SKALIERBARE AERODYNAMISCHE FORMOPTIMIERUNG DURCH EINE MODULARE FRAMEWORK-ANBINDUNG EINES LEGACY CODES

A. Zuger[†‡], Th. Backhaus[‡]

[†] Universität Stuttgart, Keplerstraße 7, 70174 Stuttgart, Deutschland
[‡] MBDA Deutschland GmbH, Hagenauer Forst 27, 86529 Schrobenhausen, Deutschland

Kurzfassung

In der industriellen Praxis prägen etablierte Prozesse, spezialisierte Werkzeuge und proprietäre Softwarelösungen seit Jahrzehnten die aerodynamische Analyse und Optimierung. Viele dieser Systeme sind technisch ausgereift und liefern verlässliche Ergebnisse, weisen jedoch Einschränkungen bei der Flexibilität und der Integration in moderne, automatisierte Entwicklungsumgebungen auf. Besonders herausfordernd ist der Umgang mit sogenannten Legacy-Codes – historisch gewachsenen, häufig monolithischen Programmen, die ursprünglich für manuelle und disziplinär getrennte Analysen entwickelt wurden. Ihre Anpassung an vernetzte, multidisziplinäre Workflows ist oft nur mit erheblichem Aufwand möglich.

Diese Arbeit beschreibt die Entwicklung eines gradientenbasierten Optimierungsrahmens für Low-Fidelity-Aerodynamik, der einen bestehenden internen Legacy-Löser direkt einbindet. Durch die Implementierung einer definierten Programmierschnittstelle (API) wird dieser ehemals abgeschlossene Code nahtlos mit dem Python-basierten Prozess-Framework OpenMDAO der NASA gekoppelt. Dadurch werden sowohl die Transparenz als auch die Wiederverwendbarkeit und Erweiterbarkeit des Codes erheblich verbessert. Gleichzeitig entsteht die Möglichkeit, den Solver in hierarchisch aufgebauten Optimierungsprozessen einzusetzen – von einfachen Einzelpunktanalysen bis hin zu komplexen Multi-Point- und Multi-Objective-Szenarien.

Die Leistungsfähigkeit des entwickelten Frameworks wird exemplarisch anhand einer generischen Geometrie demonstriert. Im Fokus steht die Optimierung der äußeren Formgebung (Outer Shape), wobei auf Basis von Low-Fidelity-Modellen signifikante Verbesserungen erzielt werden. Der Ansatz ermöglicht schnelle Entwurfsiterationen bei geringen Rechenressourcen, was den Entwicklungszyklus spürbar verkürzt. Darüber hinaus zeigt die Arbeit, dass eine gezielte Modernisierung bestehender Softwarelösungen eine nachhaltige Alternative zur vollständigen Neuentwicklung darstellen kann – insbesondere, wenn bestehendes Know-how und bewährte Rechenkerne weiter genutzt werden sollen.

Damit leistet die Arbeit einen praxisnahen Beitrag zur Weiterentwicklung industrieller Entwicklungsprozesse. Sie zeigt einen Weg auf, wie traditionelle Werkzeuge durch gezielte Modernisierung in zukunftsfähige, automatisierte Optimierungsumgebungen überführt werden können, um den steigenden Anforderungen an Effizienz, Integration und Innovationsgeschwindigkeit im Bereich der aerodynamischen Auslegung gerecht zu werden.

Keywords

Aerodynamische Optimierung; Framework-Integration; Low-Fidelity Aerodynamik; Legacy Code Einbindung

doi: 10.25967/650158

NOMENKLATUR

Formelzeichen & Symbole

α	aerodynamischer Anstellwinkel
β	aerodynamischer Schiebewinkel
γ	aerodynamischer Rollwinkel
ζ	Ruderausschlagwinkel für Roll-Kommando
η	Ruderausschlagwinkel für Nick-Kommando
ξ	Ruderausschlagwinkel für Gier-Kommando
ρ	Dichte
$\mathbf{a}$	aerodynamische Größen
C_{f^*}	integraler Kraftbeiwert
C_{Lift}	integraler Auftriebsbeiwert
C_{m^*}	integraler Momentenbeiwert
$\mathbf{D}$	Design Variable
f	Zielfunktion der Optimierung
g	Erdbeschleunigung
i	Indize für Design Variable
j	Indize für Flight Point Nummer
m	Masse
Ma	Machzahl
n_D	Anzahl der Design Variablen
N	Anzahl der Flight Points
R	Residualvektor
S_{ref}	Referenzfläche
T	Trimvektor
v	Geschwindigkeit
$\mathbf{X}$	Geometrieparameter

Hoch- und Tiefstellungen

$(..)_\alpha$	Ableitung nach α
$(..)_\eta$	Ableitung nach η
$(..)_{cg}$	Schwerpunkt
$(..)_i$	Indize für Design Variable
$(..)_j$	Indize für Flight Point Nummer
$(..)_{state}$	Flugzustandsparameter
$(..)_{sys}$	Steuerparameter
$(..)^{max}$	maximaler Wert
$(..)^{min}$	minimaler Wert
$(..)^*$	vorgegebener Wert

Abkürzungen

API	Application Programming Interface
CFD	Computational Fluid Dynamics
DoE	Design of Experiments
FD	Finite-Differenzen
FP	Flight Point
GMC	Generic Missile Configuration
MDAO	Multidisciplinary Design Analysis and Optimization
VLM	Vortex Lattice Method
Alt	Flughöhe
CER	Control Effectiveness Ratio
LF	Load Factor
SM	Static Margin

1. EINLEITUNG

1.1. Motivation

In der industriellen Luft- und Raumfahrtentwicklung ist die aerodynamische Auslegung zentral. Viele Unternehmen arbeiten jedoch weiterhin mit historisch gewachsenen, monolithischen Legacy-Codes, die zwar zuverlässig und ausgereift sind, sich aber nur schwer in heutige automatisierte, multidisziplinäre Workflows integrieren lassen. Anpassungen erfordern erheblichen Aufwand und seltenes Expertenwissen, was Modernisierung bremst und Innovationszyklen verlängert.
Diese Arbeit adressiert genau dieses Problem: Ein bestehender interner Aerodynamik-Löser wird über eine API so erschlossen, dass er sich als Baustein in ein flexibles Prozess-Framework wie *OpenMDAO* [1] integrieren lässt. Dadurch entsteht eine modulare, erweiterbare Struktur, die von Einzelpunktanalysen bis zu Multi-Point- und Multi-Objective-Optimierungen skaliert und reproduzierbare, effizientere Prozesse ermöglicht.
Für frühe Entwurfsphasen wird bewusst ein Low-Fidelity-Ansatz gewählt: Obwohl weniger detailgenau als hochauflösende CFD, erlauben solche Modelle sehr kurze Rechenzeiten und damit schnelle Iterationen [2], bevor in späteren Phasen höherwertige Modelle zum Einsatz kommen [3].
Die vorgeschlagene Methodik zeigt, wie bewährte Inhouse-Werkzeuge durch gezielte Modernisierung in moderne Optimierungsumgebungen überführt werden können — ohne gewachsene Strukturen aufzugeben — und schlägt so eine Brücke zwischen datengetriebener Produktentwicklung und industrieller Softwarepraxis.

1.2. Anwendungsszenarien

Obgleich sich die konkreten Aufgabenstellungen in der aerodynamischen Auslegung von Flugkörpern je nach Anwendungsfall unterscheiden, weisen viele Szenarien in ihrer methodischen Grundstruktur Gemeinsamkeiten auf. Zentraler Ausgangspunkt ist in nahezu allen Fällen die Definition des sogenannten Flight Envelope – jenes Parameterraums, innerhalb dessen ein Flugkörper zuverlässig und effizient betrieben werden kann. Dieser wird im Allgemeinen durch die Kombination aus Geschwindigkeit und Flughöhe beschrieben, wobei für jede mögliche Kombination die aerodynamische Leistungsfähigkeit, Stabilität und Steuerbarkeit nachzuweisen ist.
Im Rahmen dieser Arbeit erfolgt die methodische Untersuchung anhand einer abstrahierten Referenzkonfiguration, die als Generic Missile Configuration 3 (GMC3) bezeichnet wird, siehe Abb. 1, sowie im Anhang in Abb.11.

Die hier vorgestellten Analysen demonstrieren exemplarisch, wie flugmechanische Kenngrößen in einen Optimierungsprozess integriert werden können. Dabei werden sowohl Einzelpunktoptimierungen, die sich auf einen spezifischen Betriebszustand fokussieren, als auch sogenannte Multi-Point Ansätze berücksichtigt, bei denen mehrere repräsentative Flugzustände simultan in die Optimierung einbezogen werden.

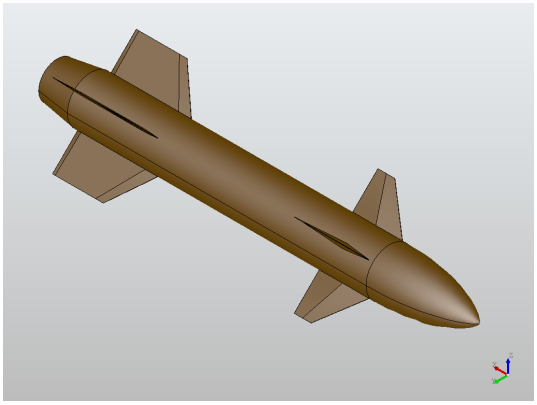

Abb. 1. Generic Missile Configuration 3 (GMC3)

1.3. Flugmechanische Zielgrößen

Um die aerodynamische Leistungsfähigkeit eines Flugkörpers zu bewerten, müssen zunächst die wesentlichen Einflussgrößen definiert werden. Dazu gehören einerseits die geometrischen Eigenschaften des Flugkörpers und andererseits die aktuellen Flugbedingungen (Flight Conditions) wie Geschwindigkeit und Flughöhe.
Darüber hinaus sind flugmechanische Zustandsgrößen wie Anstellwinkel α, Schiebewinkel β und Rollwinkel γ, sowie Steuerflächenausschläge für die Nick- (η), Gier- (ζ) und Rollebene (ξ) entscheidend [4]. Sie wirken direkt auf Kräfte und Momente und bestimmen damit Fluglage, Stabilität und Steuerverhalten. Die Stabilität, im Folgenden Static Margin (SM), ist definiert als

$$SM = \frac{\partial C_{my}}{\partial \alpha} \Big/ \frac{\partial C_{fz}}{\partial \alpha} \tag{1}$$

(aus [5]) mit der statischen Stabilität

$$\frac{\partial C_{my}}{\partial \alpha}, \tag{2}$$

und dem Auftriebsgradient

$$\frac{\partial C_{fz}}{\partial \alpha}. \tag{3}$$

Das Steuerverhalten (Control Effectiveness Ratio, CER) des Systems ist

$$CER = \frac{\partial C_{my}}{\partial \alpha} \Big/ \frac{\partial C_{my}}{\partial \eta}, \tag{4}$$

aus [4]. Die Ableitungen des Momentenbeiwertes C_{my} werden im Folgenden in verkürzter Schreibweise dargestellt: $C_{my,\alpha}$, $C_{my,\eta}$.
Ebenfalls ist das Lastvielfache, der Load Factor (LF), relevant. Dieser ist definiert als:

$$LF = \frac{1}{2} \cdot \frac{C_{Lift} \cdot S_{ref} \cdot \rho \cdot v^2}{m \cdot g}. \tag{5}$$

In einer modularen Betrachtungsweise, wie sie in Multidisciplinary Design Analysis and Optimization (MDAO) [6] verfolgt wird, steht nicht die konkrete Analysemethode im Vordergrund, sondern die Integration von Simulationsbausteinen. Für die aerodynamische Auslegung bedeutet dies, dass eine gegebene Geometrie für definierte Flugzustände so bewertet wird, dass der vollständige Kraft- und Momentenhaushalt bestimmt werden kann.

2. FAMEWORK-EINBINDUNG

Die aerodynamische Auslegung profitiert in zunehmendem Maße von einer systematischen Einbindung in multidisziplinäre Optimierungsumgebungen. Um die Komplexität der beteiligten Parameter, Zielfunktionen und Nebenbedingungen in einem konsistenten Rahmen abbilden zu können, ist der Einsatz geeigneter Frameworks erforderlich. Diese ermöglichen die Integration heterogener Werkzeuge, unterstützen eine standardisierte Datenhaltung und erlauben die Durchführung automatisierter, reproduzierbarer und skalierbarer Optimierungsprozesse.
In der wissenschaftlichen Praxis existieren unterschiedliche Ansätze, wie beispielsweise *GEMSEO* [7], *OpenMDAO* [1] oder vergleichbare Systeme. Für die vorliegende Arbeit wurde das von der NASA entwickelte Framework *OpenMDAO* gewählt, da es die flexible Kopplung von Modellen unterschiedlicher Fidelitätsstufen gestattet, sowohl gradientenfreie als auch gradientenbasierte Optimierungsverfahren unterstützt und durch den Anschluss externer Solver- und Optimierungsbibliotheken eine hohe methodische Bandbreite abdeckt.
Ein weiterer zentraler Aspekt besteht in der Unterstützung verteilter und paralleler Rechenprozesse, wodurch insbesondere auf Hochleistungsrechnern (HPC-Systemen) disziplinübergreifende Analysen simultan durchgeführt werden können [8]. Dies ist von besonderer Relevanz, wenn bestehende Legacy-Codes in großskalige Optimierungsprozesse integriert werden sollen.
Im Folgenden werden methodische Schritte erläutert, die für die Einbettung des in dieser Arbeit verwendeten aerodynamischen Legacy-Codes notwendig sind: (1) die Implementierung einer Legacy-Code-Schnittstelle (Abschnitt 2.1), sowie (2) die Methodische Integration und die Formulierung exemplarischer Optimierungsszenarien (Abschnitt 2.2).

2.1. Legacy-Code-API

Der im Rahmen dieser Arbeit eingesetzte Aerodynamik-Code stellt einen über viele Jahre hinweg entwickelten und gepflegten firmenspezifischen Rechenkern dar. Das zugrunde liegende firmenspezifische Tool – im Folgenden *SEAPT* genannt – verarbeitet geometrische Definitionen, Flugzustände (Geschwindigkeit, Flughöhe, Anstellwinkel, Schiebewinkel, Rollwinkel) sowie Ruderausschläge und liefert als Ergebnis integrale Kraft- und Momentenbeiwerte zurück. Diese Ausgangsgrößen bilden die Grundlage für die aerodynamische Bewertung.
Zur Modernisierung wurde eine standardisierte Programmierschnittstelle (Application Programming Interface, API) implementiert, die eine direkte Kopplung mit der Programmiersprache *Python* ermöglicht. Dies gestattet eine strukturierte Verwaltung der Eingabepa-

rameter in Form von Python-Dictionaries sowie deren erweiterte Ein- und Ausgabefähigkeit auf hierarchische YAML-Strukturen. Parallel dazu erfolgt die Auswertung der Simulationsergebnisse in Pandas DataFrames [9], wodurch eine speicherinterne, effiziente Verarbeitung sowie eine flexible Anbindung an nachgelagerte Analyse- und Optimierungsmodule gewährleistet wird. Dabei bleibt der zentrale Rechenkern des Codes ein unveränderlicher Blackbox-Anteil, der für jeden Simulationslauf separat ausgeführt werden muss. Die Aufgabe der API liegt folglich in der transparenten Kapselung der Eingabe-Ausgabe-Prozesse sowie in der Bereitstellung der Ergebnisse in standardisierter, weiterverarbeitbarer Form. Das so entstandene Werkzeug wird im Folgenden als *AutoSEAPT* bezeichnet und bietet die Möglichkeit, den bestehenden Aerodynamik-Code sowohl unter Windows- als auch unter Linux-Systemen auszuführen und stellt somit eine plattformübergreifende Lösung dar. Im Kontext der vorliegenden Arbeit nimmt das Modul *AutoSEAPT* die Rolle des zentralen Aerodynamikbausteins innerhalb der OpenMDAO-Struktur ein.

2.2. Methodische Integration und Optimierungsszenarien

Zur Demonstration der Funktionsweise des Frameworks sowie zur Bewertung der Leistungsfähigkeit des eingebundenen Legacy-Codes *AutoSEAPT* werden drei Optimierungsszenarien betrachtet.
Die Ergebnisse des Aerodynamikmoduls *AutoSEAPT* – insbesondere Kraft- und Momentenbeiwerte – werden an nachgelagerte Module weitergegeben, die u. a. Trim-Zustände und zulässige Lastvielfachbereiche (Load Factor, LF) bestimmen. Diese hierarchische Anordnung gewährleistet, dass aus aerodynamischen Ausgangsgrößen direkt missionstaugliche Kenngrößen entstehen, die sowohl als Zielfunktionen als auch als Nebenbedingungen in die Optimierung eingehen. Externe Vorgaben wie der Flight Envelope treten als übergeordnete Randbedingungen auf, während disziplinäre Teilprozesse (z. B. Geometrie, Schwerpunkt, Nebenbedingungen) über ein Driver-Modul zur Shape-Optimierung koordiniert werden. Der grundlegende Prozess ist anhand des Prozessflussdiagrammes der expliziten Single-Point Optimierung in Abb. 2 zu erkennen.
Einleitend ist festzuhalten, dass für alle Szenarien eine Abtastung des Flight Envelope auf Basis von DoE-Verfahren (Design of Experiments) erfolgt. Hierdurch werden repräsentative Flugzustände definiert, die mit *AutoSEAPT* automatisiert berechnet werden können. Die resultierenden Datensätze dienen als Grundlage für Analyse und Optimierung.

Im Folgenden werden die drei Optimierungsszenarien beschrieben.

Szenario 1 – explizite Single-Point Optimierung
Das erste Szenario stellt die methodisch einfachste Form der Optimierung dar und dient als Referenzfall. Es handelt sich hierbei um eine explizite Single-Point Optimierung, d. h. die Optimierung erfolgt an einem einzelnen, definierten Flugzustand.
Die freie Steuerung liegt vollständig beim Optimierer: Parameter wie Anstellwinkel α, Schiebewinkel β oder Rollwinkel γ sowie Steuerflächenausschläge (z.B. Höhenruder) werden direkt als Optimierungsvariablen behandelt. Der Optimierer kann diese Größen kontinuierlich variieren, um ein vordefiniertes Ziel – beispielsweise die Minimierung des Widerstands oder die Maximierung des Auftriebs-zu-Widerstands-Verhältnisses – zu erreichen, vgl. Abb. 2.
Die Nebenbedingungen werden in diesem Szenario explizit in den Optimierungsprozess integriert. Dazu zählen insbesondere das momentenfreie Fliegen (Trim-Bedingung, vgl. [10]) sowie die Einhaltung eines

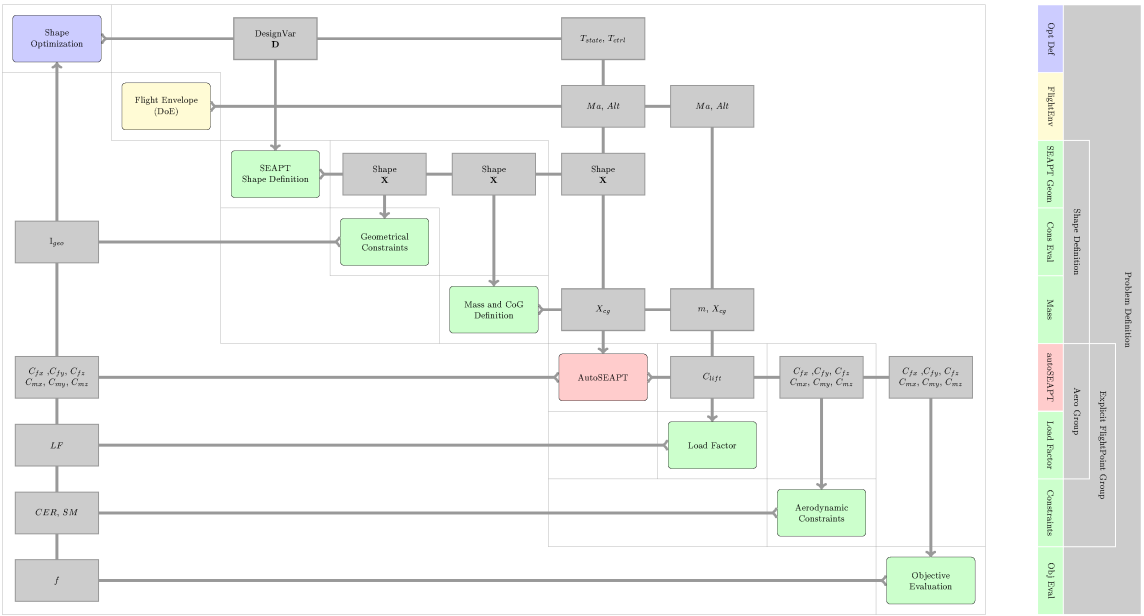

Abb. 2. Szenario 1: Prozessflussdiagram des Gesamtsystems der expliziten Single-Point Optimierung

vorgegebenen Lastvielfaches. Beide Randbedingungen werden dem Optimierer in Form harter Nebenbedingungen zur Verfügung gestellt. Dies gewährleistet, dass die gefundene Lösung physikalisch konsistent ist und den geforderten Betriebszustand realistisch abbildet.
Durch das explizite Einbinden aller Nebenbedingungen steht dem Optimierer ein großer Designraum zur Verfügung, indem dieser das Design variieren kann.

Szenario 2 – implizite Single-Point Optimierung
Das zweite Szenario baut auf dem zuvor beschriebenen Ansatz auf, geht jedoch einen Schritt weiter in Richtung Automatisierung. Es handelt sich um eine implizite Single-Point Optimierung, bei der die zuvor explizit formulierten Nebenbedingungen bereits vorab in den Lösungsprozess integriert werden, vgl. [11].
Zentral ist hierbei die sogenannte Eintrimmung auf einen vorgegebenen Lastvielfachwert LF^*. Der Lastvielfachwert ist typischerweise durch Anforderungen der Flight Envelope oder durch missionsspezifische Randbedingungen definiert. Mittels eines Newton-Lösers wird dieser Wert auf einen passenden Anstellwinkel abgebildet. Dieser interne Trim-Prozess stellt sicher, dass der Flugkörper im betrachteten Zustand das geforderte Lastvielfache erreicht. Darüber hinaus wird über zusätzliche Trim-Loops gewährleistet, dass der Flugkörper sowohl in der Nick- als auch in der Gier- und Rollebene momentenfrei steuerbar bleibt. In diesem Zusammenhang werden die erforderlichen Steuerflächenauslenkungen automatisch bestimmt, sodass der betrachtete Flugzustand aerodynamisch konsistent ist. Der Optimierer erhält demnach keine ungetrimmten Flug- und Kontrollzustände, sondern lediglich die bereits eingetrimmten Zustände, welche die physikalischen Randbedingungen erfüllen.
Die in Abb. 3 dargestellte Trim-Gruppe ist so aufgebaut, dass die aerodynamischen Parameter über den Aerodynamik-Solver berechnet und anschließend zur Erfüllung mehrerer Gleichgewichtsbedingungen verwendet werden. Im vorliegenden Fall werden die Trim-Bedingungen über den Lastvielfachwert (LF) sowie die Momentengleichungen (C_{mx}, C_{my}, C_{mz}) formuliert.

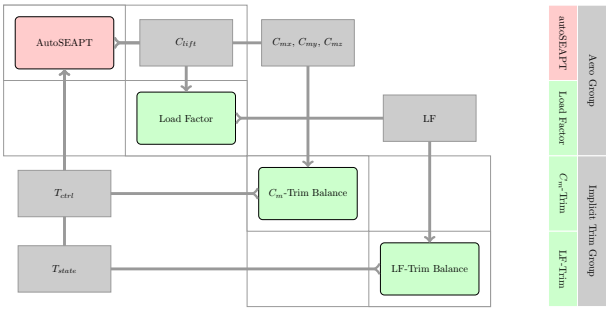

Abb. 3. Szenario 2: Prozessflussdiagramm der Trim-Gruppe

Zur Einstellung dieser Größen stehen zwei Arten von Trim-Parametern zur Verfügung: die Flugzustandsparameter

$$T_{\text{state}} = \begin{bmatrix} \alpha & \beta & \gamma \end{bmatrix}^T, \tag{6}$$

sowie die Steuerparameter

$$T_{\text{ctrl}} = \begin{bmatrix} \eta & \xi & \zeta \end{bmatrix}^T. \tag{7}$$

Zusammen bilden sie den vollständigen Vektor der Freiheitsgrade

$$T = \begin{bmatrix} T_{\text{state}} & T_{\text{ctrl}} \end{bmatrix}^T = \begin{bmatrix} \alpha & \beta & \gamma & \eta & \xi & \zeta \end{bmatrix}^T. \tag{8}$$

Die Residuen des Trim-Problems werden durch die Abweichungen der aerodynamischen Größen von ihren Sollwerten gebildet:

$$R(T) = \begin{bmatrix} LF - LF^* \\ C_{mx} - C^*_{mx} \\ C_{my} - C^*_{my} \\ C_{mz} - C^*_{mz} \end{bmatrix} = \begin{bmatrix} \Delta LF^* \\ \Delta C^*_{mx} \\ \Delta C^*_{my} \\ \Delta C^*_{mz} \end{bmatrix} \stackrel{!}{=} 0. \tag{9}$$

Hierbei beschreibt LF^* den vorgegebenen Lastvielfachwert (z. B. $LF^* = 1$ für den stationären Horizontalflug), während C^*_{mx}, C^*_{my} und C^*_{mz} die Sollwerte der Momentengleichgewichte darstellen (typischerweise $C^*_{mx} = 0, C^*_{my} = 0, C^*_{mz} = 0$). Der Residualvektor R fasst somit alle Trim-Bedingungen zusammen, die im Gleichgewichtszustand erfüllt sein müssen.
Die Sensitivitäten der Residuen bezüglich der Trim-Parameter ergeben die Trim-Jacobi-Matrix:

$$\frac{\partial R}{\partial T} = \begin{bmatrix} \frac{\partial \Delta LF^*}{\partial \alpha} & \cdots & \frac{\partial \Delta LF^*}{\partial \zeta} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Delta C^*_{mz}}{\partial \alpha} & \cdots & \frac{\partial \Delta C^*_{mz}}{\partial \zeta} \end{bmatrix} \tag{10}$$

Die Trim-Balance wird in diesem Rahmen als implizites Residualsystem modelliert. Dadurch entfällt eine explizite Bestimmung der Trim-Parameter, sondern sie ergeben sich automatisch durch die Lösung des Gleichungssystems $R(T) = 0$ innerhalb der impliziten Gruppe. Auf diese Weise wird sichergestellt, dass in jeder Optimierungsiteration ausschließlich getrimmte Flugzustände betrachtet werden.
Die Einbindung der Trim Gruppe in das Gesamtsystem ist in Abb. 18 im Anhang zu sehen.
Die Vorteile dieses Ansatzes sind zweifach: Zum einen wird die Optimierungsaufgabe deutlich vereinfacht, da der Optimierer ausschließlich mit konsistenten Flugzuständen arbeitet. Zum anderen wird die Stabilität des Optimierungsprozesses erhöht, da die implizite Behandlung der Nebenbedingungen die Wahrscheinlichkeit reduziert, dass unphysikalische Zwischenlösungen auftreten. Der Preis dieser Vorgehensweise liegt in der geringeren Transparenz: Da der Optimierer die Trim-Prozesse nicht direkt kontrolliert, wird die Lösungslogik stärker durch das Framework und die Solverarchitektur bestimmt.

Szenario 3 – implizite Multi-Point Optimierung
Das dritte Szenario stellt die methodisch anspruchsvollste Variante dar und entspricht dem industriellen Bedarf nach robusten Entwürfen. Es handelt sich um eine Multi-Point Optimierung, in der nicht nur ein einzelner, sondern mehrere repräsentative Flugzustände gleichzeitig innerhalb der Flight Envelope berücksichtigt werden.
Analog zu Szenario 2 wird auch hier die implizite Trim-Gruppe eingesetzt, um für jeden betrachteten Flugzustand automatisch die erforderlichen Trim-Parameter zu bestimmen. Alle Flugzustände werden konsistent eingetrimmt, bevor ihre Ergebnisse in die Optimierung einfließen.
Jeder dieser Zustände wird durch die Trim-Gruppe in Kombination mit dem Aero-Solver parallel analysiert, und die resultierenden Ergebnisse gehen in eine gemeinsame Zielfunktion ein. Das resultierende Prozessflussdiagramm ist im Anhang in Abb. 19 dargestellt.
Für die aggregierte Zielfunktion aus der gewichteten Summe der Einzelzustände ergibt sich:

$$f_{\text{MultiPoint}}(\mathbf{x}) = \sum_{j=1}^{N} f_j(\mathbf{a}_i(\mathbf{X}, T_j)), \tag{11}$$

wobei f_j die Zielfunktion im Flight Point j beschreibt.
Die Trim-Bedingungen werden für jeden Flugzustand implizit über das erweiterte Residualsystem

$$R_j(T_j) = \begin{bmatrix} LF_j - LF_j^* \\ C_{mx,j} - C_{mx,j}^* \\ C_{my,j} - C_{my,j}^* \\ C_{mz,j} - C_{mz,j}^* \end{bmatrix} = 0, \tag{12}$$

mit $j = 1, \ldots, N$, gelöst. Der Trim-Parametervektor lautet entsprechend

$$T_j = \begin{bmatrix} \alpha_j & \beta_j \, \gamma_j & \eta_j & \xi_j & \zeta_j \end{bmatrix}^T, \tag{13}$$

wobei jeweils sowohl die Zustandsgrößen $(\alpha_j, \beta_j, \gamma_j)$ als auch die Steuergrößen (η_j, ξ_j, ζ_j) enthalten sind. Analog zur impliziten Single-Point Optimierung ergibt sich auch die Trim-Jacobi-Matrix für die Multi-Point Optimierung. Damit ist sichergestellt, dass in jedem betrachteten Flugzustand die Trim-Bedingungen eingehalten werden.
Ziel dieser Methode ist es, eine Geometrie zu finden, die im gesamten Einsatzbereich eine hohe Leistungsfähigkeit sicherstellt. Die zentrale Schwierigkeit liegt darin, die Zielkonflikte zwischen unterschiedlichen Flugzuständen auszugleichen. Die Multi-Point Optimierung zwingt den Optimierer daher, einen ausgewogenen Kompromiss zu finden, der die Gesamtleistung maximiert.
Die Definition der Zielfunktion ist in diesem Szenario frei wählbar aus verschiedenen Parametern der einzelnen Flight Points. So kann beispielsweise der Widerstandsbeiwert während des Cruise-Flugzustandes reduziert werden, während bestimmte Nebenbedingungen beim Start eingehalten werden. Durch diese Flexibilität lässt sich das Optimierungsproblem präzise an die jeweiligen Anwendungsanforderungen anpassen.
Die Multi-Point Optimierung bietet somit die umfassendste, aber auch die rechenintensivste Methodik. Sie erfordert eine große Zahl von Funktionsauswertungen, profitiert jedoch besonders von der Parallelisierungsfähigkeit (s. [1]) des *AutoSEAPT*-Tools in Verbindung mit *OpenMDAO*.

Die drei beschriebenen Szenarien verdeutlichen die methodische Bandbreite der aerodynamischen Optimierung. Während die explizite Single-Point Optimierung eine transparente, leicht verständliche Vorgehensweise darstellt, bietet die implizite Variante eine effizientere Einbindung physikalischer Nebenbedingungen. Die Multipoint-Optimierung schließlich stellt in diesem Rahmen die höchste methodische Stufe dar, da sie eine robuste Gesamtleistung über den gesamten Flight Envelope hinweg sicherstellt.

3. AERODYNAMISCHE DESIGN-OPTIMIERUNG

Im Rahmen dieser Arbeit wird die Funktionsweise des entwickelten Optimierungsrahmens anhand des generischen Beispiels GMC3 (vgl. Abb. 1) demonstriert. Ziel ist es die grundsätzliche Leistungsfähigkeit des Ansatzes in einem kontrollierten Szenario aufzuzeigen.
Dazu werden die flugmechanischen Größen CER und die Static Margin in ein optimales Gleichgewicht gebracht. Dieses vereinfachte Beispiel erlaubt es, die Kopplung des Legacy-Solvers mit dem Optimierungsframework zu testen und zu validieren.
Dazu werden die wesentlichen methodischen Schritte vorgestellt: die genaue Problemdefinition (Abschnitt 3.1), die Einbindung und Nutzung von Gradienteninformationen (Abschnitt 3.2) sowie die Präsentation der Ergebnisse der einzelnen Szeanrios (Abschnitt 3.3, Abschnitt 3.4 und Abschnitt 3.5).

3.1. Problemdefinition

Das generische Beispiel orientiert sich hierbei an einem kleinen Canard-gesteuerten Lenkflugkörper. Im betrachteten Fall wird der Einfachheit halber nur Bewegungen um die Nickachse betrachtet und die Geometrie der Tragflügel der Konfiguration optimiert. Die Flugzustände (Flight Points), sowie die Constraints und das Optimierungsziel werden wie folgt defeiniert:

Flight Points
Die Auswahl der Flight Points orientiert sich an drei maßgebenden Manövern bzw. Auslegungspunkten, die auf einer Mission geflogen werden. Dem Start, dem stationären Horizontalflug (dem Cruise-Zustand) und den engsten Kurvenflug, nachfolgend als Flight Point 1 bis 3 bezeichnet. Die vorgegebenen Flight Points bestehen dabei aus der Machzahl (Ma), der Flughöhe (Alt) und dem Load Factor (LF). Die ausgewählten Werte der unterschiedlichen FlightPoints werden so ausgewählt, sodass der jeweilige Zustand näherungsweise abgebildet wird, s. Tabelle 1.

	FP 1	FP 2	FP 3
Manöver	Start	Kurvenflug	Cruise
Ma	0.6	0.8	0.8
Alt	10	10	10
LF	2.0	5.0	1.0

Tab. 1. Flight Points

Nebenbedingungen und Optimierungsziel
Die Nebenbedingungen bestehen aus den flugmechanischen Größen $C_{m,\alpha}$, $C_{m,\eta}$, der CER und der Static Margin. In der expliziten Optimierung werden zusätzlich noch α, η, C_{my} und der Load Factor beschränkt, vgl. Kapitel 2.2. Ebenfalls werden geometrische Nebenbedingungen gesetzt, sodass sich beispielsweise die Positionen der Flügel nicht überschneiden.
Im vorliegenden Fall wurde das folgende Szenario untersucht: Minimierung der Static Margin bei eingeschränkter CER.
Zur Validierung des entwickelten Frameworks wird zunächst eine Optimierung ohne jegliche Nebenbedingungen durchgeführt. So wird die prinzipielle Funktion des Optimierers überprüft. Zuletzt werden die Nebenbedingungen exakt auf die Zielkonfiguration gesetzt, sodass im finalen Optimierungslauf die resultierende Geometrie für die Zielkonfiguration bestimmt wird und das Einhalten von Nebenbedingungen validiert wird. Die Werte orientieren sich herbei an grundlegenden Auslegungs-Richtlinien aus [4] und [10], sowie Werten aus [12]:
In [10] wird für eine stabile Fluglage gefordert, dass $C_{m,\alpha}$ und $C_{m,\eta}$ negativ sind, während in [4] für einen adäquaten Steuerungsspielraum gefordert wird, dass $|CER| < 1$. Daraus folgt, dass $0 < CER < 1$ gelten muss. Gleichzeitig gilt dann, wie ebenfalls in [4] gefordert, dass $|C_{m,\alpha}| < |C_{m,\eta}|$ ist.

3.2. Gradientendefinition

Das entwickelte Optimierungs-Framework basiert auf einem gradientenbasierten Verfahren, sodass die Berechnung konsistenter Ableitungen zwischen Design-Variablen, Zwischenparametern und Zielfunktionen eine zentrale Rolle spielt. Da weder der eingesetzte Legacy-Code *AutoSEAPT* noch die weiteren expliziten Komponenten Gradienteninformationen nativ bereitstellen, erfolgt die Bestimmung der Ableitungen innerhalb der jeweiligen *OpenMDAO*-Komponenten. Hierbei wird ein Finite-Differenzen-Ansatz verwendet, bei dem die Wahl der FD-Schrittweiten einen erheblichen Einfluss auf die numerische Stabilität und die Robustheit der Lösung hat.
Ausgangspunkt sind die *Design Variables* $\mathbf{D}$, die geometrische Freiheitsgrade beschreiben, z. B. Spannweite, Flügelposition oder diverse Profilparameter. Diese werden in der *SEAPT Shape Definition* in konkrete Geometriegrößen $\mathbf{X}$ überführt. Auf Basis dieser Geometrie werden in der Komponente *Mass and CoG Definition* die Masse m sowie der Schwerpunkt X_{cg} bestimmt:

$$\mathbf{X} = \mathbf{X}(\mathbf{D}), \qquad X_{\mathrm{cg}} = X_{\mathrm{cg}}(\mathbf{X}). \tag{14}$$

Die zugehörigen Sensitivitäten

$$\frac{\partial \mathbf{X}}{\partial \mathbf{D}}, \qquad \frac{\partial X_{\mathrm{cg}}}{\partial \mathbf{X}} \tag{15}$$

werden mittels FD approximiert.
Diese Größen gehen anschließend in die aerodynamische Komponente ein, in der der Solver *AutoSEAPT* aufgerufen wird. Dieser liefert aerodynamische Größen:

$$\begin{aligned} &\mathbf{a} = a(\mathbf{X}, \alpha, \eta, \mathbf{X}_{\mathrm{cg}}), \\ &\mathbf{a} \in \{C_L, C_D, C_{my}, LF, \dots\}. \end{aligned} \tag{16}$$

Die Sensitivitäten der aerodynamischen Größen werden in *OpenMDAO* explizit für mehrere Eingangsgrößen per FD deklariert. Neben den Ableitungen nach den Geometrieparametern X und dem Schwerpunkt X_{cg} werden auch die direkten Gradienten nach dem Anstellwinkel α und dem Ruderwinkel η berücksichtigt:

$$\frac{\partial \mathbf{a}}{\partial \mathbf{X}}, \qquad \frac{\partial \mathbf{a}}{\partial X_{\mathrm{cg}}}, \qquad \frac{\partial \mathbf{a}}{\partial \alpha}, \qquad \frac{\partial \mathbf{a}}{\partial \eta}. \tag{17}$$

Damit ergibt sich für den Gradienten der Zielfunktion $f(\mathbf{a}, \mathbf{X}, X_{\mathrm{cg}}, \alpha, \eta)$ mit Blick auf die Design-Variablen $\mathbf{D}$ folgende Darstellung:

$$\begin{aligned} \nabla_D f = \frac{\partial f}{\partial \mathbf{a}} \cdot \Bigg(&\frac{\partial \mathbf{a}}{\partial \mathbf{X}} \cdot \frac{\partial \mathbf{X}}{\partial \mathbf{D}} + \frac{\partial \mathbf{a}}{\partial X_{\mathrm{cg}}} \cdot \frac{\partial X_{\mathrm{cg}}}{\partial \mathbf{X}} \cdot \frac{\partial \mathbf{X}}{\partial \mathbf{D}} \\ &+ \frac{\partial \mathbf{a}}{\partial \alpha} \cdot \frac{\partial \alpha}{\partial \mathbf{D}} + \frac{\partial \mathbf{a}}{\partial \eta} \cdot \frac{\partial \eta}{\partial \mathbf{D}} \Bigg) + \frac{\partial f}{\partial \mathbf{X}} \cdot \frac{\partial \mathbf{X}}{\partial \mathbf{D}}. \end{aligned} \tag{18}$$

Auf diese Weise entsteht eine konsistente Gradientenkette, die von den Design-Variablen $\mathbf{D}$ über die geometrischen Größen $\mathbf{X}$, die Schwerpunktlage X_{cg} und die aerodynamischen Parameter $\mathbf{a}$ bis hin zur Zielfunktion f reicht.

3.3. Explizite Single-Point Optimierung

Für die explizite Single-Point Optimierung ergibt sich folgendes Optimierungsproblem:

$$\begin{aligned} &\text{minimize} && SM(\mathbf{D}) \\ &\text{w.r.t.} && \mathbf{D}_i, \quad i = 1, \dots, n_D \\ &\text{subject to} && \mathbf{D}_i^{\min} \leq \mathbf{D}_i \leq \mathbf{D}_i^{\max}, \\ & && 0 \;\leq\; CER \;\leq\; 1, \\ & && -2 \;\leq\; SM \;\leq\; 0, \\ & && C_{m,\alpha} \;\&\; C_{m,\eta} \leq 0, \\ & && -30^\circ \;\leq\; \alpha \;\&\; \eta \;\leq\; 30^\circ, \\ & && -0.002 \;\leq\; C_{my} \;\leq\; 0.002, \\ & && -10\% \cdot LF^* \;\leq\; LF \;\leq\; +10\% \cdot LF^*, \end{aligned}$$

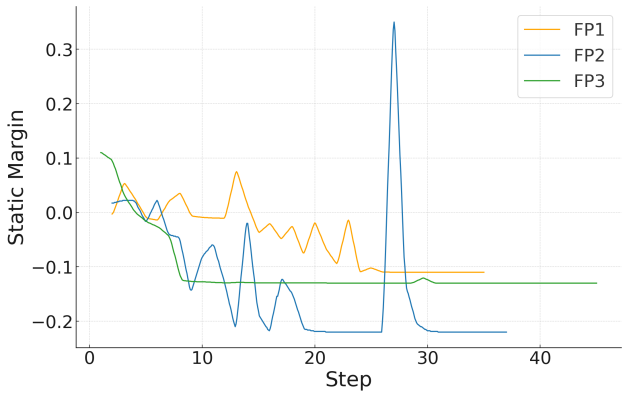

Abb. 5. Evolution der Static Margin in der nicht beschränkten, expliziten Optimierung

mit den Design Variablen $\mathbf{D}_i$.
Zunächst wird die unbeschränkten Optimierung betrachtet. Mittels des Trends der Zielfunktion ist eine deutliche Minimierung zu erkennen, s. Abb. 5. Der Optimierer zeigt das typische Verhalten eines gradientenbasierten Optimierungsverfahrens und seine Funktionsweise ist somit validiert. Ebenfalls ist zu erkennen, dass sich die optimierten Geometrien der verschiedenen Flight Points unterscheiden (s. Abb. 12 im Anhang), unterschiedliche Machzahlen und Lastvielfache die Anforderungen an den Flugkörper verändern. Somit zeigt sich hier die Notwendigkeit eines Multi-Point Optimierungsverfahrens.
Nun werden die Nebenbedingungen auf jene der Zielkonfiguration gesetzt. Diese Begrenzung der Nebenbedingungen führt auf ein Problem, welches durch die ungenaue Gradientenberechnung mittels FD verursacht wird. Bei großer Wahl der FD-Schritte, wird ein Minimum gefunden, welches außerhalb des zulässigen Bereiches liegt. Der berechnete Gradient und dementsprechend die Schrittweite des Optimierers an dieser Stelle, um in den zulässigen Bereich zu gelangen, ist jedoch so groß, dass sich der Wert der Zielfunktion aus dem Minimum heraus bewegt und der Optimierer keine Lösung erkennt (s. Abb. 15 im Anhang). Der Optimierer beendet seine Optimierung mit dem Status Failed mit einem Ergebnis außerhab der Nebenbedingungen. Wählt man die FD-Schrittweite von Beginn an zu niedrig, so findet der Optimierer zu Beginn an ein Minimum, bleibt jedoch in diesem Minimum, da die Schrittweite zu niedrig ist um andere Minima zu erkennen. Um diesem Problem entgegenzuwirken wird zu Beginn ein großer FD-Schritt gewählt und nach Beendigung der Optimierung, eine Optimierungs-Iteration identifiziert, dessen Wert der Zielfunktion innerhalb der Nebenbedingungen ist. Anhand dieser Geometrie und Anstellwinkel, sowie Ruderausschlag wird die Iteration mit einer geringeren FD-Schrittweite neu gestartet. So wird innheralb des zulässigen Bereiches ein Minimum gesucht und gefunden, s. Abb. 16 im Anhang. Die resultierenden Geometrien aller Flight Points sind in Abb. 4 dargestellt.
Diese Geometrien unterscheiden sich stark voneinander, da diese unabhängig voneinander optimiert wurden. Die Notwendigkeit eines Multi-Point Optimierungsverfahrens wird hier somit erneut aufgezeigt.

3.4. Implizite Single-Point Optimierung

Für die implizite Single-Point Optimierung lässt sich das Optimierungsproblem aufschreiben als

$$
\begin{aligned}
&\text{minimize} && SM(\mathbf{D}) \\
&\text{w.r.t.} && \mathbf{D}_i, \quad i = 1, \ldots, n_D \\
&\text{subject to} && \mathbf{D}_i^{\min} \leq \mathbf{D}_i \leq \mathbf{D}_i^{\max}, \\
& && R(T; \mathbf{D}_i) \stackrel{!}{=} 0, \\
& && 0 \leq CER \leq 1, \\
& && -2 \leq SM \leq 0, \\
& && C_{m,\alpha} \leq 0, \\
& && C_{m,\eta} \leq 0.
\end{aligned}
$$

α, η, C_{my} und der Load Factor werden nicht als Constraint behandelt, sondern werden innerhalb $R(T; \mathbf{D})$ auf die geforderten Werte eingetrimmt. Der Designraum wird somit vorab auf jene eingetrimmte Flugzustände begrenzt. Dementsprechend wird erwartet, dass sich die optimierten Geometrien nicht allzu stark unterscheiden.
Zunächst wird erneut eine Optimierung ohne jegliche Nebenbedingungen durchgeführt, um die Funktion des Optimierers zu überprüfen. Analog zur expliziten Single-Point Optimierung ist ein deutlicher Minimierungs-Trend in den Werten der Zielfunktion erkennbar, s. Abb. 7. Die Werte des Load Factors und C_{my} sind dabei exakt, wie nach Vorgabe eingehalten. Die Funktionsweise der Trim-Gruppe ist somit validiert. Auffällig, im Vergleich zum Verlauf der Static Margin im unbeschränkten ex-

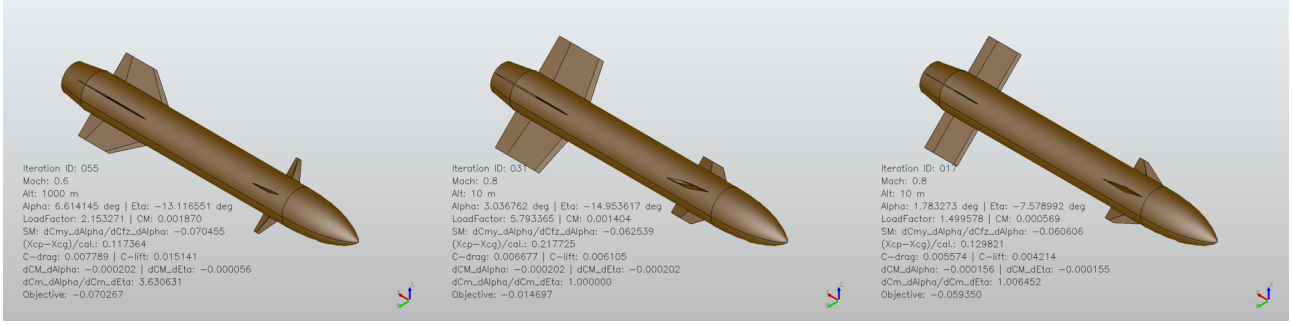

Abb. 4. Optimierte Geometrie der expliziten Optimierung der Zielkonfiguration in FP1 (links) bis FP3 (rechts)

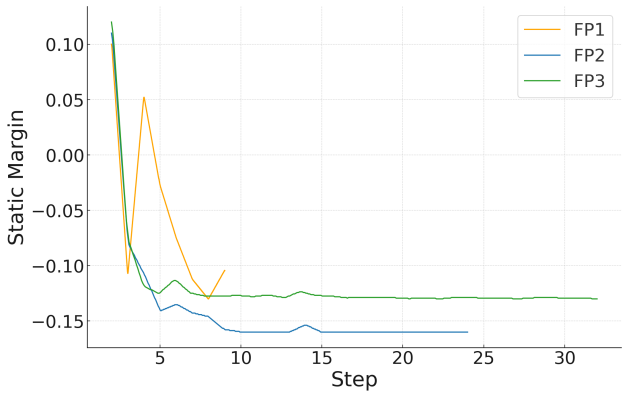

Abb. 7. Evolution der Static Margin in der nicht beschränkten, impliziten Optimierung

pliziten Fall (vgl. Abb. 5), ist dass der Wert der Static Margin nach weniger Iterationen das Minimum erreicht und im Bereich dieses Minimums nur minimal variiert.
Ebenfalls ist in den Geometrien der Optimierungsergebnisse zu erkennen, dass sich diese aufgrund der unterschiedlichen Flugzustände unterscheiden, s. Abb. 13 im Anhang. Jedoch sind diese Unterschiede, wie erwartet, geringer wie in der expliziten Optimierung, da der Designraum aufgrund der Trim-Gruppe eingeschränkt wird.
Bei Setzung der Nebenbedingungen auf die Werte der Zielkonfiguration, wird erneut erkennbar, dass der Optimierer einen Kompromiss, zwischen der Einhaltung der Nebenbedingungen und unter der Evaluierung eines Minimums, sucht. Auffällig hierbei ist, dass die Werte nach weniger Iterationen innerhalb des zulässigen Bereichs sind. Nach wenigen Iterationen sind die Werte der Nebenbedingungen und der Zielfunktion der optimierten Lösung erreicht und der Optimierer variiert die Design Parameter nur noch minimal. Dieses Verhalten ist beispielhaft an dem Konvergenz-Plot des ersten Flight Points erkennbar, s. Abb. 17 im Anhang. Die Geometrien des Optimierungsergebnisses sind in Abb. 6 dargestellt.
Allgemein lässt sich beobachten, dass die Dauer der Optimierung in der impliziten Single-Point Optimierung stark ansteigt, da das Lösen der Trim-Gruppe mittels des Newton-basierten Lösungsalgorithmus sehr zeit- und rechenaufwendig ist. Allerdings nehmen die benötigten Iterationen bis die Nebenbedingungen innerhalb des gewünschten Bereiches sind, im Vergleich zur expliziten Optimierung, ab.

Des Weiteren lässt sich feststellen, dass sich die Geometrien der Optimierungsergebnisse der verschiedenen Flight Points nicht so stark unterscheiden, wie in der expliziten Optimierung. Der Designraum wird durch die vorab gelöste Trim-Gruppe auf eingetrimmte Zustände begrenzt. Die zusätzliche Variation von α und η wird hier nicht in das Optimierungsproblem integriert und diese zusätzlichen Design-Parameter und somit die zusätzlichen Möglichkeiten der Lösung fallen weg. Die verschiedenen Ergebnisgeometrien machen eine Multi-Point Optimierung jedoch weiterhin notwendig.

3.5. Implizite Multi-Point Optimierung

Wie in den vorherigen Single-Point Optimierungen durch die unterschiedliche Ergebnisgeometrie zu erkennen ist, ist die Notwendigkeit eines Multi-Point-Optimierers unerlässlich. Eine gesamte Flight Envelope muss durch mehrere repräsentative stationäre Flight Points abgebildet werden, welche in einem Multi-Point Optimierungsverfahren zusammen berücksichtigt werden müssen. Im vorliegenden Fall wird das Optimierungsziel auf die Static Margin des dritten Flight Points, dem Cruise-Flugzustand, gesetzt. Somit ergibt sich zusammengefasst das Optimierungsproblem zu

$$
\begin{aligned}
&\text{minimize} && SM_3(\mathbf{D}) \\
&\text{w.r.t.} && \mathbf{D}_i, \quad i = 1, \dots, n_D \\
&\text{subject to} && \mathbf{D}_i^{\min} \leq \mathbf{D}_i \leq \mathbf{D}_i^{\max}, \\
& && R_j(T_j; \mathbf{D}_i) \stackrel{!}{=} 0, \quad j = 1, 2, 3, \\
& && 0 \leq CER|_j \leq 1, \\
& && -2 \leq SM_j \leq 0, \\
& && C_{m,\alpha}|_j \leq 0, \\
& && C_{m,\eta}|_j \leq 0,
\end{aligned}
$$

mit den Indizes j, den Flight Points 1 bis 3. Die Nebenbedingungen gelten für jeden Flight Point. Es wird erwartet, dass die Evaluation eines Optimierungsergebnisses schwer, bzw. nicht gelingt, da die Nebenbedingungen zu eng gesetzt sind, um eine Lösung für die gesamte Flight Envelope mit denselben Nebenbedingungen zu finden.
Anhand des Konvergenzplots der Zielfunktion in Abb. 9 kann eine deutliche Minimierung der Static Margin iden-

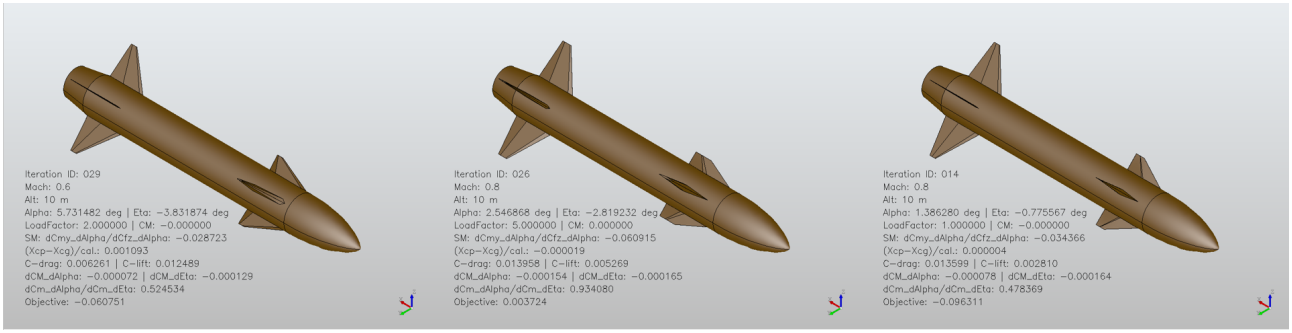

Abb. 6. optimierte Geometrien der impliziten Optimierung der Zielkonfiguration in FP1 (links) bis FP3 (rechts)

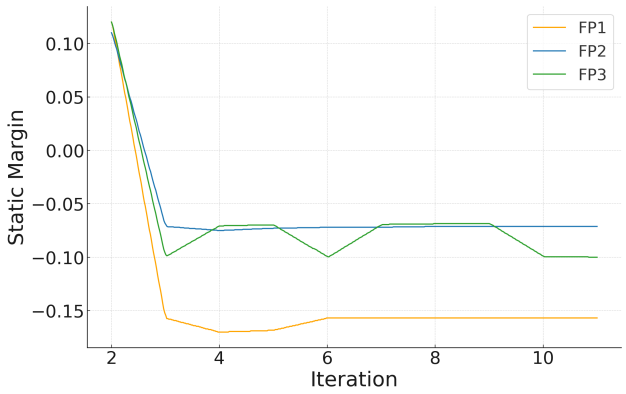

Abb. 9. Evolution der Static Margin in der impliziten Multi-Point Optimierung

tifiziert werden. Die korrekte Implementierung des Multi-Point Modus ist somit validiert. Unter Berücksichtigung aller Ausgabegrößen zu jedem Flight Point wird die Zielfunktion minimiert und eine gemeinsame Geometrie (s. Abb. 14 im Anhang) gefunden. Das Minimum wird hier erneut nach wenigen Iterationsschritten gefunden und die Design Variablen in diesem Bereich nur noch minimal variiert.

In den Konvergenzplots der Multi-Point Optimierung mit den Nebenbedingungen der Zielkonfiguration (s. Abb. 8) ist deutlich zu erkennen, dass eine Evaluation eines Ergebnisses unter Einhaltung aller Nebenbedingungen nur schwierig, bzw. gar nicht möglich ist. Bei Einhaltung der Nebenbedingungen in Flight Point 1, werden diese in den beiden anderen verletzt. Jedoch wird eine Geometrie evaluiert (s. Abb. 10), welche der Zielkonfiguration sehr nahe kommt und als bester Kompromiss für die gesamte Flight Envelope interpretiert werden kann. Interessant ist bei der optimierten Geometrie, dass sich diese von den Geometrien der impliziten Single-Point Optimierung unterscheidet, obwohl diese untereinander nicht so stark variieren, vgl. Abschnitt 3.4. Zu Erwarten wäre eine nur minimale Veränderung dieser Geometrien auf eine, welche sich für alle drei Flight Points gleichzeitig eignet.

4. ZUSAMMENFASSUNG

Die vorliegende Arbeit hat sich mit der Frage befasst, wie langjährig entwickelte industrielle Legacy-Codes, die oftmals als Blackbox-Module mit stark eingeschränkter Transparenz vorliegen, in moderne, mathematisch fundierte Optimierungs- und Prozessframeworks eingebettet werden können. Ausgangspunkt war die Beobachtung, dass in der industriellen Praxis zahlreiche etablierte Werkzeuge zwar eine hohe Reife und Zuverlässigkeit besitzen, ihre Einbindung in automatisierte, reproduzierbare und multidisziplinäre Entwicklungsumgebungen jedoch aufgrund historisch gewachsener Strukturen erheblich erschwert ist. Vor diesem Hintergrund wurde ein methodischer Ansatz entwickelt, um bestehende Softwarebestände systematisch für moderne Optimierungsstrategien zugänglich zu machen.

Im Zentrum der Arbeit stand die Entwicklung einer Programmierschnittstelle, die es erlaubt, den weitgehend verschlossenen inhouse Code *SEAPT* über

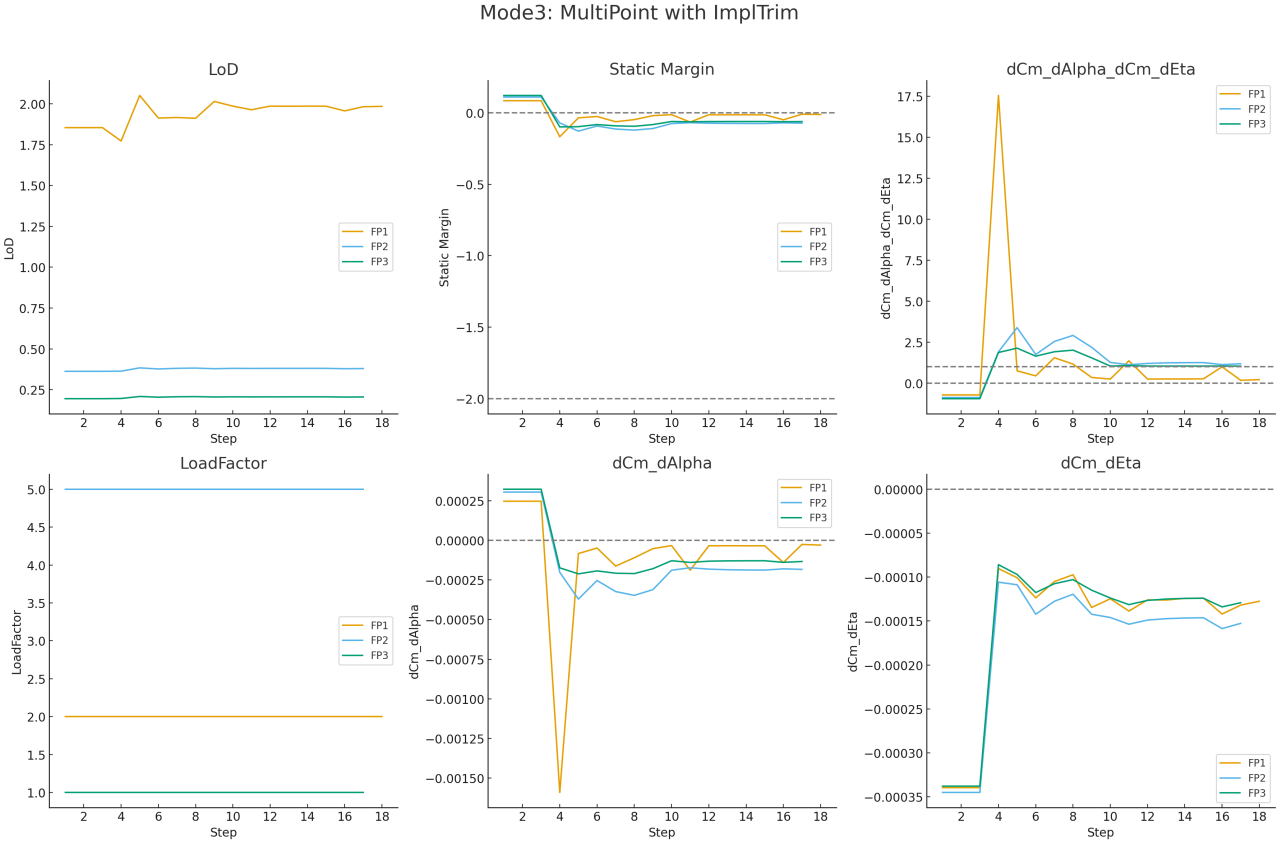

Abb. 8. Konvergenzplot der impliziten Multi-Point Optimierung der Zielkonfiguration

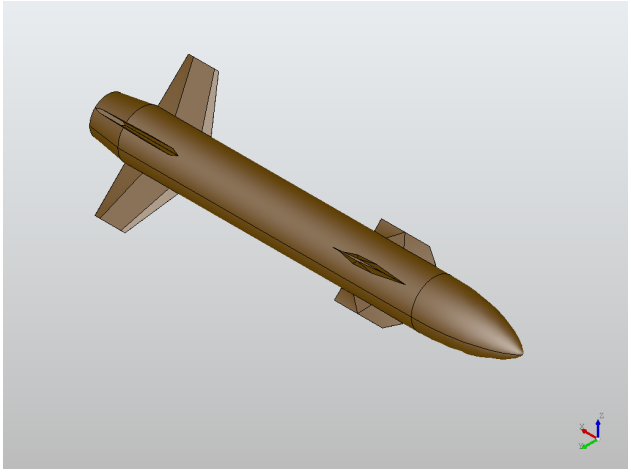

Abb. 10. Optimierte Geometrie der impliziten Multi-Point-Optimierung

das Interface *AutoSEAPT* in das NASA-Framework *OpenMDAO* zu integrieren. Durch die Implementierung standardisierter Schnittstellen konnte eine konsistente Datenhaltung etabliert werden, welche den Übergang von klassischen File-I/O-basierten Prozessen hin zu speicherinternen, automatisierten Workflows ermöglicht. Diese Modernisierung stellte die methodische Grundlage dar, auf der im weiteren Verlauf eine hierarchische Framework-Struktur aufgebaut wurde.
Anhand einer generischen Referenzkonfiguration, der Generic Missile Configuration 3 (GMC3), wurde die praktische Umsetzbarkeit des Konzepts demonstriert. Es konnte gezeigt werden, dass auch ein Blackbox-Code wie *SEAPT*, dessen innerer Rechenkern nicht zugänglich ist, durch geeignete Kapselung in hierarchische Modellstrukturen eingebettet werden kann. Dies ermöglichte es, unterschiedliche Optimierungsszenarien aufzubauen, die jeweils spezifische Anforderungen an die Interaktion zwischen Eingangsparametern, Solvermechanismen und Ergebnisgrößen stellen.
Im methodischen Kern wurden drei Szenarien entwickelt und evaluiert:

- Explizite Single-Point Optimierungen, bei denen der Optimierer direkten Zugriff auf Steuergrößen wie Anstellwinkel oder Ruderauslenkungen erhält und Nebenbedingungen explizit formuliert werden,
- Implizite Single-Point Optimierungen, in denen Trim-Bedingungen über interne Solverprozesse abgebildet und dem Optimierer nur physikalisch konsistente Zustände präsentiert werden, sowie
- Multi-Point Optimierungen, in denen verschiedene Zustände der Flight Envelope simultan berücksichtigt werden, um robuste geometrische Lösungen für ein breites Einsatzspektrum zu identifizieren.

Diese Szenarien zeigen exemplarisch, wie aus schematisch aufgebauten Modellhierarchien konkrete mathematische Problemdefinitionen abgeleitet werden können. Darüber hinaus wurde demonstriert, dass auch unter den Bedingungen eingeschränkter Zugänglichkeit Gradienteninformationen extrahiert und für gradientenbasierte Optimierungen nutzbar gemacht werden können. Damit konnte ein wesentliches Hindernis überwunden werden, das in der industriellen Nutzung von Legacy-Codes häufig als limitierender Faktor angesehen wird.

4.1. Ausblick

Der zentrale Mehrwert der vorliegenden Arbeit liegt in der modularisierten Erweiterbarkeit der vorgestellten Szenariomodelle. Während in industriellen Anwendungen traditionell vor allem isolierte Einzeldisziplinen – wie hier die Aerodynamik – für die Optimierung geometrischer Konfigurationen genutzt werden, eröffnet der hier entwickelte Ansatz die Möglichkeit einer systematischen multidisziplinären Erweiterung.
Ein vorrangiges Ziel zukünftiger Arbeiten ist die Kopplung der aerodynamischen Analyse mit strukturellen Modellen, um die bislang statisch-aerodynamischen Szenarien zu statisch-aerostrukturellen Untersuchungen auszubauen. Auf diese Weise lassen sich Wechselwirkungen zwischen aerodynamischen Lasten und strukturellen Rückwirkungen in ein konsistentes Optimierungsumfeld integrieren. Dies erhöht die Aussagekraft der Ergebnisse erheblich, da die Berücksichtigung struktureller Aspekte entscheidend für die Realisierbarkeit und Belastbarkeit optimierter Konfigurationen ist.
Darüber hinaus eröffnet die vorgeschlagene Methodik die Möglichkeit, die Modularisierung nicht ausschließlich auf den hier eingesetzten Low-Fidelity-Legacy-Code *SEAPT* zu beschränken. Vielmehr ist vorgesehen, die Schnittstellenarchitektur so zu generalisieren, dass auch weitere firmeninterne Softwarewerkzeuge eingebettet werden können – von Panelmethoden über semi-empirische Ersatzmodelle bis hin zu hochaufgelösten CFD-RANS-Simulationen wie bspw. *TAU* [13] oder *CODA* [14]. Damit wird eine skalierbare Multi-Fidelity-Architektur geschaffen, die je nach Verfügbarkeit von Rechenressourcen, Detaillierungsgrad und Fragestellung flexibel eingesetzt werden kann.
Die zukünftige Forschung wird sich folglich auf die Verallgemeinerung und den Ausbau der framework-basierten hierarchischen Modularisierung konzentrieren. Ziel ist die Etablierung einer robusten, erweiterbaren Plattform, die es erlaubt, unterschiedliche Disziplinen und Simulationsmethoden in einheitlicher Form miteinander zu koppeln. Durch diese konsequente Modularisierung entsteht eine methodische Basis, die nicht nur die hier demonstrierten aerodynamischen Optimierungsszenarien abdeckt, sondern auch den Weg für umfassende multidisziplinäre Design- und Optimierungsprozesse eröffnet.

Literatur

[1] J.S. Gray, J.T. Hwang, J.R.R.A. Martins, K.T. Moore, and B.A. Naylor. OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization. In *Struct. Multidiscipl. Optim*, 2019.

[2] E.J. Adler, A.C. Gray, and J.R.R.A. Martins. To CFD or not to CFD? comparing RANS and viscous panel methods for airfoil shape optimization. In *Congress of the International Council of the Aeronau-*

tical Sciences (ICAS 2022), Stockholm, Sweden, 2022.

[3] T. Backhaus, S. Gottfried, A.. Merle, J.T. Hwang, and A. Stück. Modularization of high-fidelity static aeroelastic MDO enabling a framework-based optimization approach for HPC. In *AIAA Scitech 2021 Forum*, 2021.

[4] E.L. Fleeman. Tactical Missile Design. In *AIAA Education Series*, Reston VA, USA, 2001.

[5] R. Voit-Nitschmann. *Vorlesungsskript zu Einführung in die Luftfahrttechnik*. Universität Stuttgart, Stuttgart, Deutschland, 2011.

[6] J.R.R.A. Martins and A. Ning. *Engineering Design Optimization*. Cambridge University Press, Cambridge, UK, 2021. ISBN: 9781108833417.

[7] F. Gallard, C. Vanaret, D. Guenot, V. Gachelin, R. Lafage, B. Pauwels, P.J. Barjhoux, and A. Gazaix. GEMS: a python library for automation of multidisciplinary design optimization process generation. In *AIAA 2018-0657*, Kissimmee, FL, USA, 2018.

[8] L. Reimer. The flowsimulator – a software framework for cfd-related multidisciplinary simulations. In *European NAFEMS Conference: CFD – Beyond the Solve*, Munich, Germany, 2015.

[9] W. McKinney. Data structures for statistical computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[10] W. Fichter und W. Grimm. *Vorlesungsskript zu Flugmechanik*. Universität Stuttgart, Institut für Flugmechanik und Flugregelung, Stuttgart, Deutschland, 2009.

[11] T. Backhaus, M.A.S. Abdul-Kaiyoom, A. Yildirim, A. Stück, and J.R.R.A. Martins. Advancing modularity and frame- work integration level for scalable high-fidelity MDO. In *AIAA Aviation 2023 Forum*, San Diego, CA, USA, 2023.

[12] J.O. Nichols, J.E. Burkhalter, Langley Research Center, and United States. Analysis and compilation of missile aerodynamic data: Data presentation and analysis. Auburn, Ala., USA, 1977. National Aeronautics and Space Administration.

[13] N. Kroll, S. Langer, and A. Schwöppe. The DLR flow solver TAU – status and recent algorithmic developments. In *AIAA 2014-0080*, National Harbor, MD, USA, 2014.

[14] T. Leicht, J. Jägersküpper, D. Vollmer, A. Schwöppe, R. Hartmann, J. Fiedler, and T. Schlauch. DLR-Project Digital-X - next generation CFD solver 'Flucs'. In *Deutscher Luft- und Raumfahrtkongress 2016*, Februar 2016.

Anhang

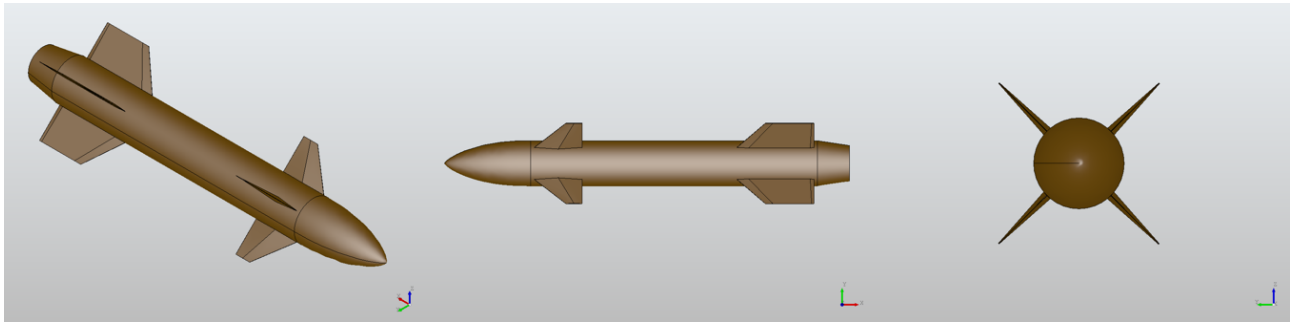

Abb. 11. GMC3: isometrische, Seiten und Front-Ansicht (von links nach rechts)

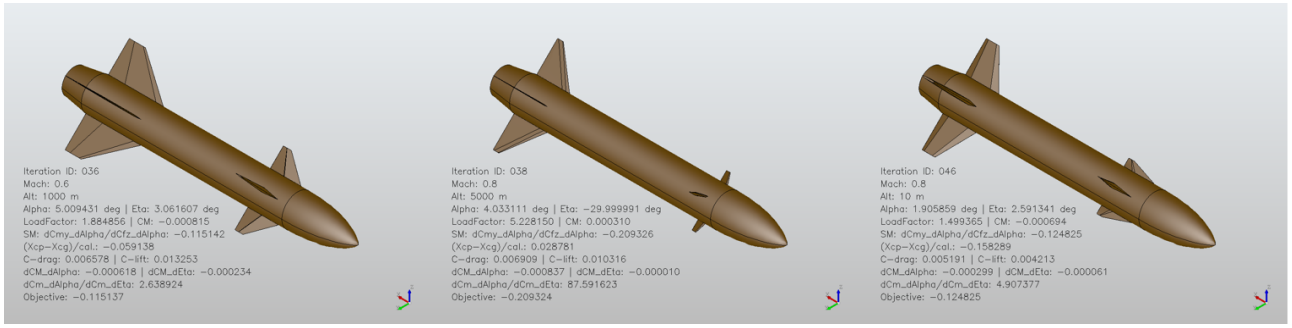

Abb. 12. optimierte Geometrien der unbeschränkten expliziten Optimierung in FP1 (links) bis FP3 (rechts)

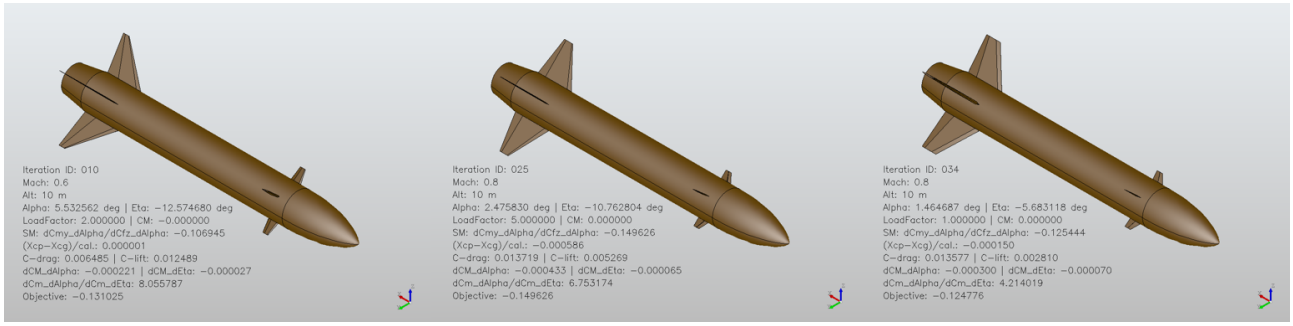

Abb. 13. optimierte Geometrie der unbeschränkten impliziten Optimierung in FP1 (links) bis FP3 (rechts)

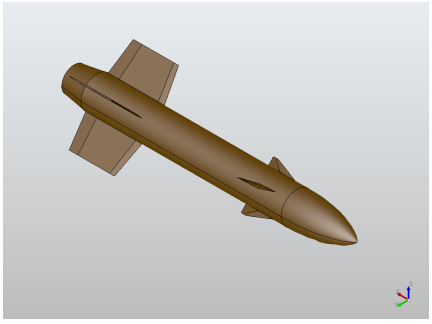

Abb. 14. optimierte Geometrie für die unbeschränkte Multi-Point Optimierung

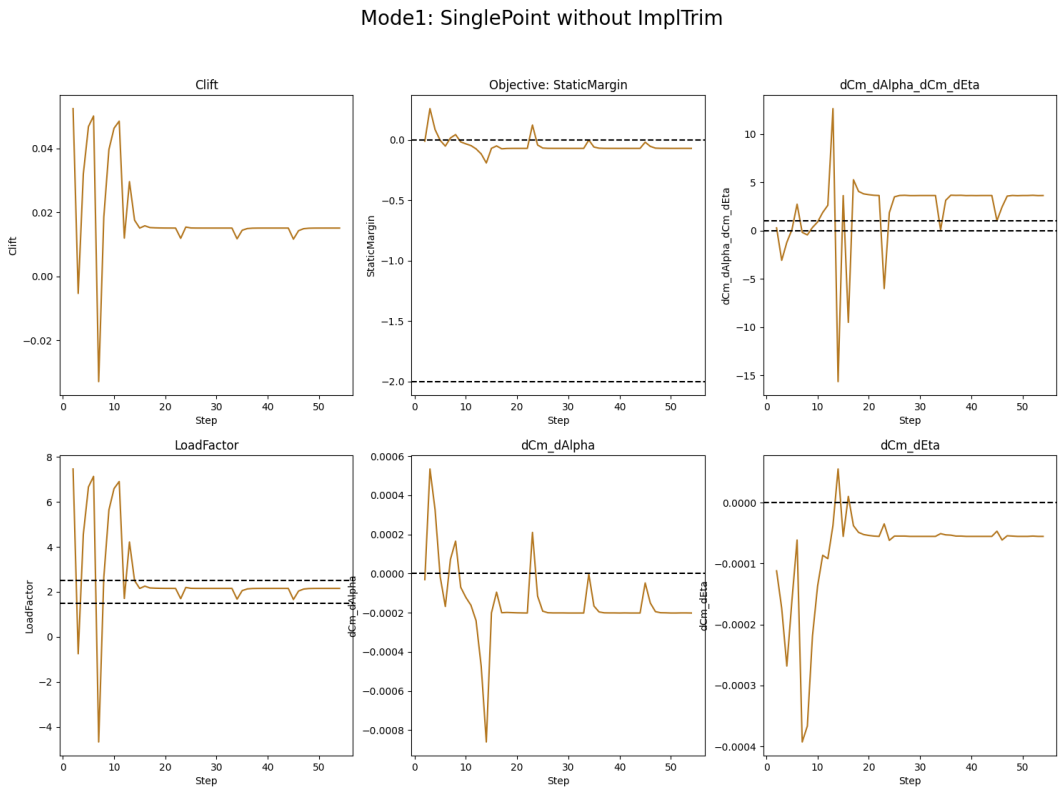

Abb. 15. Konvergenzplot der expliziten Optimierung der Zielkonfiguration in FP1

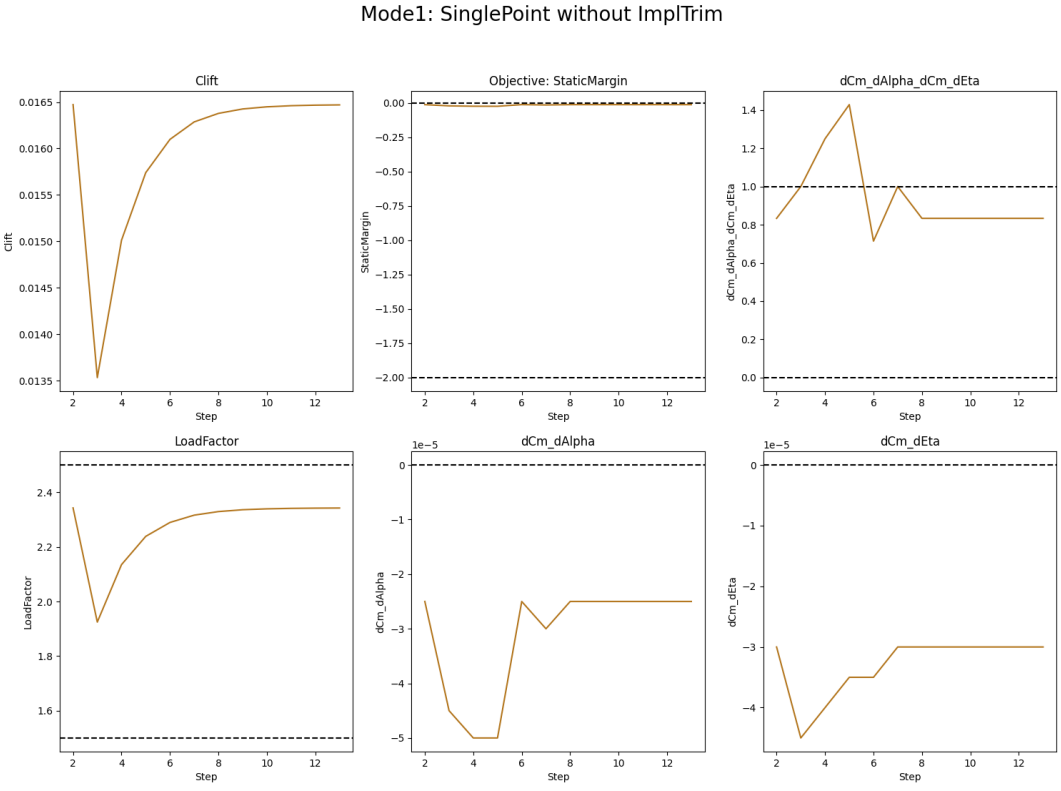

Abb. 16. Konvergenzplot der Optimierung der Zielkonfiguration in FP1 nach Restart

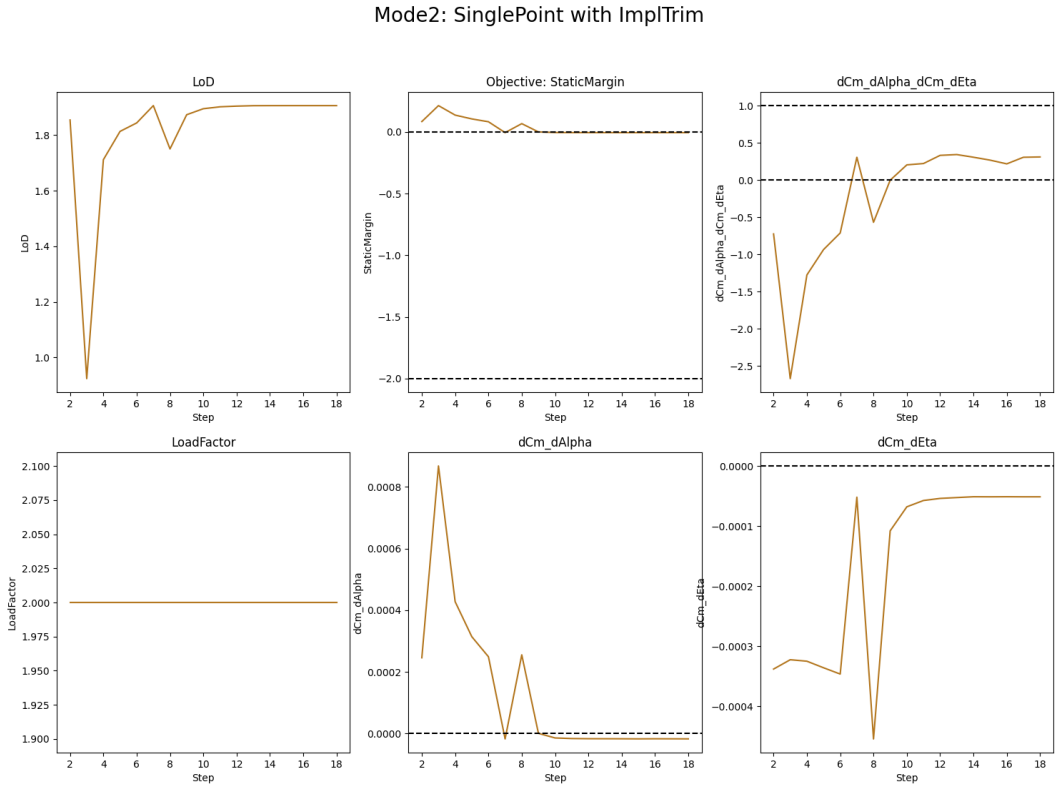

Abb. 17. Konvergenzplot der impliziten Optimierung der Zielkonfiguration in FP1

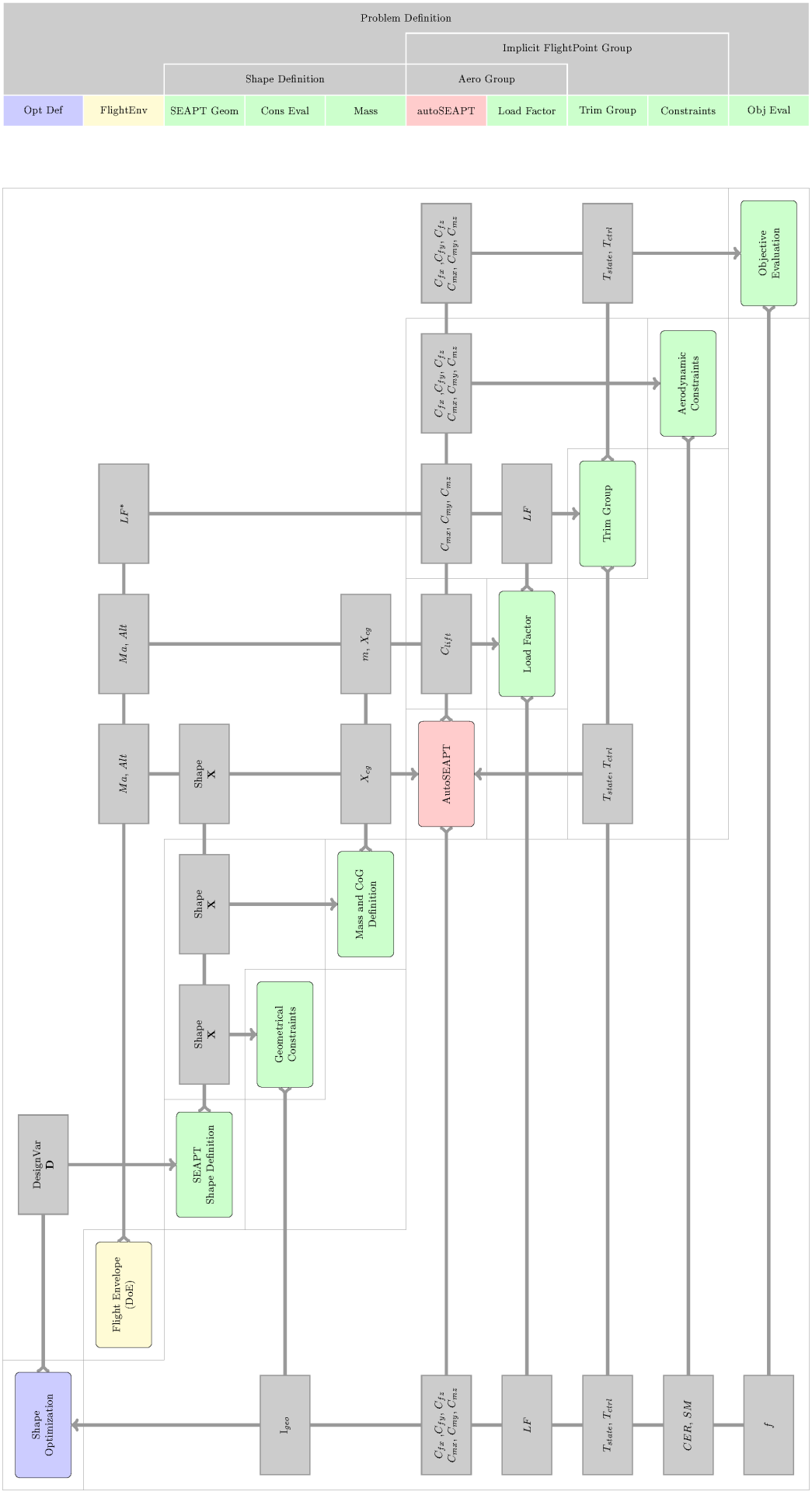

Abb. 18. Szenario 2: Prozessflussdiagramm des Gesamtsystems mit impliziter Trim-Gruppe

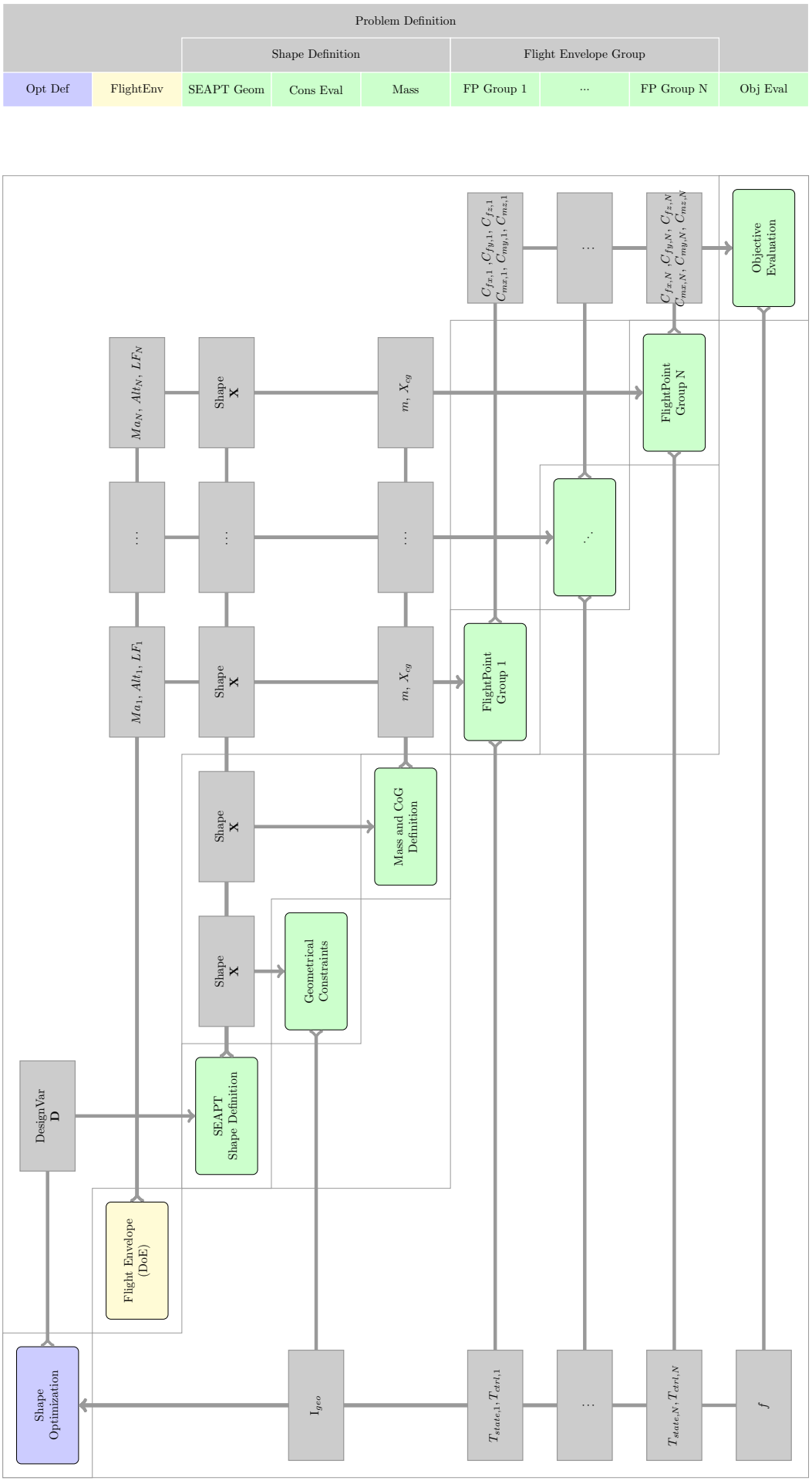

Abb. 19. Szenario 3: Prozessflussdiagramm des Gesamtsystems der Multi-Point Optimierung