Bericht des Instituts für Aerodynamik und Strömungstechnik

Report of the Institute of Aerodynamics and Flow Technology

IB 124-2014/6

Eine netzfreie Last-/Verformungsinterpolationsmethode im FlowSimulator für die Anwendung in statischen aeroelastischen Simulationen von Flugzeugen – Implementierung und Analyse

Andreas Schuster

Institut für Aerodynamik und Strömungstechnik
Braunschweig

Herausgeber:
Deutsches Zentrum für Luft- und Raumfahrt e.V.
in der Helmholtz Gemeinschaft
Institut für Aerodynamik und Strömungstechnik
Lilienthalplatz 7, 38108 Braunschweig

Stufe der Zugänglichkeit: 1
Braunschweig, im Mai 2014

Institutsdirektor:
Prof. Dr.-Ing. habil. C.-C. Rossow

Verfasser:
B.Sc. Andreas Schuster

Betreuer:
Dipl.-Ing. Lars Reimer
Dr.-Ing. Ralf Heinrich

Abteilung: $C^2A^2S^2E$
Abteilungsleiter:
Prof. Dr.-Ing. N. Kroll

Der Bericht enthält:
114 Seiten
51 Bilder
21 Literaturstellen

Eine netzfreie Last-/Verformungsinterpolationsmethode im FlowSimulator für die Anwendung in statischen aeroelastischen Simulationen von Flugzeugen

—

Implementierung und Analyse

Masterarbeit
von
Andreas Schuster

1. Gutachter: Prof. Dr.-Ing. Holger Theisel

2. Gutachter: Dr.-Ing. Ralf Heinrich
Betreuer: Dipl.-Ing. Lars Reimer

Institut für Aerodynamik und Strömungstechnik
Deutsches Zentrum für Luft- und Raumfahrt
Lilienthalplatz 7
38108 Braunschweig

Magdeburg, den 15. Mai 2014

Danksagung

An dieser Stelle möchte ich gerne einigen Menschen danken, ohne die diese Arbeit nicht möglich gewesen wäre. Zuallererst danke ich Prof. Dr. Norbert Kroll vom Institut für Aerodynamik und Strömungstechnik des DLR für die Möglichkeit der studentischen Tätigkeit im Rahmen der Arbeit. In diesem Zusammenhang möchte ich mich auch bei allen Kollegen der Arbeitsgruppe $C^2A^2S^2E$ bedanken, die mir den Aufenthalt am Institut so angenehm gestaltet haben und stets interessante Einblicke ihrer Arbeit in der Arbeitsgruppe gaben. Meinem Betreuer Lars Reimer gilt dabei besonderem Dank, der mir nicht nur während der gesamten Zeit bei Fragen mit Rat und Tat zur Seite stand, sondern mir auch wichtige Hinweise zur Arbeit liefern konnte.

Des Weiteren möchte ich all meinen Freunden danken, die mir in schwierigen Phasen der Arbeit den nötigen Ausgleich gaben. Meinen Eltern möchte ich zuletzt den größten Dank aussprechen, die mich immer unterstützt und mir das Studium überhaupt erst ermöglicht haben.

Zusammenfassung

Einer der wesentlichen Aspekte im Entwicklungsprozess moderner Flugzeuge ist die frühzeitige Erfassung aeroelastischer Einflüsse auf Flugleistung und Stabilität. Der numerischen Simulation mit hochgenauen Verfahren kommt dabei in immer zunehmendem Maße eine tragende Rolle zu. In der industriellen Praxis haben sich sogenannte partitionierte Simulationsverfahren durchgesetzt. In ihnen werden die Verfahren miteinander gekoppelt, die in der Industrie zur hochgenauen Simulation der Einzeldisziplinen „Aerodynamik“ und „Strukturdynamik“ etabliert sind. Die korrekte numerische Umsetzung der zwischen Aerodynamik und Strukturdynamik bestehenden Kopplungsbedingungen ist ein wesentlicher, sensitiver Aspekt von partitionierten Verfahren. Insbesondere herausfordernd ist die korrekte räumliche Übertragung von aerodynamischen Kräften und strukturdynamischen Verschiebungen auf unterschiedlich diskretisierten Netzen.

Vor diesem Hintergrund wird im Rahmen dieser Arbeit ein netzfreier Ansatz zur Last-Verformungs-Übertragung basierend auf der Veröffentlichung von Quaranta et al. umgesetzt, welcher nur auf Knoteninformationen basiert und keinerlei Konnektivitätsinformationen benötigt. Die Implementierung erfolgt im Rahmen der FlowSimulator-Softwareumgebung. Zur Wahrung eines physikalisch korrekten und konservativen Transfers aeroelastischer Kopplungsgrößen kommt zudem eine Bauteil-basierte Kopplung einzelner Flugzeugkomponenten zum Einsatz, welches jedoch spezielle Anpassungen des netzfreien Ansatzes erfordert. Ein wesentlicher Aspekt der Arbeit besteht in der Parallelisierung des Ansatzes nach Quaranta, um eine effiziente Berechnung detaillierter Flugzeugmodelle zu ermöglichen.

Hauptproblematik der Parallelisierung ist die verteilte Suche nach Stützstellen durch die zugrundeliegende Gebietszerlegungen der auf Seiten der Aerodynamik und Strukturdynamik eingesetzten Netze. Zur Lösung kommt in der Arbeit ein Bounding-Box-basierter k-d-Baum zum Einsatz, welcher eine effiziente partitionsunabhängige Stützstellensuche im parallelen Kontext ermöglicht. Die Effizienz des parallelen Algorithmus wurde bis zu einer Anzahl von 96 Prozessen demonstriert. Die Eignung des Algorithmus für komplexe aeroelastische Fragestellungen wurde bestätigt.

Inhaltsverzeichnis

1 Einleitung **7**
- 1.1 Motivation . . . 7
- 1.2 Das FlowSimulator-Framework . . . 8
- 1.3 Zielstellung der Arbeit . . . 11
- 1.4 Gliederung . . . 11

2 Theoretische Grundlagen **13**
- 2.1 Grundlagen der Fluid-Struktur-Kopplung . . . 13
- 2.2 Stand der Forschung . . . 17
  - 2.2.1 Methode der Finiten-Interpolationselemente . . . 17
  - 2.2.2 Methode nach Wendland . . . 18
- 2.3 Ansatz nach Quaranta et al. . . . 22
- 2.4 Blending-Methoden bei Mehrkomponenten-Konfigurationen . . . 25
- 2.5 Gebietszerlegungsmethoden . . . 28
  - 2.5.1 Begriffsdefinitionen . . . 28
  - 2.5.2 Recursive Coordinate Bisection Method . . . 29
  - 2.5.3 Graph-basierte Partitionierungsmethoden . . . 30
- 2.6 Datenstrukturen zur Stützstellensuche . . . 32
  - 2.6.1 k-d-Baum . . . 33

3 Numerische Umsetzung der Quaranta-Methode **35**
- 3.1 Übersicht über durchzuführende Prozessschritte . . . 35
- 3.2 Parallele Stützstellensuche . . . 39
  - 3.2.1 Ansätze der parallelen Nächsten-Nachbarn-Suche . . . 40
  - 3.2.2 Wahl eines geeigneten parallelen Ansatzes . . . 41
  - 3.2.3 Lokale Verarbeitung aller gesendeter Stützstellen . . . 55
- 3.3 Effiziente Berechnung der Kopplungsmatrix . . . 56
  - 3.3.1 CSR-Matrix . . . 57
  - 3.3.2 Parallele Datenspeicherung und verteiltes Matrix-Vektor Produkt . . . 58
- 3.4 Details der parallelen Kommunikation . . . 60

4 Ergebnisse **65**
- 4.1 Untersuchte Testfälle . . . 65
  - 4.1.1 Einfaches Rechteckgitter . . . 65
  - 4.1.2 HiReTT-Konfiguration . . . 66

4.1.3 Flügelkasten-Modell 69
4.1.4 Mehrkomponenten-Konfiguration 69
4.2 Durchgeführte Tests 72
4.3 Analyse von Einkomponenten-Konfigurationen 73
4.3.1 Performance-Untersuchungen der sequentiellen Quaranta-Methode 73
4.3.2 Analysen zur parallelen Quaranta-Methode 75

5 Abschließende Betrachtungen **97**
5.1 Zusammenfassung 97
5.2 Ausblick 98

6 Anhang **101**

Abbildungsverzeichnis **103**

Literaturverzeichnis **107**

1 Einleitung

Der heutige Flugverkehr, sowohl national als auch international, gewinnt zunehmend an Bedeutung. Obwohl die Passagierzahlen leicht stagnieren, nimmt der Frachtverkehr vermehrt zu. Immer mehr Güter werden über den Luftweg transportiert und sind wichtiger Teil einer wettbewerbsfähigen Wirtschaft und Industrie eines Landes. Um das hohe Niveau und den steigenden Bedarf an Waren, aber auch Passagieren zu decken, ist die Nutzung moderner und vor allem wirtschaftlicher Flugzeuge eine essentielle Voraussetzung. Modelle wie der Airbus A350 oder der Boeing 787 Dreamliner zeugen dabei von der enormen Bedeutung dieses Themas und zeigen neue Konzepte des Luftverkehrs auf. Beiden Modellen gemein ist der verstärkte Einsatz von Leichtbaukomponenten. Sie ermöglichen eine weitere Reduzierung des Treibstoffaufkommens. Insbesondere bei den Flügelkomponenten, aber auch bei anderen Bauteilen führt jene Bauweise allerdings zu deutlich nachgiebigeren Strukturen, die sich im Flug wesentlich stärker verformen. Eine Strukturverformung bewirkt als Konsequenz eine Veränderung der Strömungsverhältnisse an der umströmten Konfiguration, was infolgedessen wiederum eine geänderte Strukturverformung nach sich zieht. Diese Wechselwirkung zwischen Aerodynamik und Strukturverformung kann nicht nur die Flugleistung entscheidend beeinträchtigen. Je nach Intensität kann sie zu Instabilitäten im Flug führen, was mit einer Sicherheitsgefahr für das Flugzeug einhergehen kann. Alle diese Probleme müssen somit in der Gesamtkonzeption eines Flugzeugs möglichst früh bei der Entwicklung berücksichtigt werden.

1.1 Motivation

Bei der Entwicklung neuer Flugzeugkonzepte wird dabei die numerische Simulation als probates Mittel genutzt, um getroffene Designentscheidungen frühzeitig zu analysieren und auftretende flugphysikalische Effekte näher zu untersuchen. Windkanalversuche, mit denen das Zusammenspiel von Strömung und Struktur experimentell untersucht werden kann, kommen im Entwicklungsprozess erst sehr spät zum Einsatz. Werden dabei kritische Probleme identifiziert, so kann dies einen enormen Kosten- und Zeitaufwand zur Designänderung nach sich ziehen. Der Einsatz von Simulationen ist mittlerweile unabdingbare Voraussetzung für einen effektiven und kosteneffizienten Entwicklungsprozess.

Standardmäßig werden heute zum einen numerische Methoden der „Computational Fluid Dynamics“ (CFD) zur Untersuchung aerodynamischer sowie Methoden der

„Computational Solid Mechanics“ (CSM) zur Analyse elastostatischer Effekte des Flugverhaltens eingesetzt.

Zur Kopplung der CFD- und CSM-Verfahren im Sinne eines resultierenden aeroelastischen Simulationsverfahrens existieren zwei generelle Ansätze. Zum einen lassen sich die auf CFD- und CSM-Berechnungsebene zu lösenden Gleichungssysteme in einem Gesamtgleichungssystem vereinen. Entscheidender Vorteil ist die inherente Erfüllung der Kopplungsrandbedingungen. Diese in der Literatur genannte monolithische Methodik lässt sich allerdings aber nur auf sehr einfache, eher im akademischen Bereich angesiedelte, Problemstellungen anwenden

Als Alternative für praktisch relevante Problemstellungen haben sich sogenannte partitionierte Verfahren durchgesetzt. Alle Teilgleichungen und deren unterschiedlichen Diskretisierungen werden dabei separat mithilfe problemange-passter numerischer Verfahren gelöst. Die essentiellen Kopplungsbedingungen können dabei jedoch zunächst nicht erfüllt werden. Zur Gewährleistung der Bedingungen erfolgt bei partitionierten Verfahren eine Auftrennung in ein räumliches und zeitliches Kopplungsproblem. Räumlich bedeutet dabei ein physikalisch korrekter Transfer von aerodynamischen Kräften und strukturellen Verschiebungen zwischen den unterschiedlich diskretisierten Netzen. Die zeitliche Kopplung, die jedoch in der Arbeit nicht weiter betrachtet werden soll, bezeichnet dabei eine Synchronisation, d.h. die zeitliche Aufrufabfolge, der getrennten numerischen Lösungsverfahren.

Zur Lösung des räumlichen Kopplungsproblems haben sich in der Vergangenheit Verfahren auf Grundlage radialer Basisfunktionen als vorteilhaft erwiesen. Sie ermöglichen die wichtige Forderung nach Glattheit der Oberflächennetzverschiebungen bei aerodynamischen Berechnungen. In der Arbeit wird dabei ein spezielles, netzfreies Verfahren nach Quaranta et al. [1] auf Basis radialer Basisfunktionen (RBF) näher betrachtet. Im Gegensatz zu alternativen RBF-basierten Verfahren bietet es gewisse Vorteile, die später im Rahmen dieser Arbeit erläutert werden. Netzfrei bedeutet in diesem Kontext, dass nur Knotenkoordinaten und keinerlei Konnektivitätsinformationen vom Verfahren benötigt werden. Dies ist besonders wichtig, da bei kommerziellen CFD-Tools in der Regel nur die reinen Punktinformationen der Oberflächennetze verfügbar sind. Das Verfahren wird im Rahmen der vorliegenden Arbeit zudem im FlowSimulator-Framework implementiert, welches im Folgenden kurz beschrieben wird.

1.2 Das FlowSimulator-Framework

Das FlowSimulator-Framework ist eine Softwareumgebung im Bereich des multidisziplinären Flugzeugentwurfs. Dessen Entwicklung wird in einem gemeinsamen Projekt von europäischen Universitäten, Forschungseinrichtungen (DLR, Onera) sowie Industriepartnern (u.a. Airbus) vorangetrieben und soll den beteiligten Partnern als Rückgrat für massiv-parallele Simulationen dienen.

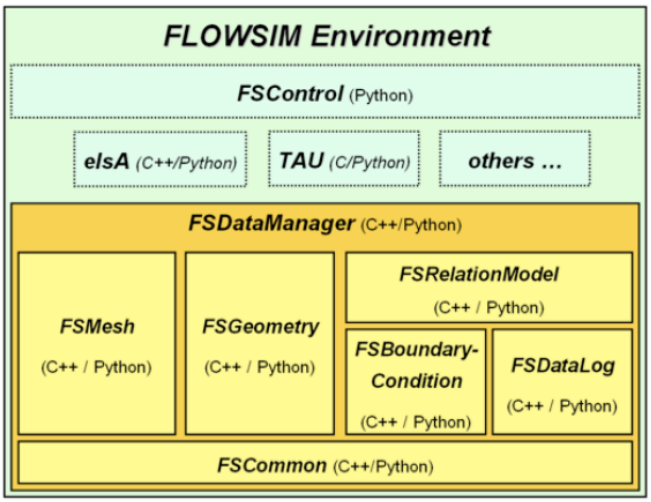


Abbildung 1.1: Darstellung der Komponenten des FlowSimulator-Frameworks

Der FlowSimulator ist, wie in Abbildung 1.1 dargestellt, modular aufgebaut. Neben einem C++-Layer zur performance-optimierten Ausführung rechenintensiver Schritte in multidisziplinären Prozessketten bietet es Python-Anbindungen für eine flexible, Skript-basierte Ausführung der Prozesse.

Der Kern des FlowSimulator stellt der FSDataManager dar. Hauptaspekt dieses Moduls ist die zentrale Verwaltung der in einer Prozesskette vorhandenen Simulationsdaten, wie bspw. den Strömungs- und Strukturnetzen sowie jeweiligen Lösungsdaten. Teilprozesse kommunizieren dabei nicht direkt miteinander, sondern stets mithilfe des FSDataManager. Dies entspricht zugleich der Hauptphilosophie des FlowSimulator-Framework, um die Entwicklung multidisziplinärer Prozessketten zu vereinfachen. Werden neue Disziplinen innerhalb einer Prozesskette berücksichtigt, so muss diese nicht an neue Module angepasst werden, sondern diese Module werden um Schnittstellen zum vorhandenen FSDataManager angepasst. Wie in Abbildung 1.1 zu sehen, ist der FSDataManager zudem von einer Kontrollschicht (*FSControl*) umgeben, mithilfe dessen Funktionalitäten eine benutzerfreundliche Ansteuerung der einzelnen FSDataManager-Funktionen in einer Prozess-Kette und den genutzten Strömungslösern (in Abbildung 1.1 z. B. elsA oder TAU) möglich ist.

Aufbau einer Fluid-Struktur-Kopplungsrechnung

Eine Prozesskette zur Fluid-Struktur-Kopplung besteht im Allgemeinen aus vier Teilberechnungen. Sie lässt sich mithilfe der FlowSimulator-Umgebung anhand von Abbildung 1.2 beschreiben.

Zunächst wird eine initiale Berechnung der aerodynamischen Kräfte auf Basis des CFD-Volumennetzes um die unverformte Konfiguration durchgeführt. Dies wird in Abbildung 1.2 mithilfe des Moduls *FSTau* durchgeführt. *FSTau* ist das

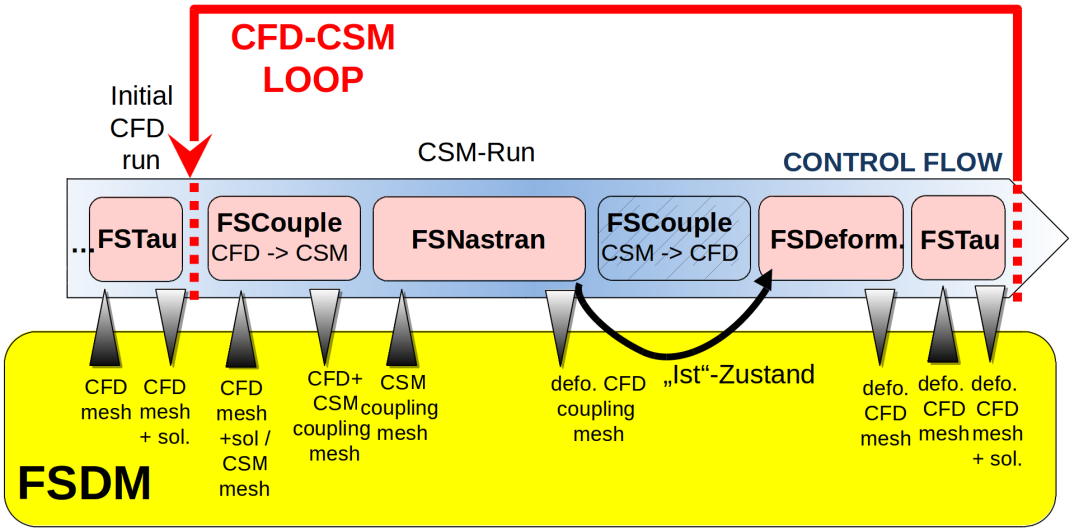


Abbildung 1.2: Darstellung einer Prozesskette zur Fluid-Struktur-Kopplungsberechnung auf Basis des FlowSimulator-Frameworks

FlowSimulator-Interface zum DLR-eigenen Strömungslöser *TAU*[2]. Die berechneten aerodynamischen Kräfte auf dem CFD-Oberflächennetz werden daraufhin mithilfe des Kopplungsmoduls *FSCouple* in Knotenkräfte des CSM-Strukturnetzes umgerechnet. *FSCouple* beinhaltet das Steuerungs-Interface zu räumlichen und zeitlichen Kopplungsverfahren des FlowSimulator.[1] Innerhalb von *FSCouple* soll das in der Arbeit implementierte Verfahren zur Last- und Verformungsinterpolation nach Quaranta et al. eingebettet werden.

Auf Basis der umgerechneten Knotenkräfte auf CSM-Strukturseite werden die resultierenden Verschiebungen der Struktur mithilfe eines FEM-Lösers bestimmt. In Abbildung 1.2 wird dies bspw. über das *FSNastran*-Interface und dem angebunden kommerziellen NASTRAN-Löser realisiert. Die Verschiebungen müssen daraufhin in Verformungen des CFD-Oberflächennetzes umgerechnet werden. Hierzu soll im FlowSimulator-Kontext wiederum FSCouple zum Einsatz kommen. Unter Verwendung der resultierenden CFD-Oberflächenverschiebungen findet dann eine Anpassung des gesamten CFD-Volumennetzes mithilfe des *FSDeformation*-Moduls statt. *FSDeformation* ist ein Netzdeformationsverfahren basierend auf radialen Basisfunktionen[3]. Das Grundprinzip von *FSDeformation* basiert auf der Wendland-Methode (s. Abschnitt 2.2.2).

Mithilfe des angepassten Volumennetzes kann in einem weiteren Schritt eine neue Berechnung der aerodynamischen Kräfte mithilfe von *FSTau* durchgeführt werden.

[1] Bisher ist in dem am DLR verwendeten CFD-CSM-Kopplungsprozess für den Lasttransfer ein Verfahren basierend auf nächsten Nachbarn verwendet.

Der beschriebene Vorgang wird nun entsprechend vorgegebener Kriterien sooft durchgeführt, bis eine Konvergenz der Kopplungsrechnung erzielt werden kann und die aeroelastische Gleichgewichtslage sich eingestellt hat.

1.3 Zielstellung der Arbeit

Der zuvor beschriebene Ablauf einer CFD-CSM-Kopplung im FlowSimulator stellt den „Soll“-Zustand dar. Mangels derzeitig im FlowSimulator vorhandener geeigneter Verfahren zur Verformungsinterpolation, weicht der „Ist“-Zustand von diesem „Soll“-Zustand ab. Statt der Übertragung der strukturellen Verschiebungen in das CFD-Oberflächennetz mit einem Verfahren, welches die bestehenden Kopplungsrandbedingungen erfüllt, findet im „Ist“-Zustand sofort eine Anpassung des CFD-Volumennetzes statt.

Die Tatsache, dass in dem geschilderten „Ist“-Zustand des FlowSimulator-Prozesses unterschiedliche Methoden für den Last- und den Verformungstransfer verwendet werden, entspricht streng genommen allerdings einer Verletzung bestehender Kopplungsrandbedingungen.

Ziel der Arbeit ist daher zum einen die Umsetzung der erwähnten Quaranta-Methode im FlowSimulator-Framework. Da sie sowohl für den Last- als auch Verformungstransfer genutzt werden kann, ermöglicht sie eine physikalisch genauere Berechnung der Fluid-Struktur-Interaktion als bisher vorhandene Methoden und schließt methodisch die derzeitige Lücke innerhalb der FlowSimulator-basierten Kopplungskette. Für eine korrekte räumliche Kopplung realitätsnaher Flugzeug-Konfigurationen muss der Quaranta-Ansatz zudem im Rahmen der Arbeit erweitert werden. Diese Erweiterung entspricht einer Bauteil-basierten, räumlichen Kopplung. Deren Grundprinzip ist dem Konzept aus [4] entnommen.

Da der Detailierungsgrad heutiger aeroelastischer Simulationen im Flugzeugentwurf jedoch weiter gestiegen ist, stößt der Quaranta-Ansatz bei einer seriellen Implementierung performance-technisch an seine Grenzen. In der Arbeit soll deswegen eine parallele Variante der Methode entwickelt werden, mit der signifikante Performance-Steigerungen erzielt werden sollen. Eine mögliche Parallelisierung soll dabei im wesentlichen nicht von der Anzahl verwendeter Prozesse abhängig sein. Da im Rahmen paralleler, numerischer Berechnungen Gebietszerlegungen auf den Simulationsnetzen durchgeführt werden, muss der umgesetze Ansatz zudem unabhängig gegenüber den zugrunde liegenden partitionierten Netze sein.

1.4 Gliederung

Im weiteren Verlauf der Arbeit werden zunächst in Kapitel 2.1 generelle Aspekte der Fluid-Struktur-Kopplung geschildert sowie wichtige Definitionen in diesem Zusammenhang eingeführt, die zum Verständnis beitragen. Daneben wird auf die theo-

retischen Grundlagen des verwendeten Last-/Verformungsinterpolationsverfahrens nach Quaranta et al. in Kapitel 2.3 eingegangen. Die Schwachstellen üblicherweise verwendeter, konkurrierender Verfahren werden zuvor in Kapitel 2.2 erläutert. Kapitel 3 befasst sich näher mit der numerischen Realisierung der Quaranta-Methode und den verfolgten Konzepten zur Parallelisierung. In Kapitel 4 werden dann exemplarisch einige Testfälle herangezogen, mit denen der parallelisierte Quaranta-Ansatz getestet und nach fest definierten Kriterien beurteilt werden soll. Die verwendeten Testfälle unterschieden sich dabei gemäß der definierten Kriterien hinsichtlich ihrer Komplexität. Eine abschließende Bewertung und Zusammenfassung der ermittelten Ergebnisse und kurzer Ausblick runden die Arbeit ab.

2 Theoretische Grundlagen

2.1 Grundlagen der Fluid-Struktur-Kopplung

Aeroelastische Phänomene, wie sie bei der Simulation von Flugzeugen zu beobachten sind, lassen sich formal als Mehrfeldproblem beschreiben, dessen Teilfelder zumeist durch partielle Differentialgleichungen auf einem bestimmten Gebiet definiert sind. Im Kontext der Aeroelastik sind diese Einzelfelder das strömende Medium ("Fluid") sowie der umströmte Festkörper („Struktur“). Die Lösung beider Teilprobleme wird hierbei mithilfe unterschiedlicher numerischer Berechnungsmethoden ermittelt, die jeweils an die spezifischen Anforderungen des einzelnen Feldproblems angepasst sind. Bei der CFD-basierten Berechnung des strömenden Mediums und dessen physikalischen Ausgabegrößen Druck, Strömungsgeschwindigkeit oder Temperatur kommen meist Finite-Volumen-Methoden (FVM) zum Einsatz, die mehrere definierte Erhaltungsgleichungen in jeder Gitterzelle des vernetzten Gebiets lösen.

Im Rahmen der CSM-Struktur-Berechnung und ihrer charakteristischen Größen Verformung oder Verdrehung wird mithilfe von Finite-Elemente-Methoden (FEM) versucht, ein Gleichgewichtszustand (Summe aus äußeren und inneren Kräften) der Struktur und somit ein Minimum an potentieller Energie bei äußerer Belastung, z. B. hervorgerufen durch aerodynamische Kräfte oder Drücke, in jedem Gitterpunkt zu ermitteln. Dies gilt jedoch zunächst nur für ein Strukturproblem innerhalb einer statischen Kopplungsrechnung, welche in dieser Arbeit ausschließlich betrachtet wird. Dynamische Effekte wie z. B. Schwingungen würden sowohl gesonderte Methoden der numerischen Berechnung auf den Teilfeldern als auch erweiterte Kopplungsmethoden benötigen und werden daher in dieser Arbeit nicht weiter betrachtet.

Wesentliches Hauptaugenmerk bei der Verwendung partitionierter Methoden im Rahmen der Lösung von Mehrfeldproblemen liegt, wie bereits einleitend erwähnt, in der Kopplung der auf den Teilfeldern arbeitenden Lösungsverfahren. Der dabei zum Einsatz kommende Kopplungsalgorithmus hat die Aufgabe, einen sinnvollen Austausch an Informationen zwischen den räumlich und zeitlich diskretisierten Teilfeldern zu gewährleisten. Räumlich bezieht sich auf einen adäquaten Transfer von Lasten und Verschiebungen zwischen den diskretisierten Netzen. Die unter dem Begriff der zeitlichen Kopplung beschriebene Bedingung bezeichnet in diesem Kontext eine schrittweise Synchronisation der einzelnen Teillöser innerhalb eines Iterationszykluss, den eine komplette Fluid-Struktur-Kopplung beinhaltet.

Der Austausch an Information unterliegt zudem Bedingungen, welche sich aus der kontinuierlichen Problembeschreibung schlussfolgern lassen. Für eine korrekte räumliche Kopplung muss zunächst eine gemeinsame Teilgrenze bzw. ein Kopplungsbereich beider Systeme definiert werden- hier mit Γ_{FS} bezeichnet- entlang derer Lasten und Verschiebungen ausgetauscht werden. Dieser Austausch zwischen den Einfeldlösern ergibt sich dabei aus der Forderung des Systemzusammenhalts und lässt sich formal folgendermaßen beschreiben:

$$\mathbf{u}_F(t, \mathbf{x}) = \mathbf{u}_S(t, \mathbf{x}) \quad \forall t, \forall \mathbf{x} \in \Gamma_{FS} \tag{2.1}$$

Die Indizes F und S kennzeichnen hierbei jeweils die auf Fluid- bzw. Strukturseite vorhandenen Knoteninformationen (Kräfte, Verformungswerte). Dabei bezeichnet $\mathbf{u}(t, \mathbf{x})$ die Verschiebung eines Punktes auf der Körperoberfläche im Vergleich zu seiner Ruhelage. Im drei-dimensionalen Fall lautet die Verschiebung somit:

$$\mathbf{u}(t, \mathbf{x}) = \begin{pmatrix} u_x(t, \mathbf{x}) \\ u_y(t, \mathbf{x}) \\ u_z(t, \mathbf{x}) \end{pmatrix} \tag{2.2}$$

Des Weiteren muss eine Äquivalenz der übertragenen Lasten und Momenten existieren, mit

$$\sum_{i=1}^{M} \mathbf{F}_{i,F} = \sum_{j=1}^{N} \mathbf{F}_{j,S} \tag{2.3}$$

$$\sum_{i=1}^{M} \mathbf{M}_{i,F} = \sum_{j=1}^{N} \mathbf{M}_{j,S} \tag{2.4}$$

M und N bezeichnen in Gleichung 2.3 und 2.4 die Anzahl an CFD-Oberflächen- und CSM-Strukturknoten. Zusätzlich muss die Gleichheit der Arbeit an der Grenze von Struktur und Strömung bei einer virtuellen Verschiebung der Oberflächenpunkte gewährleistet werden:

$$\delta W_F = \delta \mathbf{U}_{F,\lambda}^T \cdot \mathbf{F}_{F,\lambda} = \delta \mathbf{U}_{S,\lambda}^T \cdot \mathbf{F}_{S,\lambda} = \delta W_S \tag{2.5}$$

Unter einer virtuellen Verschiebung versteht man infinitesimal kleine, gedachte Verschiebungen, die jedoch mit den Randbedingungen des Systems konform sind. Die Verwendung des Index λ trägt dem räumlichen Charakter Rechnung, wobei hier über doppelt vorhandene Indices von 1 bis 3 aufsummiert wird. Analog zu den Kräften $\mathbf{F}_{F,\lambda}^T$ und $\mathbf{F}_{S,\lambda}^T$, kennzeichnen $\mathbf{U}_{F,\lambda}^T$ und $\mathbf{U}_{S,\lambda}^T$ Spaltenmatrizen mit den Verschiebungen an den Fluid- bzw. Strukturknoten. Sie besitzen somit folgenden Aufbau:

$$\mathbf{U}_{S,\lambda}^T = \begin{pmatrix} u_\lambda(\mathbf{x}_1) \\ u_\lambda(\mathbf{x}_2) \\ \vdots \\ u_\lambda(\mathbf{x}_N) \end{pmatrix}, \mathbf{U}_{F,\lambda}^T = \begin{pmatrix} u_\lambda(\mathbf{x}_1) \\ u_\lambda(\mathbf{x}_2) \\ \vdots \\ u_\lambda(\mathbf{x}_M) \end{pmatrix} \tag{2.6}$$

$$\mathbf{F}_{S,\lambda}^T = \begin{pmatrix} F_\lambda(\mathbf{x}_1) \\ F_\lambda(\mathbf{x}_2) \\ \vdots \\ F_\lambda(\mathbf{x}_N) \end{pmatrix}, \mathbf{F}_{F,\lambda}^T = \begin{pmatrix} F_\lambda(\mathbf{x}_1) \\ F_\lambda(\mathbf{x}_2) \\ \vdots \\ F_\lambda(\mathbf{x}_M) \end{pmatrix} \tag{2.7}$$

Eine räumliche Kopplung besteht somit in der Aufgabe einer passenden Umwandlung von Flächenlasten, welche auf der Fluid-Seite vorhanden sind, in Knotenkräfte, welche auf der Strukturseite als Eingabegrößen für die FEM-Berechnung benötigt werden. Demgegenüber müssen strukturseitige Verschiebungen auf die Oberfläche des CFD-Strömungsnetzes entlang des gemeinsamen Kopplungsrands übertragen werden. Wie schon anhand der Indices M und N für die am Projektionsprozess beteiligten Knoten ersichtlich wird, existieren sowohl auf Strömungs- als auch Strukturseite unterschiedliche Diskretisierungen und Netzfeinheiten, angepasst an die numerischen Anforderungen der Finiten-Volumen- bzw. Finite-Elemente-Verfahren. Abbildung 2.1 zeigt ein typisches Beispiel verschiedener Diskretisierungen.

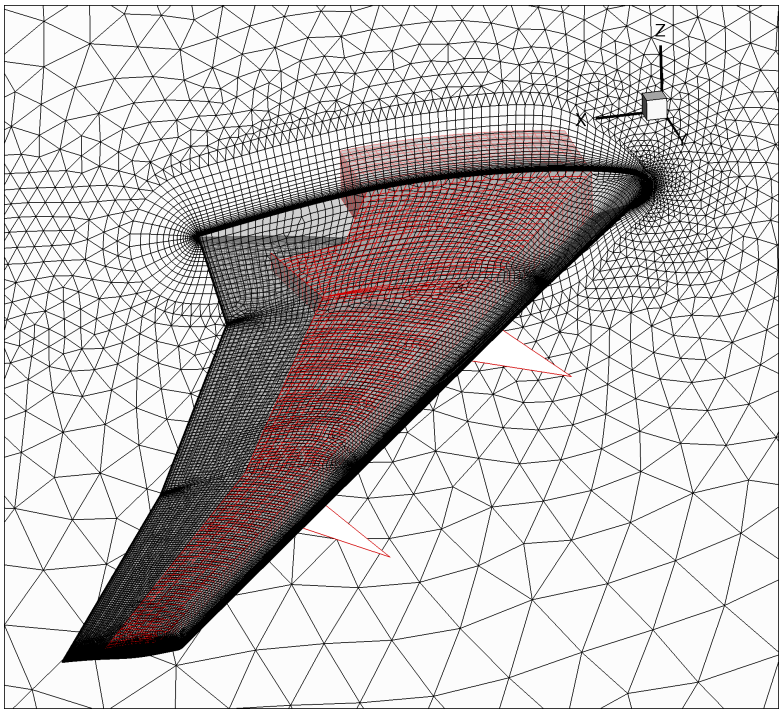


Abbildung 2.1: Gegenüberstellung eines CFD-Oberflächennetzes (schwarz) und dem dazugehörigen Strukturmodell (rot)

Zusammenfassend muss in diesem Zusammenhang ein Ansatz gefunden werden, der eine Beziehung zwischen den Verschiebungen $\mathbf{U}_{F,\lambda}^T$ und $\mathbf{U}_{S,\lambda}^T$ herstellt. Üblich ist dabei der Einsatz über eine Kopplungsmatrix H, deren Aufbau zu diesem Zeitpunkt noch unbekannt ist:

$$\mathbf{U}_{F,\lambda} = \mathbf{H} \cdot \mathbf{U}_{S,\lambda} \tag{2.8}$$

Wird dieser Ausdruck nun in Gleichung 2.5 zur Bestimmung der virtuellen Arbeit eingesetzt, ergibt sich

$$\delta W_F = \delta(\mathbf{H} \cdot \mathbf{U}_{S,\lambda})^T \cdot \mathbf{F}_{F,\lambda} = \delta \mathbf{U}_{S,\lambda}^T \cdot \mathbf{H}^T \cdot \mathbf{F}_{F,\lambda} \tag{2.9}$$

An dieser Stelle kann die Transpositionsregel des Produkts zweier Matrizen ausgenutzt werden, sodass das Produkt der transponierten Einzelmatrizen vertauscht wird. Weiterhin muss die Gleichheit der geleisteten Arbeit gewährleistet werden, was der Forderung nach

$$\delta W_F - \delta W_S = \delta \mathbf{U}_{S,\lambda}^T \cdot \mathbf{H}^T \cdot \mathbf{F}_{F,\lambda} - \delta \mathbf{U}_{S,\lambda}^T \cdot \mathbf{F}_{S,\lambda} \tag{2.10a}$$

$$= \delta \mathbf{U}_{S,\lambda}^T (\mathbf{H}^T \cdot \mathbf{F}_{F,\lambda} - \mathbf{F}_{S,\lambda}) = 0 \tag{2.10b}$$

entspricht. Diese Beziehung muss dabei für alle $\delta \mathbf{U}_{S,\lambda}^T$ gelten und ergibt somit folgendermaßen den Zusammenhang zum korrekten Austausch der Lasten auf die Knoten des Strukturnetzes:

$$\mathbf{H}^T \cdot \mathbf{F}_{F,\lambda} = \mathbf{F}_{S,\lambda} \tag{2.11}$$

Anstelle der Kopplungsmatrix muss also nun für die Lastinterpolation die Transponierte der Matrix benutzt werden.

Zur Generierung der Matrix $\mathbf{H}$ müssen nun geeignete Methoden gefunden werden. Diese müssen im Rahmen aeroelastischer Fragestellungen zwei grundsätzliche Kriterien erfüllen:

- Lokalität bei der Lastübertragung
  Aerodynamische Lasten sollten bei der Interpolation nur lokal wirken - Kräfte an der Flügelspitze haben i. d. R. keine Kraftwirkung am Rumpf oder an den Leitwerken.
- Erzeugung regulärer CFD-Oberflächennetze bei der Verformungsinterpolation
  Das aus der Verschiebungsinterpolation entstehende Oberflächennetz muss zusammenhängend und glatt sein, um eine weitere Berechnung aerodynamischer Kräfte an der verformten CFD-Oberfläche ermöglichen zu können.

Nachfolgend werden neben der Quaranta-Methode oft verwendete alternative Methoden und ihre Eigenschaften vorgestellt.

2.2 Stand der Forschung

2.2.1 Methode der Finiten-Interpolationselemente

Die Methode der Finiten-Interpolationselemente (FIE) stellt die einfachste Methode der Interpolation dar. Sie basiert auf der lokalen Zuordnung von CFD-Punkten und ihren Funktionswerten (Kräfte, Verschiebungen) zum nächstgelegenen Finiten-Element des Strukturnetzes. Während in [5] dieser Ansatz auf Balkenmodelle bei Flugzeugstrukturen geschildert ist, wird im Weiteren Verlauf eine allgemeine, vereinfachte Beschreibung gegeben.

Die Kernidee dieses Ansatzes besteht darin, dieselben Ansatzfunktionen eines finiten Elements neben der Geometriebeschreibung auch zur Interpolation zu benutzen. Diese Ansatzfunktionen sind Teil eines isoparametrischen Konzepts im Rahmen der FEM[6]. Dabei wird eine Abbildungsvorschrift genutzt, um räumliche Koordinaten der Netzknoten auf lokale Koordinaten in den einzelnen Elementen umzurechnen und eine kontinuierliche räumliche Beschreibung der Netzgeometrie innerhalb eines Elements zu erreichen. Im Fall eines zweidimensionalen Problems berechnen sich nun die Koordinaten x und y eines beliebigen Punkts mit den lokalen Koordinaten ξ und η innerhalb eines Elements aus den Ansatzfunktionen der Koordinaten der Elementpunkte wie folgt:

$$x = \sum_{i=1}^{n} N_i(\xi, \eta) x_i \tag{2.12}$$

$$y = \sum_{i=1}^{n} N_i(\xi, \eta) y_i \tag{2.13}$$

n beschreibt die Knotenanzahl eines einzelnen finiten Elements. Für weitere Informationen zum isoparametrischen Konzept sei der Leser auf [6] verwiesen. Im Rahmen des Interpolationsproblems wird nun die umgekehrte Abbildung benötigt, d. h. es sind die lokalen Koordinaten ξ und η für gegebene Koordinaten x und y zu ermitteln (siehe Abbildung 2.2)

Je nach Polynomgrad der verwendeten Ansatzfunktionen führt dies zu einem nichtlinearen Problem in ξ und η, welches mithilfe eines Newton-Verfahrens gelöst werden kann. Sind die natürlichen Koordinaten des CFD-Punkts innerhalb des Elements ermittelt, lässt sich die Verformung anhand der strukturseitigen Verformungen des Elements mithilfe von

$$\mathbf{u}_F = \mathbf{H}(\xi, \eta) \mathbf{U}_S^{ele} \tag{2.14}$$

interpolieren. Für eine energetisch korrekte Lastinterpolation der Strömungskräfte auf das Strukturelement muss als Ansatz

$$\mathbf{F}_S^{ele} = \mathbf{H}(\xi, \eta)^T \mathbf{F}_F \tag{2.15}$$

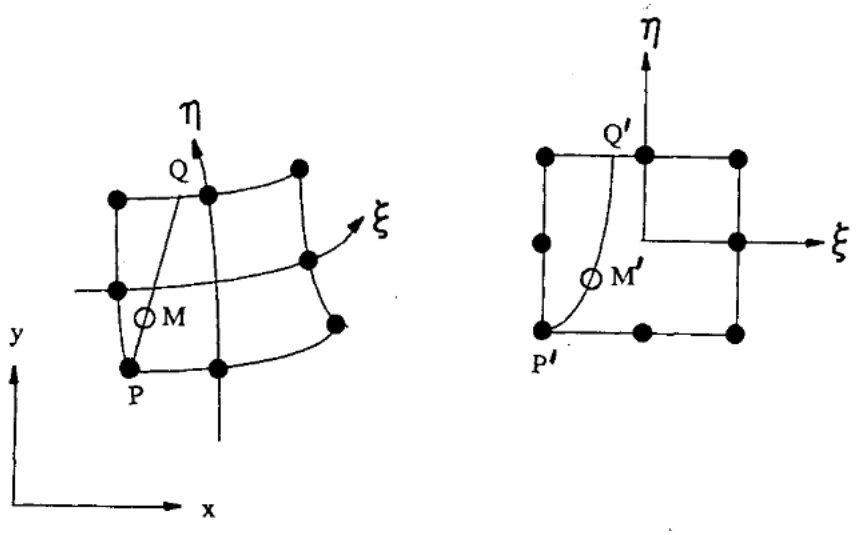


Abbildung 2.2: Projektion eines CFD-Punkts M in die natürlichen Koordinaten des finiten Elements [7]

verwendet werden.

Wesentlicher Vorteil dieser Methode ist nicht nur die einfache Zuordnung eines Punktes zum nächsten finiten Element. Durch die Summeneigenschaft der beteiligten Ansatzfunktionen $\sum_{i=1}^{n} N_i(\xi, \eta) = 1$ wird zudem die Gleichheit der Kräfte und Momente garantiert und somit ein konservativer Last- und Verformungstransfer gewährleistet.

Die Methode hat jedoch einen entscheidenden Nachteil, wenn zwischen den Elementen des Strukturnetzes und den Zellen des Strömungsgitters ein zu großer Größenunterschied herrscht. Somit werden u. U. nicht alle Knoten des Strukturnetzes von der Projektion der auf die einzelnen CFD-Knoten umgerechneten Flächenlasten erfasst und beim Lasttransfer berücksichtigt. Dies führt im Allgemeinen zu einer sehr ungleichmäßigen Verteilung der Kräfte, welches zudem unphysikalische Verformungen nach sich ziehen kann. Zudem kann die Methode aufgrund des sehr lokalen Ansatzes bei stark voneinander abweichenden Netzen an der Kopplungsfläche zu nicht glatten, verformten CFD-Oberflächennetzen führen, die zu einem Abbruch der Rechnung führen können.

2.2.2 Methode nach Wendland

Ein alternativer, häufig verwendeter Ansatz ist die sogenannten Wendland-Methode[8][1] . Sie beruht auf einer Rekonstruktion der Last- und Verformungsgrößen mithilfe von globalen Ansatzfunktionen und einer größeren Anzahl verteilter Stützstellen. Die

[1]Fußnote: Im Wesentlichen kommt derzeit die Wendland-Methode für den Verformungstransfer im „Ist“-Zustand des FlowSimulator-Kopplungsprozesses am DLR zum Einsatz.

globale Ansatzfunktion soll eine im Vergleich zur FIE-Methode gleichmäßigere Verteilung der Kräfte und Verformungen gewährleisten. Jeder Strukturknoten in einer definierten Umgebung zum CFD-Punkt besitzt den Charakter eines Stützpunktes $\mathbf{x}_i$ mit gegebenen Funktionswerten

$$f_\lambda(\mathbf{x}_i) = u_\lambda(\mathbf{x}_i) \tag{2.16}$$

Mithilfe der Funktionswerte lässt sich nun die Verschiebung eines beliebigen Punktes $\tilde{\mathbf{x}}$ in funktionaler Weise darstellen:

$$\hat{f}_\lambda(\tilde{\mathbf{x}}) = \hat{u}_\lambda(\tilde{\mathbf{x}}) = \mathbf{b}(\tilde{\mathbf{x}})^T \cdot \mathbf{K}_\lambda \tag{2.17}$$

Die Spaltenmatrix $\mathbf{b}(\tilde{\mathbf{x}})^T$ kennzeichnet in Gleichung 2.17 die einzelnen Q Terme bzw. Monome des gewählten Polynoms n-ten Grades. Für ein Polynom ersten Grades, welches in dieser Arbeit ausschließlich verwendet wird, besitzt $\mathbf{b}(\tilde{\mathbf{x}})^T$ dabei folgenden Aufbau:

$$\mathbf{b}^1(\tilde{\mathbf{x}})^T = \begin{pmatrix} 1 \\ \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} \tag{2.18}$$

Die Spaltenmatrix $\mathbf{K}_\lambda$ kennzeichnet im Weiteren Verlauf die einzelnen Koeffizienten der Monome. Zur Bildung der Approximationsfunktion $\hat{f}_\lambda(\tilde{\mathbf{x}})$ müssen diese Koeffizienten nun ermittelt werden. Ein verbreiteter Ansatz ist hierfür die Methode der kleinsten Fehlerquadrate, welche als Minimierungsproblem formulierbar ist und deren Funktional mit

$$J_{LS,\lambda} = \sum_{i=1}^{N} \|\hat{f}_\lambda(\tilde{\mathbf{x}}) - f_\lambda(\mathbf{x}_i)\|^2 \tag{2.19}$$

beschrieben wird. Die Lösung dieses Problems ergibt die einzelnen Koeffizienten der Spaltenmatrix $\mathbf{K}_\lambda$. Da es sich hierbei jedoch um eine Approximationsfunktion handelt, können die approximierten Funktionswerte an den Stützstellen von den gegebenen abweichen. Um dies zu umgehen, wird in der Methode von Wendland et al. ein Korrekturterm in das Funktional eingefügt, um eine korrekte Interpolation der Funktionswerte an den Stützstellen zu gewährleisten[9]:

$$\hat{f}_\lambda(\tilde{\mathbf{x}}) = \hat{u}_\lambda(\tilde{\mathbf{x}}) = \sum_{i=1}^{N} \alpha_{i,\lambda} \Phi(\|\tilde{\mathbf{x}} - \mathbf{x}_i\|^2) + p_\lambda(\tilde{\mathbf{x}}) \tag{2.20}$$

mit

$$p_\lambda(\tilde{\mathbf{x}}) = \mathbf{b}(\tilde{\mathbf{x}})^T \cdot \mathbf{K}_\lambda \tag{2.21}$$

Der verwendete Korrekturterm besteht aus radialen Basisfunktionen $\Phi(x)$, welche multiplikativ um Koeffizienten $\alpha_{i,\lambda}$ erweitert auf die approximierten Funktionswerte $p_\lambda(\tilde{\mathbf{x}})$ aufgetragen werden.

Die im Korrekturterm eingefügte radiale Basisfunktion Φ berücksichtigt nun den Abstand eines Punktes $\tilde{\mathbf{x}}$ zu einem beliebigen Punkt - im Rahmen der Interpolation bspw. zu einem gegebenen Stützpunkt. Φ ist somit eine Funktion des Abstands zweier Punkte $r = \|\tilde{\mathbf{x}} - \mathbf{x}_i\|^2$. Die Funktion hat zudem die wichtige Eigenschaft, dass sie nur innerhalb eines definierten Stützradius δ wirkt und für $r' > 1$ verschwindet,

$$\Phi(r') = \begin{cases} > 0, & r' < 1 \\ = 0, & r' \geq 1 \end{cases} \tag{2.22}$$

mit

$$r' = \frac{r}{\delta}$$

Typische Vertreter radialer Basisfunktion sind in Tabelle 2.1 dargestellt.

$\Phi(r')\ mit\ r' = \frac{\|x_i - x_j\|}{\delta}$	Name
$(1 - r')_+^2$	Wendland C0-Funktion
$(1 - r')_+^4(4r' + 1)$	Wendland C2-Funktion
$\frac{\pi}{12}(r' + 2)(1 - r')_+^2$	Euclid's Hat
$exp(-r'^2)$	Gauß-Funktion

Tabelle 2.1: Ausgewählte radiale Basisfunktionen

Im Rahmen dieser Arbeit wird die Wendland-C2-Funktion (im Rahmen der Umsetzung der Quaranta-Methode) verwendet.

Durch den Einfluss eines Stützpunkts in einem definierten Stützradius kann mithilfe der RBF ein glatter Verlauf der Approximationsfunktion in den Stützpunkten gewährleistet werden, sodass die approximierten Verschiebungswerte in den Stützstellen mit den gegebenen übereinstimmen:

$$\hat{f}_\lambda(\mathbf{x}_i) = f_\lambda(\mathbf{x}_i) \tag{2.23}$$

Gemäß Gleichung 2.20 müssen nun innerhalb jeder Raumrichtung entsprechend der gesamten Anzahl an Strukturknoten N Koeffizienten $\alpha_{i,\lambda}$ errechnet werden sowie so viele Koeffizienten $K_j(\tilde{\mathbf{x}})$, wie das verwendete Polynom Monome besitzt.

Demgegenüber muss als zweite zusätzliche Randbedingung gelten:

$$\sum_{i=1}^{N} \mathbf{b}(\mathbf{x}_i)\alpha_{i,\lambda} = \mathbf{0} \tag{2.24}$$

Setzt man beide Randbedingungen in die ursprüngliche Approximationsfunktion ein, ergibt sich

$$\underbrace{\begin{pmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & 0 \end{pmatrix}}_{C_{ss}} \begin{pmatrix} \alpha_\lambda(\mathbf{x}_1) \\ \alpha_\lambda(\mathbf{x}_2) \\ \vdots \\ \alpha_\lambda(\mathbf{x}_n) \\ k_{1,\lambda} \\ k_{2,\lambda} \\ \vdots \\ k_{Q,\lambda} \end{pmatrix} = \begin{pmatrix} u_\lambda(\mathbf{x}_1) \\ u_\lambda(\mathbf{x}_2) \\ \vdots \\ u_\lambda(\mathbf{x}_n) \\ \mathbf{0} \end{pmatrix} \tag{2.25}$$

A kennzeichnet dabei eine Matrix, die aus den radialen Basisfunktionen der Strukturknoten ermittelt wird, somit

$$A_{i,j} = \Phi(r'_{ss}) \qquad mit \qquad r'_{ss} = \frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\delta_{ss}} \tag{2.26}$$

δ_{ss} kennzeichnet hier den Stützradius, welcher die einzelnen Abstände der Strukturknoten zueinander skaliert. Die Lösung der Gleichung 2.25 liefert nun die gesuchten Koeffizienten α_λ und K_λ .

$$\begin{pmatrix} \boldsymbol{\alpha}_\lambda \\ \mathbf{K}_\lambda \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & 0 \end{pmatrix}^{-1} \begin{pmatrix} u_\lambda(\mathbf{x}_1) \\ u_\lambda(\mathbf{x}_2) \\ \vdots \\ u_\lambda(\mathbf{x}_n) \\ \mathbf{0} \end{pmatrix} \tag{2.27}$$

Die Matrix **P** wird hierbei zeilenweise aus den einzelnen Q Monomen der Strukturknoten gebildet, bei dem die i-te Zeile von **P** der Spaltenmatrix $\mathbf{b}(\mathbf{x}_i)^T$ entspricht:

$$\mathbf{P} = \begin{pmatrix} \mathbf{b}(\mathbf{x}_1)^T \\ \mathbf{b}(\mathbf{x}_2)^T \\ \vdots \\ \mathbf{b}(\mathbf{x}_n)^T \end{pmatrix} \tag{2.28}$$

Mithilfe der Koeffizienten lassen sich nun die Verformungen der CFD-Oberflächenpunkte folgendermaßen ermitteln:

$$\begin{pmatrix} \hat{u}_\lambda(\mathbf{y}_1) \\ \hat{u}_\lambda(\mathbf{y}_2) \\ \vdots \\ \hat{u}_\lambda(\mathbf{y}_M) \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{A}} & \tilde{\mathbf{P}} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\lambda \\ \mathbf{K}_\lambda \end{pmatrix} \tag{2.29}$$

Durch den Einsatz eines glatten Funktionals für das Verschiebungsfeld können die evtl. nicht glatten CFD-Oberflächenverformungen, die aus der Verwendung der FIE-Methode resultieren können, eliminiert werden. Ungeachtet dieses Vorteils hat die Wendland-Methode dennoch einen entscheidenden Nachteil. Sie bedingt die Invertierung der unter Umständen dicht besetzten sehr großen Matrix C_{ss}. Sie hat die Dimension „Anzahl der CFD-Oberflächenpunkte“ mal Änzahl der CSM-Strukturkopplungspunkte". Aus Rechenzeit- oder Arbeitsspeichergründen kann die C_{ss}-Matrix lediglich für einer Untermenge der Menge aller Kopplungspunkte ausgeführt werden. Durch diesen Aspekt vermindert sich allerdings die Genauigkeit der Methode, da nur an den Kopplungspunkten gewährleistet ist, dass die interpolierte Verschiebung der vorgegebenen Verschiebung entspricht.

In Bezug darauf bietet die Quaranta-Methode, wie nachfolgend erörtert wird, entscheidende Vorteile. An dieser Stelle sei auf Details der Wendland-Methode nicht weiter eingegangen. Es wird stattdessen auf [5] verwiesen, worin die Ergebnisse zur Genauigkeit der Wendland-Methode und deren Rechenzeit- und Speicherbedürfnisse ausführlicher beschrieben werden.

2.3 Ansatz nach Quaranta et al.

Die nun im Verlauf der Arbeit näher betrachtete Methode nach Quaranta et al. kompensiert die Nachteile der Wendland-Methode durch die Definition einer lokalen Approximationsfunktion anstelle einer globalen. Die Quaranta-Methode basiert im Allgemeinen auf dem Moving-Least-Squares-Verfahren (MLS).

Der MLS-Ansatz führt für einen gegebenen Punkt $\tilde{\mathbf{x}}$ zu dem gewichteten Fehlerfunktional

$$J_{MLS}(\tilde{\mathbf{x}}) = \sum_{i}^{N} \Phi(r') \left(\| \hat{f}_\lambda(\mathbf{x}_i) - f_\lambda(\mathbf{x}_i) \|^2 \right) \tag{2.30}$$

mit

$$r' = \frac{\|\tilde{\mathbf{x}} - \mathbf{x}_i\|}{\delta_Q(\tilde{\mathbf{x}})} \tag{2.31}$$

Dieses Funktional muss, wie schon bei der Wendland-Methode, für alle Raumrichtungen minimiert werden. Im Unterschied zur Wendland-Methode wird der entsprechende Fehlerterm nun multiplikativ um Gewichtungen radialer Basisfunktionen erweitert. Diese bewirken, dass im Minimierungsproblem nur Strukturpunkte $\mathbf{x}_i$ eingehen, die innerhalb eines definierten Radius $\delta_Q(\tilde{\mathbf{x}})$ liegen. Zudem kann dieser Radius anhand der lokal vorhandenen CSM-Punktanzahl unterschiedlich für jeden zu interpolierenden CFD-Oberflächenpunkt gewählt werden. Somit kann dadurch nicht nur die Lokalität der interpolierten Last- und Verschiebungsgrößen gewährleistet werden. Die Lösung des Problems führt im Weiteren zu einem wesentlich kleineren Gleichungssystem als bei der Wendland-Methode.

Als Randbedingung wird bei der Quaranta-Methode ebenfalls die Gleichheit der angenäherten Verschiebungsgrößen und vorhandenen Verschiebungen an den Stützstellen gefordert.

$$\hat{f}_\lambda(\mathbf{x}_i) = f_\lambda(\mathbf{x}_i)$$
$$\hat{u}_\lambda(\mathbf{x}_i) = u_\lambda(\mathbf{x}_i)$$

Ferner sei

$$\hat{f}_\lambda(\tilde{\mathbf{x}}) = \mathbf{b}(\tilde{\mathbf{x}})^T \cdot \mathbf{K}_\lambda(\tilde{\mathbf{x}}) \tag{2.32}$$

Damit lässt sich Gleichung 2.30 in Matrixschreibweise ausdrücken:

$$J_{MLS,\lambda}(\tilde{\mathbf{x}}) = (\mathbf{P}\mathbf{K}_\lambda(\tilde{\mathbf{x}}) - \mathbf{U}_{S,\lambda})^T \mathbf{\Phi}(\tilde{\mathbf{x}})(\mathbf{P}\mathbf{K}_\lambda(\tilde{\mathbf{x}}) - \mathbf{U}_{S,\lambda}) \tag{2.33}$$

$\mathbf{P}$ beschreibt die einzelnen Monome des gewählten Ansatzpolynoms einzelner Koordinaten der Strukturknoten.

$\mathbf{\Phi}(\tilde{\mathbf{x}})$ kennzeichnet eine Diagonalmatrix aus radialen Basisfunktionen, bei der jeder Abstand eines Strukturknotens zu einem gegebenen Punkt $\tilde{\mathbf{x}}$ einem Eintrag auf der Hauptdiagonalen entspricht:

$$\mathbf{\Phi} = \begin{pmatrix} \Phi(r'_1(\tilde{\mathbf{x}})) & 0 & \cdots & 0 \\ 0 & \Phi(r'_2(\tilde{\mathbf{x}})) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Phi(r'_N(\tilde{\mathbf{x}})) \end{pmatrix} \tag{2.34}$$

Die Lösung des Problems 2.30 führt dann zu folgender Gleichung zur Bestimmung der Koeffizientenmatrix:

$$\mathbf{K}_\lambda(\tilde{\mathbf{x}}) = \mathbf{X}(\tilde{\mathbf{x}})\mathbf{U}_{S,\lambda} \tag{2.35}$$

Die Matrix $\mathbf{X}(\tilde{\mathbf{x}}) \in R^{Q \times N}$ enstspricht dabei der Lösung von

$$\underbrace{\mathbf{\Phi}(\tilde{\mathbf{x}})\mathbf{P}}_{\mathbf{C}(\tilde{\mathbf{x}})}\mathbf{X}(\tilde{\mathbf{x}}) = \mathbf{\Phi}(\tilde{\mathbf{x}}) \tag{2.36}$$

Gleichung 2.36 lässt sich somit als Gleichungssystem mit N rechten Seiten, entsprechend der Spalten von $\mathbf{\Phi}(\tilde{\mathbf{x}})$ beschreiben. Jede Spalte i von $\mathbf{X}(\tilde{\mathbf{x}})$ beschreibt dabei die Lösung von 2.36 mit der i-ten Spalte von $\mathbf{\Phi}(\tilde{\mathbf{x}})$ als rechte Seite. Zur Lösung kann eine QR-Zerlegung auf Basis der Householder-Transformationen herangezogen werden, welche in der Lage ist, Gleichungssysteme mit mehr als einer rechten Seite zu lösen. Die Lösung des Gleichungssystems auf Grundlage einer rechten Seite lässt sich allerdings robuster mithilfe einer Singulärwertzerlegung realisieren, welche aus diesem Grund in dieser Arbeit verwendet wird. Der Vollständigkeit halber muss an dieser Stelle noch hinzugefügt werden, dass sich Gleichung 2.36 in ein reduziertes Gleichungssystem transformieren lässt. Spalten und Zeilen, die durch Strukturknoten repräsentiert sind, die außerhalb des Stützradius liegen, lassen sich

aufgrund ihrer Null-Einträge eliminieren. Auf die Form bzw. Lösungskoeffizienten der Matrix $\mathbf{X}(\tilde{\mathbf{x}})$ hat dies jedoch keinen Einfluss.

Mithilfe der Lösung von Gleichung 2.36 ließen sich nun die Koeffzienten $\mathbf{K}_\lambda(\tilde{\mathbf{x}})$ bestimmen. Diese explizite Vorgehensweise ist jedoch unzweckmäßig, da die Matrix $\mathbf{K}_\lambda(\tilde{\mathbf{x}})$ gemäß Gleichung 2.35 bereits die Verschiebungen der Strukturknoten beinhaltet und somit einen speziellen Verschiebungsfall charakterisiert. Vielmehr ist aber die Erstellung einer allgemeinen Kopplungsmatrix $\mathbf{H}$ gemäß Gleichung 2.8 von Interesse, bei der jedes beliebige, strukturseitige Verschiebungsfeld als Eingabe für die Interpolation genutzt wird, um eine Interpolation beliebiger CFD-Oberflächenverformungen zu realisieren. Zu diesem Zweck lässt sich 2.35 in 2.17 einsetzen, womit sich

$$\hat{f}_\lambda(\tilde{\mathbf{x}}) = u(\tilde{\mathbf{x}})_F = \underbrace{\mathbf{b}(\tilde{\mathbf{x}})^T \cdot \mathbf{X}(\tilde{\mathbf{x}})}_{\mathbf{\Psi}(\tilde{\mathbf{x}})} \mathbf{U}_{S,\lambda} \tag{2.37}$$

ergibt. Die Matrix $\mathbf{\Psi}(\tilde{\mathbf{x}})$ besitzt hierbei die Größe $1 \times N$ und wird auch Formfunktion genannt. Wird die Matrix für alle M CFD-Oberflächenpunkte angewandt, an denen die Verschiebungen der Strukturknoten angenähert werden sollen, so kennzeichnet $\mathbf{\Psi}(\mathbf{y}_j)$ für $j = 0, ..., M$ die j-te Zeile der Gesamtkopplungsmatrix $\mathbf{H}$. An dieser Stelle muss ferner erwähnt werden, dass zur Lösung von 2.36 mindestens Q-Stützstellen vorhanden sein müssen, die zudem alle nicht auf dem Rand des gewählten Stützradius liegen dürfen. Liegen zudem alle Stützpunkte in einer Ebene, so wird die Gleichung 2.36 singulär und somit nicht lösbar. Diese Problematik stellt eine besondere Herausforderung bei der Betrachtung quasi-eindimensionaler Strukturmodelle, z. B. Balkenmodelle, dar. Im Rahmen dieser Arbeit wird diese besondere Situation jedoch nicht weiter betrachtet, sodass auf Strukturseite von dreidimensionalen Netztopologien (aus Volumen- oder Schalenelementen) ausgegangen wird. Für den interessierten Leser wird bei obiger Problematik auf [5] verwießen.

Kernaspekt der Methode nach Quaranta ist insgesamt das Finden von Strukturknoten als geeignete Stützstellen zur Verformungsinterpolation des CFD-Oberflächennetzes. Im Rahmen dieser Arbeit soll zudem eine parallele Variante dieser Interpolationsmethode umgesetzt werden. Kernproblem dabei ist jedoch die verteilte Lage der Strukturknoten in mehreren Teilnetzen, welche somit eine effektive Suche nach Stützstellen erschweren. Diese Teilnetze werden mithilfe von Gebietszerlegungsmethoden erzeugt, welche daher in Abschnitt 2.5 näher erläutert werden.

2.4 Blending-Methoden bei Mehrkomponenten-Konfigurationen

Obwohl die Quaranta-Methode im Bereich einzelner Komponenten ein neuer praktikabler Ansatz für einen räumlichen Last- und Verformungstransfer darstellen könnte, ist ihre Verwendung im Umfeld kompletter Flugzeugkonfigurationen unangemessen und stößt an ihre Grenzen. Durch die Anwendung der Quaranta-Methode wird zwar die physikalisch lokale Wirkung der aerodynamischen Kräfte und Verschiebungen innerhalb der Interpolation berücksichtigt. Jene Kräfte und Verschiebungen würden aber bei Komponenten, welche nahe beieinander liegen, aber keine geometrische Verbindung besitzen, falsch projiziert. Um eine korrekte Projektion von Kräften und Verschiebungen weiterhin zu gewährleisten, werden sogenannte Baugruppen definiert. Unter Baugruppen wird in diesem Zusammenhang eine Funktionseinheit des Flugzeugs verstanden, z. B. Flügel, Rumpf, Landeklappe, Höhenleitwerk, Seitenleitwerk, usw. Innerhalb einer solchen Baugruppe wird eine Zuordnung der jeweiligen CFD-seitigen Repräsentierung der Baugruppe zur entsprechenden strukturseitigen Repräsentierung getroffen.

Dies hat jedoch zur Folge, dass es im Falle einer Verformungsinterpolation einzelner Baugruppen an Übergängen zwischen einzelnen Komponenten (z. B. zwischen dem Rumpf und Flügel) zu Sprüngen und Klaffungen im Verformungsverlauf kommen kann, welche zu einem Abbruch der aerodynamischen Berechnungen im Gesamtprozess der Fluid-Struktur-Kopplung führen kann. Um diese Problematik zu umgehen, muss ein sogenanntes „Blending“[4] durchgeführt werden, um die Geschlossenheit der CFD-Oberfläche gewährleisten zu können.

Kerngedanke des Blending ist die Zuordnung eines CFD-Punkts einer Baugruppe zu den Strukturnetzen, die zu den Baugruppen gehören, die an die Baugruppe des CFD-Punkt angrenzen. Ein Punkt auf der CFD-Oberfläche eines Flügels wird bspw. zunächst der direkten Strukturkomponente des Flügels zugeordnet, um somit den direkten Einfluss der strukturseitigen Flügeldeformationen auf dem CFD-Punkt ermitteln zu können. Da jedoch zwischen Flügel und Rumpf seitens der CFD-Oberfläche eine gemeinsame Schnittkurve existiert, wird dieser Punkt im nächsten Schritt auch der Strukturkomponente des Rumpfs zugeordnet (siehe Abbildung 2.3).

Damit lassen sich indirekte Einflüsse der Rumpfverformungen auf einzelne Oberflächenpunkte des Flügels nachvollziehen. Um die endgültige Verschiebung $\mathbf{u}$ eines CFD-Oberflächenpunkts zu ermitteln, wird diese nun anteilig aus den zuvor beschriebenen direkten und indirekten Zuordnungen ermittelt mit

$$u = \frac{1}{1 + w_a}\mathbf{u}_{dir} + \frac{w_a}{1 + w_a}\mathbf{u}_{indir} \tag{2.38}$$

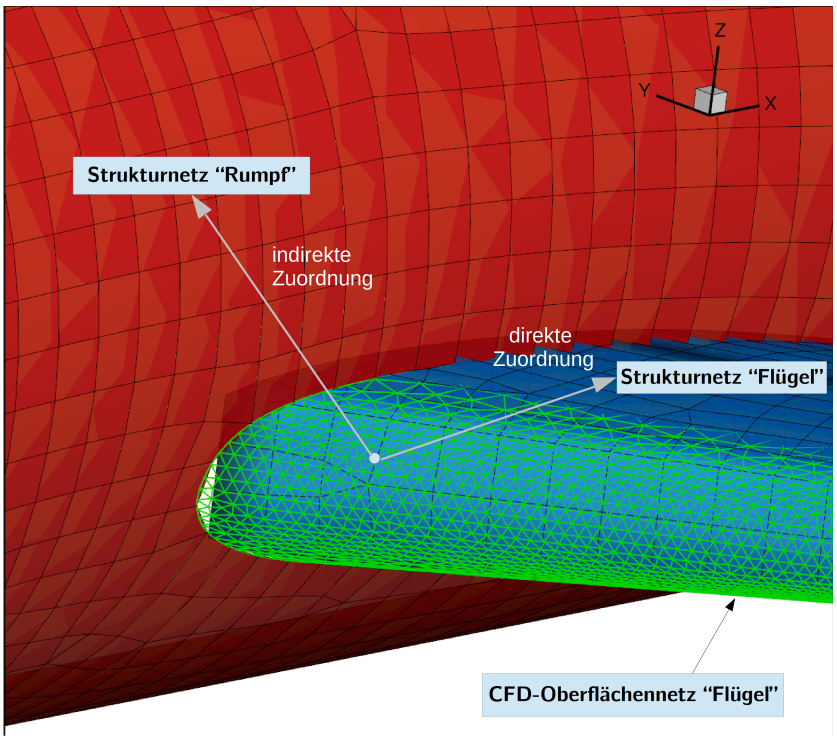


Abbildung 2.3: Bauteil-bezogene Zuordnung eines CFD-Oberflächenpunkts zu mehreren Komponenten

Gleichung 2.38 beinhaltet zunächst nur die Gewichtung der Verschiebung bei lediglich einer direkten und einer indirekten Zuordnung. Werden jedoch indirekte Einflüsse mehrerer Komponenten berücksichtigt, so ergibt sich:

$$u = \frac{1}{1 + \sum_j^n w_j} \mathbf{u}_{dir} + \sum_{j=1}^{n} \frac{w_j}{1 + \sum_j^n w_j} \mathbf{u}_{j,indir} \tag{2.39}$$

Die Faktoren w_j bestimmen sich aus dem gewichteten Abstand eines Punktes zur gemeinsamen Schnittkurve indirekt am Kopplungsprozess beteiligter Komponenten. Ausgehend vom euklidischen Abstand a des CFD-Oberflächenpunkts zur Schnittkurve wird ein maximaler Abstand zur Schnittkurve a_{max} definiert, bis zu dem Blending aktiv ist und indirekte Verformungseinflüsse berücksichtigt werden. Mathematisch führt dies zu einer Gewichtungsfunktion mit

$$w_a = w(x) \qquad mit \qquad x = \frac{a}{a_{max}} \tag{2.40}$$

und

$$w(x) = \begin{cases} 1 - 10x^3 + 15x^4 - 6x^5, & 0 < x < 1 \\ 0, & x > 1 \end{cases} \tag{2.41}$$

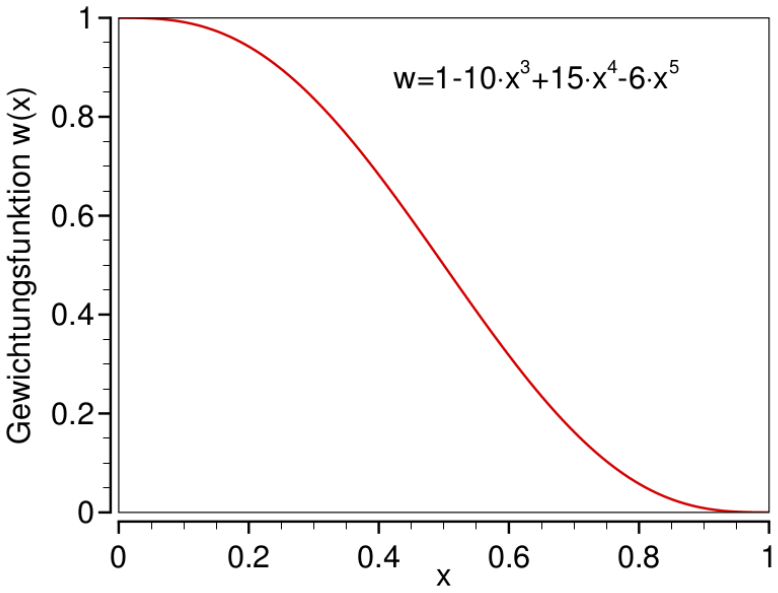


Abbildung 2.4: Beispiel einer Blending-Gewichtungsfunktion $w(x)$

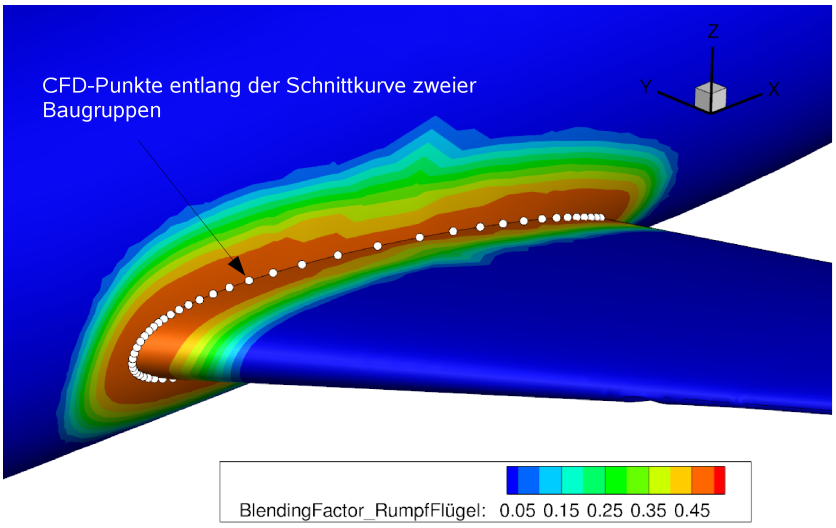


Abbildung 2.5: Blending-Bereich zwischen Rumpf und Flügel

Der Funktionsverlauf der Blendingfunktion $w(x)$ ist in Abb. 2.4 dargestellt.

Angewendet auf das Beispiel von Flügel und Rumpf bedeutet dies, dass für Oberflächenpunkte auf der gemeinsamen Schnittkurve (für diese gilt a=0) Verformungsanteile der umliegenden Flügel- und Rumpfstruktur zu gleichen Teilen berücksichtigt werden. Abbildung 2.5 verdeutlicht grafisch den zuvor beschriebenen Blending-Aspekt.

Bei der Interpolation von Lasten auf das Strukturnetz wird diese Vorgehensweise analog durchgeführt. Hierbei wird jedoch die Verteilung einer Einzellast auf mehrere Strukturkomponenten erreicht, um einen energetisch korrekten Lasttransfer zu ermöglichen. Die auf die einzelnen Komponenten zu übertragenen Lasten ergeben sich somit zu

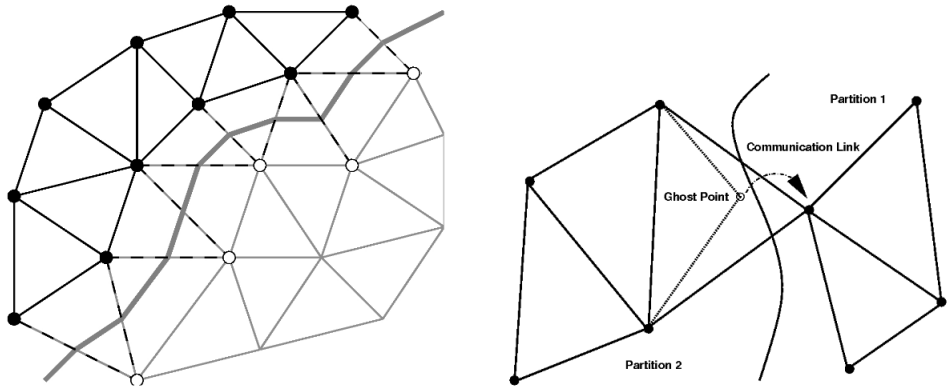
$$\mathbf{F} = \frac{1}{1 + \sum_j^n w_j}\mathbf{F}_{dir} + \frac{w_j}{1 + \sum_j^n w_j}\mathbf{F}_{j,indir} \tag{2.42}$$

2.5 Gebietszerlegungsmethoden

Die Analyse flugphysikalischer Phänomene benötigt heutzutage fein aufgelöste CFD-Gitter. Insbesondere zur adäquaten Auflösung turbulenter Strömungen werden zumeist mehrere Millionen Gitterpunkte benötigt. Dies erhöht jedoch nicht nur im erheblichen Maße den Berechnungsaufwand einer CFD-Berechnung, sondern den des gesamten Fluid-Struktur-Kopplungsprozess. Seit dem Aufkommen von Hochleistungsrechnern und vermehrtem Engagement im Bereich des parallelen Rechnens wurden Methoden entwickelt, mit deren Hilfe man diesem vermehrten Aufwand in gewissem Maße Herr werden kann. Da diese Methoden Bestandteil des in der Arbeit vorgestellten parallelen Last- und Verformungstransfers sind, wird auf sie im Folgenden näher eingegangen.

2.5.1 Begriffsdefinitionen

Gebietszerlegungsmethoden gehen einher mit der Lösung des sogenannten statischen Lastbalancierungsproblems, welches heutzutage als Indikator für effizienten parallelen Code angesehen wird. Kerngedanke zur Lösung dieses Problems ist die Aufteilung der diskretisierten Netze in separate Teile, den „Partitionen“, welche auf die einzelnen Prozessoren verteilt werden und dann quasi-unabhängig voneinander berechnet werden können. Für eine gute Balancierung des Problems ist es dabei von essentieller Bedeutung, eine gleichmäßige Aufteilung der Netzknoten auf alle Prozesse zu erreichen. Auf diese Weise werden die Wartezeiten minimiert. Neben der Lastbalancierung spielt desweiteren der Kommunikationsaufwand der Prozesse untereinander eine wichtige Rolle. Je nach verwendeten Lösungsalgorithmen und Diskretisierungsschemata werden Daten der Nachbarpartitionen benötigt, in Verbindung mit dieser Arbeit z. B. die Koordinaten der umliegenden CSM-Gitterpunkte



(a) Domain decomposition (b) Creation of ghost points

Abbildung 2.6: (a) Gebietszerlegung eines Tetraedernetzes und (b) Generierung eines Ghost-Points[10]

zu einem betrachteten CFD-Gitterpunkt. Für eine korrekte Ausführung der Lösermethoden wird dabei zwischen zwei Partitionen der Datenaustausch über eine gemeinsame Netzschicht, den „halo“ Bereich durchgeführt. Wird ein Netz aufgeteilt, so werden Punkte jeweils einem Prozess zugeordnet. Da aber zwischen zwei Punkten unterschiedlicher Prozesse ein Kante existieren kann, muss ein Punkt auf dem anderen Prozess kopiert werden. Dieser wird dort zu einem „Ghost Node“ und symbolisiert somit eine gemeinsame Netzgrenze mit einer anderen Partition. Abbildung 2.6 zeigt diesen Sachverhalt grafisch anhand eines einfachen Beispiels.

Nachfolgend werden zwei häufig verwendete Partitionierungsverfahren beschrieben: die Recursive Coordinate Bisection (RCB) Methode und eine graph-basierte Methode. Beide Methoden sind zudem im FlowSimulator-Framework implementiert und werden im Rahmen der numerischen Tests des parallelen Last-/Verformungstransfers mit der Quaranta-Methode in Kapitel 4 verwendet.

2.5.2 Recursive Coordinate Bisection Method

Die Recursive Coordinate Bisection Method stellt eine der einfachsten Gebietszerlegungsmethoden dar. Sie gehört zur Klasse der Geometrie-basierten Verfahren. Geometrisch bedeutet hierbei, dass bei der Netzaufteilung die physikalische Lage und Ausrichtung der Gitterpunkte mit ihren Koordinaten ausgenutzt wird. Die RCB-Methode unterteilt nun die Punktkoordinaten entlang einer Achse, bei der die räumliche Punktverteilung die größte ist. In den meisten Fällen ist dies zunächst die x-Achse. Die Teilung erfolgt dann in dem Maße, dass die Hälfte der Punkte zu einer Partition und der Rest der Punkte zu einer anderen Partition verteilt wird. Analog dazu wird im nächsten Schritt eine Unterteilung entlang der orthogonalen Achse durchgeführt, hier z. B. die y-Achse. Diese Vorgehensweise führt i.d.R. dann

zu guten Ergebnissen, wenn das Netz gleichmäßig über einen einfach strukturierten Bereich (z. B. Rechteck-förmiges Netz mit parallel zu den Koordinatenachsen verlaufenden Gittergrenzen) verteilt wird.

2.5.3 Graph-basierte Partitionierungsmethoden

Trotz der verhältnismäßig einfachen Punktaufteilung hat die RCB-Methode den Nachteil, dass der auftretende Kommunikationsaufwand zwischen den Prozessen und Partitionen durch große Partitionsgrenzen gekennzeichnet ist. Graph-basierte Partitionierungsmethoden versuchen hierbei, diesen Kommunikationsaufwand signifikant zu mindern. Formal betrachtet führt dies zur Lösung eines Graphen-Bisectionsproblems. Ein gegebenes Netz lässt sich dabei mithilfe eines ungerichteten Graphen G beschreiben, der eine Menge Knoten V besitzt, mit Kanten E zwischen diesen. Es wird nun nach einer Lösung in Form von k Knotenmengen gesucht, die

$$V_1 \cup \cdots \cup V_k = V \tag{2.43}$$

$$V_i \cap V_j = \emptyset \qquad \forall i \neq j \tag{2.44}$$

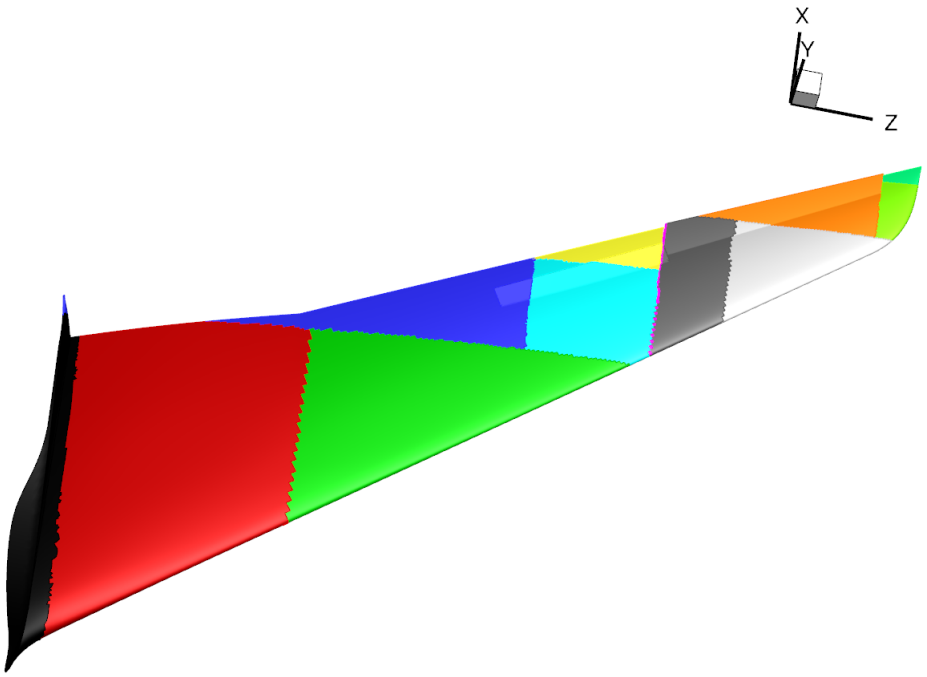
erfüllen. Im Allgemeinen existiert neben diesen Bedingungen die zusätzliche Bedingung, eine Partitionierung zu finden, die gleichzeitig eine Zielfunktion minimiert bzw. maximiert. Je nach Anwendungsfall gibt es zahlreiche Funktionen, darunter z. B.

$$totalv = \sum_{v \in V_b} Nadj(v) \quad , \tag{2.45}$$

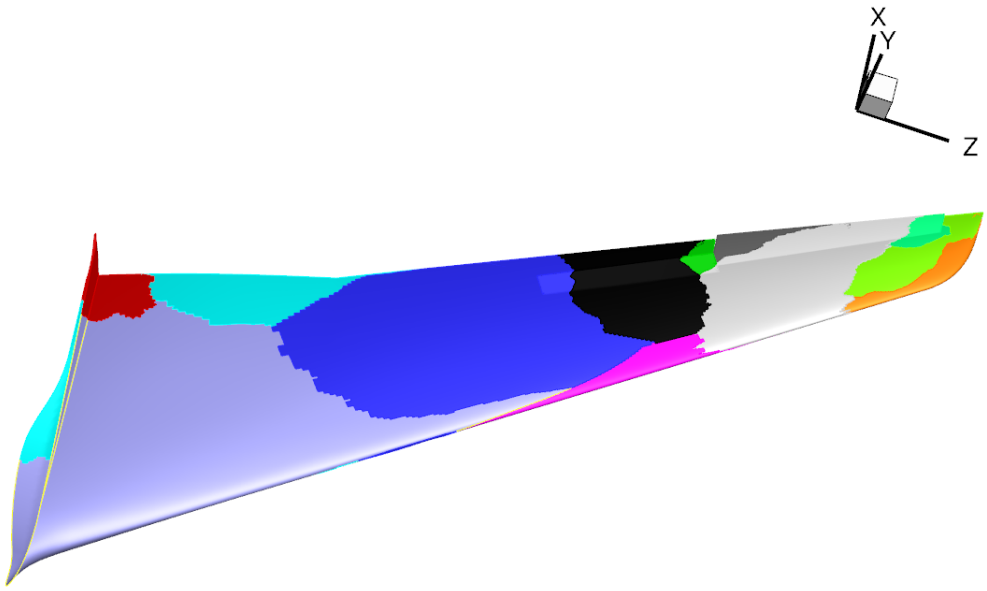
welche für eine mögliche Partitionierungslösung den resultierenden Gesamtkommunikationsaufwand bestimmt. v beschreibt hierbei einen Knoten in der Grenzschicht mehrerer Partitionen, welcher über gemeinsame Kanten mit Knoten anderer Partitionen verbunden ist. [11]

Populäre Bibliotheken, welche einen graph-basierten Ansatz der Partitionierung umsetzen, sind ParMetis, PTScotch und das Zoltan Toolkit. Letzteres wird im FlowSimulator-Framework unterstützt.

Für die im Rahmen dieser Arbeit untersuchte HiReTT-Konfiguration werden exemplarisch in der Abbildung 2.7 in unterschiedlicher Färbung die einzelnen Netzpartitionen gezeigt. Obwohl anhand von Abbildung 2.7 zunächst nicht erkennbar, so sind die durch die einzelnen Zerlegungsmethoden entstehenden Teilnetze in einigen Fällen nicht zusammenhängend. Abbildung 2.8 zeigt diesen Sachverhalt gezielt am Beispiel einer bestimmten graph-basierten Partitionierung der HiReTT Konfiguration. Die Möglichkeit nicht zusammenhängender Netzpartitionen stellt ein nicht zu vernachlässigendes Problem bei der Parallelisierung der in Kapitel 2.3 beschriebenen Quaranta-Methode dar.



(a) RCB-Partitionierung eines rückwärts gepfeilten Flügelnetzes



(b) Graph-basierte Partitionierung eines rückwärts gepfeilten Flügelnetzes mithilfe der Zoltan Bibliothek

Abbildung 2.7: Grafische Gegenüberstellung unterschiedlicher Partitionierungen des HiReTT-Flügels

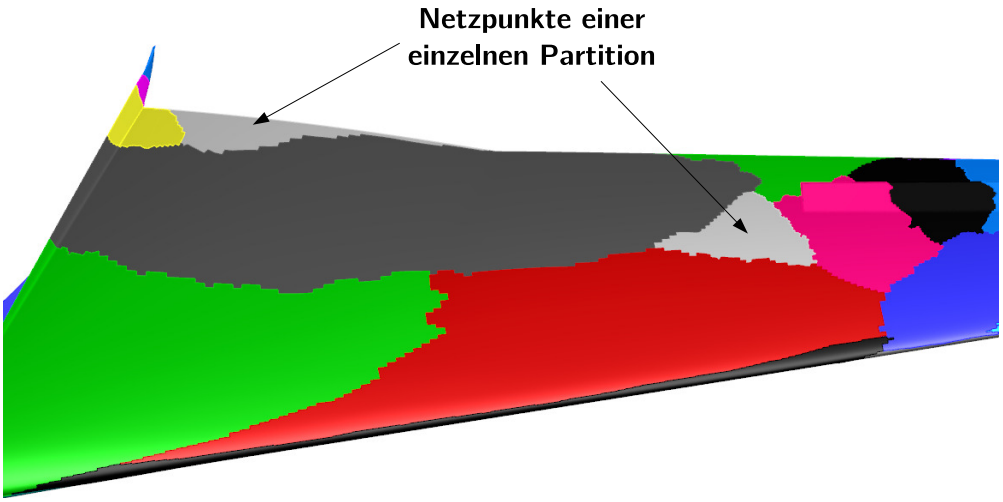


Abbildung 2.8: Darstellung nicht-zusammenhänger Partitionen

2.6 Datenstrukturen zur Stützstellensuche

Da die Quaranta-Methode mehrere lokale Stützstellen zur Interpolation jedes CFD-Oberflächenpunktes erfordert, ist eine effiziente Bestimmung der lokal nächsten Punkte essentiell.

Zur Suche der nächsten Nachbarn eines beliebigen Punktes ist eine Brute-Force Methodik zunächst unangemessen, da dies je nach Größe der Netze und Anzahl der Stützstellen inakzeptabel viel Zeit in Anspruch nehmen würde. Zu diesem Zweck existieren verschiedene (räumliche) Datenstrukturen, die eine effizientere Stützstellensuche ermöglichen. Einige populäre Datenstrukturen sind im Folgenden aufgelistet:

- R-Baum:
  Mehrdimensionale, balancierte Indexstruktur. Nahe beieinander liegende räumliche Punkte werden mithilfe eines minimal umgebenden Rechtecks beschrieben und in einem Baum einsortiert. Alle Teilrechtecke ergeben auf oberster Knotenebene des Baums ein Rechteck, in dem alle Punkte liegen.

- B-Baum
  Vollständig, balancierter Baum, ähnlich zum R-Baum. Im Gegensatz zu einem normalen Baum kann ein Knoten mehr als 2 Kind-Knoten besitzen. Somit lässt sich die Anzahl der zu lesenden Knoten bei einer Suchanfrage reduzieren.

- BSP-Baum (Binary Space Partitioning)
  Weitverbreitete Technik zur Unterteilung räumlicher Objekte in virtuellen Welten. Die Unterteilung der Punktdaten erfolgt mittels Hyperebenen, welche die Menge in zwei räumliche Teilmengen unterteilt. Die Unterteilung wird

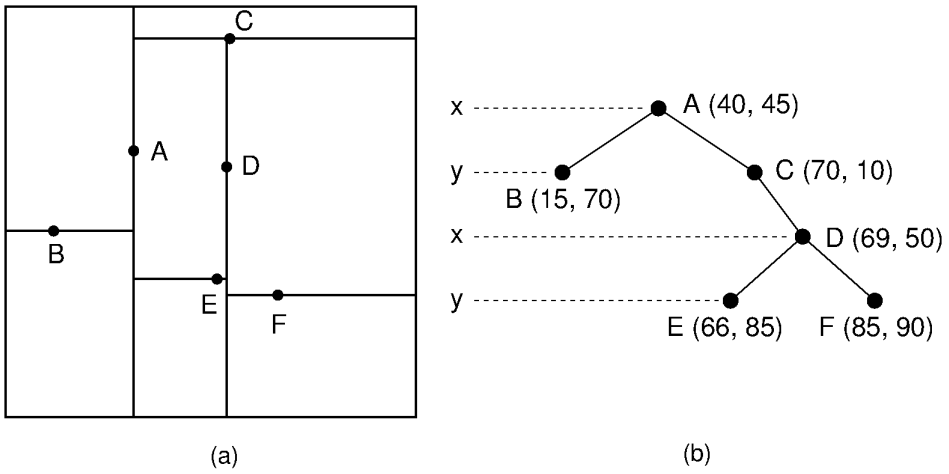


Abbildung 2.9: k-d-Baum Generierung eines einfachen 2D-Datensatzes. Hierbei dienen zugleich die einzelnen Punkte als Hyperebene bzw. Trenngeraden

rekursiv weitergeführt und die entstehen Trennungsebenen werden als Knoten in einen Baum eingefügt.

Für die Realisierung einer effizienten Stützstellen-Suche innerhalb des Quaranta-Algorithmus wird im Weiteren auf den sogenannten k-d-Baum zurückgegriffen. Denn eine Implementierung des k-d-Baums im FlowSimulator-Framework besteht bereits.

2.6.1 k-d-Baum

Bei einem k-d-Baum handelt es sich, ähnlich zu den zuvor vorgestellten Methoden, um eine räumliche Datenstruktur zur Speicherung von k-dimensionalen Datensätzen. Obwohl ein k-d-Baum, genau wie viele andere Strukturen, mit zunehmender Dimensionszahl ineffzienter wird, ist er insbesondere bei 3D-Datensätzen, wie sie im Allgemeinen bei Strukturnetzen vorherrschen, eine praktikable Suchstruktur.

Im Allgemeinen basiert ein k-d-Baum auf einer achsenparallelen Unterteilung des Datenraums. Dazu wird abwechselnd entlang den k-Dimensionen eine Hyperebene erstellt, die die vorliegende Datenmenge annähernd in zwei gleichgroße, disjunkte Teilmengen aufteilt. Die ermittelte Hyperebene bzw. ihr repräsentierender Koordinatenwert wird dabei als Knoten t in einem Baum eingetragen. Im linken Teilbaum werden dann alle Punkte eingefügt, deren Koordinate kleiner als der Knoten t ist. Je nach verwendeter Trennungsebene in den einzelnen Dimensionen kann ein k-d-Baum somit im Vergleich z. B. mit einem R-Baum unbalanciert sein. Abbildung 2.9 zeigt beispielhaft eine Unterteilung eines 2D-Punktdatensatzes mittels einer k-d-Baumstruktur[2].

[2]Bildquelle: `http://algoviz.org/OpenDSA/dev/OpenDSA/Modules/KDtree.odsa`

Nächste-Nachbarn-Suche im k-d-Baum

Eine Suche der nächsten Nachbarn lässt sich nun nach einmaliger Erstellung der Baumstruktur anschließend effizient realisieren. Der Basis-Suchalgorithmus sucht dabei zunächst allerdings nur den direkt nächsten Nachbarn. Eine Erweiterung auf n-nächste Nachbarn lässt sich aber mit wenigen Mitteln realisieren. Die Suche nach dem direkt nächsten Nachbarn wird folgendermaßen durchgeführt:

1. Für einen gegebenen Punkt wird vom Wurzelknoten rekursiv der Baum durchlaufen, als wenn dieser Punkt in den Baum eingefügt wird. Hierzu wird an jedem Knoten überprüft, ob die Koordinate des Punktes kleiner oder größer als die des Wurzelknotens in der aktuellen Trenndimension ist und je nach Ergebnis im linken oder rechten Teilbaum rekursiv fortgefahren.
2. Ist man an einem Blatt angekommen, so speichert man diesen als aktuell besten Kandidat
3. Der Baum wird nun ein zweites Mal durchlaufen, jedoch aufsteigend beginnend von den zuvor gefundenen Blattknoten. In jedem Knoten wird folgendes überprüft:
   a) Ist der Knoten näher als der aktuell beste Kandidat, so macht man diesen Knoten zum neuen Kandidaten. Da der Knoten eine Trennebene repräsentiert, muss nun überprüft werden, ob Punkte auf der anderen Seite der Trenngeraden liegen, die näher als der bisher beste gefundene Kandidat sind. Dazu wird mithilfe des Abstandes zum bisher besten Kandidaten eine Hyperkugel erzeugt, um gemeinsame Schnittpunkte mit der Trenngeraden bzw. Hyperebene zu ermitteln. Existieren Schnittpunkte, so muss der Algorithmus auch den anderen Teilbaum von aktuellen Knoten aus betrachten.
4. Ist der aktuelle Knoten der Wurzelknoten und ist dieser nicht näher als der bisher beste Kandidat, so ist die Suche beendet.

Um eine Suche nach den n-nächsten Nachbarn durchzuführen, müssen nun anstelle eines besten Kandidaten n Kandidaten ermittelt werden. Ein Teilbaum wird dabei erst dann nicht mehr betrachtet, wenn er keine näheren Punkte besitzt als irgendeiner der aktuell besten Kandidaten.

3 Numerische Umsetzung der Quaranta-Methode

In diesem Kapitel wird neben algorithmischen Anforderungen der Quaranta-Methode insbesondere auf die Umsetzung der in der Arbeit verfolgten Parallelisierung der Methode eingegangen. Kapitel 3.1 beschreibt zunächst anhand der in Kapitel 2 geschilderten theoretischen Grundlagen den Gesamtalgorithmus für einen Bauteil-basierten Last- und Verformungstransfer innerhalb eines Fluid-Struktur-Kopplungsschritts. Kapitel 3.2 befasst sich im Besonderen mit der für die Quaranta-Methode möglichst effizienten, parallelen Suche nach Stützstellen zur Interpolation. Kapitel 3.3 setzt sich darüber hinaus mit möglichen Ansätzen zur effizienten Speicherung der charakteristischen Kopplungsmatrix H auseinander. Kapitel 3.4 beschreibt abschließend technische Detailaspekte der parallelen Kommunikation.

3.1 Übersicht über durchzuführende Prozessschritte

Die in Kapitel 2.3 geschilderten Grundlagen der Quaranta-basierten Interpolationsmethode und die Berücksichtigung der Blending-Methoden aus Kapitel 2.4 für einen konservativen Last- und Verformungstransfer lassen sich nun in einem Gesamtalgorithmus vereinen. Ausgehend von den berechneten aerodynamischen Kräften einer CFD-Lösung im i-ten Kopplungsschritt beinhaltet der Algorithmus die nachfolgend beschriebenen Schritte zur Überführung des unverformten CFD-Oberflächennetzes in eine neue, verformte Lage des CFD-Oberflächennetzes. Der Algorithmus ist in Abbildung 3.1 dargestellt und besteht dabei im Wesentlichen aus folgenden Punkten:

1. Blending-Präprozess:
   Der Blending-Präprozess beinhaltet alle verwaltungstechnischen Schritte, die zur Berechnung der einzelnen Blending-Faktoren auf der Basis der unverformten CFD- und CSM-Kopplungsnetze innerhalb der FlowSimulator/FSCouple-Umgebung notwendig sind.

2. Berechnung der Blending-Faktoren für jeden CFD-Oberflächenpunkt:
   Die Berechnung der Faktoren wird gemäß Kapitel 2.4 auf Basis der im Präprozess berechneten Schnittkurven bestimmt.

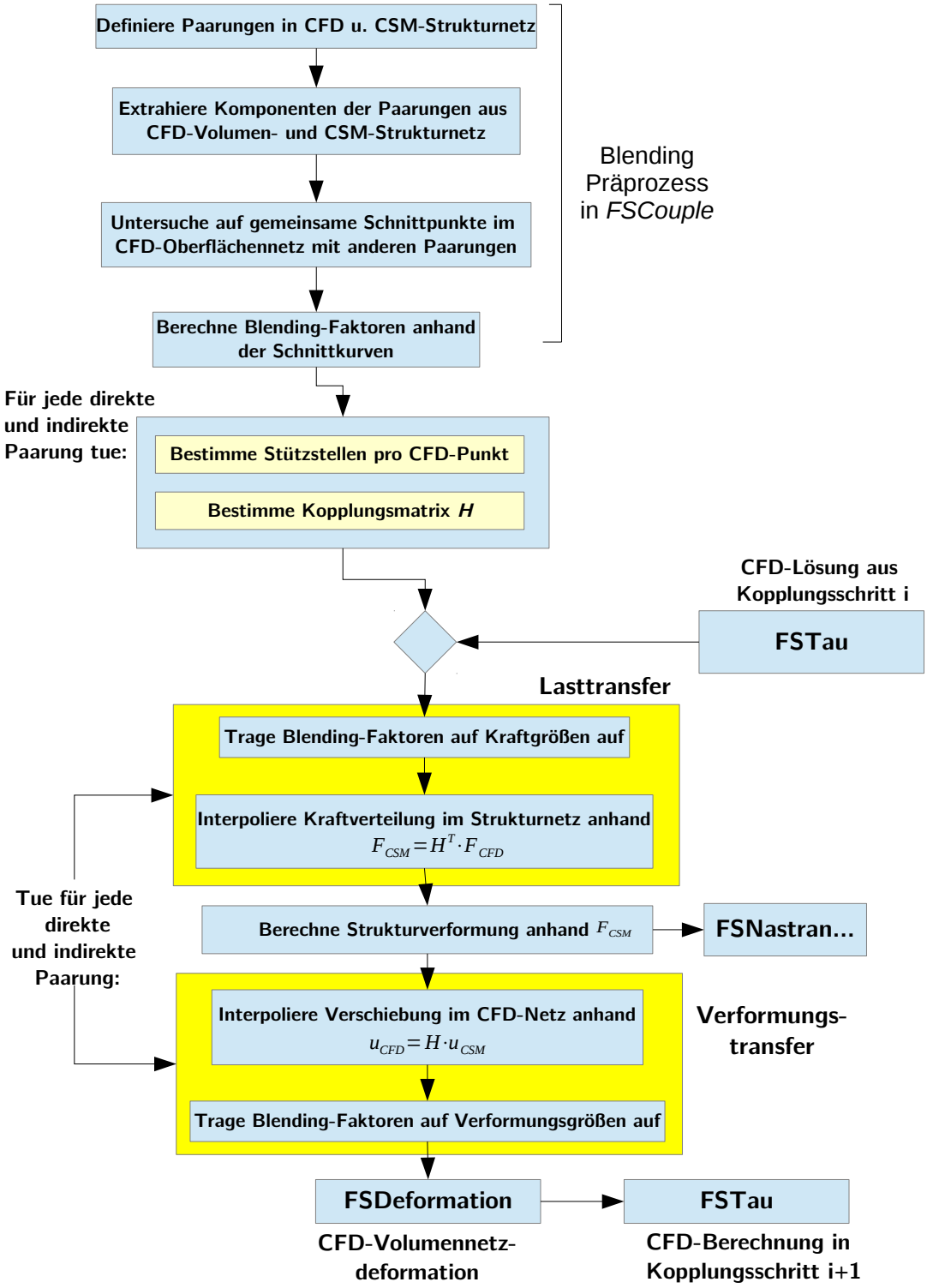


Abbildung 3.1: Algorithmus zur räumlichen Kopplung

3. Bestimmung der Kopplungsmatrix für jede direkte und indirekte Paarung:
   Die Berechnung beinhaltet dabei die Ermittlung der Stützstellen und Berechnung der RBF-basierten Koeffizienten mit der Quaranta-Methode. Beide Teilschritte werden in Abschnitt 3.2 als Bestandteil des umgesetzten parallelen Algorithmus näher betrachtet. Die Bezeichnung der Paarung greift das in Kapitel 2.4 vorgestellte Konzept direkter und indirekter Netzzuordnungen im *FSCouple*-Modul algorithmisch auf.

4. Auftragen der Blending-Faktoren innerhalb des Lasttransfers:
   Das Auftragen der errechneten Blending-Faktoren $Blend_i$ auf jede Lastgröße eines beliebigen CFD-Oberflächenpunkts i einer Paarung erfolgt mit

$$\tilde{F}_{CFD,i} = Blend_i * F_{CFD,i} \tag{3.1}$$

5. Interpolation der Kraftverteilung im CSM-Strukturnetz:
   Die Interpolation der aerodynamischen Kräfteverteilung auf dem Strukturnetz erfolgt dabei auf Basis der in Schritt 3 berechneten Kopplungsmatrix H

$$F_{CSM} = H^T \cdot \tilde{F}_{CFD} \tag{3.2}$$

6. Berechnung der Strukturverformung u_{CSM} auf Basis der interpolierten Lasten F_{CSM}:
   Die Berechnung erfolgt mithilfe externer FEM-Simulationstools (z.B. NASTRAN). Dieser Prozessschritt wird in dieser Arbeit nicht näher betrachtet.

7. Interpolation der Verschiebungen u_{CFD} des CFD-Oberflächennetzes
   Analog zu Schritt 5 erfolgt die Übertragung der Verschiebungen vom Strukturnetz auf das CFD-Oberflächennetz mittels

$$u_{CFD} = H \cdot u_{CSM} \tag{3.3}$$

8. Auftragen der Blending-Faktoren innerhalb des Verformungstransfers:
   Im Gegensatz zu Schritt 4 werden die Blending-Faktoren des Verformungstransfers innerhalb desselben erst **nach** der Interpolation der CFD-Oberflächenverschiebungen vorgenommen:

$$\tilde{u}_{CFD,i} = Blend_i * u_{CFD,i} \tag{3.4}$$

Im Rahmen der Stützstellen-Bestimmung (Unterpunkt 3 im oben beschriebenen Prozessablauf) wird der in Gleichung 2.8 gewählte Ansatz berücksichtigt, sodass stets für einen Oberflächenpunkt des Fluidnetzes die Stützstellen auf Strukturnetz-Seite gesucht werden.

Da die Stützstellensuche und entsprechende Teilschritte innerhalb des Blending-Präprozess auf den unverformten, initialen Netzen durchgeführt wird, bedarf dies im gesamten Kopplungsalgorithmus lediglich einer einmaligen Bestimmung. Der

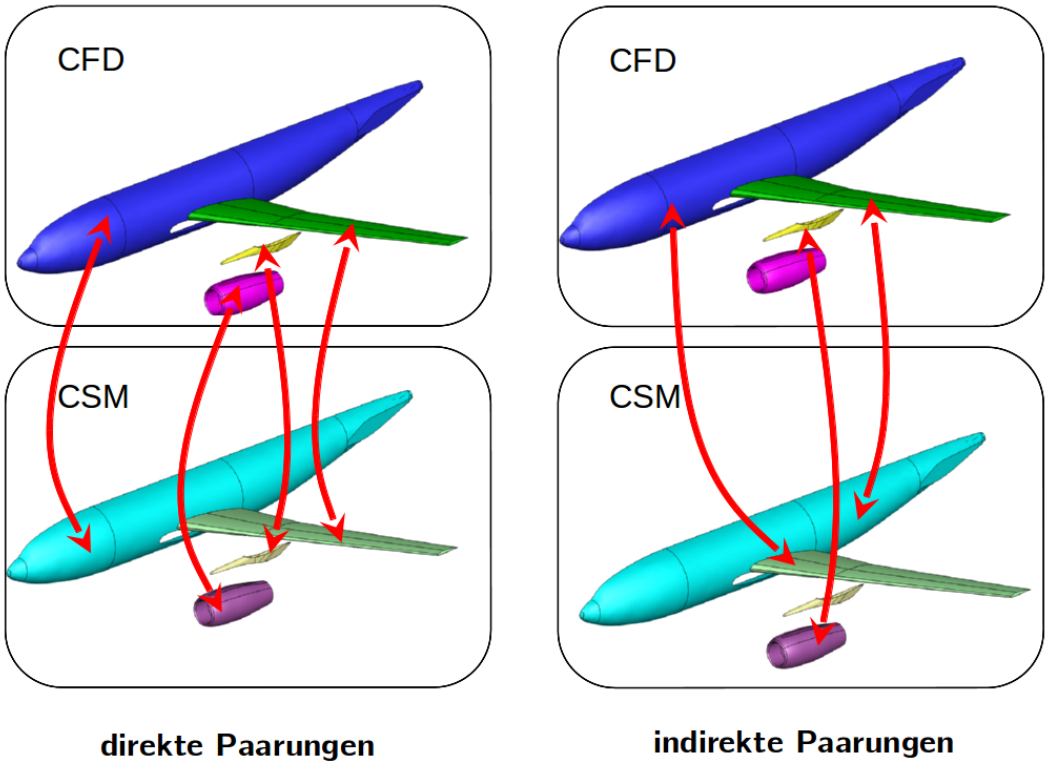


Abbildung 3.2: Gegenüberstellung ausgewählter direkter und indirekter Paarungen bei Bauteil-bezogener Kopplung einer einfachen Flugzeugkonfiguration

eigentliche Last- und Verformungstransfer sowie das Auftragen der Blending-Faktoren auf Verformungs- und Lastgrößen muss dagegen in jedem Iterationsschritt durchgeführt werden.

Blending-Präprozess

Gemäß der zuvor beschriebenen Auflistung muss vor Beginn der einzelnen Kopplungsteilschritte eine Initialisierung des Bauteil-basierten Gesamtprozesses durchgeführt werden. Für eine Bauteil-bezogene Kopplung müssen zuallererst Paarungen definiert werden. Eine Paarung bezeichnet dabei die Verknüpfung einer Teilkomponente des gesamten CFD-Oberflächennetzes mit einer Teilkomponente des CSM-Strukturnetzes. Zwischen beiden Komponenten findet dann innerhalb der räumlichen Kopplung ein Last- und Verformungstransfer mithilfe vom Nutzer ausgewählten Interpolationsmethoden statt.

Zunächst werden nur „direkte“ Paarungen erzeugt, d.h. es werden räumlich nächste Netze zugeordnet, welche den größten Last- und Verformungseinfluss zueinander haben. Auf Basis der definierten Paarungen werden aus den kompletten CFD- und CSM-Netzen die sich ergebenden Teilnetze extrahiert. Dies geschieht technisch innerhalb des FlowSimulator mithilfe spezieller Knotenattribute, mit denen

Netz-Selektionen durchgeführt werden können. Da für einen korrekten Bauteil-bezogenen Transfer Blending-Methoden erforderlich sind, muss nun für jede direkte Paarung überprüft werden, ob Schnittpunkte mit anderen Paarungen existieren. Die Generierung möglicher Schnittkurven und infolgedessen „indirekter“ Paarungen wird stets auf dem CFD-Oberflächennetz durchgeführt, da das Netz sowohl die Eingangsdaten hinsichtlich der Lastverteilung als auch die Zielstruktur bei der Verformungsinterpolation darstellt. Zudem kann aufgrund unterschiedlicher Netztypen der einzelnen Strukturkomponenten kein zusammenhängendes Strukturnetz sichergestellt werden. Entsprechende Blendingfaktoren werden deshalb auf den einzelnen CFD-Oberflächenpunkten gespeichert. Ergebnis der Schnittkurven-Untersuchung sind insgesamt neben allen direkten neu erzeugten indirekten Paarungen. Ausgewählte Beispiele indirekter und direkter Paarungen anhand einer beispielhaften Flugzeug-Konfiguration sind in Abbildung 3.2 zu sehen. Für jede Paarung sind entsprechend der Abbildung 3.1 daraufhin die einzelnen Last- und Verformungstransferschritte durchzuführen und die dazugehörenden Kopplungsmatrizen zu berechnen. Für eine korrekte Funktionalität müssen die interpolierten Lasten und Verschiebungen mithilfe von Gleichung 2.38 und 2.42 (Aufsummation der gewichteten Verformungsbeiträge bzw. Berechnung der gewichteten Lastverteilung) abschließend zusammengeführt werden.

Je nach Anzahl verwendeter Komponenten bzw. Paarungen stellt die Ermittlung der einzelnen Kopplungsmatrizen den berechnungsintensivsten Schritt des vorgestellten Algorithmus dar. Die parallele Bestimmung der Matrizen soll daher im Weiteren detailliert beschrieben werden.

3.2 Parallele Stützstellensuche

Nach ausgiebiger Literaturrecherche konnte keine Veröffentlichung gefunden werden, die einen Parallelisierungsansatz der Quaranta-Methode vorstellt. Im weiteren wird deshalb ein mögliches Vorgehen zunächst motiviert und anschließend die Umsetzung ausgewählter Ansätze erörtert.

Die Hauptaufgabe bei der parallelen Umsetzung der Quaranta-Methode besteht in der verteilten Suche nach Stützstellen im CSM-Strukturnetz. Dieser Vorgang wird im Weiteren auch als „Nächste-Nachbarn-Suche“ und die zu suchenden Stützstellen als „Nächste-Nachbarknoten“ (NN) bezeichnet. Die Nächste-Nachbarknoten in direkter Umgebung des CFD-Punkts können infolge der Partitionierung der Kopplungsnetze jedoch nicht durch lokale Operationen, d.h. Operationen auf dem selben Prozess, ermittelt werden.[1]

Ohne die Berücksichtigung der vorhandenen Partitionen und Netzinformationen (z.B. Ausdehnung der diskretisierten CFD- und CSM-Netze entlang der Koordi-

[1] Im Weiteren bezeichnen jeweils Prozesse die rechnertechnische parallele Ausführung des Last- und Verformungstransfers, Partitionen die Menge einzelner Netzpunkte auf den Prozessen

natenachsen) kann zunächst ein naiver Ansatz zur parallelen Bestimmung der Nächsten-Nachbarn definiert werden. Dieser basiert im wesentlichen darauf, für einen gegebenen CFD-Punkt und einer zuvor definierten Stützstellenanzahl $nNext$ von **allen** CSM-Partitionen deren gesamte Knotenmenge zu ermitteln (und somit auf jedem Prozess eine Kopie des kompletten CSM-Netzes zu erzeugen). Aus dieser Knotenmenge würde dann daraus eine Auswahl der $nNext$ Punkte mit dem kürzesten, euklidischen Abstand zum CFD-Oberflächenpunkt ermittelt. Bei großen Netzen und Stützstellenanzahlen erfordert dies aber einen nicht zu vernachlässigenden Speicher- und parallelen Kommunikationsaufwand. Dieser Ansatz ist daher für eine effiziente Parallelisierung nicht praktikabel und wird daher im Weiteren nicht weiter verfolgt. Es wurden vielmehr Ansätze verfolgt, die die lokalen Netzeigenschaften, d.h. die Verteilung der Partitionen, gezielt in speziellen Datenstrukturen berücksichtigen. Dies führt von einer globalen zu einer lokalen Betrachtung des parallelen Problems.

3.2.1 Ansätze der parallelen Nächsten-Nachbarn-Suche

Bevor auf den in der Arbeit umgesetzten parallelen Ansatz der verteilten Nächsten-Nachbarn-Suche eingegangen wird, sollen kurz in der Literatur vorhandene Methoden, die diesem Zweck dienen, beschrieben werden. *Verteilt* bezieht sich hierbei auf die Trennung der Punkte in logisch voneinander getrennten Untermengen, wie sie z.B. durch Gebietszerlegungen generiert werden. Ausgewählte Konzepte werden nachfolgend kurz vorgestellt.

[12] beschreibt ein Verfahren zur effizienten Suche mithilfe eines verteilten R-Baums, welcher alle potentiellen Nachbarn beinhaltet. Auf Basis eines Suchradius um einen Punkt können mithilfe von speziellen Distanzmetriken einzelne Knoten bzw. Bereiche des R-Baums und somit potentielle Kandidatenpunkte frühzeitig verworfen werden. Das Verfahren führt eine Suche iterativ durch, bis alle exakten geometrischen Nachbarn ermittelt wurden. Diese Methodik basiert jedoch auf einem Single-Processor-Multi-Disk Ansatz, sodass Punkte zwar verteilt in einzelnen „Disks“ organisiert, aber für einen Prozess zunächst komplett zugänglich sind. Die verteilte R-Baumstruktur wird innerhalb des Ansatzes auf Basis der kompletten Punktmenge generiert und dessen einzelne Punkte für eine optimale Nachbarsuche effizient verteilt. Aufgrund des komplexen Aufbaus der verteilten R-Baumstruktur eignet sich dieser Ansatz nicht im Kontext der Arbeit, bei dem die Punkte schon vorverteilt sind.

[13] beschreibt eine ähnliche Methodik. Sie basiert ebenfalls auf einem R-Baum. Im Gegensatz zu [12] ist diese Vorgehensweise aber auf Mehrkern-Anwendungen zugeschnitten, die über ein entsprechendes Netzwerk miteinander kommunizieren. Die durchgeführten Analysen in [13] wurden jedoch nur auf zweidimensionale Datensätze mit moderater Punktanzahl bezogen, sodass keine Aussagen für höhere Dimensionsanzahl der Daten getroffen werden können. Zudem wurden nur geringe Prozessanzahlen in [13] untersucht. Nachteilig stellt sich hierbei auch das zum Ein-

satz kommende Master-Slave Konzept dar, bei dem Nachbarschaftssuchen stets von einem Hauptprozess an mehrere andere Prozesse versendet werden, dieser „Master“-Prozess selber aber keine Punkte beinhaltet. Lediglich die R-Baum-Datenstruktur ist auf dem Master-Prozess gespeichert. In dieser Arbeit wird jedoch von gleichberechtigten Prozessen ausgegangen, bei denen jeder Prozess Nachbarpunkte besitzt und Nächste-Nachbarn-Suchanfragen innerhalb der parallelen Kommunikation an andere Prozesse sendet sowie Nachbar-Punkte beinhaltet.

In [14] kommt eine B-Baum-basierte Methode zum Einsatz. Dabei werden einzelne Punkte innerhalb einer Teilmenge zunächst nach einem bestimmten Distanzmaß zum Zentrum der Punkte vorsortiert. Obere und untere Begrenzung hinsichtlich dieses Maßes werden daraufhin in einem eindimensionalen B-Baum einsortiert. Für einen gegebenen Suchpunkt und definierten Suchradius können somit Teilpunktemengen frühzeitig verworfen werden. Auch hier wird iterativ vorgegangen bzw. der Suchradius verändert, bis die geforderte Anzahl geometrischer Nächster-Nachbarn ermittelt werden kann. Das Verfahren ist in [14] jedoch nur in einer sequentiellen Anwendung getestet worden.

Aufgrund des Einsatzgebiets der beschriebenen Ansätze im Bereich von Datenbanken, bei der Nachbarschaftssuchen i. d. R. über einen Prozess gesteuert werden, lassen sich die in der Literatur vorhandenen Konzepte nur bedingt im Kontext der Arbeit anwenden. Der zu entwickelnde Algorithmus in der Arbeit muss demgegenüber eine effiziente Ausführung konkurrierender Nächste-Nachbarn-Suche aller Prozesse untereinander gewährleisten können. Konkurrierend bedeutet dabei die zeitgleiche Ausführung der Stützstellen-Suche für alle verteilten CFD-Oberflächenpunkte während der Berechnung der Kopplungsmatrix. Ein im Vergleich zur Brute-Force-Methodik effizienterer Ansatz wird im Folgenden beschrieben.

3.2.2 Wahl eines geeigneten parallelen Ansatzes

Bevor auf die wesentlichen Details der gewählten Parallelisierungsmethodik eingegangen wird, lässt sich der entwickelte Algorithmus schrittweise anhand von Abbildung 3.3 grafisch darstellen.

Im Vergleich zu den zuvor in Abschnitt 3.2.1 beschriebenen Methoden findet eine Stützstellensuche innerhalb der Arbeit für einen gegebenen CFD-Oberflächenpunkt zunächst nicht iterativ, sondern pro Punkt einmalig statt. Eine iterative Ermittlung und Kommunikation der verteilten, nächsten Nachbarn zwischen den Prozessen ist hierbei hinderlich, da dies zwingend mit einer Synchronisation der verteilten Stützstellensuche auf den einzelnen Prozessen einhergeht. Dies würde insbesondere eine komplexe und fehleranfällige Kommunikationslogik erfordern. Stattdessen werden einzelne berechnungsintensive Teilschritte lokal auf den jeweiligen Prozessen durchgeführt und notwendige Kommunikationen an einzeln festgelegten Stellen innerhalb der sequentiellen Ausführung durchgeführt. Technische Details hinsichtlich der Prozess-Kommunikation werden in Abschnitt 3.4 näher erläutert.

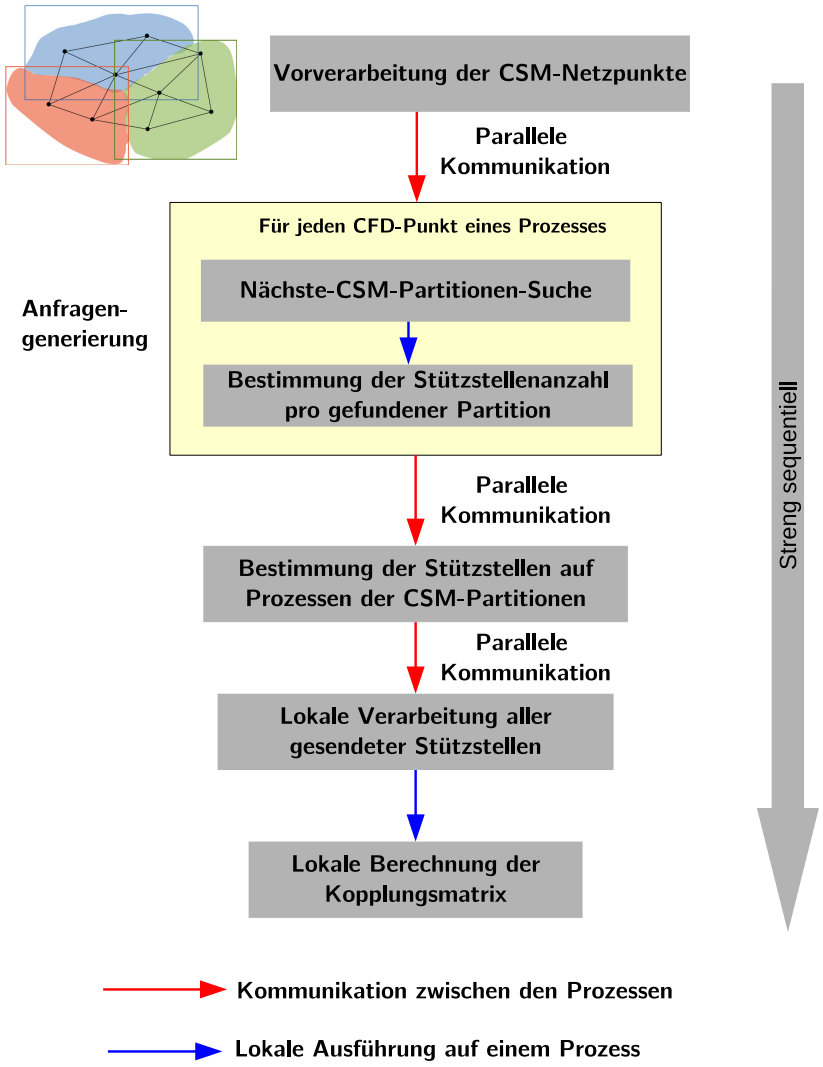


Abbildung 3.3: Schematische Darstellung des umgesetzten, parallelen Algorithmus

Obwohl die zuvor beschriebenen Ansätze nur bedingt auf die in der Arbeit geschilderte Thematik anwendbar sind, haben sie jedoch die Gemeinsamkeit, dass sie die vorhandene Punktmenge mithilfe einer räumlichen Datenstruktur (z.B. R-Baum) vorverarbeiten. Diese ist dabei unabdingbar, um als erste gute Annäherung lokale Punktbereiche (im Kontext dieser Arbeit die nächst gelegene CSM-Strukturnetzpartition) zu identifizieren, indem potentielle Stützstellen liegen. Diese ***Vorverarbeitung der CSM-Netzpunkte*** wird auch im umgesetzten Algorithmus verfolgt und stellt den initialen Schritt in Abb. 3.3 dar. Bei der Vielzahl möglicher Gebietszerlegungsmethoden, insbesondere bei Graph-basierten Ansätzen, können die verteilten CSM-Teilnetze eine unregelmäßige Form besitzen. Dies erschwert die Suche nach einer geeigneten abstrakten Datenstruktur

der CSM-Partitionen. Die einfachste Methode ist die Beschreibung der Strukturnetzpartitionen mithilfe eines umhüllenden Körpers. Der folgende Abschnitt *Vorverarbeitung der CSM-Netzpunkte* (S. 43) setzt sich näher mit unterschiedlichen Hüllkörper-Konzepten auseinander. Auf Basis der errechneten Hüllkörper werden diese im umgesetzten Algorithmus in einem k-d-Baum eingefügt und untereinander ausgetauscht. Jeder Prozess verfügt daraufhin über Informationen zur räumlichen Partitionsverteilung aller CSM-Teilnetze. Ohne diese Struktur müssten mithilfe einer Brute-Force Methodik alle Prozesse untereinander kommunizieren, um für einen CFD-Punkt auf einer lokalen Partition die räumlich nächsten CSM-Partitionen zu bestimmen.

Nach Generierung der räumlichen Datenstruktur der Partitionen kann für alle lokal auf einem Prozess vorhandenen CFD-Oberflächennetzpunke die Stützstellensuche durchgeführt werden. Ergebnis dieser Suche ist zum einen die formelle Verknüpfung eines CFD-Oberflächenknotens auf einem Prozess mit mehreren geometrisch nächsten CSM-Strukturnetzpartitionen, welche bei der ***Nächste-CSM-Partitionen-Suche*** ermittelt wurden. Zum anderen wird daraufhin festgelegt, wie viel nächste Nachbarpunkte auf den ermittelten Partitionen als potentielle Stützstellen betrachtet werden sollen. Beide Teilschritte werden in Suchanfragen gespeichert und zwischen den Prozessen untereinander ausgetauscht. Auf Basis der Anfragen erfolgt dann auf den CSM-Netzpunkten der einzelnen Prozesse die Bestimmung der nächsten Nachbarn.

Da der gesamte Ablauf der Stützstellensuche einmalig durchgeführt werden muss, müssen für eine korrekte Bestimmung der Stützstellen u.U. mehr Daten ausgetauscht werden. Aus diesem Grund muss vor der lokalen Berechnung der Kopplungsmatrix eine ***lokale Verarbeitung aller gesendeter Stützstellen*** auf einem Prozess durchgeführt werden. Nach Ermittlung der Stützstellen im Rahmen der parallelen Kommunikation werden gemäß dem Quaranta-Ansatz aus Kapitel 2.3 für alle lokalen CFD-Punkte einer CFD-Partition die Interpolationskoeffizienten der Kopplungsmatrix H berechnet.

Auf einzelne Teilaspekte des beschriebenen Ansatzes soll im Weiteren etwas detaillierter eingegangen werden.

Vorverarbeitung der CSM-Netzpunkte

Wie zuvor im Grundalgorithmus geschildert, wird beginnend für die CSM-Strukturnetzpartitionen ein geeigneter Hüllkörper ermittelt. Das einfachste Beispiel eines umhüllenden Körpers ist die sogenannte Bounding-Box. Diese wird zugleich auch zur Beschreibung der einzelnen Partitionen innerhalb der Arbeit benutzt. Ähnlich wie ein *Minimum-Bounding-Rectangle* (MBR), welches bei R-Bäumen zum Einsatz kommt, beschreibt es die Ausdehnung der umschließenden Punkte in Richtung der Koordinatenachsen. Dabei kann eine Bounding-Box durch zwei Vektoren beschrieben werden, welche die minimalen- bzw. maximalen Koordinaten

der umschlossenen Punkte entlang der Achsen beschreiben. Geometrisch betrachtet beschreiben diese Vektoren zwei gegenüberliegende Ecken der Bounding-Box. Algorithmisch lässt sich eine Bounding-Box in $\mathcal{O}(\#AnzahlPunkte)$ ermitteln. Obwohl vom zugrunde liegenden Strukturnetz abhängig, stellt eine Bounding-Box bei RCB-basierten Partitionierungen eine gute erste Annäherung der Partitionsform dar. Neben achsen-parallelen Boxen kann außerdem die Orientierung der Punkte berücksichtigt werden („Oriented Bounding-Box“). Abbildung 3.4 zeigt exemplarisch für ein generisches 2D-Netz die Abstraktion ausgewählter CSM-Partitionen mit Bounding-Boxen.

Ähnlich zu Bounding-Boxen können Punkte darüber hinaus mit einer Bounding-Ellipse beschrieben werden. Wie aus dem Namen ersichtlich wird anstatt zweier Vektoren eine Ellipsengleichung der Form

$$\varepsilon = \{x \in R^n \mid (x-c)^T E(x-c) \leq 1\} \tag{3.5}$$

errechnet, dessen Ellipse die Punktmenge umschließt. Im Gegensatz zur gewöhnlichen Bounding-Box berücksichtigt eine Bounding-Ellipse stets die Orientierung der Punkte. Nachteilig stellt sich hierbei jedoch der Berechnungsaufwand dar. Eine genaue Berechnung der Bounding-Ellipse kann in der Regel nur iterativ bestimmt werden([15]). Ist eine grobe Annäherung bereits ausreichend, kann diese ebenso in $\mathcal{O}(\#AnzahlPunkte)$ ermittelt werden.

Obwohl eine Ellipse zur Beschreibung i.d.R bessere Ergebnisse liefert als eine Bounding-Box hat sie neben der aufwändigen Berechnung einen weiteren Nachteil. Die für eine Bestimmung der nächsten Partition notwendigen Bereichsanfragen in einem Suchbaum, welcher alle ermittelten Bounding-Volumes beinhaltet, lässt sich im Falle einer Ellipse nur sehr aufwendig realisieren. Neben Boxen und Ellipsen lassen sich darüber hinaus Polyeder als Hüllkörper verwenden, bei denen eine größere Anzahl an Begrenzungsflächen definiert werden und somit die Partition enger umschließen und beschreiben. Das Finden eines geeigneten und optimalen Polyeders ist im Allgemeinen aber aufwendiger als eine Bounding-Box- bzw. Bounding-Ellipse-basierte Methode.

Die für räumliche Suchbäume, wie dem k-d-Baum, bestehenden schnellen Bereichssuchen in den einzelnen Dimensionen lassen sich bei Ellipsen und Polyedern jedoch nicht durchführen, sodass die gewonnene Effizienz durch einen Suchbaum verloren geht. Im Rahmen dieser Arbeit werden daher einfache Bounding-Boxen zur Beschreibung der Partitionen verwendet. Obwohl dies die Abstraktion darstellt, bei der die größtmöglichen Überlappungen der Bounding-Boxen der Partitionen auftreten, haben sie den entscheidenden Vorteil, dass die Bounding-Boxen innerhalb der eingesetzten k-d-Baumstruktur effizient verwendet werden können.

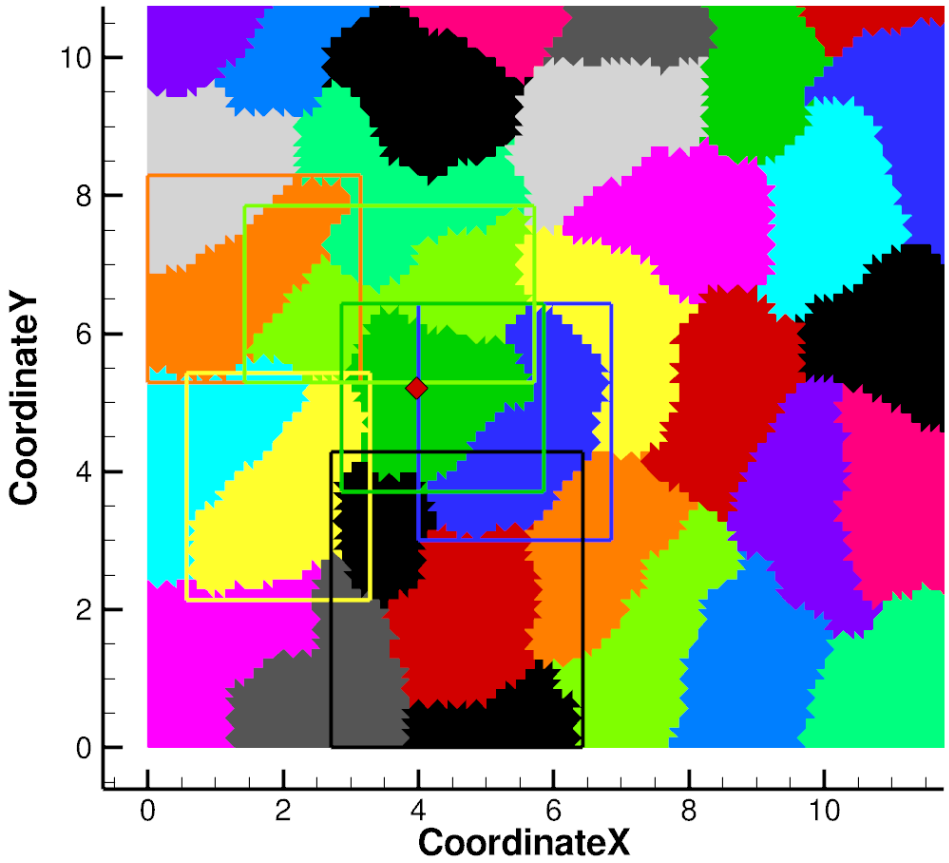


Abbildung 3.4: Bounding-Boxen einzelner CSM-Partitionen um Umkreis eines CFD-Oberflächenpunkts (rotmarkierte Raute)

Nächste-CSM-Partitionen-Suche

Auf Basis des zuvor beschriebenen Bounding-Box Ansatzes und der k-d-Baumstruktur lassen sich nun für jeden CFD-Oberflächenpunkt erste Umgebungssuchen durchführen. Diese ermöglichen eine Zuordnung des Oberflächenpunkts zu den nächsten Strukturnetz-Partitionen, welche mögliche potentielle Stützstellen beinhalten. Ein Verfahren zur Umsetzung dieses Aspekts wird im Folgenden erläutert und ist zudem in Algorithmus 1 dargestellt.

Kerngedanke des Algorithmus ist die iterative Suche im lokal gespeicherten k-d-Baum nach CSM-Partitionen, bis die geforderte Anzahl an Stützstellen durch Punkte der gefundenen Partitionen gewährleistet werden kann. Zu diesem Zeitpunkt ist jedoch die Festlegung, wie viel Punkte von den gefundenen Partitionen zur Interpolation benötigt werden, noch von untergeordneter Bedeutung.

Algorithm 1 Algorithmus zur Ermittlung nächster CSM-Partitionen

Require: Knotenkoordinate des CFD-Oberflächenpunkts *coord*, k-d-Baum der Bouding-Boxen aller CSM-Partitionen *bboxKDTree*, geforderte Anzahl nächster Nachbarn *nNearestPoints*

1: *locPtsOnProcs* ... kummulierte Anzahl lokaler CSM-Strukturpunkte einzelner Partitionen
2: *enoughPoints* ... Flag, ob genug Punkte gefunden
3: *scaleFactorRange* ... Skalierungsfaktor des Suchradius
4: *nearestProc* ... nächste Partition für CFD-Punkt

5: $scaleFactorRange \leftarrow 1$
6: $locPtsOnProcs \leftarrow 0$
7: $enoughPoints \leftarrow false$
8: **while** not *enoughPoints* **do**
9: $nearestProc \leftarrow Finde\_NächstePartition(bboxKDTree, coord)$
10: $sMin, sMax \leftarrow BerechneSuchbereich(coord, nearestProc, scaleFactorRange)$
11: $foundParts \leftarrow BereichsSuche(bboxKDTree, sMin, sMax)$
12: **for** proc in $foundParts$ **do**
13: $locPtsOnProcs \leftarrow locPtsOnProcs + LokaleAnzahlCSMPunkte(proc)$
14: **end for**
15: **if** $(locPtsOnProcs >= nNearestPoints)$ **then**
16: $enoughPoints \leftarrow True$
17: **else**
18: $scaleFactorRange \leftarrow f(scaleFactorRange)$
19: **end if**
20: **end while**
21: **return** $foundParts$

Ausgangspunkt einer Partitionssuche stellt die Zuordnung eines CFD-Oberflächenpunkts zur direkt nächsten CSM-Partition dar (Zeile 9). Dies erfolgt dabei mithilfe des in Kapitel 2.6.1 vorgestellten Algorithmus innerhalb des k-d-Baums. Da ein Oberflächenpunkt dreidimensional ist, Bounding-Boxen des Baums jedoch 6-dimensional sind, muss auf Basis der CFD-Punktkoordinaten eine Bounding-Box definiert werden, die den Punkt umschließt. Der Einfachheit halber werden dabei die Koordinaten des Punkts sowohl als untere als auch obere Begrenzung benutzt, wodurch eine Box mit leerem Volumen erzeugt wird.

Im Allgemeinen kann aber nicht davon ausgegangen werden, dass die von einer nächstgelegenen CSM-Partition stammenden Stützstellen für eine genaue Interpolation hinsichtlich der in Kapitel 2.1 definierten Gütekriterien ausreichen. Dies kann nur für den Fall abgeschätzt werden, wenn der die Stützstellen aufspannende Stützradius an keinem Punkt die tatsächliche Grenze der nächstgelegenen CSM-Partition schneidet, indem die Stützstellen liegen. Liegen hierbei alle CSM-Nachbarpunkte zudem lokal auf demselben Prozess wie der CFD-Punkt, so kann dies sogar ohne

zusätzlichen parallelen Kommunikationsaufwand überprüft werden. Da in der Regel aber CFD-Oberflächen- und CSM-Strukturnetzknoten unterschiedlich verteilt sein können, kann dies nur in seltenen Fällen gewährleistet werden.

Es muss infolgedessen im nächsten Schritt ein Suchradius[2](mit den Begrenzungen $sMin$ und $sMax$) eingeführt werden, um zusätzlich umliegende Partitionen zu finden (Zeile 10). Dabei spielt die Lage des CFD-Oberflächenpunkts innerhalb der zuvor gefundenen nächsten CSM-Partition eine wichtige Rolle zur Bestimmung möglicher Suchradien. Ohne Berücksichtigung der Bounding-Box Begrenzung der nächstgelegenen CSM-Partition lässt sich zunächst ein beliebiger, in allen Dimensionen gleicher Radius um den CFD-Oberflächenpunkt festlegen. Dies hat den Vorteil, symmetrisch in der gesamten Umgebung des CFD-Punkts zu suchen. Partitionen besitzen entlang der Koordinatenachsen aufgrund der zugrunde liegenden Partitionierungsmethodik jedoch oftmals unterschiedliche Ausdehnungen, zudem hervorgerufen durch die Gesamtgeometrie - ein Flügel besitzt bspw. entlang einer Richtung eine wesentlich größere Ausdehnung als in den anderen Raumdimensionen. Diese Information lässt sich allerdings anhand der Bounding-Box Begrenzung der nächstgelegenen Partition nutzen. Es ist daher von Vorteil, den Suchradius anhand der Entfernungen zu diesen Grenzen zu bestimmen. Da in der Arbeit zur Bestimmung der Bounding-Boxen einer CSM-Partition zusätzlich generierte Ghost-Knoten (vgl. Kapitel 2.5.1) angrenzender Partitionen verwendet werden und somit größere Überlappungen entstehen, werden angrenzende Partitionen mit größerer Wahrscheinlichkeit erfasst. Abbildung 3.5 zeigt grafisch die zuvor diskutierten, prinzipiellen Möglichkeiten einiger Suchradien bzw. „Suchrechtecke“. Im Rahmen der Analysen in Kapitel 4 wurde hauptsächlich ein Suchradius auf Basis der Entfernungen zur nächsten Partitions-Bounding-Box gewählt.

Auf Basis des Suchradius wird im Weiteren die eigentliche Bereichssuche durchgeführt. Für alle hierbei gefundenen Partitionen wird deren Anzahl vorhandener CSM-Knoten überprüft (Zeile 13). Können diese Partitionen zusammen mehr als die geforderte Stützstellenanzahl liefern, kann die iterative Suche abgebrochen werden.

Andernfalls wird ein zu Beginn definierter Skalierungsfaktor $scaleFactorRange$ verändert, welcher den zuvor verwendeten Suchraum vergrößert und somit gezielt eine Konvergenz der Suche bewirken soll. Hierbei stellt sich die Verwendung eines Suchradius auf Basis der Bounding-Box-Begrenzungen erneut als vorteilhaft heraus, wodurch bei einer iterativen Bereichssuche gezielt der Raum untersucht werden kann. Besitzt die nächste Partition in einer Dimension eine wesentlich stärkere Ausdehnung als in anderen, so wird diese in weiteren Suchiterationen durch den Skalierungsfaktor stärker durchsucht als anderen Raumdimensionen.

[2]Bei der Bezeichnung von Radius wird in diesem Kontext stets von achsen-parallelen Abständen zum Suchpunkt ausgegangen. Denn basierend auf diesen Abständen erfolgt die Suche im k-d-Baum.

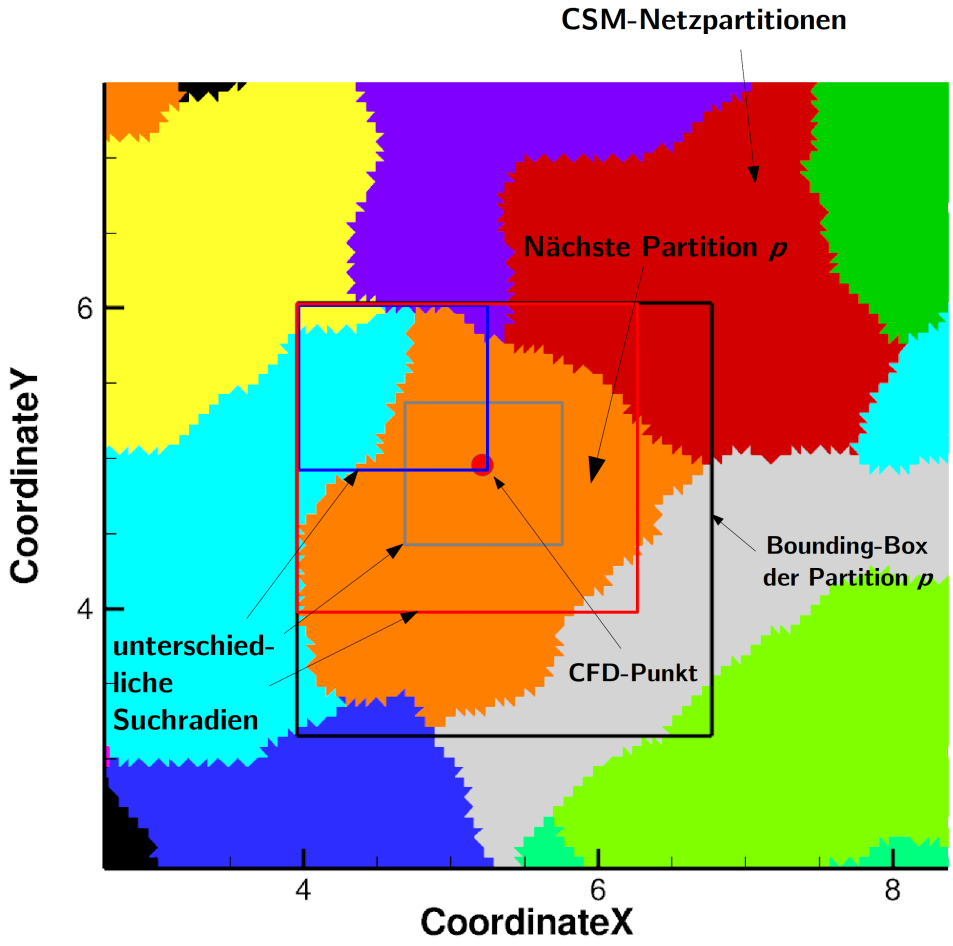


Abbildung 3.5: Unterschiedliche Suchradien der Bereichssuche: willkürlicher Radius (grau), Entfernung zur nächsten Ecke der Bounding-Box (blau), symmetrischer Radius auf Basis des Abstands zur nächsten Ecke (rot)

Im Rahmen dieser Arbeit wird der Skalierungsfaktor in jedem Iterationsschritt verdoppelt und zur Vergrößerung der Suchbox verwendet. Dies entspricht einer schrittweisen exponentiellen Vergrößerung des Suchradius in alle Richtungen. Alternativ lassen sich zur Definition eines Skalierungsfaktors Funktionen definieren, die den Suchradius auf Basis der bisher gefundenen Partitionen beschreiben. Diese werden in der Arbeit jedoch nicht weiter betrachtet.

Auf Basis der gefundenen Partitionen wird im nächsten Abschnitt erläutert, wie viele Stützstellen einer CSM-Partition maximal für einen gegebenen CFD-Oberflächenpunkt verwendet werden sollen.

Bestimmung der Stützstellenanzahl pro gefundener Partition

Für die Bestimmung der Anzahl benötigter Stützstellen gefundener Partitionen wurden in der Arbeit zwei wesentliche Ansätze verfolgt:

1. Approximative NN-Bestimmung auf der Basis des Mahalanobis-Distanzmaßes
2. Exakte Bestimmung der nächsten-Nachbarn mithilfe einer Greedy-Methode

Für beiden Methoden ist nur die Bestimmung der Stützstellenanzahlen auf einem beliebigen Prozess wesentlich, nicht die korrekte Bestimmung der Strukturknoten selbst. Die korrekte geometrische Ermittlung wird auf den entsprechenden Partitionen und deren zugehörigen parallelen Prozessen *nach* der Übermittlung der NN-Anfragen mithilfe eines k-d-Baums umgesetzt, welcher alle CSM-Strukturpunkte einer Partition effizient speichert.

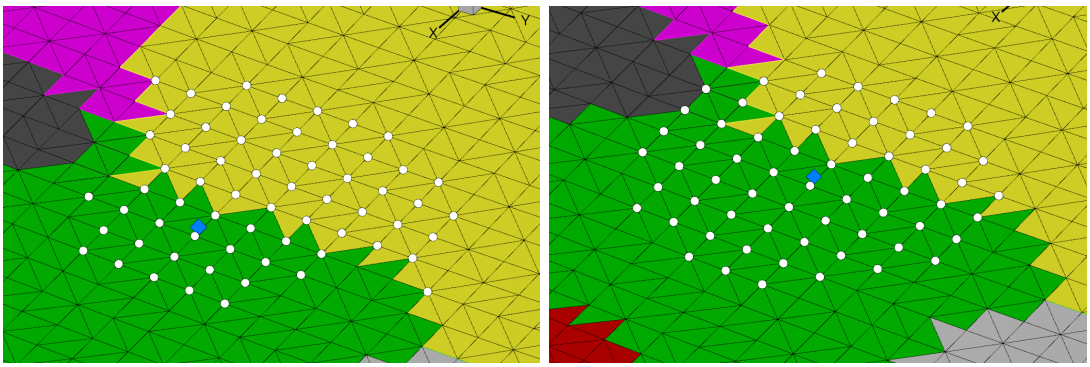
Approximative NN-Bestimmung auf Basis des Mahalanobis-Distanzmaßes

Unter einer approximativen NN-Bestimmung versteht man, dass CSM-Strukturknoten einzelner Partitionen als Stützstellen benutzt werden, welche jedoch nicht den geometrisch exakten n-Nächsten-Nachbarn eines CFD-Oberflächenpunkts entsprechen müssen. Eine Bestimmung erfolgt dabei über spezielle Distanzmaße eines CFD-Punkts zu einer CSM-Netzpartition. Wird das Distanzmaß über alle gefundenen Partitionen normiert, ergibt es für einen CFD-Punkt die zahlenmäßige Verteilung der Stützstellen auf den einzelnen CSM-Partitionen.

Entscheidender Vorteil dieser Methodik ist der entstehende Performance-Gewinn gegenüber exakten Methoden. Da nach der Übertragung alle empfangenen Stützstellen zur Koeffizientenberechnung der Kopplungsmatrix benutzt werden, muss keine zusätzliche Auswahl der Stützstellen bzw. deren Sortierung durchgeführt werden. Eine Verarbeitung der Daten gemäß des in Abbildung 3.3 vorgestellten parallelen Algorithmus (Unterpunkt ***lokale Verarbeitung aller gesendeter Stützstellen***) besteht somit nur in der losen, unsortierten Speicherung der Stützstellen pro CFD-Oberflächenpunkt.

Nachteilig stellt sich bei diesem approximativen Vorgehen jedoch die nicht zu gewährleistende Robustheit bei der Verformungsinterpolation dar. Robust bedeutet dabei in dieser Arbeit die Unabhängigkeit des Verfahrens hinsichtlich der zugrunde liegenden Partitionen und Partitionsanzahlen. Der Vollständigkeit halber soll hier aber eine Form der approximativen Suche auf Basis des Mahalanobis-Distanzmaßes kurz vorgestellt werden.

Abbildung 3.6(a) zeigt beispielhaft für einen gegebenen CFD-Oberflächenpunkt (blau markierte Raute) 64 Stützstellen (weiße Punkte), die mit der der approximativen Bestimmung der nächsten Nachbarn berechnet wurden. In einer Bereichssuche wurde zuvor die grün- und gelb-markierte CSM-Partition als nächste Partitionen identifiziert. Zur Bestimmung der Anzahl wird das im Weiteren beschriebene Mahalanobis-Distanzmaß benutzt.



(a) Approximative Suche der nächsten Nachbarn

(b) Exakte geometrische Nachbarn

Abbildung 3.6: Vergleich von approximativer und exakter CSM-Stützstellen für einen gegebenen CFD-Punkt (blau markierte Raute)

Ausgangspunkt ist zunächst die Berücksichtigung der Orientierung aller CSM-Partitionen. Die Orientierung einer Partition anhand seiner Gitterpunkte lässt sich mithilfe der Kovarianzmatrix S bestimmen:

$$S = \begin{pmatrix} s_x^2 & s_{xy} & s_{xz} \\ s_{xy} & s_y^2 & s_{yz} \\ s_{xz} & s_{yz} & s_z^2 \end{pmatrix} \tag{3.6}$$

Die Einträge auf der Hauptdiagonalen von S werden bestimmt mit (hier beispielhaft für die x-Koordinate dargestellt):

$$s_x^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - n\overline{x}^2\right) \tag{3.7}$$

$\overline{x}$ bezeichnet dabei die Durchschnittskoordinate der Punktdaten. Die jeweiligen Nebendiagonaleinträge sind gegeben durch (hier ebenso beispielhaft für die xy-Komponente dargestellt).

$$s_{xy} = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i y_i - n\overline{xy}\right) \tag{3.8}$$

Im Gegensatz zur gewöhnlichen Varianzdefinition $s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$ ermöglicht diese äquivalente Gleichung eine Bestimmung der Varianz bei einmaligem Durchlaufen der Knotenpunkte. Mithilfe der Kovarianzmatrix lässt sich nun der Abstand eines CFD-Punktes zu einer CSM-Partition berechnen. Die Kovarianzmatrix ist nun Teil der sogenannten Mahalanobis-Distanz. Sie beschreibt ein Distanzmaß zwischen Punkten in einem mehrdimensionalen Vektorraum und wird im Speziellen

in der Statistik, z.B. bei multivariaten Verteilungen eingesetzt. Formell wird der Mahalanobis-Abstand zwischen zwei Punkten mit der Gleichung

$$d_{Mahal}(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T S^{-1} (\vec{x} - \vec{y})} \tag{3.9}$$

beschrieben. Ein gegebener Punkt x wird dabei als Realisation eines durch die Kovarianzmatrix S repräsentierten Zufallsvektors angesehen. In diesem Kontext beschreibt der Zufallsvektor die statistische Verteilung der Netzpunkte entlang der Raumdimensionen. Grafisch gesehen beschreiben Punkte desselben Mahalanobis-Abstands eine Ellipse. Das Maß kann nun genutzt werden, um den Abstand eines Punktes zum Zentrum einer Punktmenge bzw. Partition (hier mit $\vec{y}$ gekennzeichnet) zu bestimmen.

Mithilfe der errechneten Mahalanobis-Distanz pro Partition muss im nächsten Schritt noch eine Normierung aller Abstände definiert werden, um eine numerische Verteilung der Stützstellen auf die einzelnen Partitionen zu berechnen. Diese Normierung erfolgt mit

$$d(Pkt_{CFD}, Part_i) = \frac{d(Pkt_{CFD}, Part_i)^{-1}}{\sum_{j=1}^{p} d(Pkt_{CFD}, Part_j)^{-1}} \tag{3.10}$$

Die Verwendung des inversen Abstandsmaßes ist erforderlich, um bei kleinem Abstand eines CFD-Punkts Pkt_{CFD} zu einer Partition $Part_i$ einen hohen Anteil an Stützstellen für diese Partition zu fordern und umgekehrt. Die hierbei beschriebene Technik ist dem sogenannten Fuzzy-Clustering angelehnt, welches anhand der Mahalanobis-Distanz für einen gegebenen Punkt die Wahrscheinlichkeit berechnet, mit der jener Punkt zu einem Punktcluster zugeordnet werden kann. Abbildung 3.7 verdeutlicht grafisch die Verteilung des Mahalanobis-Distanzmaßes für die Strukturknoten einzelner CSM-Partitionen. Es wurde auf 12 partitionierte CSM-Teilnetze eines generischen Rechteckgitters angewendet, welches in Kapitel 4 als einer der Testfälle verwendet wird.

Kritische Betrachtung der approximativen NN-Suche auf der Basis der Mahalanobis-Distanz

Die hier vorgestellte approximative Herangehensweise zur Bestimmung der nächsten Nachbarn basierend auf der Mahalanobis-Distanz erwies sich in Studien zwar als effizient und stabil. Die für aeroelastische Fragestellungen notwendige Forderung glatter Oberflächen konnte in ersten Studien mit dem beschriebenen Verfahren allerdings nicht erfüllt werden. Abbildung 3.8 zeigt beispielhaft für die HiReTT-Konfiguration (näheres zur Konfiguration in Kapitel 4, S. 66) die fehlerhafte Verformungsinterpolation der CFD-Oberfläche auf Basis der approximativen Berechnung der Stützstellen.

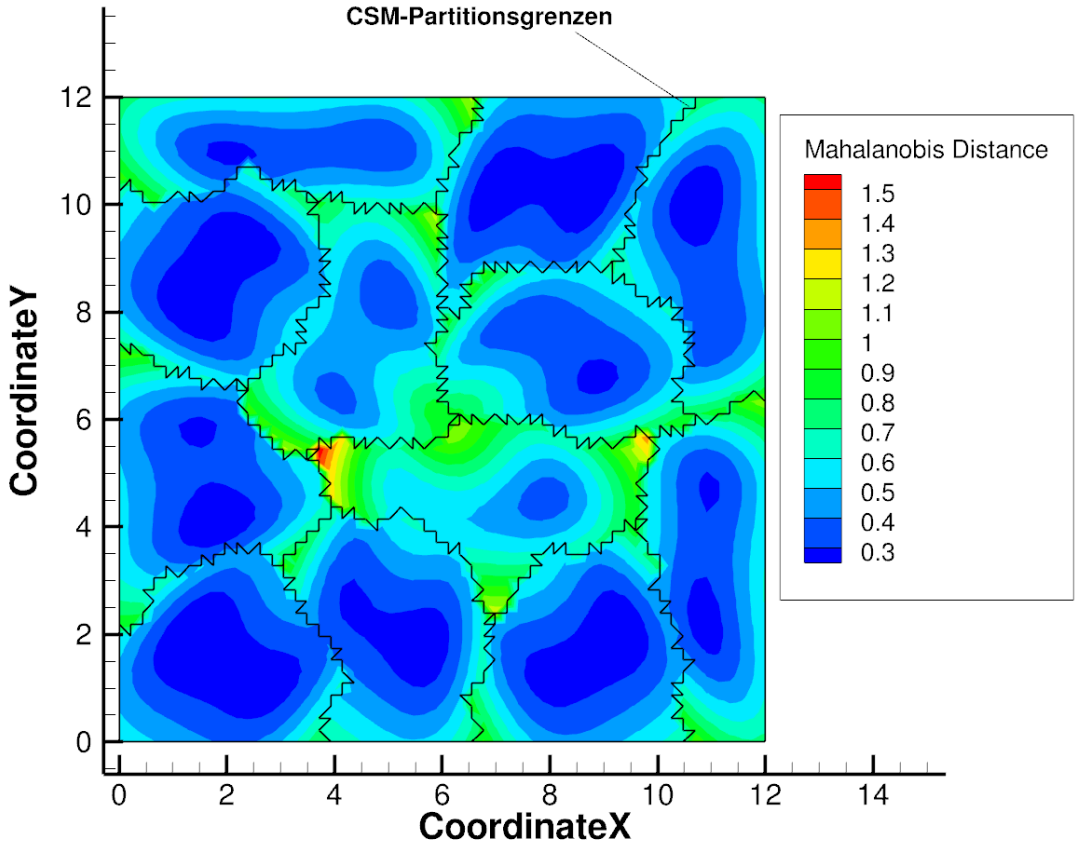


Abbildung 3.7: Mahalanobis-Abstand einzelner CSM-Strukturknoten innerhalb der CSM-Partitionen zum Zentrum der Partition.

Entscheidendes Problem des approximativen Ansatzes stellen etwaig auftretende, nicht-zusammenhängende Partitionen dar. Zwischen Netzpunkten der gleichen CSM-Partition können infolgedessen Punkte anderer Partitionen liegen. Der Bounding-Box Ansatz zur Beschreibung der Partitionen berücksichtigt dies jedoch zunächst nicht und erzeugt somit viel zu große Bounding-Boxen. Auf Basis der ermittelten Boxen und vorhandenen Punktdaten erfolgt daraufhin eine fehlerhafte Berechnung der Mahalanobis-Distanz. Aufgrund der „zerstückelten“ Partition ist die ermittelte Varianz somit nicht repräsentativ für die zugrunde liegenden Punktdatenmenge. Die Bestimmung des Partitionsmittelpunkts $\vec{y}$ wird dabei statistisch berechnet, aber nicht auf Basis der geometrisch vorliegenden Punktverteilung. Ein CFD-Punkt kann infolgedessen eine kurze Distanz zu einer Partition besitzen, dessen nächste Stützstellen allerdings weit entfernt sein können.

Darüber hinaus entscheidet der gewählte Suchradius zudem über die Genauigkeit bei der approximativen Suche. Da das Mahalanobis-Maß auf alle gefundenen Partitionen angewendet wird, werden die Stützstellen über diese Partitionen verteilt.

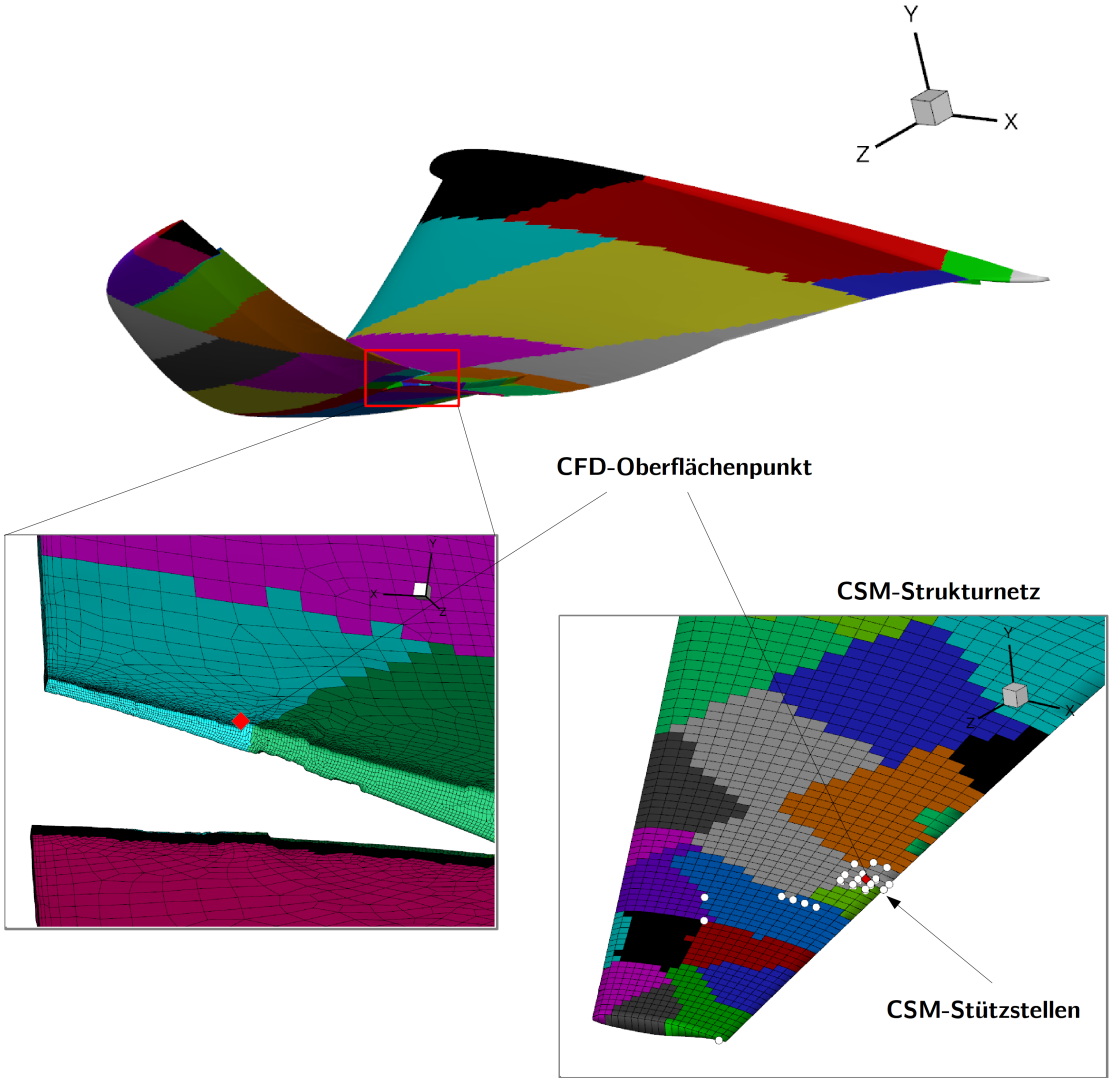


Abbildung 3.8: Fehlerhafte Verformungsinterpolation eines CFD-Oberflächenpunkts (rote Raute) auf Basis der approximativen Suche der nächsten Nachbarn (als weiße Kreise gekennzeichnet)

Weit entfernte CSM-Partitionen können bei ungünstiger Wahl des Suchradius dennoch Stützstellen beitragen. Der auf der Menge der Stützstellen errechnete Stützradius wird dementsprechend zu groß bestimmt.

Diese Problematik lässt sich zwar durch Berücksichtigung der Ausdehnung der CSM-Partition im Verhältnis zu seiner Punktemenge skalieren, eine genaue numerische Beschreibung der nicht-zusammenhängenden Bereiche einer Partition konnte damit allerdings nicht erreicht werden.

Für Analysen in Kapitel 4 wird somit nur die exakte Bestimmung der Stützstellen verwendet. Sie soll nun im Folgenden beschrieben werden.

Exakte Bestimmung der nächsten-Nachbarn mithilfe einer Greedy-Methode

Exakt bezieht sich in diesem Kontext lediglich auf die Bestimmung der geometrisch exakten Nachbarn aller Punkte der *gefundenen* nächsten CSM-Partitionen. Punkte von CSM-Partitionen, die zuvor von der Bereichssuche in Algorithmus 1 nicht erfasst wurden, werden hierbei weiterhin **nicht** betrachtet. Die in der Arbeit genutzte exakte Suche basiert im wesentlichen auf einem Greedy-Ansatz. Dabei lässt sich der Grad der „Gier“ anhand der geforderten Punktanzahl pro Partition verdeutlichen. Die trivialste Methode besteht darin, alle Punkte der gefundenen Partitionen zur Bestimmung der Stützstellen zu nutzen und im Rahmen der parallelen Kommunikation zu empfangen. Das Vorgehen hat jedoch besonders im Bereich kleiner Partitionsanzahlen entscheidende Performance-Nachteile- bei ungünstiger Wahl würden alle partitionierten Punkte des CSM-Strukturnetzes untereinander ausgetauscht- und wird daher nicht weiter betrachtet. Im Idealfall liegen alle Stützstellen in einer einzigen Partition, so dass lediglich die vorher festgelegte Stützstellenanzahl von jeder der gefundenen Partitionen gefordert werden muss. Besitzt eine Partition weniger als die geforderte Anzahl, so werden stattdessen alle Punkte der Partitionen betrachtet.

Gegenüber einer approximativen Suche hat die exakte Methode den Performance-Nachteil, dass alle potentiellen Stützstellen-Kandidaten nach ihrem Abstand zum CFD-Oberflächenpunkt sortiert werden müssen. Nur ein Teil der Stützpunkte wird dann konkret zur Bestimmung der Koeffizientenmatrix herangezogen und zu weit entfernte Punkte werden wieder verworfen. Allerdings muss dieser „Mehraufwand“ pro CFD-Oberflächenpunkt nur einmal durchgeführt werden.

Da in dieser Arbeit zur Bestimmung der lokal nächsten CSM-Punkte ein k-d-Baum eingesetzt wird, kann von einer teilweisen Vorsortierung der ermittelten nächsten Strukturknoten bzw. ihrem Abstand zum CFD-Punkt ausgegangen werden. Dies begünstigt die Wahl von Sortierverfahren, die auf einer Vorsortierung basieren. Vertreter solcher Verfahren ist z.B. das Natural-Mergesort. Der im k-d-Baum des FlowSimulator implementierte Nächste-Nachbarn-Suchalgorithmus wird zudem mit einer zusätzlichen Liste umgesetzt, die beim Durchlaufen des Suchbaums potentielle Kandidatenpunkte nach ihrem Abstand zum Suchpunkt speichert. Nach Ausführung des Suchalgorithmus liegen die ermittelten nächsten Nachbarknoten einer CSM-Partition somit in komplett sortierter Reihenfolge, gemäß ihres Abstands zum Suchpunkt, vor.

Für eine korrekte Bestimmung der exakten geometrischen Nachbarn müssen nun alle empfangenen Stützstellen auf dem Prozess des CFD-Oberflächenpunkts in einer Liste sortiert werden. Diese Sortierung ist, gemäß dem parallelen Algorithmus in Abbildung 3.3, Teil der ***lokalen Verarbeitung aller gesendeter Stützstellen*** und soll im Folgenden näher beleuchtet werden.

3.2.3 Lokale Verarbeitung aller gesendeter Stützstellen

Nach Austausch aller gesendeter Stützstellen erfolgt lokal, d.h. auf den Prozessen der einzelnen CFD-Oberflächennetze, die Verarbeitung der Daten. Aufgrund der umgesetzten technischen Aspekte der Parallelisierung werden die Daten asynchron verarbeitet. Dies bedeutet, dass empfangene Stützstellen einzelner Partitionen immer stückweise bearbeitet werden. Ein möglicher Ansatz, der empfangene CSM-Strukturknoten in eine lokal bestehende Liste der Stützstellen für einen CFD-Oberflächenpunkt einsortiert, lässt sich mithilfe von Algorithmus 2 beschreiben:

Algorithm 2 Algorithmus zur Verarbeitung aller gesendeter Stützstellen

Require: abstands-sortierte Menge an Stützstellen einer CSM-Partition $pointSet$, Liste aus bisher ermittelten Stützstellen $allSupportPoints$ für gegebenen CFD-Punkt $cfdPoint$

1: $sortIndex$ ... Vergleichsindex während Sortierung
2: $nNextCenters$ ... Gesamtanzahl Stützstellen für Interpolation eines CFD-Punkts
3: $sortIndex \leftarrow 0$
4: $k \leftarrow 0$
5: **for** point in $pointSet$ **do**
6: **if** $|allSupportPoints| < nNextCenters$ **then**
7: Füge $point$ in $allSupportPoints$ am Ende ein
8: **else**
9: $k \leftarrow sortIndex$
10: **for** $k < nNextCenters$ **do**
11: **if** $d(point, cfdPoint) < d(allSupportPoints(k), cfdPoint)$ **then**
12: Lösche letztes Element der Liste $allSupportPoints$
13: Verschiebe Teilliste $allSupportPoints(k : nNextCenters - 1)$ um eine Stelle nach rechts
14: Füge $point$ an Stelle $allSupportPoints(k)$ ein
15: $sortIndex \leftarrow k + 1$
16: break
17: **else**
18: $k \leftarrow k + 1$
19: **end if**
20: **end for**
21: **end if**
22: **end for**
23: **return** $allSupportPoints$

Innerhalb des Algorithmus werden die CSM-Punkte, welche zeitlich zuerst von einem anderen Prozess empfangen wurden, in die noch leere Gesamtliste aller empfangener Punkte eingefügt (Zeile 7). Wenn bereits $|nNextCenters|$ empfangen wurden, müssen neu empfangene Punkte diese Liste durchlaufen und ggf. Punkte,

welche einen größeren Abstand zum CFD-Oberflächenpunkt besitzen, aus der Liste verdrängen. Damit ist gewährleistet, das nach Ausführung des Algorithmus nur die zuvor definierte Menge an Stützstellen gespeichert wird.

Ist der Abstand eines empfangenen Punkts *point* nun kleiner als ein bestimmter Punkt an Stelle k in der Liste, so wird der neue Punkt an diese Position eingefügt. Alle weiteren Elemente $k+1, ..., nNextCenters-1$ werden daraufhin um eine Stelle nach rechts verschoben. Da weitere Punkte der Menge *pointSet* u.U. in die Liste eingefügt werden, kann der zuvor bestimmte Positionsindex k des Punkts *point* als Startindex *sortIndex* genutzt werden, um nicht die komplette Liste von Beginn an durchlaufen zu müssen.

Aufgrund des punktweisen Einsortierens der Daten können zudem dynamische Strukturen anstelle der vorgestellten Methodik verwendet werden. Diese ermöglichen ein effizienteres Einfügen der Punkte, bei dem die komplette Verschiebung einer Teilliste (Zeile 13) vermieden werden kann. Entsprechende Datenstrukturen werden in der Arbeit jedoch nicht weiter betrachtet.

Wird Algorithmus 2 nach Erhalt aller potentieller Stützstellen für jeden CFD-Oberflächenpunkt ausgeführt, können alle Punktkoordinaten der Liste $allSupportPoints$ zur Koeffizientenbestimmung im Rahmen der Quaranta-Methode benutzt werden. Eine effiziente Methode zur Speicherung der Koeffizienten und deren Nutzung im Rahmen einer parallelen Berechnung des Last- und Verformungstransfers wird im Folgenden Abschnitt beschrieben.

3.3 Effiziente Berechnung der Kopplungsmatrix

Die zuvor ermittelten nächsten Nachbarn auf Strukturseite werden nun mithilfe des in Kapitel 2.3 vorgestellten Quaranta-Verfahrens zur Bildung der Kopplungsmatrix H herangezogen. Im Gegensatz zur Wendland-Methode, bei der je nach verwendeter radialer Basisfunktion evtl. eine vollbesetzte Matrix erzeugt wird (für RBF mit sogenanntem globalem Träger), ist i.A. eine durch die Quaranta-Methode erzeugte Matrix von dünnbesetzter Struktur. Je nach Anzahl verwendeter Stützstellen würde eine komplette Matrix-Speicherung, welche auch die Nulleinträge enthält, inakzeptabel viel Speicher verbrauchen. Das zur Interpolation der gegebenen Last- und Verformungsgrößen durchzuführende Matrix-Vektor Produkt würde zudem ineffizient durchgeführt. Unter der Annahme, dass eine komplette Matrix u.U. nicht in den Cache eines Prozessors passen würde, müssten von Null verschiedene Koeffizienten zudem wiederkehrend dazu geladen werden und Null-Einträge verdrängen, welche keine Anteil an der Interpolation besitzen.

Es wurde daher eine angepasste Speicherform gewählt, die die zuvor genannten Effizienz-Nachteile eliminiert. Eine mögliche Lösung ist die Verwendung einer CSR-Matrix. Sie wurde als Teil der RBF-basierten Interpolationsmethode im FlowSimulator implementiert.

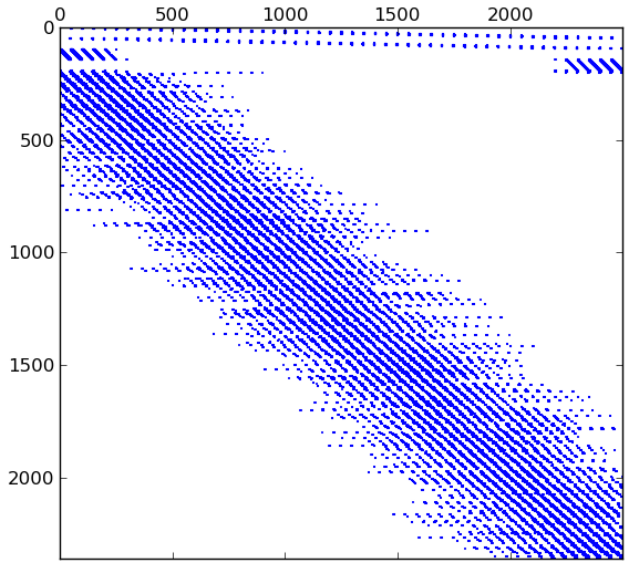


Abbildung 3.9: Besetzungsstruktur einer 2360x2500 Kopplungsmatrix mit 32 nächsten Nachbarn je CFD-Punkt

3.3.1 CSR-Matrix

Eine CSR-Matrix (Compressed-Sparse-Row) ist ein häufig eingesetztes Verfahren zur Speicherung von dünnbesetzten Matrizen. Gegenüber einer kompletten, unkomprimierten Speicherung einer Matrix speichert das CSR-Format nur die von Null verschiedenen Einträge einer Matrix. Entsprechende Einträge werden zunächst linear in einem Array $nzeroEntries$ hintereinander abgespeichert. Für die korrekte Abbildung eines Eintrags in die entsprechende Zeile und Spalte der Ursprungs-Matrix werden zwei Hilfsarrays $colInd$ und $rowPtr$ angelegt, welche Spalten und Zeileninformationen der Einträge beinhalten. Die Spaltenindizes der Einträge von $nzeroEntries$ sind Einträge der Matrix $colInd$ und besitzen somit die selbe Länge wie $nzeroEntries$. Das Array $rowPtr$ legt nun fest, wie viel von Null verschiedene Einträge eine Zeile der Matrix besitzt. Der Eintrag $rowPtr[i]$ beschreibt dabei den Startindex der Nicht-Null Einträge der Zeile i im Array $nzeroEntries$. Formal lässt sich somit ein Eintrag $A_{i,j}$ der Ursprungsmatrix mit der folgenden Vorschrift ermitteln:

$$A_{i,j} = nzeroEntries[k] \Leftrightarrow (colInd[k] = j) \wedge (rowPtr[i] \leq k < rowPtr[i+1]) \tag{3.11}$$

Für die folgende Beispielmatrix ergeben sich z.B. die folgenden Felder bei Speicherung entsprechend des CSR-Formats.

$$A = \begin{pmatrix} 3 & 0 & 8 & 4 \\ 0 & 8 & 0 & 9 \\ 0 & 1 & 0 & 0 \end{pmatrix} \qquad rowPtr = \begin{pmatrix} 0 & 3 & 5 & 6 \end{pmatrix}$$

$$nzeroEntries = \begin{pmatrix} 3 & 8 & 4 & 8 & 9 & 1 \end{pmatrix} \qquad colInd = \begin{pmatrix} 0 & 2 & 3 & 1 & 3 & 1 \end{pmatrix}$$

Um auf einen Eintrag $A_{i,j}$ der Matrix zuzugreifen, müssen somit zunächst Einträge der Zeile mit einem Spaltenindex $m<j$ durchlaufen werden. Diese Eigenschaft ist allerdings bei der Matrix-Vektor-Multiplikation förderlich, da pro Zeile einer Matrix über alle Einträge iteriert und diese mit dem entsprechenden Vektor-Eintrag verknüpft werden.

3.3.2 Parallele Datenspeicherung und verteiltes Matrix-Vektor Produkt

Die Bestimmung der auf dem CSR-Format basierenden Kopplungsmatrix erfolgt zunächst vollkommen lokal auf jedem Prozess. Gegenüber der seriellen Ausführung verringert sich der Aufwand zur Speicherung der Matrix pro Prozess bei der parallelen Version infolge der pro Prozess geringeren Anzahl an CFD-Oberflächenknoten. Für eine korrekte Durchführung des Matrix-Vektor Produkts $H^T \cdot F$ bzw. $H \cdot u$ müssen bei paralleler Ausführung jedoch gegenüber der sequentiellen Umsetzung einige Anpassungen durchgeführt werden.

Ein CFD-Oberflächennetzpunkt besitzt im Allgemeinen neben seinen Knotenkoordinaten eine eindeutige Knoten-ID id. Zur Berechnung der Verschiebungen $u_{cfd}(id)$ des Punkts wird der Verschiebungseintrag eines CSM-Knotens mit der Knoten-ID j mit dem Interpolationskoeffizienten $H(id, j)$ multipliziert. Summiert über alle Nicht-Null Einträge der Zeile id ergibt dies somit die Gesamtverschiebung eines CFD-Oberflächenpunkts der jeweiligen Komponente.

Bei partitionierten Netzen tritt in diesem Zusammenhang allerdings die Problematik der lokalen Knotennummerierung auf. Dies bewirkt, dass CSM-Knoten unterschiedlicher Partitionen, zumindest global betrachtet, derselben Nummer j zugeordnet sind. Für eine korrekte Speicherung einer einzigen lokalen Kopplungsmatrix H pro Prozess der Größe $|Anzahl\ lokaler\ CFD\text{-}Knoten| \times |Gesamtanzahl\ der\ CSM\text{-}Knoten|$ müsste somit eine Umrechnung in globalen Knotennummern durchgeführt werden. Die korrekte Berechnung des Produkts $H \cdot u$ könnte zudem erst dann erfolgen, wenn alle Verschiebungsinformationen im Rahmen der Kommunikation auf einem Prozess vorhanden sind.

In der Arbeit werden aus diesem Grund pro Prozess mehrere, sogenannte CSR-Teilkopplungsmatrizen generiert, insgesamt $|Anzahl\ CSMPartitionen|$ Matrizen pro Prozess. Jede dieser Matrizen speichert die Interpolationskoeffizienten, welche

aus der Verwendung eines Knoten einer CSM-Partition als Stützstelle resultiert. Dies verhindert eine Umrechnung der Knotennummern in globale Knoten-ID's, sodass die lokale Knotennummerierung beibehalten werden kann.

Eine solche verteilte CSR-Matrix ist in Bild 3.10 zu sehen. Die Koeffizienten ausgewählter Stützstellen für einen gegebenen CFD-Punkt (schwarzes Dreieck) der Knoten-ID k sind dabei als Einträge der einzelnen Teilkopplungsmatrizen farbig hervorgehoben.

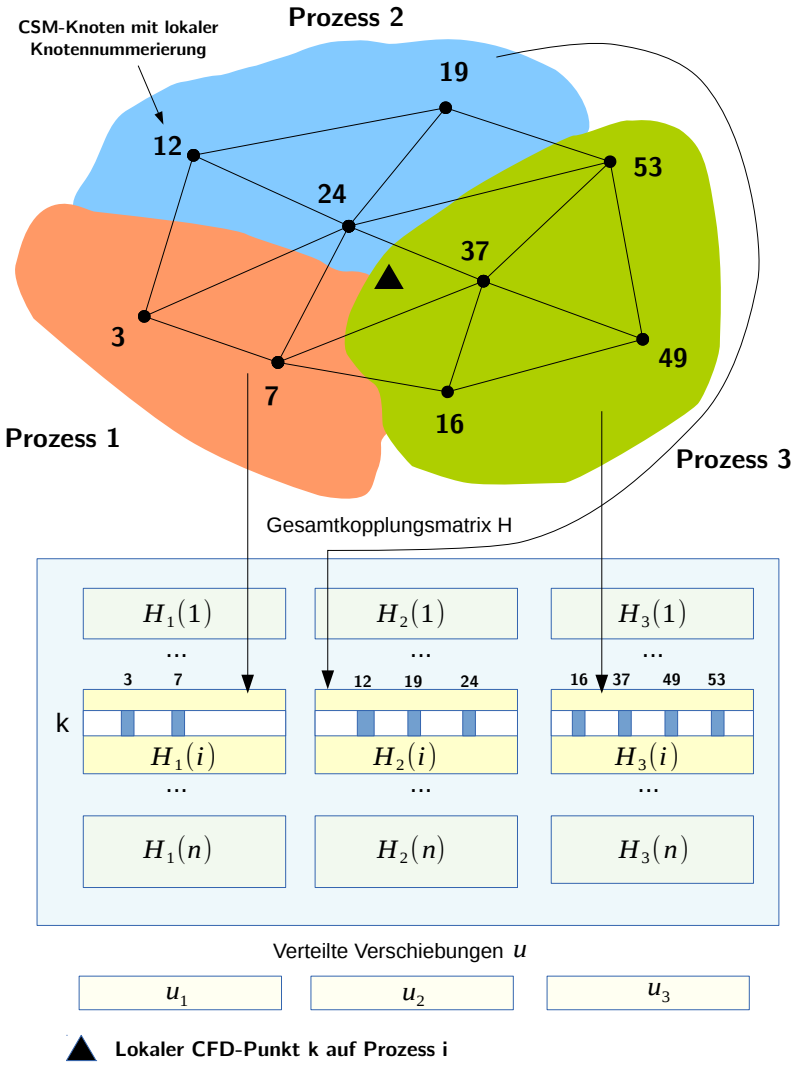


Abbildung 3.10: Verteiltes Speicherungsschema der Kopplungsmatrix H für eine lokale Knotennummerierung

Paralleler Last- und Verformungstransfer

Unter Berücksichtigung der verteilten Last- und Verschiebungsgrößen lässt sich daraus nun ein einfacher umsetzbares, verteiltes Matrix-Vektor Produkt definieren:

$$u_{CFD}(i) = H_i(i) \cdot u_{CSM}(i) + \sum_{j=1, j\neq i}^{n} H_j(i) \cdot \boldsymbol{u_{CSM}(j)} \tag{3.12}$$

$$F_{CSM}(i) = H_i(i)^T \cdot F_{CFD}(i) + \sum_{j=1, j\neq i}^{n} \boldsymbol{H_i(j)^T} \cdot \boldsymbol{F_{CFD}(j)} \tag{3.13}$$

Fett markierte Terme bzw. Daten sind bei Beginn der Ausführung des Produkts auf einem lokalen Prozess zunächst nicht vorhanden. $H_i(j)$ beschreibt in Gleichung 3.12 und 3.13 die i-te Teilkopplungsmatrix des Prozesses j. Zur Interpolation der lokalen Gesamtverschiebungen der CFD-Knoten eines Prozesses muss anhand Gleichung 3.12 zunächst ein Austausch der Verschiebungsgrößen $u_{CSM}(j)$ stattfinden, um anschließend das entsprechend Matrix-Vektor-Produkt ausführen zu können. Um Lastgrößen mithilfe der transponierten Kopplungsmatrix auf einem lokalen Prozess i errechnen zu können, wird gemäß Gleichung 3.13 demgegenüber auf jedem entfernten „Remote-Prozess“ j das entsprechende Matrix-Vektor-Produkt mithilfe der Kopplungsmatrix $H_i(j)$ ausgeführt, um die anteiligen, interpolierten Lastgrößen dann an Prozess i zu versenden. Da bei beiden Matrix-Vektor-Produkten eine stückweise Aufsummierung der Kräfte und Verschiebungen durchgeführt wird, spielt die Kommunikationsreihenfolge der Prozesse während der parallelen Ausführung keine Rolle.

Abschließend zu den erläuterten Methoden der parallelen Stützstellensuche und der Speicherung der Kopplungsmatrix soll im Folgenden kurz auf technische Details hinsichtlich der parallelen Kommunikation eingegangen werden.

3.4 Details der parallelen Kommunikation

Anfragengenerierung

Wie bereits anhand des in Abbildung 3.3 vorgestellten Algorithmus geschildert, erfolgt eine Trennung zwischen der kompletten Sammlung an strukturseitigen Stützstellen für jeden CFD-Punkt und der eigentlichen RBF-basierten Berechnung der Kopplungsmatrix. Diese Stützstellen-Sammlung entspricht dabei technisch betrachtet der Generierung von Nächste-Nachbarn-Suchanfragen, welche unter den Prozessen versendet werden. Zur Generierung werden diese zunächst für **alle** lokalen CFD-Punkte in Form eines 2-dimensionalen Arrays erstellt und anschließend über Ein- und Ausgangspuffer der Prozesse untereinander versendet. Ein derartiges

Array, welches auch als Anfragetabelle bezeichnet werden kann, ist in Tabelle 3.1 beispielhaft dargestellt.

Anfrageprozess	lokale Knoten-ID	Benötigte Stützstellen	Koordinaten des lokalen CFD-Punkts
0	0	32	0.235,0.792,1.384
0	1	32	0.249,0.829,1.431
⋮	⋮	⋮	⋮
0	3917	32	1.375,0.269,-0.911

Tabelle 3.1: Anfragetabelle eines Prozesses der Nummer 0

Mithilfe der Koordinaten in der letzten Spalte in Tabelle 3.1 lassen sich dann auf dem „Remote“-Prozess, der diese Anfrage erhält, innerhalb seines lokalen k-d-Baums der Strukturknoten die dort lokal nächsten Strukturknoten ermitteln. Zur korrekten Übermittlung der Anfragedaten innerhalb der FlowSimulator-Umgebung wird zudem ein zweites Array angelegt, in dem pro Anfragezeile der zugehörige „Remote“-Prozess eingetragen ist. Die Übermittlung der Daten erfolgt dann zunächst synchron, sodass die Übertragung erst beginnen kann, wenn alle lokalen Anfragen von allen Prozessen gesammelt wurden.

Abbildung 3.11 zeigt diesen Prozess für einen einfachen 2D-Beispielfall. Die im Beispiel zu generierende Tabelle besteht der Einfachheit halber zunächst nur aus einer Zeile für den CFD-Punkt der Knoten-ID 9. Die Anzahl Stützstellen wurde mit $n_{next} = 10$ festgelegt.

Bearbeitung der Nächste-Nachbarn-Suchanfragen

Die zuvor über die Anfragetabelle gesammelten Suchanfragen können nun zur Bearbeitung und Lokalisation der nächsten Strukturknoten genutzt werden. Diese Form des Ablaufs hat zudem den Vorteil einer effizienteren Programmsteuerung. Jeglicher Datentransfer, dies betrifft die Übertragung der nächsten Strukturknoten und die Bearbeitung des verteilten Matrix-Vektor-Produkts innerhalb des Last- und Verformungstransfers (siehe Abschnitt 3.3.2), kann nun asynchron durchgeführt werden, da sich Sender und Empfänger (in Abbildung 3.11 sind dies Prozess 4 und 2) nun „kennen“.

Zur Übermittlung der Strukturknoten kann anstelle eines 2-dimensionalen Arrays ein lineares Array verwendet werden, das gezielt über Ausgabe-Puffer an entfernte Prozesse versendet werden kann. Darüber hinaus kommt ein Mehrpuffer-Modus zum Einsatz, sodass die Daten über getrennte Puffer versendet werden können. Jeder von n-Prozessen besitzt dabei somit n Ein- und Ausgabepuffer. Ein über entsprechende Puffer versendetes Array mit Punktinformationen der Stützstellen

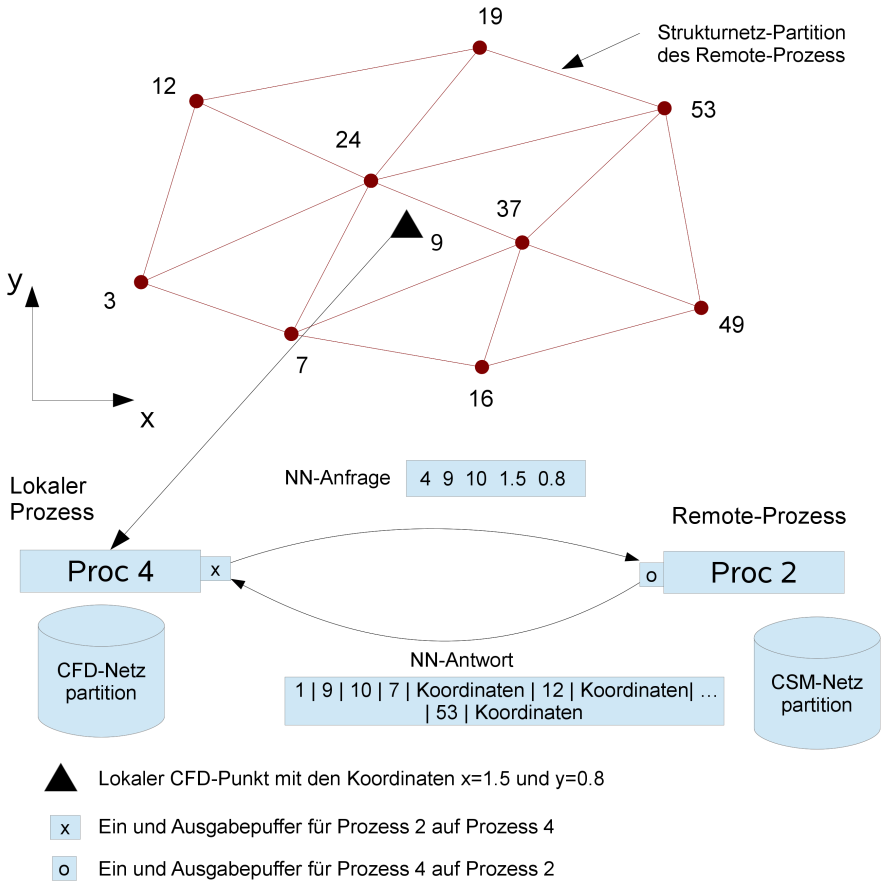


Abbildung 3.11: Grafische Darstellung des Anfrageprozesses

lässt sich nun, wie in Abbildung 3.12 geschehen, grafisch in seinem Grundaufbau darstellen.

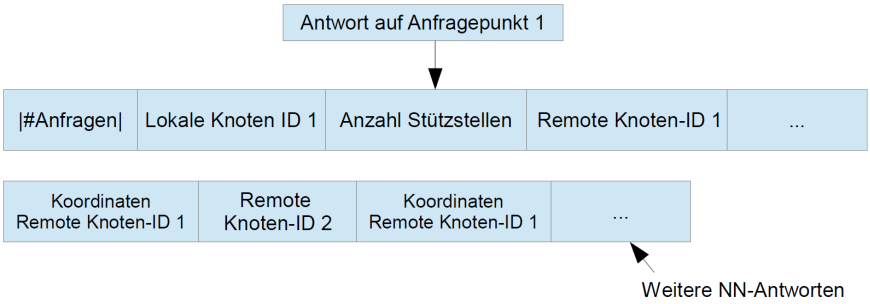


Abbildung 3.12: Schematische Darstellung der Puffer-Daten

Der erste Eintrag (|#Anfragen|) in Abbildung 3.12 kennzeichnet dabei wie viele Anfragen zuvor gesendet wurden, d.h. wie viele CFD-Oberflächenpunkte von einem Prozess potentielle CSM-Stützstellen eines anderen Prozesses erhalten sollen. In

Abbildung 3.11 wurde dies, da im Beispiel nur ein einziger CFD-Punkt existiert, mit der Zahl 1 gekennzeichnet. Weitere Einträge des Arrays kennzeichnen dann für jede einzelne Anfrage, welche Knoten-ID der CFD-Punkt auf Sender-Seite besitzt und welche Knoten-IDs und Koordinaten die Stützstellen besitzen.

Für eine effiziente Durchführung der Kommunikation zwischen den Prozessen müssen hierbei zudem Statistiken im Rahmen der Anfrageerstellung generiert werden. Diese werden nach der Übermittlung der eigentlichen Anfragen an alle Prozesse geschickt. Ein solche Statistik speichert dabei für jeden Prozess, wie viele Anfragen und wie viele Stützstellen insgesamt (über jeden CFD-Punkt eines Prozesses aufsummiert) von einem anderen Prozess übermittelt wurden. Dies ermöglicht lokal eine Vorherbestimmung der Paketgröße der zu übermittelnden Stützstellen für einen anderen Prozess.

Zusammenfassung

Mithilfe der im Kapitel beschriebenen Methoden ist nun eine parallele Form eines Bauteil-basierten Last- und Verformungstransfers nach der Quaranta-Methode im FSCouple-Modul verfügbar. Die geschilderten und umgesetzten Ansätze werden im nächsten Kapitel anhand ausgewählter Testfälle evaluiert.

4 Ergebnisse

Die in Kapitel 3 beschriebene Umsetung der Quaranta-Methode im Rahmen des FlowSimulator-Framework soll im Folgenden anhand von einzelnen numerischen Experimenten getestet werden. Neben Analysen der Robustheit der parallelen Umsetzung im Rahmen realitätsnaher, aeroelastischer Testkonfigurationen wird das Augenmerk auch auf die Bestimmung der parallelen Performance gelegt.

An dieser Stelle wird jedoch in den Analysen auf eine vollständig gekoppelte Rechnung unter Nutzung der Quaranta-Methode in der CFD-CSM-Prozesskette aus Kapitel 1.2 verzichtet. Stattdessen werden sämtliche Teilabschnitte des Kopplungsalgorithmus anhand generischer Last- u. Verformungsfelder verifiziert. Damit lässt sich die korrekte Funktionsfähigkeit der implementierten Quaranta-basierten Methodik, wie sie in einer statischen aeroelastischen Simulation genutzt würde, hinreichend nachweisen.

4.1 Untersuchte Testfälle

Die im Rahmen dieses Kapitels untersuchten Testfälle seien nachfolgend kurz bezüglich ihren Eigenschaften und Besonderheiten vorgestellt. Bei allen untersuchten Testfällen wurde bei der Interpolation mit der Quaranta-Methode ein lineares Polynom sowie die Wendland-C2-Funktion als radiale Basisfunktion verwendet (siehe Kapitel 2.2 und 2.3). Die zur Interpolation notwendige Bestimmung der Koeffizienten der Kopplungsmatrix in Gleichung 2.35 bzw. 2.36 wird mithilfe einer Singulärwertzerlegung umgesetzt, welche durch Nutzung der Routinen der LAPACK-Bibliothek softwaretechnisch realisiert wurde.

4.1.1 Einfaches Rechteckgitter

Zur allgemeinen Untersuchung der Performance und des Ressourcenaufwands bei der parallelen Ausführung wird ein einfaches Rechteckgitter genutzt, dessen x- bzw. y-Koordinatenbegrenzung sowie Anzahl der Gitterpunkte vom Nutzer beliebig festgelegt werden kann. Damit lassen sich im Gegensatz zu den nachfolgend beschriebenen Flugzeug-Konfigurationen und Modelle, welche feste Knotenanzahlen aufweisen, gezielt Einflüsse von Netzgrößen im Rahmen der parallelen Performance-Tests untersuchen. Die Anzahl der Gitterpunkte für das CFD-Oberflächen- und CSM-Strukturnetz ist jedoch annähernd gleich gewählt worden.

Während das CSM-Strukturnetz als strukturiertes Gitter generiert wurde, wurden die Punkte des CFD-Oberflächennetzes innerhalb des Rechtecks zufallsbasiert mithilfe eines gezielt aufgetragenen Rauschens um ihre zuvor erzeugten Koordinaten verschoben (siehe Abbildung 4.1) . Obwohl die Knotenanzahlen annähernd gleich bleiben, fallen CFD-Oberflächenknoten und Stützstellen somit räumlich nicht zusammen und ermöglichen eine sinnvolle Untersuchung des Last- bzw. Verformungstransfers.

Für eine korrekte Durchführung der Interpolation mit der Quaranta-Methode muss das Rechteckgitter in ein dreidimensionales Netz überführt werden. Im Falle eines zweidimensionalen Netzes würde es aufgrund der Tatsache, dass dann Stützstellen in einer Ebene liegen, zu Singularitäten bei der Lösung der Gleichungssysteme innerhalb der Koeffizientenberechnung (Gleichung 2.35) führen. Um dies zu vermeiden, wurde die z-Koordinate der Netzpunkte im Rechteckgitter gleich der Entfernung zum Zentrum des Netzes gewählt.

4.1.2 HiReTT-Konfiguration

Obwohl das zuvor beschriebene Rechtecknetz sich zu ersten Performanceuntersuchungen der parallelen Methode eignet, so stellt es jedoch keine repräsentative Netzkonfiguration im Rahmen von aeroelastischen Simulationsstudien dar.

Für eine genaue Einschätzung der Anwendungstauglichkeit der Methode müssen daher realitätsnahe Modelle einzelner Flugzeugkomponenten herangezogen werden. Für die Arbeit kommt dafür ein spezielles Flügelmodell zum Einsatz, im folgenden auch als HiReTT-Flügel bezeichnet.[1] Dieses in Abbildung 4.2 und 4.3 dargestellte Modell ist ein im Rahmen von Forschungsprojekten bereits mehrfach untersuchter aeroelastischer Testfall[16][17]. Im Vergleich zum Aufbau realer Flugzeugkonfigurationen ist es dennoch nur bedingt repräsentativ, da es sich um ein Windkanalmodell handelt. Windkanalmodelle sind von ihrem inneren Strukturaufbau zumeist als Vollmodelle ausgeführt, während reale Flugzeugstrukturen eher Strukturen in dünnwandiger Kastenträgerbauweise sind. Im Gegensatz zum Rechteckgitter eignet sich jedoch besser für realitätsnahere Untersuchungen und ist zudem für die am HiReTT-Projekt beteiligten Partner frei zugänglich.

Das CFD-Oberflächennetz[2] besteht überwiegend aus Quadrilateralelementen mit insgesamt etwa 130 000 Knoten. Das CSM-Strukturnetz besteht aus einem Volumennetz mit Hexaeder- und Pentaederelementen mit insgesamt etwa 3600 Knoten. Im Inneren sind zudem Kabelschächte modelliert. Sie dienen zur Führung von Kabeln, die in den HiReTT-Windkanaltests zur Übertragung der an der Modelloberfläche gemessenen Drucksensordaten eingesetzt wurden. Zur Erprobung der

[1] HiReTT Projekt:High Reynolds Number Tools and Techniques for Civil Transport Aircraft Design

[2] Das dazugehörende Volumennetz wurde mit der SOLAR-Software erstellt. Es besteht in der Grenzschicht hauptsächlich aus Hexaederelementen, im Fernfeld überwiegend aus Tetraederelementen.

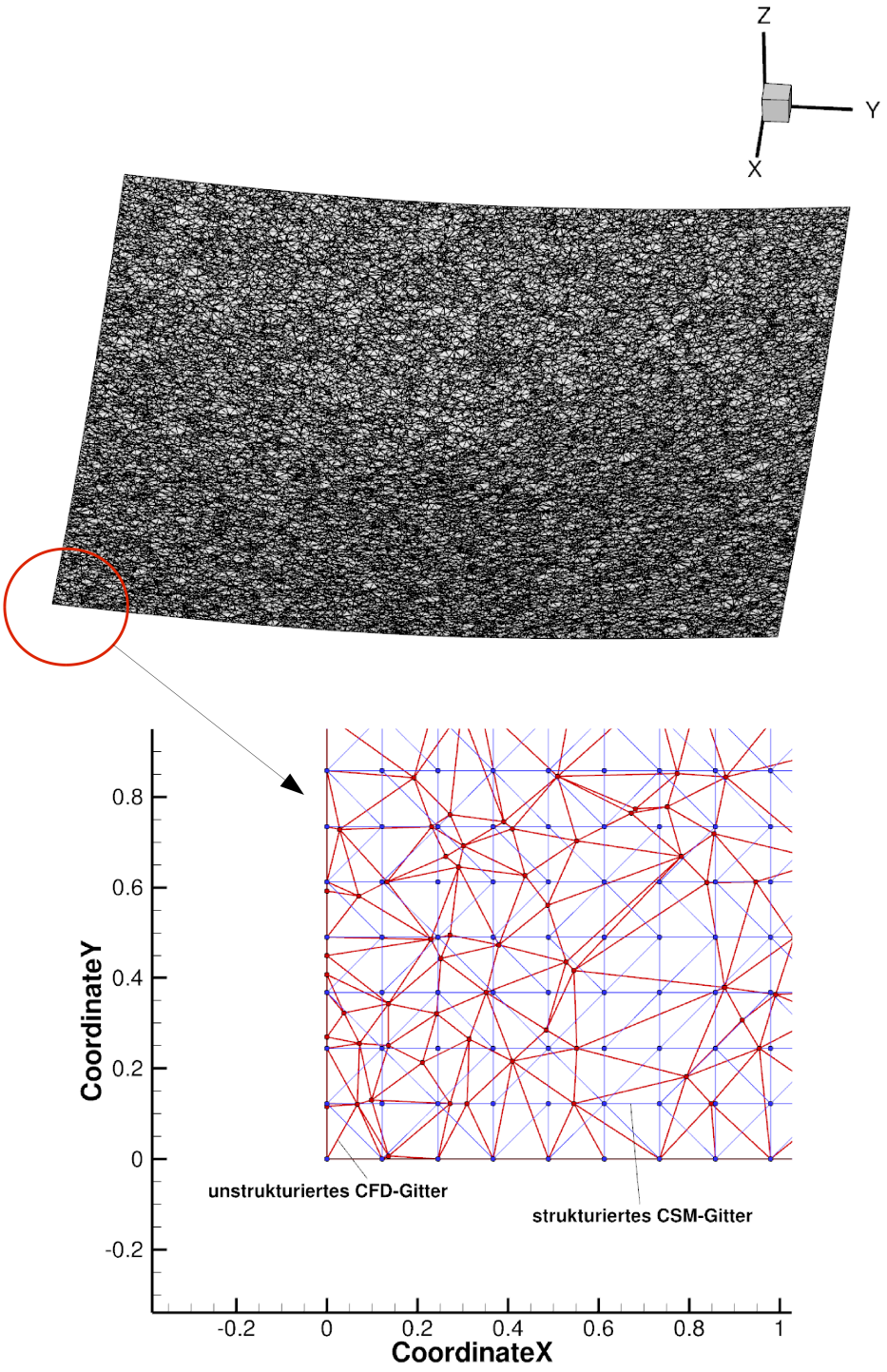


Abbildung 4.1: Generisches Rechteckgitter aus ∼32000 CFD- und CSM-Gitterpunkten

Quaranta-Methode wird jedoch auf die innere Struktur verzichtet. Stattdessen wird auf CSM-Strukturnetzseite nur die Oberfläche des Flügelnetzes verwendet. Die auf CFD-Netzseite modellierten Spoiler- und Querruderausschläge sind auf Strukturseite nicht modelliert. Dies stellt eine weitere Herausforderung an das Interpolationsverfahren. Entsprechende Details des HiReTT-Modells sind in den Abbildungen 4.2 und 4.3 zu sehen. Der in Abbildung 4.2 dargestellte Rumpf, der eigentlich auf CFD-Seite offiziell zur HiReTT-Konfiguration gehört, wird im Rahmen des Last-/Verformungsinterpolationsproblems, auf das sich diese Arbeit konzentriert, nicht betrachtet.

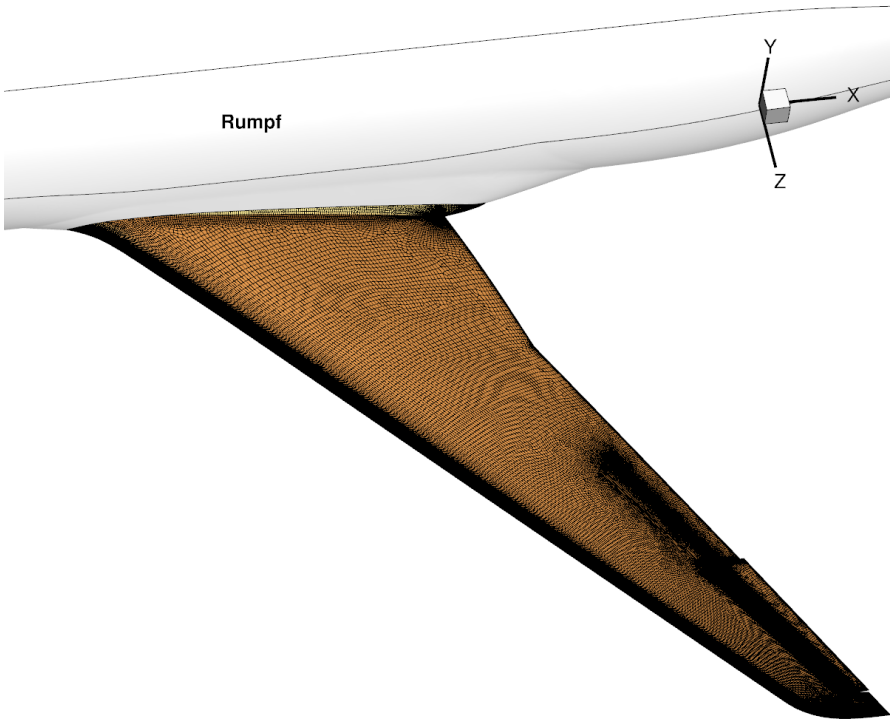


Abbildung 4.2: CFD-Oberflächennetz des HiReTT-Flügels

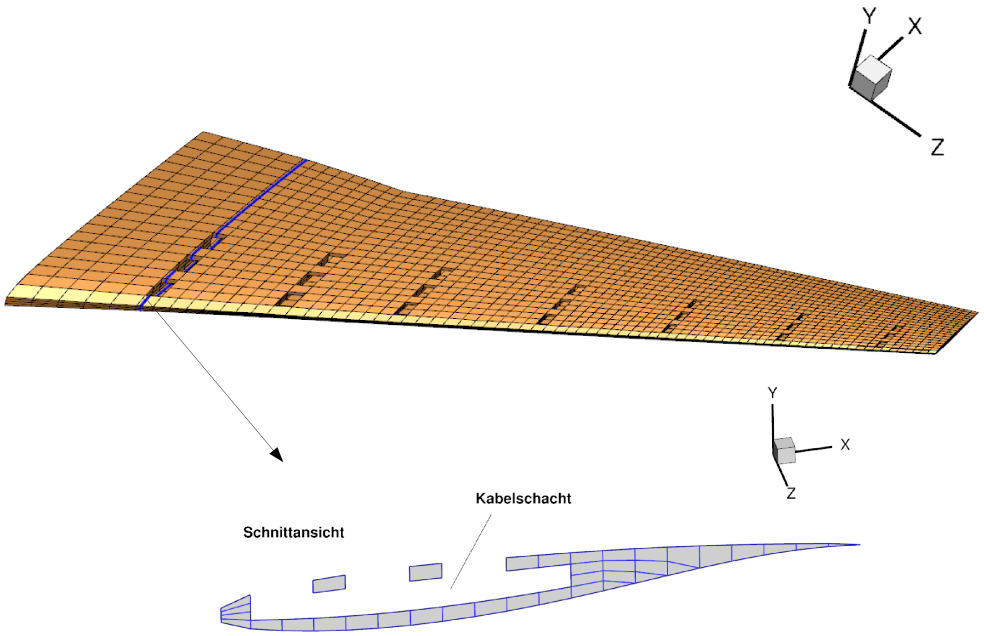


Abbildung 4.3: CSM-Strukturnetz des HiReTT-Flügels

4.1.3 Flügelkasten-Modell

Da die HiReTT-Konfiguration nur ein Windkanal-Modell darstellt, kann sie, wie beschrieben, nur bedingt für die Abbildung realer Flügelstrukturen genutzt werden. Zur Verifikation der umgesetzten parallelen Quaranta-Methodik im aeroelastischen Kontext wird daher ein sogenanntes „Flügelkasten-Modell“ genutzt. Im Gegensatz zur Struktur des HiReTT-Modells besteht das Strukturnetz dabei im Inneren aus schalenförmigen Strukturen, die im Flugzeugkomponentenbau typischerweise verwendet werden. Abbildung 4.4 und 4.5 zeigen die für das Modell verwendeten CFD-Oberflächen- ($\sim$ 41300 Knoten) bzw. CSM-Strukturnetze ($\sim$ 7850 Knoten).

4.1.4 Mehrkomponenten-Konfiguration

Bei den zuvor beschriebenen Testfällen wurde die in Kapitel 2.4 vorgestellte Blending-Methode noch nicht eingesetzt. Bei den Testfällen handelte es sich bisher stets um einzelne Baugruppen. Im Fall der HiReTT-Konfiguration bspw. um die Zuordnung "Flügel-Flügel".

Für die Analyse der Blending-Methode im Zusammenhang der Bauteil-basierten räumlichen Kopplung mit der Quaranta-Methode wird deshalb ein aus mehreren Komponenten bestehendes Windkanalmodell der DLR-F11-Konfiguration in Hochauftriebsanordnung benutzt.

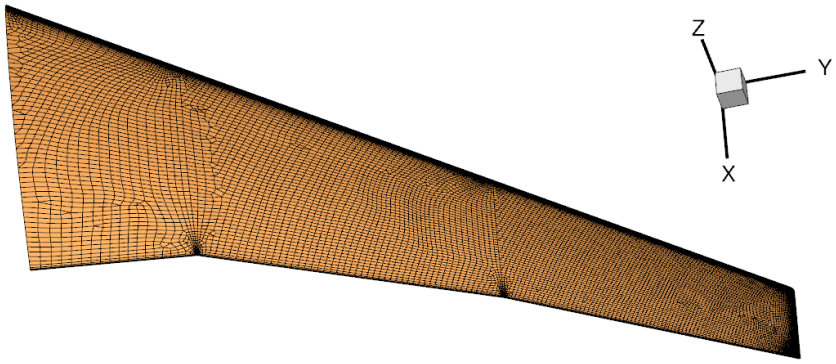


Abbildung 4.4: CFD-Oberflächennetz des Flügelkasten-Modells

Abbildung 4.5: CSM-Strukturnetz des Flügelkasten-Modells

Wie in Abbildung 4.6 zu sehen, besteht die Flugzeugkonfiguration aus 5 Komponenten: Rumpf (rot), Vorflügel (blau), Tragflügel (grün), Landeklappen (orange) und Verkleidungen (auch „Flap-Track-Fairings“ genannt) (gelb). Die Komponenten des CFD-Oberflächennetzes in Abbildung 4.6 bestehen dabei insgesamt aus etwa 340 000 Knoten. Das entsprechende Strukturnetz dieser Konfiguration, zu sehen in Abbildung 4.7, besteht aus etwa 9700 Knoten. Während Landeklappen, Vorflügel und Tragflügel auch strukturseitig an der Kopplungsfläche aus flächigen Elementtypen (in diesem Fall Schalenelemente) aufgebaut sind, sind Rumpf und Flap-Track-Fairings mit eindimensionalen Balkenelementen modelliert. Die Balkenelemente zur Modellierung der Flap-Track-Fairings werden aber im Rahmen des Verformungstests aus zwei Gründen nicht verwendet: (1) Die Quaranta-Methode ist in der in dieser Arbeit umgesetzen Form nicht in der Lage auf Balkenelemente angewendet zu werden (siehe Erläuterungen in Kapitel 2.3). (2) Die im FlowSimulator außer-

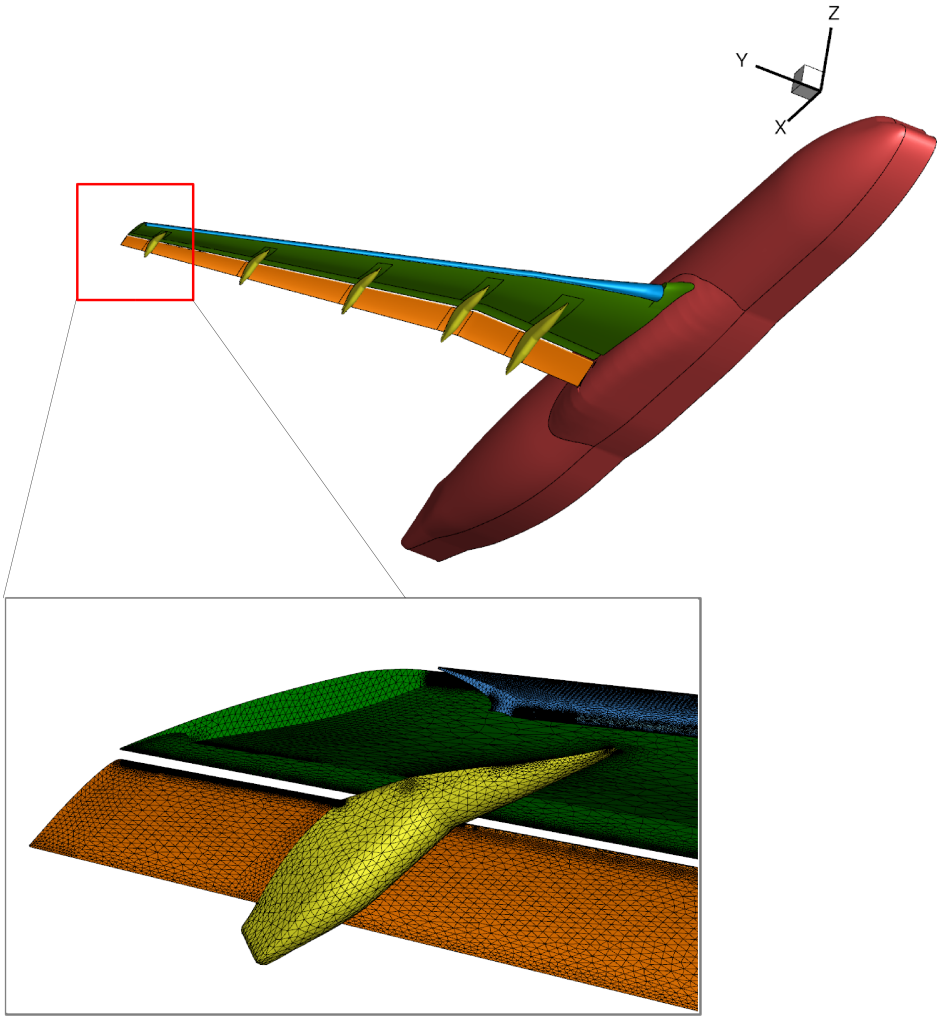


Abbildung 4.6: Komponenten-basiertes CFD-Oberflächennetz eines DLR-F11-Modells in Hochauftriebskonfiguration

dem verfügbare Implementierung der Finiten-Interpolationselemente berücksichtigt bisher nicht die für das Verformungsfeld wesentlichen Verdrehungsinformationen bei Balken. Sie ist daher ebenfalls nicht tauglich. Ausschließlich der Balken zur Modellierung der Rumpfstruktur wird bei der Verformungsinterpolation berücksichtigt. Für die Projektion wird die Methode der Finiten-Interpolationselemente verwendet. Sie funktioniert im speziellen Fall des Rumpfes, da für diesen Nullverschiebungen vorgeschrieben wurden. Wie in Bezug auf die Verformungsinterpolation bei den Flap-Track-Fairings verfahren wurde, wird in Kapitel 4.3.2 beschrieben (im Zusammenhang mit dem eigentlichen Interpolationstest für das F11-Modell).

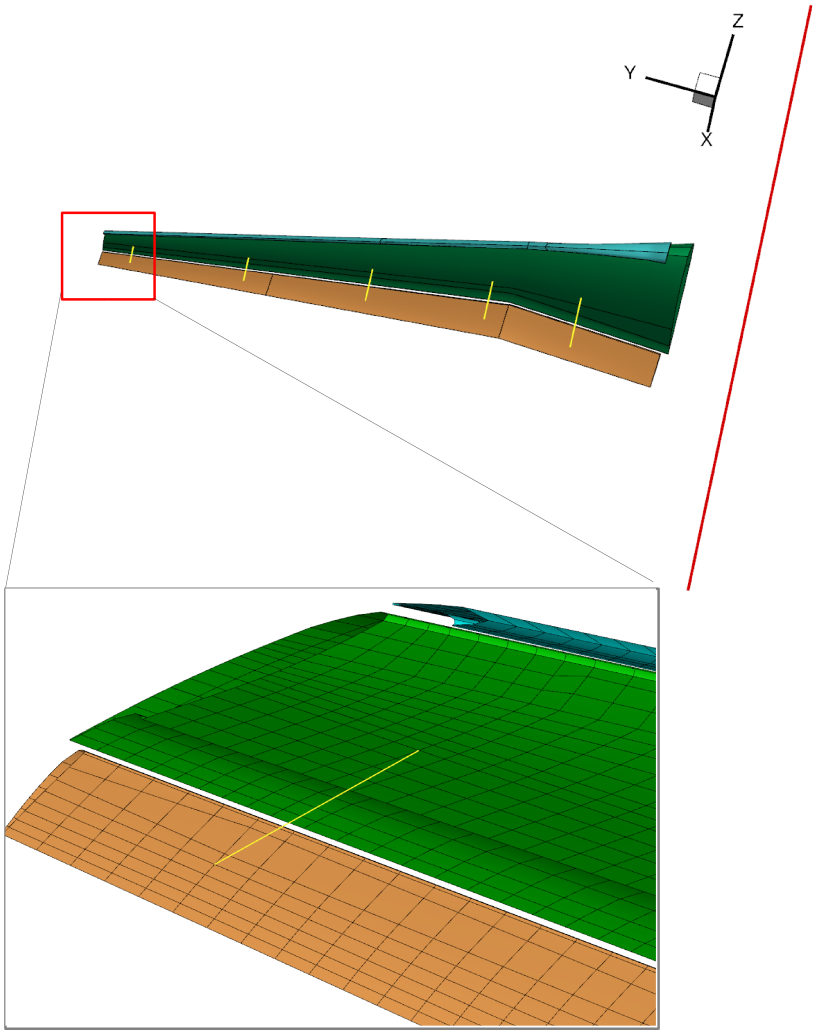


Abbildung 4.7: Oberflächennetz des Strukturmodells eines DLR-F11-Modells in Hochauftriebskonfiguration

4.2 Durchgeführte Tests

Zur Überprüfung der numerischen und leistungstechnischen Eigenschaften der umgesetzten Quaranta-Methode wurden folgende Tests durchgeführt:

1. Untersuchungen an Einkomponenten-Konfigurationen

    a) Untersuchung des sequentiellen Algorithmus
    Dies dient dazu, die Komplexität des sequentiellen Algorithmus zu bestimmen. Die Untersuchung bildet die Grundlage für die anschließende Beurteilung der Komplexität des parallelen Algorithmus. Dabei wird der Zeitbedarf

        i. bei gegebener Netzgröße und Variation der Anzahl geforderter Stützstellen sowie

   ii. bei Variation der Netzgröße

   am Beispiel des generischen Rechteckgitters betrachtet.

b) Performance-Untersuchungen bei paralleler Ausführung
   Zur Beurteilung der parallelen Gesamtperformance wird anhand des generischen Rechteckgitters folgendes betrachtet:
   i. ermittelter Speedup bei Steigerung der Prozessoranzahl
   ii. Zeitbedarf bei Veränderung der Netzgröße
   iii. Zeitbedarf zum Versenden und Empfangen Nächster-Nachbarn als Indikator der parallelen Kommunikationskomplexität
   iv. Robustheits-Untersuchungen
   Der Einfluss der gegebenen Prozessorzahlen sowie Partitionierung auf die Qualität der Last-/Verformungsinterpolation wird untersucht. Für diesen Test wird statt des generischen Rechteckgitters die HiReTT-Konfiguration und die Konfiguration mit Flügelkastenträgerstruktur verwendet.

2. Analyse von komplexen Mehrkomponenten-Flugzeugkonfigurationen
   a) Interpolation bei Bauteil-basierter Zuordnung der Teilnetze **ohne** zusätzliches Blending
   b) Interpolation bei Bauteil-basierter Zuordnung der Teilnetze **mit** zusätzlichem Blending

Die in den einzelnen Teiluntersuchungen durchgeführten Zeitmessungen am generischen Rechteckgitter beinhalten dabei neben der Bestimmung der Kopplungsmatrix zudem die Ausführung eines Matrix-Vektor-Produkts einer skalaren Größe, welche auf dem jeweiligen Strukturnetz aufgetragen wurde.

Ziel der oben definierten Testszenarien ist der schrittweise Nachweis der Funktionsfähigkeit der implementierten parallelen Version der Quaranta-Methode inklusive Bauteil-basierter Interpolation mit Blending. Insbesondere die parallele Performance und die Robustheit der Implementierung soll studiert werden.

Die generelle Funktionsfähigkeit der seriellen Quaranta-Methode wird nachfolgend nicht zur Disposition gestellt. Sie wurde hinlänglich von Quaranta selbst [1] und von [5] nachgewiesen.

4.3 Analyse von Einkomponenten-Konfigurationen

4.3.1 Performance-Untersuchungen der sequentiellen Quaranta-Methode

Für eine korrekte Bewertung der parallelen Umsetzung der Quaranta-Methode wird zunächst kurz auf die Performance der sequentiellen Methoden eingegangen. Aufgrund der Charakteristik der Interpolationsmethode ist zu erwarten, dass sowohl

die Anzahl der Stützstellen als auch die gewählte Netzgröße die entscheidenden Kenngrößen bei der Performanceanalyse darstellen.

Für den sequentiellen Test der Methode wurde folgende Rechnerkonfiguration benutzt:

- Intel Core i7 870 2,93 GHz (nur 1 Kern von 8 möglichen genutzt)
- 8GB RAM
- SUSE Linux Enterprise Desktop
- gcc-Compiler 4.7.0 mit Optimierungsstufe 3
- Testfall: generisches Rechteckgitter

Abbildung 4.8 zeigt dabei den Einfluss der Variation der Stützstellen-Anzahl bei etwa 130 000, 62000 und 32000 Oberflächen bzw. Strukturknoten.

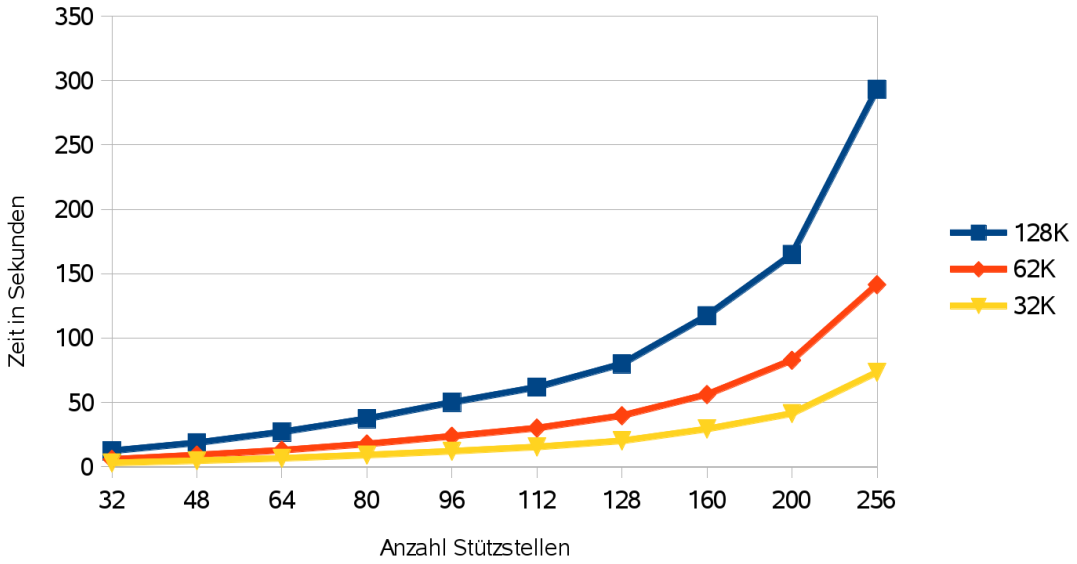


Abbildung 4.8: Benötigte Zeit bei Variation der Stützstellenzahl und gegebener Netzgröße

Wie in Abbildung 4.8 zu sehen, wächst der Berechnungsaufwand der Methode zunächst nur sehr moderat. Während bei einer Netzgröße von etwa 32000 Knoten (gelbe Linie) bis zu einer Anzahl von 200 Stützstellen ein nahezu linearer Anstieg des Berechnungsaufwandes zu erkennen ist, steigt der Berechnungsaufwand bspw. bei 128K Knoten (blaue Linie) ab etwa 128 Stützstellen spürbar an. Grund dafür ist zum einen der erhöhte Berechnungsaufwand der Singulärwertzerlegung im Rahmen der Koeffizientenberechnung für eine Zeile der Kopplungsmatrix. Zudem steigt der Aufwand zur Ermittlung der strukturseitigen Stützstellen mithilfe des k-d-Baums.

Eine Variation der Netzgrößen im Vergleich dazu bei 32, 64 und 96 Stützstellen führt zu dem in Abbildung 4.9 dargestellten Ergebnis. Im Vergleich zur Variation

der Stützstellenanzahl steigt der Berechnungsaufwand etwa linear mit der Vergrößerung der Punktanzahl, sodass bei Verdopplung der Netzgrößen auch die doppelte Dauer beansprucht wird. Da jedoch sowohl die Größe der Kopplungsmatrix als auch die Anzahl insgesamt vorhandener Strukturknoten als Stützstellen skaliert werden, hat die Größe der k-d-Baumstruktur zur Stützstellensuche somit einen unwesentlicheren Einfluss auf die Berechnungszeit als die eigentliche Berechnung der Interpolationskoeffizienten.

Aus der Kombination von Abbildung 4.8 und 4.9 lässt sich ein nicht-linearer Einfluss der Anzahl verwendeter Stützstellen feststellen. Bei einer Verdoppelung der Stützstellen steigt der Aufwand durchschnittlich etwa um den Faktor 2.5 , bei einer Verdreifachung jedoch etwa um den Faktor 4.75.

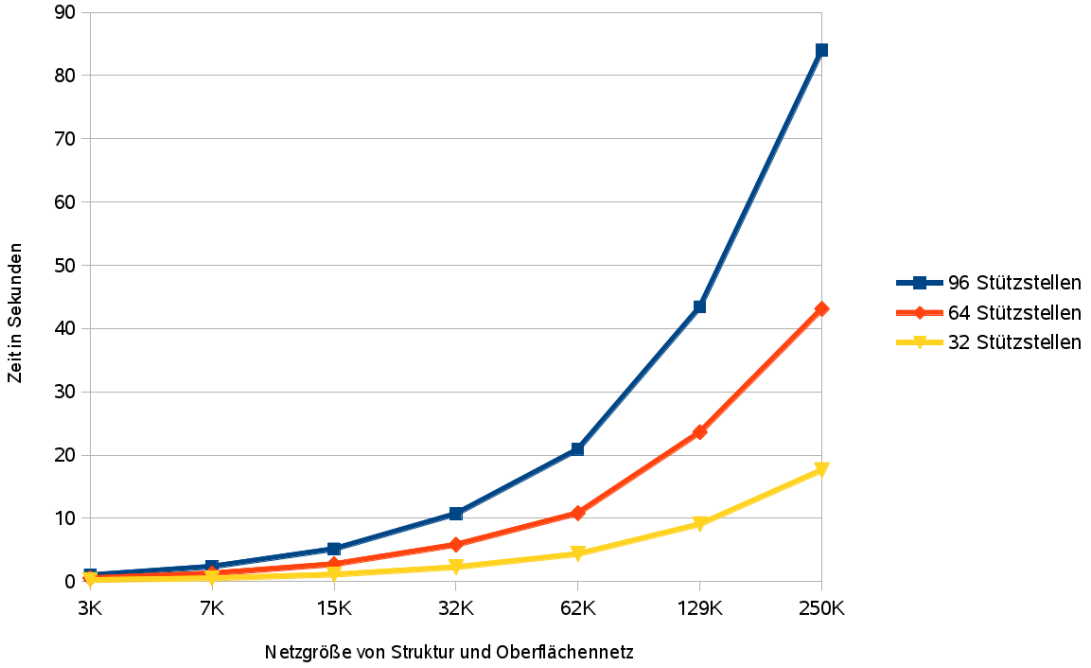


Abbildung 4.9: Berechnungsaufwand bei Variation der Netzgröße und 32,64 und 96 Stützstellen

Anhand der Ergebnisse lässt sich insgesamt festhalten, dass die sequentielle Quaranta-Methode zunächst nur für moderate Netzgrößen und Stützstellenanzahlen verwendet werden sollte. Bei größeren Netzen und Stützstellenanzahlen ist der Einsatz der parallelen Methode zu empfehlen.

4.3.2 Analysen zur parallelen Quaranta-Methode

Bevor auf die Untersuchung der parallelen Performance eingegangen wird, soll für das gewählte Rechtecknetz zur Verdeutlichung der parallelen Funktionalität

ein erster rudimentärer Funktionstest durchgeführt werden. Hierbei wurde auf dem Strukturnetz ein skalares Feld als fiktive „Verschiebung“ definiert. Es ist in Abb. 4.10 auf dem „Strukturnetz“ dargestellt. Das Verschiebungsfeld wurde dann mit dem parallelen Algorithmus in das CFD-Oberflächennetz transformiert. Abbildung 4.11 zeigt das Interpolationsergebnis, welches nach Ausführung des parallelen Algorithmus auf dem partitionierten CFD-Netz vorlag. Die Ähnlichkeit von Ausgangsfeld und interpoliertem Feld demonstriert erfolgreich die generelle Funktionsfähigkeit des parallelen Algorithmus.

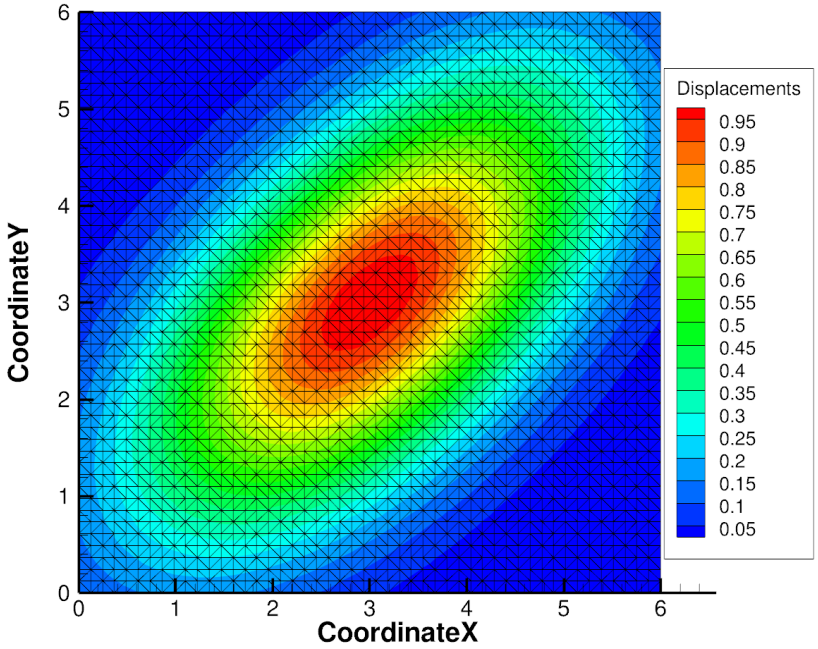


Abbildung 4.10: Vorgegebenes, eindimensionales Feld auf dem Strukturnetz aus etwa 2500 Knoten

Performance-Untersuchungen

Anhand der vorherigen Schilderung wird deutlich, dass die sequentielle Interpolation mithilfe der Quaranta-Methode stark von der Netzgröße und Anzahl verwendeter Stützstellen abhängt. Da für eine effektive und genaue Interpolation im Kontext aeroelastischer Simulationen sowohl große Netze als auch u. U. eine hohe Anzahl an Stützstellen benötigt wird, sind Performancesteigerungen durch Parallelisierung der Methode von essentieller Bedeutung.

Im Folgenden wird zur Ermittlung eines möglichen Beschleunigungspotentials das generische Rechteckgitter mit jeweils etwa 250000 Netzknoten benutzt. Dies entspricht in etwa der Größenordnung gängiger Netzkonfigurationen im Rahmen aeroelastischer Fragestellungen. Obwohl Strukturnetze dabei in der Regel eine

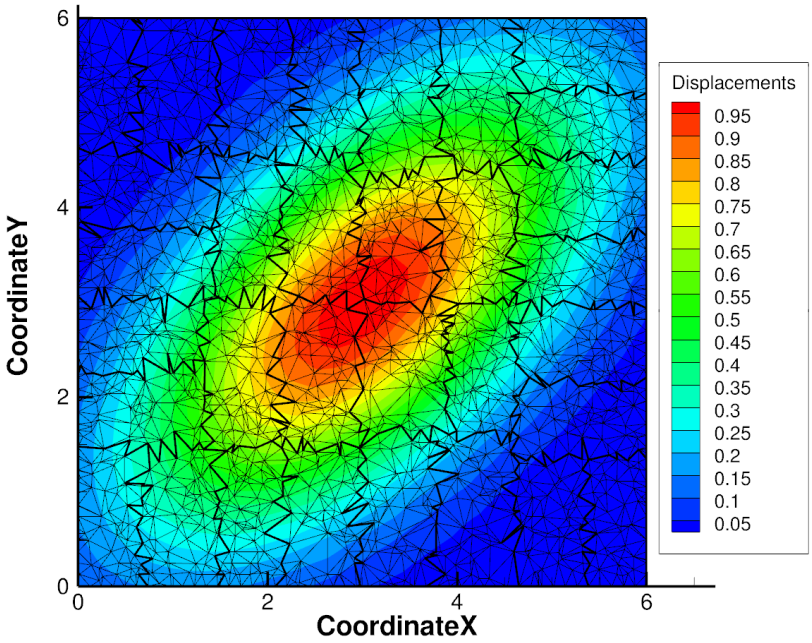


Abbildung 4.11: Rekonstruiertes, skalares Feld auf dem CFD-Oberflächennetz bei 32 Partitionen und RCB-basierter Partitionierung

geringere Auflösung aufweisen, können mit der hier gewählten Größe u. U. bestimmte Aspekte paralleler Komplexität aufgezeigt werden. Aufgrund des künstlich erhöhten Berechnungsaufwands der Nächsten-Nachbarn im Strukturnetz stellen hier gemessene Zeiten somit eine untere Grenze für realitätsnahe aeroelastische Berechnungen dar.

Das CFD-Oberflächennetz wurde im Weiteren mithilfe des RCB-Partitionierungsalgorithmus und das CSM-Strukturnetz mithilfe des Zoltan-Toolkits in einzelne Partitionen zerlegt. Es wurden zudem 64, 128 und 256 Stützstellen verwendet, sowie die Anzahl an parallelen Prozessen zwischen 1 und 96 variiert. Die beanspruchte Zeit ist dabei in Abbildung 4.12 aufgetragen.

Deutlich anhand Abbildung 4.12 zu erkennen, ist die Tatsache, dass die Anzahl verwendeter Stützstellen besonders im Bereich geringer Prozessorkernanzahlen Einfluss auf den Zeitaufwand hat. Dabei sinkt der Berechnungsaufwand bei 256 Stützstellen bei steigender Prozessoranzahl deutlicher bis zu einer Prozessoranzahl von etwa 24 im Vergleich zu 64 und 128 Stützstellen. Ab einer Prozessorzahl von 32 ist für alle 3 gewählten Stützstellenanzahlen eine geringere Reduzierung des Zeitaufwands zu erkennen.

Im Zusammenhang zur absolut gemessenen Zeit wird zur Bestimmung des Speedups, welcher in Abbildung 4.13 dargestellt ist, die gängige Beziehung

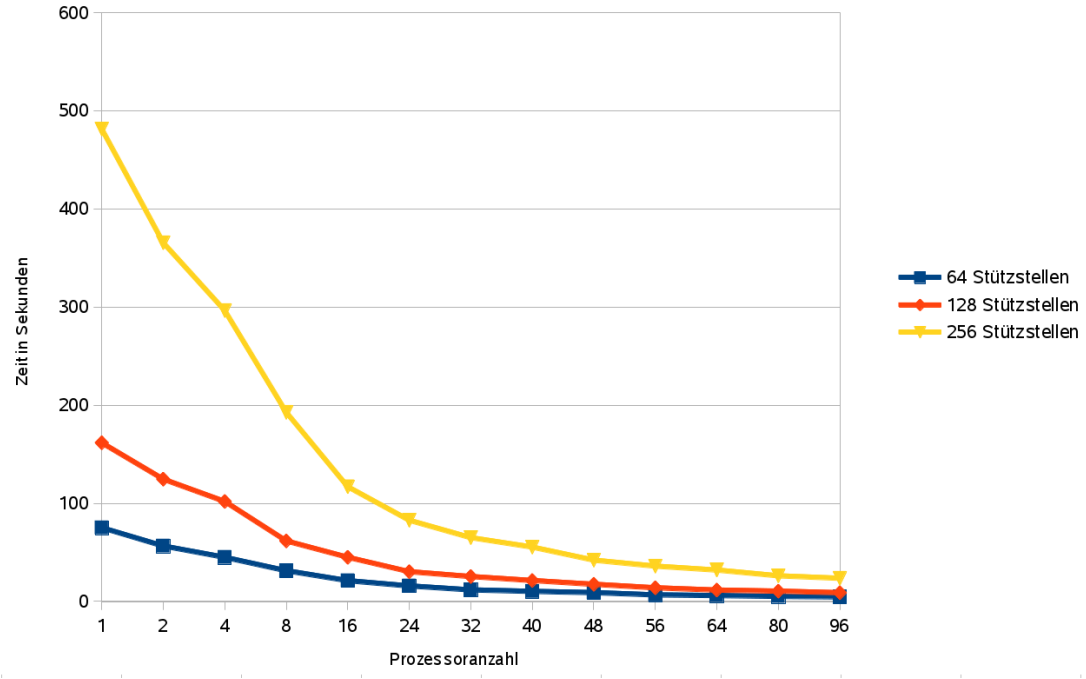


Abbildung 4.12: Zeit zur Berechnung der Kopplungsmatrix und Interpolation bei unterschiedlicher Prozessor- und Stützstellenanzahl

$$Speedup = \frac{T_s}{T_p} \tag{4.1}$$

verwendet, die das Verhältnis des Zeitaufwands bei sequentieller Berechnung T_s zur parallelen Ausführung T_p beschreibt (siehe auch [18]). Der Vollständigkeit halber soll hier noch erwähnt werden, dass für p=1 nicht die sequentielle Implementierung, sondern die parallele Codeversion verwendet wurde, jedoch nur auf einem einzigen Prozess. Die im Folgenden verwendete Bezeichnung der Prozessoranzahl kennzeichnet die Menge physisch vorhandener Recheneinheiten.

Im Vergleich zur absolut gemessenen Zeit ist in Abbildung 4.13 im Bereich geringer Prozessoranzahlen eher nur ein geringer Anstieg des Speedups zu erkennen. Ab 8 Prozessen ist im Weiteren Verlauf dann allerdings ein linearer Anstieg des Speedups zu verzeichnen. Interessant ist zudem die Tatsache, dass mit der Erhöhung der Anzahl verwendeter Stützstellen sogar ein leicht erhöhter Speedup zu erreichen ist. Dieser kann u.a. damit begründet werden, dass trotz erhöhter Datenmengen (NN-Punktkoordinaten) im Rahmen der Kommunikation ein höherer Durchsatz erzielt wird. Die den Nachrichtenprozess verzögernde Latenz zum Aufbau einer Datenverbindung hat dabei einen geringeren Anteil am Prozess des Nachrichtenaustauschs.

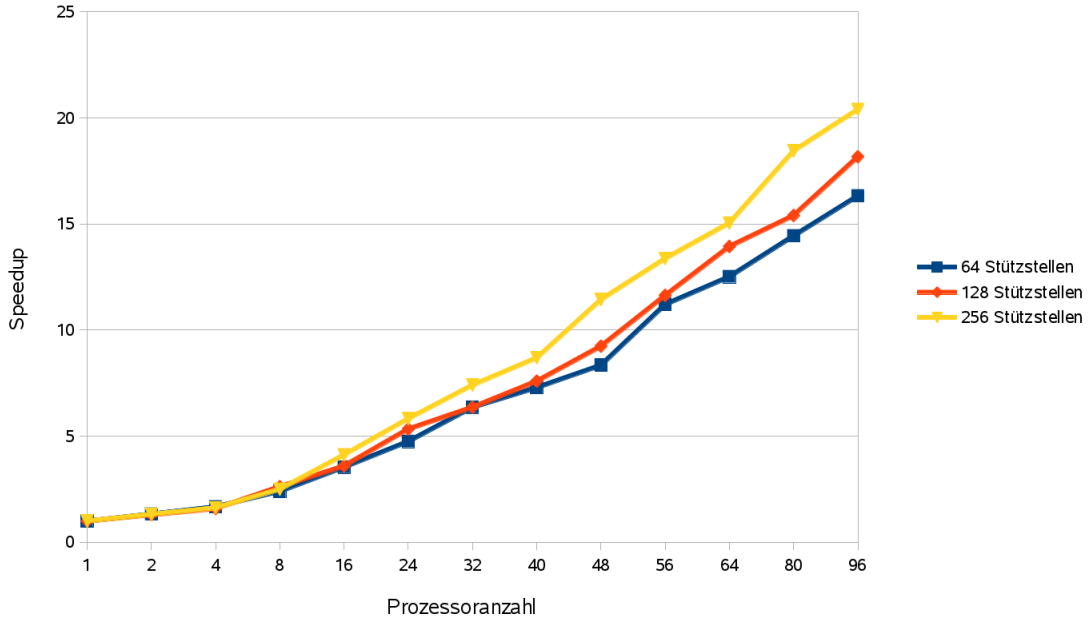


Abbildung 4.13: Gemessener Speedup für das gewählte Rechtecknetz aus etwa 250000 Oberflächen- und Strukturknoten

Im Zusammenhang dazu ist in Abbildung 4.14 die durchschnittlich gemessene Zeit zum Versenden und Empfangen aller Nächster-Nachbarpunkte eines einzelnen Prozesses bei steigender Prozessorzahl dargestellt. Sowohl für n=128 als auch n=256 Stützstellen ist deutlich zu erkennen, dass zunächst ein erhöhter Kommunikationsaufwand bis p=4 Prozessoren benötigt wird. Ab p=8 sinkt daraufhin allerdings die Zeitdauer zum Versenden der Daten unter den Prozessen. Dies lässt sich auf die zunehmende Lokalität der Suchanfragen und die Bounding-Box-basierte Suche der nächsten Partitionen zurückführen. Bei geringer Anzahl an Partitionen findet besonders an einzelnen Partitionsgrenzen eine Überlappung aller Bounding-Boxen statt, sodass NN-Suchanfragen nahezu an alle Prozesse versendet werden. Der entstehende Kommunikationsprozess kommt infolgedessen einer All-to-All Kommunikation nahe. Da jedoch mit zunehmender Prozessoranzahl die Anzahl an CFD-Oberflächenknoten sinkt, ist trotz erhöhtem Kommunikationsaufwand die parallele Umsetzung bereits ab zwei Prozessen schneller als die sequentielle Ausführung. Ab einer größeren Anzahl an Prozessen wird die Lokalität der Partitionen sehr gut ausgenutzt, sodass einzelne Prozesse autark im Vergleich zu anderen untereinander kommunizieren.

Neben der durchschnittlich gemessenen Zeit ist in Abbildung 4.14 zudem die Dauer des langsamsten und schnellsten Prozesses beim Versenden der Daten aufgetragen (schwarz markierte Bereiche auf den einzelnen Balken). Dabei ist erkennbar, dass die Abweichung der benötigten Zeit mit zunehmender Prozessoranzahl sinkt, was durch Lokalitätseffekte der Partitionen begründet werden kann.

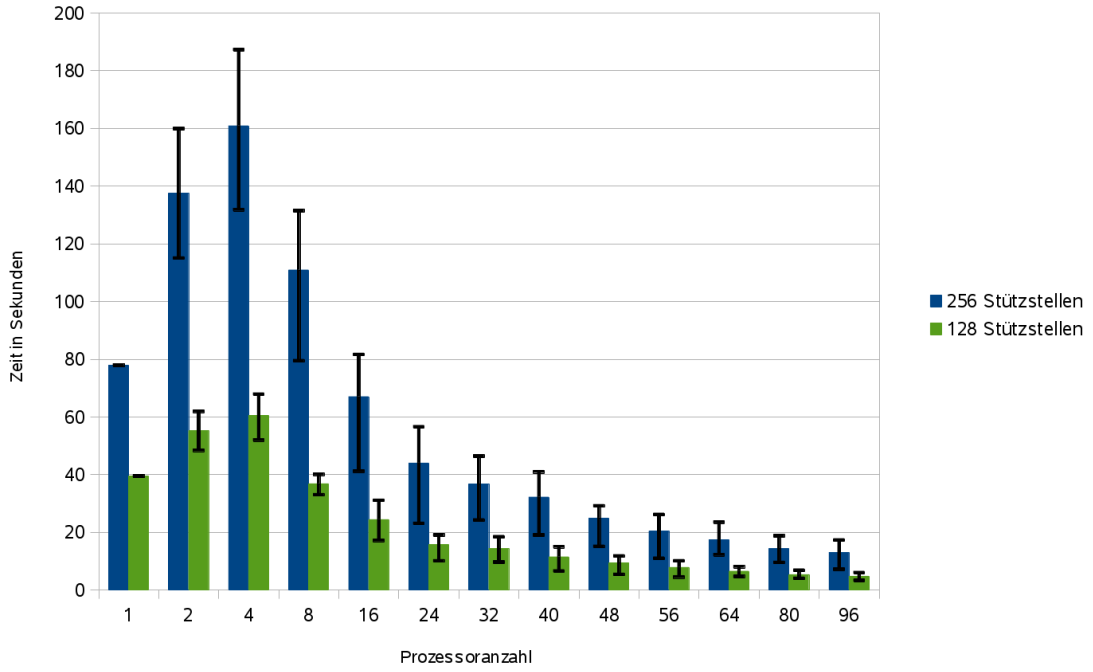


Abbildung 4.14: Zeitdauer zum Versenden und Empfangen der Koordinaten der Nächsten-Nachbarn für 2 unterschiedliche Stützstellenanzahlen

Abbildung 4.15 zeigt nun die Auswirkung der Netzgrößenvariation auf den Gesamtberechnungsaufwand bei 16, 32 und 64 parallelen Prozessen. Zu erkennen ist, dass bis zu einer Netzgröße von etwa 32000 Knoten der Berechnungsaufwand bei allen drei betrachteten Prozessoranzahlen nahezu identisch ist. Hierbei sind die resultierenden Kopplungsmatrizen derart klein, dass ihre eigentliche lokale Berechnung einen verhältnismäßig geringeren Teil im Vergleich zum Gesamtaufwand der parallelen Stützstellensuche einnimmt. Aspekte der parallelen Kommunikation dominieren hier den Gesamtberechnungsaufwand. Anhand der Messwerte lässt sich jedoch zudem erkennen, dass sich bei Verdopplung der Netzgröße auch die Zeit zur Berechnung der Interpolation verdoppelt.

Anhand dieser und der zuvor beschriebenen Ergebnisse kann somit von einer linearen Skalierung bei paralleler Ausführung der in dieser Arbeit implementierten Quaranta-Methode ausgegangen werden. Dies resultiert jedoch aus der Umsetzung der in Kapitel 3 beschriebenen Parallelisierungsansätze und nicht aus den Eigenschaften der Quaranta-Methodik.

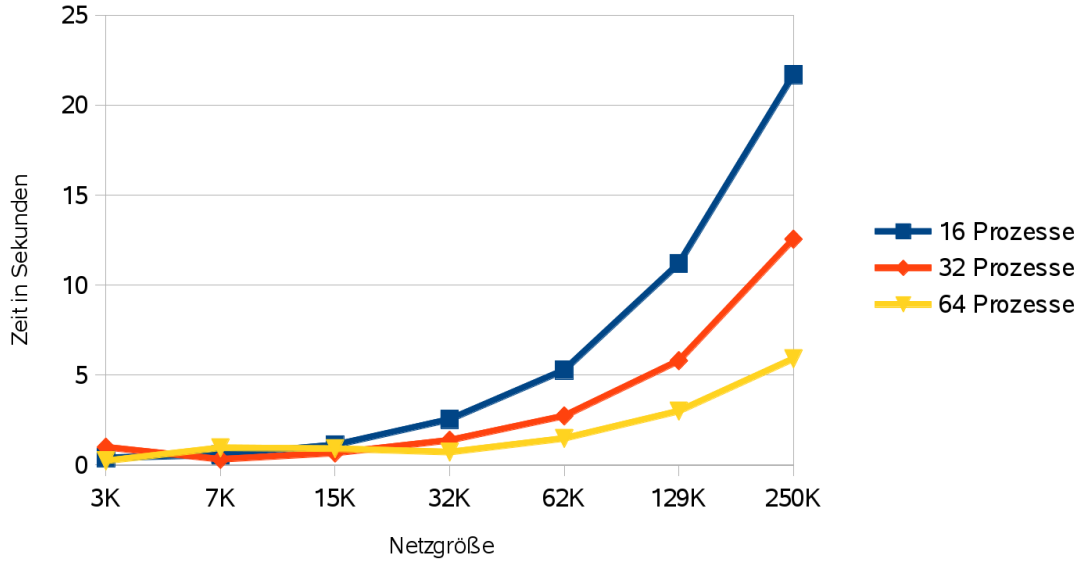


Abbildung 4.15: Berechnungsdauer für unterschiedliche Prozesszahlen bei Variation der Netzgrößen und 64 Stützstellen

Robustheitsstudien

Die zuvor beschriebenen Analysen bezogen sich zunächst auf ein einfaches Rechteckgitter, bei dem die Interpolationsgenauigkeit jedoch nicht betrachtet wurde. Trotz aller Performancesteigerungen, die evtl. erzielt werden können, ist im aeroelastischen Kontext die Interpolationsgenauigkeit der Lasten und Verschiebungen von übergeordneter Bedeutung. Entscheidendes Kriterium ist hierbei die Gewährleistung der lokalen Lastprojektion sowie die Glattheit der Verschiebungen des CFD-Oberflächennetzes seitens des Verformungstransfers. In der Regel kann jedoch nicht von einer analytischen Funktion der Lasten und Verschiebungen ausgegangen werden, sodass absolute bzw. relative Fehlerbetrachtungen bei aeroelastischen Problemen sehr eingeschränkt genutzt werden können. Im weiteren sollen daher keine speziellen Fehlermaße betrachtet werden. Genauere Fehleranalysen der sequentiellen Quaranta-Methode sind Teil von [5]. Im folgenden wird zur Einschätzung der Robustheit der parallelen Methode eine optische Überprüfung der Glattheit der entstehenden Oberflächennetze vorgenommen. Falls notwendig, wird der Leser für etwaige, punktuell auftretende Fehler sensibilisiert.

Im Folgenden wird zur Untersuchung der parallelen Methode die bereits beschriebene HiReTT-Flügelkonfiguration verwendet. Auf dem Strukturgitter des Flügels wurde ein für Fluid-Struktur-Wechselwirkungen übliches Deformationsfeld verwendet. Es wurde in einer statischen FEM-Verformungsrechnung ermittelt, bei der das HiReTT-Strukturmodell mit einer Einzelkraft an der Flügelspitze belastet wurde.

Zur besseren Einschätzung der numerischen Sensitivität u. Robustheit sind die berechneten Deformation in ihrer Amplitude verstärkt worden. Das genutzte Verformungsfeld ist in Abbildung 4.16 zu sehen.

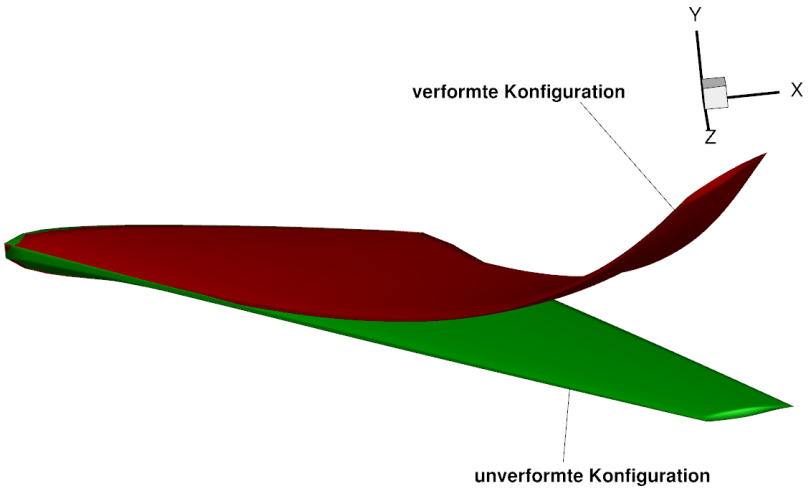


Abbildung 4.16: Verformungswerte des Strukturnetzes

Im weiteren wurde zunächst für das CFD-Oberflächennetz eine RCB-Partitionierung verwendet sowie eine Zoltan-basierte Partitionierung des CSM-Strukturnetzes. Abbildung 4.17 zeigt eine Verformungsinterpolation für 32 Partitionen. Für die Bestimmung der Koeffizienten der Kopplungsmatrix wurden 128 Stützstellen verwendet.

Anhand des Vergleichs der Verteilungen lässt sich zunächst eine glatte Interpolation der Verformungen des Strukturgitters erkennen. Die für CFD-Berechnungen wichtigen Bereiche wie z. B. die Hinterkante wurden korrekt anhand der Verformungen des Strukturnetzes „rekonstruiert“. In Abbildung 4.18 ist für die gewählte Partitionsanzahl zudem eine Schnittansicht aus verformter und unverformter CFD-Oberfläche in der Nähe des Spoilers dargestellt. Unterschiede zwischen dem CFD-Oberflächennetz und CSM-Strukturnetz infolge des statischen Spoilerausschlags beeinflussten dabei nicht die Güte der Verformungsinterpolation.

Im Weiteren wurde, analog zu Abbildung 4.17, bei einer Variation der Stützstellen-Anzahl kein Einfluss auf die Güte der Verformungsinterpolation festgestellt. Aufgrund der CSM-Partitionsgröße im Verhältnis zur Stützstellen-Anzahl werden im Rahmen der Nächste-CSM-Partition-Suche keine neuen Partitionen gefunden, die einen Einfluss auf die Genauigkeit der Stützstellensuche haben könnten. Ein Einfluss von einzelnen nicht-zusammenhängenden CSM-Partitionen konnte obendrein nicht identifiziert werden.

Bei einer Erhöhung der Partitionsanzahl auf 96, wobei statt 128 zunächst nur 64 Stützstellen verwendet wurden, konnte ebenfalls eine glatte Verformungsinterpola-

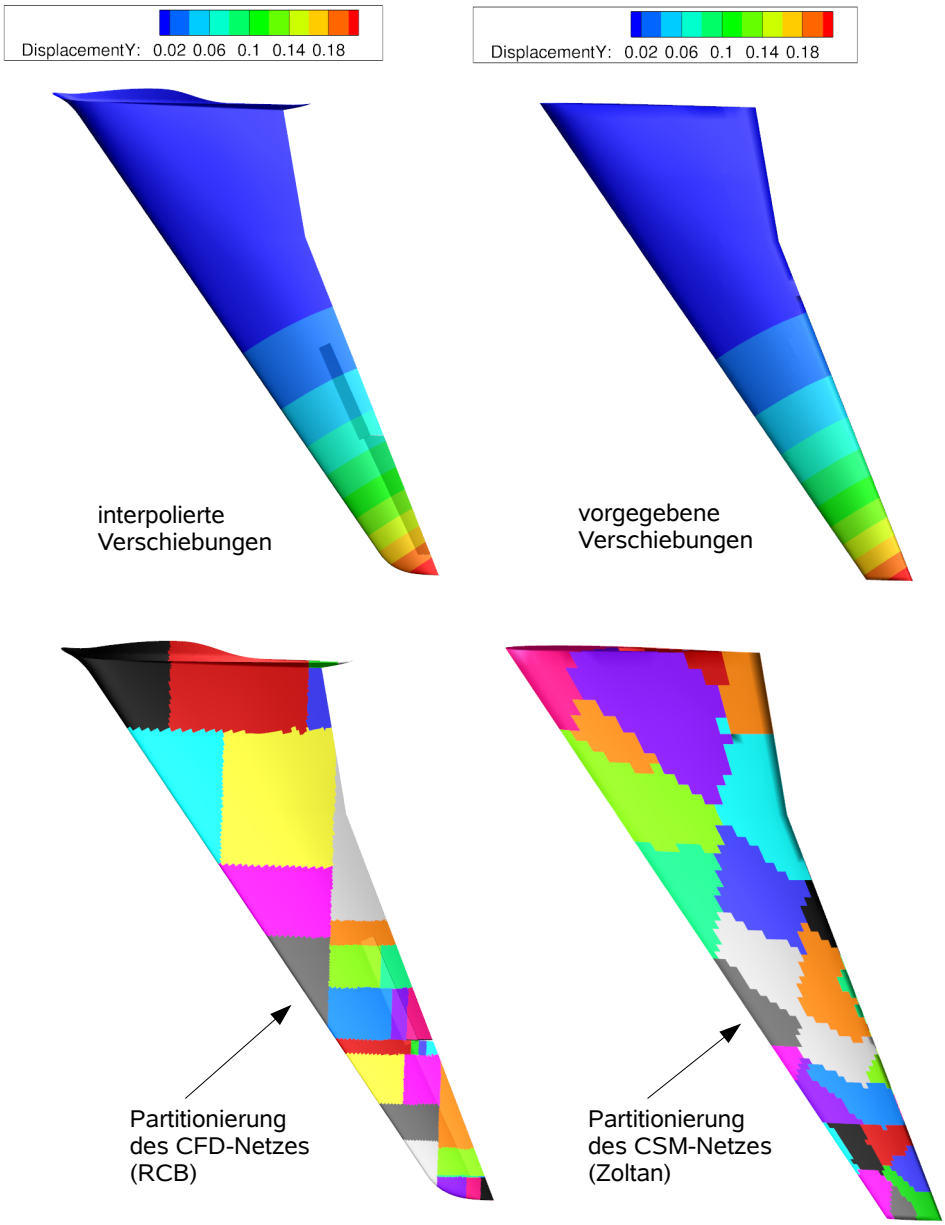


Abbildung 4.17: Verformungsinterpolation des HiReTT-Flügels bei 32 Prozessen und 128 Stützstellen

tion auf dem CFD-Oberflächennetz ermittelt werden. Dies bestätigt Abbildung 4.19.

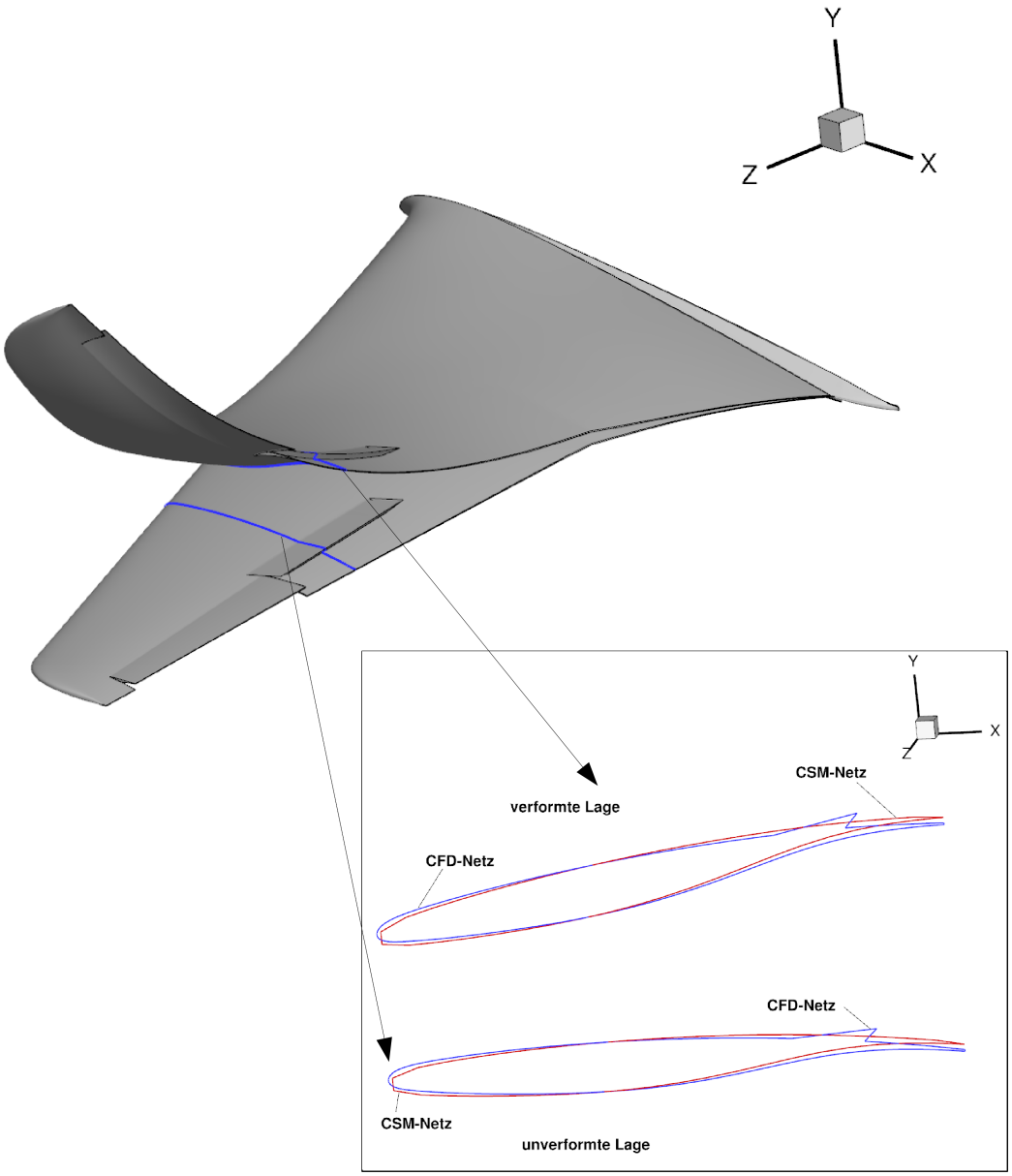


Abbildung 4.18: Schnittansicht des Flügels in Spoiler-Nähe im Vergleich zur unverformten Lage

Bei Verwendung des RCB-Partitionierungsalgorithmus und 32 bzw. 96 Partitionen für das CSM-Strukturnetz konnte gleichermaßen ein korrekter Verformungstransfer ermittelt werden. Hierbei findet darüber hinaus eine sehr vereinfachte parallele Kommunikation zwischen den Prozessen statt, da CFD-Oberflächen- und CSM-Strukturpunkte eines Prozesses nahe beieinanderliegen.

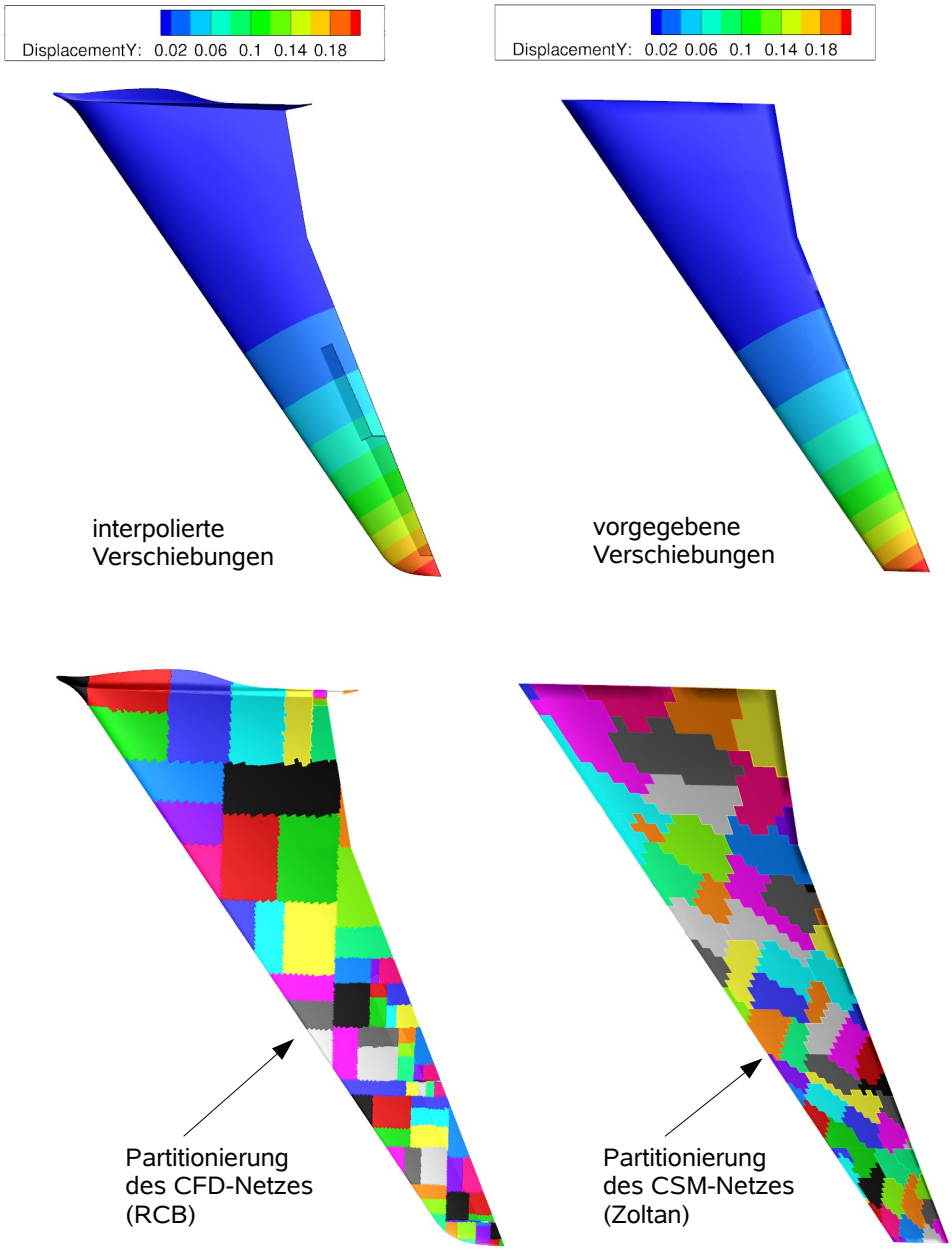


Abbildung 4.19: Verformungsinterpolation des HiReTT-Flügels bei 96 Prozessen und 64 Stützstellen

Einfluss der Partitionen auf die Genauigkeit der Stützstellensuche

Da im Rahmen der parallelen Ausführung eine exakte Stützstellensuche nur auf Basis der gefundenen CSM-Partitionen durchgeführt wurde, konnte zunächst nicht von den geometrisch exakten Nächsten-Nachbarknoten ausgegangen werden. Es wurden daher stichprobenweise Stützstellen einzelner CFD-Oberflächenpunkte, die in der parallelen Ausführung ermittelt wurden, mit denen aus einer sequentiellen Ausführung der Quaranta-Methode verglichen. Hierbei konnte im Bereich kleiner Partitionsanzahlen kein signifikanter Unterschied festgestellt werden und Stützstellen, ermittelt auf Basis der Bounding-Box basierten Suche, entsprachen den geometrisch exakten Stützstellen. Aufgrund der großen Überlappungen wurden alle geometrisch nächsten Partitionen ermittelt. Die Verwendung einer bestimmten Partitionierungsmethode (RCB oder Zoltan) hatte darauf zudem keinen Einfluss.

Demgegenüber konnte bei Steigerung der Partitionszahlen ein Unterschied der verwendeten Stützstellen verzeichnet werden. Signifikante Unterschiede traten hier zunächst entlang der Grenze der Flügeloberfläche zur CFD-Rumpfoberfläche auf. Wird bspw. auf Strukturnetzseite eine RCB-basierte Partitionierung vorgenommen und ein Suchradius auf Basis der Bounding-Box Begrenzungen der nächsten CSM-Partition, so verteilen sich die verwendeten Stützstellen entlang des Flügelwurzelschnitts (dem Übergang zum Rumpf). Für einzelne CFD-Punkte am Übergang zwischen Rumpf und Flügel wurde dabei eine nächste CSM-Partition gefunden, deren Punkte fast komplett in einer Ebene lagen. Die Stützstellensuche wurde daraufhin nur entlang des Übergangs Rumpf/Flügel durchgeführt. Einen Einfluss auf die Glattheit bzw. Güte der interpolierten Verschiebungen konnte aufgrund der geringen Verformungen an beschriebener Stelle aber nicht festgestellt werden. Dieser Effekt konnte jedoch gemindert werden, indem der Suchradius in allen Richtungen gleich groß festgelegt wurde. Als Maß wurde bspw. der durchschnittliche Abstand zur Bounding-Box Begrenzung der nächsten CSM-Partition gewählt.

Wird anstelle der RCB-Methode eine Zoltan-Partitionierung im CSM-Gitter bei großer Partitionsanzahl verwendet, so traten jedoch vereinzelt signifikante Interpolationsfehler auf. Eine fehlerhafte Rekonstruktion der CFD-Oberfläche ist dabei in Abbildung 4.20 zu sehen. Entscheidendes Problem hierbei sind einzelne nicht-zusammenhängende Partitionen, welche durch die in der Arbeit umgesetzte parallele Stützstellensuche zunächst nicht erkannt werden können. Die Problematik lässt sich jedoch innerhalb des parallelen Algorithmus durch Einbau eines Sicherheitsfaktor während der Partitionensuche beseitigen. Dieser bewirkt, dass solange nach Partitionen gesucht wird, bis ein Vielfaches der eigentlich geforderten Menge an Stützstellen von gefundenen Partitionen gewährleistet werden kann. Aus dieser vergrößerten Stützstellenmenge wird anschließend die Menge der relevanten Stützstellen identifiziert. Im Rahmen der Studien erwies sich die dreifache Menge als ausreichend und beseitigte die in Abbildung 4.20 sichtbaren Artefakte. Daraus resultierende Performanceverluste stellten sich als vernachlässigbar heraus.

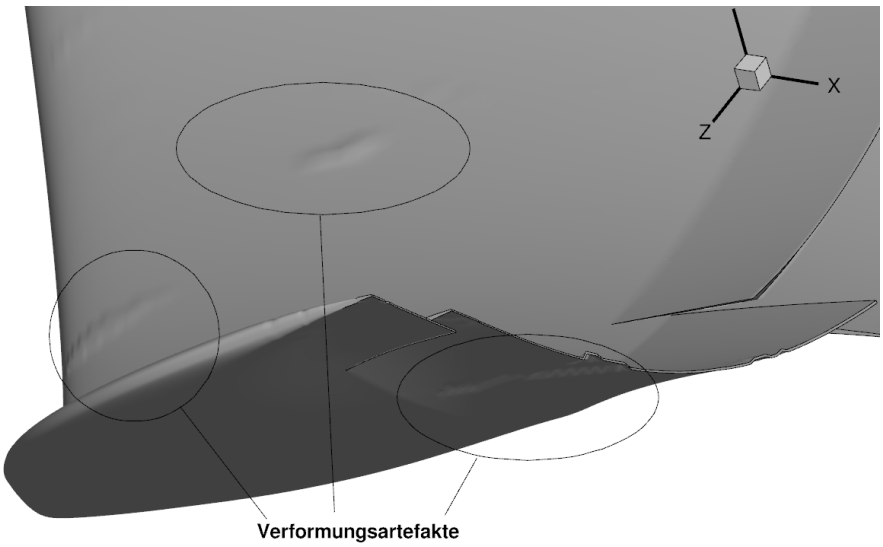


Abbildung 4.20: Verformungsartefakte entlang des Oberflächennetzes bei Zoltan-basierter Partitionierung des Strukturnetzes und 96 Partitionen sowie 128 Stützstellen

Im Vergleich zu den parallelen Untersuchungen des generischen Rechteckgitters ergaben sich für die HiReTT-Konfiguration folgende (ausgewählte) Performance-Ergebnisse und folgender durchschnittlicher Speicherbedarf der Kopplungsmatrix:[3]

Prozessoranzahl	gemessene Zeit	Speedup	Speicherbedarf von **H**
1	55s	1	191.4 MB
12	28.05s	1.96	16.4 MB
24	21.4s	2.57	8.5 MB
48	12.3s	4.46	4.48 MB

Tabelle 4.1: Parallele Performance-Ergebnisse der HiReTT-Konfiguration bei 128 Stützstellen

Auf Grundlage der Ergebnisse ist in Tabelle 4.1 ein im Vergleich zum Rechteckgitter geringerer Speedup bei Erhöhung der Prozessorzahl erkennbar. Aufgrund der direkten Abhängigkeit zwischen der Anzahl CFD-Oberflächenknoten und der Zeilenanzahl der Kopplungsmatrix nimmt der Speicheraufwand zudem auch linear bei Erhöhung der Prozessorzahl ab.

Neben Robustheits-Untersuchungen anhand der HiReTT-Konfiguration soll abschließend am Flügelkastenmodell von Kapitel 4.1.3 die korrekte Funktionalität der

[3]Der angegebene Speicherbedarf von H bezieht sich auf die speichertechnische Aufsummierung aller Teilkopplungsmatrizen eines Prozesses

parallelen Quaranta-Methode bei realen Flügelmodellen demonstriert werden. Hierfür wurde im Strukturnetz des Kastenmodells eine Verformung, ähnlich der HiReTT-Konfiguration, aufgetragen. Es wurden zur parallelen Ausführung 32 Prozesse und 128 Stützstellen verwendet. Das CFD-Netz wurde dabei mit dem RCB-Algorithmus, dass Strukturnetz mit dem graph-basierten Zoltan-Partitionierungsalgorithmus in Teilnetze zerlegt. Abbildung 4.21 zeigt nun eine glatte Rekonstruktion der Strukturnetz-Deformationen auf dem CFD-Oberflächennetz.

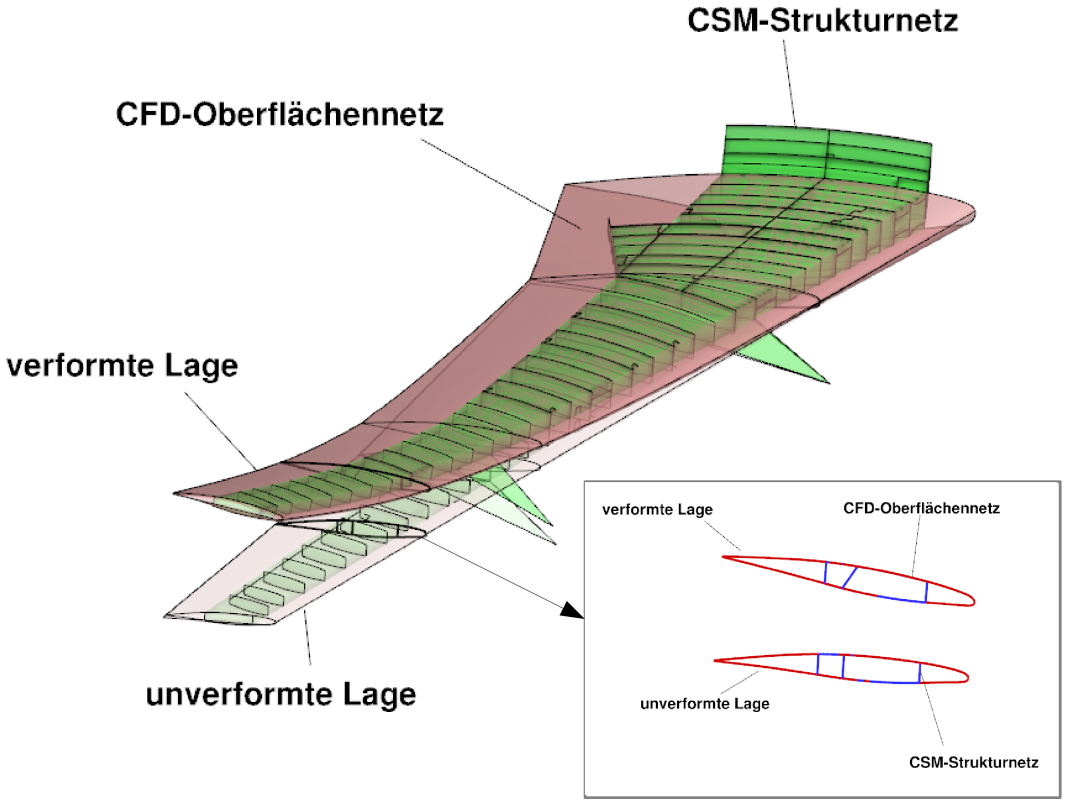


Abbildung 4.21: Verformungsinterpolation eines Flügelkasten-Modells im Vergleich zur unverformten Lage

Analyse von Mehrkomponenten-Konfigurationen

Dieser Abschnitt widmet sich der Analyse des Zusammenspiels aus implementierter Blending-Methodik (beschrieben in Kap. 2.4) und Bauteil-basierter Anwendung der Interpolationsmethode nach Quaranta. Als Testfall wird die DLR-F11-Hochauftriebskonfiguration verwendet. Es wird an dieser Stelle lediglich der Verformungstransfer untersucht. Das zu Testzwecken verwendete strukturseitige Verformungsfeld ist in Abb. 4.22 dargestellt.

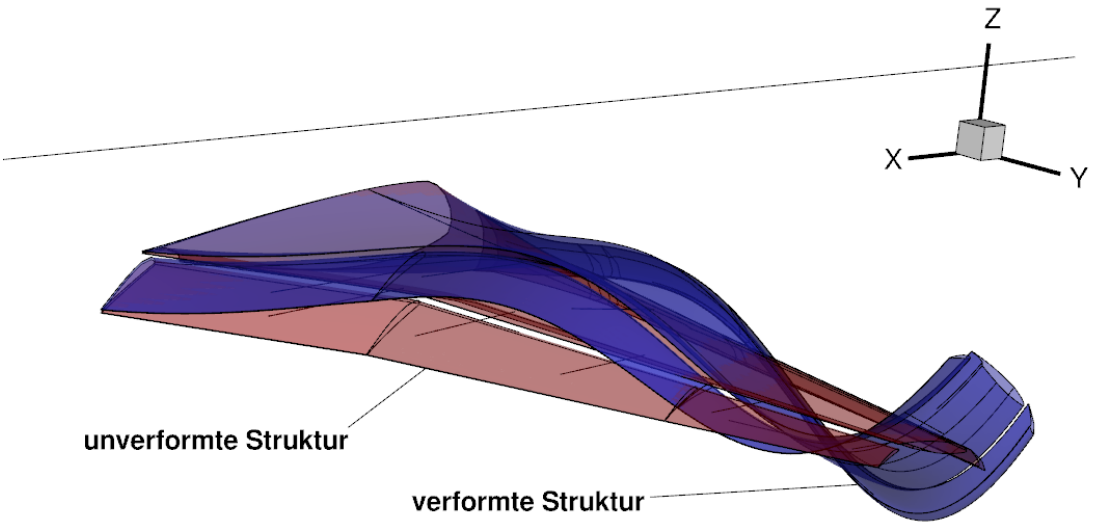
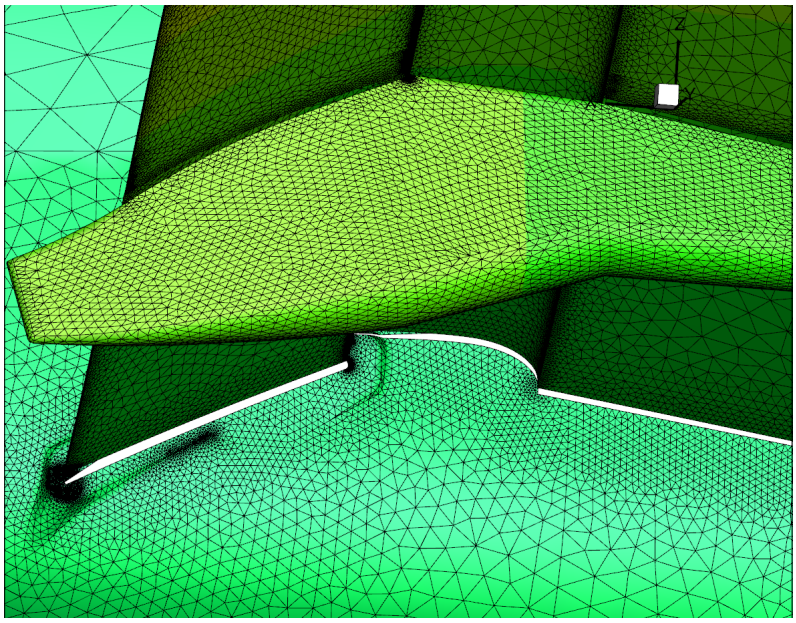


Abbildung 4.22: Aufgetragenes Verformungsfeld auf einzelnen Strukturkomponenten

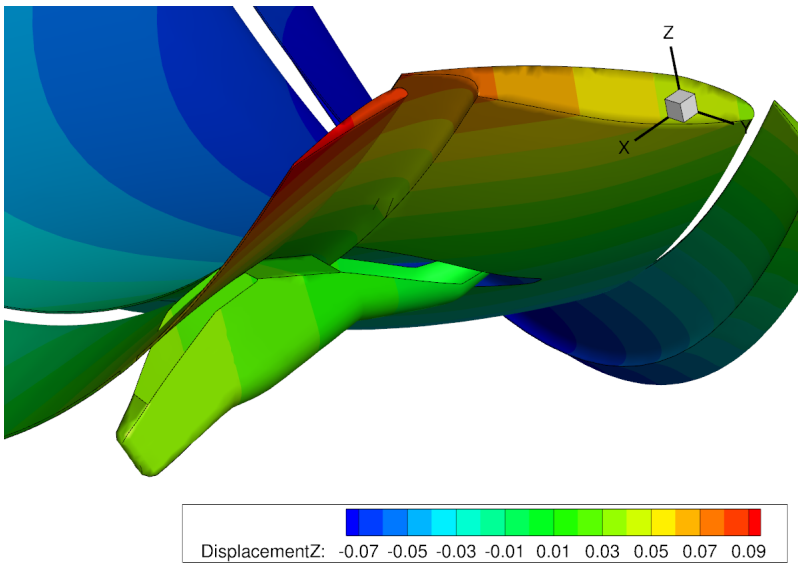
Als Paarungen wurden die bereits in Abbildung 4.6 und 4.7 farblich markierten Komponenten des Oberflächen- und Strukturnetzes definiert. Bei der Paarung der Flap-Track-Fairings wurden jedoch auf CSM-Seite nicht die gezeigten Balkenelemente verwendet, sondern die jeweiligen Strukturnetze des Flügels und der Landeklappe. Wie bereits in Kap. 4.1.4 angedeutet wurde, ist dieses Vorgehen damit begründet, dass zum einen bei der FIE-basierten Interpolation im FlowSimulator keine Rotationsmomente der Balken berücksichtigt werden können und zum anderen die derzeitige Umsetzung der Quaranta-Methode nicht in Verbindung mit Baugruppen zu nutzen ist, die alleinig aus Balkenelementen bestehen.

Bauteilbezogene Interpolation mit und ohne Blending

Auf Basis der definierten Paarungen wurde zunächst eine Blending-lose Verformungsinterpolation durchgeführt. Ausgewählte Ausschnitte des daraus resultierenden Ergebnisses sind in Abbildung 4.23 dargestellt.



(a) Klaffung am Übergang Rumpf/Flügel



(b) Klaffung entlang der Verkleidungen

Abbildung 4.23: Ungenaue Verformungsinterpolation bei Blending-loser Interpolation

Anhand der ausgewählten Details sind die für Blending-lose Interpolation charakteristischen Klaffungen an der Nahtstelle einzelner Baugruppen zu erkennen. Im betrachteten Modellbeispiel sind dies die Übergänge zwischen dem Rumpf und Flügel (siehe Abbildung 4.23(a)) und entlang der Fairinggrenzen zu den Landeklappen und dem Flügel (siehe Abbildung 4.23(b)). Aufgrund der größeren Verformung bei letzteren ist die Klaffung auch stärker ausgeprägt als am Übergang zwischen Rumpf und Flügel.

Für eine Blending-basierte Interpolation, die einen sauberen Verformungstransfer gewährleistet, wurde nun zunächst ein maximaler Abschnitt zur Schnittkurve zwischen angrenzenden Komponenten gewählt (Faktor a_{max} in Kapitel 2.4), in dem Einflüsse indirekt am Interpolationsprozess beteiligter Komponenten berücksichtigt werden. Abbildung 4.24 zeigt exemplarisch eine sich dafür ergebende Verteilung der Blending-Faktoren. Rot markiert sind Bereiche, in denen nur der direkte Verformungseinfluss einer Paarung wirkt und indirekte Einflüsse durch die Gewichtungsfunktion $w(x)$ keinen Anteil an der Gesamtverformung der jeweiligen Komponente erhalten. Der Faktor a_{max} ist dabei für alle Paarungen gleich groß gewählt worden. Die aus den indirekten Paarungen ergebenden Blending-Bereichen, in denen indirekte Verformungseinflüsse wirken, sind für FlapTrack-Fairings in Abbildung 4.25(a) und für die Flügelkomponente in Abbildung 4.25(b) exemplarisch dargestellt. Im Bereich der FlapTrack-Fairings findet dabei ein lückenloser Übergang zwischen dem aus der indirekten Zuordnung der Landeklappe resultierenden Blending-Radius und dem Blending-Radius, resultierend aus der indirekten Flügelzuordnung, statt.

In Abbildung 4.26 ist das Ergebnis einer Verformungsinterpolation der Konfiguration zu sehen, wie sie sich unter Anwendung des Blendings ergibt. Anteile aus direkter und indirekter Verformungsinterpolation, die aus den Blending-Radius-Definitionen resultieren, sind für die z-Richtung in Abbildung 4.27(a) und 4.27(b) dargestellt. Es wurde im betrachteten DLR-F11-Modell bis auf die Verformungsinterpolation der Rumpf-Paarung, welche mit einer FIE-basierten Methode im FlowSimulator gerechnet wurde (die Strukturverformung am Rumpf wurde mit Null vorgeschrieben), für alle anderen Paarungen der implementierte Quaranta-Ansatz verwendet. Es mussten jedoch mindestens 80 Stützstellen benutzt werden. Bei Verwendung einer geringeren Stützstellenanzahl (z. B. 64 Stützpunkte) führte dies zu einem Abbruch der Rechnung. Das Problem tritt bei der Berechnung des indirekten Verformungseinflusses für den Rumpf auf, welcher von der Flügelstruktur verursacht wird. Der direkte Verformungseinfluss der Rumpfstruktur auf die Rumpf-CFD-Oberfläche wird zwar mit dem FIE-Verfahren berechnet. Bei der Berechnung des indirekten Verformungseinflusses des Flügels auf den Rumpf kommt allerdings die Quaranta-Methode zum Einsatz. Denn diese ist für die Flügelbaugruppe als Primär-Interpolationsverfahren selektiert worden. Die nächsten Nachbarpunkte im Flügelstrukturnetz, die im Rahmen der Quaranta-Methode benötigt werden, liegen für Rumpf-CFD-Oberflächenpunkte komplett in einer Ebene (in der Ebene des Flügelwurzelschnitts). Dies führte im Rahmen der Berechnung der Interpo-

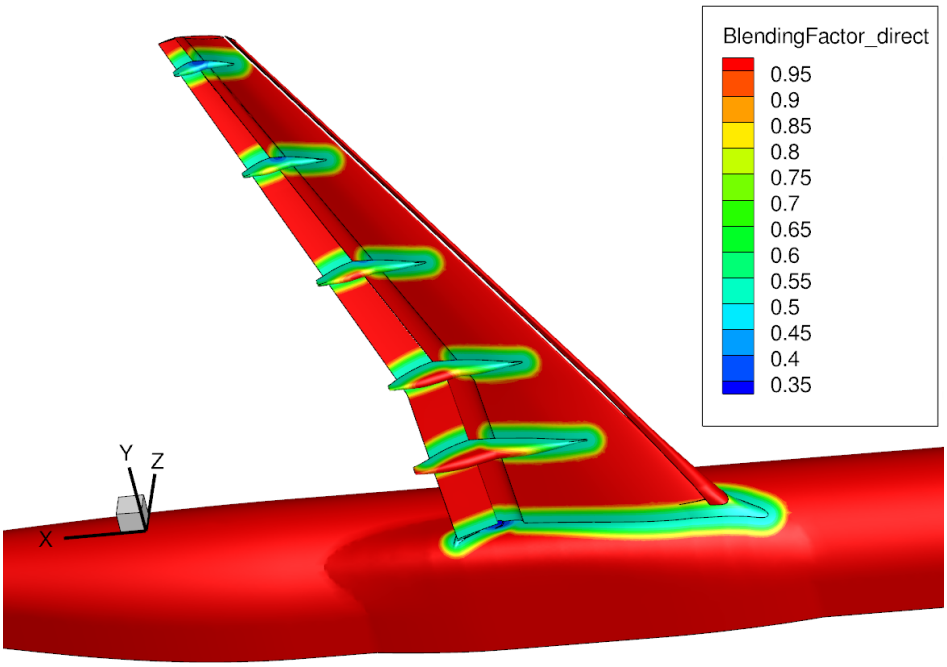
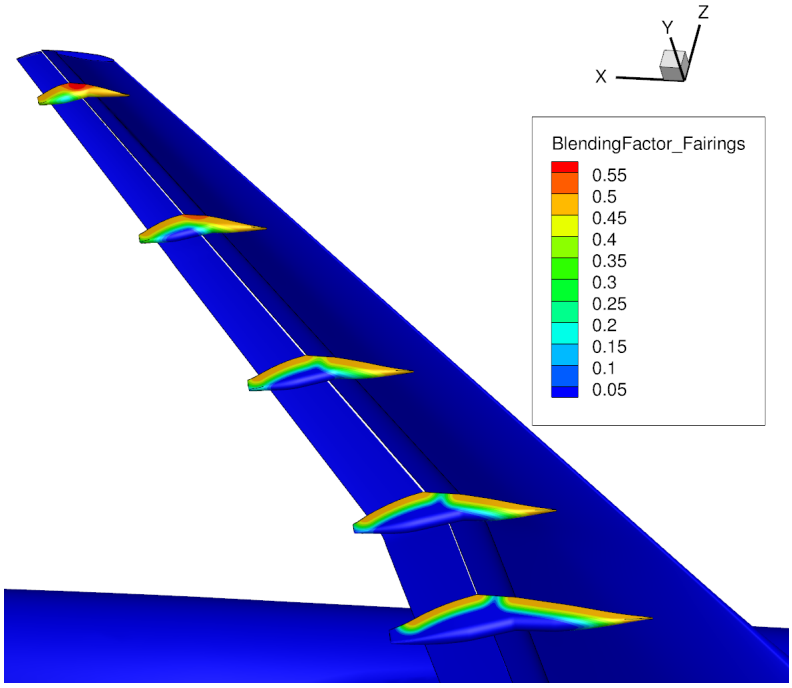


Abbildung 4.24: Blending-Faktoren der direkten Baugruppenzuordnungen

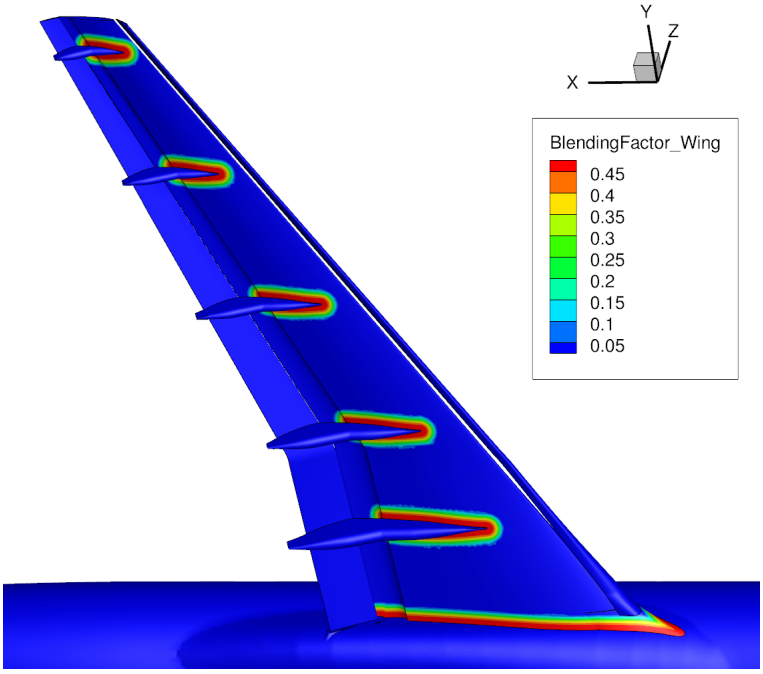
lationskoeffizienten zu einer Singularität bei der Gleichungssystemlösung. Die Erhöhung der Stützpunktanzahl behebt dieses Problem, da infolgedessen auch Flügelstrukturnetzpunkte außerhalb des Flügelwurzelschnitts gefunden werden.

Beim Verformungstransfer der CFD-Oberfläche der Fairings musste im Vergleich dazu die Stützstellenanzahl stark heraufgesetzt werden, um den glatten Verformungsverlauf insgesamt zu erreichen. Es wurden im gerechneten Beispiel 500 Stützstellen verwendet, um punktuell aufgetretene Konditionierungsprobleme zu beseitigen, welche bei einzelnen Fairing-Oberflächenpunkten im Rahmen der Koeffizientenbestimmung verzeichnet wurden. Diese äußerten sich in einem punktuell auftretendem Beulen-Verhalten.

Trotz des starken Verformungsfeldes konnte insgesamt im Rahmen der Analysen mithilfe der Quaranta-Methodik ein zusammenhängendes CFD-Oberflächennetz erzielt werden, die zudem eine glatte Oberfläche der Beispielkonfiguration erzeugte. Die Methode eignet sich somit nicht nur zur parallelen Berechnung einfacher Modelle, sondern auch im Kontext komplexer Konfigurationen.



(a) Blending-Faktoren indirekter Baugruppenzuordnungen der Flap-Track-Fairings



(b) Blending-Faktoren indirekter Baugruppenzuordnungen des Flügels

Abbildung 4.25: Ausgewählte Blending-Radien, resultierend aus indirekter Baugruppenzuordnung

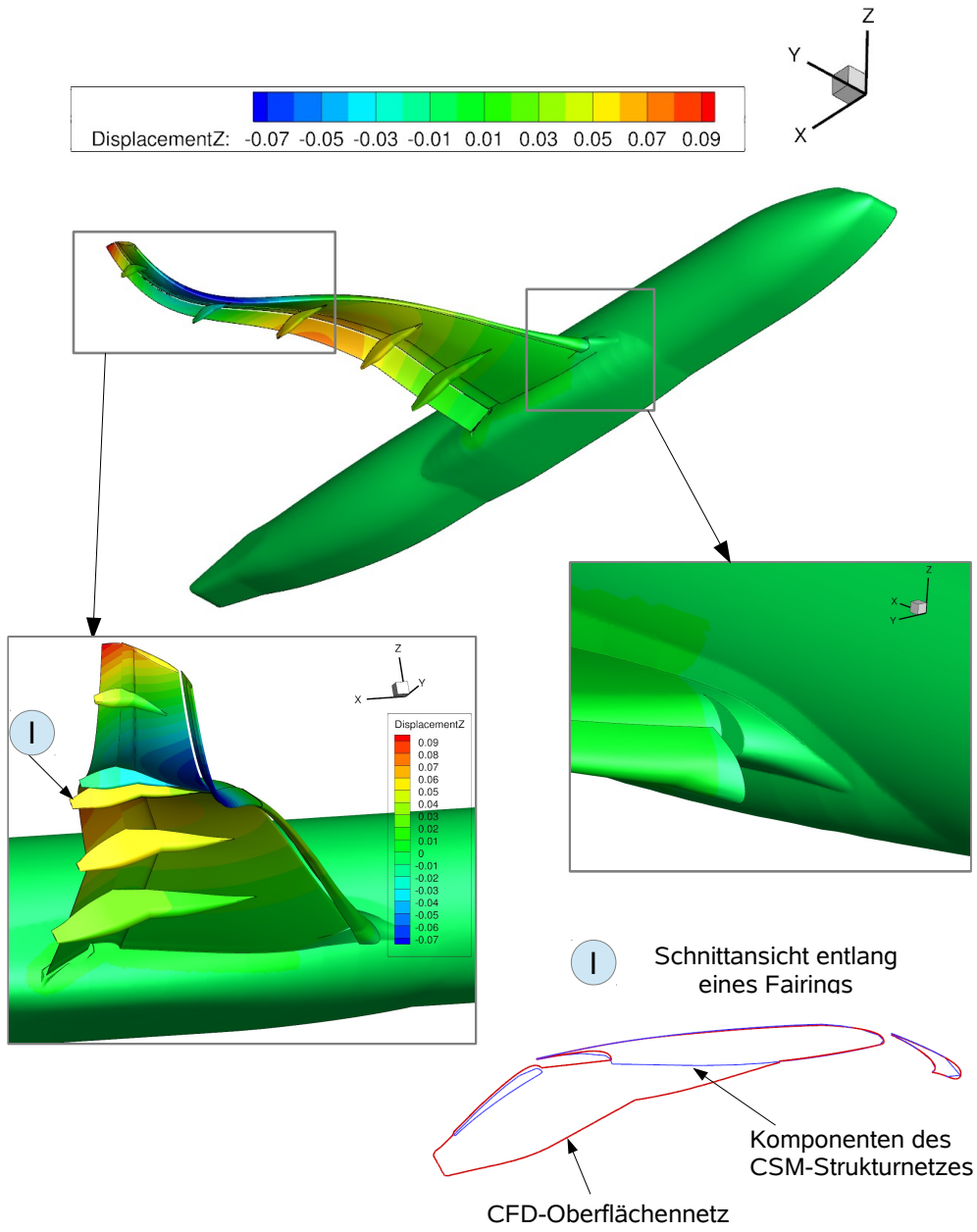
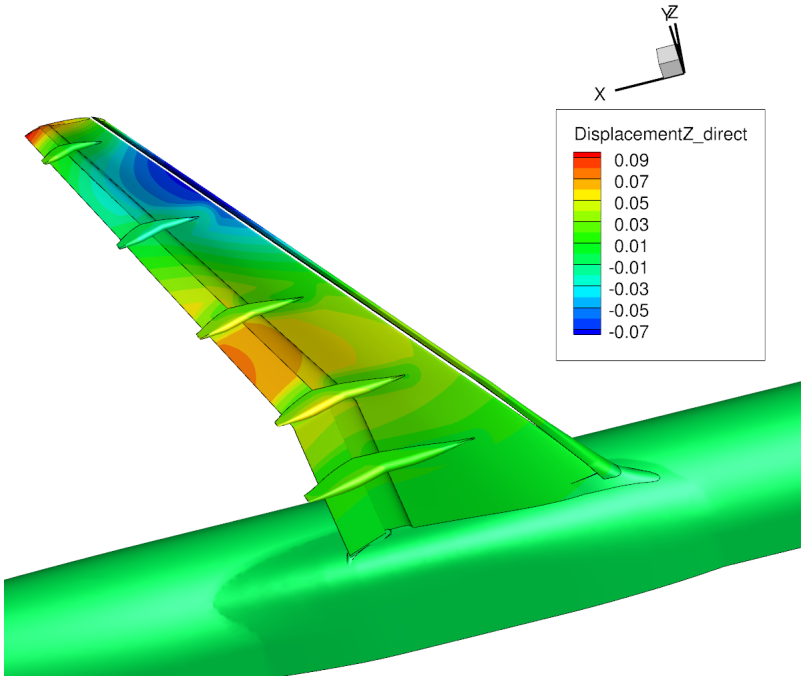
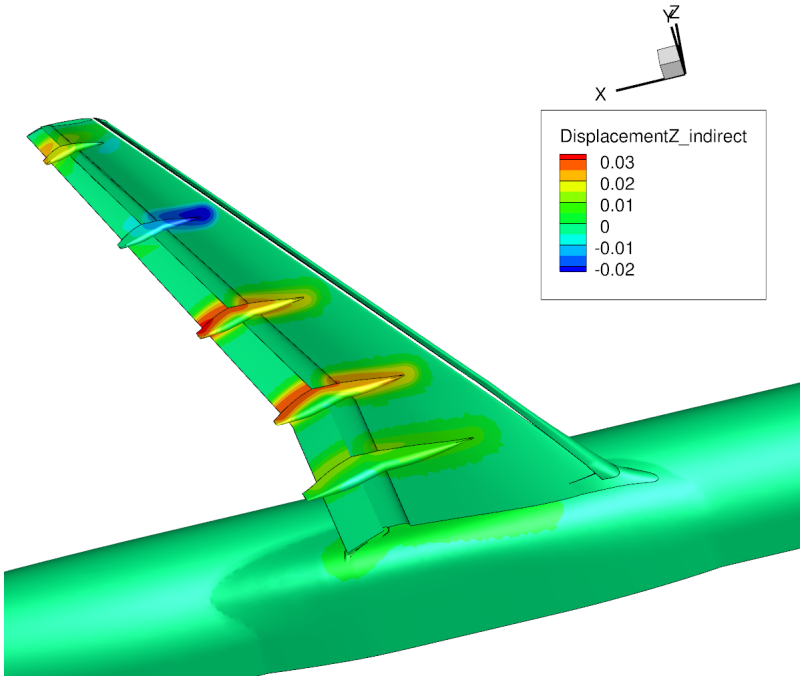


Abbildung 4.26: Ergebnis der Bauteil-basierten Verformungsinterpolation auf dem CFD-Oberflächennetz



(a) Verformungsgrößen direkter Paarungen



(b) Verschiebungsanteile aus der Berechnung indirekter Paarungen

Abbildung 4.27: Gegenüberstellung von direkten und indirekten Verformungsanteilen in z-Richtung

5 Abschließende Betrachtungen

Im Folgenden soll abschließend eine Bilanz zu den erläuterten und implementierten Methoden der Arbeit gezogen werden.

5.1 Zusammenfassung

In der vorangegangenen Arbeit wurde eine netzfreie Methode nach Quaranta et. al. zur Last- bzw. Verformungsinterpolation im Rahmen der Fluid-Struktur-Interaktion vorgestellt und im FlowSimulator implementiert. Im Vergleich zu artverwandten Methoden wie der Wendland-Methode (siehe Kapitel 2.2.2) bietet sie wichtige Vorteile hinsichtlich der Lokalität des Last-/Verformungstransfers sowie algorithmischen Komplexität. Motiviert wird die Umsetzung des Ansatzes innerhalb der Arbeit durch einen physikalisch nicht-konservativen Datentransfer bisheriger Methoden innerhalb einer FlowSimulator-basierten Fluid-Struktur-Kopplungsrechnung. In dieser werden für den Lasttransfer und den Verformungstransfer unterschiedliche Verfahren verwendet, die jeweils den spezifischen Beschränkungen unterlagen. Aufgrund der Verwendung einer allgemeinen Kopplungsmatrix zum Austausch von Last- **und** Verformungsgrößen zwischen CFD-Oberflächen- und CSM-Strukturnetzen kann die Methode demgegenüber einen für aeroelastische Fragestellungen wichtigen energiekonservativen (falls nur translatorische Freiheitsgrade übertragen werden) Datentransfer gewährleisten. Durch eine gezielte Wahl der Stützstellenanzahl zeichnet sie sich darüber hinaus durch die beliebige Steuerung der Lokalität aus.

Für eine korrekte Interpolation bei Flugzeugkonfigurationen wurde der Quaranta-Ansatz zudem auf mehrere Baugruppen erweitert. Dabei wurde eine sogenannte „Blending“-Methodik eingeführt, die die für aerodynamische Berechnungen wichtige Forderung nach Geschlossenheit des CFD-Oberflächennetzes zwischen den einzelnen Baugruppen gewährleistet. Zur Verifikation der Methode wurde eine aus mehreren Baugruppen bestehende realistische Flugzeugkonfiguration (DLR-F11-Modell) gewählt. Das ausgewählte Rechenbeispiel konnte dabei die korrekte Funktion der Quaranta-basierten Methode im Kontext mehrerer Baugruppen demonstrieren.

Aufgrund der steigenden Feinheit heutiger Netzgeometrien im Bereich der Fluid-Struktur-Interaktion wurden in der Arbeit zudem Parallelisierungsansätze der Quaranta-Methode erläutert und umgesetzt. Hauptproblem dieser Umsetzung war die vorhandene Gebietszerlegung der diskretisierten CFD- und CSM-Netze, die

eine Bestimmung der Stützstellen, welche für die Quaranta-Methode benötigt wird, erschwerte. Zur Lösung des Problems wurde ein Algorithmus auf Basis einer Bounding-Box-basierten Beschreibung der zerlegten CSM-Strukturnetze sowie Einsatz effizienter, räumlicher Datenstrukturen entwickelt. Mithilfe einer Reihe von Testfällen (generisches Rechteckgitter, HiReTT-Konfiguration, generischer Flügel in Kastenbauweise) konnte die Unabhängigkeit des entwickelten Ansatzes von der Partitionierungsmethode nachgewiesen werden. Im Rahmen paralleler Analysen konnte zudem eine lineare Skalierbarkeit des parallelen Algorithmus festgestellt werden. Aufgrund der ermittelten Effizienz eignet sich der parallelisierte Ansatz somit zur Verwendung in komplexen Fluid-Struktur-Kopplungsrechnungen, die sich durch sehr starke Parallelität auszeichnen. Weiterer, entscheidender Gewinn der umgesetzten Parallelisierung ist eine mögliche Integration weiterer, netzfreier Interpolationsmethoden, die ebenso auf der Verwendung verteilter Stützstellen basieren.

Die anhand Kapitel 1.3 und 1.2 aufgezeigte Lücke einer bisherigen Fluid-Struktur-Kopplung im FlowSimulator kann nun mithilfe der entwickelten, parallelen Quaranta-Methode geschlossen werden. Die Erkenntnisse der Arbeit tragen somit insgesamt entscheidend zur Verbesserung moderner aeroelastischer Berechnungsprozesse, wie denen innerhalb des FlowSimulator, in der Flugzeugentwicklung bei.

5.2 Ausblick

Trotz der guten Skalierungswerte des parallelen Ansatzes ist eine weitere Geschwindigkeitsverbesserung möglich. Da die Ausführung des parallelen Algorithmus hauptsächlich von der Anzahl der CFD-Oberflächenpunkte einer Partition abhängt, besteht eine Möglichkeit der weiteren Performance-Steigerung darin, die Anzahl der Nachbarschaftssuchen zu reduzieren. Dabei kann bspw. nur eine Teilmenge der CFD-Oberflächenpunkte ausgewählt werden und zur verteilten Suche der Stützstellen betrachtet werden. CFD-Oberflächenpunkte einer Partition liegen i. d. R. nahe beieinander, so dass strukturseitige CSM-Stützstellen von Oberflächenpunkten gemeinsam geteilt werden. Aufwendige Kommunikationen zwischen einzelnen Prozessen innerhalb der Stützstellen-Suche sowie die Verarbeitung der gesendeten Stützstellen ließen sich somit signifikant reduzieren. Das Kernproblem dieser Methodik liegt jedoch in der passenden Auswahl von CFD-Punkten zur Nächsten-Nachbarn-Suche. Die Problematik der nicht-zusammenhängenden Partitionen tritt hierbei zudem nicht nur auf Strukturnetz-Seite, sondern somit auch seitens des CFD-Oberflächennetzes auf.

Neben dem umgesetzten Ansatz zur Problematik nicht-zusammenhängender Partitionen lassen sich alternative Ansätze verwenden. Im Rahmen der Bounding-Box-basierten Verarbeitung der CSM-Partitionen könnten einzelne Untermengen nicht-zusammenhängender Partitionen mithilfe von Graphen-Algorithmen erfasst werden. Dazu sind jedoch Konnektivitätsinformationen der Punkte erforderlich,

um entsprechende minimale aufspannende Teilbäume der Partitionen zu berechnen. Für alle dabei ermittelten, nun zusammenhängenden Punktmengen ließe sich zudem die gewählte Bounding-Box Methodik anwenden. Im Vergleich zum umgesetzten parallelen Algorithmus müsste statt eines k-d-Baums zur Punktspeicherung nun für jede Untermenge ein separater Baum erzeugt werden. Die daraus resultierenden Anpassungen betreffen somit nicht das Gesamtkonzept des parallelen Algorithmus.

Obwohl der in der Arbeit betrachtete Ansatz für einen konservativen Last- und Verformungstransfer wesentliche Vorteile gegenüber anderen Ansätzen in sich birgt (siehe Kap. 2), unterliegt er in der bis dato in der Arbeit umgesetzten Form jedoch gewissen Einschränkungen. In der Arbeit wurden stets von Strukturnetzen ausgegangen, die primär aus Volumen- oder Membranelementen bestehen. Sie besitzen an den Knotenpunkten jeweils nur translatorische Freiheitsgrade. Neben den genannten Elementtypen existieren jedoch im Bereich der FEM-basierten Verformungsrechnung weitere Elementtypen, wie z.B. Schalen- oder Balkenelemente, die an den Knotenpunkten auch über Rotationsfreiheitsgrade verfügen. Die Übertragung von strukturseitig bestehenden Rotationsfreiheitsgrade kann für den gelungenen Verformungstransfer wesentlich sein, insbesondere bei Strukturen, die als Balken idealisiert sind. Auf letztere ist der implementierte Algorithmus derzeit gar nicht anwendbar. Ziel sollte es also sein, den mit Abschluss dieser Arbeit bestehenden Algorithmus so zu erweitern, dass er zum einen bestehende Rotationsfreiheitsgraden beim Last-/Verformungstransfer berücksichtigt und zum zweiten auch auf Baugruppen anwendbar ist, die vollständig aus Balkenelementen aufgebaut sind. Mögliche netzbasierte Ansätze, wie dies erreicht werden kann, werden in [1] und [5] vorgestellt. Im Rahmen der in Kapitel 2.2.2 vorgestellten Wendland-Methode wurde zudem in [19] ein netzfreier Ansatz vorgestellt, für den jedoch derzeit keine Aussage über die Adaptierbarkeit im Rahmen der Quaranta-Methode getroffen werden kann.

6 Anhang

Der in Kapitel 3.3 beschriebene parallele Algorithmus verwendet einen k-d-Baum zur Suche nach nächsten CSM-Partitionen. Aufgrund seiner allgemeinen Formulierung für k-Dimensionen lässt sich dabei zeigen, dass ein k-d-Baum neben der Speicherung von Punktkoordinaten auch zur effizienten Bereichssuche von Bounding-Boxen genutzt werden kann. Im Vergleich zur Bereichssuche bei dreidimensionalen Punkten sind allerdings kleine Anpassungen nötig. Diese für einen entsprechenden „Intersection Test“ von Bounding-Boxen innerhalb einer Bereichssuche notwendigen Anpassungen werden deshalb hier kurz vorgestellt.

Gegeben seien dabei zwei Bounding-Boxen $bb1$ und $bb2$ (siehe Abbildung 6.1) mit den jeweiligen oberen und unteren Grenzen $(x_{bb1,min}, x_{bb1,max})$ und $(x_{bb2,min}, x_{bb2,max})$. $bb2$ sei ferner die durch die Bereichssuche spezifizierte Suchbox. Für eine Überschneidung im R^N beider Boxen müssen nun die Bedingungen

$$\begin{aligned} x^i_{bb1,min} \leq & x^i_{bb2,max} \\ x^i_{bb1,max} \leq & x^i_{bb2,min} \qquad \text{für} \quad i = 1, ..., N \end{aligned} \tag{6.1}$$

erfüllt werden. Somit muss zudem gelten

$$\begin{aligned} -\infty \leq x^1_{bb1,min} &\leq x^1_{bb2,max} \\ &\vdots \\ -\infty \leq x^N_{bb1,min} &\leq x^N_{bb2,max} \\ x^1_{bb2,min} \leq x^1_{bb1,max} &\leq \infty \\ &\vdots \\ x^N_{bb2,min} \leq x^N_{bb1,max} &\leq \infty \end{aligned} \tag{6.2}$$

Mithilfe dieser Definitionen lässt sich nun das Objekt $bb1$ im R^N mit den Koordinatenbegrenzungen als Punkt im R^{2N} mit den Koordinaten x^i_k für $i = 1, ..., 2N$ betrachten:

$$x_{bb1} = [x^1_{bb1,min}, ..., x^N_{bb1,min}, x^1_{bb1,max}, ..., x^N_{bb1,max}]^T \tag{6.3}$$

Damit lassen sich 6.2 und 6.3 als

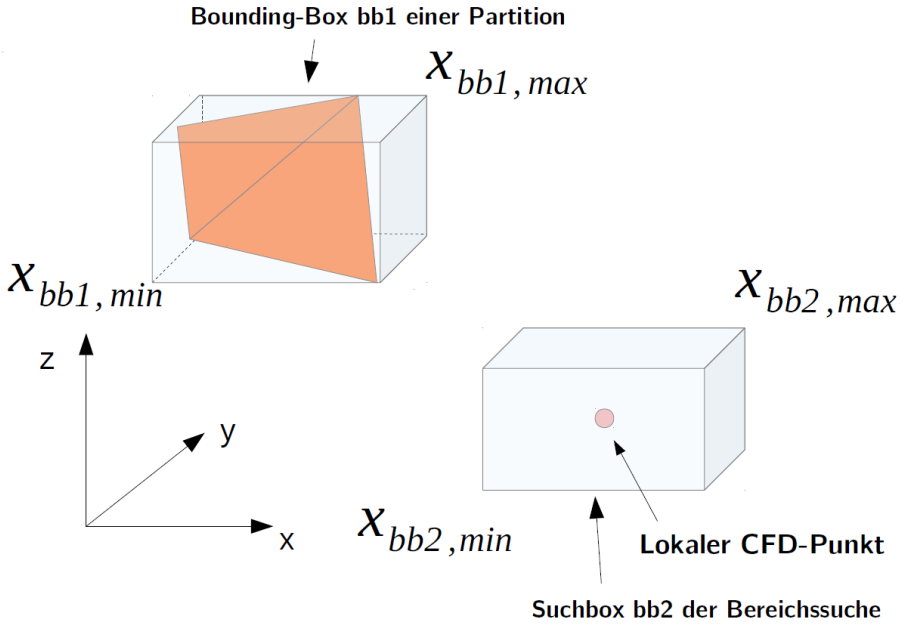


Abbildung 6.1: Bounding-Box einer Netzmenge und Suchbox der Bereichssuche

$$lbound^i \leq x^i_{bb1} \leq upbound^i \quad i = 1, 2, ..., 2N \tag{6.4}$$

beschreiben, wobei

$$\begin{aligned} lbound &= [-\infty, ..., -\infty, x^1_{bb2,max}, ..., x^N_{bb2,max}]^T \\ upbound &= [x^1_{bb2,min}, ..., x^N_{bb2,min}, \infty, ..., \infty]^T \end{aligned} \tag{6.5}$$

lbound und *upbound* beschreiben somit ein Hyper-Rechteck im R^{2N}, in dem Objekte im R^N mit gegebenen Koordinatenbegrenzung liegen müssen, um von der Bereichssuche ermittelt zu werden.

Abbildungsverzeichnis

1.1 Darstellung der Komponenten des FlowSimulator-Frameworks . . 9

1.2 Darstellung einer Prozesskette zur Fluid-Struktur-Kopplungsberechnung auf Basis des FlowSimulator-Frameworks 10

2.1 Gegenüberstellung eines CFD-Oberflächennetzes (schwarz) und dem dazugehörigen Strukturmodell (rot) 15

2.2 Projektion eines CFD-Punkts M in die natürlichen Koordinaten des finiten Elements [7] 18

2.3 Bauteil-bezogene Zuordnung eines CFD-Oberflächenpunkts zu mehreren Komponenten 26

2.4 Beispiel einer Blending-Gewichtungsfunktion $w(x)$ 27

2.5 Blending-Bereich zwischen Rumpf und Flügel 27

2.6 (a) Gebietszerlegung eines Tetraedernetzes und (b) Generierung eines Ghost-Points[10] 29

2.7 Grafische Gegenüberstellung unterschiedlicher Partitionierungen des HiReTT-Flügels 31

2.8 Darstellung nicht-zusammenhänger Partitionen 32

2.9 k-d-Baum Generierung eines einfachen 2D-Datensatzes. Hierbei dienen zugleich die einzelnen Punkte als Hyperebene bzw. Trenngeraden 33

3.1 Algorithmus zur räumlichen Kopplung 36

3.2 Gegenüberstellung ausgewählter direkter und indirekter Paarungen bei Bauteil-bezogener Kopplung einer einfachen Flugzeugkonfiguration 38

3.3 Schematische Darstellung des umgesetzten, parallelen Algorithmus 42

3.4 Bounding-Boxen einzelner CSM-Partitionen um Umkreis eines CFD-Oberflächenpunkts (rotmarkierte Raute) 45

3.5 Unterschiedliche Suchradien der Bereichssuche: willkürlicher Radius (grau), Entfernung zur nächsten Ecke der Bounding-Box (blau), symmetrischer Radius auf Basis des Abstands zur nächsten Ecke (rot) 48

3.6 Vergleich von approximativer und exakter CSM-Stützstellen für einen gegebenen CFD-Punkt (blau markierte Raute) 50

3.7 Mahalanobis-Abstand einzelner CSM-Strukturknoten innerhalb der CSM-Partitionen zum Zentrum der Partition. 52
3.8 Fehlerhafte Verformungsinterpolation eines CFD-Oberflächenpunkts (rote Raute) auf Basis der approximativen Suche der nächsten Nachbarn (als weiße Kreise gekennzeichnet) 53
3.9 Besetzungsstruktur einer 2360x2500 Kopplungsmatrix mit 32 nächsten Nachbarn je CFD-Punkt 57
3.10 Verteiltes Speicherungsschema der Kopplungsmatrix H für eine lokale Knotennummerierung 59
3.11 Grafische Darstellung des Anfrageprozesses 62
3.12 Schematische Darstellung der Puffer-Daten 62

4.1 Generisches Rechteckgitter aus ∼32000 CFD- und CSM-Gitterpunkten 67
4.2 CFD-Oberflächennetz des HiReTT-Flügels 68
4.3 CSM-Strukturnetz des HiReTT-Flügels 69
4.4 CFD-Oberflächennetz des Flügelkasten-Modells 70
4.5 CSM-Strukturnetz des Flügelkasten-Modells 70
4.6 Komponenten-basiertes CFD-Oberflächennetz eines DLR-F11-Modells in Hochauftriebskonfiguration 71
4.7 Oberflächennetz des Strukturmodells eines DLR-F11-Modells in Hochauftriebskonfiguration 72
4.8 Benötigte Zeit bei Variation der Stützstellenzahl und gegebener Netzgröße 74
4.9 Berechnungsaufwand bei Variation der Netzgröße und 32,64 und 96 Stützstellen 75
4.10 Vorgegebenes, eindimensionales Feld auf dem Strukturnetz aus etwa 2500 Knoten 76
4.11 Rekonstruiertes, skalares Feld auf dem CFD-Oberflächennetz bei 32 Partitionen und RCB-basierter Partitionierung 77
4.12 Zeit zur Berechnung der Kopplungsmatrix und Interpolation bei unterschiedlicher Prozessor- und Stützstellenanzahl 78
4.13 Gemessener Speedup für das gewählte Rechtecknetz aus etwa 250000 Oberflächen- und Strukturknoten 79
4.14 Zeitdauer zum Versenden und Empfangen der Koordinaten der Nächsten-Nachbarn für 2 unterschiedliche Stützstellenanzahlen . . 80
4.15 Berechnungsdauer für unterschiedliche Prozesszahlen bei Variation der Netzgrößen und 64 Stützstellen 81
4.16 Verformungswerte des Strukturnetzes 82

4.17 Verformungsinterpolation des HiReTT-Flügels bei 32 Prozessen und 128 Stützstellen 83
4.18 Schnittansicht des Flügels in Spoiler-Nähe im Vergleich zur unverformten Lage 84
4.19 Verformungsinterpolation des HiReTT-Flügels bei 96 Prozessen und 64 Stützstellen 85
4.20 Verformungsartefakte entlang des Oberflächennetzes bei Zoltan-basierter Partitionierung des Strukturnetzes und 96 Partitionen sowie 128 Stützstellen 87
4.21 Verformungsinterpolation eines Flügelkasten-Modells im Vergleich zur unverformten Lage 88
4.22 Aufgetragenes Verformungsfeld auf einzelnen Strukturkomponenten 89
4.23 Ungenaue Verformungsinterpolation bei Blending-loser Interpolation 90
4.24 Blending-Faktoren der direkten Baugruppenzuordnungen 92
4.25 Ausgewählte Blending-Radien, resultierend aus indirekter Baugruppenzuordnung 93
4.26 Ergebnis der Bauteil-basierten Verformungsinterpolation auf dem CFD-Oberflächennetz 94
4.27 Gegenüberstellung von direkten und indirekten Verformungsanteilen in z-Richtung 95

6.1 Bounding-Box einer Netzmenge und Suchbox der Bereichssuche . 102

Literaturverzeichnis

[1] Giuseppe Quaranta, Pierangelo Masaratiy, and Paolo Mantegazzay. A Conservative Mesh-free Approach for Fluid-Structure Interface Problems. *Int.Conf. on Computational Methods of Coupled Problems in Science and Engineering*, 2005.

[2] Dieter Schwammborn, Thomas Gerhold, and Ralf Heinrich. The DLR TAU-code: Recent Applications in Research and Industry. In *Proceedings of European Conference on Computational Fluid Dynamics*, 2006.

[3] Bernd Stickan. Implementation and Extension of a Mesh Deformation Module for the Parallel FlowSimulator Software Environment. Master's thesis, RWTH Aachen, 2009.

[4] Carsten Braun. *Ein modulares Verfahren für die numerische aeroelastische Analyse von Luftfahrzeugen*. PhD thesis, 2007.

[5] Horst Flister. Netzfreie Methoden zur Lösung des räumlichen Kopplungsproblems in aeroelastischen Simulationen. Studienarbeit, RWTH Aachen, 2009.

[6] Klaus-Jürgen Bathe. *Finite-Elemente-Methoden*. Springer Verlag, 1990.

[7] R. M. V. Pidaparti. Structural and aerodynamic data transformation using inverse isoparametric mapping. *Journal of Aircraft*, 29(3):507–509, 1992.

[8] Regine Ahrem, Armin Beckert, and Holger Wendland. A new multivariate interpolation method for large-scale spatial coupling problems in aeroelasticity. In *the Proceedings to Int. Forum on Aeroelasticity and Structural Dynamics*, 2005.

[9] Holger Wendland. Spatial Coupling in Aeroelasticity by meshless kernel-based methods. *European Conference on Computational Fluid Dynamics*, 2006.

[10] Paul Lettich. Parallel and fullscalable partitioning of hybrid grids. Master's thesis, Universität Göttingen, 2009.

[11] Aydin Buluc, Henning Meyerhenke, Ilya Safri, Peter Sanders, and Christian Schulz. Recent Advances in Graph Partitioning. 2013.

[12] Yunjun Gao, Ling Chen, Gencai Chen, and Chun Chen. Efficient Parallel Processing for K-Nearest-Neighbor Search in Spatial Databases. In *ICCSA (5)*, volume 3984 of *Lecture Notes in Computer Science*, pages 39–48. Springer, 2006.

[13] Apostolos Papadopoulos and Yannis Manolopoulos. Parallel Processing of Nearest Neighbor Queries in Declustered Spatial Data. 1997.

[14] Cui Yu, Beng CHin Ooi, Kian-Lee Tan, and H.V.Jagadish. Indexing the Distance: An Efficient Method to KNN Processing, 2001.

[15] Nima Moshtagh. Minimum Volume Enclosing Ellipsoids. 2005.

[16] Frank Spiering, Ralf Heinrich, and Stefan Keye. Development of a Parallel Fluid-Structure Coupling Environment and Application to a Wind Tunnel Model under High Aerodynamic Loads. In *Notes on Numerical Fluid Mechanics and Multidisciplinary Design New Results in Numerical and Experimental Fluid Mechanics VIII*. Springer Verlag, 2013.

[17] Jean-Luc Hantrais-Gervois, David Sawyers, Abdul Rampurawala, Lesmana Djayapertapa, Nicola Ceresola, Ralf Heinrich, Lars Tysell, Jaap van Muijden, and Ernst Totland. GARTEUR AD/AG-45: Application of CFD to Predict High "g" Loads. Technical report, 2013.

[18] Georg Hager and Gerhard Wellein. *Introduction to High Performance Computing for Scientists and Engineers*. CRC Press, 2010.

[19] R.Ahrem, A.Beckert, and H.Wendland. Recovering rotations in aeroelasticity. *Journal of Fluids and Structures*, 2007.

[20] Javier Bonet and Jaime Peraire. An Alternating Digital Tree (ADT) Algorithm for 3D Geometric Searching and Intersection Problems. *International Journal for Numerical Methods in Engineering*, 1991.

[21] Holger Wendland. Hybrid Methods for Fluid-Structure-Interaction Problems in Aeroelasticity. In *Meshfree Methods for Partial Differential Equations IV*. Springer Verlag, 2008.

IB 124-2014/6

Eine netzfreie Last-/Verformungsinterpolationsmethode im FlowSimulator für die Anwendung in statischen aeroelastischen Simulationen von Flugzeugen – Implementierung und Analyse

Andreas Schuster

Verteiler:

Institut für Aerodynamik und Strömungstechnik BS	1	Exemplar
Verfasser	1	Exemplar
Prof. Dr.. C.-C. Rossow	1	Exemplar
Prof. Dr.-Ing. N. Kroll	1	Exemplar
Dr.-Ing. R. Heinrich	1	Exemplar
Dipl.-Ing. L. Reimer	1	Exemplar
Technische Informationsbibliothek Hannover	1	Exemplar
Niedersächsische Landesbibliothek Hannover	1	Exemplar
Deutsch Bibliothek Frankfurt am Main	2	Exemplare
DLR Zentralbibliothek Braunschweig	2	Exemplare
Reserve	2	Exemplare
	14	Exemplare