Toward Large-Scale Multidisciplinary Design Optimization of Aircraft With CFD and Mixed-Fidelity System Models

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

Aircraft conceptual design is a high-dimensional, multidisciplinary design optimization (MDO) problem. It involves a large number of design variables (~10-1000s), multiple disciplines, and diverse operating conditions, including nominal on-design and off-design scenarios such as structural sizing and system failure modes. Recent advancements in modeling and adjoint-based sensitivity analysis have facilitated the use of low- to mid-fidelity physics-based models to achieve computationally efficient solutions for this complex, large-scale MDO problem. However, employing higher-fidelity simulations, such as CFD-based aerodynamic analysis, for large-scale MDO of aircraft concepts remains challenging. This is largely due to the high computational cost and the complex mesh movement requirements associated with significant design geometry changes, particularly at the conceptual design stage. In this paper, we propose initial steps toward a mesh movement algorithm designed to bridge the gap between high-fidelity aerodynamic analysis and large-scale MDO of aircraft concepts using mixed-fidelity system models. The algorithm minimizes the projection error volume between the mesh and the central geometry while incorporating a non-uniformity penalty to maintain mesh quality. Additionally, it computes intersection curves between intersecting geometries, ensuring that the mesh conforms to the outer hull of the geometry. We demonstrate the proposed algorithm on a simple test case involving a wing and fuselage configuration.

Nomenclature

Γ	parametric surface
M	manifold defined by a union of parametric surfaces (or curves)
$i \in \mathbb{N}$	index of a parametric surface 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
$d := \|\vec{x} - \hat{\vec{x}}\|_2$	shortest Euclidean distance from an arbitrary point $\vec{x}$ to a point $\hat{\vec{x}}$ 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
$PE_M : \vec{x} \mapsto d$	projection error map
A_{PE}	projection error area
V_{PE}	projection error volume
J	Jacobian matrix of a parametric surface (or curve)
$\det(J)$	Jacobian determinant
Ω	domain of a parametric surface (or curve)
$\|\cdot\|_2$	L2 norm

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

I. Introduction

Comprehensive system-level analysis and optimization of aircraft concepts require computational models for all relevant disciplines (e.g., weights, aerodynamics, propulsion) across a number of design conditions (e.g., steady cruise, one-engine-inoperative, gust response). At the conceptual level, regression-based models are frequently chosen to model 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 robustness of numerical solvers, and implementing sensitivity analysis (e.g., via the adjoint method) to systematically 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, using a compact solver toolsuite called the Comprehensive Analysis and high-Dimensional Design Environment for Engineering systems (CADDEE). In recent studies [2–4], we have leveraged CADDEE 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 cases. 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 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 potentail flow theory and can model non-lifting surfaces such as the fuselage. One of the main benefits of the panel method is its relative computational efficiency compared computational fluid dynamics (CFD). However, panel methods are limited to subsonic flows and do not model the boundary layer by themselves.

Applying computational fluid dynamics (CFD), e.g., by 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 consider 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. This leads to heterogeneous and oftentimes mixed-fidelity computational models of the aircraft. While it is computationally intractable (without the use of super computers) to model each discipline using high-fidelity methods, this work aims toward CFD-based aerodynamic analysis while considering lower-fidelity physics-based models for other disciplines.

To successfully apply panel methods or CFD to large-scale aircraft conceptual design, a key requirement is a robust surface mesh movement algorithm that permits large changes in the design geometry. In this paper, we present the first steps toward a new surface mesh movement algorithm that re-computes the intersections between components while simultaneously updating the surface mesh nodes accordingly and preserving mesh quality. As proof of concept, we demonstrate the algorithm on a simple fuselage-wing geometry. The initial results of this paper suggest that the algorithm could support the use of panel methods and, eventually, CFD-based aerodynamic analysis in large-scale system-level design optimization of aircraft concepts.

In the remainder of this paper, we first give a short overview of our general geometry modeling approach and summarize notable mesh movement algorithms in Sec. II. Then, we present the mesh movement algorithm in Sec. III before showing the results of the algorithm in Sec. IV.

II. Background

A. Geometry Modeling

We use a geometry-centric approach with a high-fidelity representation of the outer mold line (OML) that is typically 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 basis spline (B-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) and projections. Free-form deformation 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 and mesh quality, making it particularly valuable in applications like aircraft design and shape optimization. A projection is the solution to an optimization problem to find the parametric coordinates corresponding to the minimum distance between points and the OML. Throughout optimization, the parametric coordinates do not change and therefore we can easily re-evaluate geometric quantities such as meshes or distances as the optimizer manipulates the FFD blocks. This is done through a simple matrix-vector (or matrix-matrix) product of the basis matrix, which is assembled from the evaluation of the basis functions at the parametric coordinates, and a vector (or matrix) of control points, which are manipulated through FFD.

B. Mesh Movement Algorithms

Surface mesh movement algorithms are critical for managing aerodynamic simulations in aircraft design, especially when accommodating large geometric deformations. We provide a brief overview of some of the notable approaches and their applications:

- *Spring-Analogy Methods* [17, 18]: These methods use a system of virtual springs connected between mesh nodes to redistribute the mesh. They are computationally efficient and widely used for moderate deformations but can struggle with preserving mesh quality under large transformations.
- *Elliptic (Laplace-Based) Smoothing* [19–21]: By solving a Laplace equation for mesh node displacement, these methods distribute deformations smoothly across the mesh. They are more robust than spring-based methods but may fail when handling highly non-linear deformations or near singularities.
- *Radial Basis Functions (RBF)* [22, 23]: RBF interpolation allows for accurate and smooth movement of mesh nodes based on control points. It is particularly effective for large-scale deformations, such as translating and rotating wings relative to fuselages, but it can become computationally expensive for dense meshes.
- *Elasticity-Based Methods* [24, 25]: These algorithms treat the mesh as an elastic body and solve continuum mechanics equations to move the nodes. They are highly robust for large deformations, maintaining mesh quality, but they require significant computational resources.

In modern aircraft design, these techniques are integrated with multi-fidelity frameworks to handle diverse operational and design requirements. For instance, mesh deformation techniques are tailored to align with variable-fidelity modeling tools ranging from VLM to CFD-based aerodynamics to support multidisciplinary design optimization (MDO) workflows effectively.

III. Methodology

In this section, we present the mesh movement algorithm for updating a surface mesh defined over a set of intersecting manifolds. This algorithm solves an augmented surface area and volume minimization problem to re-compute the intersection of two intersecting manifolds while simultaneously optimizing the surface mesh quality. We start by introducing some relevant nomenclature and definitions for the algorithm.

A. Algorithm Nomenclature and Definitions

$M_{1,2}$	2-D intersecting manifolds one and two
M_3	2-D manifold that approximates the outer hull of $M_{1,2}$ (e.g., surface mesh)
M_{int}	1-D manifold that represents the intersection between M_1 and M_2

$\vec{c}$ vector of coefficients for surfaces/curves contained in M_3 or M_{int}
$\phi_i(\vec{c})$ scalar-valued function the penalizes non-uniformity in M_3 or M_{int} (i.e., bad mesh quality)
N_3 number of parametric surfaces in M_3
N_{int} number of parametric curves in M_{int}
$\tilde{\mathcal{S}}$ set of parametric surfaces Γ_j contained in M_1 and M_2
$\hat{\mathcal{S}}$ set of parametric surfaces Γ_j contained in M_3
$\vec{r}^{k=1:3}$ three reference points $E_{M_3}(i^k, \vec{u}^k)$ on M_3 (intended to "anchor" M_3 in place during mesh deformation)
$\tilde{A}_{PE}$ projection error area of $M_{1,2}$: the area that arises when projecting M_{int} onto $M_{1,2}$

$$\tilde{A}_{PE} := \sum_{i}^{N_{int}} \int_{\Omega_{int}^i} \min_{j\in\tilde{\mathcal{S}}} \left(PE_{\Gamma_j}\left(E_{M_{int}}(i,\vec{u})\right)\right) \|\det(J_{int})\|_2 \, d\vec{u} \tag{1}$$

$\hat{A}_{PE}$ projection error area of M_3: the area that arises when projecting M_{int} onto M_3

$$\hat{A}_{PE} := \sum_{i}^{N_{int}} \int_{\Omega_{int}^i} \min_{j\in\hat{\mathcal{S}}} \left(PE_{\Gamma_j}\left(E_{M_{int}}(i,\vec{u})\right)\right) \|\det(J_{int})\|_2 \, d\vec{u} \tag{2}$$

V_{PE} projection error volume: the volume that arises when projecting M_3 onto $M_{1,2}$

$$V_{PE} := \sum_{i}^{N_3} \int_{\Omega_{M_3}^i} \min_{j\in\tilde{\mathcal{S}}} (PE_{\Gamma_j}(E_{M_3}(i,\vec{u}))) \|\det(J_{M_3})\|_2 \, d\vec{u} \tag{3}$$

B. Mesh Movement Algorithm

Before presenting the mesh movement algorithm, we state some assumptions about the preceding nomenclature.

1) Manifolds $M_{1,2}$, M_3, and M_{int} are initially given
2) The projection error areas $\tilde{A}_{PE}$, $\hat{A}_{PE}$ and volume V_{PE} are initially close to zero
3) The total measure of non-uniformity $\sum_i \phi_i(\vec{c})$ is initially minimized
4) The projection error maps PE_{Γ_j}, $j \in \mathcal{S}$ are given and are differentiable

Having established the relevant nomenclature, definitions, and assumptions, we now present the mesh movement algorithm below.

In Step 3, requiring the distance of the reference points $\vec{r}^k$ to $M_{1,2}$ to match the initial distance determined in Step 1 effectively "anchors" the coefficients of M_3 This is analogous to determining a unique surface mesh on a sphere. Without sufficient anchoring, or if the mesh is anchored at only one or two locations (especially if opposite), the mesh can rotate freely around the sphere, leading to an infinite number of possible configurations. The second constraint ($\hat{A}_{PE} = 0$) ensures that M_3 tightly approximates the outer hull of $M_{1,2}$ as the intersection changes. For example, consider a wing-fuselage intersection. If the wing translates upwards, the intersection moves inboard due to the curvature of the fuselage. Without the constraint $\hat{A}_{PE} = 0$, M_3 (i.e., the mesh) will not reflect this change in the intersection and protrude outwardly, depending how much the wing is moved.

The objective functions in Prob. 5 and Prob. 6 combine minimizing the projection error area and volume with ensuring high mesh quality through the inclusion of a non-uniformity penalty function, $\phi(\vec{c})$. For the 1-D intersection manifold M_{int}, "mesh quality" refers to uniform or otherwise structured spacing of mesh nodes along the intersection curve. For the 2-D manifold M_3, the penalty function depends on the representation of M_3. For instance, if M_3 is represented as a triangulation $\mathcal{T}$ consisting of N_3 triangles, the non-uniformity penalty function may be expressed as

$$\phi_{\mathcal{T}_i}(\vec{c}_i) = \left(\frac{e_{1_i}}{e_{2_i}}\right)^p + \left(\frac{e_{2_i}}{e_{1_i}}\right)^p + \left(\frac{e_{1_i}}{e_{3_i}}\right)^p + \left(\frac{e_{3_i}}{e_{1_i}}\right)^p + \left(\frac{e_{2_i}}{e_{3_i}}\right)^p + \left(\frac{e_{3_i}}{e_{2_i}}\right)^p, \tag{7}$$

where $e_{(1,2,3)_i}$ are the edges of the i^{th} triangle, with $i = 1, \ldots, N_3$. The parameter p can be increased to impose stronger penalties on skewed triangles. It is worth pointing out that other mesh quality such aspect ratio, angle quality, or skewness are also possible.

Algorithm 1 Mesh movement algorithm

1: **Step 1** Finding parametric coordinates of the surfaces in $M_{1,2}$ that minimize the projection error of $\vec{r}^k$ ($k = 1 : 3$)

$$\min_{i^k_{1,2}\vec{u}^k_{1,2}} \quad PE_{\Gamma_j}(\vec{r}^k) \tag{4}$$

Denote the solution to Prob. 4 as d^{k^*}

2: **Step 2**: Compute the coefficients $\vec{c}_{i_{int}}$, $i = 1 : N_{int}$ of the intersection manifold M_{int} by solving

$$\begin{aligned} \min_{\vec{c}_{i_{int}}} \quad & \alpha\tilde{A}_{PE} + \beta \sum_{i}^{N_{int}} \tilde{\phi}_i(\vec{c}_{i_{int}}), \\ \text{s.t.} \quad & E_{M_{int}}(i_0, \vec{u}_0) = E_{M_{int}}(i_{N_{int}}, \vec{u}_{N_{int}}), \end{aligned} \tag{5}$$

where α and β are tuning parameters and the constraint ensures that M_{int} remains a closed manifold

3: **Step 3**: Computing the coefficients $\vec{c}_{i_{M_3}}$, $i = 1 : N_3$ of M_3 by solving

$$\begin{aligned} \min_{\vec{c}_{i_{M_3}}} \quad & \gamma V_{PE} + \theta \sum_{i}^{N_3} \hat{\phi}_i(\vec{c}_{i_{M_3}}) \\ \text{s.t.} \quad & PE_{\Gamma_j}(\vec{r}^k) = d^{k^*}, \\ & \hat{A}_{PE} = 0 \end{aligned} \tag{6}$$

where γ and θ are tuning parameters and the constraint ensures that the solution is unique.

For Alg. 1 to work reliably, the projection error map PE_{Γ_j} needs to be fast and robust. When M_3 is a triangulation, the projection error map can be computed analytically, as there are three possible cases for the solution: the closest point may lie on a vertex, an edge, or within the interior of a triangle. From this, the analytic derivative can be formulated in a straightforward manner; however, the derivative will be discontinuous, which may lead to difficulties when solving Prob. 6, due to increased numerical noise. For fine meshes, searching across all triangles can be computationally expensive. To enhance efficiency, the search can be optimized by leveraging vectorization (if implemented in high-level languages such as Python) and using KD-trees to reduce the number of elements searched.

Lastly, for large mesh deformations that may occur when M_1 and M_2 undergo significant translations or rotations relative to one another, solving Prob. 6 can become challenging without a good initial guess for the coefficients $\vec{c}_{i_{M_3}}$. For instance, consider M_1 as a manifold representing a fuselage and M_2 as a manifold representing a wing. If the wing undergoes a significant translation and/or rotation relative to the fuselage, the mesh (represented by M_3) may distort beyond the corrections handled by Alg. 1. To address this, we adopt a two-step approach. First, the nodes are moved according to their corresponding manifold (e.g., nodes on the wing are translated along with the wing). Second, nodes near the intersection are adjusted using an inversely weighted scheme. This scheme applies the translations and rotations of the nodes on the intersection to the surrounding nodes, with the influence decaying exponentially as the distance from the intersection increases. This prevents issues such as overlapping element edges while ensuring that the nodes on M_3 remain relatively close to $M_{1,2}$, thus maintaining a small projection error volume.

IV. Results

We demonstrate Alg. 1 using a simple test geometry shown in Fig. 1 that consists of a fuselage (M_1) and a half-wing (M_2). We generate a coarse surface mesh (M_3) using the Gmsh software [26]. All 2-D manifolds are represented using triangulations and the intersection curve (M_{int}) is determined using OpenVSP [15].

We create two mesh deformation scenarios. First, we rotate the wing by 15 degrees. We then rotate the mesh nodes on the wing accordingly and apply the inverse-distance weighting scheme to move the nodes on the fuselage. Figure 2 shows the corresponding sequence of steps. We then apply Alg. 1 and show the results in Fig. 5. As expected, the algorithm minimizes the projection error volume created around the mid-section of the fuselage due to the inverse-distance mesh movement applied (see Fig. 4c). In addition, the penalization of skewed triangle leads to higher-quality elements.

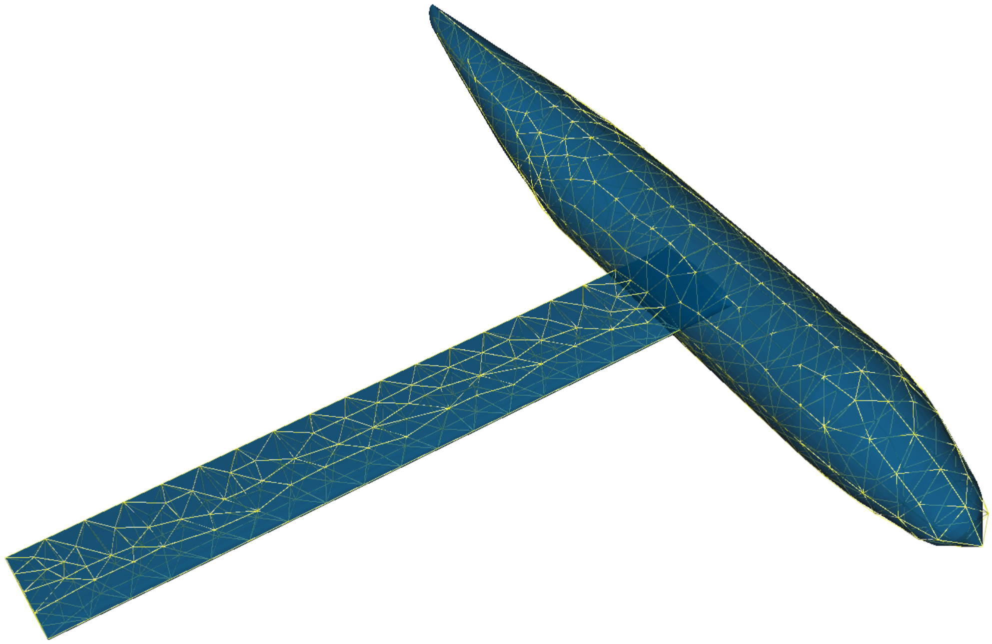
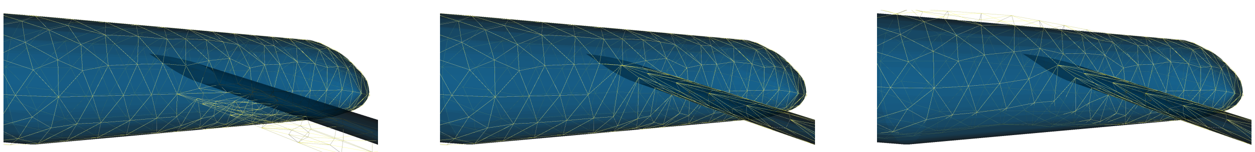


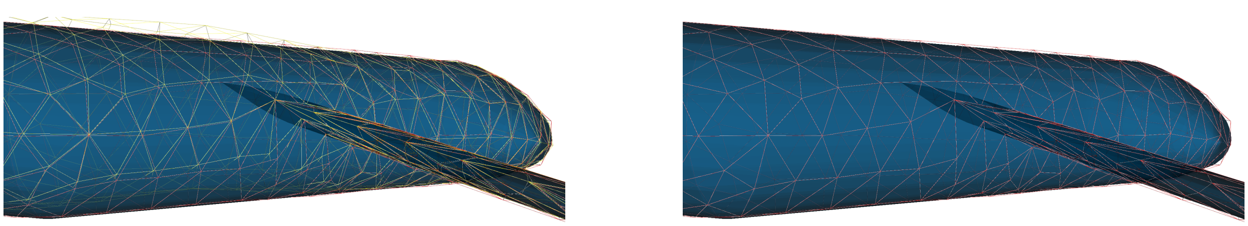
Fig. 1 Fuselage-wing test geometry with a coarse surface mesh

In the second mesh deformation scenario, we translate the wing aft by 0.5 m and follow the same steps as in the first scenario to find a good initial guess for the surface mesh node design variables. The results shown in Fig. 5 confirm that the algorithm indeed minimizes the projection error volume between the mesh the central geometry, while improving the mesh quality



(a) Rotation of wing only **(b) Rotation of wing and corresponding wing mesh nodes** **(c) Movement of fuselage nodes based on inverse-distance weighting scheme**

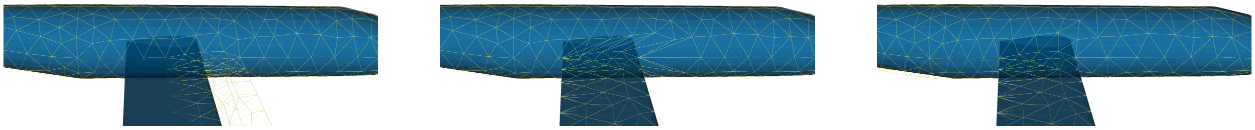
Fig. 2 Mesh movement scenario 1 (wing rotation): Steps for generating a good initial guess of the mesh coordinates before applying Alg. 1



(a) Optimized (red) and initial (yellow) mesh **(b) Optimized mesh only**

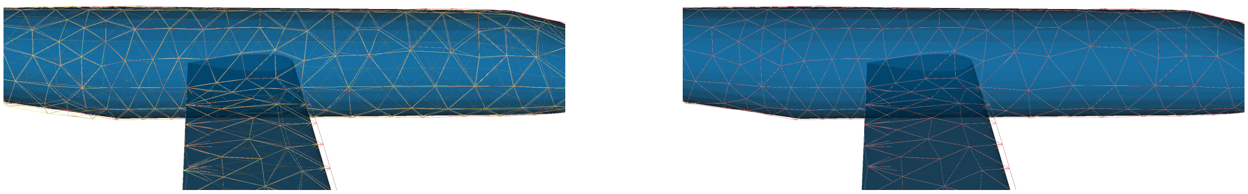
Fig. 3 Mesh movement scenario 1: optimization results

While these results look promising, there a few limitations. First, the minimization of the projection error area and volume are too costly (on the order of ten minutes) with the current implementation. However, by leveraging vectorization techniques or lower-level programming languages we will be able to significantly reduce the computational



(a) Translation of wing only **(b) Translation of wing and corresponding wing mesh nodes** **(c) Movement of fuselage nodes based on inverse-distance weighting scheme**

Fig. 4 Mesh movement scenario 2 (wing translation): Steps for generating a good initial guess of the mesh coordinates before applying Alg. 1



(a) Optimized (red) and initial (yellow) mesh **(b) Optimized mesh only**

Fig. 5 Mesh movement scenario 2: optimization results

cost. Second, the projection error (PE) maps used in this study are tailored specifically toward triangulations. However, future versions of the algorithm should be agnostic of the functions used to represent parametric surfaces. A foreseeable challenge is to ensure that the projection error maps are differentiable. Third, the algorithm tends to be less robust for sub-optimal initial guesses for the area and volume error minimization problems.

V. Conclusion

A key requirement for integrating CFD-based aerodynamic analysis in large-scale system-level design optimization of aircraft concepts is a mesh movement algorithm that can gracefully permit large changes in the central geometry. In this paper, we describe a surface mesh movement algorithm that updates mesh node locations by minimizing a projection error volume between the mesh and the central geometry, while ensuring high mesh quality. In the process, the algorithm recomputes the intersection between two manifolds (i.e., two unions of parametric surfaces) by minimizing the projection error area between the intersection curve and the parametric surfaces of the intersecting manifolds. This helps ensure that the surface mesh remains on the outer hull of the central geometry. We demonstrate this algorithm on a simple test geometry consisting of a fuselage and a wing, to which we apply rigid-body translations and rotations.

The algorithm shows promise in bridging the gap between high-fidelity, CFD-based aerodynamics and large-scale system-level design optimization with mixed-fidelity models. However, several improvements are required to fully realize this capability.

1) *Efficiency*: The minimization of the projection error area and volume needs to be faster in order for the algorithm to be practical
2) *Generality*: For this work, the algorithm was tailored toward triangulations. However, in general the intersecting manifolds may contain parametric surfaces that belong to other function spaces.
3) *Robustness*: The algorithm can be improved to be less reliant on a good initial guess for the projection error area/volume minimization problems

Acknowledgments

This work was supported in part by NASA, award No. 80NSSC21M0070.

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] 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.

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

[19] 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.

[20] 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.

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.