Title
Gradient-based design optimization using CAD-based parameterization

Permalink
https://escholarship.org/uc/item/8rx2900b

Author
Nascenzi, Thomas Richard

Publication Date
2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Gradient-based design optimization using CAD-based parameterization

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Mechanical Engineering

by

Thomas Richard Nascenzi

Committee in charge:

Professor John Hwang, Chair
Professor Nicholas Gravish
Professor David Kamensky

2020

The thesis of Thomas Richard Nascenzi is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

____

____

____

Chair

University of California San Diego

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 5 Section 1, in part, is a reprint of the material as it appears in Nascenzi, Thomas, Tae H. Ha, and John T. Hwang. “A CAD-interoperable geometry parameterization for large-scale design optimization.” AIAA Aviation 2019 Forum. 2019. The thesis author was the primary author of this paper.

ABSTRACT OF THE THESIS

Gradient-based design optimization using CAD-based parameterization

by

Thomas Richard Nascenzi

Master of Science in Mechanical Engineering

University of California San Diego, 2020

Professor John Hwang, Chair

Reconciling optimized designs with the original model can be a time-consuming process. This thesis presents a process for gradient-based optimization using CAD-based parameterization that avoids the reconciliation process by directly altering CAD parameters and automatically updating the design. However, the complexities surrounding most commercial CAD tools make analytically obtaining design sensitivities necessary for gradient-based optimization virtually impossible. The value of this process lies in its ability to numerically compute design sensitivities.

This is done with the use of an intermediary surface discretization of the model called the master-mesh. The master-mesh maintains continuity as CAD parameters change through the use of a mesh smoothing process consisting of two optimizations. This smoothing process ensures that the master-mesh deforms smoothly with the design. Other meshes can be derived from the master-mesh, and therefore, can also deform smoothly with the design. This allows for the finite-difference method to be used to calculate their gradients.

This master-mesh is demonstrated with three distinct applications. The first application consists of an aerodynamic optimization of a wing designed with a custom geometry engine. The second application consists of an aerodynamic shape optimization of a jet plug nozzle designed in CATIA V5. The third application consists of an aerostructural optimization of a wing designed in FreeCAD.

Chapter 1

Introduction

The engineering design process has changed little over the years, even as tools themselves change. This traditional design process generally takes the form of a repeating cycle of design analysis and revision. Each iteration of the cycle tries to improve upon the previous iteration's design with respect to the specifications or general design goal. In this cycle, it is typical that each engineering discipline performs a separate analysis with only superficial connections to other disciplines, or even in complete isolation, in a loosely coupled way. It then becomes the job of experts to reconcile the loosely coupled results and recommendations to create a new iteration of the design. This design process has the benefits of allowing specialists to focus only on their area of expertise, but it is ill-suited to capture inter-disciplinary effects that may provide insight necessary to improve the design.

Multidisciplinary design, analysis, and optimization (MDAO) attempts to overcome these shortcomings. Optimization provides a numerically driven method for iterating and improving a design. It also captures inter-disciplinary interactions through tightly coupled analysis and simulation, and it optimizes all design variables simultaneously.

Although improvements within the MDAO field are being made rapidly, it has still not advanced enough to replace traditional design in its entirety, and it may never. Therefore, MDAO should not be expected to replace traditional design. Instead, it should be treated as an increasingly powerful tool in the increasingly metaphorical engineering toolbox. The objective of the MDAO tool is to reduce the engineering-hours required to meet design specifications, or produce a design with increased efficiency or ability.

A class of problems in MDAO is shape optimization. Shape optimization concerns itself with optimizing the geometric shape of a model. Models can be parameterized a number of ways, but the most commonplace parameterization is found in computer-aided design (CAD) tools. CAD can refer to many tools, but in this paper it will refer to comprehensive and modeling tools, such as CATIA, which can be described as offering CAD, computer-aided manufacturing (CAM), and/or computer-aided engineering (CAE). Shape optimization methods automatically change and tweak design variables in search of an optimal shape.

I have identified two important qualities of any shape optimization method. The first is that the method must be time efficient while still producing meaningful results. Meaningful results come from optimization when meaningful analysis tools are used. Many high-fidelity analysis tools can operate on timescales of hours or days, even with modern computing clusters. Unfortunately, an optimization solver has to call these high-fidelity tools for every iteration of a design. The benchmarking paper by Lyu et al. [21] shows that gradient-free methods tend to grow at a greater than quadratic rate, while gradient-based methods grow at roughly a linear rate, even when utilizing the finite-difference method to compute the gradients. Although gradient-based methods are more difficult to implement than gradient-free methods, they are necessary to keep the optimization time of design problems within a reasonable time frame. And even if low-fidelity models are being used, increased efficiency will increase the rate of design space exploration. Because of this, it is my conclusion that gradient-based methods are greatly preferable to gradient-free methods, even if they are more difficult to implement.

One potential downside to using gradient-based optimization methods is that they find a local minimum as opposed to the global minimum found by gradient-free methods. This means that the result of the optimization can be highly dependent on the starting values used for the design variables. However, in another benchmarking paper by Lyu et al. [20], it is shown that the initial starting values used in shape optimization have little impact on the results. The benchmarking paper only looked at a representative problem, and is by no means exhaustive, but still implies that shape optimization can resemble a convex optimization. Additionally, the efficiency gained by using gradient-based methods can allow for multiple optimization runs with different starting values to be run in the same amount of time as it takes to run one gradient-free method.

The next quality of a useful shape optimization method is interoperability with CAD, the cornerstone of the modern design process. It is from this cornerstone that all supplemental design engineering tools, including optimization, are often centered around. Approaches to design optimization should respect and maintain the original CAD representation. The parametric modeling found in CAD usually includes a feature tree or model, which allows for the changing of one feature to propagate to all other features. Ideally, a shape optimization tool should be able to seamlessly utilize a CAD model without a need for labor intensive model translation. Unfortunately, this means that not only does the optimization tool need to seamlessly import a CAD model, it also needs to export the updated model back into CAD. Additionally, if the model is reparameterized for optimization, the optimal model will not contain any of the feature-based parameters originally found in the CAD model. In order save engineering hours, it is ultimately better to not reparameterize a CAD model and, instead, maintain the CAD design parameterization in order to avert the laborious process of model translation. Additionally, the parameters present in a well-posed design can make for intuitive design variables for an optimization problem. This conclusion, in addition to the first, means that a useful shape optimization method will maintain the parameterization present in CAD and will utilize gradient-based methods.

A major obstacle in developing a general gradient-based, shape optimization method that utilizes CAD parameterization is the calculation of design sensitivities. Design sensitivities capture how the shape of the design changes as design parameters change. Calculating these design sensitivities is required for gradient-based optimization. Ideally, these sensitivities would be computed analytically, but the opaqueness of commercial CAD (due to their proprietary nature) prevents an obvious solution. Overcoming this obstacle is necessary for a gradient-based shape optimization method that maintains the CAD parameterization.

The following is a summary of the remainder of the thesis. First, a review and summary of the current state of design parameterization and sensitivity calculation is presented. This is followed by two chapters introducing a novel approach and methodology for computing design sensitivities for a CAD-parameterized design. Then, three separate implementations of the method are introduced. The three implementations are a CAD-free implementation, a CATIA and FUN3D integrated implementation, and an open-source implementation. An example optimization is also presented for each implementation. Then, concluding remarks will be made, along with recommendations for future work to be done in the area.

Chapter 2

Background

As stated before, design sensitivities describe how the geometry changes with respect to design variables. When it comes to shape optimization, this can take two forms. The first form is sometimes referred to as “geometric” sensitivity. Geometric sensitivities answer how does the local shape of a design change as design variables are changed. The second form is sometimes referred to as “tessellation” sensitivity. Tessellation sensitivities describe how elements on the surfaces change as design variables change. Many analysis methods, such as computational fluid dynamics (CFD) or finite element analysis (FEA), require that the design be represented by a mesh or tessellation. Therefore, tessellation sensitivities are the sensitivities that are more useful, even though they are only approximations of the true geometric sensitivity.

Additionally, designs can be parameterized in a variety of ways. It is important to understand the different methods of parameterizing a design and how those different parameterizations influence design sensitivity calculation. In a shape parameterization survey by Samareh [27], all of the methods surveyed have analytically obtainable derivatives, except for CAD parameterization. Table 2.1 provides a list of the shape parameterizations and their CAD-interoperability and sensitivity calculation qualities as determined by Samareh. Since the review came out, there have been a few developments towards analytically obtaining derivatives for a CAD parameterized

Table 2.1: List of relevant shape parameterizations and their CAD-interoperability and sensitivity calculation qualities.

Parameterization	CAD Interoperability	Analytical Sensitivities	Summary
Basis Vector	Poor	Yes	Redefines design surface gradients as a set of basis vectors. [22]
PDE Approach	Poor	Yes	Uses elliptic partial differential equations to define surfaces. [6]
Analytical	Poor	Yes	Linearly adds shape functions along with a coefficient, and treats the coefficients as design variables. [17]
Discrete	Poor	Yes	Treats discrete boundary points as design variables. [9]
Free-form deformation	Neutral	Yes	Applies global and local deformation functions to twist, translate, or bend various parts of the shape. [3]
Spline	Neutral	Yes	Utilizes splines to represent surfaces and treats spline parameters, usually control points, as design variables. [7, 28]
MASSOUD	Neutral	Yes	Uses the free-form deformation approach but parameterizes shape perturbations as opposed to the geometry. [25]
CAD	Good	Sometimes	Uses the feature-based design parameterization found in CAD tools. [26]

geometry.

There are a number of reasons why analytical sensitivities for CAD parameterized designs are difficult to obtain. Commercial CAD tools often aim to be comprehensive in regards to what geometries can be realized. This means that many of the parameterizations shown in Table 2.1 may be contained within one CAD model. Also, it is not always clear which topological representation the CAD tool is using for each local geometry. For example, a cylinder could be described by NURBS surfaces or by elliptical PDEs. Additionally, each unique geometry parameterization can create intersections with other parameterizations further increasing the complexity. It may be possible to analytically obtain the sensitivities for simple designs, but for complex geometries it is virtually impossible. One would need access to the source code and a great deal of time.

However, there have been a few cases where design sensitivities were obtained analytically from a CAD parameterized geometry.

The Engineering Sketch Pad (ESP) [15] is a specially designed CAD tool that keeps optimization in mind. Gradients are analytically obtainable for nearly every operation in ESP, and the finite-difference method is automatically used for every other operation. The design sensitivities can be calculated from these gradients. Dannenhoffer et al. [10] provide a method for, and an example of, coupling ESP into a greater optimization framework.

Another development involves automatic differentiation [14]. Banović et al. [2] apply automatic differentiation to the open source CAD kernel Open CASCADE (OCC). This means that every CAD tool that is based on the OCC kernel could potentially provide analytical design sensitivities. However, all of the analytically obtained CAD sensitivities were obtained for open-source, non-commercial CAD, so their usefulness is limited. Additionally, these methods are unlikely to be applied to commercial CAD because of the lack of access to source code and general proprietary nature. Therefore, the price of maintaining commercial CAD parameterization is forgoing analytical sensitivity calculation for numerical sensitivity computation.

However, CAD parameterization is, perhaps unsurprisingly, the only parameterization that Samareh gave a ”Good” rating for CAD connectivity. Some parameterizations were given a “neutral” for CAD connectivity, but neutral is not sufficient. Although neutral methods do not require any additional mesh post-processing or smoothing, they still require being reconciled with a CAD model. In an Altair topology optimization paper by Tomlin et al. [30] a major obstacle to greater optimization usage at Airbus was the amount of work required per optimization. Each part required multiple optimization runs, and, after each run, it had to be re-interpreted in CATIA V5. Although this paper considers topology optimization, it is still relevant because topology and shape optimization have similar goals and obstacles. Therefore, having good CAD connectivity removes an obstacle to greater use of optimization.

To overcome CAD parameterization’s lack of analytical sensitivities, the finite-difference method can be used to compute design sensitivities. Each design parameter has to individually be perturbed, and the CAD tool has to generate a new model for each set of design parameters. Unfortunately, as the number of parameters grows, so does the number of perturbed models. This is potentially ill-suited to scaling, but computing power is continuously growing and is already sufficient for many design optimization problems. Additionally, MDAO is suitable for evaluating conceptual designs where the fidelity of the model is low.

Papers have already been published regarding numerical computation of design sensitivities. Truoing et al. [31] utilizes CAPRI, a third party program that provides a framework for interfacing with CAD tools, to help guide mesh deformation. Brock's thesis [8] tracks surface movement using parametric coordinates. Agarwal et al. [1] builds on the work done by Robinson et al. [24]. This method uses a faceted surface representation of the design to compute sensitivity information while maintaining the CAD parameterization. One major advantage of this method is that it is topology agnostic, which gives the optimizer a greater range with which to alter the design parameters. However, this work specifically calculates design velocities, not sensitivities. Design velocity is the normal component of design sensitivity, so it can have difficulty describing design sensitivities with a shear component. No method has yet to be widely adopted, so work is still required in this area.

Chapter 3

Approach

Design optimization requires design analysis or simulation, and typical design analysis tools require a discrete representation of the design that usually takes the form of a volume or surface mesh. When considering tessellation sensitivities, calculating how these meshes deform as design parameters change is calculating the design sensitivities. As discussed in the previous section, these sensitivities must be calculated with the finite-difference method because it is virtually impossible to analytically calculate how a CAD design changes with respect to design parameters, let alone analytically calculating how a discrete representation of the design changes with respect to design parameters. Therefore, in order to numerically calculate the design sensitivities, it is necessary to obtain perturbed meshes along with the unperturbed mesh. In addition to perturbed meshes, an updated unperturbed mesh will need to be regenerated when design parameters change.

Perturbed and updated meshes are easily obtained in certain parameterizations, such as the free-form deformation parameterization. Free-form deformation started as a method for deforming solid geometric models in the field of computer graphics [29]. Since then, it has successfully been applied in the field of optimization [19, 13, 20]. The parameters used in free-form deformation directly control the mesh nodes. This makes perturbing and re-evaluating meshes trivial. However, this method strips away all relation between the mesh and the CAD model and the free-form deformation parameters and the CAD parameters.

One proposed solution to create smoothly deforming meshes while maintaining CAD connectivity is to track the mesh nodes using their parametric coordinates. After the design is perturbed, the perturbed mesh can be obtained by re-evaluating the nodes using their parametric coordinates. This method was used with success for a design parameterized in OpenVSP by Yildirim et al. [32] to optimize a boundary layer ingestion system. However, this method does not work for all CAD tools. OpenVSP [16] is a parametric aircraft geometry tool that allows users to rapidly develop a 3D model. OpenVSP also strictly deals with parametric surfaces, which allows mesh nodes to be parametrically re-evaluated as the design changes. Universal CAD tools have to be capable of generating more complex geometries, and because of this, virtually no design contains a consistent set of parametric surfaces. This means that re-evaluating a node's parametric coordinates in most CAD tools may become impossible as design variables change.

3.1 Master-Mesh

I propose a novel approach that perturbs and re-evaluates meshes smoothly while maintaining CAD parameterization. To calculate perturbed meshes for CAD parameterized designs, all methods presented in this paper utilize a "master-mesh" and "sub-meshes." The goal of these two meshes is to separate two competing mesh requirements of design optimization. Typically, mesh generating algorithms try to provide a discrete representation most suited for a particular analysis tool or simulation. These mesh generation methods typically do not have design optimization in mind, so they do not necessarily create smoothly deforming meshes as design variables are perturbed. These meshes form the sub-meshes, or one half of the master-mesh approach. The master-mesh concerns itself with maintaining smoothness and continuity as the design deforms. This mesh can be generated using a more consistent manner that does not concern itself with a particular analysis tool or simulation. Additionally, this mesh can act like a parametric surface representation of CAD surfaces. Therefore, the sub-meshes can be re-evaluated parametrically using the master-mesh. With this setup, it is possible to calculate the design sensitivities using the finite-difference method since smoothness is maintained.

The form and method for generating the master-mesh varies in the three implementations of this approach, but it always meets a few requirements. First, the number of nodes and elements of the master-mesh never changes with respect to design parameters. Next, the relative position of each node must be maintained in order to maintain smoothness. This means that a node that lies on the corner of where two surface edges meet will always lie on that corner, no matter how the design changes. Finally, parametric coordinates cannot be used to generate the master-mesh. This makes the master-mesh suitable for CAD tools like CATIA V5, which does not maintain consistent parametric surfaces. If these requirements are met, the master-mesh will provide accurate design sensitivities.

Chapter 4

Methodology

The ultimate goal of this method is to provide a general and accessible method for calculating design sensitivities of a CAD model. A primary version of this method is outlined in Section 4.1, but it has only been used within a CAD-free optimization framework. An alternative version of the method that is less general is outlined in Section 4.2, and it has been used with two different CAD tools.

4.1 Method

The user is expected to have two items ready before using this method. The first item is a parameterized CAD model. This model can be created in any CAD tool that is capable of generating STEP and STL files via its API, which includes most, if not all, commercial CAD tools. During optimization, the parameters of the CAD model will be used directly as design variables. This means that the optimization method will be changing CAD parameters and rebuilding the CAD model throughout the optimization process. Second, the user must provide a sub-mesh. The sub-mesh, or analysis mesh, is the mesh that the design sensitivities will be calculated for, and can be used by user defined analysis or simulation tools during the optimization.

Once these items are ready, the user must create an initialization script. Within the initialization script, the user must specify a number of things. The name and directory of the CAD file and the mesh file. The name of each design surface in the CAD tool. And finally, a list containing the name, starting value, and step size of each CAD parameter. With this information, connections are established with the CAD design parameters, and the initial STEP and STL files are generated. Then, the initial master-mesh is created automatically.

To generate the master-mesh, the STEP and STL files are first read. The STEP files are read using an open-source, third-party tool that generalizes STEP file surfaces as queryable surfaces. Next, every edge of every design surface is discretized using the queryable STEP file and STL file, The goal of the edge discretization algorithm is to evenly space the STL edge nodes along the edge curve. This is done so that as the curve deforms and design variables change, each node represents the same section of the curve. This means that a node lying in the middle of the curve will always lie in the middle of the curve, and a node lying on one end of the curve will always lie on that end of the curve.

4.1.1 Edge Discretization

Equally spaced nodes on a curve in parametric space do not necessarily correlate to equally spaced nodes in real space, so an algorithm was developed to evenly space the edge nodes along a curve in real space. The algorithm uses a piecewise linear approximation of the curve consisting of $rn-1$ lines where r is a user specified resolution value and n is the number of STL nodes on the curve. To create the approximation, the edge parametric coordinate of each STL node is found. Then, $r-1$ nodes are created between each node in parametric space. Connecting theses nodes results in the linear approximation. These new nodes, along with the original STL nodes, form the starting values of an optimization problem defined as

$$\begin{aligned} \text{minimize} \quad & \sum_{i=0}^{rn-1} L_i^2(u_i, u_{i+1}) \\ \text{w.r.t} \quad & u_1, u_2, \ldots u_{rn-1} \end{aligned}$$

where L is the length of each line, and u_i is the parametric coordinate of each node. Notice that u_0 and u_{rn} are used, but are not included as design variables. This is because those two nodes lie on the ends of the curve and should not move. After this optimization is run, a set of linearly spaced nodes of size rn is left on the curve where every rth node is one of the original edge nodes given by the STL file. Then, the in-between nodes are removed, and only the original STL nodes are left.

4.1.2 Interior Surface Discretization

After the edge STL nodes are adjusted, the interior STL nodes are put through a smoothing filter. First, each node is projected onto the surface and the parametric coordinates are obtained. Then, the nodes are adjusted by another optimization problem similar to the one used for the edge nodes. The optimization problem is defined as

$$\begin{aligned} \text{minimize} \quad & A_{ij} I_{ij}^2(u_i, v_i, u_j, v_j) + B_{ij} E_{ij}^2(u_i, v_i) \\ \text{w.r.t} \quad & u_0, v_0, u_1, v_1, \ldots u_m, v_m \end{aligned}$$

where u and v are parametric coordinates, m is the number of interior nodes, I is a matrix containing the distances between each interior node, E is a matrix containing the length between each interior node and each edge node, and A and B are mapping matrices. The mapping matrices represent the node to node connections present within the STL file. To help demonstrate how the mapping matrices are created, a simplified triangulation is shown in Fig. 4.1. The corresponding mapping matrices

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

show the connection information. Each value of 1 indicates a connection between two nodes. In the A matrix, there are five 1 values, and in the figure, there are five interior node to interior node connections shown in blue. While in the B matrix, there are eight 1 values, each corresponding to an interior node to exterior node connection and are shown in orange. The rows and columns of each matrix correspond to the indices of the two nodes in the connection.

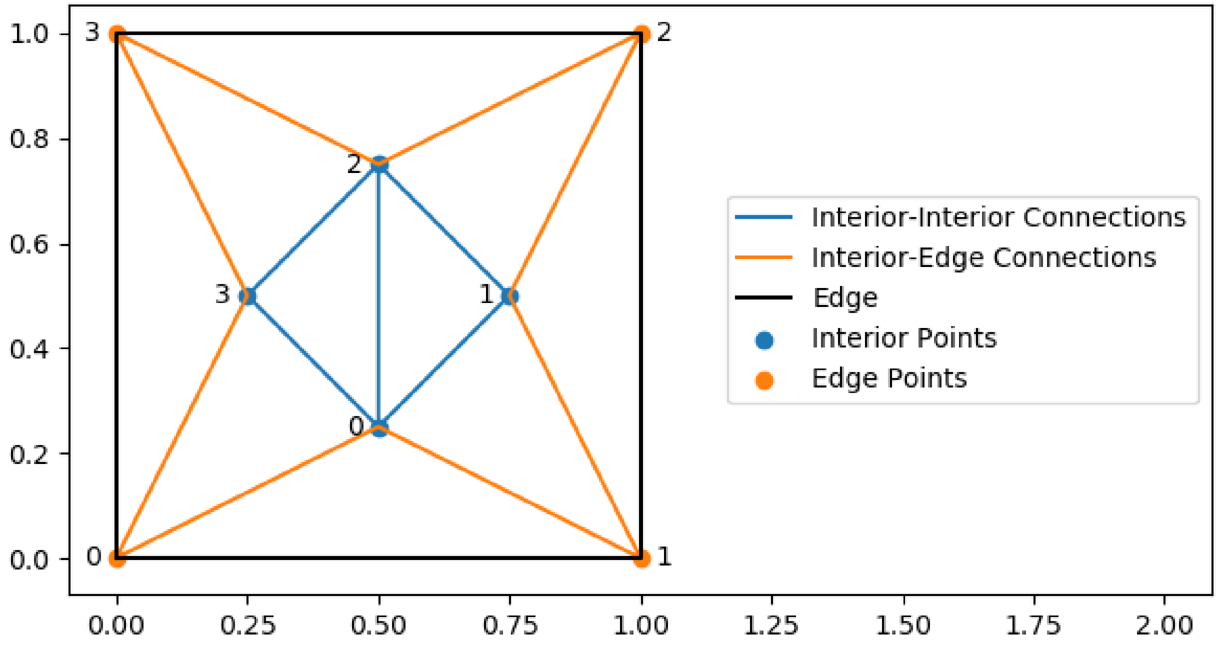

Figure 4.1: Example of interior and edge connections. All lines connecting interior nodes to edge nodes are considered edge connections. Lines connecting two interior nodes are considered interior connections.

This filtered surface triangulation is the master-mesh. The goal of the master-mesh is to deform smoothly as design variables change and the underlying surface morphs. However, the STL exported by the CAD tool cannot be expected to maintain a consistent triangulation.

Therefore, instead of obtaining a new triangulation every time the design changes, the previous triangulation will be used. Even though the previous triangulation will not match the new surface, both the edge and interior node smoothing algorithms can be rerun. Rerunning these two algorithms ensures that the triangulation will conform to the new surface and that each mesh element will deform smoothly with respect to the design variables.

4.1.3 Analysis-Mesh Interpolation

However, the master-mesh itself is not always a suitable discretization of the model for all analysis tools. This is what the sub-meshes are for, and sub-meshes can be defined as functions of the master-mesh. To do this, each node on each sub-mesh is projected onto the master-mesh, and the corresponding triangular element and barycentric coordinates are obtained. This is done using Alg. 1, which uses Alg. 2.

Algorithm 1: Point Mapper

```
**Data:** $master\_mesh, sub\_mesh$
**Result:** $tri\_vert\_indices, rst\_matrix$
$tree \leftarrow KDTree(master\_mesh)$
$tri\_vert\_indices \leftarrow empty(length(sub\_mesh), 3)$
$rst\_matrix \leftarrow empty(length(sub\_mesh), 3)$
$i \leftarrow 0$
**for** *node in sub_mesh* **do**
    $index \leftarrow tree.Query(node)$
    **for** *triangle in Get_Triangles(master_mesh [index])* **do**
        $rst\_vec[i,:], is\_inside \leftarrow Calc\_RST(point, triangle)$
        $tri\_vert\_indices[i,:] = triangle.master\_mesh\_indices$
        **if** *is_inside* **then**
            *break*
        **end**
    **end**
    $i++$
**end**
```

Algorithm 2: Calc RST

```
Data: $point, triangle$
Result: $rst, is\_inside$
$u \leftarrow point[1] - point[0]$
$v \leftarrow point[2] - point[0]$
$n \leftarrow cross(u,v)$
$w \leftarrow p - point[0]$
$t \leftarrow cross(u,w).dot(n)/n.dot(n)$
$s \leftarrow cross(w,v).dot(n)/n.dot(n)$
$r \leftarrow 1 - s - t$
$rst \leftarrow [r,s,t]$
$is\_inside \leftarrow ($
    $(0 <= alpha)$ && $(alpha <= 1)$ &&
    $(0 <= beta)$ && $(beta <= 1)$ &&
    $(0 <= gamma)$ && $(gamma <= 1)$
$)$
```

The first algorithm starts by generating a k-d tree of the master-mesh. A k-d tree is a space-partitioning data structure that is useful for finding closest nodes [4]. Next, the algorithm loops through each node on the sub-mesh. The closest master-mesh node of each sub-mesh node is found using the k-d tree. Then, the barycentric coordinates of each sub-mesh node are calculated using Alg. 2 for each triangle that the closest master-mesh node is a part of. This stops once the barycentric coordinates found are between 0 and 1, which means that the node exists within that triangle. After doing this for every sub-mesh node, each node's master-mesh triangle index and barycentric coordinates are compiled into matrices. These matrices can be used to generate the sub-mesh from the master-mesh where one node is calculated with $T_{ij}V_i$ where T contains the three coordinates of the three vertices that make a triangle element and V contains the barycentric coordinates of the node within that triangle. This makes the sub-mesh a function of the master-mesh, which means that if the master-mesh deforms smoothly, the sub-mesh will deform smoothly.

4.1.4 Remarks

The primary goal of the master-mesh method is to provide a framework that facilitates gradient-based optimization of a CAD model. Smooth deformation of the sub-mesh implies that the sub-mesh is continuously differentiable with respect to the design variables. And even if the exact relation is obscured by the CAD tool, this still means that the finite-difference method can be used to approximate the derivatives. Therefore, design sensitivities can be calculated and gradient-based optimization can be used.

One limiting factor of this method is that it expects the design to maintain constant topology. If, for example, a new surface emerges during the optimization due to changing design variables, neither the master-mesh nor the sub-mesh could smoothly deform to capture it. This is because the adding and removing of topology is an inherently noncontinuous event. A surface is either there, or it is not there. One potential solution to this problem is to allow the master-mesh and sub-mesh to deform in a non-smooth way when these topological changes occur. The occasional noncontinuous function evaluation should not be a major problem for a gradient-based optimization solver, unless the minimum exists near one of the topological changes. In these cases, the erratic gradient information will prevent the optimization from converging. Although not flawless, the functionality needed to handle topological changes would be a welcomed addition to this method.

4.2 Alternative Method

4.2.1 Overview

As previously discussed, the main objective of this method is to compute design sensitivities for CAD parameterized designs. The method outline here will follow the same structure as the full implementation, but uses a different method for generating the master-mesh. The master-mesh has a few general requirements that will be reiterated here, but this implementation has a few additional restrictions. Most importantly, the mesh has to maintain continuity and smoothness as design variables change. The first step to meeting this requirement is ensuring that the number of nodes in the mesh remains constant. Additionally, each node of the mesh must continuously represent a consistent part of the design. This means that a node that lies on one edge must continuously lie on that edge. This allows the analysis tools and optimizer to accurately analyze how changing design variables affects the overall design.

To meet the requirements outlined above, an additional requirement is placed on the input geometry. All design surfaces must be able to have a structured quad mesh represent it, and as design variables change, a mesh with the same structure can be applied to it. Typically, this means that each design surface must be described as having four sides. This is the main difference and limitation of this alternative method.

With this new requirement, the process is as follows. First, the user generates initial meshes required by the analysis tools. It is from this initial mesh that all following iteration's meshes will be derived. Next, the user generates a structured quad mesh on each design surface, using a third-party tool. In this implementation, these structured quad meshes form the master-mesh. These meshes are saved and are used to generate the initial information required for following iterations. The initial information is generated first by overlaying the sub-meshes on top of the master-mesh. Then, each analysis mesh node's corresponding structured mesh cell and internal cell coordinates are obtained. This information is saved and will allow for the analysis mesh to be evaluated using only the master-mesh.

Since the master-mesh is generated using a consistent layout and consistent parameters, it deforms smoothly along with the CAD-design. Additionally, since the analysis-mesh can be interpolated from the master-mesh, the analysis-mesh also deforms smoothly. This means that the finite difference method can be uses to calculate the design sensitivities with respect to the CAD parameters.

The following outlines how this is done. First, one design variable is perturbed by an appropriate step size and the design is updated in the CAD tool. Then, the design is saved as a STEP file. From the STEP file, the master-mesh (a structured quad mesh) is generated using the same layout and same tool used to generate the initial structured mesh. This means that each cell of the structured mesh should represent the same section of a surface. From this structured mesh, the analysis mesh is re-evaluated using the saved cell and coordinate information obtained previously. The original analysis mesh and the perturbed analysis mesh are then used to calculate the design sensitivities via the finite-difference method. This process is repeated for every design variable until the entire Jacobian matrix is obtained.

4.2.2 Remarks

The alternative method follows the original method exactly except for how the master-mesh is created. This method has a couple of notable negative aspect relative to the original method. It requires that each surface be described as having four sides. In effect, this makes the alternative method less general. Additionally, it requires the use of a third-party tool to generate the structured quad mesh. Since the mesh generation is third-party, its quality and robustness can be unpredictable. One advantage of this method, however, is that the master-mesh is typically computed more efficiently when compared to the original method. This means that this implementation can be more desirable for certain high-fidelity or time-sensitive applications.

Chapter 5

Applications and Results

The master-mesh method was applied to three problems. Section 5.1 contains a wing aerodynamic optimization that minimizes drag with respect to wing design variables. The design is modeled and parameterized in a CAD-free design tool, but is treated like a black-box in order to mimic commercial CAD tools. The aerodynamic analysis is provided by the vortex lattice method (VLM). Section 5.2 contains a nozzle CFD optimization that minimizes drag and acoustic noise with respect to nozzle-plug shape design variables. The jet nozzle is modeled and parameterized in CATIA V5. The CFD simulation is provided by the tool FUN3D [5]. Section 5.3 contains a wing aerostructural optimization that minimizes fuel burn with respect to wing design variables. The wing is modeled and parameterized in FreeCAD. The aerostructural analysis is provided by a modified version of the tool OpenAeroStruct [18].

5.1 CAD-Free Wing and Aero-Analysis Application

This application utilizes a CAD-free implementation of the master-mesh method in order to help reveal any potential obstacles for a CAD-based implementation and demonstrate the feasibility of the approach without having to develop CAD interfacing scripts These objectives are achieved if the CAD-free approach is able generate design sensitivities suitable for gradient-based optimization of a parameterized model. A simple way to prove this is to perform a gradient-based optimization using the generated design sensitivities. If this is done, the method will be demonstrated as feasible. An in-house design tool capable of representing aircraft geometries as B-spline surfaces is used instead of a commercial CAD tool. In a similar vein, an in-house aerodynamic simulation tool is used to provide the analysis. By using in-house tools, more time can be spent developing the novel parts of the method as opposed to developing code necessary to interface with third-party tools. Additionally, the in-house design tools are treated like a black box in order to mimic interfacing with a commercial CAD tool.

5.1.1 Application and Results

Sub-mesh Creation

Although the master-mesh is a surface representation of the model, it is not designed to be used for any analysis or simulation. The aforementioned sub-meshes serve as the discretization suitable for design analysis. Initial analysis-meshes need to be created in addition to the master-mesh. The method used to generate the analysis-meshes can vary greatly by case and discipline.

The test case of this application utilizes the VLM to simulate the aerodynamics of the model. The VLM requires that each lifting surface be represented as a grid of infinitely thin quadrilateral panels. In the case of a wing, the grid of panels will follow the camber line. Each lifting surface's panels will be constructed from a grid of discrete points. Once the initial sub-mesh has been generated, deformed sub-meshes can then be obtained via the master-mesh.

In the test case example, each lifting surface is described by an upper and lower B-spline surface. The average of these two surfaces will result in a sufficient camber line approximation. Additionally, the average of an upper and lower quadrilateral mesh representation will result in the grid of infinitely thin quadrilateral panels required for the VLM. In order to obtain the upper and lower meshes, each surface needs to be described as having four sides. B-spline surfaces always have four sides, but trim curves may result in a surface with more than four sides. Therefore, a method for combining sides until four roughly equally sized sides is applied. These new four edges are then used to generate a Coons patch.

A Coons patch is a parameterized surface that is defined by four parametric curves that form a loop. The Coons patch is given by

$$C(u,v) = L_0(u,v) + L_1(u,v) + B(u,v). \tag{5.1}$$

where

$$\begin{aligned}
&L_0(u,v) = (1-v)C_0(u) + vC_2(u) \\
&L_1(u,v) = (1-u)C_1(v) + uC_3(v) \\
&B(u,v) = P_0(1-u)(1-v) + P_1u(1-v) + P_2(1-u)v + P_3uv
\end{aligned}$$

L_0 and L_1 are linear interpolations of the two pairs of non-adjacent curves and B is the bilinear interpolation of the four corner points defined as

$$\begin{aligned}
P_0 &= C_0(0) = C_3(1) \\
P_1 &= C_0(1) = C_1(0) \\
P_2 &= C_1(1) = C_2(0) \\
P_3 &= C_2(1) = C_3(0).
\end{aligned}$$

The main purpose of the Coons patch is not to obtain the final grid, but to obtain an initial four sided grid of quadrilaterals that is only a rough approximation of the lifting surface. To obtain the initial grid, a parametric grid of points is evaluated on the Coons patch. Next, the grid of points is projected onto the surface, resulting in an accurate approximation of the lifting surface. This projected four sided grid now describes a surface with any number of sides greater than or equal to four. And finally, this grid of points, averaged with the complimentary surface's grid of points, results in the necessary panels for the VLM.

This grid will have to be re-obtained every iteration, and the above method does not result an inherently smooth mesh deformation. The master-mesh, however, will deform smoothly. Therefore, from iteration to iteration, each grid point can be obtained by interpolating it from the master-mesh. When the mesh is initially obtained, each point's corresponding triangle index and barycentric coordinates are recorded by the program. For future iterations, the grid is re-evaluated using its triangle index and barycentric coordinates. Therefore, any mesh that can be defined by its relation to surfaces can be tracked by this method from iteration to iteration. Additionally, since a requirement of the master-mesh is that it maintains continuity, any mesh derived from the master-mesh will also maintain continuity.

Setup

The test case of this application consists of minimizing drag for a wing-fuselage-tail geometry while maintaining cruise conditions. To do this, the master-mesh implementation was integrated into a larger in-house aero-analysis framework. The framework can generate an initial parametric geometry and allows the parametric variables to be used as optimization design variables in OpenMDAO. OpenMDAO is an open-source framework for efficient multidisciplinary optimization, with special focus on gradient-based optimization [12].

Within the aero-analysis framework wings, the wing geometries are derived from the UIUC Airfoil Coordinates Database, which describes airfoils via lists of points. Two B-spline curves are fitted (via a least squares fit) to the desired airfoil profile; one for the top half and one for the bottom half. Additionally, the user specifies various sections of the wing via chord, twist, sweep, span, and dihedral parameters. From these sections, a top and bottom B-spline surface is created. The fuselage is created similarly. Again, two B-splines are fitted to a set of user defined points describing the cross sectional shape of the fuselage. Then width, height, and length are specified for various sections of the fuselage. And again, top and bottom B-spline surfaces are calculated from all of the parameters. For the test case, all lifting surfaces have the airfoil profile defined by the four digit NACA code 2411 and the fuselage has a circular cross section. Fig. 5.1 shows the complete aircraft, while Fig. 5.2 show the components being aerodynamically analyzed and a corresponding VLM mesh with exaggerated z-scales.

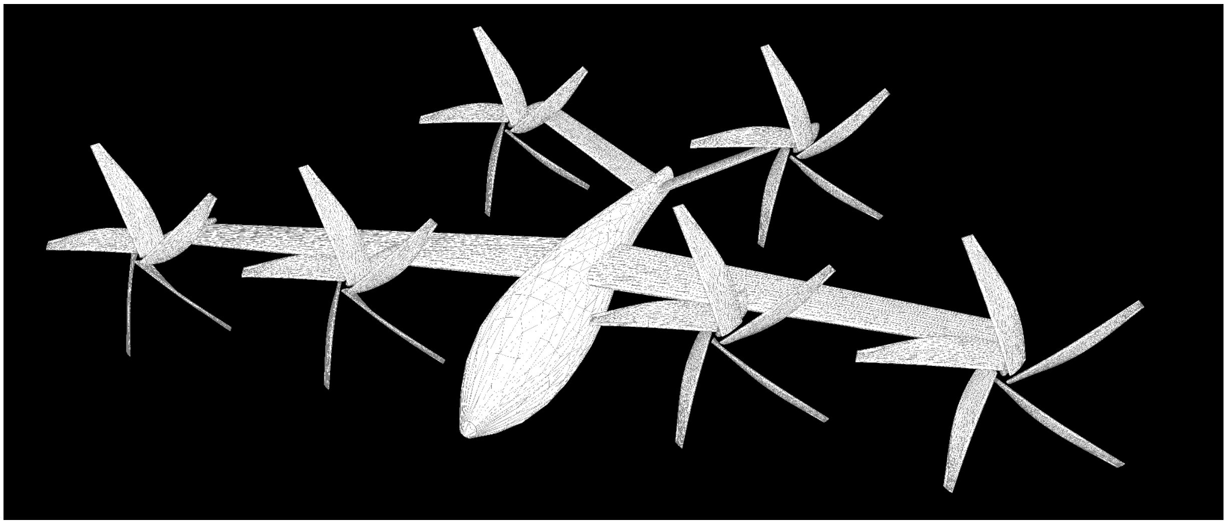

Figure 5.1: Initial aircraft geometry modeled after an urban air mobility vehicle. The nacelles and rotors are ignored during analysis and optimization.

For this setup, three optimizations were run. All of them optimized drag via wing twist

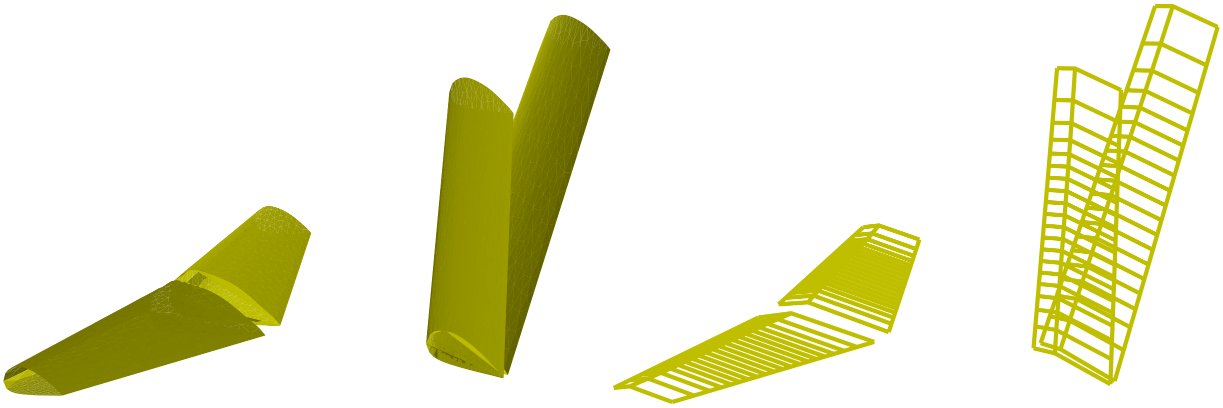

Figure 5.2: Initial wing and tail lifting surfaces (left) and VLM meshes (right).

Table 5.1: A list of design variables and their design spaces, parameters, and objective and constraint functions for Test 1 of the CAD-free implementation.

Variable	Value	Lower Bound	Upper Bound	Size
Objective Function:				
Lift induced drag				1
Design Variables:				
Wing twist distribution		-3°	8°	5
Tail twist distribution		-3°	8°	5
Parameters:				
Angle of attack	4.1°			
Altitude	2.44 km			
Inflow speed	77.2 m/s			
Payload	400 kg			
Battery mass	700 kg			
Battery x-position	3.8 m			
Sectional wing parameters	*			
Sectional tail parameters	*			
Constraints:				
Coef of lift		.5	.5	1
Net moment		0	0	3
Total design variables				10
Total constraints and objectives				5

distribution while maintaining a C_L of .5 and a net moment of 0. Additionally, the wing and tail are mirrored, so parameters affecting one side also affects the other. The first test had the twist distribution of both the wing and the tail as design parameters. The second test had the twist distribution of the wing and the trim of the tail as design parameters. The third test had the twist distribution of the wing, the trim of the tail, and the angle of the aircraft as design parameters. The bounds of the design parameters, along with the fixed values of all other parameters, can be found in Table 5.1.

Results

Table 5.2 contains the resulting design parameters and optimized value of each test. It makes sense that Test 1 would have the lowest drag because it was given the largest number of

Table 5.2: The optimized twist distribution, design parameters, and objective function results of the three test cases.

Variable Name	Test1	Test2	Test3
Lift induced drag (N)	412	416	415
Tail twist 1 (°)	2.65	NA	NA
Tail twist 2 (°)	1.17	NA	NA
Tail twist 3 (°)	0.11	NA	NA
Tail twist 4 (°)	0.00	NA	NA
Tail twist 5 (°)	0.00	NA	NA
Angle of attack (°)	NA	NA	1.61
Tail trim (°)	NA	4.10	3.05
Wing twist 1 (°)	-2.60	4.43	2.45
Wing twist 2 (°)	-1.96	2.15	0.08
Wing twist 3 (°)	5.21	8.00	8,00
Wing twist 4 (°)	-2.30	3.22	0.36
Wing twist 5 (°)	2.71	6.00	5.22

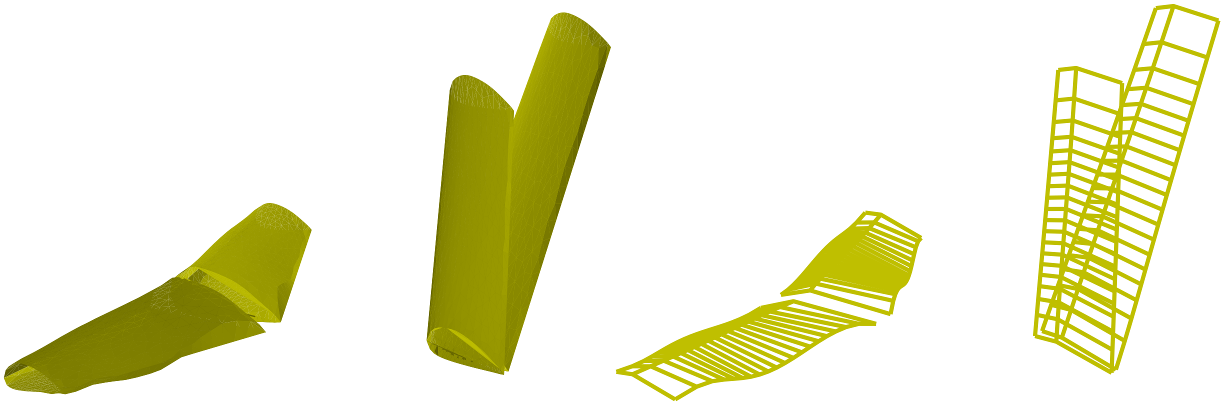

Figure 5.3: Optimized lifting surfaces (left) and VLM meshes (right) with twist distribution.

design variables. Additionally, an exaggerated twist distribution for Test 1 lifting surfaces and corresponding VLM meshes can be seen in Fig. 5.3.

Additionally, more focused tests of the method were also performed. The first of these tests was to confirm the smoothness of the method. Fig. 5.4 shows the relationship between a parameter that controls wing x-position and the node representing the leading-edge fuselage intersection in the master-mesh. For both coarse and fine parameter changes, the master-mesh changes smoothly and appropriately with respect to the changes made to the design parameter.

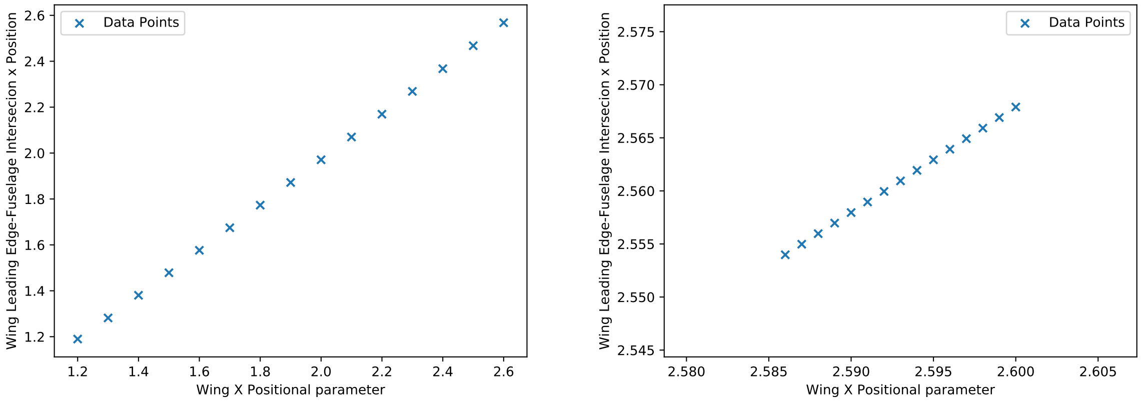

Figure 5.4: Continuity of the master-mesh with respect to coarse (left) and fine (right) adjustments to the wing x-position parameter.

The second focused test, seen in Fig. 5.5, shows an extreme moving of the wing from one iteration to another. The method successfully recomputes the intersection curve and recalculates the master-mesh for the wing. The exact same triangle indices and barycentric coordinates are used to generate the initial and moved wing VLM mesh. This shows how much a design can move while maintaining a consistent master-mesh.

After the initial three tests were performed and presented, an additional fourth test was performed. This test increased the number of twist design variables and included a wing base and tip positional design variables. Although the effect of these additional design variables may be minimal, they demonstrate that the proof-of-concept implementation can handle more drastic design changes. The results of this test are shown in 5.3. And finally, all test cases' convergence history can be viewed in Fig. 5.6.

5.1.2 Conclusion

Even though the CAD tool used is from an in-house framework, it still has all of the relevant characteristics of a commercially available CAD tool. A parametric geometry was taken from the CAD tool and a smooth and continuous master-mesh could be generated for a wide

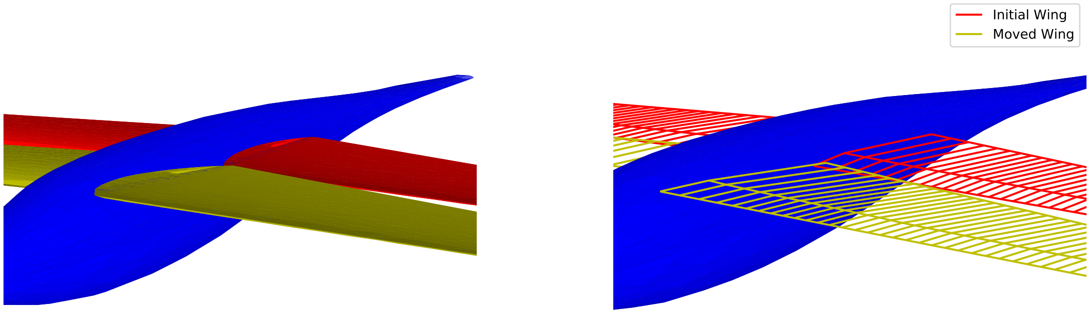

Figure 5.5: Wing-fuselage intersection before and after major wing displacement (left), and corresponding original and regenerated VLM mesh (right).

Table 5.3: Optimization results of a fourth test. Includes new design variables that have a more drastic effect on the design.

Variable Name	Large Fuselage Test
Lift induced drag (N)	427
Tail trim (°)	.009643
Base wing x-pos (m)	3.128
Tip wing x-pos (m)	2.728
Wing twist 1 (°)	8.00
Wing twist 2 (°)	8.00
Wing twist 3 (°)	2.67
Wing twist 4 (°)	-3.00
Wing twist 5 (°)	6.32
Wing twist 6 (°)	6.73
Wing twist 7 (°)	1.19
Wing twist 8 (°)	8.00
Wing twist 9 (°)	3.29
Wing twist 10 (°)	-3.00
Wing twist 11 (°)	5.89

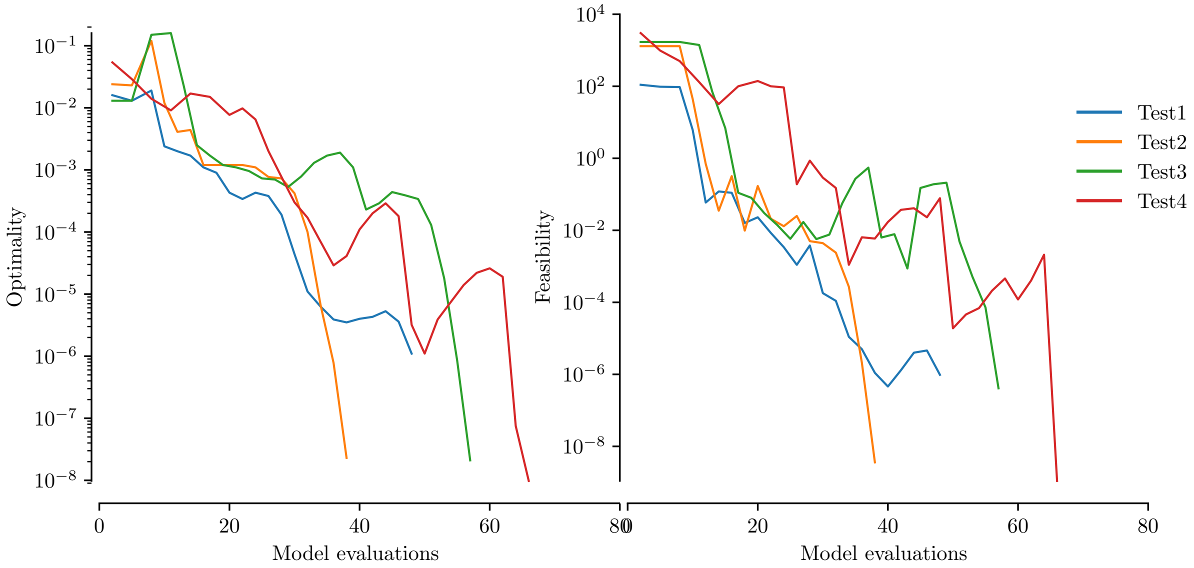

Figure 5.6: Convergence achieved for all four tests in less than 70 model evaluations.

range of design parameters. And, sub-meshes required by various analysis tools were able to be generated from the master-mesh, allowing design sensitivities to be calculated. The same steps can be followed with a commercial CAD tool with only the addition of superficial interfacing code not central to the method.

Ultimately, and most importantly, the application demonstrated a successful gradient-based optimization. As design parameters changed, the master-mesh of the design changed in a smooth manner. Because of this, the finite-difference method could be used to provided accurate gradients without any apparent discontinuities. This meant that the gradient-based optimizer was able to converge efficiently.

An important objective to keep in mind is that the end goal of the master-mesh method is to use it in conjunction with a true, commercial CAD model and perform shape optimization. This means that this application is an incomplete demonstration of the master-mesh method. For one, interoperability with a CAD model needs to be developed. Also, the master-mesh method needs to be made more computationally efficient. This is needed because the method slows down substantially as the number of geometry related design parameters increases, since calculating gradients relies on the finite-difference method. Even with these obstacles, this implementation provides a firm foundation from which to develop a complete master-mesh implementation.

5.2 CATIA V5 Nozzle and CFD Application

The CATIA implementation of the master-mesh method was developed with a single purpose in mind: generate design sensitivities for the CAD tool CATIA V5, which were to be used by the optimization framework and CFD solver FUN3D [5]. However, the CATIA implementation sacrifices an element of the full method to expedite development. The master-mesh generation is simplified and only works for a subset of surfaces. However, it still follows the general master-mesh approach. Additionally, the implementation requires substantially more resources, specifically the CAD tool CATIA V5, the software Pointwise, and processing time on a computing cluster. Pointwise is especially useful as it provided an API to access the information contained within STEP files, a model neutral file format [23]. These resources allowed for an implementation of the alternative master-mesh method to be developed that specifically proviedes design sensitivities for a design parameterized in CATIA V5, but for only a subset of surface types.

5.2.1 Application and Results

The master-mesh implementation used for this applications is the alternative method outlined in Chapter 4. Orginally, the regular triangle smoothing optimization used in the core master-mesh method was used for this application, but it was found to be less computationally efficient. The high-fidelity nature of the CFD analysis and the limited access to the computing cluster made efficiency a premium. Therefore, since the design being optimized only consists of surfaces that can be described as having four sides, the alternative method can be used. The structured quad meshes are generated with the software Pointwise. When tested on a variety of design and surfaces, it always generate a smoothly deforming mesh when design variables changed. This allows for the structured quad mesh to be used as the master-mesh in this application.

Setup

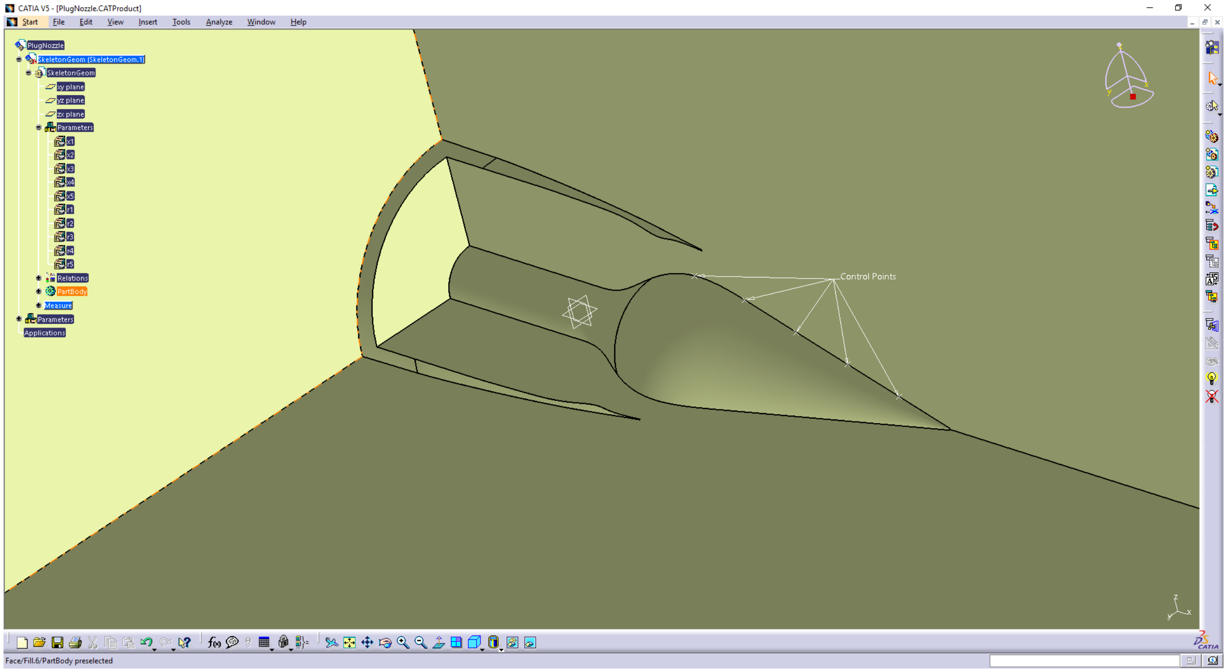

Figure 5.7: The nozzle plug modeled in CATIA V5 and parameterized with 5 control points shown in white.

The test case for this method consists of minimizing drag in a jet nozzle while maintaining a constant flow rate. The initial nozzle was designed in CATIA V5, and the negative space is used to generate the flow field, as shown in Fig. 5.7. Due to the symmetry of the design, only a quarter of the design is modeled. An indirect goal of this optimization is to reduce the noise generated by a boundary-layer shock wave that forms on the nozzle's plug. It is theorized that, in this specific case, drag reduction correlates to noise reduction. This objective will be minimized by altering control points that define the profile of the nozzle's plug. There are seven control points in total, but the two ends are fixed, leaving five control points as design variables. The spline derived from these control points, when revolved about the x-axis, create the outer surface of the plug. Therefore, the only variable controlling each control point is its distance from the x-axis. Another way to pose this would be to say that the radius of various sections of the plug are being treated as design variables.

Next, a STEP file is created from the CATIA model, and then imported into Pointwise. In Pointwise, surface meshes are created, and then an unstructured volumetric mesh is created. FUN3D automatically deforms the volumetric mesh elastically with respect to deformations that occur on the surface meshes. Therefore, only changes in surface meshes need to be tracked via the master-mesh method across iterations. Additionally, a structured mesh is also applied to each design surface, which, in this case, is only the nozzle plug. In this example, the volumetric mesh contained a little over 24 million cells, while both the structured and unstructured mesh applied to the design surface contained roughly 30 thousand nodes.

The fluid dynamics of this model are analyzed using NASA's own computational fluid dynamics model, FUN3D. Before optimization, an initial fluid solve is run in order to obtain a baseline. The results of this baseline are shown in Fig. 5.8. It is predicted that the high turbulence

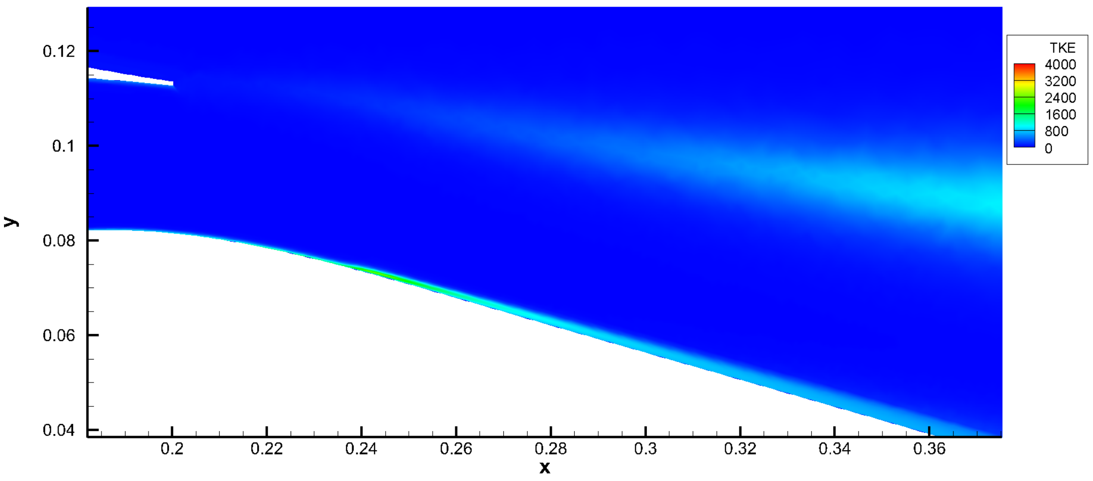

Figure 5.8: Cross section of the unoptimized geometry showing the turbulent kinetic energy as calculated by FUN3D.

Table 5.4: A list of the plug nozzle optimization design variables, parameters, and objective and constraint functions.

Variable	Value	Lower Bound	Upper Bound	Size
Objective Function:				
Nozzle Wall Drag				1
Design Variables:				
Control Point X Position		*	*	5
Control Point Radius		*	*	5
Parameters:				
Reynolds Number	6771100			
Temperature	293.2 K			
Pressure	100367 Pa			
Altitude	2.44 km			
Mach	0.30			
Angle of Attack	0°			
Constraints:				
Nozzle Flow Rate		1.73 kg/s	1.73 kg/s	1
Total design variables				10
Total constraints and objectives				2

along the surface of the plug is what generates a large portion of the noise. Therefore, the objective will be formulated in such a way as to, at least indirectly, reduce the magnitude of the boundary turbulence. Additionally, the fluid flow rate needs to be preserved, so the flow-rate obtained from the initial solve will be used as a constraint for the optimization. The complete formulation is shown in Table 5.4. The parameters given in this table are representative of cruise conditions.

Results

The flow solver used a total of 2250 iterations to calculate the flow values, and a further 2250 iterations to calculate the partials. There were 13 optimization iterations over the course of the 16 hour total run time. The vast majority of that time was spent performing the CFD analysis. Each iteration's results are shown in Table 5.5. Ultimately, the objective was improved by about 4.5 percent, but the constraint remained violated by about 0.7 percent. The constraint probably

Table 5.5: The objective and constraint values of thirteen iterations of the plug nozzle optimization.

Iteration	Objective	Improvement	Constraint	Violation
1	9.0880	0.000%	0.2080	-0.00%
2	8.8057	3.106%	0.2110	-1.45%
3	8.6723	4.574%	0.2094	-0.69%
4	9.0831	0.054%	0.2086	-0.29%
5	8.6976	4.296%	0.2094	-0.65%
6	8.6761	4.532%	0.2094	-0.69%
7	8.6724	4.573%	0.2094	-0.69%
8	8.6723	4.574%	0.2094	-0.69%
9	8.6723	4.574%	0.2094	-0.69%
10	8.6723	4.574%	0.2094	-0.69%
11	9.0833	0.052%	0.2086	-0.29%
12	8.6998	4.271%	0.2093	-0.65%
13	8.6768	4.525%	0.2094	-0.69%

would have been satisfied if the optimization process could have been run longer, but it ran out of allocated time on the Pleiades super computer. The resulting geometry changes can be viewed in Fig 5.9.

The fluid dynamic analysis of the optimized design can be viewed in Figs 5.10,5.11. The first figure shows a mirrored Mach comparison of the two geometries. The unoptimized geometry contains one large shock wave in the fluid flow, while the optimized geometry contains roughly

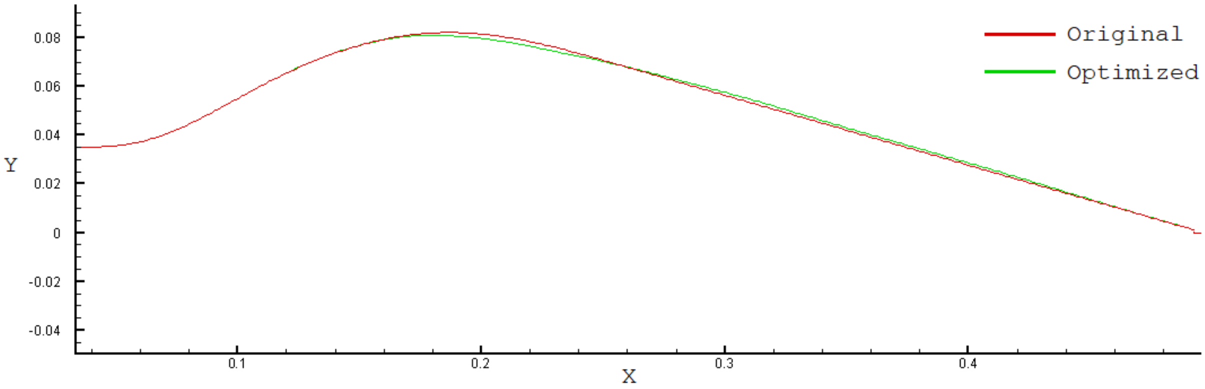

Figure 5.9: A shape comparison of the unoptimized plug profile and the optimized plug profile.

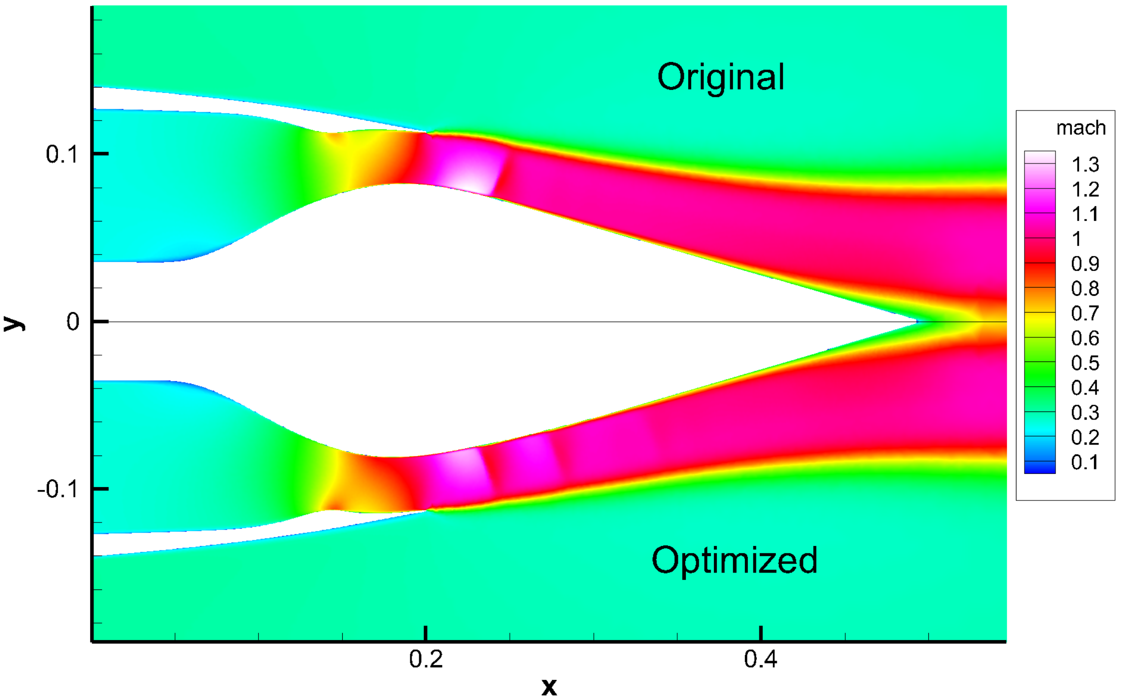

Figure 5.10: A cross sectional comparison of the unoptimized mach field and optimized mach field.

three smaller shock waves. It was also predicted that a decrease in the magnitude of the shock wave might lead to lower turbulent energy on the boundary. The next figure shows the turbulent energy, and, when compared to Fig. 5.8 shows that the turbulent energy did decrease as predicted. The acoustic effects of this, however, cannot be verified by these results; further computational or experimental analysis would be required.

5.2.2 Summary

Ultimately, the goal of this application is to perform gradient-based optimization of a design parameterized in CATIA V5 with FUN3D. The test case demonstrates an incomplete optimization run, but it is a partial gradient-based optimization of a design parameterized in

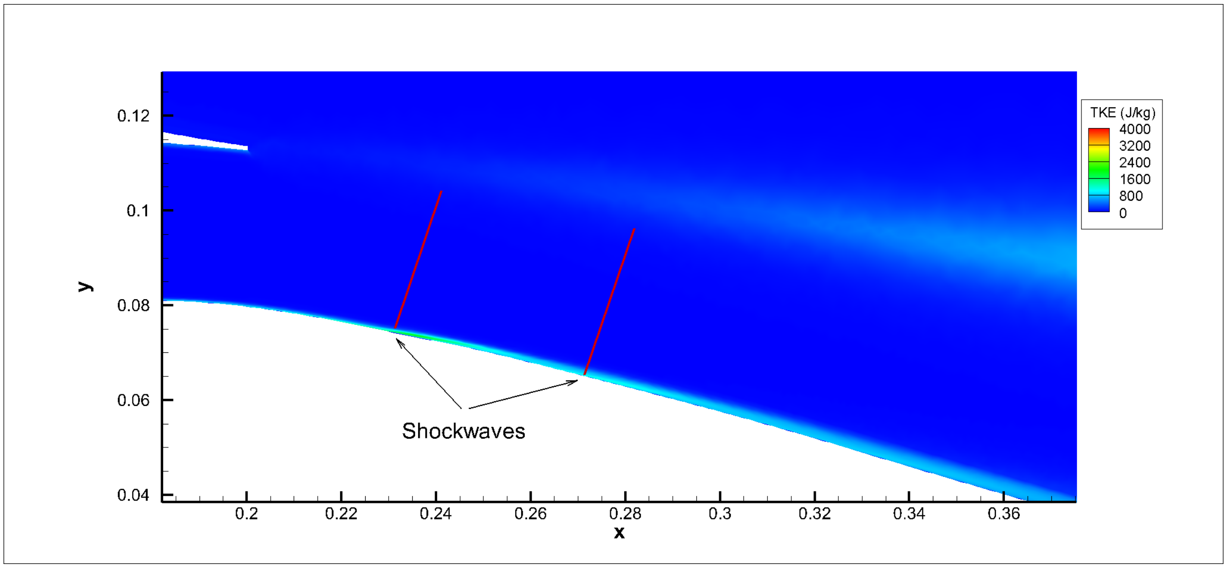

Figure 5.11: The resulting turbulent kinetic energy generated by the optimized plug.

CATIA. Additionally, the parameterization and feature tree of the CAD model is preserved in the optimized geometry. This shows that this implementation is suitable for this specific application. It would be necessary to try the implementation on different designs in order to gauge its robustness and perhaps identify any bugs. However, it is foreseen that any problems will be due to bugs, and not due to any inherent problem with the implementation.

Additionally, this implementation could be further expanded to include other features, such as multidisciplinary analysis. This could take the form of an aerostructural analysis that performs CFD alongside finite element analysis. Other features could take the form of increased usage of parallel computing. The gradients, for example, could be calculated simultaneously as opposed to sequentially when using the finite-difference method. Or, in the case of large CAD designs, the generating of perturbed geometries could also be parallelized.

5.3 FreeCAD Wing and Aerostructural Application

This application consists of an aerostructural optimization of a wing modeled in FreeCAD. The code used for this application was developed as an open-source version of the alternative master-mesh method developed for CATIA in order to make it as accessible as possible. Additionally, the CATIA implementation is designed to be run on computing clusters, while this open-source implementation is designed to be run on standard desktop computers. Because of this, only low-fidelity analysis and simulation tools are tested as opposed to the high fidelity CFD solver used in the CATIA implementation.

As with all of the other implementations, the main objective of this implementation is to compute design sensitivities for CAD parameterized designs. The implementation outlined here follows a very similar process as the CATIA implementation, but forgoes the use of the commercial tools in favor of open-source tools. Using only open-source software will allow this implementation of the method to be open-source and free to distribute. Being open source, this method can be distributed to anyone wanting to do shape optimization on their CAD models. Additionally, it allows users to provide feedback and add functionality to the source code themselves. Ideally, this means that the tool will continue to grow and stay relevant even if I stop working on it.

5.3.1 Application and Results

The master-mesh implementation present in the module is the alternative version outlined in Chapter 4. Gmsh is an open source 3D finite element mesh generator useful for reading geometries from other CAD software in standard exchange formats, such as STEP [11]. The master-mesh is generated by calls to the Gmsh API. First, a STEP file is imported by Gmsh. Then, Gmsh generates a structured mesh for each surface following parameters set by the user. The structured quad mesh serves as the master-mesh for the design. Because the master-mesh is based off of a quad-based structured mesh, every surface has to be defined as having four sides. However, for most surfaces, this can be achieved in the CAD tool by simply adding or removing superficial point and line definitions in the CAD model.

Once the master-mesh is obtained, the sub-mesh reparameterization is obtained using the method outlined in Chapter 4. Even though this master-mesh implementation is a simplified version, it still has the smoothness characteristics of the original implementation. This means that design sensitivities of the sub-mesh can be calculated.

Python Module cadsa

The code utilized for this application is published as the open source Python module cadsa. The first step of computing design sensitivities is interfacing with a CAD tool through its API. Each CAD tool has its own API so one set of functions will not be able to interface with every CAD tool. Therefore, an abstract base class is included in the module that serves as a template for additional CAD API interfacing classes. The module also includes one subclass that integrates with the CAD tool FreeCAD. Additional subclasses will need to be programmed to interface with other CAD tools.

To use cadsa, analysis meshes first need to be created for each design surface. These meshes can be created using any tool and can be made up of any element type with any number of nodes. However, the more nodes within a mesh, the longer it will take to calculate the design sensitivities. Then, a file must be provided that contains a list containing a name, a value, and a step size for each design variable The step sizes are used for the finite-difference method, and the names connect the design variables with the CAD parameters.

Before calculating any sensitivities with cadsa, an initialization script is first run. The initialization script creates necessary connectivity data between the CAD model and the mesh. This information is saved as a Python readable file that can be reused for the same model and mesh. Now, the sensitivities can be calculated. The command requires two input arguments: the directory of the design variable names, values, and step sizes, and the directory of the initialization data. The script updates the CAD model with the design variables provided and calculates the design sensitivity of the updated model using the step sizes provided. Finally, it outputs three files. The updated CAD file, a file containing the updated sub-mesh that contains the same mesh nodes, elements, and layout, and a file containing the X, Y, and Z sensitivities of each mesh node with respect to each design variable. The Jacobian matrix representing these sensitivities will be of *mn* size where *m* is the number of mesh nodes, and *n* is the number of design variables times three. This means that a mesh containing 1000 nodes and a design containing 4 parameters will require 12000 values to represent the design sensitivities.

In the Python module, these design sensitivities can be generated and saved to a file by a single command line call. This allows for the design sensitivity calculation functionality of this module to be called in isolation and used by other software. In addition, this module also provides functionality for optimization with the tool OpenMDAO. In OpenMDAO, an optimization is set up through the use of user defined components whose inputs and outputs are linked together. These components can take the form of a simple function or an entire simulation. This module contains a component wrapper of the CAD API and sensitivity analysis, which essentially allows CAD tools to be plugged into any OpenMDAO-based optimization, gradient-based or gradient-free. The component can be combined with any other OpenMDAO component so long as input design parameters and output analysis mesh connectivity is defined.

Setup

This application features a parameterized half-wing and fuselage model. Fig 5.12 shows the combined wing and fuselage model in FreeCAD, while Fig 5.13 shows only the trimmed wing, modeled as two surfaces. For these examples, design sensitivity calculation will be demonstrated only for the top and bottom surfaces of the wing. The fuselage, however, does serve a purpose by providing a trim curve. The parameters of this model control the wingspan, taper ratio, chord length, and twist distribution.

The optimization test case for this method is the aerostructural analysis of a wing modeled in FreeCAD. The VLM will be used to simulate the forces applied to the wing while a one dimensional finite element model will be used to simulate the structure of the wing. Although

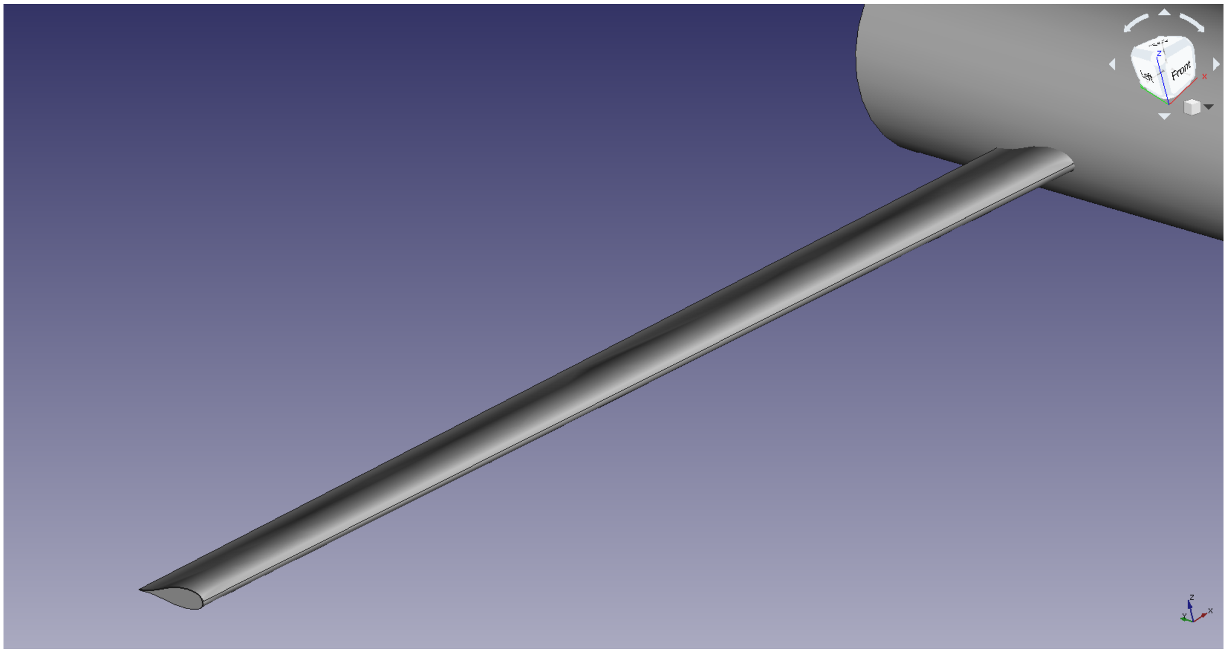

Figure 5.12: The wing and fuselage parametrically modeled in FreeCAD.

these methods are low-fidelity, they can be useful for conceptual design. The low computational requirement of these low-fidelity methods allows a large number of simulations to be run. This allows for a larger design space to be explored in less amount of time when compared to high-fidelity methods. Additionally, these two analyses will demonstrate the effectiveness of the design sensitivities numerically obtained by this method in a multidisciplinary setting. The analysis for this test case is provided by a modified version of OpenAeroStruct, an open-source Python package [18].

The VLM requires that each lifting surface be represented as a grid of infinitely thin quadrilateral panels. For a wing, this generally means that the panels will conform to the camber line of the wing. Since the camber line is not modeled directly, it has to be interpolated from the top surface and the bottom surface of the wing. To do this, a quad mesh will be generated for both surfaces. The averaging of these two quad meshes will provide a single quad mesh approximating the camber line of the wing. The one dimensional finite element model does not require a mesh because it can be analytically obtained with the CAD parameters, specifically the wingspan,

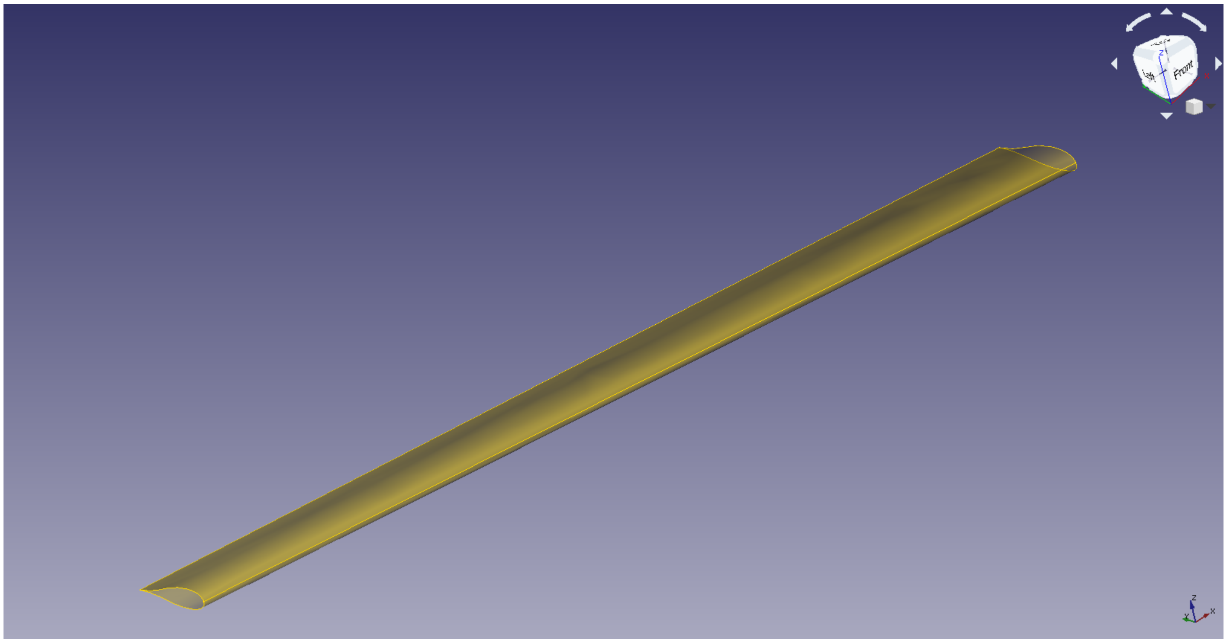

Figure 5.13: The upper and lower surfaces of the wing extracted in FreeCAD.

chord ratio, sweep, and taper ratio. These two models will be tightly coupled: the aerodynamic forces simulated will affect the structural model, and the structural displacements will affect the aerodynamic model. SNOPT will be used to as the master solver and OpenMDAO's built in nonlinear block Gauss-Seidel solver will be used to solve the cyclic connection created by the aerostructural analysis. A list of the design variables, mission parameters, and constraint and objective functions for the aerostructural optimization can be seen in Table 5.6.

Results

The unoptimized aerostructural results are shown in Figure 5.14 and the optimized results are shown in Figure 5.15. Table 5.7 shows the optimized and unoptimized values. A FreeCAD model containing the optimized design parameters was automatically generated. This model is shown in Figure 5.16. This example clearly demonstrates that the design sensitivities calculated by this method are sufficient for gradient-based optimization and maintain CAD connectivity throughout the optimization process. This CAD connectivity will save the engineering hours

Table 5.6: Aerostrucutral wing optimization parameters, design variables, constraints, and objective.

Variable	Value	Lower Bound	Upper Bound	Size
Objective Function:				
Fuel Burn				1
Design Variables:				
Thickness Control Points		.01 m	1 m	3
Angle of Attack		-10°	10°	1
Chord Normalized		-8	20	1
Wingspan Normalized		-8	30	1
Taper Ratio Normalized		-25	5	1
Wing Twist Normalized		-20	20	4
Parameters:				
CL0	0.0			
CD0	0.015			
k laminar	.05			
Thickness to chord ratio	0.15			
Chordwise location of max thickness	.303			
Young's modulus	70e9 Pa			
Shear modulus	30e9 Pa			
Yield stress	500e6 / 2.5 Pa			
Density	3e3 kg/m^3			
Velocity	248.136 m/s			
Mach number	.84			
Dimensionalized Reynolds number	1e6 1/m			
Air density	.36 kg/m^3			
Specific fuel consumption	166e-6 1/s			
Range	11e6 m			
Empty weight	4.5e5 kg			
Speed of sound	295.4 m/s			
Empty weight	4.5e5 kg			
Constraints:				
Lift equals weight		N/A	N/A	1
Von Mises		N/A	Failure Limit	1
Total design variables				10
Total constraints and objectives				3

Table 5.7: The optimized FreeCAD wing design parameters.

Parameter	Optimized Value
Tip Thickness CP	1.00
Midway Thickness CP	1.97
Root Thickness CP	16.3
Chord Normalized	0.71
Wingspan Normalized	-4.68
Taper Ratio Normalized	-25.0
Twist CP 1	19.8
Twist CP 2	0.14
Twist CP 3	-2.48
Twist CP 4	-14.6

typically spent converting a CAD model into an optimizable model and further hours spend converting the optimized model back into a CAD model.

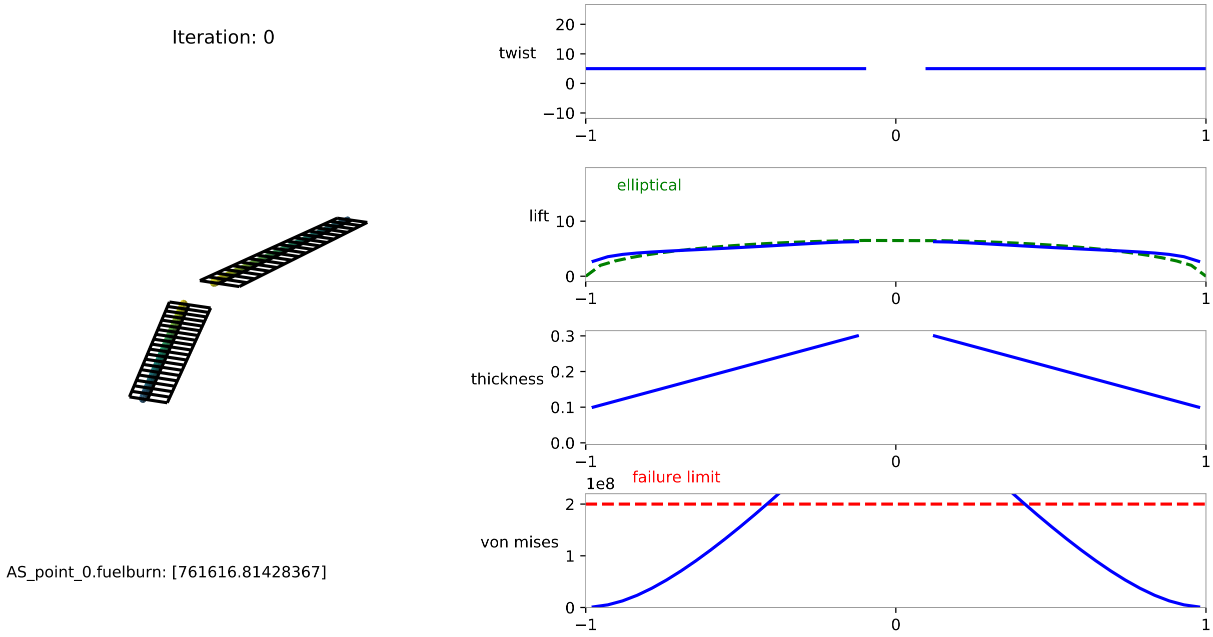

Figure 5.14: Comprehensive aerostructural results of the unoptimized wing.

As previously mentioned, the aerostructural optimization problem is solved using SNOPT within the OpenMDAO optimization framework. Together, these two tools are able to solve nonlinear constrained problems efficiently, so long as accurate gradients are provided. Figure 5.17 shows the convergence history after 64 iterations, and each iteration required design sensitivities

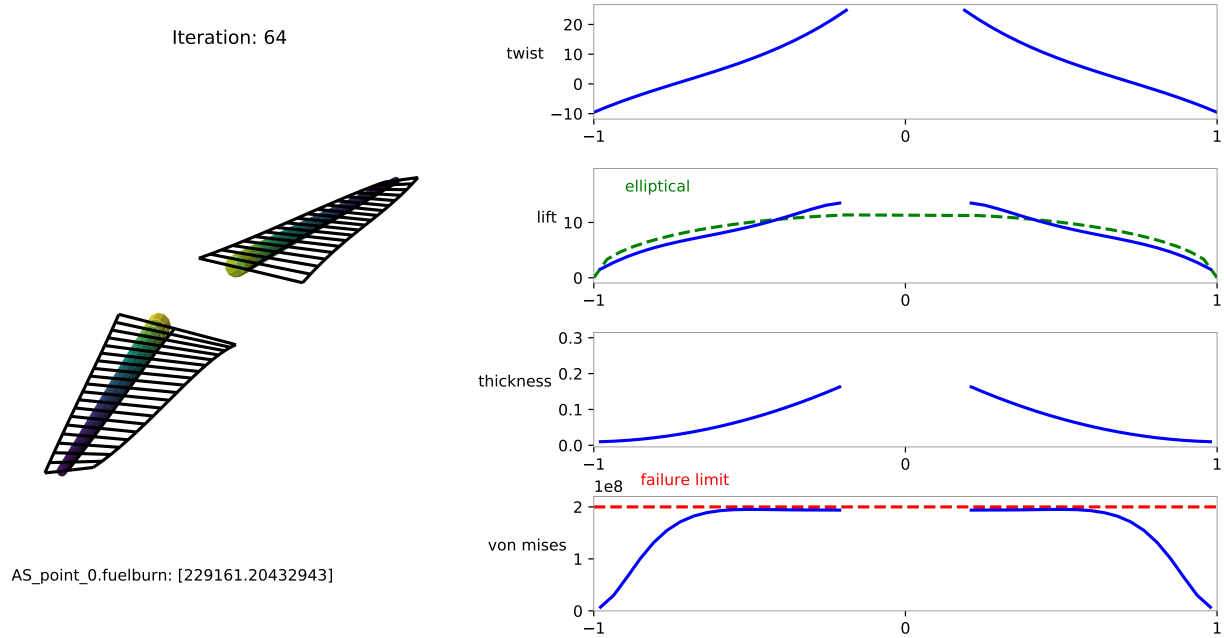

Figure 5.15: Comprehensive aerostructural results of the optimized wing.

to be calculated for the CAD-parameterized model. Computing the design sensitivities on a single processor took roughly 4-4.5 seconds.

5.3.2 Summary

Ultimately, the goal of this application was to demonstrate the master-mesh implementation present in the Python module cadsa was able to provide sensitivity analysis suitable for gradient-based, CAD-parameterized, multidisciplinary, shape optimization. The example shown in Section 5.3.1 clearly demonstrates a successful multidisciplinary optimization. Additionally, CAD connectivity was maintained for the entirety of the optimization, meaning no CAD feature based modeling information was lost. However, one major shortcoming of the test was the limited number of examples tried. Hopefully, now that this package is open source, other users will perform optimization on other models. Additionally, any bugs can be fixed or features can be added by the community as necessitated.

The most immediate source of improvement for this method could be integrating other

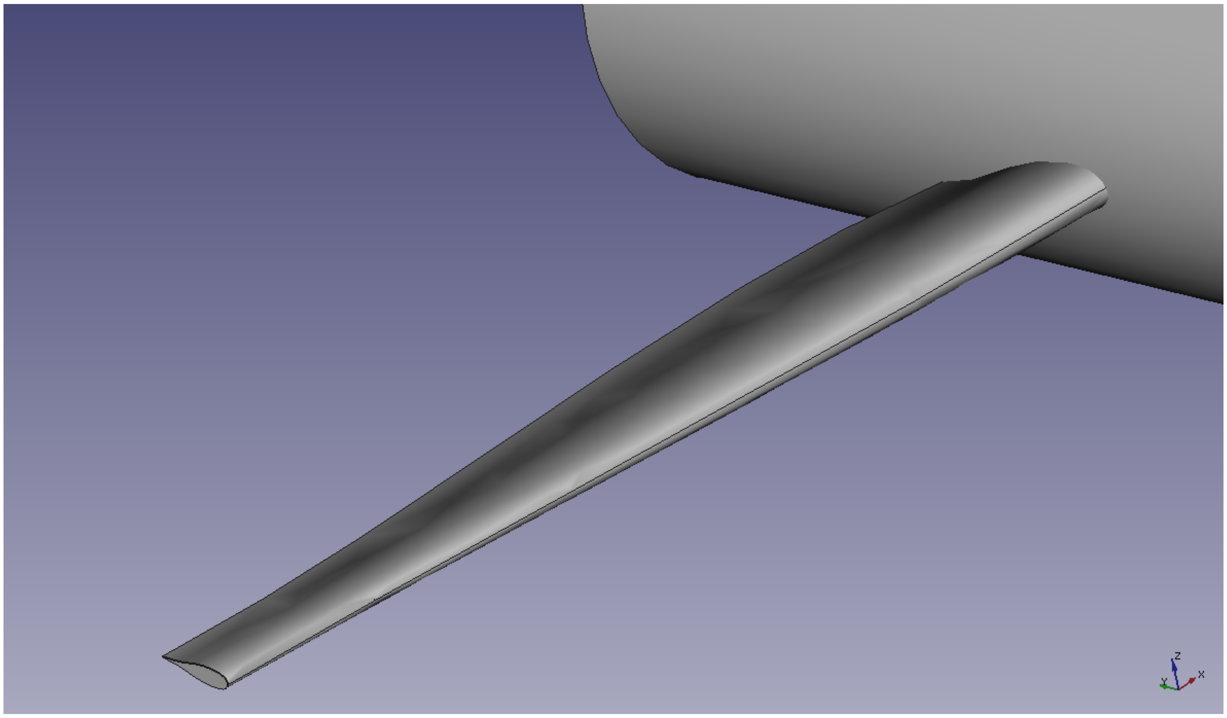

Figure 5.16: The optimized wing loaded in FreeCAD without requiring any model translation and healing.

CAD tools. Unfortunately, this was not possible because no funding was available with which to purchase additional CAD tools. Other features could take the form of parallel computing. The perturbed geometries, for example, could be calculated simultaneously as opposed to sequentially. Another potential improvement for cadsa would be integration into a larger optimization framework for conceptual development. This framework would ideally remove the need for the user provide an initial analysis-mesh of their model. Also, the framework should include a number of built in analysis tools capable of performing gradient-based optimization. Such a framework would be able to rapidly explore a design space.

Chapter 5 Section 1, in part, is a reprint of the material as it appears in Nascenzi, Thomas, Tae H. Ha, and John T. Hwang. "A CAD-interoperable geometry parameterization for large-scale design optimization." AIAA Aviation 2019 Forum. 2019. The thesis author was the primary author of this paper.

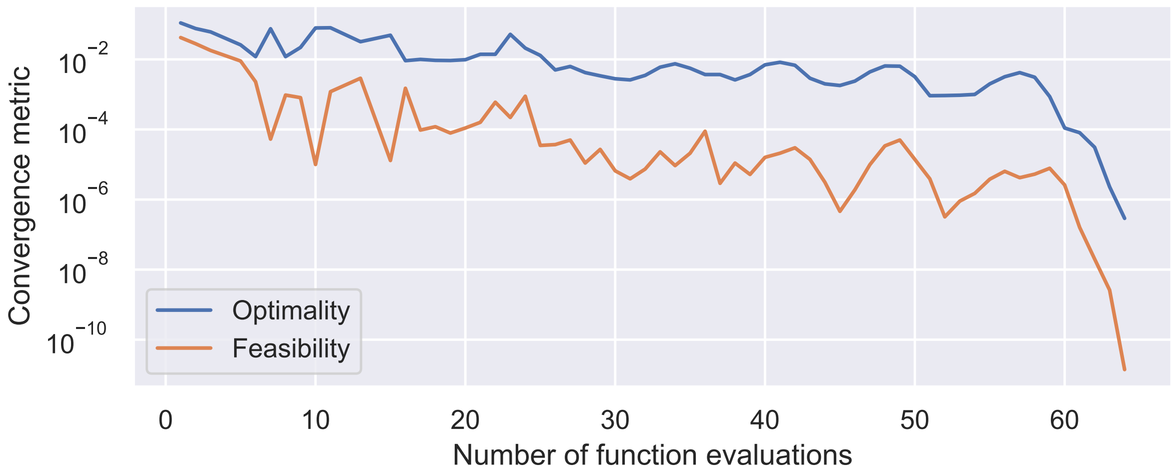

Figure 5.17: Optimality and feasibility reached convergence criteria in 64 iterations.

Chapter 6

Conclusion

6.1 Summary

First, a literature review was made and an opportunity to contribute to the field was identified. CAD-interoperability and gradient-based were identified as desirable characteristics of a new shape optimization method. A method for achieving these qualities was outlined, which revolved around the creation of a master-mesh.

Next, the methodology of the master-mesh was described. The master-mesh needs to maintain smoothness and continuity as CAD design variables change in order to be useful. This is achieved with two optimization methods. The first optimization evenly spaces points along each surface edge. The second optimization then uses the edge points to evenly space the interior nodes of the master-mesh. Analysis meshes can then be projected onto the master-mesh and can be reparameterized in terms of the master-mesh. Therefore, analysis meshes are able to deform smoothly with respect to design variables enabling the use of the finite-difference method to compute design sensitivities necessary for gradient-based optimization.

The method was then demonstrated across three unique applications. Each application featured a unique implementation of the method, and remains useful within the context in which they were made.

The first application of the method served as a CAD-free implementation of the method that demonstrated the feasibility of the master-mesh approach. This application demonstrated that the master-mesh was able to deform smoothly and continuously as design variables changed and could be used for gradient-based optimization. The CAD-free implementation is suitable for optimizing designs parameterized as B-spline surfaces.

The second application of the approach was developed to interface with CATIA V5 and FUN3D. This implementation utilized an alternative method for generating the master mesh, but required that the design surfaces be represented as having four sides. Additionally, the CATIA V5 implementation showed that the master-mesh approach is suitable for high-fidelity shape optimization. The second implementation was successfully tested on an unbiased design created by a third-party, not a design created specifically for optimization testing.

Finally, a third implementation was developed as an open-source version of the CATIA V5 implementation. This method was published as the Python module cadsa. Although cadsa wasn't tested with high-fidelity optimization, it was proven to handle low-fidelity, multidisciplinary, and gradient-based optimization. Additionally, it is more accessible and broadly applicable than the previous two applications.

6.2 Significance and Future Work

Together, these three applications verified the approach while identifying its potential uses and shortcomings. A complete implementation of the method that utilizes the more general implementation of the master-mesh used in the CAD-free application combined with the CAD interoperability of the other two methods, has yet to be realized. Additionally, a major flaw of all methods is that they require that designs maintain a constant topology as design variables change. A topology agnostic implementation of the master-mesh approach would be a significant contribution. A second flaw of this method is that it requires that the CAD design be well-posed or parameterized with optimization in mind. One potential solution to this problem is to develop a CAD pre-processing method that would automatically add parameters to a CAD model to redefine or expand the design space.

However, its novelty and main strength lies in its ability to perform gradient-based shape optimization while maintaining a CAD-parameterized design. Even though the design sensitivities had to be calculated computationally as opposed to analytically, the computationally obtained design sensitivities were proven to be usable for both high-fidelity and low-fidelity optimization. Ultimately, these characteristics make the approach ideal for performing optimization on a conceptual design modeled with a commercial CAD tool. The method would be of most use if packaged with additional conceptual design optimization tools that make optimization even more convenient for an engineer.

Bibliography

[1] Dheeraj Agarwal, Trevor T Robinson, Cecil G Armstrong, Simão Marques, Ilias Vasilopoulos, and Marcus Meyer. Parametric design velocity computation for cad-based design optimization using adjoint methods. *Engineering with Computers*, 34(2):225–239, 2018.

[2] Mladen Banović, Orest Mykhaskiv, Salvatore Auriemma, Andrea Walther, Herve Legrand, and Jens-Dominik Müller. Algorithmic differentiation of the open cascade technology cad kernel and its coupling with an adjoint cfd solver. *Optimization Methods and Software*, 33(4-6):813–828, 2018.

[3] Alan H Barr. Global and local deformations of solid primitives. In *Readings in Computer Vision*, pages 661–670. Elsevier, 1987.

[4] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[5] Robert T Biedron, Jan-Reneé Carlson, Joseph M Derlaga, Peter A Gnoffo, Dana P Hammond, William T Jones, Bil Kleb, Elizabeth M Lee-Rausch, Eric J Nielsen, Michael A Park, Christopher L. Rumsey, James L. Thomas, Kyle B. Thompson, and William A. Wood. Fun3d manual: 13.5. 2019.

[6] Malcolm IG Bloor and Michael J Wilson. Efficient parametrization of generic aircraft geometry. *Journal of Aircraft*, 32(6):1269–1275, 1995.

[7] Vincent Braibant and Claude Fleury. Shape optimal design using b-splines. *Computer methods in applied mechanics and engineering*, 44(3):247–267, 1984.

[8] William Brock, Chad Burdyshaw, Steve Karman, Vincent Betro, Bruce Hilbert, Kyle Anderson, and Robert Haimes. Adjoint-based design optimization using cad parameterization through capri. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 968, 2012.

[9] Richard Lawson Campbell. *An approach to constrained aerodynamic design with application to airfoils*, volume 3260. National Aeronautics and Space Administration, Office of Management . . . , 1992.

[10] John Dannenhoffer and Robert Haimes. Using design-parameter sensitivities in adjoint-based design environments. In *55th AIAA Aerospace Sciences Meeting*, page 0139, 2017.

[11] Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering*, 79(11):1309–1331, 2009.

[12] Justin S. Gray, John T. Hwang, Joaquim R. R. A. Martins, Kenneth T. Moore, and Bret A. Naylor. OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization. *Structural and Multidisciplinary Optimization*, 2019. In Press.

[13] Justin S Gray, Charles A Mader, Gaetan KW Kenway, and Joaquim RRA Martins. Modeling boundary layer ingestion using a coupled aeropropulsive analysis. *Journal of Aircraft*, 55(3):1191–1199, 2018.

[14] Andreas Griewank and Andrea Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*, volume 105. Siam, 2008.

[15] Robert Haimes and John Dannenhoffer. The engineering sketch pad: a solid-modeling, feature-based, web-enabled system for building parametric geometry. In *21st AIAA Computational Fluid Dynamics Conference*, page 3073, 2013.

[16] Jason E Hicken and David W Zingg. Aerodynamic optimization algorithm with integrated geometry parameterization and mesh movement. *AIAA journal*, 48(2):400–413, 2010.

[17] Raymond M Hicks and Preston A Henne. Wing design by numerical optimization. *Journal of Aircraft*, 15(7):407–412, 1978.

[18] John P Jasa, John T Hwang, and Joaquim RRA Martins. Open-source coupled aerostructural optimization using python. *Structural and Multidisciplinary Optimization*, 57(4):1815–1827, 2018.

[19] Gaetan Kenway, Graeme Kennedy, and Joaquim Martins. A cad-free approach to high-fidelity aerostructural optimization. In *13th AIAA/ISSMO multidisciplinary analysis optimization conference*, page 9231, 2010.

[20] Zhoujie Lyu, Gaetan KW Kenway, and Joaquim RRA Martins. Aerodynamic shape optimization investigations of the common research model wing benchmark. *AIAA Journal*, 53(4):968–985, 2015.

[21] Zhoujie Lyu, Zelu Xu, and JRRA Martins. Benchmarking optimization algorithms for wing aerodynamic design optimization. In *Proceedings of the 8th International Conference on Computational Fluid Dynamics, Chengdu, Sichuan, China*, volume 11, 2014.

[22] RM PlCKETT JR, MF Rubinstein, and RB Nelson. Automated structural synthesis using a reduced number of design coordinates. *Aiaa Journal*, 11(4):489–494, 1973.

[23] Michael J Pratt. Introduction to iso 10303—the step standard for product data exchange. *Journal of Computing and Information Science in Engineering*, 1(1):102–103, 2001.

[24] Trevor T Robinson, Cecil G Armstrong, Hung Soon Chua, Carsten Othmer, and Thorsten Grahs. Optimizing parameterized cad geometries using sensitivities based on adjoint functions. *Computer-Aided Design and Applications*, 9(3):253–268, 2012.

[25] Jamshid Samareh. Multidisciplinary aerodynamic-structural shape optimization using deformation (massoud). In *8th Symposium on Multidisciplinary Analysis and Optimization*, page 4911, 2000.

[26] Jamshid A Samareh. Status and future of geometry modeling and grid generation for design and optimization. *Journal of Aircraft*, 36(1):97–104, 1999.

[27] Jamshid A Samareh. Survey of shape parameterization techniques for high-fidelity multidisciplinary shape optimization. *AIAA journal*, 39(5):877–884, 2001.

[28] Uwe Schramm and Walter D Pilkey. Optimal shape design for thin-walled beam cross-sections. *International journal for numerical methods in engineering*, 37(23):4039–4058, 1994.

[29] Thomas W Sederberg and Scott R Parry. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160, 1986.

[30] Matthew Tomlin and Jonathan Meyer. Topology optimization of an additive layer manufactured (alm) aerospace part. In *Proceeding of the 7th Altair CAE technology conference*, pages 1–9, 2011.

[31] Anh H Truong, David W Zingg, and Robert Haimes. Surface mesh movement algorithm for computer-aided-design-based aerodynamic shape optimization. *AIAA Journal*, 54(2):542–556, 2016.

[32] Anil Yildirim, Justin S Gray, Charles A Mader, and Joaquim Martins. Aeropropulsive design optimization of a boundary layer ingestion system. In *AIAA Aviation 2019 Forum*, page 3455, 2019.