Author version

Published as:

Ruh, M. L., & Hwang, J. T. (2025). Differentiable surface mesh deformation for non-conformal, independently parameterized components with large shape changes. In AIAA AVIATION FORUM AND ASCEND 2025 (Paper No. AIAA 2025-3683). https://doi.org/10.2514/6.2025-3683

BibTeX for this reference:

https://lsdolab.github.io/lsdo-publications/publication/ruh2025differentiable/

```
@inproceedings{ruh2025differentiable,
  author = {Marius L. Ruh and John T. Hwang},
  title = {Differentiable Surface Mesh Deformation for Non-Conformal, Independently
    Parameterized Components with Large Shape Changes},
  booktitle = {AIAA AVIATION FORUM AND ASCEND 2025},
  year = {2025},
  doi = {10.2514/6.2025-3683},
  url = {https://doi.org/10.2514/6.2025-3683}
}
```

Differentiable Surface Mesh Deformation for Non-Conformal, Independently Parameterized Components with Large Shape Changes

Marius L. Ruh*, John T. Hwang†
University of California San Diego, 9500 Gilman Dr, La Jolla, CA 92093

System-level aircraft conceptual design can be viewed as a high-dimensional, multidisciplinary design optimization (MDO) problem involving potentially thousands of design variables across multiple subsystems and design conditions. Recent advances in modeling and adjoint-based sensitivity analysis have enabled the use of low-fidelity physics-based models to solve this complex problem in a computationally efficient manner using established MDO algorithms. However, integrating higher-fidelity aerodynamic simulations into comprehensive, large-scale MDO of aircraft concepts remains challenging. This is in large part due to the increased computational cost and the large geometric changes that may occur at the conceptual design stage. Since many mid- to high-fidelity aerodynamic solvers require conforming surface meshes, there is a need for a general deformation method that efficiently updates surface mesh nodes based on changes to the design geometry in a differentiable manner. In this paper, we aim to fill this need by developing such a method using a two-step process: 1) recompute intersections between components given large geometric changes and 2) deform the surface mesh using thin plate splines to conform to the updated intersections. The method is reasonably efficient and all steps are differentiable. We test our method on a simple tube-and-wing geometry for which we translate and rotate the wing with respect to the fuselage. The method performs well for all considered rotation angles (± 10 degrees) with over 95% of triangles being of acceptable quality. For translations, the performance is similar up to 1.5 times the chord length. We couple the method with a panel-method aerodynamic solver, confirming its applicability in aircraft conceptual design.

M	closed manifold defined by a union of parametric surfaces (or curves)
$i \in \mathbb{N}$	index of a parametric surface/curve contained in M
$\vec{u}$	vector of parametric coordinates
$\vec{x} \in \mathbb{R}^3$	point in Euclidean space
$\hat{\vec{x}} \in \mathbb{R}^3$	point in Euclidean space that lies on M
$E_M : (i, \vec{u}) \mapsto \hat{\vec{x}}$	manifold evaluation map
$E_M^{-1} : \hat{\vec{x}} \mapsto (i, \vec{u})$	manifold inverse evaluation map
$P : \vec{x} \mapsto \hat{\vec{x}}$	projection map
$\|\cdot\|$	L2 norm
$\mathcal{P}$	generic point cloud

I. Introduction

Large-scale multidisciplinary design optimization (MDO) of aircraft concepts aims to efficiently search a high-dimensional design space to find the best design that meets performance and safety requirements. To capture the coupling between the various subsystem disciplines (e.g., airframe and powertrain), we require a comprehensive system model for all relevant disciplines (e.g., weights, aerodynamics, propulsion) across a large number of design conditions that represent normal (i.e., anticipated) modes of operation (e.g., steady cruise) as well as failure conditions such as one-engine-inoperative (OEI) or structural failure conditions. At the conceptual level, regression-based models

*PhD Candidate, Department of Mechanical and Aerospace Engineering, AIAA student member
†Associate Professor, Department of Mechanical and Aerospace Engineering, AIAA senior member

are frequently chosen to model the individual subsystems due to fast evaluation times and reasonable accuracy for conventional aircraft configurations. However, for novel aircraft concepts, such as electric vertical take-off and landing (eVTOL) vehicles, using regression-based models is often not feasible due to a lack of historical data. Physics-based models do not have this limitation and often produce more interpretable results, based on the underlying physics and modeling assumptions.

There are several challenges that arise when using physics-based models in large-scale MDO of aircraft concepts, including the increased complexity of the system model, the need for greater robustness of numerical solvers, and implementing sensitivity analysis (e.g., via the adjoint method) to efficiently search the high-dimensional design space. Recent advances in automated adjoint-based sensitivity analysis [1] have enabled the construction of comprehensive and differentiable system models of aircraft concepts with reduced implementation effort. The authors have leveraged automated sensitivity analysis in recent studies [2–4], to optimize NASA's lift-plus-cruise air taxi concept using physics-based models such as blade element momentum (BEM) theory and the vortex lattice method (VLM) for aerodynamics, beam theory for the wing structure, frequency-domain tonal noise and empirical broadband noise models for aeroacoustics, equivalent circuit models (ECM) for motor analysis, and others. We considered steady on-design conditions such as hover and cruise, a quasi-steady transition segment with ten transition points, and off-design conditions that simulate a one-engine-inoperative (OEI) scenario and structural sizing scenarios. In total, we considered 18 design conditions, and the optimization problem included over 300 design variables and over 200 constraints. We achieved a feasible design with an improved objective (gross weight) in about 17 hours on a standard desktop computer.

One of the main limitations of these studies was the modeling fidelity, particularly in the aerodynamics discipline. While the VLM can predict lift and induced drag with reasonable accuracy for thin wings at low flight speeds, it cannot capture viscous drag or model solid bodies. Therefore, important aerodynamic interactions, e.g., between the fuselage and the wing or the booms and the wing, cannot be captured accurately, which reduces the interpretability of the final optimized design. To address this limitation, panel methods (see, e.g., Drela [5] or Katz and Plotkin [6]) can be applied, which are based on potential flow theory and can model non-lifting surfaces such as the fuselage. One of the main benefits of panel methods are their relative computational efficiency compared to computational fluid dynamics (CFD). However, panel methods are limited to subsonic flows and do not capture viscous effects, without special treatment.

Solving the Reynold-Averaged Navier-Stokes (RANS) equations is a significant improvement in terms of fidelity compared to panel methods as phenomena like turbulence, flow separation, and compressibility are described by the governing equations and can be modeled with reasonable accuracy. For design optimization, RANS analysis can be applied, e.g., using software frameworks such as ADFlow [7] or the SU2 code base [8], which provide adjoints to enable aerodynamic shape optimization (ASO). Research on high-fidelity MDO using RANS-based aerodynamic solvers [9–14] has progressed significantly over the past decade. However, many of these studies considered well-established aircraft concepts like the tube-and-wing configuration and focus on aerostructural design optimization for a single subsystem (e.g., the main wing) and one or only a few design conditions. For novel aircraft concepts, it is important to also consider other subsystems and design conditions that may size the aircraft.

A challenge of using CFD (and panel methods) in aircraft conceptual design (in addition to the increased computational cost) are the significant changes to the design geometry that are expected at the conceptual level, where the configuration is permitted to change freely. Some CFD approaches like the overset (Chimera) method or cut-cell (immersed-boundary) methods are well-suited to deal with large geometric changes as they do not rely on conforming surface meshes that depend on component intersections. On the other hand, CFD approaches based on unstructured meshes (and most panel methods) do require conforming surface meshes that need to be updated as intersections change to maintain valid mesh topology. In addition, for comprehensive design, the coupling of disciplines such as aerodynamics and structures often necessitates knowledge of component intersections to effectively define the transfer of state variables (e.g., pressures and displacements) across solver domains. Thus, in the context of comprehensive system-level large-scale MDO of aircraft, an algorithm or methodology that 1) efficiently computes component intersections, 2) updates surface mesh nodes accordingly, 3) permits large changes to the design geometry, and 4) is differentiable, would be an important step toward integrating high-fidelity aerodynamics in conceptual design. We identify the lack of a method that meets all four criteria as a gap in the current literature.

In this paper, we aim to fill this research gap by proposing a method that meets all four criteria. Our method first computes component intersections by solving an optimization problem whose solution is the set of parametric coordinates that lie on the intersection curve of parametric surfaces. This process is agnostic of the magnitude of the geometric change and allows for intersections to cross boundaries of parametric surfaces. Next, the method combines thin plate spline (TPS) deformation with a projection step to update the surface mesh nodes based on the intersections. The method maintains acceptable mesh quality for medium to large geometric changes, and all steps are differentiable

and reasonably fast to evaluate. We provide an initial demonstration based on a tube-and-wing reference geometry, for which we translate and rotate the wing with respect to the fuselage and pass the resulting deformed mesh into a panel code. The results suggest the method's potential to enable the integration of mid- to high-fidelity aerodynamic solvers in comprehensive large-scale MDO of aircraft concepts.

The remainder of this paper proceeds as follows. In Sec. II, we provide a brief overview of our approach to geometry modeling, and give a summary of existing mesh deformation methods and medium to high-fidelity aerodynamic solvers. Next, in Sec. III, we describe our surface mesh deformation method in detail, covering our approach for computing the intersections as well as the thin plate spline (TPS) deformation step. We then present the results of this paper in Sec. IV, showing the methods ability to effectively handle large geometric changes while preserving acceptable mesh quality for the evaluation of a panel code.

II. Background

A. Geometry Modeling and Manipulation

We use a geometry-centric approach with a high-fidelity representation of the outer mold line (OML) that is generated using NASA's Open Vehicle Sketch Pad (OpenVSP) software [15]. Another frequently used geometry modeler is Engineering Sketch Pad (ESP) [16], which places a stronger emphasis on flexibility and rule-based design. OpenVSP uses B-spline (basis spline) surfaces, which are advantageous due to their analytic form and their ability to smoothly represent complex geometries with few control points. During optimization, changes in the central geometry are effectively propagated to physics-based solvers through the use of free-form deformation (FFD). Free-form deformation (see, e.g., [17]) is a geometric modeling technique used to smoothly manipulate complex shapes by embedding them in a flexible parametric lattice, typically defined by control points. By adjusting the positions of these control points, FFD allows for intuitive and efficient transformations of the embedded shape while preserving its continuity, making it an effective tool in aerodynamic shape optimization [18, 19].

B. Mesh Deformation Methods

Mesh deformation refers to the process of updating mesh nodes according to geometric changes. While in this paper, the primary focus is on surface meshes, the subsequent methods also extend to volume meshes. In general, mesh deformation methods can be grouped into interpolation and physics-based approaches, where interpolation based approaches typically allow for larger deformations at the expense of increased computational cost.

Over the past two decades, interpolation-based techniques such as radial-basis-function (RBF) thin-plate-spline (TPS) deformation, and inverse-distance weighting IDW [20–23] have been widely used choices for deforming surface (and volume) meshes in response to large geometric changes. By interpolating prescribed boundary displacements of control points through a global kernel (e.g., Gaussian), these methods yield smooth deformations that preserve mesh orthogonality and enable rigid-body motions without remeshing. However, the dense linear solves scale as $O(K^3)$ in the number of boundary control points, increasing the computational cost for large number of control points. In addition, extreme deformations can lead to element inversion or excessive stretching unless supplemented by iterative quality improvement or remeshing.

In contrast to interpolation-based mesh deformation, physics-based approaches borrow different concepts from physics. Spring-analogy methods [24, 25] model the mesh as a system of linear springs. They are computationally efficient and widely used for moderate deformations but can struggle with preserving mesh quality under large transformations, requiring torsional extensions or load-stepping to avoid inversion. Laplacian (Winslow) methods [26–29] update mesh nodes by solving Laplace's equation, resulting in a sparse linear system. These methods are fast but are limited to relatively small design changes [30]. Lastly, elasticity-based methods [31, 32] treat the mesh as an elastic solid and move the mesh nodes by solving the continuum mechanics equations. They are robust for large deformations, but are less computationally efficient than Laplacian smoothing or spring analogy methods. We summarize existing mesh deformation techniques in Tab. 1. We note that while FFD can also be used to update surface and volume mesh nodes, we omit it here.

C. Aerodynamic Solvers

While the focus of this work is on an efficient surface mesh deformation method, we briefly discuss the aerodynamic solvers that this method is intended to enable. In the context of comprehensive large-scale MDO, vortex panel methods

Table 1 Comparison of surface mesh deformation methods for large-scale MDO.

Method	Scalability & Cost	Robustness	Mesh Quality	Differentiability
RBF / TPS / IDW	Dense solves $O(K^3)$ for K anchor points	Handles large geometric changes; may fold under extreme deformation	Naturally preserves mesh quality; may need post-processing for extreme stretches	Fully differentiable (linear system)
Spring Analogies	Sparse solves $O(N)$	Good for small–moderate motions; needs torsional terms or sub-stepping for large deflections	Can degrade unless torsional springs added; may need local remeshing	Piecewise differentiable; adjoint derivation non-trivial
Laplacian / Winslow	Sparse solves $O(N)$	Stable for low-frequency deformations; can invert near tight bends	Filters high-frequency detail; good global smoothness, locally poor in tight regions	Fully differentiable (linear system)
Elasticity (FEM)	$O(N^{1.5})$–$O(N^2)$	Physically consistent large deformations; resists inversion	Good volumetric control; preserves element shapes	Fully differentiable with PDE libraries such as FEniCS [33]

(see e.g., [6]) are considered mid-fidelity methods that allow for modeling blunt bodies and non-lifting surfaces. Panel methods are based on potential flow theory and assume the flow to be inviscid, irrotational, and generally incompressible, although compressibility effects can be modeled via a simple correction. In addition, boundary layer models can be integrated. Panel methods require conformal surface meshes that may be structured or unstructured. For this work we use a vortex panel code developed by Scotzniovsky and Hwang [34] to demonstrate the surface mesh deformation algorithm.

CFD solvers represent a significant improvement in fidelity compared to panel methods as they model compressibility effects as well as viscosity. For large-scale MDO, most solvers discretize the Euler or Reynolds-averaged Navier–Stokes equations. One of the most important aspects of a CFD solver is the domain discretization method. We identify four distinct methods that are frequently used in the analysis and design of aerospace vehicles. First, structured multiblock grids deliver high accuracy and solver efficiency but are sensitive to geometry changes, requiring manual block-layout rework [35]. Second, overset (Chimera) meshes overlap multiple structured blocks and interpolate data in the overlapping region. This greatly eases mesh generation for multi-component configurations [36]. Third, unstructured tetrahedral or polyhedral meshes are flexible in representing arbitrary shapes as they eliminate the need for manual block structuring. Fourth, cut-cell/immersed-boundary (IB) methods bypass conformal meshing entirely, embedding the geometry in a Cartesian grid. Cut-cell methods provide high flexibility but must address non-smooth cut-cell connectivity changes to preserve derivative continuity [37]. While some of these methods do not require a conformal surface mesh, we developed the present surface mesh deformation method to enable a geometry-centric MDO framework that can integrate panel codes as well as any type of CFD solver.

III. Surface Mesh Deformation Method

We now describe the surface mesh deformation method, which is composed of two steps. First, we compute the intersection curves between component manifolds. Second, we use thin plate splines (TPS) to deform the surface mesh, treating nodes that lie on intersection curves as control (i.e., anchor) points in the TPS fitting step. As discussed in Sec. II, TPS is an established mesh deformation method. We therefore only provide the basic equations and provide more detail on the intersection computation.

A. Computation of Component Intersections

Our starting point is a set of intersecting, topologically two-dimensional closed manifolds in a three-dimensional space. Component manifolds are sets of parametric surfaces that belong to arbitrary function spaces. In CAD geometries, a popular choice is B-spline surfaces. Given two intersecting manifolds M_1 and M_2, for each point $P \in M_1 \cap M_2$, there exists $i_1 \in \mathcal{T}_1,\ i_2 \in \mathcal{T}_2,\ (u_1, v_1), (u_2, v_2) \in [0, 1]^2$ such that $P = E_{M_1}(i_1, u_1, v_1) = E_{M_2}(i_2, u_2, v_2)$, where $\mathcal{T}_1$ and $\mathcal{T}_2$ denote the sets of parametric surfaces in M_1 and M_2. A surface mesh is a discretization of the outer hull of the design geometry, which changes during design optimization. In order to update the surface mesh accordingly, we need to recompute component intersections.

In this work, we assume that the topology of the design geometry does not change between optimization iterations; i.e., components do not form new intersections. There are several requirements for the intersection computation.

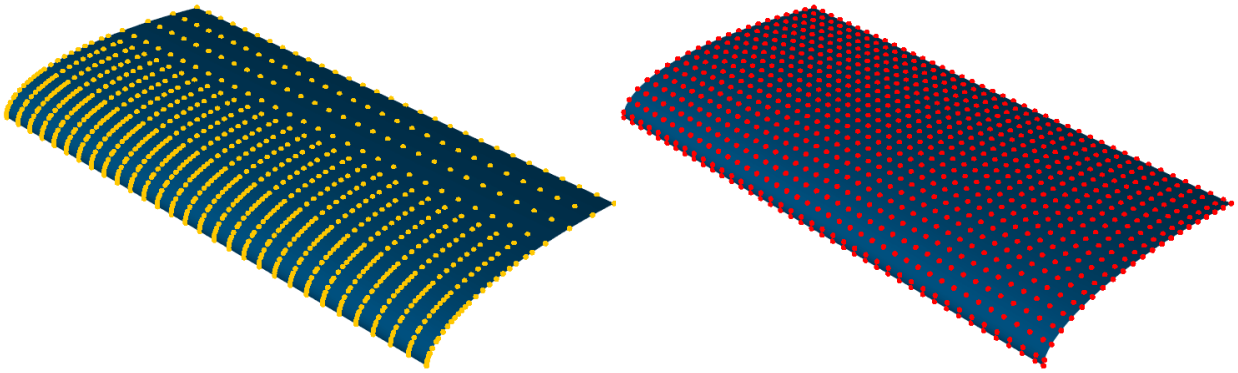
1) The intersection must be allowed to cross boundaries of parametric surfaces within a component manifold.
2) The intersection computation must be differentiable.
3) The intersection computation must be robust and reasonably efficient.

1. Generation of Point Cloud Near Intersection

Given two intersecting manifolds M_1 and M_2, the first step is to identify the pairs of parametric surfaces that share an intersection. An axis-aligned bounding-box overlap test quickly discards any pair of parametric surfaces that cannot intersect. For each pair of remaining surface candidates, we evaluate a dense parametric grid, producing point clouds $\mathcal{P}_1 \subset \mathcal{R}^3$ and $\mathcal{P}_2 \subset \mathcal{R}^3$. An important consideration is that, depending on the function space, a uniformly sampled grid of parameter values may not lead to a uniform grid of evaluated points. This is shown in Fig.1a for evaluating a uniformly sampled grid over a B-spline surface. The resulting point cloud is only uniform in one direction and highly skewed in the other direction, which is undesirable for identifying points that lie close to the intersection. To remedy this, we adaptively sample the parametric space once in the u direction for a fixed value of v and once in the v direction for a fixed value of u. We march independently along each parametric direction using a one-dimensional root-finding procedure. Starting from $u = 0$ (or $v = 0$), we solve for the next parameter value u^{k+1} such that

$$\left\| S(u^{k+1}, v_0) \; - \; S(u^k, v_0) \right\| \; = \; \Delta \tag{1}$$

via a bracketed Brent–q solver [38] , where $S(u^{k+1}, v_0)$ is the evaluation of a parametric surface at parametric coordinates u^{k+1}, v_0 and Δ is the desired spacing. To robustly identify the next parameter value that yields the target physical spacing, we begin with a small parametric step and iteratively double it until the physical distance exceeds the desired threshold or the end of the domain is reached, effectively identifying a bracket. This exponential bracketing strategy ensures that the method adapts to the local surface curvature, taking small parametric steps in flat regions and larger steps in tightly curved regions. As a result, it avoids expensive global searches while maintaining reliable convergence. Finally, the tensor-product grid $\{(u_i, v_j)\}$ formed by marching once in u and once in v defines the parametric values whose evaluation leads to a largely uniform grid of points, as shown in Fig. 1b. Since the final marching step may not exactly align with the target spacing Δ, the second-to-last row is optionally discarded if the distance between the last and second-to-last rows of points is too small. In addition, for complex geometries with largely varying dimensions (see e.g., Fig. 2), the evaluated tensor-product grid may not be globally uniform, with denser spacing along shorter edges. After



(a) Evaluation of a uniform parametric grid **(b) Evaluation of an adaptively sampled parametric grid**

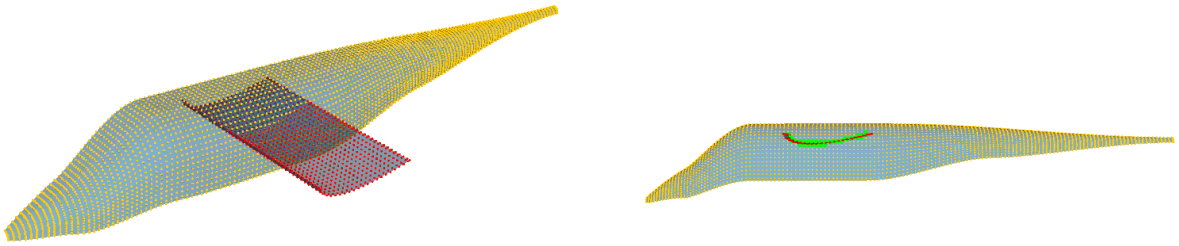
Fig. 1 Evaluation of parametric grids for a B-spline surface of a wing component manifold.

generating two (largely) uniform point clouds $\mathcal{P}_1$ and $\mathcal{P}_2$ for the pair of intersecting parametric surfaces, the goal is to efficiently identify mutual nearest neighbors between the surfaces whose distances are below a certain threshold ϵ. To avoid a brute-force search, we make use of k-dimensional (k-d) trees[39], which are spatial portioning structures that reduce the cost of the nearest neighbor search from $O(n^2)$ to $O(\log(n))$ on average, or $O(n)$ in the worst case. We build k-d trees $\mathcal{K}_1$ and $\mathcal{K}_2$ for $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively. For each point $p \in \mathcal{P}_1$, we query $\mathcal{K}_2$ to find its closest point in $\mathcal{P}_2$.

Conversely, for each $q \in \mathcal{P}_2$, we query $\mathcal{K}_1$. This allows us to form the set of mutual nearest-neighbor pairs, $\mathcal{P}'$, given by

$$\mathcal{P}' = \{(p, q) \ : \ q = \arg\min_{q' \in \mathcal{P}_2} \|p - q'\|, \ p = \arg\min_{p' \in \mathcal{P}_1} \|q - p'\|\}.$$

From the mutual nearest-neighbor set, we retain only those pairs whose Euclidean distance is below ϵ. We use the corresponding parametric coordinates as initial guesses for the optimization problem introduced in the next section, whose solution corresponds to points that lie exactly on the intersection. In Fig. 2, we provide a notional example of how point clouds from two intersecting parametric surfaces (belonging to a wing and fuselage manifold) are used to find points in the neighborhood of the true intersection.



(a) Isometric view of point clouds over two intersecting parametric surfaces (fuselage and wing).

(b) Side view of the intersecting surfaces with points near the intersection highlighted in green.

Fig. 2 Notional example of generating point clouds for intersecting parametric surfaces to identify points (green) that lie in the neighborhood of the true intersection.

2. Computation of Intersection Points

In addition, $\mathcal{T}_1' \subset \mathcal{T}_1$ and $\mathcal{T}_2' \subset \mathcal{T}_2$ denote the sets of parametric surfaces that share an intersection and $i_1' \in \mathcal{T}_1'$, $i_2' \in \mathcal{T}_2'$ are the corresponding indices. Then, for each point $x_0 \in \mathcal{P}'$, we solve the following optimization problem

$$\begin{aligned} \underset{(u_1, v_1, u_2, v_2)}{\text{minimize}} \quad & \frac{1}{2}\left\| E_{M_1}(i_1', u_1, v_1) - x_0 \right\|^2 \\ \text{subject to} \quad & E_{M_1}(i_1', u_1, v_1) = E_{M_2}(i_2', u_2, v_2) \end{aligned} \tag{2}$$

The solution to this optimization problem is the parametric coordinates, whose evaluation results in a point that lies exactly (to machine precision) on the intersection curve and that is closest to x_0. Since there are an infinite number of points that lie on the intersection curve, we find the ones that are closest to the points in $\mathcal{P}'$; that is, the previously determined points that lie close to the intersection. This ensures a unique solution to Prob. (2). We note that in the objective, replacing $E_{M_1}(i_1', u_1, v_1)$ with $E_{M_2}(i_2', u_2, v_2)$ yields the same result. Solving Prob. (2) for each point $x_0 \in \mathcal{P}'$ generates a new set of points $\mathcal{P}^*$ that lie on the intersection curve. Based on these points, the next step is to find a functional representation of the intersection manifold, described in the next section.

3. Computation of Intersection Manifold

To find a functional representation of the intersection manifold, we must first sort the intersection points $\mathcal{P}^*$, as they may be in random order. This ordering problem is an instance of the familiar Traveling Salesman Problem (TSP), which is NP-hard. A brute force search over all permutations is guaranteed to find the shortest path but is intractable for larger point clouds as its scaling is $O(n!)$ for n points. Therefore, we use heuristics, combining a greedy nearest-neighbor walk with the 2-opt algorithm [40]. A greedy nearest-neighbor walk incrementally constructs an ordered sequence of points given a starting point (e.g., leading edge) and repeatedly appends the nearest unvisited point. A potential issue is that for intersections with sharp corners or turns like the trailing edge of an airfoil, a greedy algorithm may produce a path with crossed segments (i.e., detours) that lead a globally suboptimal path. The 2-opt heuristic addresses this issue by iteratively swapping pairs of edges to reduce the total length. Specifically, the algorithm examines all pairs of non-adjacent edges (a–b) and (c–d) and reverses the sub-path between b and c if reconnecting the edges as (a–c) and (b–d) shortens to the total path length. In practice, provided with the result of the nearest-neighbor walk as a starting path, the 2-opt heuristic typically converges within ten iterations.

Given an ordered set of intersection points, an important consideration is that between optimization iterations, the ordering may be reversed. For example, points on a wing-fuselage intersection may be ordered clockwise in one iteration, and counter-clockwise in the next one. To ensure consistent ordering, we compute the signed area vector of the ordered intersection points and compare its direction to a reference normal. For point cloud $\mathcal{P}^*$, the signed area vector is computed from the sum of cross products of adjacent points, i.e., $\boldsymbol{a} = \frac{1}{2}\sum_i^{N-1} p_i \times p_{i+1}$. The reference normal $\boldsymbol{n}$ is found by applying principal component analysis (PCA) to $\mathcal{P}^*$ to estimate the plane in which the intersection curve lies. Based on the sign of $\boldsymbol{a} \cdot \boldsymbol{n}$, the ordering is reversed to match the initial ordering at the zero-th optimization iteration. Finally, a least-squares fit determines the coefficients $\boldsymbol{c}$ of the intersection manifold. Regularization may be added for improved numerical stability. This approach works for any linear function space, whose basis functions allow for the construction of a basis matrix. The result is a one-dimensional manifold M_{int}, whose evaluation map $E_{M_{int}}(i,t) = \boldsymbol{B}_i(t)\boldsymbol{c}_i$ is the matrix-vector product of the basis matrix $\mathbf{B}$ and coefficients $\mathbf{c}$. Here, the index i denotes the i-th parametric curve in the intersection manifold. For example, the intersection manifold for a wing-fuselage intersection may be represented as two B-spline curves, one for the upper and one for the lower surface. In Fig. 3, we provide a notional example of the ordering procedure of the intersection points and the fitting of the intersection manifold. Once all intersection curves are found, we project surface mesh nodes that lie on the intersection (at the zero-th iteration only) to determine the corresponding parametric coordinates. At subsequent optimization iterations, intersection mesh nodes are then simply re-evaluated after the intersection curves are updated.

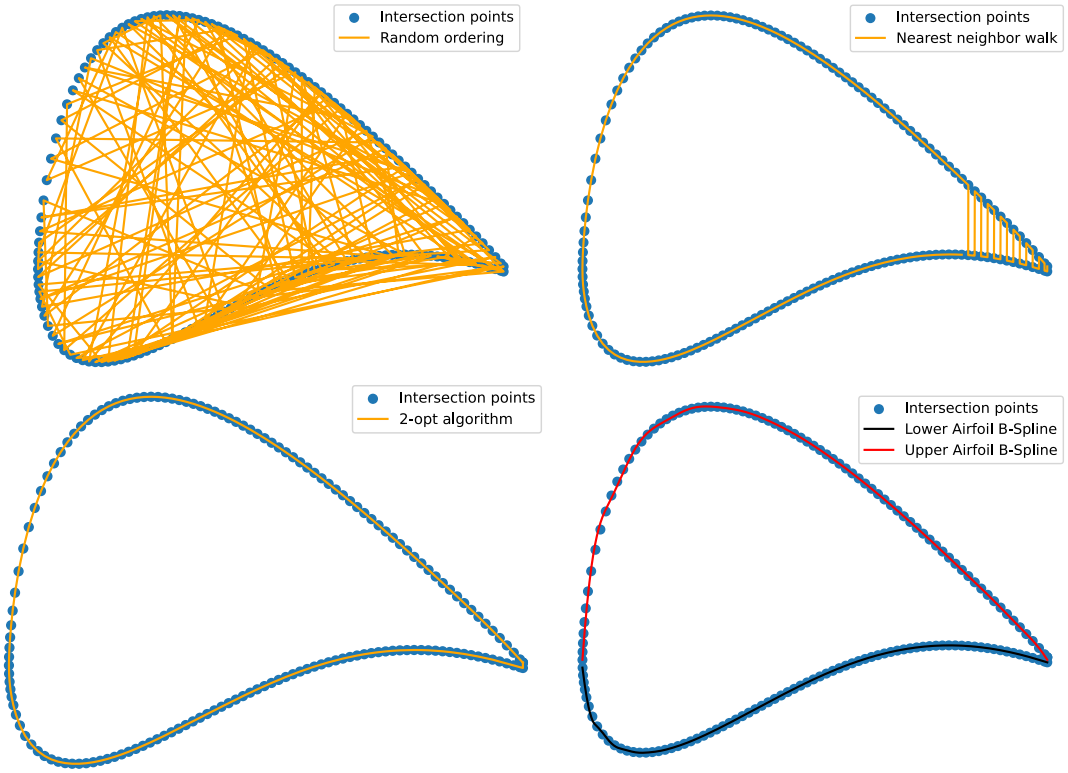


Fig. 3 Illustration of ordering process of intersection points from which the intersection manifold is fitted (two B-spline curves in this example).

B. Surface Mesh Deformation

In this work, we use thin plate splines (TPS) to deform the surface mesh. As stated in Sec. II, TPS methods have been successfully applied to mesh deformation. We briefly summarize the method's background and main steps. TPS was first introduced by Duchon [41] in 1976 as the unique interpolant in $\mathbb{R}^2$ that minimizes the bending-energy of thin plates, and was popularized by Bookstein [42] in 1989 for the analysis and warping of biological shapes and anatomical images. TPS defines a smooth, global deformation that interpolates between sets of control points (also known as

"anchor" or "landmark" points) while minimizing the bending energy functional $E[f]$ of a thin plate, given (in 3D) by

$$E[f] \;=\; \int_{\mathbb{R}^3} \|\|\nabla^2 f(x,y,z)\|\|_F^2 \, dV,$$

subject to interpolation constraints

$$f(\mathbf{x}_i) \;=\; \mathbf{y}_i, \quad i = 1, \ldots, N_{cp},$$

where $\mathbf{y}_i$ denotes the i-th control (anchor) point. We note that the bending energy functional in 3D is a direct generalization of the 2D case. The function f that minimizes the bending energy functional can be found using variational calculus and its general form is given as

$$f(\mathbf{x}) = \mathbf{A}\,\mathbf{x} + \mathbf{b} \;+\; \sum_{i=1}^{N} \mathbf{w}_i \, U\big(\|\mathbf{x} - \mathbf{x}_i\|\big), \tag{3}$$

where $\mathbf{A} \in \mathbb{R}^{3\times3}$ is an affine transform matrix, $\mathbf{b} \in \mathbb{R}^3$ is a translation vector, $\mathbf{w}_i \in \mathbb{R}^3$ are the TPS weights and $U(\cdot)$ is the radial basis kernel (e.g., Gaussian). The unknown quantities are determined by solving the following linear system of equations.

$$\begin{bmatrix} \mathbf{K} + \lambda \mathbf{I} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{W} \\ \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{Y} \\ \mathbf{0} \end{bmatrix}, \tag{4}$$

where

$$K_{ij} = U\big(\|\mathbf{x}_i - \mathbf{x}_j\|\big), \quad P = \begin{bmatrix} \mathbf{1} & X \end{bmatrix}, \quad C = \begin{bmatrix} \mathbf{b} \\ \mathrm{vec}(\mathbf{A}) \end{bmatrix},$$

and $\lambda \geq 0$ is a regularization parameter. For N_{cp} control points, the size of the linear system is $(N_{cp}+d+1)\times(N_{cp}+d+1)$, where $d = 3$. After solving the linear system in Eq. 4, the surface mesh is deformed using Eq.3. Since the intersection points act as anchor points, the mesh is naturally deformed toward the computed intersections. Depending on the magnitude of the deformations, some mesh nodes may be moved off the design geometry. To remedy this problem, we 1) define an influence radius beyond which the deformation smoothly decays and 2) re-project the surface mesh nodes onto the design geometry. The influence radius is determined as a multiple of the largest distance between points in point cloud $\mathcal{P}^*$ (i.e., the intersection points).

C. Derivative Computation

We implement the mesh deformation method in a recently developed graph-based modeling language (CSDL) [1] that automates sensitivity analysis. This means that n-th order gradient information for most generic mathematical operations is automatically available, including solving linear systems of equations like Eq. 4. CSDL also has built-in support for nonlinear residual equations via a nonlinear block Gauss–Seidel or a Newton algorithm and leverages the implicit function theorem for fast reverse-mode differentiation using vector-Jacobian products. This allows us to efficiently compute the post-optimality sensitivities of optimization problems such as Prob. 2 (computation of intersection points) or the point-to-surface projection problem with minimal implementation effort. While the demonstration of the proposed mesh deformation algorithm in an optimization setting is outside the scope of this paper, it is a natural extension, and we anticipate first aerodynamic shape optimization results in the near future.

IV. Results

In this section, we present the results of applying the surface mesh deformation method to a generic tube-and-wing reference geometry, created in OpenVSP [15], which we also use to generate a surface mesh consisting of 9116 triangles. The quality and resolution of the (initial) surface is not the main focus and for practical applications, a more refined mesh is necessary, which may require special meshing software such as Gmsh [43]. Using the previously mentioned panel code, we evaluate the flow solution for the deformed meshes.

A. Mesh Deformation

The first deformation test case is a wing translation along the length of the fuselage. The maximum translation is two times the mean aerodynamic chord (MAC) length in the forward and backward direction. Figure 4 shows the results compared to the initial wing geometry. We see that the method smoothly deforms the surface mesh, preserving high mesh quality for most elements. However, for the extreme translation cases, the mesh elements surrounding the intersection are significantly compressed on one side and stretched on the opposite side, which is to be expected. In Fig. 6, we show the quality of the mesh elements for each translation case. The mesh quality is worst for the extreme backward translation (-2 MAC), with only 76% of triangles having minimum and maximum angles bounded by 30 and 120 degrees, respectively, defined as the acceptability threshold. However, for the ±1 MAC cases (which already constitute significant geometric changes), the mesh quality is within 1% of the initial mesh, with over 97% of triangles falling within the acceptable region.

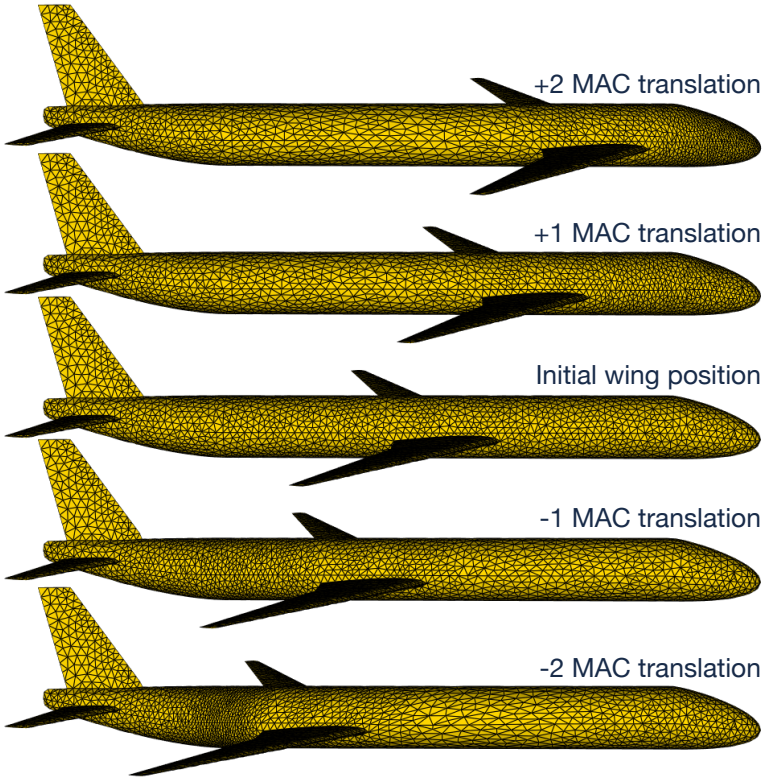


Fig. 4 Mesh deformation results for translating the wing along the fuselage. The translation is expressed in terms of mean aerodynamic chord (MAC) lengths.

For the second deformation test case, we rotate the wing by ±10° about the y axis, effectively changing the wing incidence angle. Figure 5 shows the corresponding mesh deformation. We see that the method smoothly deforms the mesh around the intersection with minimal loss of element quality across all rotation angles. In the the worst case (+10° rotation), 94.5% of all mesh elements fall within the acceptable region of minimum and maximum angles. This is a good result and highlights the ability of the surface mesh deformation method to deal with large geometric changes. However, it is important to point out that while these results are promising, a single badly skewed element or a single pair of overlapping elements can cause an aerodynamic solver to fail. In some cases, large deformations may require load-stepping, a process in which the total deformation is divided into smaller load steps.

B. Aerodynamic Results

In this section, we present the aerodynamic results obtained with the panel code. The purpose is to demonstrate the ability of the mesh deformation method to produce meshes of high quality, even for large deformations, and to verify expected trends that match with engineering intuition. As such, we use the deformed meshes of the previous section as

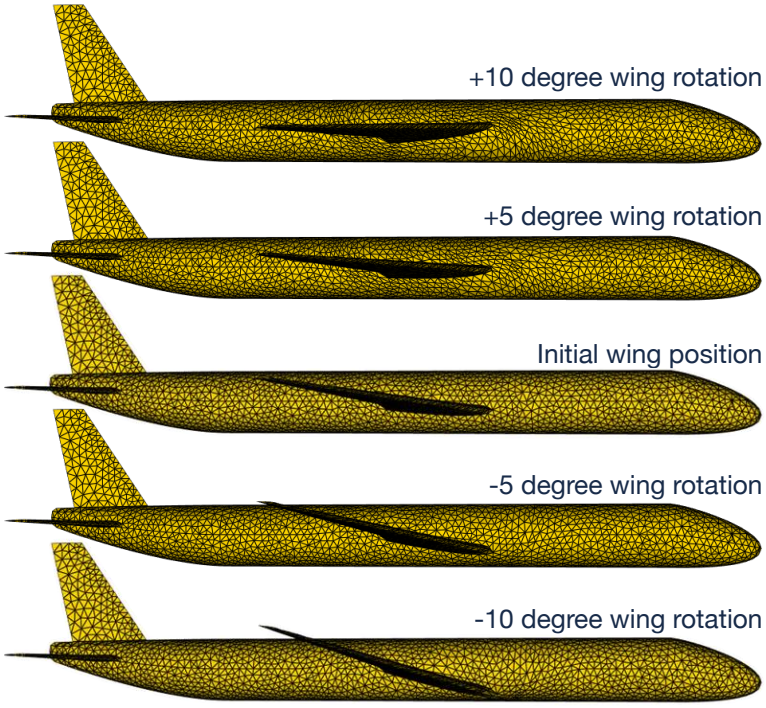


Fig. 5 Mesh deformation results for rotating the wing about the y-axis.

inputs to the panel code. The free stream Mach number is set to 0.4 at atmospheric conditions. Figure 7 shows the pressure and doublet-strength contours (top-down view) over the aircraft for varying wing rotation angles as well as the stream lines around the body (isometric view). The results match with engineering intuition, showing a decrease in pressure on the section side with positive rotation angles and the opposite trend for negative rotations. Another indicator that confirms the correctness of the trends are the wakes. For positive rotation angles, the wake rolls up as expected, and for negative rotation angles, the wake rolls down.

Lastly, we investigate the effect of the wing rotation and translation on the aircraft's total lift coefficient. The results are shown in Fig. 8. For the wing rotation, we observe a largely linear lift polar, which is expected. We note that although the wing is cambered, the mesh resolution is likely insufficient near the leading edge. Again, the emphasis is not on the correctness of the flow solution but rather to confirm that the mesh deformation method works properly. In Fig. 8b, we show the aircraft's lift coefficient as a function of the wing position (i.e., translation). The lift coefficient decreases as the wing is moved from the back toward the nose. This matches with engineering intuition since for forward positions, the wing's wake reaching the tail is more developed, increasing the downwash at the tail, which decreases total lift.

V. Conclusion

We present a method for efficient surface mesh deformation given large shape changes. This method is intended for comprehensive, system-level large-scale multidisciplinary design optimization (MDO) of aircraft concepts using medium to high-fidelity aerodynamic solvers such as panel codes and CFD. For conceptual design, the design geometry is expected to change significantly, making it challenging to maintain a high-quality conforming surface mesh. While not all CFD solvers use discretizations techniques that require a conforming surface mesh (e.g., overset), many popular (unstructured) solvers (e.g., SU2) that are frequently used in MDO and most panel codes do require a conforming surface mesh. In addition, for multi-physics problems (e.g., aero-elasticity) conforming meshes significantly simplify data-transfer accuracy and conservation enforcement. In the context of comprehensive, system-level large-scale MDO, an effective surface mesh deformation approach should 1) efficiently compute component intersections, 2) update surface mesh nodes accordingly while preserving mesh quality, 3) permit large shape changes, and 4) be differentiable. Current approaches do not simultaneously meet all four criteria.

The method proposed in this paper aims to address all four requirements. It first computes component intersections

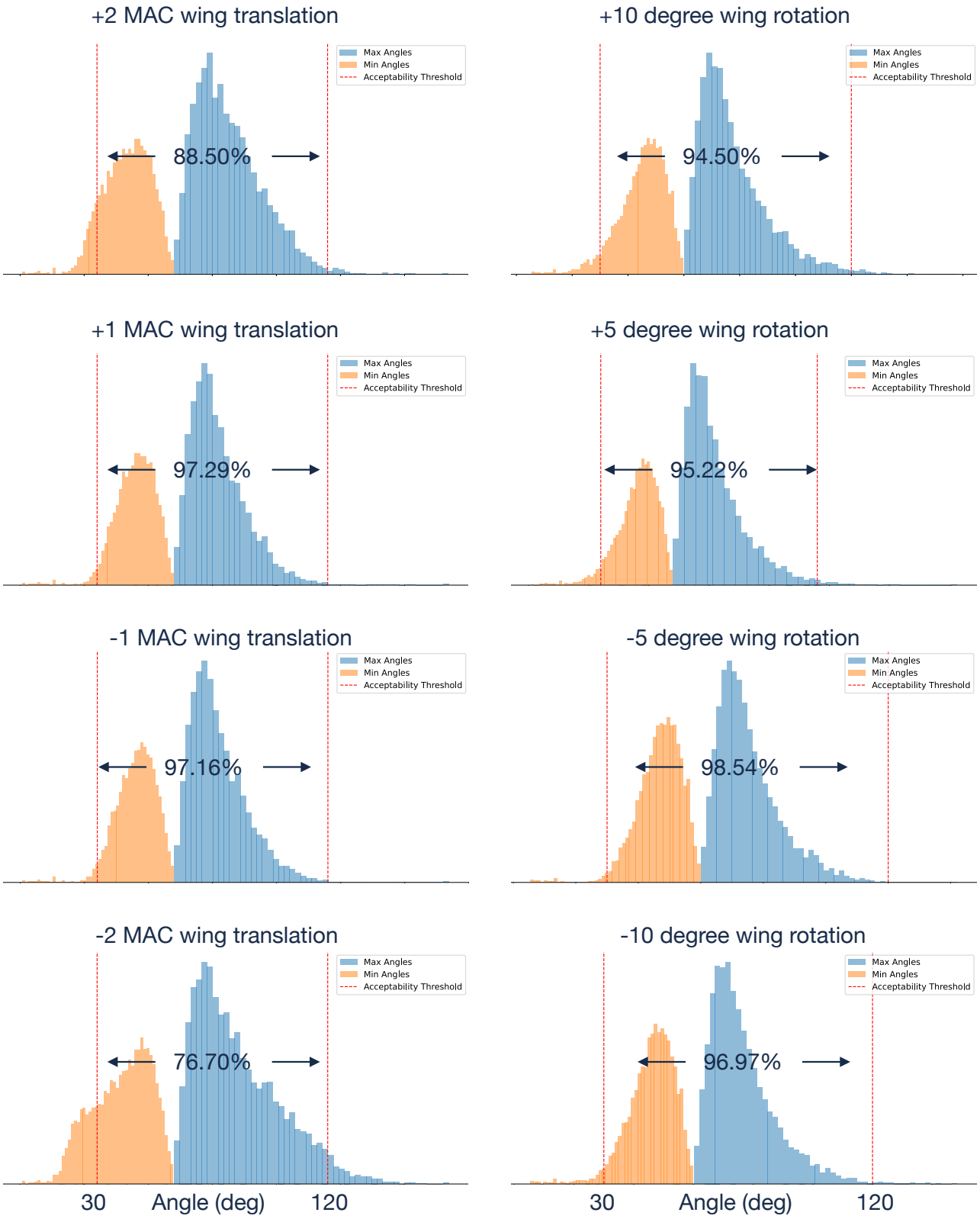


Fig. 6 Triangle quality (angle-based) histograms for wing rotations and translations.

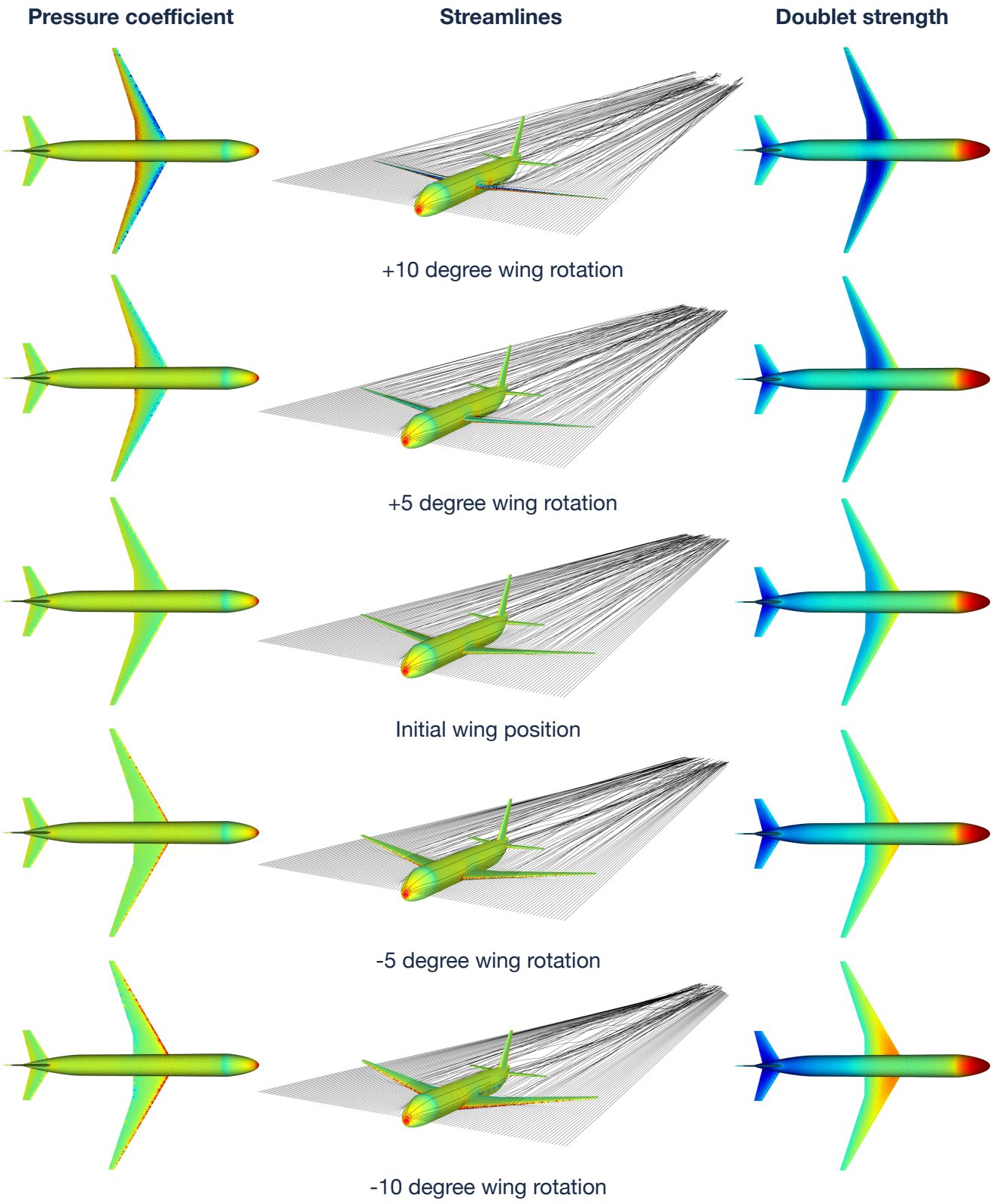
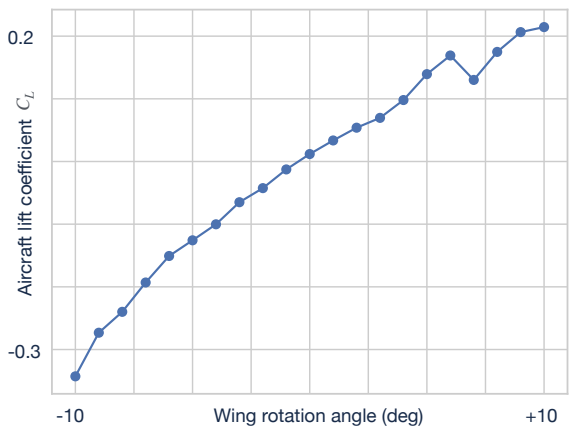
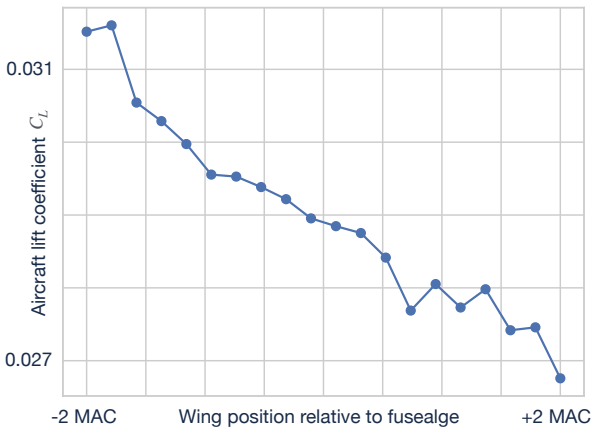


Fig. 7 Pressure contours, streamlines, and doublet strength contours for different wing rotation angles.



(a) Aircraft lift coefficient vs. wing rotation angle.



(b) Aircraft lift coefficient vs. wing position in terms of mean aerodynamic chord (MAC) lengths.

Fig. 8 Aircraft lift coefficient vs. wing rotation and wing position.

via a robust sequence of steps that produce point clouds that lie on the intersection curves. These point clouds are generated by adaptively sampling intersecting parametric surfaces and solving an optimization problem. A functional representation (e.g., B-spline curves) is then used as an approximation of the true intersection curves. The process of computing intersection curves works for arbitrary shape changes, provided that the topology of the geometry remains unchanged. Next, the method uses thin plate splines (TPS) to deform the surface mesh. The intersection points serve as control (anchor) points in the TPS fitting step. Paired with a projection step, these anchor points ensure that the surface mesh conforms to the (updated) design geometry. The proposed method is reasonably fast (on the order of seconds) and is made differentiable by implementing it in a newly developed graph-based modeling language that automates adjoint-based sensitivity analysis and leverages efficient reverse-mode differentiation using vector-Jacobian products.

We demonstrate the method on a generic tube-and-wing test problem in which we translate and rotate the wing with respect to the fuselage. The method performs reasonably well for large deformations with over 95% of triangles being of acceptable quality when rotating the wing by $\pm 10^\circ$. For translations the method performs comparably up to 1.5 times the mean aerodynamic chord (MAC). For each deformation test case, we use the deformed mesh as an input to a panel code to verify that the computed flow solutions match with engineering intuition. The aerodynamic results are reasonable, showing a linear lift polar when changing the rotation (i.e., incidence) angle of the wing. This trend is confirmed by the pressure and doublet strength contours. Translating the wing forward leads to a decrease in total lift due to the influence of the more developed wake on the tail.

These results show the potential of the proposed method to enable the integration of medium to high-fidelity aerodynamic solvers in comprehensive system-level large-scale MDO of aircraft concepts. However, to achieve this goal, more work is necessary in the following areas.

- *Computational efficiency*: Considering more component intersections and using finer surface meshes will increase computational cost. To improve efficiency, the current method can be enhanced by exploiting two embarrassingly parallel subproblems. First, the computation of the intersections involves solving an optimization problem for each intersection point, which is currently solved sequentially. Second, point-to-surface projections should also be parallelized, leading to significant speed ups for dense surface meshes.
- *Surface mesh quality*: While the TPS approach naturally deforms the surface in a way that preserves mesh quality, this characteristic diminishes for large deformations. A solution is to include mesh quality metrics (e.g., edge length loss) in the TPS fitting step. This leads to a nonlinear least-squares problem, which can still be solved efficiently and robustly using optimization techniques like the Levenberg–Marquardt algorithm.
- *Complex geometries*: The results of this paper are based on a simple tube-and-wing geometry. However, conceptual design geometries may be much more complex and the proposed method should work for any reasonable air-and rotorcraft configuration.

VI. Funding Sources

The material presented in this paper is supported, in part, by NASA under award No. 80NSSC21M0070.

VII. Acknowledgments

The authors would like to thank Luca Scotzniovsky for his help with generating the aerodynamic results in this paper. The authors acknowledge the use of generative AI (ChatGPT) in proofreading this manuscript.

Appendix

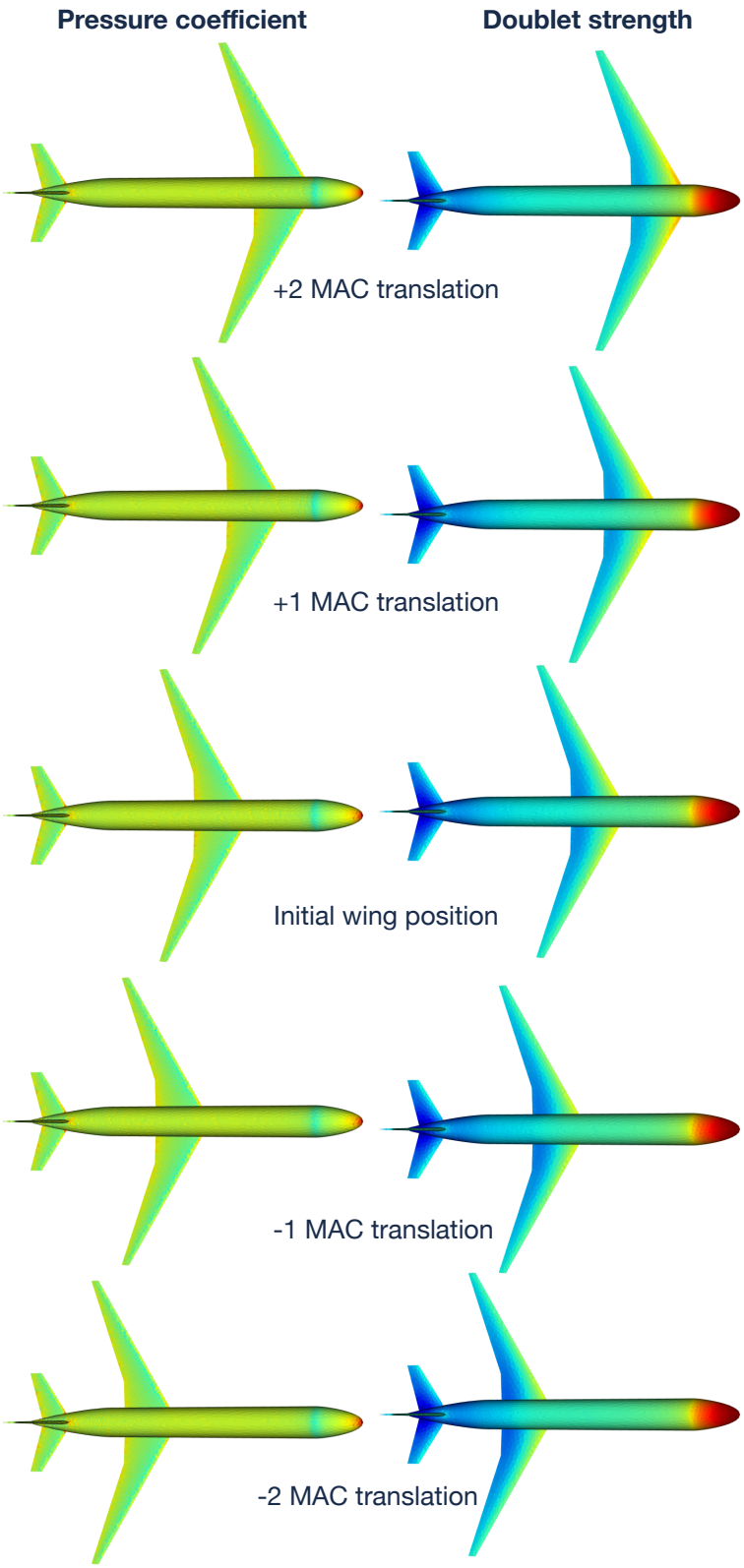


Fig. 9 Aerodynamic solutions for translating the wing.

References

[1] Gandarillas, V., Joshy, A. J., Sperry, M. Z., Ivanov, A. K., and Hwang, J. T., "A graph-based methodology for constructing computational models that automates adjoint-based sensitivity analysis," *Structural and Multidisciplinary Optimization*, Vol. 67, No. 5, 2024, p. 76.

[2] Sarojini, D., Ruh, M. L., Joshy, A. J., Yan, J., Ivanov, A. K., Scotzniovsky, L., Fletcher, A. H., Orndorff, N. C., Sperry, M., Gandarillas, V. E., et al., "Large-scale multidisciplinary design optimization of an evtol aircraft using comprehensive analysis," *AIAA SciTech 2023 Forum*, 2023, p. 0146.

[3] Ruh, M. L., Sarojini, D., Fletcher, A., Asher, I., and Hwang, J. T., "Large-scale multidisciplinary design optimization of the NASA lift-plus-cruise concept using a novel aircraft design framework," *arXiv preprint arXiv:2304.14889*, 2023.

[4] Ruh, M. L., Fletcher, A., Sarojini, D., Sperry, M., Yan, J., Scotzniovsky, L., van Schie, S. P., Warner, M., Orndorff, N. C., Xiang, R., et al., "Large-scale multidisciplinary design optimization of a NASA air taxi concept using a comprehensive physics-based system model," *AIAA SCITECH 2024 Forum*, 2024, p. 0771.

[5] Drela, M., "XFOIL: An analysis and design system for low Reynolds number airfoils," *Low Reynolds Number Aerodynamics: Proceedings of the Conference Notre Dame, Indiana, USA, 5–7 June 1989*, Springer, 1989, pp. 1–12.

[6] Katz, J., *Low-speed aerodynamics*, Cambridge University Press, 2001.

[7] Mader, C. A., Kenway, G. K., Yildirim, A., and Martins, J. R., "ADflow: An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization," *Journal of Aerospace Information Systems*, Vol. 17, No. 9, 2020, pp. 508–527.

[8] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An open-source suite for multiphysics simulation and design," *Aiaa Journal*, Vol. 54, No. 3, 2016, pp. 828–846.

[9] Burdette, D. A., and Martins, J. R., "Design of a transonic wing with an adaptive morphing trailing edge via aerostructural optimization," *Aerospace Science and Technology*, Vol. 81, 2018, pp. 192–203.

[10] Adler, E. J., and Martins, J. R. R. A., "Efficient Aerostructural Wing Optimization Considering Mission Analysis," *Journal of Aircraft*, Vol. 60, No. 3, 2023. https://doi.org/10.2514/1.c037096.

[11] Gray, A. C., Riso, C., Jonsson, E., Martins, J. R. R. A., and Cesnik, C. E. S., "High-fidelity Aerostructural Optimization with a Geometrically Nonlinear Flutter Constraint," *AIAA Journal*, Vol. 61, No. 6, 2023, pp. 2430–2443. https://doi.org/10.2514/1.J062127.

[12] Jonsson, E., Riso, C., Monteiro, B. B., Gray, A. C., Martins, J. R. R. A., and Cesnik, C. E. S., "High-Fidelity Gradient-Based Wing Structural Optimization Including Geometrically Nonlinear Flutter Constraint," *AIAA Journal*, Vol. 61, No. 7, 2023, pp. 3045–3061. https://doi.org/10.2514/1.J061575.

[13] Kenway, G., Kennedy, G., and Martins, J. R., "Aerostructural optimization of the Common Research Model configuration," *15th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2014, p. 3274.

[14] Brooks, T. R., Martins, J. R., and Kennedy, G. J., "High-fidelity aerostructural optimization of tow-steered composite wings," *Journal of Fluids and Structures*, Vol. 88, 2019, pp. 122–147.

[15] Hahn, A., "Vehicle sketch pad: a parametric geometry modeler for conceptual aircraft design," *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2010, p. 657.

[16] Haimes, R., and Dannenhoffer, J., "The engineering sketch pad: A solid-modeling, feature-based, web-enabled system for building parametric geometry," *21st AIAA Computational Fluid Dynamics Conference*, 2013, p. 3073.

[17] Sederberg, T. W., and Parry, S. R., "Free-form deformation of solid geometric models," *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, 1986, pp. 151–160.

[18] Samareh, J., "Aerodynamic shape optimization based on free-form deformation," *10th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2004, p. 4630.

[19] Hicken, J. E., and Zingg, D. W., "Aerodynamic optimization algorithm with integrated geometry parameterization and mesh movement," *AIAA journal*, Vol. 48, No. 2, 2010, pp. 400–413.

[20] Rendall, T. C., and Allen, C. B., "Efficient mesh motion using radial basis functions with data reduction algorithms," *Journal of Computational Physics*, Vol. 228, No. 17, 2009, pp. 6231–6249.

[21] Coulier, P., and Darve, E., "Efficient mesh deformation based on radial basis function interpolation by means of the inverse fast multipole method," *Computer methods in applied mechanics and engineering*, Vol. 308, 2016, pp. 286–309.

[22] De Boer, A., Van der Schoot, M. S., and Bijl, H., "Mesh deformation based on radial basis function interpolation," *Computers & structures*, Vol. 85, No. 11-14, 2007, pp. 784–795.

[23] Secco, N. R., Kenway, G. K., He, P., Mader, C., and Martins, J. R., "Efficient mesh generation and deformation for aerodynamic shape optimization," *AIAA Journal*, Vol. 59, No. 4, 2021, pp. 1151–1168.

[24] Zeng, D., and Ethier, C. R., "A semi-torsional spring analogy model for updating unstructured meshes in 3D moving domains," *Finite Elements in Analysis and Design*, Vol. 41, No. 11-12, 2005, pp. 1118–1139.

[25] Blom, F. J., "Considerations on the spring analogy," *International journal for numerical methods in fluids*, Vol. 32, No. 6, 2000, pp. 647–668.

[26] Winslow, A. M., "Numerical solution of the quasilinear Poisson equation in a nonuniform triangle mesh," *Journal of computational physics*, Vol. 1, No. 2, 1966, pp. 149–172.

[27] Hwang, J. T., and Martins, J. R., "An unstructured quadrilateral mesh generation algorithm for aircraft structures," *Aerospace Science and Technology*, Vol. 59, 2016, pp. 172–182.

[28] Karman Jr, S. L., Anderson, W. K., and Sahasrabudhe, M., "Mesh generation using unstructured computational meshes and elliptic partial differential equation smoothing," *AIAA journal*, Vol. 44, No. 6, 2006, pp. 1277–1286.

[29] Field, D. A., "Laplacian smoothing and Delaunay triangulations," *Communications in applied numerical methods*, Vol. 4, No. 6, 1988, pp. 709–712.

[30] Selim, M. M., and Koomullil, R., "Mesh deformation approaches–a survey," *Journal of Physical Mathematics*, 2016. https://doi.org/10.4172/2090-0902.1000181.

[31] Stein, K., Tezduyar, T. E., and Benney, R., "Automatic mesh update with the solid-extension mesh moving technique," *Computer Methods in Applied Mechanics and Engineering*, Vol. 193, No. 21-22, 2004, pp. 2019–2032.

[32] Tonon, P., Sanches, R. A. K., Takizawa, K., and Tezduyar, T. E., "A linear-elasticity-based mesh moving method with no cycle-to-cycle accumulated distortion," *Computational Mechanics*, Vol. 67, 2021, pp. 413–434.

[33] Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N., "The FEniCS project version 1.5," *Archive of numerical software*, Vol. 3, No. 100, 2015.

[34] Scotzniovsky, L., and Hwang, J. T., "A fast, memory-efficient panel method for large-scale multidisciplinary design optimization under uncertainty using graph-based modeling," *AIAA Aviation 2025 forum*, 2025.

[35] Thompson, J. F., "A survey of dynamically-adaptive grids in the numerical solution of partial differential equations," *Applied Numerical Mathematics*, Vol. 1, No. 1, 1985, pp. 3–27.

[36] Kenway, G. K., Mishra, A., Secco, N. R., Duraisamy, K., and Martins, J. R., "An efficient parallel overset method for aerodynamic shape optimization," *58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2017, p. 0357.

[37] Mittal, R., Dong, H., Bozkurttas, M., Najjar, F., Vargas, A., and Von Loebbecke, A., "A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries," *Journal of computational physics*, Vol. 227, No. 10, 2008, pp. 4825–4852.

[38] Brent, R. P., "Algorithmsfor minimization without derivatives," *Prentiee-Hall, Englewood Cliffs, New Jersey*, 1973.

[39] Bentley, J. L., "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, Vol. 18, No. 9, 1975, pp. 509–517.

[40] Croes, G. A., "A method for solving traveling-salesman problems," *Operations research*, Vol. 6, No. 6, 1958, pp. 791–812.

[41] Duchon, J., "Splines minimizing rotation-invariant semi-norms in Sobolev spaces," *Constructive theory of functions of several variables: Proceedings of a conference held at Oberwolfach April 25–May 1, 1976*, Springer, 1977, pp. 85–100.

[42] Bookstein, F. L., "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 11, No. 6, 1989, pp. 567–585.

[43] Geuzaine, C., and Remacle, J.-F., "Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities," *International journal for numerical methods in engineering*, Vol. 79, No. 11, 2009, pp. 1309–1331.