parametric coordinates of the vertices of the triangle which enclose it. This is an important step because it guarantees that the new mesh nodes will be on the CAD face.

5. Convert the CAD parametric coordinates of the surface nodes into physical coordinates in 3D by interrogating the CAD program.

In this paper, a "CAD" geometric entity is always explicitly stated so to avoid confustion with mesh related entities. For example a "CAD surface" is a geometric entity, whereas as "surface" alone is a mesh entity.

II.A. Clustering Algorithm

A clustering algorithm similar to the method of Sheffer *et al.*[35,36,37] is implemented here to ensure consistent topology as the geometry changes during shape optimization. However, the criterion for the clustering of faces is based on the blocking scheme of the initial structured, multi-block volume mesh. Since the mesh will generally be deformed and reused throughout the optimization cycle, the sides of the blocks which lie on the CAD surface are used to guide the clustering algorithm such that the boundary of the block sides coincides with the boundary of the clustered CAD faces. For a given initial geometry and volume mesh, the designer chooses a set of rules that will partition the CAD faces into the least number of patches. The boundaries of these patches have to coincide with the block edges. The patches themselves have to be sufficiently large (compared to the size of the associated surface mesh element) and smooth so that when the surface mesh on these clustered faces is re-parameterized, it will be least distorted. In addition, the rule has to be valid throughout the optimization process in order to correctly relate the position of the mesh node to the CAD surface.

II.B. Survey of Parameterization Methods

The parametrization problem is inherently difficult, as one is trying to flatten a surface from 3D to 2D in such a way as to minimize distortion.[38] Generally, parametrization techniques can be classified into two types: fixed or non-fixed boundaries in the parametric domain. Fixed boundary methods typically use very simple formulations and are very fast.[39] However, fixed boundary methods often have large distortions because the boundary shapes of the original models can be very different from those of their flattened surfaces in the 2D domain.[38] Free-boundary techniques, which determine the boundary as part of the solution, are often slower, but typically introduce significantly less distortion. In recent years, numerous methods for parameterizing meshes have been developed, targeting diverse parameter domain and focusing on minimizing the distortion of different intrinsic measures of the original mesh.

In general, we want to preserve as much of the intrinsic qualities of a surface as we can during its parametrization. The intrinsic qualities for a discrete surface are length, angle and area. A length-preserving or isometric mapping is ideal in that it preserves not only length, but also angle and area. However, it is well known that isometric mappings only

exist in very special cases, where the surface is developable.[a] Therefore, many approaches to surface parametrization attempt to find a mapping which minimizes distortion of either angle or area, or some combination of these.[40]

Maps that minimize the angular distortion are called *conformal*, and maps that minimize area distortion are called *authalic*. In theory, angular distortion can be eliminated completely by conformal mapping, but it is impossible for conformal mappings to further eliminate area distortion completely, except for developable surfaces. As pointed out by Floater and Hormann,[40] although authalic parameterizations are achievable, they are not very useful by themselves, as they allow extreme angular and linear distortion. Thus, area preservation[41,42] methods are typically combined with angle preservation[39] to minimize the distortion.

A common approach is to minimize a certain energy to control the distortion. A simple, yet effective fixed-boundary parametrization method that uses linear spring energy with a uniform spring constant was introduced by Tutte.[43] Tutte's embedding method is established in the following steps: First, n vertices which make up a boundary segment of the surface mesh are positioned on some convex polygon in R^2. The positions of the interior vertices are calculated so that the total energy is minimized. This energy can be represented as a sum of the energy of a configuration of springs with one spring placed along each edge of the surface mesh. Tutte's method results in a sparse, diagonally-dominant linear system that can be solved easily with any iterative method.[44] Floater[45] proposed a different set of weights for the edge spring model. He generalized Tutte's procedure to generate all possible valid embeddings of the 3D graph in the plane, given the (convex) positions of the boundary. Floater's shape-preserving parameterization method has the advantage that the weights are always positive. This guarantees that the linear system derived can be solved robustly.[44] Others[46,47,48] have developed variations of Tutte's method, aiming for some effect related to reflecting the geometry of the mesh in the parameterization, i.e. minimizing its metric distortion. However, some of the methods,[46,47] cannot guarantee a valid map. Variations of harmonic energies were also optimized using discrete Laplace-Beltrami operators in.[49,47,45,50,51,48] However, harmonic maps may contain face flips which violate the bijectivity [b] of a parameterization.[42]

II.C. Surface Mesh Parameterization Considerations

In choosing an appropriate surface mesh parameterization method, the following factors are taken into consideration: free vs fixed boundary, robustness and numerical complexity. Since we need to find a correspondence between the source and target surface meshes, the

[a] surfaces with zero Gaussian curvature, which means that it is a "surface" that can be flattened onto a plane without distortion (i.e. "stretching" or "compressing"). Examples are cylinders and cones.

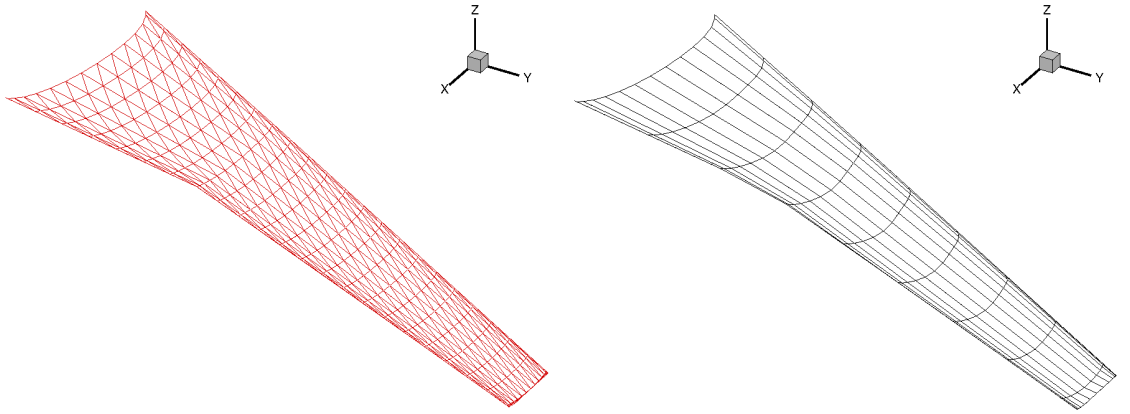
[b] there is a one-to-one correspondence between the 3D surface and its parameterization

boundaries of the surfaces in parametric space must be the same. This limits our choice of methods to those with fixed boundaries. For simplicity, a unit square is used as the boundary for the mapped surfaces. For our application, it is sufficient (or even desirable) to take a square as the parametric domain, with the advantage that such a convex shape guarantees the bijectivity of the parameterization if positive barycentric coordinates like the mean value coordinates are used to compute the parameter points for the interior vertices. The four edges on the surface patch can be related to the four sides of the square. For simplicity, the parameter points are uniformly spaced on the rectangular domain. Although some heuristic procedures for placing the boundary points have been proposed[52] and tried, having to fix the boundary vertices is already a limitation.

Most applications of a parameterization require it to be bijective (invertible). The bijectivity can be local, which means that there are no triangle flips, or global, which means that the boundary does not self-intersect. Since we use the map as a means to locate the position of any point on the 3D surface, any technique which provides a one-to-one mapping is sufficient. Finally, parameterization methods which require a linear solution method (as opposed to nonlinear method) are typically significantly faster and simpler to implement, at a cost of increased distortion. Since our problems contain hundreds of design variables and we need to find the sensitivity of the surface with respect to each of the design variables, a method that can provide a fast solution is required.

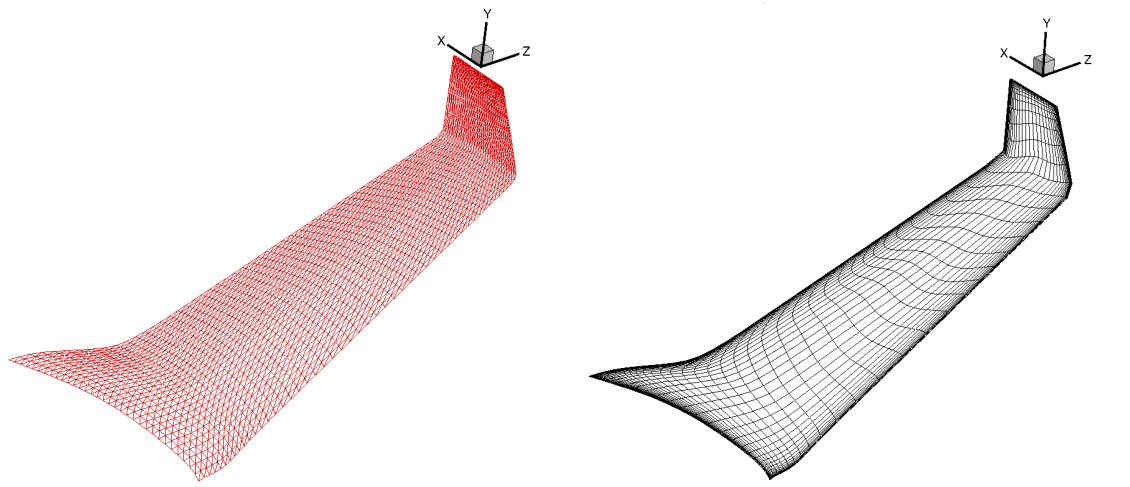
Tutte's method satisfies these criteria. However, since Tutte's method does not preserve any intrinsic property of the mesh, we need a triangulation that has elements of uniform length. Any non-uniformity or irregularity in the mesh will introduce distortion in the mapping, and subsequently, the final mesh. The quality of the parameterization is directly affected by the regularity of the triangulation, both topologically and geometrically. Topological regularity refers to meshes where the vertices have the same degree (connectivity), and geometric regularity implies that the triangles are similar to each other in terms of shape and size, and have vertices close to the centroid of their neighbors.[39] For example, the regularity of the triangulations shown in Figures 1(a) and 1(c) results in the smooth surface meshes shown in Figures 1(b) and 1(d) upon inverse map. This regularity cannot be easily obtained using CAPRI, only in cases where the boundary of a CAD face is similar to that of a square, such as the straight portion of the wing shown in Figure 2. This regularity is lost when the wing is given some angular deformation, such as a sweep of the leading edge, resulting in distortion of the surface mesh in the region where the irregularities occurred. This is evident in Figure 3. Section II.D describes how the CAPRI triangulation is parameterized on a unit square using Tutte's method.

The initial structured surface mesh is parameterized on the uv-parametric space based on the idea of scaling, to maintain the ratio of the mesh cell with respect to the given patch.



(a) Surface triangulation of the deformed geometry

(b) New surface mesh



(c) Triangulation of the target surface

(d) New surface mesh

Figure 1: The quality of the deformed surface mesh is dependant on the uniformity of the triangulation.

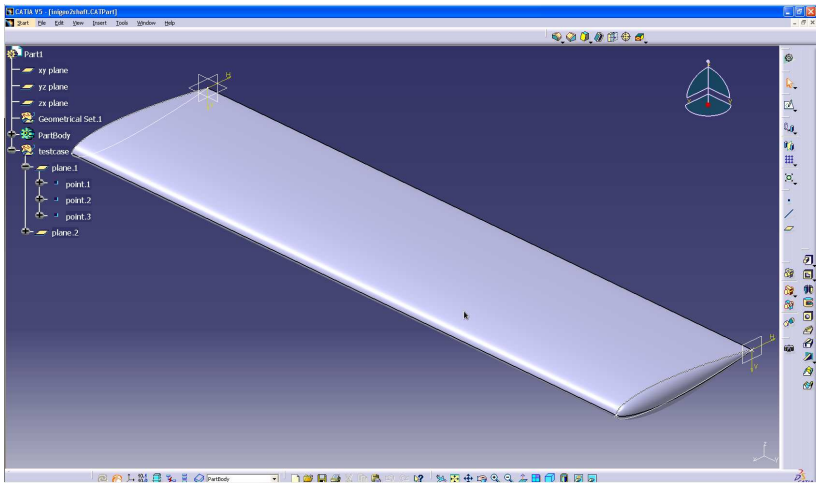
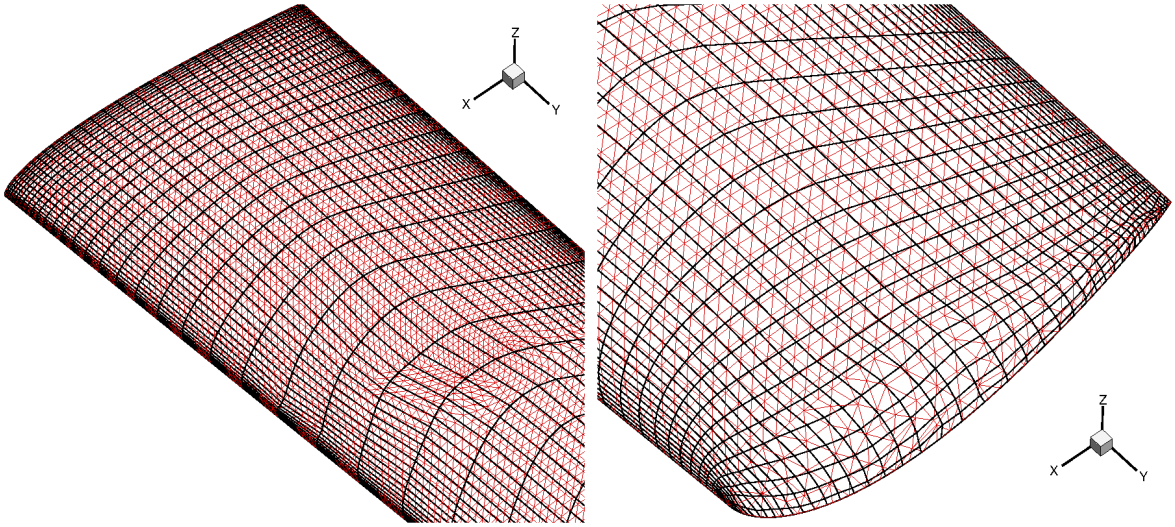


Figure 2: CATIA part



(a) Wing root (b) Wing tip

Figure 3: Zoomed-in view of the root and tip region on the top surface of the wing with the surface mesh (black) overlaying the CAPRI triangulation (red). The surface mesh is slightly distorted where the triangulation is irregular.

To preserve the mesh spacing of the initial surface mesh, it is parameterized onto a unit square using the arclength method, where u_j is the normalized arc length given by:

$$u_1 = 0 \tag{1}$$

$$u_{j_{max}} = 1 \tag{2}$$

$$u_j = \frac{1}{L_g} \sum_{i=2}^{j} L_i \qquad j = 2, \ldots, j_{max-1} \tag{3}$$

L_i is the length of a segment between nodes j and $j-1$, and L_g is the grid line length from one side of the patch to the other:

$$L_g = \sum_{i=2}^{j_{max}} L_i \tag{4}$$

A similar calculation is performed for the other parametric coordinate, v_j.

II.D. Barycentric Mapping

A simple idea for constructing a parameterization of a triangular mesh is based on the spring model, where the edges of the triangular mesh are springs that are connected at the vertices. If the boundary of this spring network is fixed somewhere in the plane, then the interior of this network will relax in the energetically most efficient configuration, and we can simply assign the positions of the vertices as parameter points.[52]

Following the derivation given by Hormann,[52] each spring is assumed to be ideal in the sense that the rest length is zero and the potential energy is $\frac{1}{2}Ds^2$, where D is the spring constant and s the length of the spring. For a mesh with n interior points and b boundary points, the parameter points $\mathbf{u}_i = (u_i, v_i)$, $i = n+1, \ldots, n+b$ for the boundary vertices $\mathbf{p}_i \in \mathcal{V}_\mathcal{B}$ of the mesh are projected onto a planar convex polygon. Then minimize the overall spring energy:

$$E = \frac{1}{2} \sum_{i=1}^{n} \sum_{j \in N_i} \frac{1}{2} D_{ij} \|\mathbf{u}_i - \mathbf{u}_j\|^2 \tag{5}$$

where $D_{ij} = D_{ji}$ is the spring constant of the spring between $\mathbf{p}_i$ and $\mathbf{p}_j$, with respect to the unknown parameter positions $\mathbf{u}_i = (u_i, v_i)$ for the interior points. As the partial derivative of E with respect to $\mathbf{u}_i$ is

$$\frac{\partial E}{\partial \mathbf{u}_i} = \sum_{j \in N_i} D_{ij}(\mathbf{u}_i - \mathbf{u}_j) \tag{6}$$

the minimum of E is obtained if

$$\sum_{j \in N_i} D_{ij}\mathbf{u}_i = \sum_{j \in N_i} D_{ij}\mathbf{u}_j \tag{7}$$

for all $i = 1, \ldots, n$, where n is the number of interior nodes and N_i is the number of vertex p_i's neighbors. In other words, each interior parameter point $\mathbf{u}_i$ is an *affine combination* of its neighbors,

$$\mathbf{u}_i = \sum_{j \in N_i} \kappa_{ij} \mathbf{u}_j \tag{8}$$

with normalized coefficients

$$\kappa_{ij} = \frac{D_{ij}}{\sum_{k \in N_i} D_{ik}} \tag{9}$$

that sum to 1.

By separating the parameter points for the interior and the boundary vertices in the sum on the right hand side of Eq. 8 we get

$$\mathbf{u}_i - \sum_{j \in N_i, j \leq n} \kappa_{ij} \mathbf{u}_j = \sum_{j \in N_i, j > n} \kappa_{ij} \mathbf{u}_j \tag{10}$$

and see that computing the coordinates u_i and v_i of the interior parameter points $\mathbf{u}_i$ requires the solution of the linear systems

$$AU = \bar{U} \qquad \text{and} \qquad AV = \bar{V} \tag{11}$$

where $U = (u_1, \ldots, u_n)$ and $V = (v_1, \ldots, v_n)$ are the column vectors of unknown coordinates, $\bar{U} = (\bar{u}_1, \ldots, \bar{u}_n)$ and $\bar{V} = (\bar{v}_1, \ldots, \bar{v}_n)$ are the column vectors with coefficients

$$\bar{u}_i = \sum_{j \in N_i, j > n} \kappa_{ij} u_j \qquad \text{and} \qquad \bar{v}_i = \sum_{j \in N_i, j > n} \kappa_{ij} v_j \tag{12}$$

and $A = (a_{ij})_{i,j=1,\ldots,n}$ is the $n \times n$ matrix with elements

$$a_{ij} = \begin{cases} 1 & \text{if} \quad i = j, \\ -\kappa_{ij} & \text{if} \quad j \in N_i, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

The simplest value for the spring constant $D_{ij} = 1$ goes back to the work of Tutte,[43] who used it in a more abstract graph-theoretic setting to compute straight line embeddings of planar graphs. This approach lacks the linear reproduction characteristic, which is a production of an isometric (and thus optimal) mapping in the case where the surface is contained in a plane so that its vertices have coordinates $p_i = (x_i, y_i, 0)$ with respect to some appropriately chosen orthonormal coordinate system. In this case, the parameter points themselves are defined using the local coordinates, i.e. $\mathbf{u}_i = \mathbf{x}_i$, for $i = 1, \ldots, n + b$.

Other researchers[47,49,48,53,54] have developed various values for D_{ij} to achieve the linear reproduction property, which is useful for computer graphics, geometric modeling and other applications,[52] but can affect the solvability of the linear systems defined in Eq. 11. Since the linear reproduction characteristic is inconsequential to our application, the method of Tutte is chosen for the reparameterization of the target surface (i.e. the CAPRI triangulation).

II.E. Inverse Mapping

When a vertex p is included in a face of an input mesh $f = (p_i, p_j, p_k)$ in the parametric domain, the 3D position p is calculated by a barycentric coordinate (α, β, γ):

$$p = \alpha p_i + \beta p_j + \gamma p_k \tag{14}$$

Before the calculation of p, we need to find a face f in which p is included. In general, this problem can be regarded as a point location problem in the parametric domain, and can be processed in $\mathcal{O}(\log m)$-time for each vertex, where m is the number of faces in the CAPRI triangulation. Then, the calculation for the uniform subdivision fitting at each level is processed in $\mathcal{O}(n \log m)$-time, where n is the number of vertices in the 3D surface patch. In order to ensure that the surface node is on the CAD face, we obtain the coordinates of the node in 2D using the CAD parametric coordinates of the triangle vertices, then a CAPRI function gi_qNormalToFace[55] is called to obtain the corresponding physical coordinates in 3D space. For nodes lying on boundary edges, their parametric positions are obtained by interpolating the edge vertices of the triangles which bound the nodes. This is possible because the vertices in the CAPRI tessellation are guaranteed to be on the CAD face (for interior nodes) and on the CAD edge (for boundary nodes). Since the gi_qNormalToFace query is needed for all surface nodes, it can be executed in a "grouped" mode where all calls are collected and executed in a single command to minimize communication time.

II.F. Barycentric coordinates in 2D

Consider a 2D triangle whose vertices are $a = (x_a, y_a)$, $b = (x_b, y_b)$, $c = (x_c, y_c)$ and a point $p = (x, y)$. Barycentric coordinates allow us to express the coordinates of p in terms of a, b, c. More specifically, the barycentric coordinates of p are the numbers α, β and γ such that

$$p = \alpha + \beta(b - a) + \gamma(c - a) \tag{15}$$

If we regroup a, b and c, we obtain

$$\begin{aligned} p &= \alpha + \beta b - \beta a + \gamma c - \gamma a \\ p &= (1 - \beta - \gamma)a + \beta b + \gamma c \end{aligned} \tag{16}$$

If we define α as

$$\alpha = 1 - \beta - \gamma \tag{17}$$

we then have

$$p = \alpha a + \beta b + \gamma c \tag{18}$$

The values of α, β, γ can be determined by writing Eq. 15 in terms of the coordinates of the various points involved to yield the following system of equations:

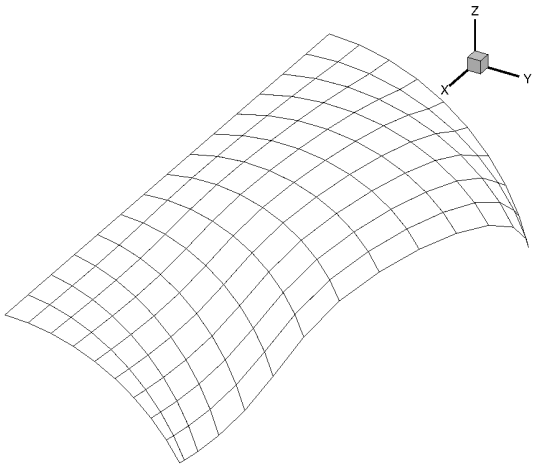
$$\begin{cases} x &= x_a + \beta(x_b - x_a) + \gamma(x_c - x_a) \\ y &= y_a + \beta(y_b - y_a) + \gamma(y_c - y_a) \end{cases} \tag{19}$$

Once the values for α, β and γ are obtained, they are substituted in Eq. 18 to determine the position of point p in the 3D space.

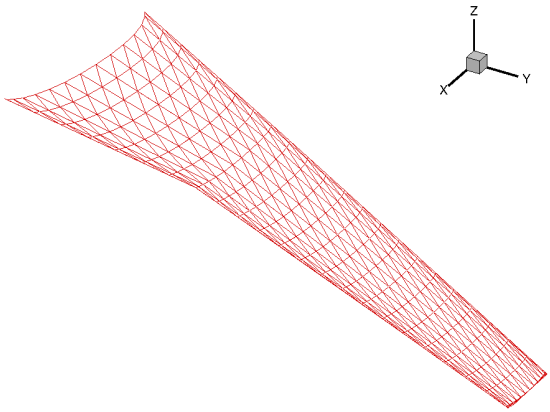
II.G. Surface Mesh Deformation Examples

Figure 4 illustrates the surface mesh deformation process. Given an initial structured surface mesh, generated for example using a mesh generation method on the initial geometry, as shown in Figure 4(a), and a tessellation of the deformed surface, Figure 4(b), the surface mesh movement algorithm parameterizes them onto a unit square, shown in Figures 4(c) and 4(d), respectively. By overlapping the parameterizations, shown in Figure 4(e), it searches for a triangle in the tessellation that encloses each of the surface mesh points and determines the barycentric coordinates of each of the points with respect to the coordinates of the vertices of triangle. The coordinates of the new surface mesh, Figure 4(f), are obtained by interpolating the 3D physical coordinates of the vertices of the triangles using the corresponding barycentric coordinates. It can be seen in Figure 4(f) that the new surface mesh has similar features (node clustering and distribution) to the tessellation. This example demonstrates the importance of having a uniform triangulation, as any pattern or distortion introduced by the tessellation will be transferred to the resulting surface mesh.

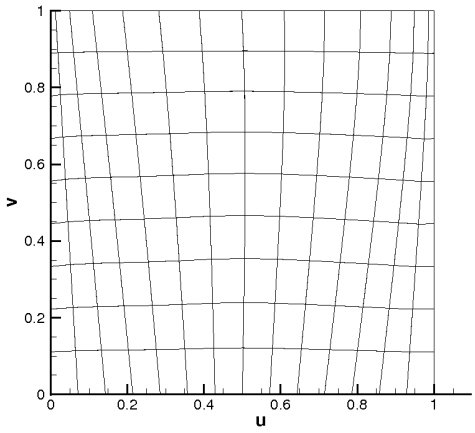
The second example, shown in Figure 5, involves the deformation of a dense surface mesh that is designed for viscous flow simulations. The tessellation, Figure 5(b), is uniform, and the characteristics of the original surface mesh, Figure 5(a), such as the clustering around the edges and the spacing along the spanwise direction, are preserved in the deformed surface



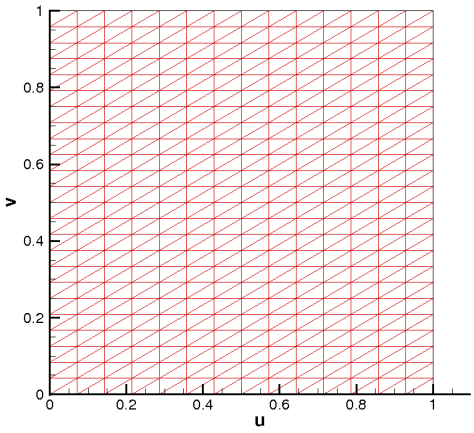
(a) Surface mesh of the initial geometry



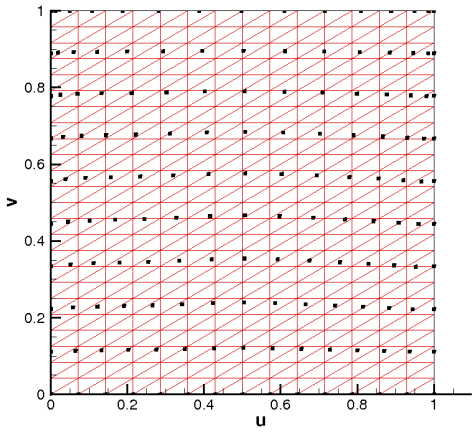
(b) Surface triangulation of the deformed geometry



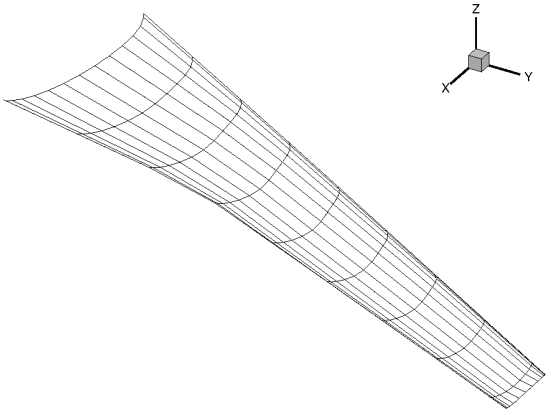
(c) Parameterization of the initial mesh



(d) Parameterization of the triangulation



(e) Overlaying of the parameterizations



(f) New surface mesh

Figure 4: Obtaining a surface mesh from a triangulation of the new surface

mesh, shown in Figure 5(d). Though it may be hard to discern these similarities in such a drastic transformation (from a planar wing to a wing with a winglet), this example demonstrates the capability of the surface mesh movement algorithm to handle large deformations while preserving the mesh quality and characteristics of the original surface mesh. Note that Figure 5(c) is included here to convey the fact that it may not be visually apparent in which triangle the surface mesh lies, but the algorithm can numerically handle the computation.

The third example, Figure 6, shows the mesh characteristic preservation of the algorithm more clearly. In this case, the leading edge of the original wing is reduced by 30% chord. It can be seen that the original and deformed mesh near the root, where the deformation is minimal, are almost identical. The change is smooth and gradual towards the tip as the surface mesh adapts to the new shape.

Finally, Figure 7 shows the same wing in the previous example experiencing a greater extent of deformation whereby the chord is increased by 30%, the wing thickness is reduced by 30%, and the span is increased by 30% in addition to a decrease of 30% in the leading edge sweep. This example shows that CAPRI cannot always produce a tessellation with uniform triangles. Depending on the quality and shape of the CAD face, there may be some distortions introduced which affect the quality of the final surface mesh. This is seen in Figures 3(a) and 3(b), where the distortion of the CAPRI triangulation (red) results in non-smooth surface mesh (black). This necessitates the development of a smoothing algorithm that can correct such irregularities in the surface mesh.

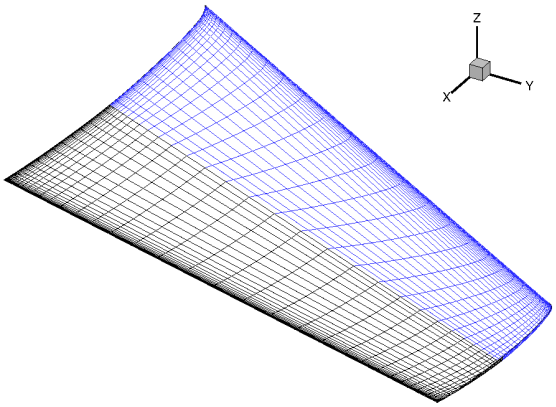
II.H. Winslow Smoothing Algorithm

The Winslow equations are derived from a Laplacian operator applied to the computational coordinates, (ξ, η)

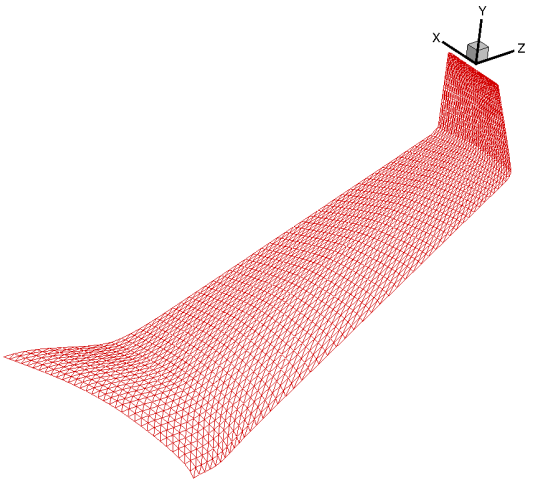
$$\begin{aligned} \nabla^2 \xi &= \xi_{xx} + \xi_{yy} = 0 \\ \nabla^2 \eta &= \eta_{xx} + \eta_{yy} = 0 \end{aligned} \tag{20}$$

The equations describe a smooth distribution of computational coordinates (ξ, η) in physical space (x, y).[56] The Laplace equations satisfy the max-min property, which states that the parameter on the interior domain will not exceed the values on the boundary, i.e. the grid lines will not cross. The Winslow equations are obtained by transformation of the unknown variables x and y to known variables ξ and η in the computational space. For a 2D structured mesh with mesh coordinates at the integer indices (i, j), Eq. 20 can be written as

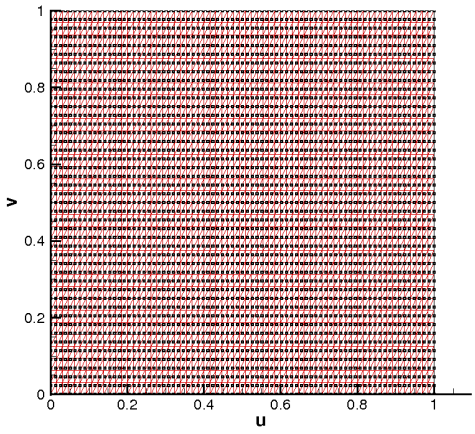
$$\alpha \frac{\partial^2 f}{\partial \xi^2} - 2\beta \frac{\partial^2 f}{\partial \xi \partial \eta} + \gamma \frac{\partial^2 f}{\partial \eta^2} = 0 \tag{21}$$



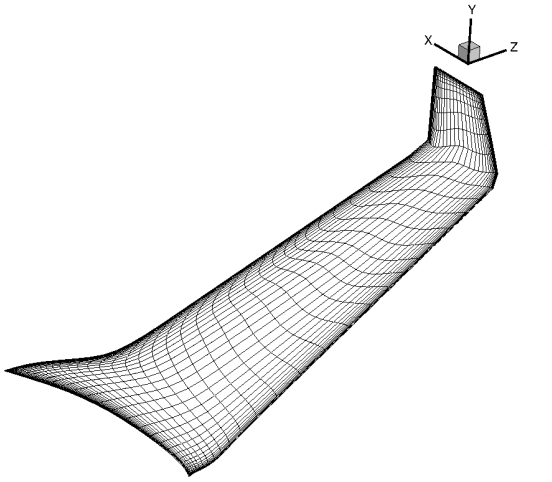
(a) Initial surface made up of 2 patches



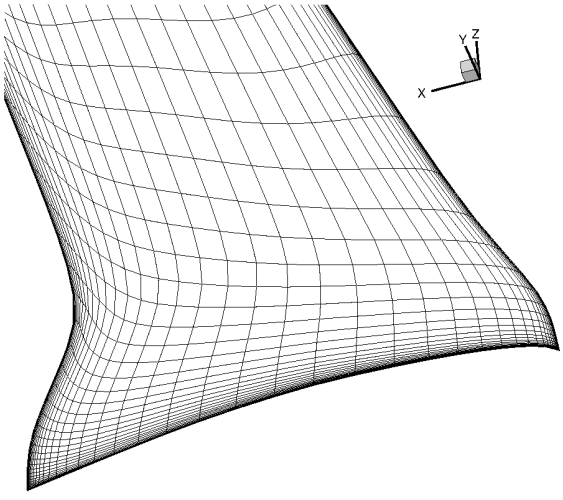
(b) Triangulation of the target surface



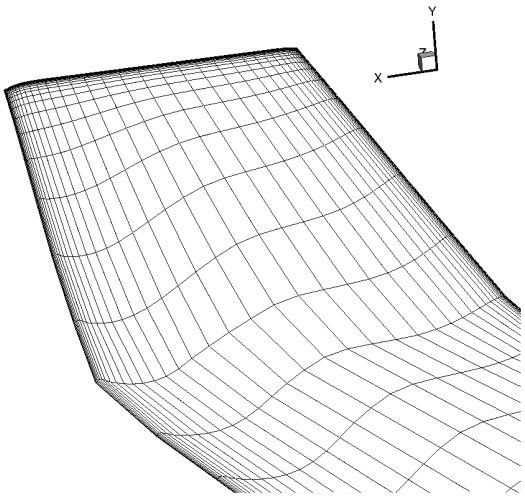
(c) Overlaying of the parameterizations



(d) New surface mesh

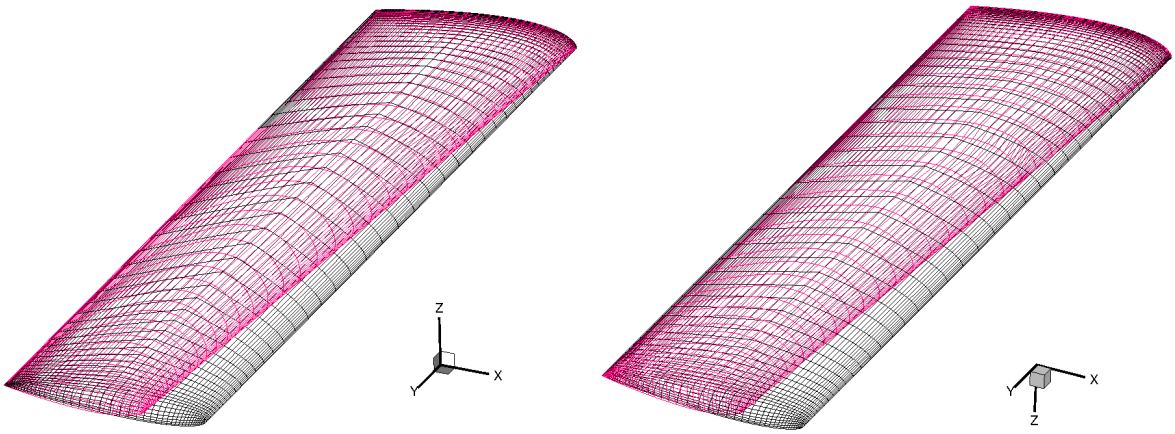


(e) New surface mesh at root



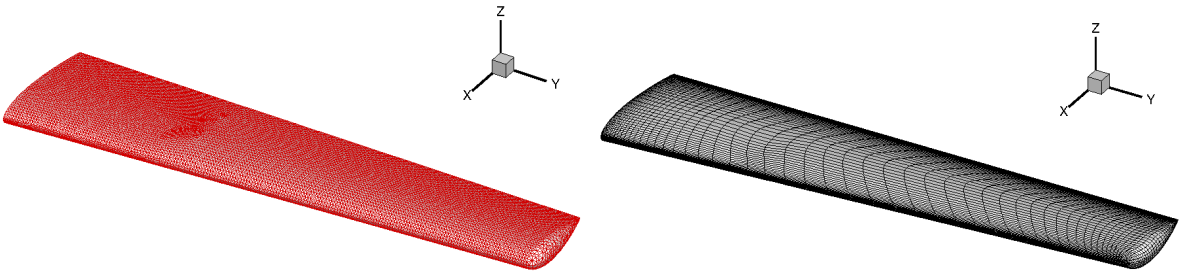
(f) New surface mesh at tip

Figure 5: Obtaining a surface mesh for a wing with a winglet



(a) Top surface (b) Bottom surface

Figure 6: Comparing the mesh quality and characteristics of the original (black) and deformed (pink) meshes.



(a) CAPRI tessellation of the deformed wing (b) Deformed surface mesh

Figure 7: The wing in Figure 6 is modified by +30% chord, −30% thickness, +30% span. Its leading edge is also swept back by 30% chord.

which can be discretized as

$$\alpha(f_{i+1,j}-2f_{i,j}+f_{i-1,j})-\frac{\beta}{2}(f_{i+1,j+1}-f_{i-1,j+1}-f_{i+1,j-1}+f_{i-1,j-1})+\gamma(f_{i,j+1}-2f_{i,j}+f_{i,j-1})=0 \tag{22}$$

where

$$\alpha = \left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2 \tag{23}$$

$$\approx \frac{1}{4}\left[(x_{i,j+1}-x_{i,j-1})^2 + (y_{i,j+1}-y_{i,j-1})^2\right] \tag{24}$$

$$\beta = \frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \eta} + \frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \eta} \tag{25}$$

$$\approx \frac{1}{4}\left[(x_{i+1,j}-x_{i-1,j})(x_{i,j+1}-x_{i,j-1}) + (y_{i+1,j}-y_{i-1,j})(y_{i,j+1}-y_{i,j-1})\right] \tag{26}$$

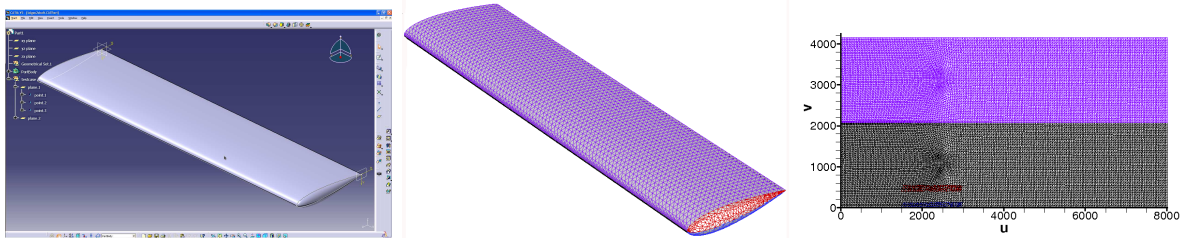
$$\gamma = \left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2 \tag{27}$$

$$\approx \frac{1}{4}\left[(x_{i+1,j}-x_{i-1,j})^2 + (y_{i+1,j}-y_{i-1,j})^2\right] \tag{28}$$

and f is either x or y. The solution to the coordinate position at (i, j) can be obtained by solving Eq. 22 for $f_{i,j}$. Smoothing is applied to the mesh in the CAD uv-space, where each CAD face exists in a coordinate system that is independent of the other surfaces. During smoothing, mesh points are restricted to move within the CAD face boundary to ensure that they remain on the CAD face. Figure 8 displays the CAPRI tessellation of a wing model in CAD parametric coordinates, showing four CAD faces existing as independent entities (i.e. not necessarily connected at the boundary where the connection exists in physical space): two faces on the top surface (a big rectangular face representing the straight portion of the wing and a much smaller parallelogrammatic face representing the tip portion) and two on the bottom. The leading edge in Figure 8(c) in this case happens to be the common edge between the black and purple mesh. This is coincidental, as there is no relationship in the mapping of CAD entities. In order to smooth the mesh across patch boundaries, it is necessary to consolidate all of the CAD faces of a patch from different coordinate systems into one. This process is called coordinate transformation, which is described in Section II.I.

II.I. Coordinate Transformation of Surface Meshes

Coordinate transformation is the process of determining and applying the relationship between two sets of points on different coordinate systems.[57] It is one of the fundamental operations in computer graphics,[58] and has applications in photogrammetry,[59,60] geographic information systems[57] and others. To consolidate all of the CAD faces of a patch, a transformation is applied to the surfaces so that they all exist in one coordinate system. This



(a) CATIA part (b) CAPRI triangulation in physical space (c) CAPRI triangulation in parametric space

Figure 8: CAPRI tessellation of the CAD model. Figure 8(b) shows the water-tight CAPRI triangulation of the wing in physical space. The tessellation in parametric space, Figure 8(c) is color-coded the same way as its corresponding tessellation in physical space, Figure 8(b). The wing itself is consisted of 2 sections, wing body (straight portion), and wing tip. It is consisted of 4 surfaces: the wing body top (purple), wing body bottom (black), wing tip top (red), wing tip bottom (blue). In the physical space (Figure 8(b)), the wing body top and wing body bottom are attached at 2 edges. In the CAD parametric space (Figure 8(c)), they are attached only on 1 edge. Furthermore, the wing tip is attached to the wing body in the physical space. However, they exist as separate entities in the CAD parametric space.

is possible because neighboring surfaces share a common boundary. Along this boundary, CAPRI supplies two sets of overlapping nodes belonging to each of the neighbors. By applying a coordinate transformation, we can establish a relationship between the two systems to fit one set of coordinates into the other.

In general, there are six parameters that characterize a two-dimensional coordinate system. These parameters are:

1. a_0 = translation of the origin in the x direction
2. b_0 = translation of the origin in the y direction
3. φ = rotation of the axes of one coordinate system with respect to the other
4. s_x = scale change in the x axis
5. s_y = scale change in the y axis
6. ε = nonorthorgonality angle between the axes of the two coordinate systems

Basic coordinate transformation methods can be classified based on the number of parameters involved. The four- and six-parameter transformations can be modified to become sets of linear equations. And the formulas for computing a least-squares-based transformation are simple. For the three- and five-parameter transformations, the least-squares solution becomes nonlinear and requires iterations until the solution converges. After experimenting

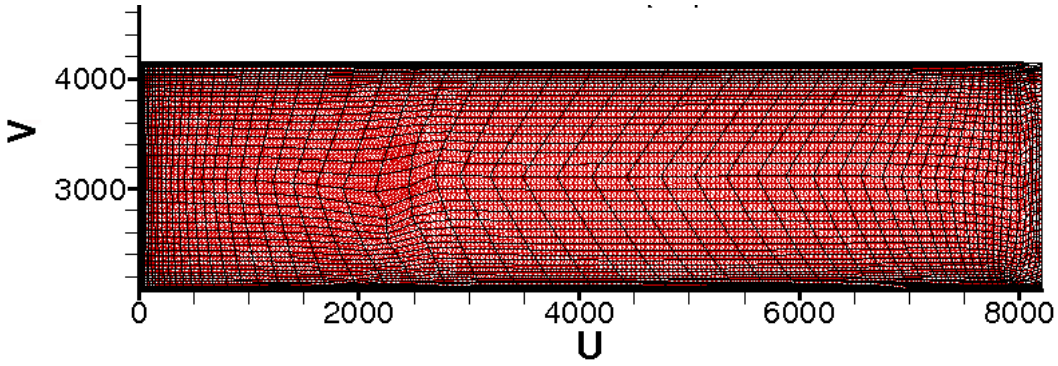


Figure 9: The wing tip section is transformed into the parametric space of the wing span.

with the linear transformations (conformal and affine), obtained using all of the common points between neighboring surfaces, it was determined that the surfaces can be transformed using conformal transformation. Conformal or isogonal transformation involves two systems that may have different scales, and is defined by four parameters: $a_0, b_0, \varphi, s_x = s_y$:

$$\begin{aligned} X &= a_0 + x \cdot \cos\varphi - s_x \cdot y \cdot \sin\varphi \\ Y &= b_0 + x \cdot \sin\varphi + s_y \cdot y \cdot \cos\varphi \end{aligned} \tag{29}$$

Eq. 29 can be expressed in terms of four coefficients (a_0, b_0, a_1, b_1):

$$\begin{aligned} X &= a_0 + a_1 x - b_1 y \\ Y &= b_0 + b_1 x + a_1 y \end{aligned} \tag{30}$$

In many instances, a coordinate transformation brings the neighboring surface to the target coordinate system but the positions of the nodes need to be flipped about the line of attachment between the two surfaces so that they do not fold onto each other. The reflection about an arbitrary line is described in Section II.J. Coordinate transformation is usually carried out using least squares because it provides a best fit between the coordinate systems by analyzing all the common points simultaneously. The result of applying the above transformations is shown in Figure 9. The original CAD parameterization can be seen in Figure 8. We chose to transform surfaces with fewer mesh points to minimize computational effort.

II.J. Reflection About an Arbitrary Line

The equation of the line about which the reflection is calculated is obtained using a least-squares method. The transformation matrix for reflection about either the $x-$ or $y-$axis is:

$$\text{Refl}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{Refl}_y = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{31}$$

In general, the reflection about an arbitrary line is obtained by transforming the line to one of the axes, reflecting in that axis, and then taking the inverse of the first transformation. More specifically, the reflection is accomplished in five steps, as follows:

1. Translate the line to intersect the origin. For an arbitrary line $Ax + By + C = 0$, the translation that maps the y-intersection, $(0, -C/B)$, to the origin is:

$$\text{Transl}_{(A,B,C)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -C/B \\ 0 & 0 & 1 \end{bmatrix} \tag{32}$$

2. Rotate the line about the origin through an angle -θ to coincide with the x-axis. The transformation matrix that describes this transformation is:

$$\text{Rot}_{(A,B,C)} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{33}$$

3. Apply a reflection in the x-axis.

4. Rotate about the origin by θ.

5. Translate by $(0, C/B)$.

The above transformations can be concatenated into one reflection matrix

$$\text{Refl}_{(A,B,C)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -C/B \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & C/B \\ 0 & 0 & 1 \end{bmatrix} \tag{34}$$

$$\text{Refl}_{(A,B,C)} = \begin{bmatrix} \cos^2\theta - \sin^2\theta & 2\sin\theta\cos\theta & \frac{2C}{B}\sin\theta\cos\theta \\ 2\sin\theta\cos\theta & \sin^2\theta - \cos^2\theta & -\frac{2C}{B}\cos^2\theta \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

Since $\tan\theta = -A/B$, it follows that $\cos^2\theta = 1/(1+\tan^2\theta) = B^2/(A^2+B^2)$, $\sin^2\theta = 1-\cos^2\theta = A^2/(A^2+B^2)$, and $\sin\theta\cos\theta = \tan\theta\cos^2\theta = -AB/(A^2+B^2)$. Substituting these expressions into Eq. 35 yields

$$\text{Refl}_{(A,B,C)} = \begin{bmatrix} \frac{B^2-A^2}{A^2+B^2} & -\frac{2AB}{A^2+B^2} & -\frac{2AC}{A^2+B^2} \\ -\frac{2AB}{A^2+B^2} & \frac{A^2-B^2}{A^2+B^2} & -\frac{2BC}{A^2+B^2} \\ 0 & 0 & 1 \end{bmatrix} \tag{36}$$

The above matrix can be scaled by a factor of A^2+B^2 to remove all the denominators in the entries to yield

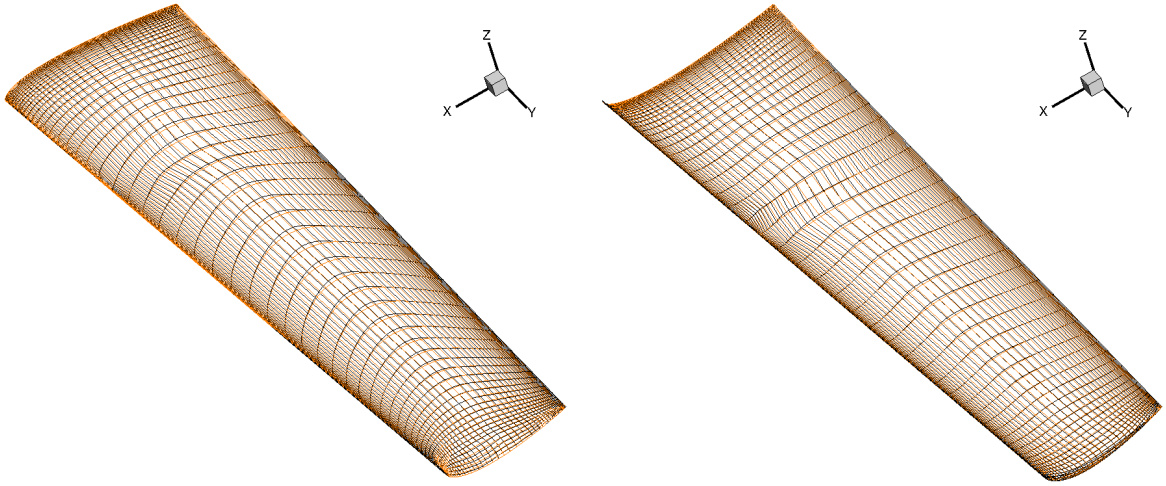
$$\text{Refl}_{(A,B,C)} = \begin{bmatrix} B^2-A^2 & -2AB & -2AC \\ -2AB & A^2-B^2 & -2BC \\ 0 & 0 & A^2+B^2 \end{bmatrix} \tag{37}$$

II.K. Surface Mesh Smoothing Example

Figures 10 and 11 compare the original and smoothed meshes after 2 iterations. In this case, the leading edge of the wing shown in Figure 8(a) has been swept back by 30% chord. The smoothing procedure changes the orginal spacing, i.e. the original characteristics, of the mesh. Thus, the number of iterations should be kept to a minimum. Fortunately, smoothing is 2D is extremely effective and usually a few iterations are sufficient.

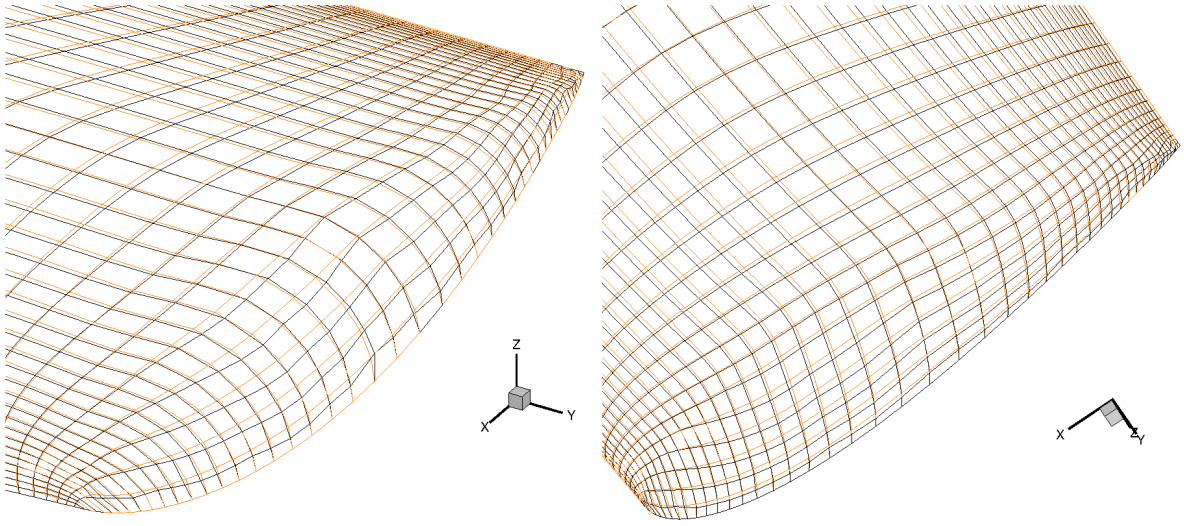
III. Gradient Computation

This section presents examples to examine whether accurate gradients can be obtained using the proposed CAD-based geometry parameterization and surface mesh movement algorithm when integrated with a flow solver,[33] volume-mesh mover[32,61] and optimizer.[34] The baseline wing used for all cases is a straight, rectangular wing with a NACA0018 airfoil profile. The wing has a half span of eight meters and a chord of two meters. All of the optimization results presented are computed on a $1,040,000$ node mesh with an H-topology consisting of 16 blocks. The farfield boundary is approximately 20 chords from the wing, and the off-wall spacing of the mesh is 10^{-3} m. Although the same mesh is used for all optimization cases, the CAD geometry is slightly different in the number of spanwise sections, as required by the problem.



(a) Top surface (b) Bottom surface

Figure 10: Smoothed surface mesh (orange) overlaying the unsmoothed mesh (black) shown in Figure 7(b)



(a) Top surface (b) Bottom surface

Figure 11: Zoomed-in view of the tip region of the smoothed surface mesh (orange) overlaying the unsmoothed mesh (black) shown in Figure 7(b)

III.A. Discrete-Adjoint Gradient Computation

The optimization problem can be posed as minimizing the design objective, $\mathcal{J}$, with respect to the design variables, $\mathcal{X}$, the grid, G, and the conservative flow field variables, Q, within a feasible region of the design space Ω, subject to the constraint that the flow solver and grid movement equations must have converged (i.e the flow residual, $\mathcal{R}$, and the grid movement residual, r, are zero):

$$\begin{aligned} \min_{X} \quad & \mathcal{J}(\mathcal{X}, Q, G) \\ \text{s.t.} \quad & \mathcal{R}(\mathcal{X}, Q, G) = 0 \qquad \forall \quad \mathcal{X} \in \Omega \\ & r(G, G_{\text{surf}}(\mathcal{X})) = 0 \end{aligned} \tag{38}$$

where G_{surf} is the surface mesh of the solid boundary.

In general, there may be other constraints, $\mathcal{C}(\mathcal{X}, Q)$, associated with the design variables or flow conditions; these are considered separately by the optimizer and are not shown here as part of this augmented adjoint formulation. To enforce the constraints in Eq. 38, let the Lagrangian, $\mathcal{L}$, be defined as follows:

$$\mathcal{L}(\mathcal{X}, Q, G, \psi, \lambda) = \mathcal{J}(\mathcal{X}, Q, G) + \psi^{\mathrm{T}}\mathcal{R}(\mathcal{X}, Q, G) + \lambda^{\mathrm{T}} r(G, G_{\text{surf}}(\mathcal{X})) \tag{39}$$

where λ and ψ are Lagrange multipliers. Setting each of the partial derivatives of the Lagrangian to zero then provides optimality conditions for the objective function:

$$\frac{\partial \mathcal{L}}{\partial \lambda} = 0 = r \tag{40}$$

$$\frac{\partial \mathcal{L}}{\partial \psi} = 0 = \mathcal{R} \tag{41}$$

$$\frac{\partial \mathcal{L}}{\partial Q} = 0 = \frac{\partial \mathcal{J}}{\partial Q} + \psi^{\mathrm{T}} \frac{\partial \mathcal{R}}{\partial Q} \tag{42}$$

$$\frac{\partial \mathcal{L}}{\partial G} = 0 = \frac{\partial \mathcal{J}}{\partial G} + \lambda^{\mathrm{T}} \frac{\partial r}{\partial G} + \psi^{\mathrm{T}} \frac{\partial \mathcal{R}}{\partial G} \tag{43}$$

$$\frac{\partial \mathcal{L}}{\partial \mathcal{X}} = \frac{\partial \mathcal{J}}{\partial \mathcal{X}} + \psi^{\mathrm{T}} \frac{\partial \mathcal{R}}{\partial \mathcal{X}} + \lambda^{\mathrm{T}} \frac{\partial r}{\partial G_{\text{surf}}} \frac{\partial G_{\text{surf}}}{\partial \mathcal{X}} \tag{44}$$

Eqs. 40 and 41 represent the residuals of the linear elasticity mesh movement algorithm and the flow, respectively, while Eqs. 42 and 43 represent the discrete adjoint equations for the flow simulation and mesh movement, respectively.

A number of approaches can be taken to find a solution to these optimality conditions to minimize the Lagrangian, including possibilities such as solving for all of the variables simultaneously—the approach that defines simultaneous analysis and design (SAND).[62] Using this one-shot approach, a large-scale solver sets the entire gradient of the Lagrangian

to zero. The sequential approach taken here delegates the computation to the existing specialized solvers. For a given $\mathcal{X}$, Eq. 40 can be solved using the mesh movement code to yield G. Using that solution, the flow solver can be used to solve Eq. 41 to yield Q. The linear systems in Eqs. 42–43 can then be solved in the order they appear to give ψ and each λ. The size of these linear systems is independent of the number of design variables. In cases where nonlinear aerodynamic constraints, such as lift or pitching moment are used, additional adjoint gradient computation has to be performed for each of the constraints.

The evaluation of $\frac{\partial G_{\text{surf}}}{\partial \mathcal{X}}$ in Eq. 44, which is the sensitivity of the surface grid with respect to the design variables, is determined using the finite-difference method of Nemec and Aftosmis.[11]

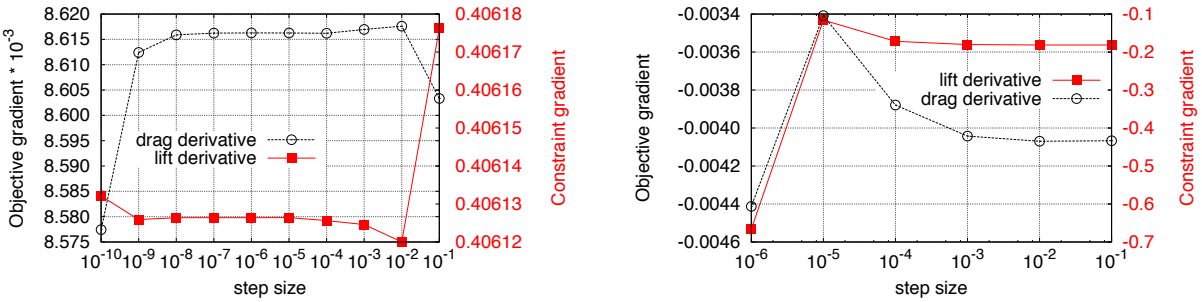
III.B. Computation of Surface Sensivity

The surface sensitivity procedure uses the coordinates of the surface mesh in the parameterized CAD (2D) space. For each CAD face, the CAD parameterized (u, v) coordinates of the surface grid are normalized such that $u, v \in [0, 1]$. The procedure involves the generation of a surface grid for the CAD model to reflect the current values of the design variables, thereby obtaining a baseline model. Then two additional model regenerations and surface mesh movements are obtained for each design variable, which correspond to the plus and minus perturbations. The normalized (u, v) values on the perturbed model are re-scaled by their new (u, v) range from which the corresponding physical coordinates (x, y, z) can be queried from the CAD system. This procedure is very general in that it can be applied to all CAD entities (faces, curves, points), regardless of the method of construction. It is also efficient since the query is done in the parameter space, assuming that the face topologies of baseline and perturbed models are the same. This is a reasonable assumption considering the size of the perturbation. In fact, this condition is checked every time to ensure the validity of the gradients. Changes in topology may necessitate a systematic reduction in the stepsize. In the worst case where changes in the topology are unavoidable, the finite-difference calculations may have to be one-sided or the optimzation procedure may have to be restarted.

The size of the perturbation is chosen based on trial and error. It is 0.002% for most design variables except control point and sectional displacement design variables where a value of 0.0002 mm is used. The accuracy of the derivative with respect to the latter design variables seems to be more sensitive to the step size.

III.C. Verification of Gradient Accuracy

The total gradient calculated using the adjoint method, i.e. $\frac{\partial \mathcal{L}}{\partial \mathcal{X}}$ in Eq. 44, is compared with that calculated using central differencing to ensure that it is sufficiently accurate for



(a) Angle of attack

(b) Tip twist

Figure 12: Objective and constraint gradients as a function of step size for the angle of attack and tip twist design variables.

gradient-based optimization algorithms. The gradient accuracy verification test problem is a lift-constrained drag-minimization of the NACA 0018 wing shown in Figure 8(a). The Mach number is $M = 0.50$. The design variables are the linear twist, τ, of the wing and the angle of attack, ω. This problem is formulated as:

$$\begin{aligned} \min_{\omega,\tau} \quad & C_D S \\ \text{s.t.} \quad & C_L S = (C_L S)_{\text{ref}} \end{aligned} \tag{45}$$

where C_L and C_D are the coefficients of lift and drag, respectively, and S the wetted planform area of the wing. The reference or target lift, $(C_L S)_{\text{ref}}$, is set to the initial value of 1.0153, and $S_{\text{ref}} = 4.249$ units squared.

The second-order finite-difference approximation of a given objective is

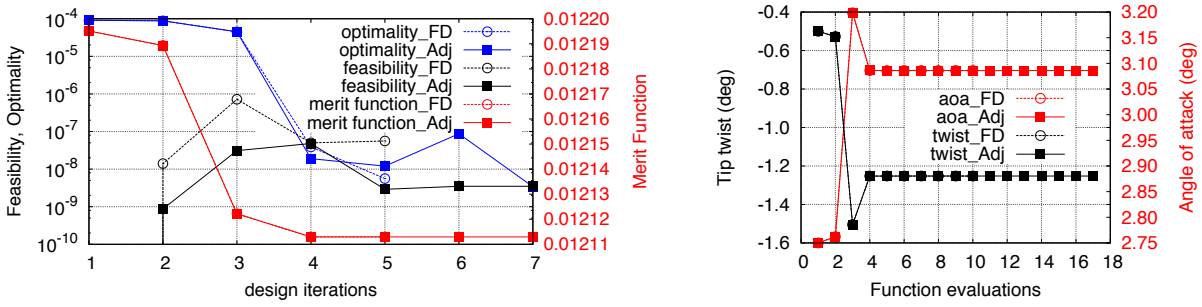
$$\frac{d\mathcal{J}}{d\mathcal{X}} = \frac{\mathcal{J}(\mathcal{X} + \epsilon\mathcal{X}) - \mathcal{J}(\mathcal{X} - \epsilon\mathcal{X})}{2\epsilon} + \mathcal{O}(\epsilon^2) \tag{46}$$

where ϵ is the finite-difference step size. A step size study was conducted to determine a suitable step size for each design variable using the initial conditions, where the twist is $-0.5°$ and angle of attack is $2.75°$. Figure 12 shows the total gradient for the objective and constraint as a function of the step size used. Based on this study, step sizes of 10^{-2} and 10^{-6} are chosen for the twist and angle of attack design variables, respectively, to obtain the finite-difference gradients.

The objective function and constraint accuracy are presented in Table 1 for the first iteration. The agreement between the adjoint and centered-difference gradient values is excellent. Small differences of approximately 1% for both design variables are attributed to numerical errors of the finite-difference operation. Figure 13 compares the optimization convergence histories for the finite-difference and adjoint methods. Both have approximately

Table 1: Gradient accuracy for the lift-constrained drag-minimization problem.

Design variable	Method	Objective gradient	Constraint gradient
Angle of attack	Adjoint	-0.0040671	-0.1786727
	Finite difference	-0.0040655	-0.1812989
Tip twist	Adjoint	0.0086162	0.4000044
	Finite difference	0.0086163	0.4061264



(a) Optimizer

(b) Design variables

Figure 13: Convergence history for the finite difference (FD) and adjoint (Adj) methods.

the same degree of convergence and yield nearly identical results: The final angle of attack of 3.09° is higher than the initial in order to maintain the lift while a slightly lower negative final twist angle of 1.25° is added to minimize the induced drag. These results are consistent with those published in the literature.[33,63] Note that the optimality reported here is a measure of the gradient convergence provided by SNOPT, which takes constraint gradients into account, while the feasibility describes how well the optimizer is able to meet the constraint. Finally, the merit function, also reported by SNOPT, is an augmented Lagrangian merit function. When the constraints are satisfied, the merit function is equal to the objective. For example, for the objective function defined in Eq. 45, the merit function will equal the coefficient of drag times the wetted wing planform area at convergence.

Finally, the objective function for different values of the twist angle with the angle of attack chosen to satisfy the above lift constraint is calculated so as to map out the feasible region of the design space, shown in Figure 14. It can be seen that the minimum drag is at $\tau = -1.25°$, which confirms the optimization results shown in Figure 13(b) and verifies that the optimum was found.

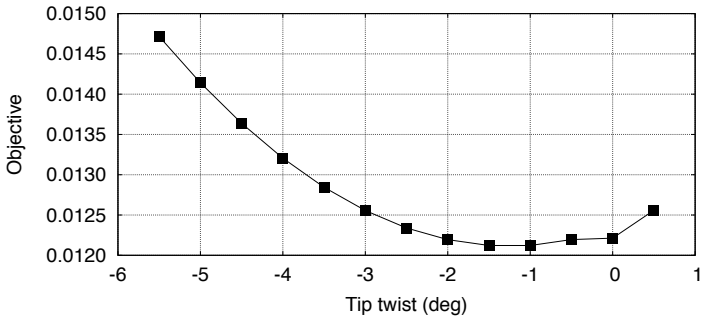


Figure 14: Objective function value for different twist angles at the specified lift.

III.D. Simultaneous Wave and Induced Drag Reduction

This test case validates the use of twist, sweep angle and airfoil thickness by the optimizer to reduce the wave and induced drag that is present on the original unswept wing in transonic flow conditions with $M = 0.76$. Table 2 shows the upper and lower limits set for each of the design variables. The objective is to minimize the drag while maintaining a lift of $(C_L S)_{ref} = 2.125$.

Figures 15 and 16 show the Mach number contours and sectional coefficient of pressure distribution over the initial and optimized wings. The optimizer has swept the wing back to the limit set at the beginning of the optimization in an attempt to reduce the wave drag. A strong shock was present over much of the initial wing, but is significantly weakened in the optimized wing. The thickness is also reduced to the minimum allowable value in order to reduce the wave drag, while a negative twist angle of 3.1° is added to reduce the induced drag. Finally, the angle of attack is increased to 6.03° to maintain the required lift. Note also how, other than a slight distortion at the tip, the characteristics of the initial surface mesh are generally preserved in the final mesh, as seen in Figure 15.

Figure 17 shows the convergence histories for the wave drag reduction case. The optimizer is able to reduce the optimality by about 5 orders of magnitude, achieving an overall reduction of drag of 47% from an initial value of 0.1946 to the final value of 0.1029, while achieving the specified lift. An optimality reduction of five orders of magnitude is only possible if gradients are computed accurately. This example demonstrates that the proposed surface mesh movement algorithm preserves mesh quality and enables accurate gradient computation even under substantial shape changes.

Table 2: Upper and lower limits for the design variables in the wave drag reduction problem

Design variable	Lower limit	Initial Value	Upper limit	Units
Sweep	-25	0	25	° (rel. to the init. sweep)
Tip twist	-15	0	15	° (rel. to the init. twist)
Airfoil thickness	-33	0	10	% (initial thickness)
Angle of attack	-5.25	3.75	8.75	°

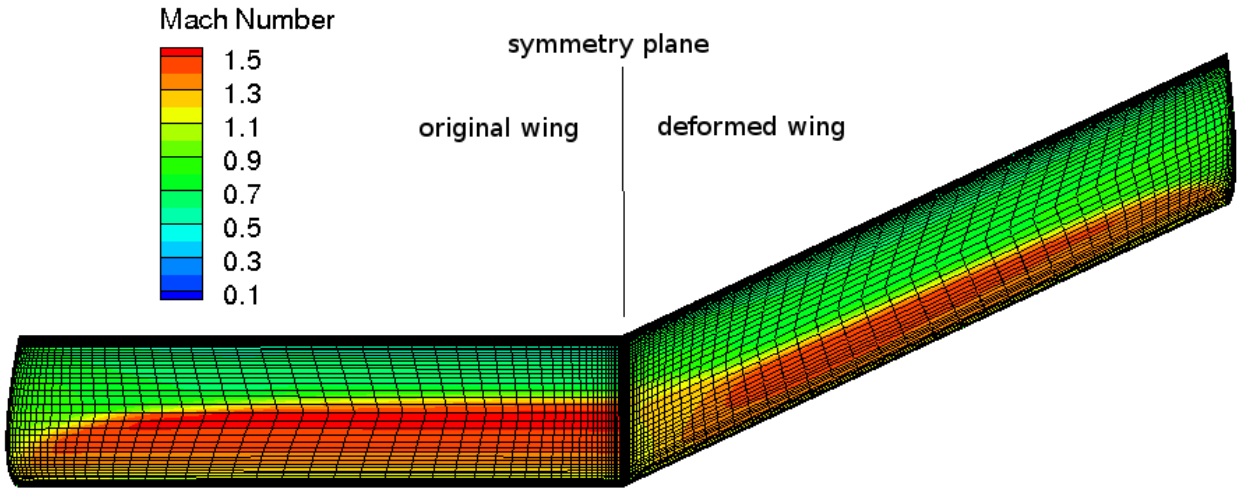


Figure 15: Contours of Mach number on the surface of the initial (left) and optimized (right) wing.

IV. Conclusions

We have presented a surface mesh movement technique for three-dimensional CAD-based aerodynamic shape optimization. The new surface mesh is guaranteed to lie on the CAD geometry and have the same general characteristics (mesh density and spacing) as the original surface mesh even for large shape changes. The procedure is efficient in that once the new geometry is created and a corresponding tessellation obtained, no subsequent communication with the CAD description is required to determine the surface mesh points in CAD parametric space. All calculations are done in two dimensions. The discrete adjoint equations have been augmented to include volume and surface mesh sensitivities, ensuring efficiency and accuracy in the gradient computations. The CAD surface sensitivity derivatives are obtained using centered-difference approximations where the deformed surface is obtained for positive and negative perturbations of the geometry and the difference calculated. The three-dimensional perturbed surface meshes are determined from their scaled uv-coordinates. This

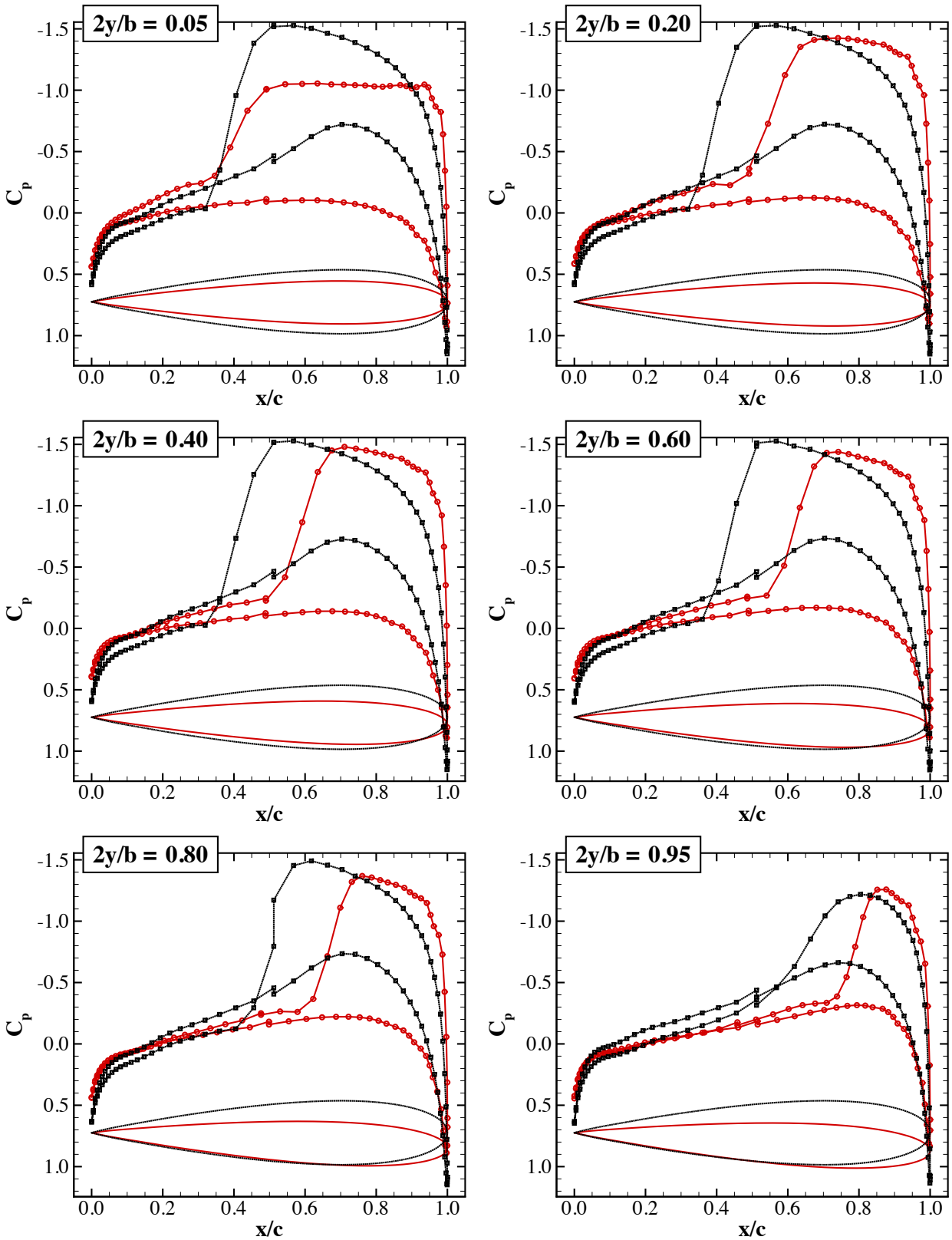
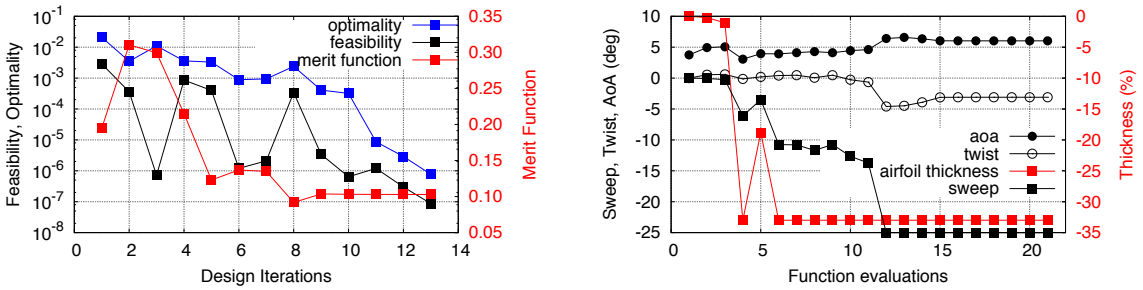


Figure 16: Sectional C_p distribution over the initial (black) and optimized (red) wing.



(a) Optimizer (b) Design Variables

Figure 17: Convergence history for the wave drag reduction case with 4 design variables.

method is straightforward and efficient, as it incurs no additional calculations by the CAD software in going from parametric to three-dimensional space. The validity and accuracy of the algorithm for three-dimensional aerodynamic shape optimization with CAD geometries on multi-block structured grids are demonstrated. In particular, it has been shown that the surface mesh movement algorithm can handle very large changes in the geometry while maintaining the initial mesh quality and geometric fidelity.

V. Acknowledgments

This work was sponsored by the Canada Research Chairs Program, MITACS, Bombardier Aerospace, the University of Toronto, and Zonta International. Their financial support is gratefully acknowledged.

References

[1]Kahrs, M., "The Heart of IGES," *Software: Practice and Experience*, Vol. 25, No. 8, 1995, pp. 935–946.

[2]3D Systems Inc., *Stereolithography Interface Format Specification*, 1988.

[3]"STEP Application Handbook: ISO 10303," June 2006.

[4]Farouki, R. T., "Closing the Gap Between CAD Model and Downstream Application," *SIAM News*, June 23 1999.

[5]*Current State of the Art on Multidisciplinary Design Optimization (MDO)*, American Institute of Aeronautics and Astronautics, 1991.

[6]Tautges, T. J., "The Common Geometry Module (CGM): A Generic, Extensible Geometry Interface," *Proceedings of the 9th International Meshing Roundtable*, Sandia National Laboratories, New Orleans, Louisiana, October 2000, pp. 337–359.

[7]Beall, M. W., Walsh, J., and Shepphard, M. S., "Accessing CAD Geometry for Mesh Generation," *Proceedings of the 12th International Meshing Roundtable*, Sandia National Laboratories, Santa Fe, NM, September 2003, pp. 33–42.

[8]Alonso, J. J., Martins, J. R. R. A., Reuther, J. J., and Haimes, R., "High-Fidelity Aero-Structural Design Using a Parametric CAD-Based Model," AIAA Paper 2003-3429, 16th AIAA Computational Fluid Dynamics Conference, Orlando, FL, June 2003.

[9]Wood, A., Sainmont, C., Merchant, A., Leblond, D., and Laurendeau, E., "CAD-Based Parametric Design Optimization with Structured Multi-block Grid Re-Generation," *58th Annual CASI Conference*, Montreal, QC, 2011.

[10]Brock, W. E., Burdyshaw, C., Karman, S. L., Betro, V. C., Hilbert, C. B., Anderson, W. K., and Haimes, R., "Adjoint-Based Design Optimization Using CAD Parameterization through CAPRI," AIAA Paper 2012–0968, 50th AIAA Aerospace Sciences Meeting and Exhibit, Nashville, Tennessee, January 2012.

[11]Nemec, M. and Aftosmis, M. J., "Aerodynamic Shape Optimization Using a Cartesian Adjoint Method and CAD Geometry," AIAA Paper 2006–3456, 24th AIAA Applied Aerodynamics Conference, San Francisco, CA, June 2006.

[12]"OpenCASCADE," 2012, [Online; accessed 20-April-2012].

[13]Jones, W. T., Lazzara, D., and Haimes, R., "Evolution of Geometric Sensitivity Derivatives from Computer Aided Design Models," AIAA Paper 2010-9128, 13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Fort Worth, Texas, September 2010.

[14]Kleinveld, S., Roge, G., Daumas, L., and Dinh, Q., "Differentiated Parametric CAD Used Within the Context of Automatic Aerodynamic Design Optimization," AIAA Paper 2008-5952, 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia, September 2008.

[15]Yu, G., Müller, J.-D., Jones, D., and Christakopoulos, F., "CAD-based Shape Optimisation Using Adjoint Sensitivities," *Computers and Fluids*, Vol. 46, 2011, pp. 512–516.

[16]Xu, S., Jahn, W., and Müller, J.-D., "CAD-based Shape Optimization with CFD Using a Discrete Adjoint," *International Journal for Numerical Methods in Fluids*, Vol. 74, 2014, pp. 153–168.

[17]Robinson, T. T., Armstrong, C. G., Chua, H. S., Othmer, C., and Grahs, T., "Optimizing Parameterized CAD Geometries Using Sensitivities Based on Adjoint Functions," *Computer-Aided Design and Applications*, Vol. 9, No. 3, 2012, pp. 253–268.

[18]Zhang, W.-H., Beckers, P., and Fleury, C., "A Unified Parametric Design Approach to Structural Shape Optimization," *International Journal for Numerical Methods in Engineering*, Vol. 38, 1995, pp. 2283–2292.

[19]Toivanen, J. I. and Martikainen, J., "A new Method for Creating Sparse Design Velocity Fields," *Computer Methods in Applied Mechanics and Engineering*, Vol. 196, 2006, pp. 528–537.

[20]Hardee, E., Chang, K.-H., Tu, J., Choi, K. K., Grindeanu, I., and Yu, X., "A CAD-based Design Parameterization for Shape Optimization of Elastic Solids," *Advances in Engineering Software*, Vol. 30, 1999, pp. 185–199.

[21]Chen, J., Freytag, M., and Shapiro, V., "Shape Sensitivity of Constructively Represented Geometric Models," *Computer Aided Geometric Design*, Vol. 25, 2008, pp. 470–488.

[22]Choi, K. K. and Chang, K.-H., "A Study of Design Velocity Field Computation for Shape Optimal Design," *Finite Elements in Analysis and Design*, Vol. 15, 1994, pp. 317–541.

[23]Choi, J. H., Won, J. H., and Yoon, J. M., "Boundary Method for Shape Design Sensitivity Aanlysis in the Optimization of Three-Dimensional Elastostatics," *6th World Congresses of Structural and Multidisciplinary Optimization*, Rio de Janeiro, 30 May - 03 June 2005.

[24]Chen, S. and Tortorelli, D. A., "Three-dimensional Shape Optimization with Variational Geometry," *Structural Optimization*, Vol. 13, 1997, pp. 81–94.

[25]Haimes, R. and Dannenhoffer, J. F., "The Engineering Sketch Pad: A Solid-Modeling, Feature-Based, Web-Enabled System for Building Parametric Geometry," AIAA Paper 2013–3073, 21st AIAA Computational Fluid Dynamics Conference, San Diego, CA, June 2013.

[26]Nemec, M. and Aftosmis, M. J., "Adjoint Sensitivity Computations for an Embedded-Boundary Cartesian Mesh Method," *Journal of Computational Physics*, Vol. 227, No. 4, Feb. 2008, pp. 2724–2742.

[27]Choi, S., Alonso, J. J., and Kroo, I. M., "Multifidelity Design Optimization of Low-Boom Supersonic Jets," *Journal of Aircraft*, Vol. 45, No. 1, 2008, pp. 106–118.

[28]Nemec, M. and Aftosmis, M. J., "Parallel Adjoint Framework for Aerodynamic Shape Optimization of Component-Based Geometry," AIAA Paper 2011–1249, 49th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, FL, January 2011.

[29]Samareh, J. A., "Survey of Shape Parametrization Techniques for High-Fidelity Multidisciplinary Shape Optimization," *AIAA Journal*, Vol. 39, No. 5, 2001, pp. 877–883.

[30]Fudge, D. M., Zingg, D. W., and Haimes, R., "A CAD-Free and a CAD-Based Geometry Control System for Aerodynamic Shape Optimization," AIAA Paper 2005–0451, 43rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, January 2005.

[31]Haimes, R. and Follen, G., "Computational Analysis PRogramming Interface," *Proceedings of the 6th International Conference on Numerical Grid Generation in Computational Field Simulations*, University of Greenwich, June 1998.

[32]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, 2008, pp. 1695–1704.

[33]Hicken, J. E. and Zingg, D. W., "A Parallel Newton-Krylov Solver for the Euler Equations Discretized Using Simultaneous Approximation Terms," *AIAA Journal*, Vol. 46, No. 11, 2008, pp. 2773–2786.

[34]Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm For Large-Scale Constrained Optimization," *SIAM Review*, Vol. 47, No. 1, 2005, pp. 99–131.

[35]Sheffer, A., Blacker, T., Clements, J., and Bercovier, M., "Virtual Topology Construction and Applications," *Theory and Practice of Geometric Modeling, Blaubeuren II*, Springer-Verlag, 1996, pp. 247–259.

[36]Sheffer, A., Blacker, T., Clements, J., and Bercovier, M., "Virtual Topology Operators for Meshing," *International Journal of Computational Geometry and Applications*, Vol. 10, 2000, pp. 309–331.

[37]Inoue, K., Itoh, T., Yamada, A., Furuhata, T., and Shimada, K., "Face Clustering of a Large-Scale CAD Model for Surface Mesh Generation," *Computer-Aided Design*, Vol. 33, 2001, pp. 251–261.

[38]Lee, T.-Y. and Yen, S.-W., "Texture Mapping on 3D Surfaces Using Clustering-Based Cutting Paths," *International Journal for Computational Science and Engineering*, Vol. 3, No. 1, 2007, pp. 71–79.

[39]Sheffer, A., Praun, E., and Rose, K., "Mesh Parameterization Methods and their Applications," *Foundations and Trends in Computer Graphics and Vision*, Vol. 2, No. 2, 2006, pp. 105–171.

[40]Floater, M. S. and Hormann, K., "Surface Parameterization: a Tutorial and Survey," *Advances in Multiresolution for Geometric Modelling*, edited by N. A. Dodgson, M. S. Floater, and M. A. Sabin, Springer Verlag, 2005, pp. 157–186.

[41]Desbrun, M., Meyer, M., and Alliez, P., "Intrinsic Parameterizations of Surface Meshes," *Computer Graphics Forum*, Vol. 21, 2002.

[42]Degener, P., Meseth, J., and Klein, R., "An Adaptable Surface Parameterization Method," *Proceedings of the 12th International Meshing Roundtable*, 2003, pp. 201–213.

[43]Tutte, W. T., "How to Draw a Graph," *Proceedings of The London Mathematical Society*, Vol. 13, No. 3, 1963, pp. 743–767.

[44]Michikawa, T., Kanai, T., Fujita, M., and Chiyokura, H., "Multiresolution Interpolation Meshes," *Proceedings of Pacific Graphics*, 2001, pp. 60–69.

[45]Floater, M. S., "Parametrization and Smooth Approximation of Surface Triangulations." *Computer Aided Geometric Design*, 1997, pp. 231–250.

[46]Lévy, B., Petitjean, S., Ray, N., and Maillot, J., "Least Squares Conformal Maps for Automatic Texture Atlas Generation," *SIGGRAPH 02 Conference Proceedings*, 2002, pp. 362–371.

[47]Pinkall, U., Juni, S. D., and Polthier, K., "Computing Discrete Minimal Surfaces and Their Conjugates," *Experimental Mathematics*, Vol. 2, 1993, pp. 15–36.

[48]Floater, M. S., "Mean value coordinates," *Computer Aided Geometric Design*, Vol. 20, No. 1, March 2003, pp. 19–27.

[49]Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., and Stuetzle, W., "Multiresolution analysis of arbitrary meshes," *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, ACM, New York, NY, USA, 1995, pp. 173–182.

[50]Sheffer, A. and de Sturler, E., "Smoothing an Overlay Grid to Minimize Linear Distortion in Texture Mapping," *ACM Transactions on Graphics*, Vol. 21, 2002, pp. 874–890.

[51]Gu, X. and Yau, S.-T., "Global Conformal Surface Parameterization," *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '03, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003, pp. 127–137.

[52]Hormann, K., "Barycentric Mappings," June 2007.

[53]Floater, M. S. and Pham-Trong, V., "Convex Combination Maps Over Triangulation, Tilings, and Tetrahedral Meshes," *Advances in Computational Mathematics*, Vol. 25, No. 4, 2006, pp. 347–356.

[54]Floater, M. S., Hormann, K., and Kós, G., "A General Construction of Barycentric Coordinates over Convex Polygons," *Advances in Computational Mathematics*, Vol. 24, No. 4, 2006, pp. 311–331.

[55]Haimes, R., *CAPRI: Computational Analysis Programming Interface, A Solid Modeling Based Infrastructure for Engineering Analysis and Design*, 2nd ed., December 2004.

[56]Jr., S. L. K., "Virtual Control Volumes for Two-Dimensional Unstructured Elliptic Smoothing," *Proceedings of the 19th International Meshing Roundtable, Chattanooga, TN*, 2010, pp. 121–142.

[57]Greenfeld, J. S., "Least Squares Weighted Coordinate Transformation Formulas and Their Applications," *Journal of Surveying Engineering*, Vol. 123, No. 4, 1997, pp. 147–161.

[58]Foley, J. D. and Van Dam, A., *Fundamentals of interactive computer graphics*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1982.

[59]Meid, A., "Object Based Determination of Comparator Coordinates in Blurred Images," *International Archives of Photogrammetry and Remote Sensing*, Vol. 29, No. 5, 1993, pp. 79–86.

[60]Lowe, D., "Object Recognition from Local Scale-Invariant Features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2, 1999, pp. 1150–1157.

[61]Hicken, J. E. and Zingg, D. W., "Aerodynamic Optimization Algorithm with Integrated Geometry Parameterization and Mesh Movement," *AIAA Journal*, Vol. 48, No. 2, February 2010, pp. 400–413.

[62]Newman III, J. C., Taylor III, A. C., Barnwell, R. W., Newman, P. A., and Hou, G. J.-W., "Overview of Sensitivity Analysis and Shape Optimization for Complex Aerodynamic Configurations," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 87–96.

[63]Mader, C. A. and Martins, J. R. R. A., "Optimal Flying Wings: A Numerical Optimization Study," *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, HI, April 2012, AIAA 2012-1758.