Automatische Erstellung von Submodellen für die Crashoptimierung von Fahrzeugkarosserien

Dissertation
zur Erlangung eines Doktorgrades

in der
Fakultät für Maschinenbau und Sicherheitstechnik
der
Bergischen Universität Wuppertal

vorgelegt von
Sven Wielens
aus Gelsenkirchen

Wuppertal 2022

Tag der mündlichen Prüfung: 22.06.2022

Sven Wielens

Automatische Erstellung von Submodellen für die Crashoptimierung von Fahrzeugkarosserien

Dissertation, Bergische Universität Wuppertal,
Fakultät Maschinenbau und Sicherheitstechnik,
Lehrstuhl für Optimierung mechanischer Strukturen, Juli 2022

Kurzfassung

Bei der Entwicklung von Fahrzeugkarosserien spielt die Crashauslegung eine große Rolle. Dabei stehen der Insassenschutz und der Schutz kritischer Komponenten wie der Kraftstoffversorgung im Vordergrund. Ein Werkzeug zur Unterstützung der beteiligten Ingenieure kann die auf Crashsimulationen basierte Strukturoptimierung sein. Das Problem bei der Optimierung von Crashproblemen sind die langen Rechenzeiten der Crashsimulationen. Im Laufe einer Optimierung können hunderte Simulationen notwendig sein, was zu einem enormen Ressourcenbedarf und somit zu hohen Kosten führt.

Durch die Reduktion der Berechnungsmodelle kann der Ressourcenbedarf eingeschränkt werden. In dieser Dissertation werden verschiedene Ansätze zur Modellreduktion umgesetzt. Dabei werden Bauteile oder Elemente aus dem Modell entfernt und durch geeignete Randbedingungen ersetzt. So wird etwa eine Ersatzmasse an das verbleibende Modell angebunden, um die Trägheit und fehlende Masse auszugleichen. Damit das reduzierte Modell in seinen Eigenschaften dem Ursprungsmodell entspricht, wird eine Validierung durchgeführt.

Reduzierte Modelle können z. B. in Fallstudien oder Optimierungen eingesetzt werden. Durch den verringerten Ressourcenbedarf können in derselben Zeit mehr Simulationen durchgeführt werden, wodurch bessere Ergebnisse erzielt werden.

Anhand von Beispielen wird die entwickelte Methodik in der Praxis angewandt. Dabei dienen zwei Fahrzeugoptimierungen als Benchmark.

Stichworte

Crashoptimierung; Fahrzeugkarosserie; Submodell; Multi-Level Optimierung

Sven Wielens

Automatically generated submodels in vehicle crash optimization

PhD thesis, University of Wuppertal,
School of Mechanical Engineering and Safety Engineering,
Chair of Optimization of Mechanical Structures, July 2022

Abstract

Crash structure development is a key element in the design of vehicle bodies. The focus here is on occupant protection and the protection of critical components such as the fuel supply. Crash simulation based structural optimization can be a tool to support the engineers involved. The problem with optimizing crash problems is the long computing times of crash simulation. Hundreds of simulations may be necessary in the course of an optimization, which leads to enormous resource requirements and thus to high costs.

By reducing the computational model, the resource requirements can be limited. In this dissertation different approaches to model reduction are realized. Components or elements are removed from the model and replaced by suitable boundary conditions. For example, a substitute mass is attached to the remaining model to compensate for inertia and missing mass. To ensure that the reduced model corresponds to the original model in its characteristics, a validation is carried out.

Reduced models can be used e.g. in case studies or optimizations. Due to the reduced resource requirements, more simulations can be carried out in the same time, which leads to better results.

Examples are used to apply the developed methodology in practice. Two vehicle optimizations serve as benchmarks.

Keywords

crash optimization; vehicle body; submodel; surrogate model; multi-level optimization

Danksagung

Ich möchte mit einer Warnung für all diejenigen, die sich überlegen zu promovieren beginnen: Überlegt es euch gut! Der Weg ist lang und euer Fortbewegungsmittel ist ein brennendes Fahrrad mit platten Reifen.

Wieso habe ich es trotzdem ans Ziel geschafft? Ich hatte jede Menge Hilfe und bei einigen, die mich besonders unterstützt haben, möchte ich mich jetzt bedanken.

Zuerst sind da meine Eltern

Heike & Ralf Wielens

zu nennen. Ohne euch wäre ich nie bis hier gekommen. Ihr habt mich immer unterstützt und schon als Kind meinen Forschergeist geweckt. Vielen lieben Dank!

Dann möchte ich mich bei meiner Frau

Stefanie Breuer

bedanken. Danke dass du, vor allem in der letzten Zeit, so viele Nerven für mich geopfert hast. Auf dich kann ich mich immer verlassen.

Mein bester Freund

Nils Schäfer

darf hier natürlich nicht fehlen. Vielen Dank für die vielen gemeinsamen Jahre und die, die noch kommen werden.

An dieser Stelle möchte ich mich auch bei meinen Kollegen bedanken. Insbesondere sind hier *Simon Link, Katrin Weider* und *Jens Trilling* zu nennen:

Simon, ich danke dir vielmals für die gemeinsame Zeit im Büro. Die vielen Schlachten mit Gummifletschern, Papierfliegern und diverse andere Streiche haben immer wieder für gute Laune während der Arbeit gesorgt. Natürlich gab es auch sinnvolle Gespräche, nicht dass jetzt jemand denkt wir hätten nur Unfug gemacht.

Katrin, vielen Dank für deine Hilfe und Tipps während meiner Zeit am Lehrstuhl und insbesondere bei der Promotion. Auf deinen Rat kann man immer zählen.

Jens, dir möchte ich für deinen Beitrag zu meiner Forschung danken.

Zum Schluss möchte ich mich herzlich bei meinem Doktorvater

Axel Schumacher

bedanken. Danke, dass du mich in dein Team aufgenommen und mir die Dissertation ermöglicht hast.

Ohne euch alle wäre diese Arbeit hier nie möglich gewesen. Ich danke euch und auch denen, für die ich hier keinen Platz mehr hatte!

Inhaltsverzeichnis

Abkürzungs- und Symbolverzeichnis

Abkürzungen

ASCO	Automatic Submodel generation for Crash Optimization
BFS	Beifahrerseite
CAD	Computer-Aided-Design
CAE	Computer-Aided-Engineering
CBS	Coordinate Based Submodelgeneration
CFS	Crushable Frame Springs
CI	Connecting Island
COG	Center Of Gravity
D2RA	Deformable To Rigid Automatic
DMA	Displacement Model Approach
DoE	Design of Experiment (Versuchsplan)
DV	Designvariable (Entwurfsvariable)
eEgO	Effiziente Ersatzmodell gestützte Optimierung
EID	Element-Identifier
ePBS	extended Performance Based Submodelgeneration
FE	Finite Elemente
FEM	Finite-Element-Methoden
FS	Fahrerseite
GHT	Graphen- und Heuristikbasierte Topologieoptimierung
ID	Identifier
ISF	Interface Segment File
KMA	Kinematic Mass Approach
MPP	Massively Parallel Processing
NCAP	New Car Assessment Programme
NID	Node-Identifier
NSID	Nodeset-Identifier
OEM	Original Equipment Manufacturer (Erstausrüster)
OLC	Occupant Load Criterion
PBS	Performance Based Submodelgeneration
PCA	Principal Component Analysis
PID	Part-Identifier

RB	Randbedingung
RBE	Rigid Body Element
RBFN	Radial Basis Function Network
RCAR	Research Council for Automobile Repairs
SID	Set-Identifier
SML	Submodellbasierte Multi-Level-Optimierung
SRSM	Sequential Response Surface Method
VPS	Virtual Performance Solution

Symbole

a	Abstand zwischen Knoten der Spritzschutzwand und einem Knoten im Dach
d	Abstand der Stützstellen
DV	Wert der Entwurfsvariablen
E_i	Wert der Evaluationsfunktion für ein Bauteil i
ε	Dehnung
f	Wert der Zielfunktion
F	Kraft
i	Iterator für Bauteile
I_p	Intrusion der Spritzschutzwand mit P-Norm
n	Iterator für Knoten
n_{DV}	Anzahl der Entwurfsvariablen
p	Exponent für die P-Norm
R	Bewertungsparameter
r_k	Wert der *k*-ten Restriktion
σ	Standardabweichung
s	Verschiebung
$S_{i,n,t}$	auszuwertende Strukturgröße für einen Knoten n eines Bauteils i in einem Zeitschritt t
t	Iterator für Zeitschritte
x_k	Wert der *k*-ten Strukturantwort

1. Einleitung

Der Entwicklungsprozess eines Fahrzeugs ist durch eine Vielzahl von Disziplinen und Varianten eines Modells geprägt. Eine für die Auslegung des Fahrzeugs wichtige Disziplin ist die Fahrzeugsicherheit. Diese hat mehrere Teilgebiete. Zuerst wird zwischen aktiver und passiver Sicherheit unterschieden. Bei der aktiven Sicherheit handelt es sich um die Kollisionsvermeidung. Die passive Sicherheit hingegen befasst sich mit „Unfallfolgen mindernde[n] Maßnahmen“ (KRAMER 2009, S. 3).

1.1 Passive Sicherheit im Entwicklungsprozess

Die passive Sicherheit greift, wenn der Unfall nicht mehr vermieden werden kann und die Aufgabe vor allem darin besteht, den Insassen zu schützen. Dazu gehört nicht nur der direkte Schutz des Insassen vor Verletzungen, sondern auch der Schutz von kritischen Komponenten, wie die Kraftstoffversorgung oder bei Elektrofahrzeugen der Akku. Eine Beschädigung dieser Bauteile könnte durch die Explosions- und Brandgefahr katastrophale Folgen für die Insassen und andere Beteiligte haben.

Innerhalb der passiven Sicherheit gibt es verschiedene Teilbereiche. So kann sich mit den Rückhaltesystemen beschäftigt werden, die die Unfallfolgen für den Insassen abmindern, indem die auf ihn wirkenden Beschleunigungen reduziert werden. Beispiele dafür sind Airbags oder Gurte. Die Karosserieentwicklung spielt hier ebenfalls eine große Rolle. Auch durch eine geeignete Deformation der Karosserie können die wirkenden Beschleunigungen geringgehalten werden. Außerdem gilt es, einen Überlebensraum im Fahrzeuginneren zu schaffen, der im Falle des Unfalls bestehen bleibt, sodass die Insassen keine Verletzungen durch Quetschungen erfahren.

Im heutigen Entwicklungsprozess spielt die Simulation eine große Rolle, da mit ihrer Hilfe viele verschiedene Fahrzeugvariationen in relativ kurzer Zeit bewertet werden können. Der Aufbau von Prototypen für jede Disziplin und Variante würde den Entwicklungszyklus bremsen und so dem kürzer werdenden Wechsel der Fahrzeuggenerationen entgegenstehen (vgl. KRAMER 2009, S. 391). Durch die Simulation im Allgemeinen besteht die Möglichkeit in relativ kurzer Zeit Vorhersagen für das Verhalten des Fahrzeugs zu generieren. In der Karosserieauslegung, für den Fahrzeugcrash im Speziellen, sind Simulationen eines der wichtigsten Werkzeuge. Allerdings hat eine Simulation einer Fahrzeugkollision einen hohen Bedarf an Zeit und Ressourcen. Eine einzelne Simulation benötigt, je nach vorhandener Rechenleistung, mehrere Stunden bis Tage. Beispielsweise sind, um ein vergleichsweise kleines Gesamtfahrzeug mit

knapp 2 Mio. Elementen über Nacht, also in ca. 15 Stunden, zu simulieren, 128 CPU-Kerne[1] eines Rechenclusters notwendig.

Ein weiteres Werkzeug in der Entwicklung ist die Optimierung. Sie dient zur Ermittlung der bestmöglichen Kombination von veränderlichen Parametern, wie Blechdicken oder Sickenpositionen des Berechnungsmodells. Dazu werden Optimierungsalgorithmen eingesetzt, welche die Werte der Parameter steuern und anhand der Ergebnisse des Berechnungsmodells, den optimalen Parametersatz finden. Eine typische Optimierung eines Crashs könnte die Reduktion der Fahrzeugmasse sein. Dabei dürfen die auf die Insassen wirkenden Beschleunigungen und die Deformationen der Fahrgastzelle einen definierten Grenzwert nicht überschreiten. Für eine solche Optimierung werden mehrere hundert Simulationen benötigt, da der Optimierungsalgorithmus zur Bestimmung des Optimums Daten benötigt. Die hohe Anzahl nötiger Simulationen, kombiniert mit dem hohen Ressourcenbedarf einer einzelnen Simulation, führt zu hohen Kosten. Außerdem ist der Zeitaufwand für eine solche Optimierung nicht immer mit ihrem Nutzen vereinbar. Wird nur eine geringe Verbesserung während der Optimierung erzielt, ist es nicht sinnvoll, eine Optimierung über mehrere Wochen laufen zu lassen.

Um den Zeit- und Ressourcenbedarf einer Optimierung von Crashmodellen zu reduzieren, gibt es diverse Ansätze. Beispielsweise können effizientere Optimierungsalgorithmen die Menge nötiger Simulationen reduzieren. Vereinfachte Modelle dagegen können den Aufwand für eine einzelne Simulation verringern. Auch in dieser Arbeit ist die Reduktion des Simulationsaufwands Thema. Es werden Ansätze vorgestellt, mit denen sich das Simulationsmodell verkleinern lässt, ohne dabei zu viel Aussagekraft der Daten zu verlieren. Dabei erfordern unterschiedliche Anwendungen unterschiedliche reduzierte Modelle, die auf die Anforderungen der speziellen Aufgabe angepasst sind.

1.2 Problemstellung und Motivation

Durch eine geschickte Manipulation des Simulationsmodells ist eine Einsparung der Rechenzeit von Crashsimulationen möglich. Bauteile, welche nicht unmittelbar benötigt werden, z. B. weil sie nur geringfügig am Crashvorgang beteiligt sind, werden aus dem Modell gelöscht.

[1] Für dieses Beispiel wurden vier AMD EPYC 7452 Prozessoren auf dem PLEIADES Rechencluster mit je 32 Kernen bei 2,35 GHz verwendet (siehe PLEIADES Hardware 2022).

Dadurch, dass aus dem Modell große Teile entfernt werden, wird das strukturmechanische Verhalten verändert. Gelöschte Komponenten sind durch Randbedingungen, welche das ursprüngliche Modellverhalten wiederherstellen, zu ersetzen. Dabei müssen z. B. die Energien, Impulse und das globale, sowie lokale kinematische Verhalten, dem Ursprungsmodell entsprechen. Erreicht werden kann dies nur durch geeignete Randbedingungen und eine detaillierte Validierung des reduzierten Modells.

1.3 Aufbau der Dissertation

Die Arbeit besteht neben der Einleitung aus fünf weiteren Kapiteln. Beginnend mit dem Stand der Technik erfolgt eine Einführung in die Crashsimulation und die Optimierung. Darauf folgt ein Überblick über die verschiedenen Ansätze von Ersatzmodellen. Es werden sowohl mathematische, als auch das Modell manipulierende physikalische Ersatzmodelle dargelegt. Zum Schluss des zweiten Kapitels wird der Bogen zurück zur Optimierung geschlagen; diesmal die Optimierung mit Submodellen, also physikalischen Ersatzmodellen.

In dem darauffolgenden dritten Kapitel wird die konkrete Umsetzung der Submodelltechniken beschrieben. Dabei werden zu Beginn die Anforderungen an das Submodell skizziert. Anschließend folgt die Submodellerstellung in drei Schritten: Modellreduktion, Randbedingungen und Validierung. Jeder der drei Schritte bietet mehrere Ansätze, welche bereits aus Kapitel 2 bekannte Verfahren aufgreifen, oder in ihrer Art neu sind. Den Abschluss des dritten Kapitels bildet der Workflow zur Erstellung und Nutzung von Submodellen.

Die Nutzung der Submodelle wird im vierten Kapitel aufgenommen. Darin geht es hauptsächlich um den Einsatz von Submodellen in der Optimierung. In einem weiteren Abschnitt werden alternative Anwendungsbereiche für Submodelle vorgestellt.

Im fünften und letzten inhaltlichen Kapitel werden zwei Optimierungsbeispiele vorgestellt, in denen Submodelle genutzt wurden. Dabei wird zuerst eine Prinzipuntersuchung mit einem relativ kleinen Fahrzeugmodell gezeigt. Die zweite Optimierung ist ein industrienahes Beispiel mit einem detaillierteren Modell und einer realistischen Optimierungsaufgabe.

Zum Abschluss der Arbeit werden die Ergebnisse zusammengefasst und bewertet. In einem ausführlichen Ausblick werden die Handlungsfelder für zukünftige Forschungsarbeiten formuliert.

2. Stand der Technik

Um die Reduktion von Crashsimulationsmodellen besser zu verstehen, wird zuerst die Crashsimulation in den Entwicklungsprozess eingeordnet. Daraufhin gilt es, die prinzipielle Struktur eines Crashmodells darzulegen, da viele Probleme der Modellreduktion auf den Aufbau des Modells zurückzuführen sind. Als letzter Teil der Grundlagen folgen Erklärungen zur Optimierung von Crashstrukturen. Durch den hohen Ressourcenbedarf stellen Crashmodelle einen besonderen Anspruch an die Optimierungsalgorithmen.

Nachdem die Grundlagen dargestellt wurden, folgen Vorstellungen von verschiedenen Ansätzen zur Erstellung von Ersatzmodellen. Dabei werden die mathematischen Ersatzmodelle umrissen, jedoch liegt der Schwerpunkt auf den physikalischen Ersatzmodellen. Zum Abschluss dieses Kapitels wird die Einbringung von physikalischen Ersatzmodellen in die Optimierung eingebracht.

2.1 Crashsimulation im Entwicklungsprozess

Neben Vorschriften und Gesetzen gibt es zum Beispiel das *European New Car Assessment Programme* (Kurz: Euro NCAP) als einen Herausgeber von Crashanforderungen. In anderen Ländern gibt es ähnliche Verbraucherschutzinstitutionen, die auch ähnliche Crashszenarien fordern. Beim Euro NCAP gibt es Protokolle für verschiedene Crashszenarien, aus denen am Ende eine Gesamtbewertung für einen Neuwagen ermittelt wird. Dabei sind bis zu fünf Sterne erreichbar. Eine Auswahl solcher Szenarien aus Gesetzen oder Euro NCAP (zusammengefasst in CARHS.TRAINING GMBH 2019, S. 22–23) sieht wie folgt aus:

- Seitenaufprall gegen einen Pfahl (siehe Bild 1a): Das Fahrzeug rutscht schräg seitlich, unter 75°, mit 32 km/h gegen einen Pfahl mit 254 mm Durchmesser. Es gibt einen Dummy auf dem Fahrersitz, der in Aufprallrichtung des Pfahls sitzt.
- Frontalaufprall mit voller Überdeckung gegen eine starre Wand (siehe Bild 1b): Hier wird das Fahrzeug mit zwei Dummys gegen eine starre Barriere gefahren. Die Geschwindigkeit beträgt dabei 50 km/h.
- Frontalaufprall gegen eine deformierbare Barriere mit 40% Überdeckung (siehe Bild 1c): Ähnlich wie beim vorherigen Lastfall fährt das Fahrzeug mit zwei Dummys frontal gegen eine Barriere. Hier besteht die Barriere allerdings aus einem deformierbaren Material und überdeckt zu 40% mit dem Fahrzeug. Das Fahrzeug hat eine Geschwindigkeit von 56 km/h.

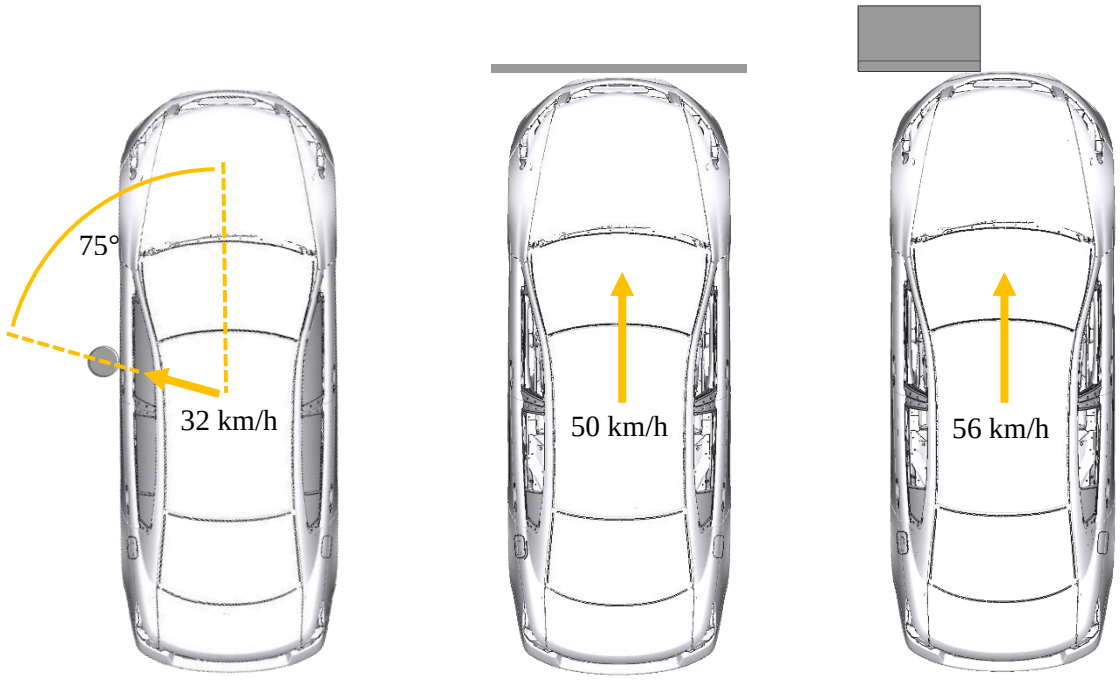


Bild 1: Beispielhafte Fahrzeugcrashszenarien nach (CARHS.TRAINING GMBH 2019, S. 22–23): (a) Pfahlaufprall; (b) Frontalaufprall bei voller Überdeckung; (c) Frontalaufprall gegen eine deformierbare Barriere

Ein weiterer wichtiger Herausgeber von Crashlastfällen für den Fahrzeughersteller soll hier nicht unerwähnt bleiben: Versicherungen. Beispielsweise vom *Research Council for Automobile Repairs* (Kurz: RCAR), in dem u. a. die Allianz Versicherung Mitglied ist, gibt es einige Lastfälle. Diese Lastfälle sind vor allem Crashs mit niedriger Geschwindigkeit, um die Reparaturkosten eines leichten Unfalls abzuschätzen und somit die Ersteinstufung für die Versicherungskosten eines Neuwagens zu bestimmen. (vgl. RCAR 2011a, 2011b)

Diese Vielzahl von Crashanforderungen an ein Fahrzeug verdeutlichen den Aufwand im Entwicklungsprozess. Erfüllt ein neues Fahrzeug z. B. alle gesetzlichen Vorgaben, so kann es theoretisch auf den Markt gebracht und zugelassen werden. Problematisch könnte dabei aber sein, wenn es beim Euro NCAP nur zwei oder drei Sterne erreicht hat, da dies Kunden vom Kauf abschrecken könnte. Eine andere Gefahr ist, dass das Fahrzeug bei den RCAR Lastfällen schlecht abschneidet, sodass die Versicherungsbeiträge in der Ersteinstufung sehr hoch angesetzt werden. Dies ist vor allem für Flottenfahrzeuge, die z. B. von Vermietungen genutzt werden relevant.

Mit Crashsimulationen können Entwürfe im Entwicklungsprozess schnell bewertet werden. Somit sind Änderungen des Designs schnell überprüft und es kann das Verbesserungspotential herausgearbeitet werden.

Eine Crashsimulation birgt jedoch die Problematik, dass ein Crash hochgradig nichtlinear und hochdynamisch ist. Crashsimulationen sind, z. B. aufgrund der folgenden Eigenschaften, nichtlinear (vgl. WAGNER 2019):

- Geometrische Nichtlinearität wird hervorgerufen durch große Verformungen und Verschiebungen.
- Materielle Nichtlinearität besteht aufgrund von dehnungsabhängigem Materialverhalten.
- Nichtlinearität durch Randbedingungen bedeutet, dass durch Kontaktsituationen das Modellverhalten beeinflusst wird.

Dadurch ist der Ressourcenbedarf einer Crashsimulation um ein Vielfaches größer als der einer linear-statischen Struktursimulation. Auch auf Hochleistungsrechnern ergeben sich Zeiten von einigen Stunden, bis hin zu Tagen für eine einzelne Simulation.

2.2 Bestandteile eines Modells für die Crashsimulation

In diesem Abschnitt sollen einige grundlegende Hinweise zu Crashsimulationen gegeben werden. Die hier verwendeten Erklärungen beziehen sich auf den *Finite-Elemente-Solver* (Kurz: FE-Solver) LS-DYNA® der Firma LST (ANSYS), da dieser im Rahmen der Arbeit hauptsächlich verwendet wird.

Generell bestehen FE-Modelle immer aus Knoten (engl.: *Nodes*) und Elementen (engl.: *Elements*). Dabei beschreiben die Knoten die Eckpunkte der Elemente, wodurch zwei aneinandergrenzende Elemente sich Knoten teilen. Mehrere Elemente ergeben ein Bauteil (engl.: *Part*), welche entweder durch gemeinsame Knoten oder besondere Verbindungselemente miteinander verbunden sein können. Bild 2 zeigt die Zusammenhänge am Beispiel eines Fahrzeugmodells.

Jedem Baustein des Modells wird ein Identifikator (ID) zugeordnet. Dabei dürfen Nummerierungen auch mehrfach vorkommen, solange sie in verschiedenen Kategorien von Bausteinen (Knoten, Elemente etc.) auftreten. Die Identifikatoren werden dann nach der entsprechenden Kategorie bezeichnet, wie die folgenden Beispiele zeigen:

- NID: Node-ID
- EID: Element-ID
- PID: Part-ID

Bei Materialkarten werden häufig dieselben IDs verwendet, wie die der zugehörigen *Parts*. Dies ermöglicht eine einfache Zuordnung der Materialien zu den entsprechenden Bauteilen. Voraussetzung hierbei ist jedoch eine eigene Materialkarte für jedes Bauteil zu haben.

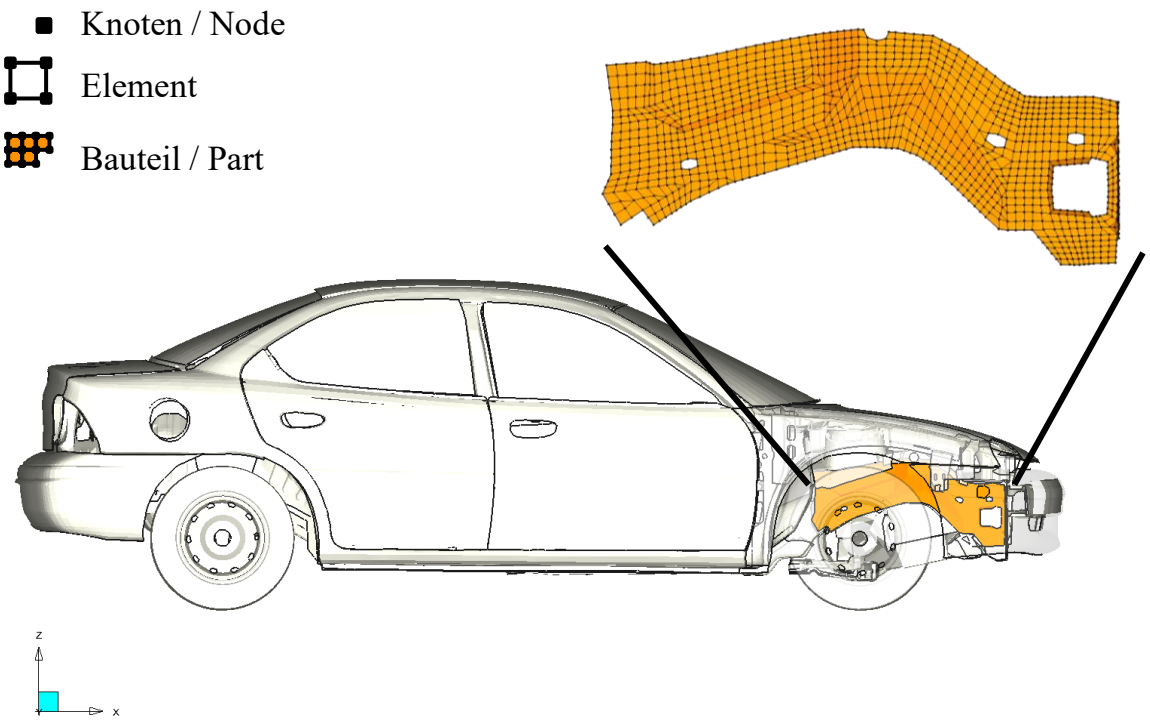


Bild 2: Veranschaulichung der Komponenten eines FE-Modells

Um ganze Gruppen ansprechen zu können, besteht die Möglichkeit *Sets* zu bilden. Ein Set kann aus Knoten, Elementen oder Bauteilen bestehen, wobei diese nicht vermischt werden können. Auch einer erstellten Gruppe wird eine ID zugeordnet, die *Set*-ID (SID). Ein *Nodeset* zum Beispiel, also eine Gruppe aus Knoten, erhält eine *Nodeset*-ID (NSID). Bei Fahrzeugmodellen werden oft Baugruppen in *Sets* zusammengefasst, um diese einfach austauschen zu können oder Kontaktbedingungen nur auf bestimmte Bereiche anzuwenden.

Damit Kontaktbedingungen greifen können, sind Durchdringungen des Modells zu identifizieren. Damit ist sowohl der Kontakt zwischen zwei Objekten, als auch der Kontakt eines Objekts mit sich selbst gemeint. Anschließend sind am detektierten Kontakt die Kontaktbedingungen zu erfüllen, sodass keine Durchdringung stattfindet (vgl. WAGNER 2019, S. 223 ff.). Diese Überprüfung erfordert einen großen Teil der Zeit bei einer Crashsimulation. In Bild 3 ist die Aufteilung des Zeitbedarfs in einem Diagramm dargestellt. Darin gibt es neben den Kontaktalgorithmen einen zeitintensiven Anteil der Elementverarbeitung, in den die Ermittlung der Deformation fällt. Unter Sonstiges

sind Aufgaben, wie das Importieren des *Inputdecks*, herausschreiben der Ergebnisse usw. zusammengefasst. Kontaktalgorithmen in LS-DYNA® sind in (LIVERMORE SOFTWARE TECHNOLOGY CORPORATION 2006) aufgeführt.

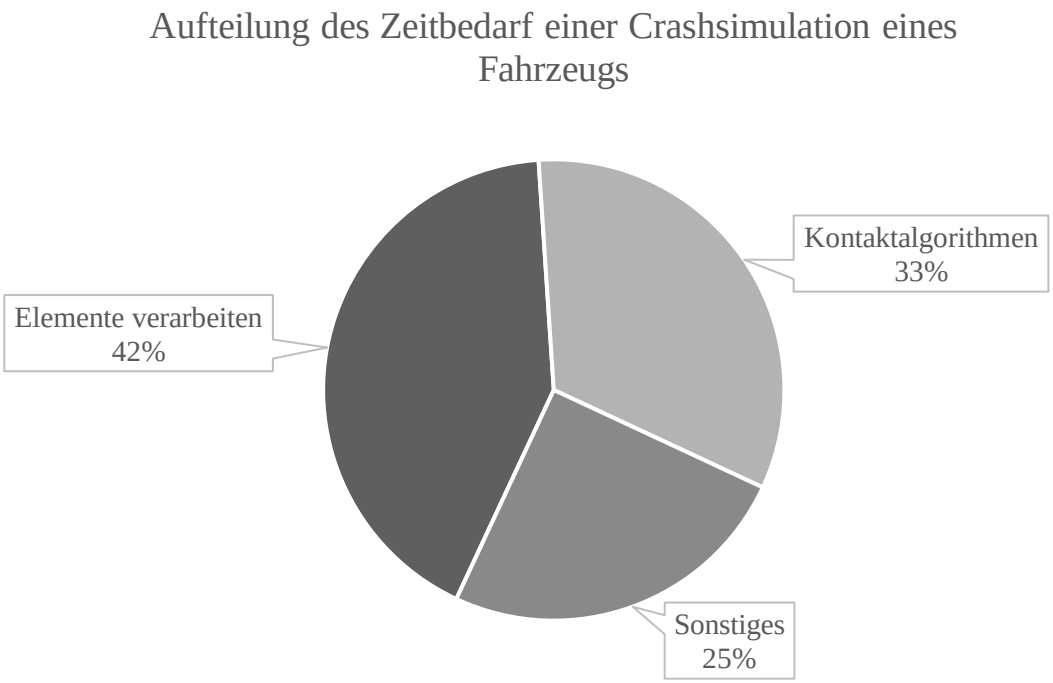


Bild 3: Aufteilung des Zeitbedarfs bei einer Crashsimulation des Toyota Yaris (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

Ein FE-Modell kann auf mehrere Dateien aufgeteilt sein. Das ist sinnvoll, um eine Ordnung zu schaffen, indem z. B. Baugruppen in einer einzelnen Datei gespeichert werden oder das Gesamtmodell von verschiedenen Personen erstellt wird. Um Überschneidungen der IDs zu vermeiden, können alle IDs einer Datei beim Import ein Offset erhalten. Das bedeutet, es wird jede ID in der Datei um einen vorgegebenen Wert verschoben. Gleiches ist für Koordinaten möglich, sodass der Inhalt einer Datei beim Import im Koordinatensystem verschoben oder rotiert wird.

Damit der *Solver*, also der Berechnungsalgorithmus das FE-Modell interpretieren kann, benötigt er ein *Solverdeck* oder auch *Inputdeck* genannt. Dies kann wie oben beschrieben eine einzelne oder mehrere Dateien sein, wobei es immer eine Hauptdatei gibt, von der aus alle weiteren Dateien verknüpft sind. Damit das *Inputdeck* interpretierbar ist, hat jeder *Solver* eine standardisierte Skriptsprache. Bei LS-DYNA® besteht diese aus *Keywords* und den zugehörigen Daten (vgl. LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY 2020, S. 362). Ein *Keyword* ist dabei eine Art Überschrift, welches durch ein Sternchen markiert wird und spezifiziert, wie die nachfolgenden Daten zu verarbeiten sind. Dabei kann ein *Keyword* mehrere

Optionen enthalten, wie das Beispiel in Bild 4 zeigt. Das *Keyword* `SET_NODE` bedeutet, dass Knoten einer NSID zugeordnet wird. Die erste Erweiterung `LIST` bedeutet dabei, dass es eine Liste von Knoten ist. Mit der letzten Option `TITLE` wird angezeigt, dass in der ersten Zeile nach dem *Keyword* nicht die NSID folgt, sondern ein Titel, der die Gruppe bezeichnet. (vgl. LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY 2020, 3197 ff.)

In dem konkreten Beispiel aus Bild 4 wird eine Liste mit dem Titel `INTERIOR_FLOOR_INSULATION_SYSTEM` erstellt, die die NSID 10000026 hat. Darunter folgt die Liste der Knoten, die diesem Set zugeordnet sind. Zeilen, die mit dem Dollar-Symbol beginnen, sind Kommentare und werden nicht vom *Solver* interpretiert. Für eine Datenzelle sind immer zehn Zeichen Platz, Ausnahmen davon sind möglich und abhängig vom *Keyword*, wie der Titel zeigt.

```
*SET_NODE_LIST_TITLE
$#   Title
INTERIOR_FLOOR_INSULATION_SYSTEM
$#    nsid
  10000026
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
  10301480  10301481  10301483  10301507  10301547  10301554  10301606  10301622
  10301958  10301990  10302046  10302055  10302075  10302119  10302137  10302176
  10302180  10302189  10302194  10302241  10302250  10302264  10302297  10302298
  10302301  10302302  10302313  10302335  10312369  10312381  10312384  10312387
```

Bild 4: Beispiel für ein *Keyword* mit zugehörigen Daten in LS-DYNA®; Ausschnitt aus dem Toyota Silverado aus (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

Es gibt eine Vielzahl an *Keywords*, die jeweils ihre eigenen Optionen und Kombinationen von Optionen haben. In (LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY 2020) sind alle von LS-DYNA® verarbeitbaren *Keywords* mit Erklärungen aufgezeigt.

2.3 Optimierung von Crashstrukturen

Allgemein bedeutet Optimierung für ein Problem die bestmögliche Lösung zu finden: das Optimum. Für die Strukturoptimierung bedeutet dies konkret:

> *"[…] structural optimization is the subject of making an assemblage of materials sustain loads in the best way."*
>
> (CHRISTENSEN & KLARBRING 2009, S. 1)

Dazu ist es notwendig, die Optimierungsaufgabe zu definieren. Es sind die veränderbare Struktur (Entwurfsvariablen, engl. *design variables*, Kurz: DV), die

Anforderungen (Ziele und Restriktionen) und die Lastfälle zu berücksichtigen. Eine Übersicht über die einzelnen Begriffe in der Optimierung ist in Tabelle 1 aufgeführt. (vgl. SCHUMACHER 2020, S. 5)

Die Optimierungsaufgabe ist so zu definieren, dass die Optimierungssoftware sie interpretieren und bearbeiten kann. Je nach verwendeter Software variiert die konkrete Umsetzung. Mithilfe eines geeigneten Optimierungsalgorithmus lässt sich das Problem dann lösen. Die Algorithmen werden meist vom Softwarehersteller bereitgestellt oder es können auch eigene implementiert werden. Aktuell stehen drei große Gruppen von Optimierungsalgorithmen zur Verfügung:

1. Approximationsbasierte Optimierungsalgorithmen: Hier werden mathematische Ersatzmodelle erstellt, mithilfe derer das Optimum bestimmt wird. Das Thema mathematische Ersatzmodelle wird im nachfolgenden Abschnitt im Detail ausgeführt.
2. Stochastische Optimierungsalgorithmen: Bei dieser Art der Optimierungsalgorithmen handelt es sich um zufallsgesteuerte Suchstrategien (vgl. BAIER ET AL. 1994, S. 77). Je nach konkreter Suchstrategie findet die Auswahl der weiteren Entwurfspunkte nach einem mehr oder weniger intelligenten Verfahren statt.
3. Künstliche Intelligenz: Der Einsatz von künstlicher Intelligenz ist dort sinnvoll, wo bereits große Datenmengen verfügbar sind. Sie ergänzt approximationsbasierte Algorithmen, wenn „mathematische[n] Optimierungsalgorithmen nicht nutzbar sind“ (SCHUMACHER 2020, S. 118).

Daneben gibt es weitere Ansätze, welche z. B. in (CHRISTENSEN & KLARBRING 2009) oder (SCHUMACHER 2020) nachzulesen sind.

Als Beispiel für einen stochastischen Optimierungsalgorithmus sei an dieser Stelle *Simulated Annealing* zu nennen, welches einem Abkühlprozess von Metallschmelze nachempfunden ist. Dabei ist die Bewegungsfreiheit der einzelnen Atome abhängig von der Temperatur der Schmelze. Langsames Abkühlen führt zu dem niedrigsten Energieniveau des erstarrten Metalls. Das Prinzip der erstarrenden Schmelze wird auf den Optimierungsalgorithmus übertragen. Bei hohen virtuellen Temperaturen haben die Entwurfsvariablen die Möglichkeit stark zu variieren, dies wird mit zunehmender Abkühlung eingeschränkt. Bei diesem Algorithmus werden auch Entwürfe zugelassen, die schlechter als der Vorherige sind. Dabei wird ebenfalls die virtuelle Temperatur als Kriterium herangezogen (vgl. RAO 2009, S. 702 ff). Weitere Details zu dieser Suchstrategie sind in (STANDER ET AL. 2020, S. 615) oder (KACPRZYK ET AL. 2012) zu finden. Eine andere Möglichkeit für stochastische Optimierungsalgorithmen sind genetische Algorithmen, welche sich an der biologischen Evolution orientieren. Es kann

dabei zu Mutationen und Paarungen zwischen verschiedenen Entwürfen (vgl. STANDER ET AL. 2020, S. 605 ff.) kommen.

Bild 5 zeigt den Ablauf einer Optimierungsschleife. Ausgehend vom Startentwurf wird das zugehörige Analysemodell berechnet und ausgewertet. Die Abzweigung unten rechts „Optimum erreicht?“ beschreibt die Konvergenz- bzw. Abbruchbedingung, also wann die Optimierungsschleife gestoppt wird. Bei einem Abbruch hat der Algorithmus das Optimum gefunden, andernfalls wird eine neue Iteration begonnen, für die die Entwurfsvariablen zu variieren sind. (vgl. SCHUMACHER 2020, S. 3)

Tabelle 1: Begriffe der Strukturoptimierung ergänzt nach (SCHUMACHER 2020, S. 6)

Optimierungsalgorithmus	Mathematisches Verfahren zur Minimierung einer Zielfunktion unter Berücksichtigung von Restriktionen
Optimierungsverfahren	Zusammenspiel zwischen Optimierungsalgorithmen und einem Optimierungsansatz
Optimierungsprozedur	Software zur Lösung einer Optimierungsaufgabe
Optimierungsstrategie	Vorgehen zur Vereinfachung einer komplexen Optimierungsaufgabe in einfache Ersatzprobleme
Zielfunktion	Mathematische Beschreibung des Optimierungsziels
Restriktionen	Einzuhaltende Anforderungen als mathematische Formulierung
Analysemodell	Mathematische Beschreibung des Zusammenhangs zwischen Entwurfs- und Zustandsvariablen unter Berücksichtigung des Lastfalls. Allgemein auch Simulationsmodell genannt
Zustandsvariablen	Auszuwertende Ergebnisse der Simulation
Entwurfsvariablen	Veränderliche Größen des Modells
Entwurfsraum	Bereich, in dem sich die Entwurfsvariablen verändern und somit der Bereich, in dem die Optimierung durchgeführt wird
Startentwurf	Werte der Entwurfsvariablen zu Beginn der Optimierung
Stützstelle	Ein Datenpunkt im Entwurfsraum, an dem das Analysemodell simuliert wurde und somit alle Zustandsvariablen ermittelt sind

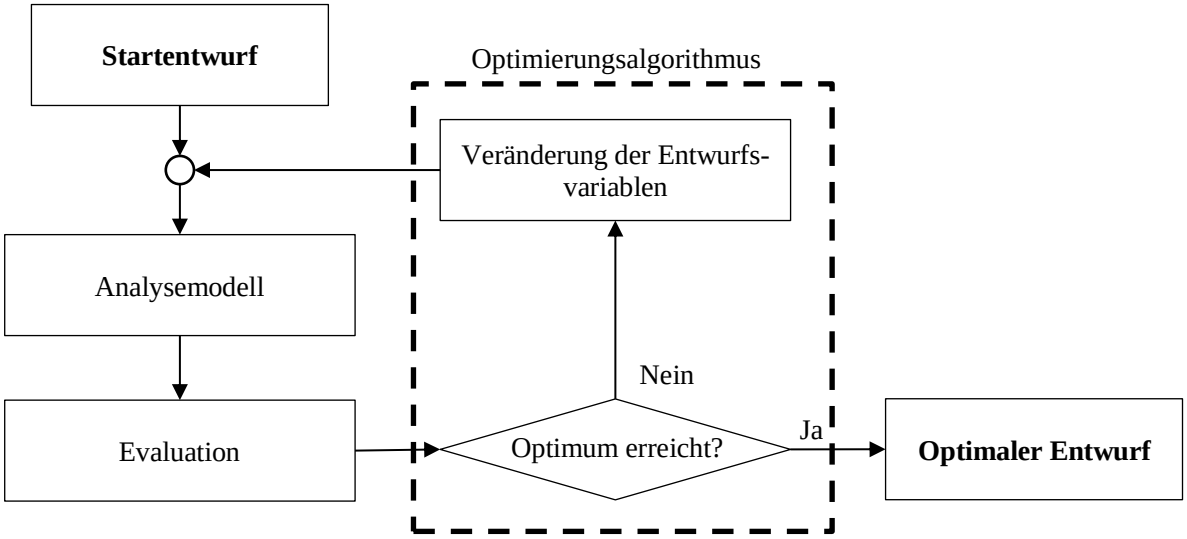


Bild 5: Ablaufplan einer Optimierungsschleife

Bis hierher sind die Erklärungen für die meisten Optimierungsprobleme gültig. Ab jetzt geht es darum, das Crashverhalten eines Fahrzeugs zu verbessern. Im Vergleich zur Optimierung linear statischer Probleme ist die Bestimmung von Sensitivitäten bei nichtlinearen Simulationen, wenn überhaupt, nur mit sehr großem Aufwand möglich (siehe WEIDER 2021).

Aufgrund der hohen Nichtlinearität und dem komplexen Verhalten des Modells ist das Finden des globalen Optimums nicht garantiert. Es besteht die Gefahr, dass der Optimierungsalgorithmus nur lokale Optima findet (vgl. FANG ET AL. 2017).

Eine weitere Problematik für den Optimierungsalgorithmus besteht in Bifurkationen. Dies sind Verzweigungen, bei denen sich durch kleine Änderungen der Entwurfsvariablen das Verhalten des Modells stark verändert (vgl. PLASCHKO & BROD 1995, S. 71). Beispielsweise kann ein Bauteil ausknicken, wodurch es zu einer sprunghaften Änderung der Strukturantworten und Deformationen kommt. Je nach Richtung in die das Teil knickt, sind verschiedene Ergebnisse zu erwarten (siehe Bild 6). Solche Bifurkationen erschweren die Interpretation für den Optimierungsalgorithmus, da sie ein unstetes Verhalten hervorrufen.

Durch die nichtlineare Simulation ergibt sich der, in der Einführung bereits erwähnte, hohe Ressourcenbedarf. Im Folgenden werden einige Möglichkeiten dargestellt, um den Ressourcenbedarf für die Optimierung eines Crashproblems zu reduzieren.

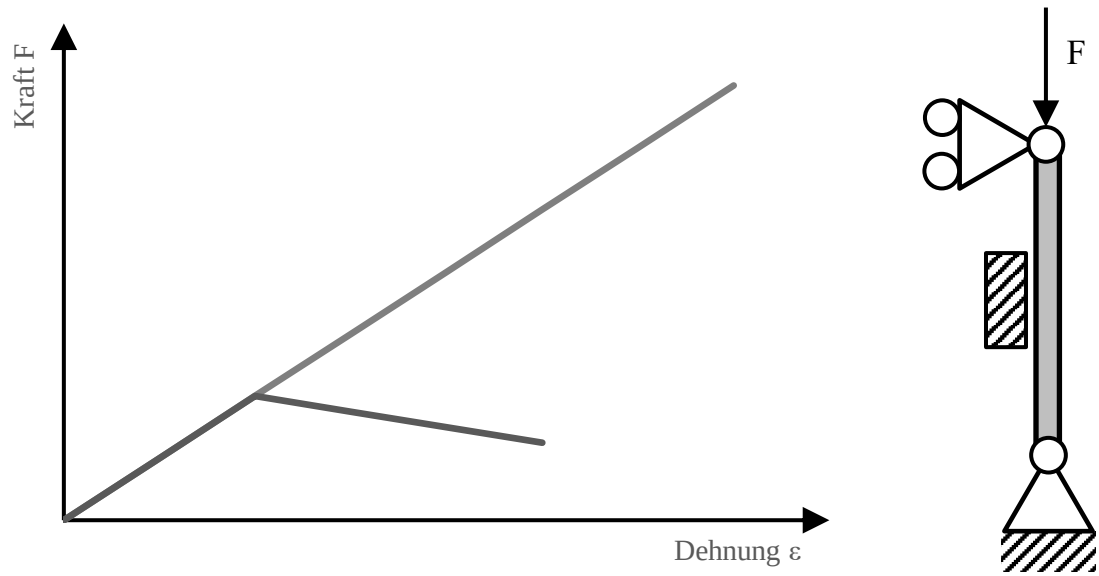


Bild 6: Veranschaulichung zu Bifurkationen anhand eines Knickstabs, der in zwei Richtungen ausknicken kann, nach (SCHUMACHER 2020, S. 176)

2.4 Mathematische Ersatzmodelle

Der erste Ansatz zur Reduktion des Ressourcenbedarfs einer Optimierung liegt in der Approximation des Entwurfsraums durch mathematische Ersatzmodelle, auch Metamodelle genannt (DUDDECK & WEHRLE 2015), wie sie im vorherigen Abschnitt bereits erwähnt wurden. Bei den mathematischen Ersatzmodellen ist das Ziel, den Rechenaufwand für die Optimierung zu verringern, indem eine Vorhersage über den Verlauf getroffen wird. Dazu werden Daten benötigt, anhand derer die Metamodelle trainiert werden. Solche Daten werden von Versuchsplänen (engl.: *Design of Experiment*, Kurz: DoE) erzeugt.

Bei einem DoE wird der Entwurfsraum systematisch abgetastet (vgl. STANDER ET AL. 2020, S. 545). So genannte Stützstellen sind Punkte im Entwurfsraum, an denen die Daten erfasst werden, in den meisten Fällen durch eine Simulation. Solche DoE können nach verschiedenen Schemata ablaufen, wie beispielsweise durch einen Zufallsgenerator. Ein Beispiel für ein restringiertes DoE, bei dem die Positionierung der Stützstellen gesteuert wird, ist *Latin Hypercube*. Für die Durchführung wird der Bereich einer jeden Entwurfsvariable in eine definierte Anzahl Intervalle aufgeteilt. Es wird dann pro Intervall ein zufälliger Wert bestimmt. Die Werte pro Intervall und pro Entwurfsvariable werden zufällig zusammengesetzt, sodass jede Stützstelle einen Wert für jede Entwurfsvariable erhält, jedoch keine zwei Stützstellen in einem Intervall liegen. Bild 7 verdeutlicht das Ergebnis für zwei Entwurfsvariablen. (vgl. STANDER ET AL. 2020, S. 548–549)

Basierend auf den vorhandenen Stützstellen des DoE werden Approximationsfunktionen erstellt, die den Entwurfsraum repräsentieren. Mithilfe dieser Metamodelle können Vorhersagen über weitere geeignete Stützstellen, bis hin zum Optimum getroffen werden. Problematisch dabei ist, dass insbesondere bei den hochdynamischen nichtlinearen Crashsimulationen die Erstellung der Approximationen sehr aufwendig sein kann. Dadurch, dass Crashverhalten nicht durch glatte Funktionen dargestellt werden kann, entstehen vor allem lokal große Abweichungen zwischen Metamodell und Simulation (DUDDECK & WEHRLE 2015). Außerdem sind Crashsimulationen kosten- und zeitintensiv, wodurch der Entwurfsraum durch wenige Stützstellen abgebildet werden muss. Anhand dieser wenigen Datenpunkte ist eine Validierung des Metamodells nicht möglich (SCHUMACHER & ORTMANN 2015).

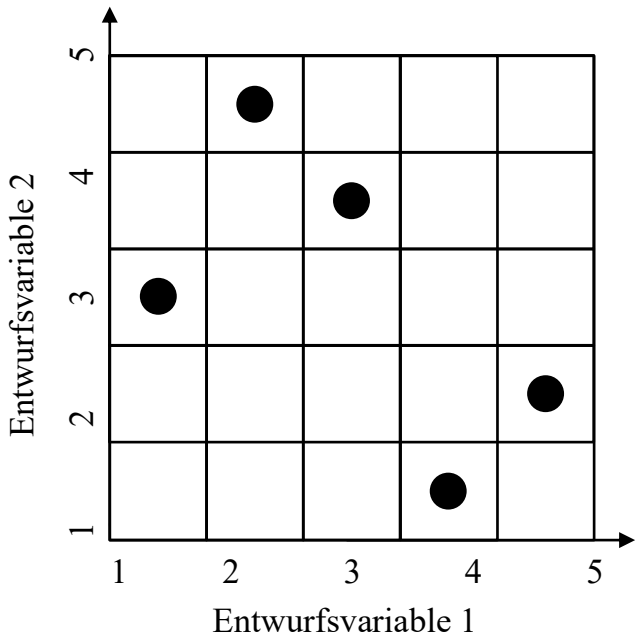


Bild 7: Veranschaulichung des Latin Hypercube Versuchsplans mit zwei Entwurfsvariablen

Neben den klassischen Metamodellen gibt es die Bestrebung *Reduced Order Modelling* für Crashsimulationen anzuwenden. Dabei müssen die Daten aus den Crashsimulationen so aufbereitet werden, dass sie von *Machine-Learning*-Algorithmen verarbeitet werden können (DIEZ 2019, S. 1 ff). Hierzu ist die Reduzierung der Geometrie auf ein- oder zweidimensionale Beschreibungen möglich, sowie die geeignete Darstellung der gewonnen Simulationsdaten (DIEZ 2019, S. 26 ff). Nach

Aufbereitung der Daten können diese von Entscheidungsfindungsalgorithmen verarbeitet werden. Das Ziel dabei ist, Cluster[2] in den Daten zu finden, in denen das Verhalten der Struktur äquivalent ist. Beispielsweise könnte es ein Cluster geben, in dem der Längsträger immer im vorderen Bereich zu beulen beginnt. Gleichermaßen wäre ein Cluster möglich, bei dem immer der hintere Bereich des Längsträgers zuerst beult. Anhand dieser Cluster können zugehörige Kombinationen der Entwurfsvariablen ermittelt werden und bestimmtes Verhalten hervorgerufen oder vermieden werden. (vgl. DIEZ 2019)

2.5 Physikalische Ersatzmodelle

In Ergänzung oder als Alternative zu den mathematischen Ersatzmodellen stehen die physikalischen Ersatzmodelle (vgl. DUDDECK & WEHRLE 2015). Hier steht die Vereinfachung des FE-Modells im Vordergrund. Dazu werden Änderungen des *Inputdecks* vorgenommen.

2.5.1 Mögliche Vereinfachungen des physikalischen Modells

Konkrete Anpassungen des *Inputdecks* können die Variation der Elementgröße oder -art sein. Mit Änderung der Elementart ist gemeint, dass es auch für zweidimensionale Schalenelemente eine Vielzahl verschiedener Konfigurationen gibt. Diese können sich in ihrem Berechnungsaufwand erheblich unterscheiden. So kann z. B. die Anzahl der Integrationspunkte, also Berechnungspunkte, in einem Element variieren. Weitere Informationen zu den Integrationspunkten sind in (WAGNER 2019) zu finden.

Eine weitere Möglichkeit der Vereinfachung der *Inputdecks* liegt in der Entfernung von Teilen des Modells. Die entfernten Teile müssen substituiert werden, damit das Modell seine ursprünglichen Eigenschaften zumindest annähernd beibehält. Im Folgenden werden einige Ansätze zur Reduktion der Modelle und für Randbedingungen (Kurz: RB) zur Wiederherstellung der Eigenschaften des Gesamtmodells im reduzierten Modell dargestellt. Ein reduziertes Modell mit Randbedingungen wird dann als Submodell bezeichnet.

2.5.2 Ersatzmasse angebunden durch lineare Federn

Ein möglicher Ansatz zur Erstellung von Submodellen wurde am Institut für Fahrzeugkonzepte des Deutschen Zentrum für Luft- und Raumfahrt erforscht (SCHÄFFER ET AL.

[2] Als Cluster wird in diesem Abschnitt eine zusammenhängende Gruppe von Datenpunkten in einer Datenmenge bezeichnet. Dagegen beschreibt der Begriff Rechencluster einen Zusammenschluss mehrerer Computereinheiten zu einem Großrechner.

2019a, 2019b). In dem dabei entstandenen Tool *CS-OPT* werden lediglich Lastfälle für den Frontalaufprall berücksichtigt. Es wird der Ansatz verfolgt, das Modell an einer gegebenen Koordinate zu teilen. Der vordere Teil des Fahrzeugs soll als Submodell verbleiben, während der Hintere durch Massen und lineare Federn modelliert wird.

Der Ablauf dieser Submodellerstellung ist in Bild 8 dargestellt und wird folgend erklärt: Zuerst wird an der Schnittfläche ein Raster definiert, damit dieses unabhängig vom Modell anwendbar ist. Für jedes Rasterelement, das einen Bereich der Modellgeometrie schneidet, wird mithilfe einer gesamten Fahrzeugsimulation ein maximaler Crashimpuls ermittelt. Dazu wird der Kraftverlauf im Gitterelement über die Zeit ausgegeben und über die Zeit integriert. Es wird hierbei nur der Impuls senkrecht zur Schnittfläche (normalerweise die x-Richtung) berücksichtigt. Zusätzlich erfolgt eine statische Simulation zur Ermittlung der Steifigkeit des Hinterwagens. Bei dieser statischen Simulation wird eine Kraft von 1000 Newton auf die Stirnfläche der Schnittebene aufgebracht. Die Knoten der Stirnfläche sind durch einen *Nodal Rigid Body* verbunden. Im Schwerpunkt des entfernten Hinterwagens wird ein Massepunkt erzeugt, der die gesamte Masse dieses Fahrzeugteils beinhaltet. Die Trägheit wird dabei nicht weiter berücksichtigt. An Schwerpunkten der einzelnen Gitterelemente werden Federn angebracht, deren Steifigkeiten sich aus der Steifigkeit des Hinterwagens, gewichtet mit den jeweiligen Crashimpulsen ergeben. Die Federn sind mit dem Schwerpunkt des Hinterwagens verbunden. Zusätzlich wird ein Punkt in der Mitte der hinteren Achse erzeugt. Zwischen diesem Punkt und dem Schwerpunkt wird eine Verbindung eingefügt. Durch eine letzte Verbindung zwischen Achsmittelpunkt und Submodell wird die Rotation gesperrt. Bild 9 zeigt das Ergebnis dieses Verfahrens. (vgl. SCHÄFFER ET AL. 2019a, 2019b)

2.5.3 Substitution von Bauteilen durch nichtlineare Federelemente

Bei der Firma Altair wird an Möglichkeiten der Reduzierung von Rechenmodellen geforscht. Dabei wurde eine Methode entwickelt, welche die Fahrzeugstruktur aus besonderen eindimensionalen Elementen darstellt (vgl. MORTISHED ET AL. 2018). Dazu werden die hohlen Blechstrukturen, wie sie z. B. beim Längsträger vorkommen durch *Crushable Frame Springs* (Kurz: CFS) ersetzt. Diese eindimensionalen Elemente können sowohl den unverformten, als auch plastisch verformten Zustand eines Balkens darstellen. Damit CFS verwendet werden können, müssen Materialkarten erstellt werden, welche das Element charakterisieren. Eine initiale Simulation wird zur Ermittlung der Materialkarten benötigt. Dazu wird ein neues Rechenmodell erzeugt, dass auf einer Seite eingespannt und von einer rigid wall auf der anderen Seite belastet

ist. Das Rechenmodell wird aus dem Querschnitt des zu ersetzenden Bauteils extrudiert. Es sind ähnliche Beschleunigungen wie beim Crash einzuhalten. Aus dem Verformungsverhalten kann die Materialkarte für eine CFS erzeugt werden. (vgl. MORTISHED ET AL. 2018)

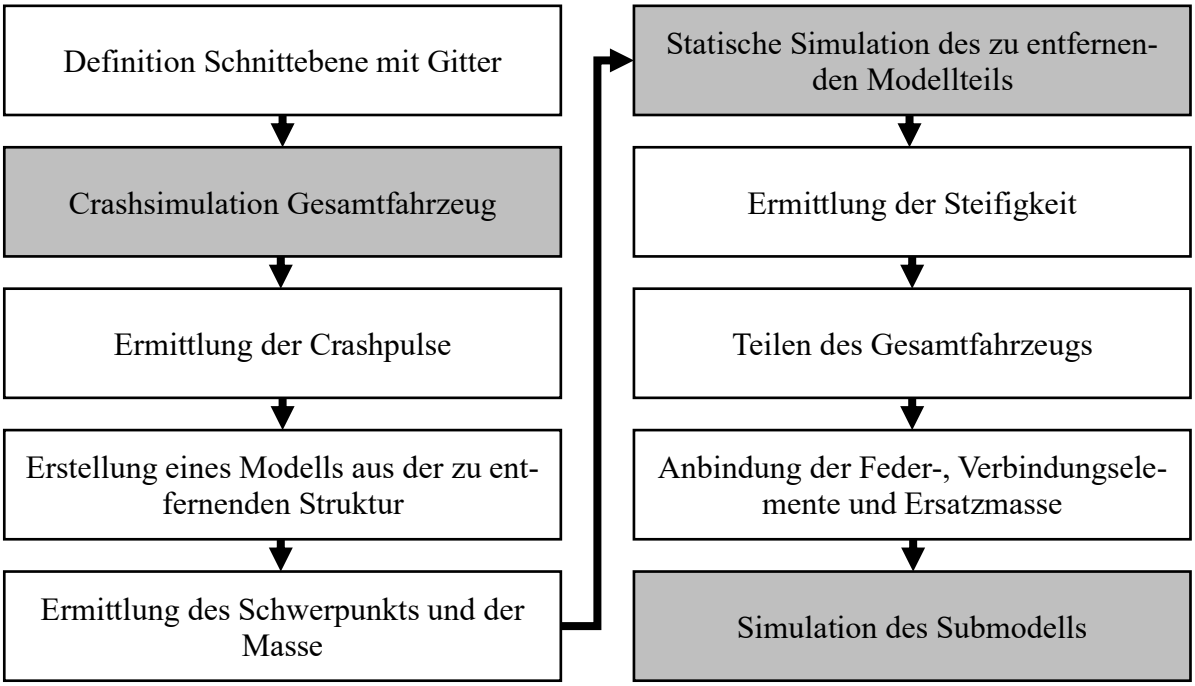


Bild 8: Ablaufplan zur Submodellerstellung mit *CS-OPT*, nach (SCHÄFFER ET AL. 2019a).

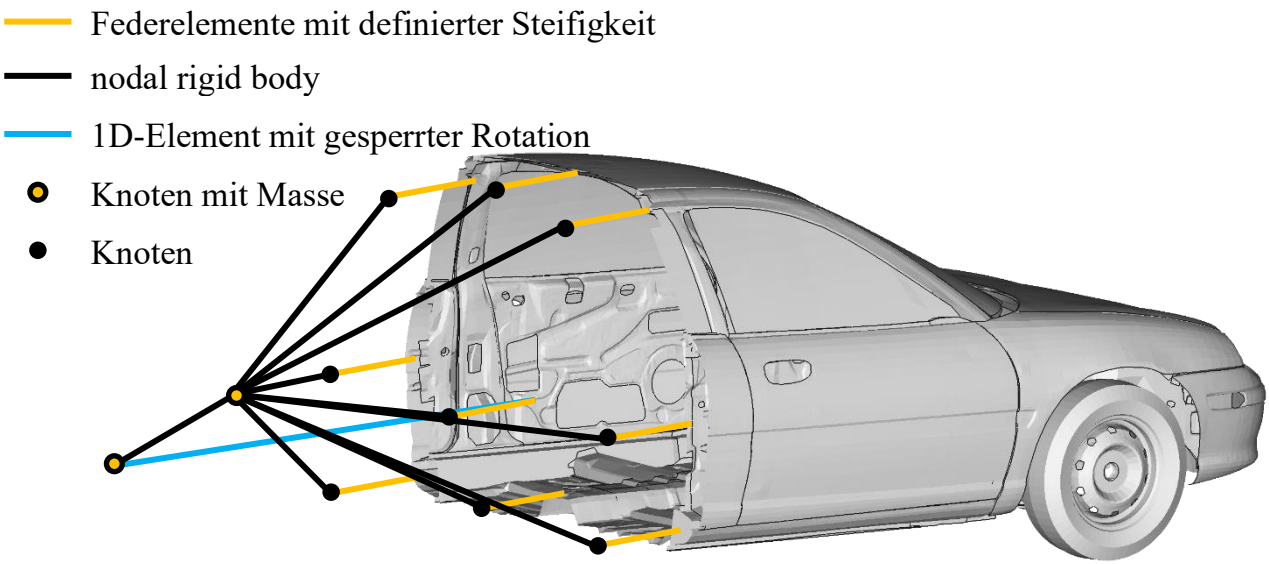


Bild 9: Schematische Darstellung eines reduzierten Fahrzeugmodells mit linearen Federn, nach (SCHÄFFER ET AL. 2019a).

Die Erstellung der Materialkarte ist der Nachteil an diesem Verfahren. Es ist der Aufbau eines Lastfalls und die Simulation für jedes einzelne Bauteil notwendig, dass ersetzt werden soll. Bei einem manuellen Aufbau eines solchen Modells mit Lastfall, wäre dieses Verfahren nicht für eine Optimierung einsetzbar. Bis zu einem gewissen Grad kann das Verfahren jedoch automatisch ablaufen. Voraussetzung hierzu ist, dass der Querschnitt der zu ersetzenden Komponente als einzelne Datei an den Prozess übergeben werden kann (vgl. MORTISHED ET AL. 2018). Für Bauteile mit veränderlichem Querschnitt müssten mehrere CFS, basierend auf den Querschnitten, in Reihe aufgebaut werden.

Dieser Ablauf wird von Altair im *C123*-Prozess genutzt. Dabei wird eine vorhandene Geometrie durch die CFS dargestellt. Anhand des vereinfachten Modells können verschiedene Optimierungen durchgeführt werden. Das Ergebnisstruktur des Prozesses wird auskonstruiert.

2.5.4 Verschiebungsrandbedingungen bei evaluationsabhängiger Modellreduktion

Eine weitere Entwicklung im Bereich der Submodelltechnik fand im Zuge des *eEgO*-Projekts[3] statt. Daraus entstand das in (WALSER ET AL. 2018) vorgestellte *SML*-Tool[4]. In *eEgO* wurde an der automatischen Submodellgenerierung speziell für die Multi-Level-Optimierung geforscht. In dieser Dissertation vorgestellte Methoden basieren zu Teilen auf den Ergebnissen des Projekts und den im *SML*-Tool umgesetzten Algorithmen, was in den nachfolgenden Kapiteln detailliert dargestellt wird.

Im Gegensatz zum Submodelltool *CS-OPT* (SCHÄFFER ET AL. 2019a) findet die Abgrenzung des reduzierten Modells hier nicht pro Element, sondern pro Bauteil statt. Es wird eine Simulation des gesamten Fahrzeugs benötigt, aus der die Wichtigkeit eines jeden Bauteils zu ermitteln ist. Dazu ist zuerst die Evaluationsfunktion festzulegen, in der jede aus der Simulation auswertbare skalare Größe verarbeitbar ist. Als Beispiel ist die interne Energie zu nennen. Mithilfe der Funktion werden die Bauteile gefiltert. Das bedeutet, Bauteile deren Evaluationswert über einem Grenzwert (*Threshold*) liegen, werden dem Submodell zugeordnet, die anderen als zu löschende Teile gespeichert. Die Evaluationsergebnisse werden für den *Threshold*-Grenzwert auf einen Bereich zwischen 0 und 1 normiert. Damit große, jedoch gering belastete Bauteile gegenüber

[3] eEgO: „Entwicklung von Softwaremethoden zur effizienten Ersatzmodell gestützten Optimierung für die Crashauslegung im Fahrzeugentwicklungsprozess“. Online: https://asc-s.de/projekte/eego (Zuletzt aufgerufen am: 29.12.2021)
[4] SML: „Submodel-based Multi-Level optimization“.

kleinen Bauteilen nicht bevorzugt werden, findet in der Evaluationsfunktion eine Mittelung statt.

Das bedeutet die auszuwertende Strukturgröße $S_{i,n,t}$ wird über das Bauteil i und die Konten n in einem Bauteil räumlich und über die Zeitschritte t zeitlich gemittelt. Für jedes einzelne Bauteil i ist die Evaluationsfunktion E_i zu ermitteln (vgl. WALSER ET AL. 2018):

$$E_i = \frac{\sum_t \left(\frac{\sum_n S_{i,n,t}}{n} \right)}{t} \tag{1}$$

Nachdem alle Bauteile des Modells einer Gruppe (Submodell oder Löschen) zugeordnet sind, wird diese Zuordnung durch einen weiteren Algorithmus geprüft. Es wird hinsichtlich der Verbindungen zwischen den Bauteilen im Submodell geprüft. Dadurch, dass nur noch ein Teil der Bauteile zum Submodell gehört, ist es möglich, dass sogenannte Inseln entstanden sind. Eine Insel besteht aus einer oder mehrerer zusammenhängender Bauteile, die aber unabhängig von anderen Inseln existieren. Dabei gibt es eine Hauptinsel, welche die höchste Summe der Evaluationswerte enthält und die restlichen sekundären Inseln. Mithilfe des *Connecting Island*-Algorithmus (Kurz: CI) werden die Inseln miteinander verbunden oder gelöscht. Der Algorithmus sucht in den zu löschenden Bauteilen Verbindungen zwischen Inseln, die einen möglichst hohen Evaluationswert haben (siehe Bild 10a). Hierbei gibt es einen Grenzwert für den CI-Algorithmus, der entscheidet, ob der Evaluationswert für die Verbindung ausreichend hoch ist. Liegt keine Verbindung zu einer Insel oberhalb des Grenzwertes, wird die Insel gelöscht (siehe Bild 10b). Die übrigen Inseln werden durch den Algorithmus miteinander verbunden, dazu sind zusätzliche Bauteile in das Modell aufzunehmen (siehe Bild 10c). Der CI-Grenzwert kann ebenfalls zwischen 0 und 1 liegen, wobei bei 0 alle sekundären Inseln verbunden und bei 1 gelöscht werden. (vgl. FALCONI D. ET AL. 2018)

Sind alle Bauteile zugeordnet und die Inseln miteinander verbunden kann das Submodell erstellt werden. Die Bauteile werden entsprechend ihrer Zuordnung gelöscht und ein rechenfähiges Modell erzeugt. Damit dieses Modell wieder der Kinematik des Gesamtmodells entspricht, sind Randbedingungen aufzuprägen. Zuerst sind dazu die Knoten zu finden, an denen zuvor Bauteile verbunden waren, die jetzt gelöscht sind. Ein weiterer Algorithmus findet alle Knoten mit gelöschten Verbindungen und speichert diese als sogenannte *Interface Nodes*. Aus einer Simulation des Gesamtmodells

werden die Verschiebungen über die Zeit für alle *Interface Nodes* extrahiert. Die extrahierten Daten können dem Submodell als Randbedingungen mitgegeben werden. So wird der Einfluss der entfernten Bauteile ausgeglichen.

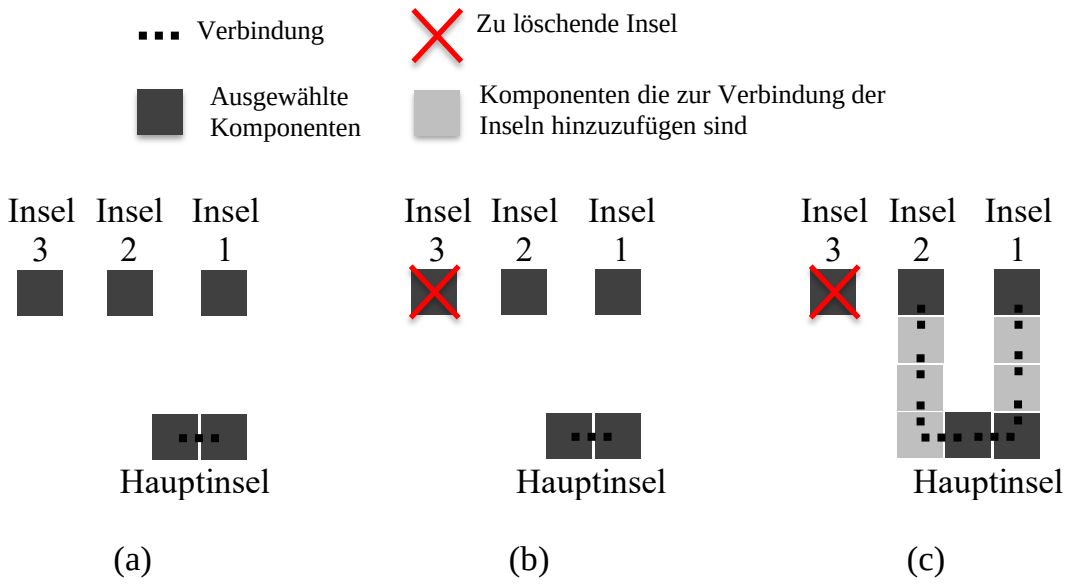


Bild 10: Vorgehen des *Connecting-Island*-Algorithmus, nach (BENITO CIA ET AL. 2020): a) Verbindungen zwischen Inseln suchen; b) Löschen zu weit entfernter Inseln; c) Verbinden der übrigen Inseln

Durch die vorgegebenen Verschiebungen ist eine Veränderung der Fahrzeugkinematik nur in sehr geringem Maße möglich. Insbesondere durch *Interface Nodes* in der Crashzone ist das Submodell stark eingeschränkt und kann nur geringfügig auf Strukturveränderungen, z. B. während einer Optimierung, reagieren.

2.6 Aufbau der Optimierungsschleifen mit Submodellen

In den in Abschnitt 2.3 dargestellten Optimierungsablauf werden die zuvor beschriebenen Submodelle integriert, wie auch in (SINGH ET AL. 2017; BENITO CIA ET AL. 2020; WALSER ET AL. 2018; SCHUMACHER ET AL. 2019) dargestellt wird. Hier gibt es zwei grundlegende Vorgehensweise bei der Verwendung eines Submodells in der Strukturoptimierung. Eine Möglichkeit besteht darin, aus einem Gesamtfahrzeugmodell ein Submodell zu erzeugen und damit einen DoE durchzuführen. Auf den Ergebnissen des DoE können Metamodelle trainiert und damit das Optimum bestimmt werden. Im Folgenden wird auf die direkte Verwendung des Submodells in der Optimierungsschleife eingegangen. Dabei wird das Submodell vor oder während der Optimierung generiert und für die Funktionsaufrufe des Analysemodells verwendet. Einen Sonderfall bildet die Optimierung in mehreren Ebenen oder auch Multi-Level Optimierung genannt.

Diese Ebenen sind hierarchisch geordnet und voneinander abhängig (vgl. SINGH ET AL. 2017). Hierdurch können verschiedene Analysemodelle und somit auch verschiedene Submodelle innerhalb einer Optimierung verwendet werden.

2.6.1 Direkte Optimierung

Bei der direkten Optimierung wird eine Optimierungsschleife, wie in Bild 5 genutzt. Als Analysemodell dient dann das Submodell. Da das Submodell automatisch generiert wird, kann der Prozess der Erzeugung in das Diagramm für die Optimierungsschleife aufgenommen werden. In Bild 11 ist der Ablauf für den Fall mit einem Submodell und einer Ebene dargestellt. Diese Optimierungsschleife kann auch als *submodellbasierte Single-Level Optimierung* bezeichnet werden.

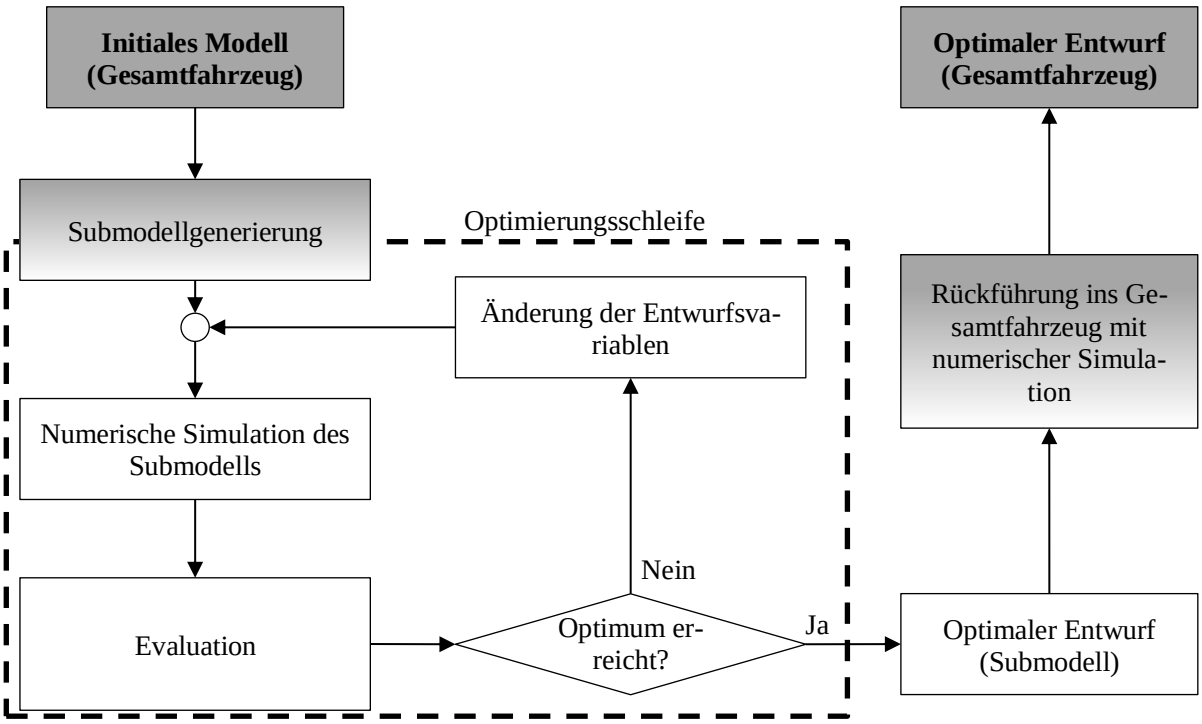


Bild 11: Ablaufplan einer submodellbasierten Single-Level Optimierung. Grauer Hintergrund: Verwendung des Gesamtmodells; Weißer Hintergrund: Verwendung des Submodells

Wichtig bei dieser Art der Optimierungsschleife ist die Rückführung des optimalen Submodellentwurfs in das Gesamtmodell. Die Entwurfsvariablen müssen auf das Gesamtfahrzeug angewendet und das Optimum durch eine Simulation bestätigt werden. Hier besteht das Risiko, dass die Simulationsergebnisse von Sub- und Gesamtmodell im Optimum stark variieren. Das Risiko kann durch die Verwendung eines validierten Submodells und Voruntersuchungen zum Verhalten des Modells im Entwurfsraum reduziert werden.

2.6.2 Optimierung in mehreren Ebenen

Eine Erweiterung der Optimierung in einer Ebene stellt die Optimierung in mehreren Ebenen dar, die Multi-Level Optimierung. Dazu werden mehrere Optimierungsschleifen ineinander verschachtelt. Jede Optimierungsschleife kann eigene Zielfunktionen, Restriktionen, Optimierungsalgorithmen, Simulationsmodelle etc. verwenden. Dadurch bietet dieses Verfahren ein hohes Maß an Flexibilität. Die Herausforderung besteht darin, dass die „Detaillierungsebenen […] miteinander verzahnt werden [müssen]“ (SCHUMACHER 2020, S. 170). Damit ist gemeint, dass die Veränderungen innerhalb einer Schleife Einfluss auf andere Schleifen haben können. Es ist somit unerlässlich, dass die Ergebnisse der verschiedenen Optimierungsschleifen aus den unteren Ebenen zu einem Modell in der Hauptebene zurückzuführen sind.

Ein Beispiel für eine Multi-Level Optimierung eines Fahrzeugs mit drei verschachtelten Ebenen, wobei die beiden Unterebenen parallel zueinander angeordnet sind, könnte wie folgt aussehen:

- Hauptebene: Gesamtes Fahrzeugmodell, Optimierung des globalen Designkonzepts
- Unterebene 1: Modell des Vorderwagens, Formoptimierung der Knautschzone bei einem Frontalcrash
- Unterebene 2: Modell der Seitenschweller und des Innenraums, Formoptimierung hinsichtlich des Insassenschutz bei einem Pfahlaufprall

In diesem Beispiel beziehen sich die einzelnen Ebenen auf den Fahrzeugcrash, jedoch mit unterschiedlichen Lastfällen, Modellen und Zielen. Aufgrund der Abhängigkeiten zwischen den Ebenen ist ein Austausch von Informationen notwendig. So sind z. B. Entwurfsvariablen zwischen den einzelnen Ebenen zu aktualisieren, damit eine Optimierung mit verzahnten Ebenen, anstelle einzelner Optimierungen entsteht.

Wie die Aufzählung oben bereits zeigt, ist es sinnvoll in den einzelnen Ebenen entsprechende Submodelle zu verwenden, dadurch wird es zu einer *submodellbasierten Multi-Level Optimierung*. Die Submodelle müssen durch die Verzahnung der Ebenen bei jeder Iteration der darüberliegenden Ebene neu generiert oder aktualisiert werden. Für zwei Ebenen ist ein solcher Ablauf beispielhaft in Bild 12 dargestellt.

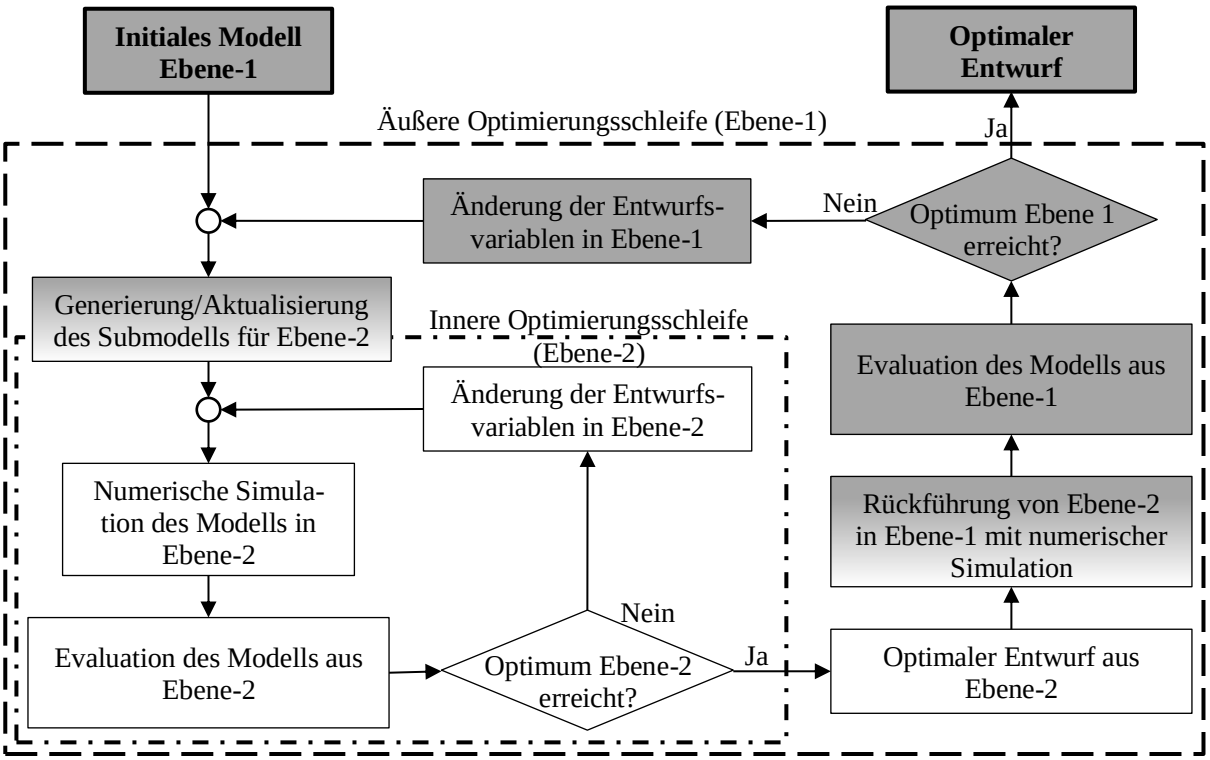


Bild 12: Ablaufplan einer submodellbasierten Multi-Level Optimierung mit zwei Ebenen. Grauer Hintergrund: Verwendung des Gesamtmodells (Ebene-1); Weißer Hintergrund: Verwendung des Submodells (Ebene-2)

Die *submodellbasierte Single-Level Optimierung* ist als zweite Ebene innerhalb der Multi-Level Optimierung erkennbar. Für die innere Schleife ist es besonders wichtig, ein kleines Modell zu verwenden, da die Optimierung der zweiten Ebene bei jeder Iteration der übergeordneten Ebene durchgeführt wird. Somit führen die sekundären Ebenen viel mehr Funktionsaufrufe aus als die Hauptebene.

Das dargestellte Beispiel aus Bild 12 ist nahezu beliebig erweiterbar. Es können entweder parallel zur dargestellten Schleife in Ebene-2 noch weitere Optimierungen in derselben Ebene durchgeführt werden, oder eine weitere Hierarchieebene hinzugefügt werden. Die zusätzliche Hierarchieebene würde dann aus der inneren Optimierungsschleife gestartet werden.

Es bleibt die Frage zu beantworten, welchen Vorteil die Aufteilung eines Optimierungsproblems auf mehrere Ebenen mit sich bringt. Wie bereits genannt, ist die Flexibilität ein Argument dafür. In jeder Optimierungsschleife können eigene Zielfunktionen, Restriktionen, Lastfälle etc. definiert werden. Diese sollten jedoch das gleiche Hauptziel anstreben, da sich sonst die Ebenen gegenseitig behindern. Wichtig ist, dass das Modell abhängig von den übergeordneten Ebenen ist. Hierdurch können Entwurfsvariablen in einzelnen Optimierungsschleifen gruppiert werden und mehr

Funktionsaufrufe mit kleineren Modellen ausgeführt werden. Bei einer Optimierung in einer Ebene würden alle Entwurfsvariablen in einem Modell sein. Dies erfordert viele Funktionsaufrufe mit dem gesamten Modell. Durch die Aufteilung auf mehrere hierarchisch abhängige Ebenen, würde für jeden Entwurf der äußersten Schleife durch die inneren Schleifen das Optimum gefunden werden.

3. Aufbau von Submodellen

Es wurden nun einige bereits existierende Möglichkeiten zur Reduktion des Ressourcenbedarfs vorgestellt. Diese haben gemein, dass einem Reduktionsverfahren eine Art von Randbedingung zugeordnet ist. Die Abläufe lassen keine Variation der Verfahren zu, das heißt Submodell und Randbedingung sind fest miteinander verknüpft. Eine andere Kombination aus Submodell und Randbedingung ist nicht zulässig.

In diesem Kapitel wird ein umfassendes Konzept zum Aufbau von Submodellen beschrieben. Die Investitionen basieren insbesondere auf dem in Abschnitt 2.5.4 vorgestellten *SML*-Tool (WALSER ET AL. 2018). Das *SML*-Tool wird durch zusätzliche Optionen und Alternativen für die Randbedingungen und Submodelltechniken ergänzt. In der mit dieser Dissertation entstandenen Software *ASCO*[5] (siehe auch BENITO CIA ET AL. 2020) werden die einzelnen Module der Submodellgenerierung, Randbedingungen und Nutzung in einem Programm vereint. Allerdings sind die einzelnen Module getrennt voneinander ausführbar und damit beliebig zu kombinieren. Das bedeutet, Submodellgenerierung und Erstellung der Randbedingungen sind unabhängige Aufgaben, welche miteinander kombiniert werden können.

Bevor die neuen Verfahren erklärt werden, erfolgt ein Überblick über die Anforderungen an ein reduziertes Modell. Diese Anforderungen sollte im Idealfalle jedes reduzierte Modell erfüllen. Danach werden erst die einzelnen Verfahren zur geometrischen Submodellgenerierung vorgestellt. Es folgen die Arten der Randbedingungen der geometrischen Submodelle und ein Abschnitt zur Validierung von Submodellen. Zum Abschluss des Kapitels wird ein Überblick über den Workflow und die Kombinationsmöglichkeiten aufgezeigt.

3.1 Anforderungen an die Submodelle

Beim Aufbau von Submodellen sind folgende Kriterien zu beachten:

- Mögliche Einsparung der Rechenzeit,
- ausreichende Genauigkeit im Bezug zum originalen Modell,
- Automatisierungsgrad,
- Rechenfähigkeit des *Inputdecks* und
- Zeitaufwand für die Modellerstellung.

Es kann kein konkreter Wert für die notwendige Rechenzeiteinsparung genannt werden, damit sich ein Ansatz lohnt. Wie gut das jeweilige Submodell ist, ist dann vom

[5] ASCO: “Automatic submodel generation for crash optimization”

konkreten Fall abhängig und auch ob die Einsparung an Ressourcen einen Verlust an Genauigkeit rechtfertigt. Trotzdem können Ziele festgelegt werden, die ein Submodell erreichen sollte. So ist die Reduktion der Rechenzeit bzw. des Ressourcenbedarfs durch das Submodell, um die Hälfte zum originalen Modell ein definierter Meilenstein.

Damit der Aufwand für die Simulation um die Hälfte sinkt, müssen mindestens 50 % der Elemente aus dem Modell entfernt werden. Eine solche Veränderung des Modells führt unweigerlich zu Änderungen im Modellverhalten, welche durch geeignete Randbedingungen auszugleichen sind. Elementar ist hier die Überprüfung der Validität, also die Übereinstimmung der Eigenschaften des Submodells mit dem Gesamtmodell. Dazu sind sowohl kinematische Aspekte als auch Energien und Verformungen zwischen Submodell und Gesamtfahrzeug zu untersuchen. Ziel ist eine möglichst gute Übereinstimmung zwischen den beiden Modellen. Dabei ist wichtig, dass das Submodell im gesamten Entwurfsraum ausreichend genau ist.

Der Programmcode muss vollständig automatisch ablaufen. Dabei muss das originale *Inputdeck* eingelesen, angepasst und als neues Modell exportiert werden. Das neue *Inputdeck* des Submodells muss rechenfähig sein, ohne Fehler bei der Simulation hervorzurufen. Aufgrund der Variationen die beim Aufbau eines Modells möglich sind, folgt hieraus eine große Herausforderung. Neben der erforderlichen Flexibilität des Programms führen auch neue Entwicklungen bei den Simulationsmodellen, z. B. neue *Keywords* (siehe Abschnitt 2.2) und neue Programmversionen zu den hohen Anforderungen an den Programmcode zur Erstellung der Submodelle.

3.2 Geometrische Reduktion des Modells

Als erster Schritt bei der Erstellung von Submodellen steht die geometrische Reduktion des Modells. Dies beinhaltet die Auswahl der Bauteile oder Elemente, die aus dem Modell zu entfernen sind und die Aufbereitung des *Inputdecks* zu einem rechenfähigen Submodell.

3.2.1 Notwendige Maßnahmen

Bei der Erstellung von Submodellen ergeben sich einige Herausforderungen. Zum einen bestehen Schwierigkeiten in der Flexibilität des Programmcodes, da der Prozess automatisch ablaufen soll, die Modelldaten jedoch sehr stark in Aufbau und Form variieren können. Auf der anderen Seite ist die Auswahl der zu löschenden Teile die Kernherausforderung.

Die Probleme liegen insbesondere in der möglichen Vielfältigkeit der *Modellinputdecks*. Neben verschiedenen Möglichkeiten der Syntax und unterschiedlichen Programmversionen, mit anderen unterstützten *Keywords*, gibt es unzählige Varianten ein Modell aufzubauen. Ein Beispiel ist in Bild 13 gegeben, dort ist links ein Modell in einer einzelnen Datei gespeichert. Rechts hingegen wird das Modell auf mehrere Dateien aufgeteilt. Dabei werden die einzelnen Dateien, über das Keyword `INCLUDE` in einer Hauptdatei eingebunden. Wichtig hierbei ist, dass keine Dubletten vorhanden sind, das heißt keine Komponenten, Elemente etc. mit der gleichen ID. Werden zwei Dateien mit einem gleichen Nummerierungsbereich zusammengeführt, ist die Nummerierung einer der Dateien zu ändern. Beim Inkludieren der Datei ist dazu eine Option vorhanden, ein Offset der Nummerierung. Durch dieses Offset ändert sich auch die Nummerierung eines referenzierten *Keywords*, wie rechts in dem Bild dargestellt: In der Definition des Parts wird das Element 011 referenziert. Durch das Offset von 10 beim Inkludieren der Datei muss aber in der zweiten Datei das Element 021 gesucht werden. Beim Einlesen und Verändern des *Inputdecks* sind solche Phänomene zu berücksichtigen. Besonders aufwendig wird es, wenn einige der inkludierten Dateien verschlüsselt sind und nicht ohne weiteres verändert werden können. Dies ist häufig bei Materialkarten in der Industrie so, da die Erstellung einer solchen Materialkarte viele Versuche und Arbeit benötigt und daher der Geheimhaltung unterliegen. Bei den öffentlich zugänglichen Modellen sind solche verschlüsselten Daten nicht vorhanden, wodurch diese Problematik nicht weiter berücksichtigt wird.

Auch einzelne *Keywords* können auf verschiedenste Weisen aufgebaut werden. So haben die meisten *Keywords* Optionen, die die Reihenfolge der Parameter oder deren Bedeutungen verändern. Zusätzlich sind einige Parameter optional und damit nur unter bestimmten Umständen vorhanden. Die Beispiele in Bild 14 in der LS-DYNA® Syntax zeigen vier Möglichkeiten, eine Gruppe von Bauteilen zu generieren. Dabei werden mit Beispiel (a), (b) und (d) die gleichen Gruppen erreicht, obwohl das vorangestellte *Keyword* sich unterscheidet. Beispiel (c) führt die zwei Gruppen oder Sets aus den ersten Beispielen zusammen in eine dritte Gruppe. Dem Anwender ist dabei klar, dass Beispiel 3 eine andere Anwendung hat, als die ersten beiden. Für die Submodellgenerierung sind jedoch die Beispiele (a) und (b) einfacher zu interpretieren als das Dritte. Im dritten Fall werden andere Gruppen referenziert und es sind deshalb nicht direkt die zugehörigen Bauteile ersichtlich. Das vierte Beispiel ist eine Erweiterung des *Keywords* aus Beispiel (a), wodurch eine zusätzliche Zeile für eine Benennung der Gruppe eingefügt wird. Diese zusätzliche Zeile ist für den Algorithmus wichtig, da

hierdurch die PIDs der Bauteile erst in der dritten Zeile nach dem *Keyword* beginnen, statt der Zweiten.

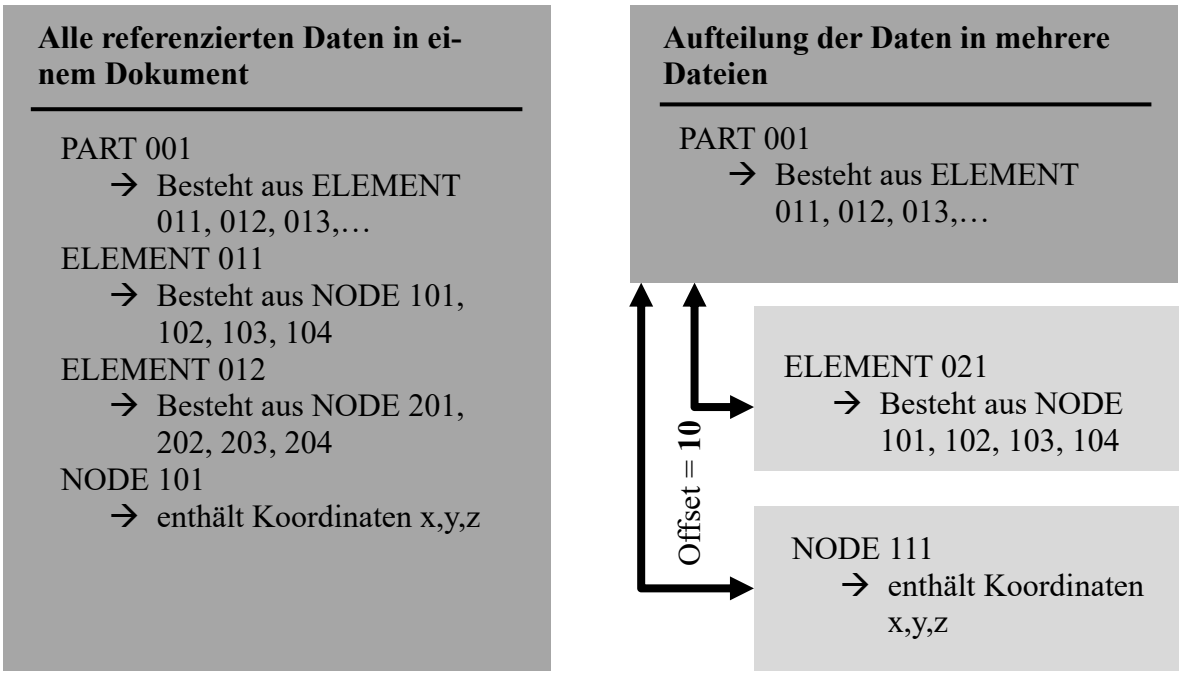


Bild 13: Aufteilung eines Modells in mehrere Dateien mit Verschiebung der Bezeichnungen

Die nächste Herausforderung liegt in der Verbindungstechnik. Im Laufe ihrer Entwicklung bieten FE-*Solver* immer mehr Möglichkeiten, Verbindungen wie Schweißen, Kleben, Schrauben etc. darzustellen. Die einfachste Art einer Verbindung besteht darin, dass zwei Bauteile des Modells sich Knoten teilen. Eine solche Verbindung kann die Eigenschaften einer Schweißverbindung, wie z. B. Materialeigenschaften, Wärmeeinfluss etc. nicht abbilden. Dies gilt neben Schweißverbindungen auch für die anderen Verbindungstechniken. Eine Möglichkeit, wie aktuell Verbindungen modelliert werden, erfolgt durch zusätzliche Elemente. Diese zusätzlichen Volumenelemente beinhalten alle Eigenschaften der Verbindung. Beispielsweise kann so ein einzelner Schweißpunkt oder eine längliche Klebenaht als Reihe mehrerer dieser Elemente modelliert werden. Während der Simulation werden die Verbindungselemente mit den Bauteilen verbunden. In Bild 15 sind in grün Klebeverbindungen und in lila Schweißpunkte dargestellt, die die vorgestellte Verbindungstechnik nutzen. Das *Keyword* des *Solvers* LS-DYNA® für diese Art der Verbindung lautet `CONTACT_TIED_SHELL_EDGE_TO_SURFACE`.

(a)

```
*SET_PART
$#     sid
         1
$#     pid       pid       pid       pid       pid       pid       pid       pid
        10        11        12        13        14        15        16        17
        18        19        20        21        22        23        24        25
```

(b)

```
*SET_PART_LIST
$#     sid
         1
$#     pid       pid       pid       pid       pid       pid       pid       pid
        10        11        12        13        14        15        16        17
        18        19        20        21        22        23        24        25
```

(c)

```
*SET_PART_ADD
$#     sid
         1
$#    psid      psid
         2         3
```

(d)

```
*SET_PART_TITLE
$# title
Beispielgruppe
$#     sid
         1
$#     pid       pid       pid       pid       pid       pid       pid       pid
        10        11        12        13        14        15        16        17
        18        19        20        21        22        23        24        25
```

Bild 14: Vier LS-DYNA® Beispiele zur Erstellung von Bauteilgruppen: a) `SET_PART`; b) `SET_PART_LIST`; c) `SET_PART_ADD`; d) `SET_PART_TITLE`

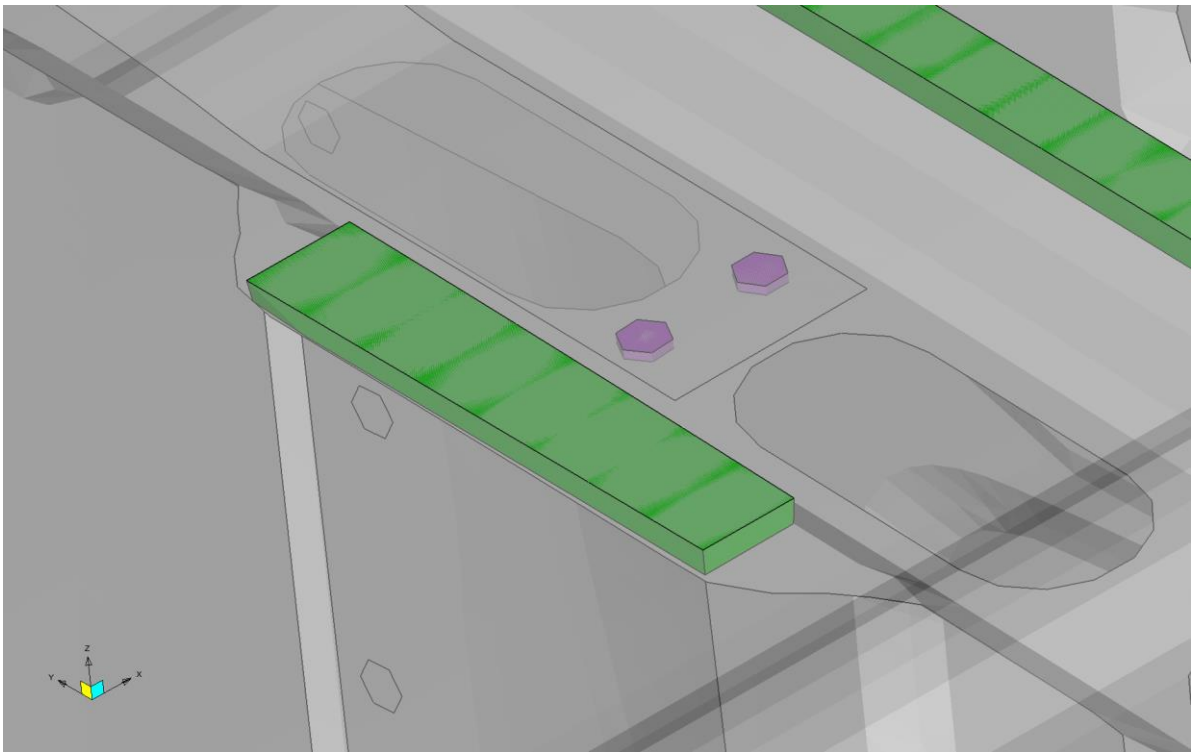


Bild 15: Beispiel für Verbindungstechnik in einem Fahrzeugmodell. Grün: Klebeverbindung; Lila: Schweißpunkt. Modell: Chevrolet Silverado (03/2017) aus (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

Neben der *Tied-Contact*-Technik gibt es diverse weitere Möglichkeiten für Verbindungen, welche sich bei jedem *Solver* unterscheiden. Die Software für die Erstellung der Submodelle muss alle Arten erkennen und bearbeiten können.

Weitere Herausforderungen liegen in der Materialbeschreibung und der Verwendung von Umformdaten. Die Materialkarten, welche die Eigenschaften verwendeter Materialien beschreiben, sind in Unternehmen oftmals verschlüsselt, sodass eine Verarbeitung der Daten in der Software nicht möglich ist. Wichtig ist, dass der Algorithmus die Materialkarten unberührt lässt, sodass die Verschlüsselung intakt bleibt. In aktuellen Modellen werden Umformdaten, z. B. aus einer Tiefziehsimulation verwendet, um die Blechdicken eines Bauteils für jedes einzelne Element zu bestimmen. Wird ein Bauteil während der Submodellerstellung geteilt, sind auch die Umformdaten zu aktualisieren. Diese liegen jedoch in extra Dateien vor, auf welche nicht immer zugegriffen werden kann. Aufgrund fehlender öffentlich zugänglicher Simulationsmodelle mit solchen Daten kann keine weitere Auskunft zu dieser Problematik gegeben werden.

3.2.2 Modellreduktion abhängig von Evaluationsfunktion

Die Reduktion abhängig von einer Evaluationsfunktion wurde bereits in Abschnitt 2.5.4 beschrieben. Sie stellt den Kern des *SML*-Tools dar. Dort ist die Modellreduktion ein Teil des gesamten Prozesses und die Trennung zwischen Modellerstellung und den Randbedingungen nicht ohne weiteres möglich.

Unter dem Namen *Performance based Submodel Generation* (Kurz: PBS) werden die Algorithmen des *SML*-Tools in *ASCO* verwendet. Die prinzipiellen Abläufe bestehen weiterhin. Zu Beginn wird eine Simulation des gesamten Fahrzeugmodells ausgeführt, anhand derer alle Bauteile evaluiert werden. Als Strukturantwort für die Evaluationsfunktion kann eine auswertbare skalare Größe verwendet werden. Gängige Strukturantworten sind die interne Energie, plastische Dehnung oder Spannung. Für jedes Bauteil des Fahrzeugmodells wird die Größe der Strukturantwort ermittelt und über die Zeitschritte und Anzahl der Elemente gemittelt, um so den Evaluationswert zu erhalten (siehe Gl. (1) in Abschnitt 2.5.4).

Für die Entscheidung, ob ein Bauteil dem Submodell zugeordnet wird oder nicht, werden alle Evaluationswerte zwischen 0 und 1 normiert. Auf der daraus entstandenen Liste basierend kann das Submodell generiert werden. Hierbei stehen zwei mögliche Verfahren zur Auswahl. Entweder werden die Parameter *Threshold* und *CI* aus Abschnitt 2.5.4 oder eine relative Größe des Submodells verwendet.

Der Parameter *Threshold* legt die Grenze der Evaluierungswerte fest, ab wann ein Bauteil dem Submodell zugeordnet wird. Beim *CI*-Grenzwert handelt es sich um die Grenze, ab wann eine sekundäre Insel nicht mehr mit der Hauptinsel zu verbinden ist. Der CI-Algorithmus sucht die Verbindung mit dem höchsten Evaluierungswert zwischen zwei Inseln. (vgl. WALSER ET AL. 2018)

Bei der Verwendung der relativen Größe eines Modells werden aus der Liste der evaluierten Bauteile solange Bauteile zum Submodell hinzugefügt, bis mindestens die gewünschte Größe erreicht ist. Als Größe ist dabei das Verhältnis der Anzahl an Elementen definiert. Bei einer relativen Größe von 0,5 sind mindestens 50% der Elemente des Gesamtfahrzeugs im Submodell. Anschließend wird auch hier der CI-Algorithmus angewendet, mit einem Grenzwert von 0, sodass alle Inseln verbunden werden. Hierdurch kann die Größe des Submodells noch einmal leicht variieren.

Bevor das Submodell erstellt wird, besteht die Möglichkeit, Gruppen oder Bauteile auszuwählen, die zwingend dem Submodell zugeordnet werden sollen. Dazu sind diese Sets und Bauteile in der *ASCO*-Konfigurationsdatei anzugeben. Die Algorithmen fügen die angegeben Komponenten dann zwingend dem Submodell hinzu und der CI-Algorithmus verbindet diese. Sinnvoll ist diese Funktion dann, wenn Bauteile für die Auswertung von Strukturantworten erforderlich sind. Ein Beispiel wäre die hintere Stoßstange bei einem Frontalaufprall. Sie nimmt nahezu keine interne Energie auf und würde deswegen nicht im Submodell sein. Häufig werden Knoten auf der Stoßstange und im Hinterwagen ausgewertet, mit denen ein lokales Fahrzeugkoordinatensystem erzeugt wird, welches unabhängig von der Kinematik ist. Anhand dieses Koordinatensystems können Strukturantworten im Fahrzeug ausgewertet werden, ohne dass die Deformation oder ein Ausbrechen des Fahrzeugs einen Einfluss auf die Werte hat. Die drei Knoten stellen dabei den Ursprung und die Vektoren der x- und y-Achse dar. Als Normale zur aufgespannten xy-Ebene im Ursprung ergibt sich die z-Achse. Bild 16 veranschaulicht das lokale Koordinatensystem der drei Knoten für das Koordinatensystem vor und nach einem Crash.

Nach der Erstellung des Submodells werden alle Knoten, an denen sich zuvor Verbindungstechnik befand (siehe Abschnitt 2.5.4) in einer Gruppe als *Interface Nodes* gespeichert. Auf diese werden später die Randbedingungen aufgeprägt. Aus diesem Verfahren ergibt sich ein großer Nachteil für Optimierungen der Karosseriestruktur beim Crash: Durch die Positionierung von *Interface Nodes* im Bereich der Aufprallzone hat die Struktur kaum Möglichkeit, ihr Verhalten zu verändern. Als Randbedingung ist nur die Verschiebung von Randbedingungen sinnvoll nutzbar, da

ein *Rigid-Body-Element* (Kurz: RBE) die Struktur zu sehr versteifen würde, wie in Abschnitt 3.3.4 dargelegt wird. Es besteht keine Möglichkeit die Positionierung der *Interface Nodes* zu steuern, wodurch nahezu immer auch die Aufprallzone betroffen ist.

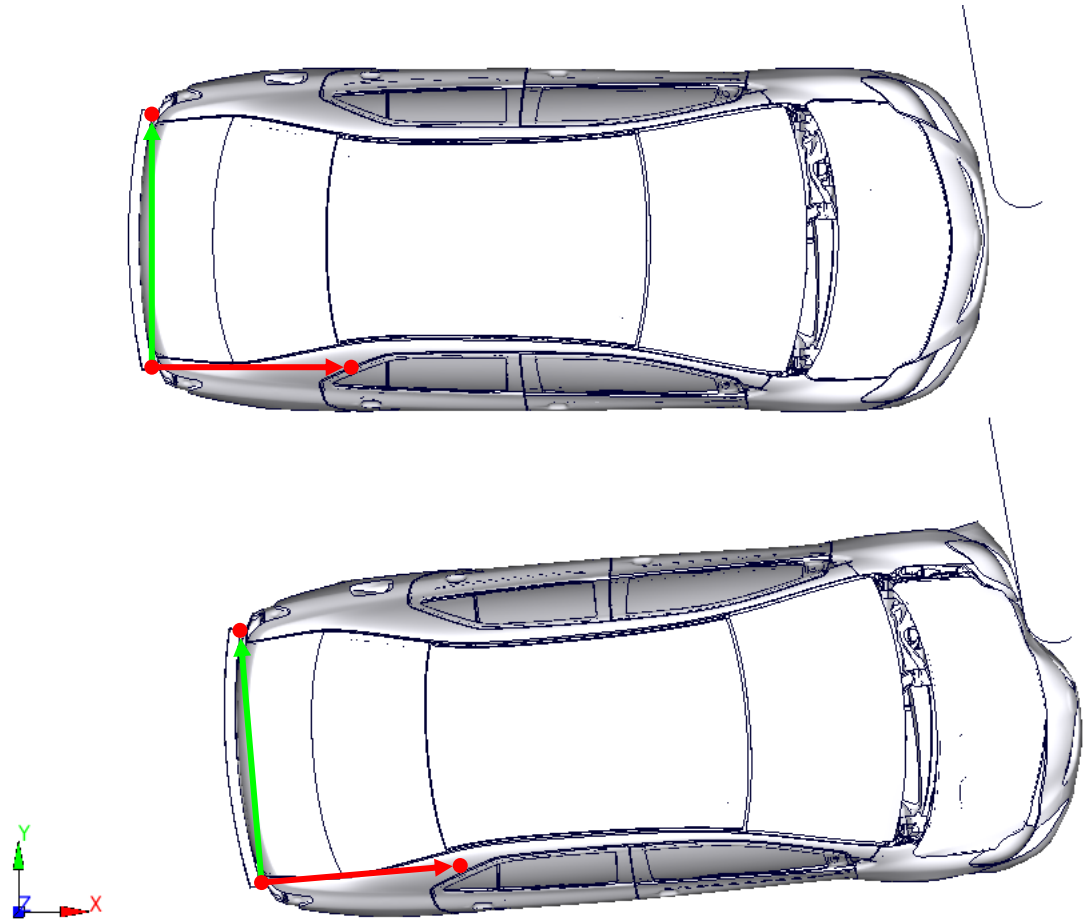


Bild 16: Beispiel eines lokalen Koordinatensystems im Fahrzeugmodell anhand von drei Knoten (rote Punkte) im Hinterwagen. Rot: x-Achse; Grün: y-Achse

Als Vorteile dieses Verfahrens sind die einfache Bedienbarkeit und der hohe Automatisierungsgrad zu sehen. Der Bediener hat im einfachsten Fall lediglich eine relative Submodellgröße einzugeben, um ein geeignetes Submodell zu erhalten. Dazu ist das Verfahren lastfallunabhängig, da die Evaluationsfunktion alle Bauteile gleichbehandelt.

3.2.3 Koordinatenbasierte Modellreduktion

Eine Alternative zum PBS-Ansatz bietet die *Coordinate based Submodel Generation* (Kurz: CBS). Hierbei erfolgt die Auswahl des Submodells rein geometrisch. Durch zwei Punkte im Raum wird eine Box definiert, welche als Auswahlraum dient. Es stehen dann zwei Optionen zur Verfügung: eine Auswahl anhand der Bauteile oder

Elemente. Die Auswahl anhand von Bauteilen belässt jedes Bauteil im Submodell, welches mit mindestens einem Knoten in der definierten Box liegt. Als Alternative kann der Schnitt pro Element durchgeführt werden. Damit werden Elemente, die einen Knoten innerhalb der Box haben ausgewählt. Hierdurch werden Bauteile auf der Grenze zerteilt. Das Beispiel in Bild 17 zeigt eine Box, die durch die beiden grün dargestellten Punkte definiert ist. Das gelbe Bauteil liegt ca. zur Hälfte in der Box und würde bei einer Auswahl pro Bauteil in das Submodell aufgenommen werden. Erfolgt die Auswahl über die Elemente, wird das Bauteil an der orange markierten Schnittkante geteilt.

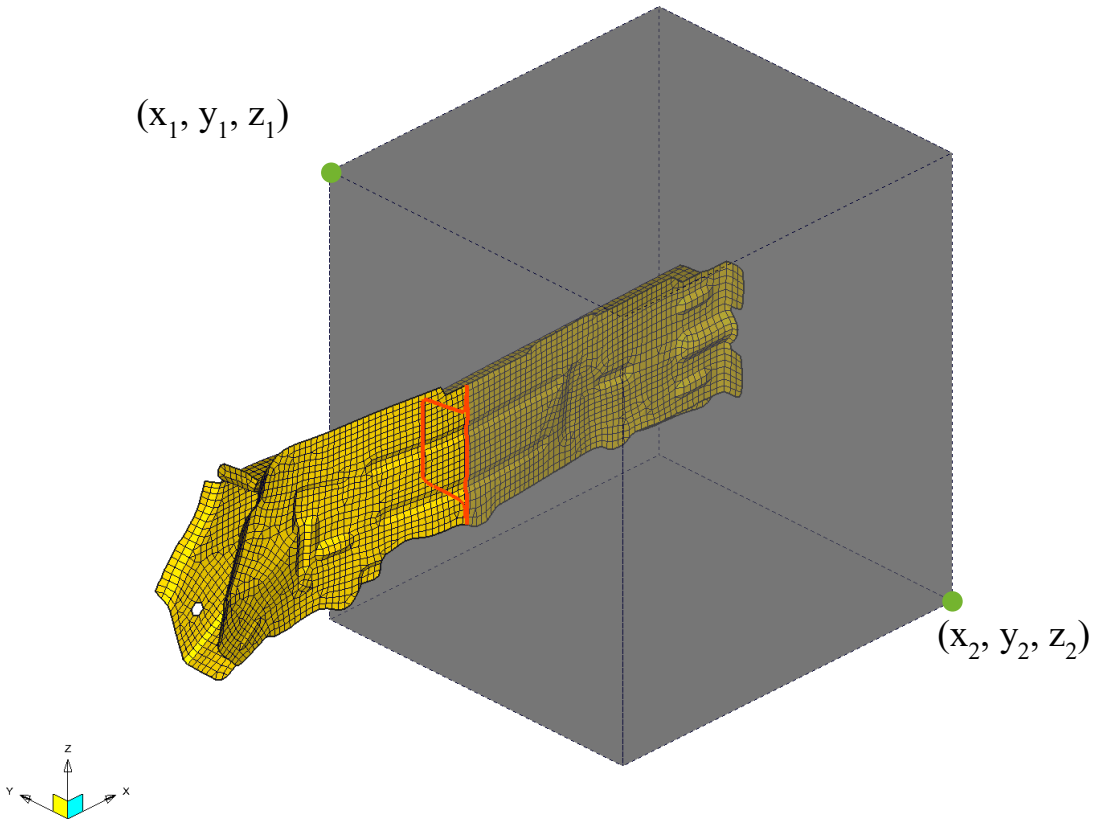


Bild 17: Beispiel für die *Coordinate based Submodel Generation* anhand eines einzelnen Bauteils. Grün: Punkte zur Definition der Auswahlbox; Orange: Schnittkante

Wird die vorgestellte Methode auf ein gesamtes Fahrzeugmodell angewendet, ergeben sich die Modelle aus Bild 18. Die Schnittkante der elementweisen Auswahl ist zur Verdeutlichung rot gekennzeichnet. Mit den vorgestellten Beispielen wird sinnvollerweise ein Frontalaufprall simuliert, da die Box so platziert ist, dass der Vorderwagen als Submodell bleibt. Positioniert werden kann die Box jedoch beliebig, sodass auch für andere Lastfälle Submodelle erstellbar sind.

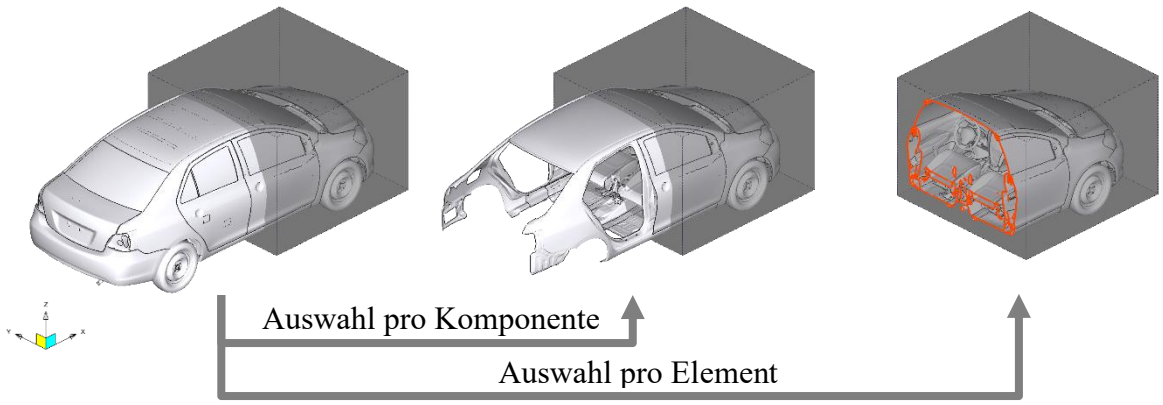


Bild 18: Submodellgenerierung mit der *Coordinate based Submodel Generation*

Es besteht die Möglichkeit eine Gruppe aus Bauteilen dem Submodell zuzuordnen, zusätzlich zu dem in der Box. Beispielsweise kann eine Gruppe bestehend aus dem hinteren Fahrwerk mit Rädern integriert werden. Das ist dann sinnvoll, wenn eine Ersatzmasse als Randbedingung eingesetzt wird und die Kinematik möglichst unverändert sein soll. Dort wo die Gruppe vorher mit dem Fahrzeug verbunden war, werden dann *Interface Nodes* eingefügt. Das in Bild 19 gezeigte Beispiel ist ein Submodell des Toyota Yaris aus (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020), bei dem das hintere Fahrwerk in einem Set zusammengefasst ist und dem Submodell zugeordnet wurde. Die *Interface Nodes* am Fahrwerk und der Schnittkante sind grün hervorgehoben.

Der Vorteil von CBS liegt darin, dass innerhalb der Box alle Knoten vorhanden sind und dort keine *Interface Nodes* liegen. Somit beeinflussen die Randbedingungen, die auf die *Interface Nodes* angebunden werden nicht die Aufprallzone des Modells.

Nachteilig hingegen ist die manuelle Eingabe der Box. Es gibt keine automatische Bestimmung der Größe des Submodells, wodurch der Nutzer sich selbst Gedanken über das Ergebnis machen muss.

3.2.4 Erweiterung der Evaluationsfunktion basierten Modellreduktion

Als dritte Methode zur Submodellerstellung folgt die Erweiterung der Evaluationsfunktion abhängigen Modellreduktion. In Tool *ASCO* wird diese als *extended Performance based Submodel Generation* (Kurz: ePBS) bezeichnet.

Die Erweiterung der Methode liegt darin, dass PBS zuerst angewendet wird, um ein geeignetes Submodell zu finden und dann auf dieses Submodell CBS angewendet wird, damit keine *Interface Nodes* im Bereich der Aufprallzone liegen. Beides wird

automatisch nacheinander ausgeführt, sodass kaum Eingaben durch den Nutzer erforderlich sind. Dadurch werden die Vorteile der beiden bisher vorgestellten Methoden vereint und die Nachteile reduziert.

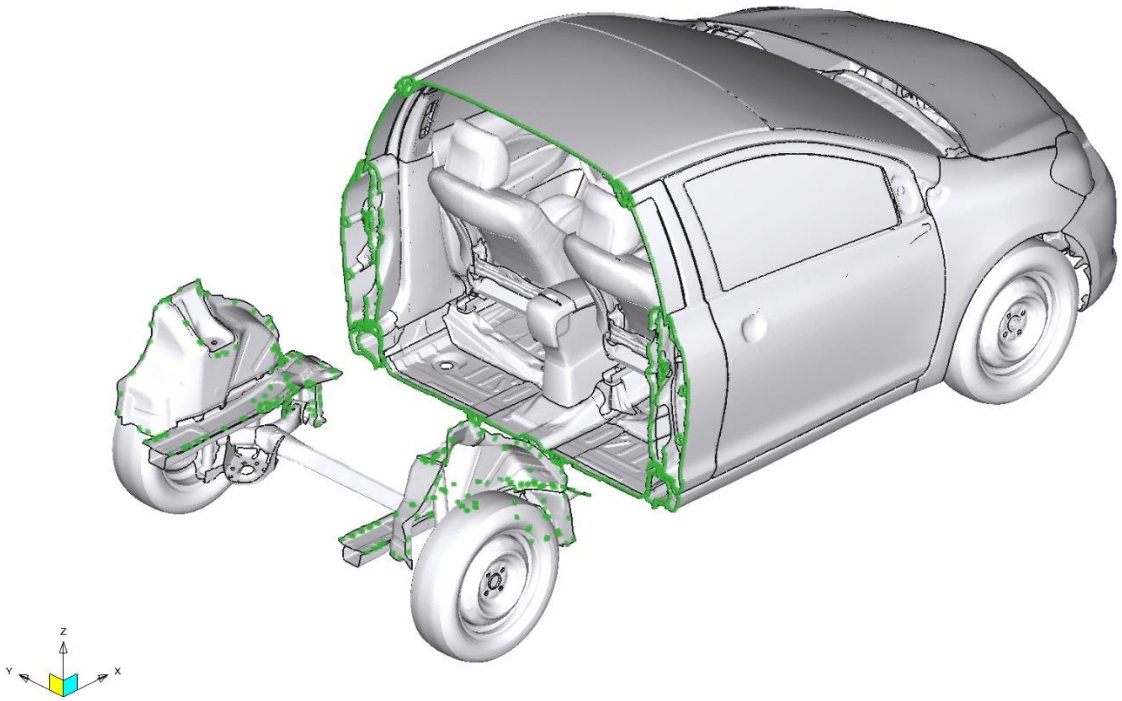


Bild 19: Submodell der *Coordinate based Submodel Generation* mit hinterem Fahrwerk als zusätzliche Bauteilgruppe und in grün markierten *Interface Nodes*. Modell: Toyota Yaris aus (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

Das Prozedere beginnt damit, dass der PBS-Algorithmus auf das Fahrzeugmodell angewendet wird und ein Submodell erzeugt. Anschließend wird der Schwerpunkt (engl.: *Center of Gravity*, Kurz: COG) des Submodells bestimmt. Die Koordinaten des COG dienen als Grenze der Box, die für den CBS-Algorithmus nötig ist. Es wird also automatisch eine Box gewählt, die vom COG zur Crashzone alles abdeckt. Damit der Algorithmus die richtige Richtung wählt, ist von dem Nutzer im Vorfeld der Lastfall anzugeben (Front-, Heck- oder Seitenaufprall). Danach wird der CBS-Algorithmus mit der definierten Box auf das Gesamtmodell angewendet. Alle Bauteile aus dem PBS und CBS Modell werden zusammengefügt und es wird ein Submodell erstellt, was die Komponenten aus den beiden Modellen enthält, wie unten in Bild 20 zu sehen ist.

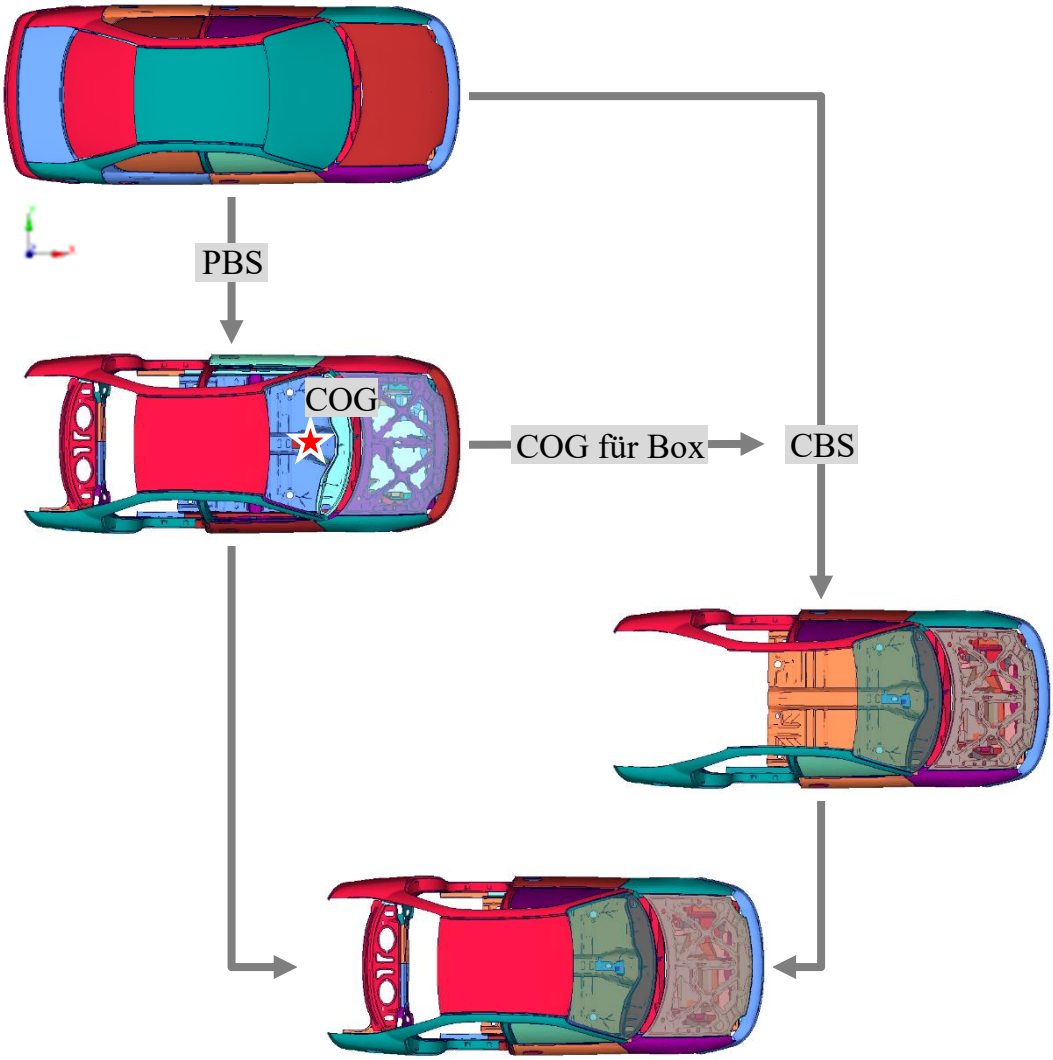


Bild 20: Schritt für Schritt Darstellung der *extended Performance based Submodel Generation*, als Kombination der *Performance based Submodel Generation* (PBS) und *Coordinate based Submodel Generation* (CBS), basierend auf dem Schwerpunkt (COG) des PBS Modells

3.3 Randbedingungen zur Wiederherstellung der Kinematik

Im nächsten Schritt nach der Erstellung des Submodells gilt es, die Kinematik des Modells wiederherzustellen. Dazu sind geeignete Randbedingungen nötig. Die Positionierung der Randbedingungen wird bereits während der Submodellgenerierung durch die *Interface Nodes* festgelegt.

Als Randbedingung kommen 1) Verschiebungen, 2) Kräfte und 3) die Anbindung einer Ersatzmasse in Frage. Eine weitere hier dargestellte Möglichkeit bietet die Änderung der Materialkarte der entfernten Bauteile. Diese letzte Möglichkeit lässt sich nur auf Submodellverfahren anwenden, die pro Bauteil und nicht pro Element reduzieren. In

den nächsten Abschnitten werden die drei genannten Randbedingungen, sowie die Änderung der Materialkarte vorgestellt und bewertet.

3.3.1 Aufprägung von Verschiebungen

Der Ansatz, Verschiebungen aus einer Simulation des Gesamtmodells zu extrahieren und dem Submodell aufzuprägen, wurde während des *eEgO*-Projekts verfolgt (siehe Abschnitt 2.5.4). Im *ASCO*-Tool wird dieser Ansatz der Randbedingungen als *Displacement Model Approach* (Kurz: DMA) betitelt. Es wird eine Simulation des gesamten Modells benötigt, welche abhängig vom gewählten Verfahren der Submodellgenerierung bereits vorhanden ist oder noch durchgeführt werden muss. Aus dieser Simulation werden die Verschiebungen der *Interface Nodes* extrahiert. Diese Extraktion ist wiederum abhängig vom FE-*Solver*.

Mit der *Virtual Performance Solution*® (VPS, ehemals PAM-CRASH) der Firma ESI ist es möglich, die Verschiebungen nachträglich aus der Simulation zu extrahieren. Der bereits genannte *Solver* LS-DYNA® bot bisher nicht die Möglichkeit einer nachträglichen Extraktion. Somit war es immer zwingend erforderlich eine weitere Simulation durchzuführen, bei der die Verschiebungen in ein *Interface Segment File* (Kurz: ISF) exportiert werden. Dazu sind eine Gruppe mit den *Interface Nodes* und das *Keyword* aus Bild 21 notwendig. Die referenzierte Gruppe hat hier die SID 3003 und enthält alle *Interface Nodes*.

```
$---------$---------$---------$---------$---------$---------$---------$---------
$ Interfaces
$---------$---------$---------$---------$---------$---------$---------$---------
*INTERFACE_COMPONENT_NODE
$       sid
       3003
$---------$---------$---------$---------$---------$---------$---------$---------
```

Bild 21: *Keyword* für LS-DYNA® zum Export der Verschiebungen an den *Interface Nodes*

Bei der Simulation des Submodells ist dann das ISF über das in Bild 22 dargestellte *Keyword* zu referenzieren. Hier ist neben der SID noch die ID des ISF anzugeben.

```
$---------$---------$---------$---------$---------$---------$---------$---------
$ Interfaces
$---------$---------$---------$---------$---------$---------$---------$---------
*INTERFACE_LINKING_NODE_SET
$       sid      ifid
       3003         1
$=========================================================================
```

Bild 22: LS-DYNA® *Keyword* zum Import der Verschiebungen an den *Interface Nodes*

Inzwischen bietet auch LS-DYNA® die Möglichkeit, ein solches ISF aus einer bereits bestehenden Simulation zu extrahieren. Dazu ist das Tool *plot2bc* (vgl. DYNAMORE

GMBH) notwendig, welches zusätzlich zu lizensieren ist. Mithilfe des Tools können aus einer *d3plot*-Datei die Verschiebungen extrahiert und im selben, oben beschriebenen Format des ISF gespeichert werden. Die d3plot-Datei wird von LS-DYNA® bei jeder Simulation erstellt und enthält die Ergebnisse der Simulation. Es ist möglich, den Inhalt dieser Datei durch Einstellungen des *Solvers* anzupassen, die Verschiebungen sind jedoch immer vorhanden. Das Tool *plot2bc* erspart somit die zusätzliche Simulation.

Die Aufprägung von Verschiebungen bietet eine sehr hohe Genauigkeit des Submodells, da die Verschiebungen exakt mit dem des Gesamtmodells übereinstimmen. Die Schwierigkeit daran ist jedoch, dass eine Veränderung der Struktur nicht zu anderen Ergebnissen führt, da die Verschiebungen nur durch eine Simulation des gesamten Modells aktualisiert werden können. An den *Interface Nodes* wird keine andere Deformation als die Aufgeprägte zugelassen, was die Kinematik des Modells stark einschränkt.

Verschiebungsrandbedingungen eignen sich daher vor allem für Submodelle, bei denen die Änderungen der Modellkinematik gering ausfallen. Deshalb ist DMA zur Optimierung der Karosserie eher ungeeignet, während die Optimierung von Rückhaltesystemen in der Fahrgastzelle gut möglich wäre. Mit einem Submodell mit aufgeprägten Verschiebungen könnte statt eines simulierten Schlittentest das reale Crashverhalten für die Untersuchungen verwendet werden.

3.3.2 Aufprägung von Kräften

Als Ergänzung zu aufgeprägten Verschiebungen sind Kräfte eine naheliegende Alternative. Das heißt, aus einer Simulation des Gesamtmodells werden Kräfte und Momente an den Knoten extrahiert und dem Submodell aufgeprägt. Es ist vorwegzugreifen, dass sich Kräfte nicht als Randbedingungen für Submodelle eignen. Unter anderem, weil sie nicht ausreichend genau sind und zu große Datenmengen benötigt werden.

Die Zusammenhänge werden mit einem einfachen Biegebalken mit 1000 x 100 mm und bestehend aus 10 Elementen dargestellt. Die Dicke des Stahlblechs beträgt 10 mm (siehe Bild 23). Eine Seite ist fest eingespannt, an der anderen wird eine Kraft F von 50 N, bzw. im zweiten Versuch eine Verschiebung s von 300 mm aufgeprägt. Es erfolgt ein Schnitt bei der halben Länge des Balkens, wodurch sich zwei *Interface Nodes* ergeben (rote Markierungen in Bild 23).

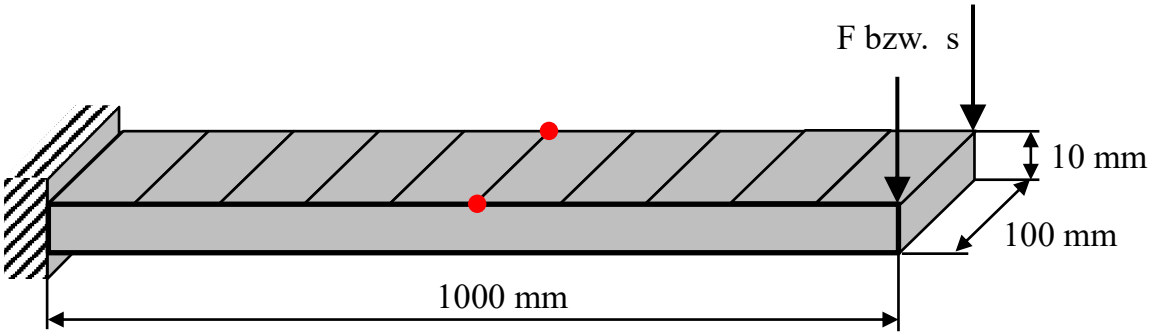


Bild 23: Biegebalkenmodell für die Untersuchung der Kraftrandbedingungen mit zwei *Interface Nodes* (rot)

Die Kräfte und Momente an den jeweiligen *Interface Nodes* können auf zwei Wegen ermittelt werden. Entweder ist für jeden Knoten eine sogenannte *Section* zu erstellen, welche die gesuchten Werte dann in eine Datei ausgibt oder es werden die Spannungen der umliegenden Elemente integriert und dann auf die Knoten extrapoliert. Bei beiden Ansätzen wird eine Ausgabe der Kräfte und Momente in der Größenordnung des Rechenzeitschritts benötigt, da es sonst zu einer Oszillation der Verschiebungen im Submodell kommt, wie in Bild 24 zu sehen ist. Dort schwingt die Linie des größten Zeitschritts mit einer relativ großen Amplitude um die Linie des Gesamtmodells. Beim nächstkleineren Zeitschritt verringert sich die Amplitude der Schwingung. Erst beim Berechnungszeitschritt (dt=1,8735E-05 s) ist kaum eine Oszillation im Vergleich zum Gesamtmodell auszumachen. Dieser geringe Zeitschritt führt zur Problematik der Datenmengen. Für eine einfache Simulation eines Biegebalkens, mit zwei *Interface Nodes* fallen bereits mehrere hundert Megabyte an Daten an. Diese Mengen skalieren sich linear und sind in Anbetracht eines Fahrzeugmodells mit mehreren tausend *Interface Nodes* nicht tragbar. Es würden für eine einzelne Simulation Daten im Terrabytebereich anfallen.

Hinzu kommt, dass eine Simulation in einfacher Präzision nicht ausreichend ist, sodass die Crashsimulationen in doppelter Genauigkeit durchgeführt werden müssen, was wiederum die Dauer für eine Simulation erhöht. Mit einfacher Präzision können die ausgelesenen Kräfte für den einfachen Biegebalken um 3 % gegenüber dem analytischen Ergebnis abweichen (siehe Tabelle 2). Als Beispiel ergibt sich die berechnete Normalkraft des Biegebalkens zu 50 N, während das Ergebnis aus der Simulation 48,4 N beträgt. In dieser Größenordnung führen Abweichungen dazu, dass das Modell in keiner Weise mehr seine ursprüngliche Kinematik aufweist. Zuletzt birgt die Integration über die Elemente einen hohen Programmieraufwand, da es diverse

Elementformulierungen gibt und diese alle unterschiedliche Positionen und Anzahlen von Integrationspunkten aufweisen.

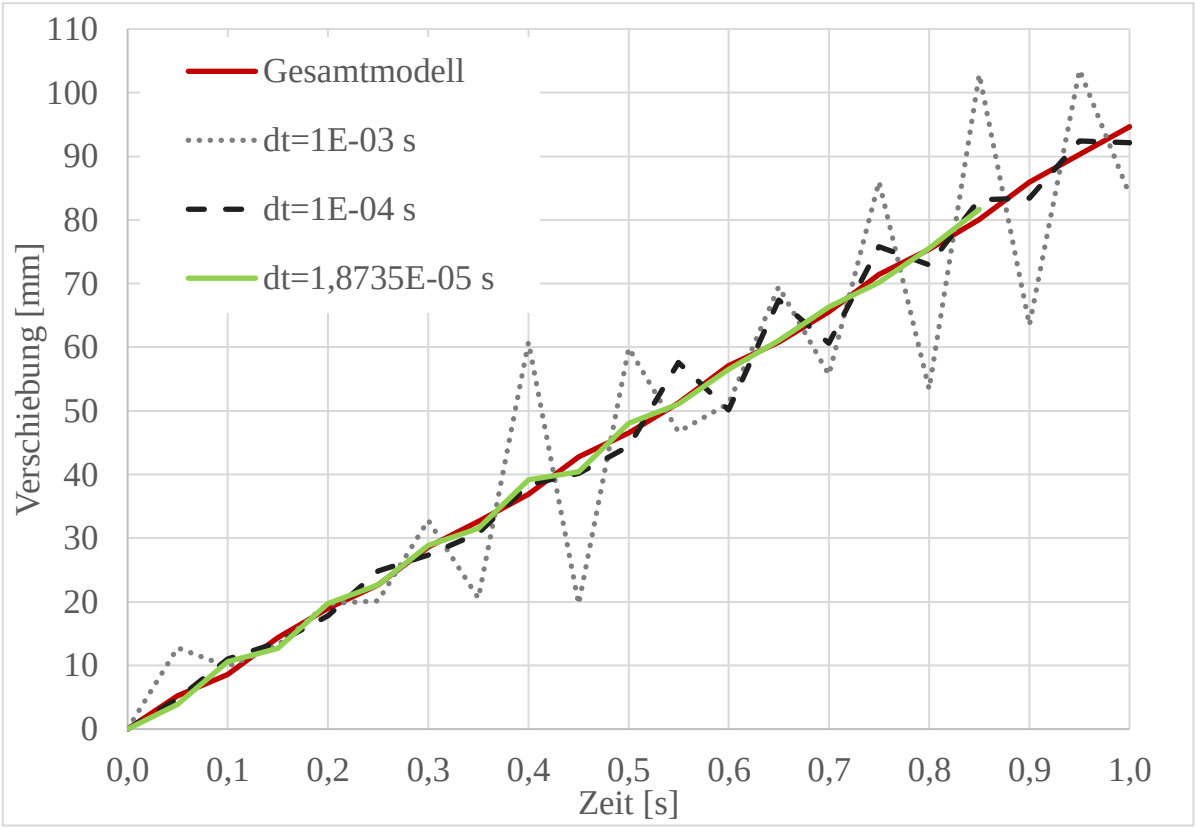


Bild 24: Verschiebungen eines einfachen Biegebalken über die Zeit mit Kräften als Randbedingungen

Tabelle 2: Vergleich der Kräfte in den *Interface Nodes* eines einfachen Biegebalkens mit einfacher und doppelter Präzision des *Solvers*

	Einfache Präzision	Doppelte Präzision
F_x	0 N	0 N
F_y	0 N	0 N
F_z	48,384 N	50 N
M_x	-286,27 Nmm	-262,31 Nmm
M_y	-24528 Nmm	-25000 Nmm
M_z	0 Nmm	0 Nmm

Sieht man von den programmtechnischen und numerischen Problemen ab, bringen Kräfte ein weiteres Problem. Bei einer Änderung des Submodells und daraus folgend einer Veränderung der Deformation, würde sich mit den gleichen Kräften eine völlig

andere Kinematik ergeben. Greift durch eine andere Deformation eine Kraft minimal anders an, kann das, aufgrund des instabilen Gleichgewichts, zum Kollabieren des Gleichgewichts und damit zu einer nicht nachvollziehbaren Deformation des Modells führen. Bei einer linearstatischen Simulation beispielsweise sind entweder Einspannungen notwendig, damit das System statisch bestimmt ist, oder die Option *Inertia Relief*. Mit letzterer ist es möglich, statisch unbestimmte Systeme zu berechnen, da Starrkörperbewegungen unterdrückt werden. Ohne eine der zwei Optionen wird das Modell in eine Raumrichtung beschleunigt. Für ein Crashmodell sind beide Optionen nicht möglich, da Starrkörperbewegungen notwendig sind und das Modell nicht statisch bestimmt sein muss. Bei der Anwendung von Kräften auf ein Crashmodell käme es aber zu ähnlichen Phänomenen, wie bei der linearstatischen Simulation, sodass das Modell in undefinierte Richtungen beschleunigt wird.

Kräfte und Momente werden aus den genannten Gründen nicht weiter betrachtet und auch nicht in das *ASCO*-Tool aufgenommen. Der Zeitaufwand zur Erstellung der Randbedingungen und der benötigte Speicherbedarf verhindern einen sinnvollen Einsatz der Technik. Hinzu kommt der hohe Programmieraufwand bei einem geringen Nutzen.

3.3.3 Anbindung einer Ersatzmasse

Ein vielversprechender Ansatz für Randbedingungen eines Submodells ist die Anbindung einer Ersatzmasse. Dies ist keine direkte physikalische Randbedingung, sondern eher eine vereinfachte Abbildung des entfernten Modellteils. Im *ASCO*-Tool wird dieser Ansatz als *Kinematic Mass Approach* (Kurz: KMA) bezeichnet.

Zur Ermittlung der notwendigen Größen wird neben dem Submodell ein Modell aus den entfernten Teilen erstellt. Mithilfe des Präprozessors Generator4® werden die Masse und Trägheit ermittelt. Außerdem wird der COG des invertierten Submodells bestimmt. Im COG wird ein neuer Knoten erzeugt, der die ermittelten Eigenschaften erhält. Das heißt dieser Knoten hat die Masse und Trägheit des inversen Submodells.

Als Anbindung des neuen Masseknotens zum Submodell dient ein *Nodal Rigid Body*. Dieser verbindet alle *Interface Nodes* mit dem Masseknoten. Ein Beispiel, bei dem das Submodell mit dem CBS-Algorithmus erstellt wurde, ist in Bild 25 zu sehen. Bei diesem Submodell wurde außerdem das Fahrwerk als Gruppe definiert, die nicht gelöscht werden soll.

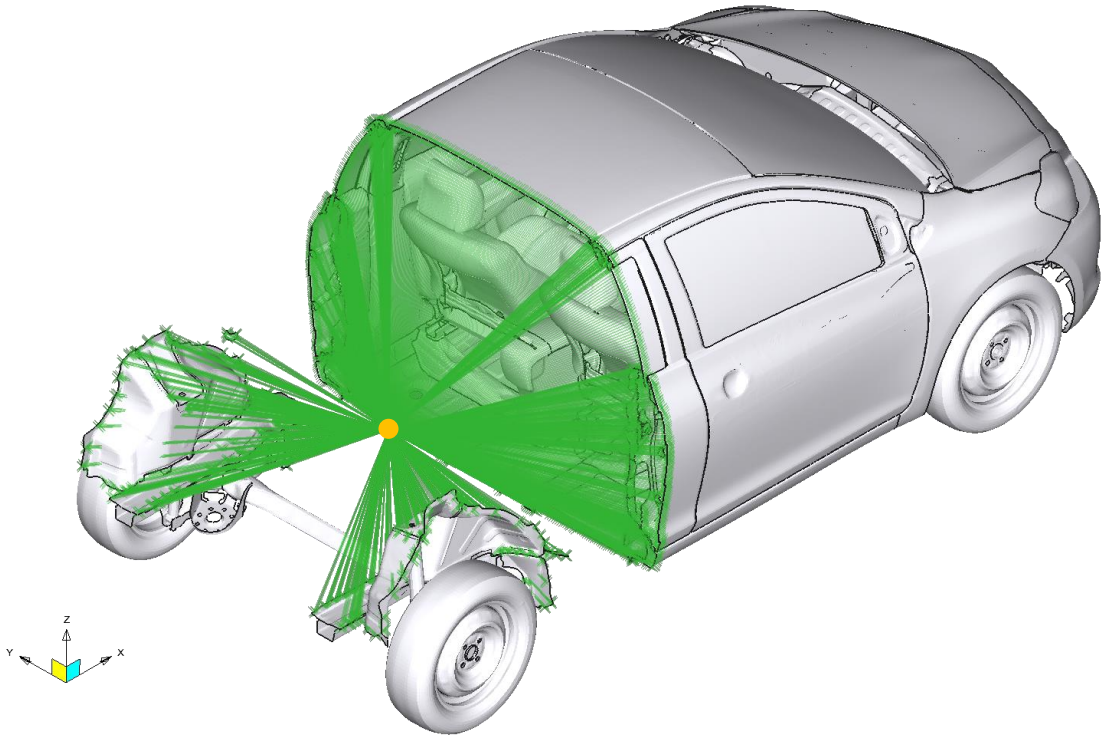


Bild 25: Mit einem Nodal Rigid Body (Grün) angebundene Ersatzmasse (Orange) an ein Submodell des Toyota Yaris aus (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020), erstellt mit dem CBS-Algorithmus und dem hinteren Fahrwerk, welches als nicht zu löschende Gruppe definiert ist

3.3.4 Anpassen der Materialkarte für ausgewählte Bauteile

Die letzte hier vorgestellte Methode verändert das Materialverhalten einiger Bauteile im Modell. Bauteile werden dann als *Rigid*, also unverformbar angenommen. Daraus ergibt sich das Problem, dass kein Submodell mit einer Reduktion anhand der Elemente verwendet werden kann. Die Materialkarte liegt pro Bauteil vor und deswegen ist von einer Veränderung immer das gesamte Bauteil betroffen. Außerdem ist es nicht sinnvoll, ein Submodell zu nutzen, bei dem Bauteile in mehreren Bereichen des Modells entfernt werden. Durch folgendes Beispiel wird diese Bedingung deutlich: Es soll ein Submodell für einen Frontcrash erstellt werden. Dazu wird der PBS-Algorithmus verwendet. Der Algorithmus entfernt den größten Teil des Hinterwagens und einige Bereiche um den Kühler und Motor. Werden die entfernten Bereiche nun durch unverformbare Komponenten ersetzt, führt das zu einer extremen Versteifung in der Aufprallzone und würde die Ergebnisse verfälschen. Anhand dieses Beispiels wird deutlich, dass die rigid-Bauteile möglichst weit von der Deformationszone entfernt sein müssen. Als Algorithmen kommen dann nur ePBS oder CBS mit einer Auswahl pro Bauteil in Frage.

Nachdem die Auswahl der Bauteile getroffen wurde, müssen diese als *Rigid* definiert werden. Hier gibt es zwei Ansätze um das Ziel zu erreichen. Der erste Ansatz nutzt ein *Keyword* von LS-DYNA, mit dem bereits bestehende Materialkarten nachträglich vom *Solver* als *Rigid* interpretiert werden. Mit dem *Keyword* `DEFORMABLE_TO_RIGID_AUTOMATIC` (Kurz: D2RA) können Bauteile angegeben werden, die dann intern vom *Solver* als starr angenommen werden. Wichtig zu wissen ist, dass alle Verbindungen zwischen betroffenen Bauteilen als *Nodal Rigid Body* definiert sind, da deformierbare Konnektoren zwischen zwei starren Bauteilen nicht rechenfähig sind. Es werden also zusammenhängende Inseln aus betroffenen Bauteilen gesucht und dann alle Konnektoren innerhalb des Verbunds durch *Nodal Rigid Bodies* ersetzt. Anschließend werden die Bauteile dem genannten *Keyword* zugeordnet und das Modell wäre fertig. Bei Simulationen auf mehreren Prozessoren (MPP) kann so allerdings keine Rechenzeiteinsparung erreicht werden. Der Grund liegt in der Dekomposition des Modells bei der Simulation. Für die Simulation mit mehreren Prozessoren wird das Modell auf die einzelnen CPUs aufgeteilt und die jeweiligen Berechnungsergebnisse werden zwischen den Knoten ausgetauscht. Sinnvoll ist eine Aufteilung, bei der jede CPU in etwa den gleichen Rechenaufwand hat, damit alle gleichzeitig die Berechnung abschließen. Standardmäßig unterscheidet LS-DYNA nicht zwischen Bauteilen die starr oder verformbar sind, wenn diese erst nachträglich durch das *Keyword* angepasst wurden. Es ist daher notwendig, die Dekomposition zu steuern. Mithilfe des *Keywords* `CONTROL_MPP_PFILE` kann ein so genanntes *Pfile* angegeben werden, in welchem die Dekomposition definiert ist. Möglichkeiten zur Beschreibung der Dekomposition im *Pfile* sind in (vgl. LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY 2020, S. 3463) aufgeführt.

Wie bereits angesprochen, gibt es auch eine zweite Möglichkeit das Modell anzupassen. Hierbei werden die Materialkarten der betroffenen Bauteile ersetzt. Vorhandene Materialdefinitionen werden dann durch `MAT_RIGID` ausgetauscht. Auch hier müssen wieder alle Konnektoren innerhalb einer Insel gesucht und gelöscht werden. Allerdings sind keine neuen Konnektoren erforderlich, da die einzelnen Bauteile über ihre Gruppe miteinander verbunden sind. Wichtig ist der Übergang zwischen starren und deformierbaren Bauteilen. Die vormals verbundenen Knoten der deformierbaren Bauteile müssen als sogenannte *Extra Nodes* zur Insel hinzugefügt werden. Eine Anpassung der Dekomposition ist bei diesem Ansatz nicht erforderlich, da das Standardverfahren mit den veränderten Materialkarten umgehen kann.

Bei beiden Methoden fällt die gewünschte Rechenzeiteinsparung gering aus. Selbst wenn ca. 50 % des Fahrzeugmodells als starr modelliert sind, liegt die Rechenzeiteinsparung bei maximal 15 %. Die Methode mit den ausgetauschten Materialkarten schneidet dabei minimal besser ab als über das D2RA *Keyword*.

Beide Varianten sind unter der Option `RIGID` im *ASCO*-Tool als Randbedingungen verfügbar (siehe Bild A - 3 im Anhang 0).

3.4 Validierung von Submodellen

Es ist essentiell ein erzeugtes Submodell zu validieren. Dabei sind die aufgenommenen Energien und die Kinematik entscheidende Größen. Im Laufe der Bearbeitung haben sich mehrere Verfahren zur Submodellvalidierung ergeben. Bereits im *eEgO*-Projekt wurden die Submodelle validiert. Die dort verwendete Metrik nutzt globale und lokale Kriterien, um das Sub- mit dem Gesamtmodell zu vergleichen, wie im Folgenden gezeigt wird. Während dieser hier vorliegenden Dissertation ist ein Verfahren entwickelt worden, welches das Submodell anhand einiger ausgewählter Bauteile validiert. Außerdem ist die Nutzung einiger externer Programme und Tools zur Validierung in *ASCO* realisiert.

3.4.1 Validierung anhand der Evaluationsfunktion

In *eEgO*-Projekt wurde eine Submodellvalidierungsmetrik umgesetzt. In (FALCONI D. ET AL. 2018) wird das Vorgehen der Validierung in vier Schritte aufgeteilt: globale, lokale, Mittelwert- und Strukturantwortvalidierung.

Wie bereits erklärt, werden zur Submodellgenerierung alle Bauteile anhand einer gewählten Strukturantwort evaluiert. Die gleiche Evaluation wird ebenfalls für das Submodell durchgeführt. Bei der globalen Validierung werden für jedes Bauteil des Submodells die Ergebnisse der Evaluationsfunktion mit denen des Gesamtfahrzeugs verglichen. Anhand der Summe aller Differenzen, bezogen auf die Anzahl der Bauteile im Submodell, ergibt sich der globale Validierungswert. Dabei gilt das Submodell als validiert, wenn dieser Wert unter 1 % liegt.

Die lokale Validierung basiert auf der globalen Validierung. Hierzu wird der Anteil der Bauteile ermittelt, deren Abweichung oberhalb des globalen Validierungswerts liegen. Es ergibt sich ein Prozentsatz, der als lokaler Validierungsprozentsatz bezeichnet wird. Dieser darf nicht über 50 % liegen, wodurch weniger als die Hälfte der Bauteile eine höhere Abweichung als der Mittelwert haben. Bei Erfüllung dieses Kriteriums wird das Submodell als valide eingestuft.

Es folgt die Mittelwertvalidierung. Auch diese Validierung beruht auf den Ergebnissen der Evaluierungsfunktion der Bauteile im Submodelle. Sowohl im Submodell als auch im Gesamtmodell wird der Mittelwert der Evaluation über die Bauteile ermittelt. Dabei werden im Gesamtmodell nur die Bauteile berücksichtigt, die ebenfalls im Submodell enthalten sind. Die relative Abweichung zwischen den beiden Werten sollte unter 1 % liegen.

Im letzten Schritt der Submodellvalidierung wird die Abweichung einer bestimmten Strukturantwort ausgewertet. Dabei ist nicht festgelegt, welche Strukturantwort hier zu vergleichen ist. Beispiele wären die Zielfunktion der Optimierung oder die Verschiebung eines einzelnen Knotens. Zur Validierung wird die Strukturantwort im Submodell und Gesamtfahrzeug über die Zeit integriert. Die Differenz der beiden Ergebnisse wird auf den Wert des Gesamtfahrzeugs bezogen, sodass sich ein relatives Ergebnis ergibt. Damit das Submodell als valide gilt, sollte die relative Abweichung unter 1 % liegen.

Beim letzten Validierungsschritt gilt zu beachten, dass die gewählte Strukturantwort nicht von der Größe des Submodells abhängig ist. Beispielsweise kann die interne Energie des Gesamtmodells nicht als Vergleichskriterium gewählt werden, da durch die größere Anzahl an Elementen im Gesamtmodell auch eine höhere interne Energie als im Submodell vorhanden ist.

3.4.2 Validierung durch Auswahl geeigneter Bauteile

Die zuvor vorgestellte Validierung aus dem *eEgO*-Projekt lässt sich so wie vorgestellt nur auf Submodelle mit dem PBS-Algorithmus anwenden. Basierend auf der Idee hinter dieser Metrik wurde ein eigenständiges Validierungsverfahren erstellt, welches unabhängig von der Submodellgenerierung ist.

Diese Metrik wird ebenfalls Bauteile im Submodell mit dem Gesamtfahrzeug vergleichen. Dazu wird anhand der genannten Evaluationsfunktion eine Auswahl an Bauteilen getroffen. Bedingung dieser Auswahl ist eine definierte Mindestanzahl an Elementen. Die Mindestanzahl ergibt sich als der Median der vorhandenen Bauteilgrößen. Es werden die ersten N Bauteile mit dem höchsten Evaluationswert und der Mindestelementanzahl zum Vergleich herangezogen.

Für jedes ausgewählte Bauteil wird der zeitliche Verlauf einer Reihe an Strukturantworten untersucht. Als Vergleichsverfahren dient die *Rating Metric* nach (BARBAT ET AL. 2013). Diese Metrik ist ebenfalls in der ISO/TS-18571 festgehalten. Das Ergebnis dieses Verfahrens setzt sich aus vier gewichteten Einzelvergleichen zusammen (siehe

Bild 26). Wobei das Hauptaugenmerk, mit einer Gewichtung von 40 %, auf einer Korridorbewertung liegt. Die anderen drei Verfahren zur Bestimmung der Phasenverschiebung, Amplitude und Steigung gehen mit jeweils 20 % in die Endwertung ein.

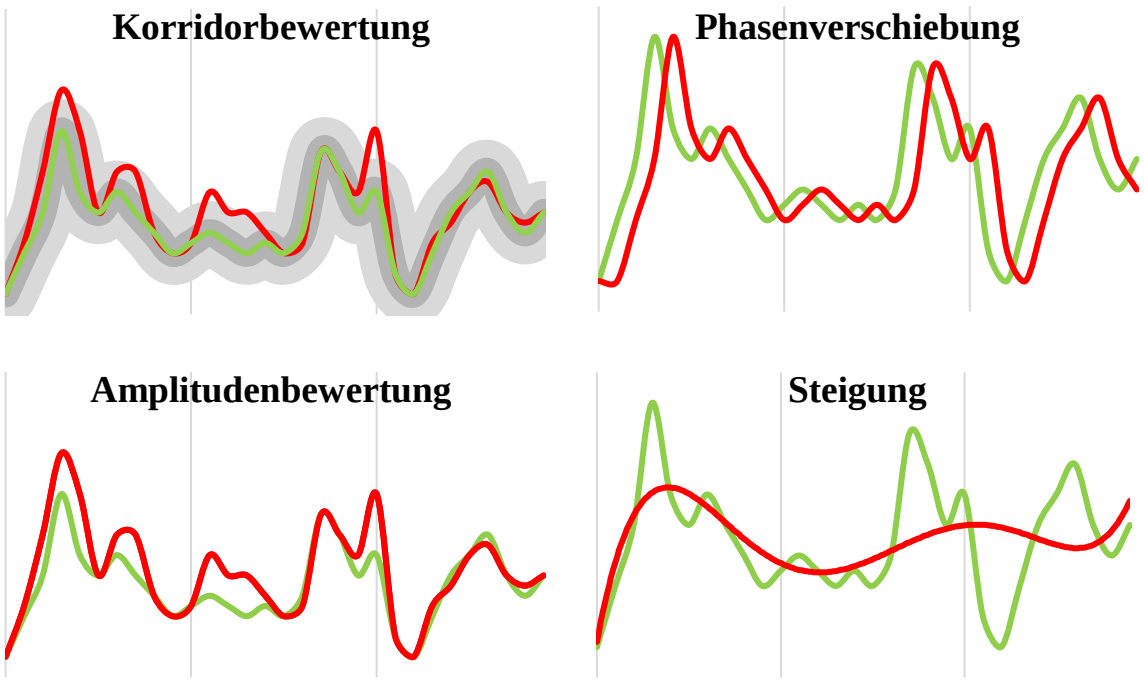


Bild 26: Bewertungskriterien nach (BARBAT ET AL. 2013) mit den Referenzkurven (grün) und Vergleichskurven (rot). Bei der Korridorbewertung sind ebenfalls der innere (dunkelgrau) und äußere Korridor (hellgrau) eingezeichnet

Bei der Korridorbewertung werden zwei Korridore entlang der Referenzkurve definiert: ein innerer und ein äußerer Korridor. Die Weite der Korridore berechnet sich aus der maximalen absoluten Amplitude der zugrundeliegenden Referenzkurve. Ziel ist es, einen schmalen inneren und einen weiten äußeren Korridor zu erhalten, sodass eine präzise Unterscheidung der Güte möglich ist. Für jeden Zeitpunkt wird die Vergleichskurve anhand der Korridore bewertet. Liegt die Vergleichskurve innerhalb des inneren Bereichs wird dieser Punkt mit 1 bewertet. Für den Bereich zwischen dem inneren und äußeren Korridor erfolgt eine Abstufung von 1 bis 0 und alle Werte außerhalb des äußeren Korridors werden mit 0 bewertet. Die Gesamtbewertung der Korridorvalidierung ergibt sich aus dem Durchschnitt aller Zeitpunkte.

Zur Ermittlung der Phasenverschiebung wird die Vergleichskurve nacheinander in beide Richtungen zur Referenzkurve verschoben. Dabei wird jeweils die Kreuzkorrelation ermittelt. Die benötigte Verschiebung im Vergleich zur maximal zulässigen

Verschiebung bei der besten Übereinstimmung dient als Bewertung. Ist keine Verschiebung notwendig, wird die Phasenverschiebung mit 1 bewertet, ist die notwendige Verschiebung größer als der zulässige Schwellenwert, ergibt sich die Bewertung 0.

Damit die Amplituden verglichen werden können, ist zuerst die Phasenverschiebung auszugleichen. Die Bewertung der Amplitude erfolgt anhand der korrigierten Kurve, da die Phasenverschiebung wie dargestellt extra bewertet wird. Besteht keine Differenz zwischen den Amplituden der Referenz- und Vergleichskurve, ergibt sich die Bewertung 1. Beim Überschreiten einer zulässigen Differenz ist die Bewertung 0.

Zuletzt werden die Steigungen der zwei Kurven miteinander verglichen. Auch hier werden die verschobenen Kurven verwendet. Ist keine Differenz vorhanden, wird dieses Kriterium mit 1 bewertet, absteigend bis 0 bei Überschreiten der zulässigen Schwelle.

Die vier Kriterien werden entsprechend der genannten Gewichtungen verrechnet, sodass sich eine Beurteilung im Bereich von 0 bis 1 ergibt. In (BARBAT ET AL. 2013) wird eine Einordnung der Bewertungen gegeben, welche in Tabelle 3 dargestellt ist.

Tabelle 3: Bewertungen des *Curve Ratings* nach (BARBAT ET AL. 2013)

Note	**Bewertung R**
Exzellent	$R < 0{,}94$
Gut	$0{,}8 < R \leq 0{,}94$
Mäßig	$0{,}58 < R \leq 0{,}8$
Schlecht	$R \leq 0{,}58$

Nach dem vorgestellten Verfahren ergibt sich für jede Strukturantwort und jede Komponente eine Bewertung. Die aktuell ausgewerteten Strukturantworten sind die interne und kinetische Energie, sowie die Starrkörperbewegungen in die drei Raumrichtungen. In Bild 27 ist ein Ausschnitt der Ergebnisdatei zu sehen. Es ist dort nur eine Strukturantwort dargestellt, die weiteren werden rechts anschließend aufgeführt. Die Gewichtung, die in der Spalte *Weight* aufgelistet ist, entspricht den normierten Werten der Evaluationsfunktion. Sie wird zur Ermittlung der *Weighted Scores* verwendet. Es wird für jede Strukturantwort der mittlere und ein gewichteter Wert angegeben, sowie die Bewertungen aller Strukturantworten in eine Gesamtbewertung (*Overall Mean Score*) und eine gewichtete Gesamtbewertung (*Overall Weighted Score*) verrechnet.

Die gewichtete Gesamtbewertung wird dem Tool als Validierungskriterium zurückgegeben.

```
Overall weighted score:      0.9033816830388512
Overall mean score:          0.908


                                | Internal Energy
PID             | Weight        | Score          | Grade
-----------------------------------------------------------------
2000377         | 1.0           | 0.65           | Fair
2000378         | 0.88226632    | 0.79           | Fair
2000451         | 0.81823848    | 0.87           | Good
2000178         | 0.8173154     | 0.92           | Good
2000380         | 0.7276312     | 0.90           | Good
2000383         | 0.7057994     | 0.95           | Excellent
2000450         | 0.64196231    | 0.98           | Excellent
2000330         | 0.63429439    | 0.57           | Poor
2000379         | 0.59789098    | 0.98           | Excellent
2000185         | 0.58710444    | 0.94           | Good
-----------------------------------------------------------------
weighted score                  | 0.8451460749643789
mean score                      | 0.8550000000000001
```

Bild 27: Ausschnitt aus der Ergebnisdatei der Validierung durch Auswahl geeigneter Bauteile

3.4.3 Verwendung der externen Software DIFFCRASH®

Die Firma Sidact GmbH bietet mit ihrer Software DIFFCRASH® eine Möglichkeit zwei oder mehr Simulationen miteinander zu vergleichen. Dazu werden Differenzen zwischen den Verschiebungen aller Knoten der Modelle untersucht. Schwierig ist die Verwendung zweier verschiedener Modelle, was ein Gesamtmodell und ein Submodell für das Tool sind. Dann ist es erforderlich, das Submodell als Hauptmodell zu verwenden, sodass alle Knoten des Submodells auch im Gesamtfahrzeug auffindbar sind.

Beim Vergleich der Modelle wird eine *Principal Component Analysis* (Kurz: PCA) durchgeführt, sodass die Differenzen zwischen zwei Modellen auf einen Wert reduziert werden (vgl. THOLE ET AL. 2010). Dieser Wert ist jedoch nicht auf den Bereich 0 bis 1 normiert, da hier prinzipiell die Differenz der Modelle beschrieben wird. So bietet ein Wert von 0 exakte Übereinstimmung zwischen den beiden Modellen. In zukünftigen Arbeiten sollte eine Metrik erforscht werden, welche die Ergebnisse der PCA aus DIFFCRASH® auf einen einzelnen Wert zwischen 0 und 1 skaliert. Dazu sind umfangreiche DoE-Untersuchungen von verschiedenen Modellen notwendig, aus denen ein Berechnungsverfahren hergeleitet werden kann.

3.4.4 Nutzung des Validierungstools der Porsche AG

Im Rahmen eines Submodell-Arbeitskreises zwischen der Volkswagen AG, Porsche AG und Bergischen Universität Wuppertal wurde das in der Porsche AG entstandene,

automatisierte Tool *Resultcompare* zur Modellvalidierung weiterentwickelt. Das Tool validiert Modelle anhand von bis zu fünf Kategorien: globale Energien, lokale Energien, Fahrzeug-Barriere-Interaktion, Verschiebungen und Schnittkräfte. Bei den globalen Energien werden die gesamte kinetische und interne Energie der Modelle miteinander verglichen. Dieser Vergleich der globalen Energie ist deaktivierbar, da er bei verschieden großen Modellen nicht sinnvoll ist. Für die lokalen Energien findet eine Bauteilauswahl ähnlich zu der in 3.4.2 vorgestellten Methodik statt. Eine Auswertung der Kraft auf die Barriere erfolgt bei der Fahrzeug-Barriere-Interaktion. Dabei werden die Kräfte in den drei Raumrichtungen anhand der maximalen Amplituden gewichtet. Ähnliche Gewichtungen werden auch bei den Verschiebungen und Schnittkräften durchgeführt. Für die Verschiebungen werden sechs Knoten im Modell ausgewertet. Die Knotenauswahl kann manuell oder automatisch erfolgen. Einzig die Schnittkräfte erfordern manuell erstellte sogenannte *Cross-Sections* in denen die Kräfte auswertbar sind. Diese Kategorie ist deshalb ebenfalls optional und kann, falls keine *Cross-Sections* erstellt wurden, deaktiviert werden. Alle Größen werden über die Zeit miteinander verglichen, dazu kommt auch hier die in 3.4.2 vorgestellte Metrik der *Curve Ratings* von (BARBAT ET AL. 2013) zum Einsatz. (vgl. BÜTTNER 2022)

Das *ASCO*-Tool bietet eine Schnittstelle für das *Resultcompare*-Tool der Porsche AG.

3.4.5 Einbindung individueller Validierungsskripte

Beim Aufbau der *ASCO*-Software ist vorgesehen worden, auch individuelle Validierungsskripte integrieren zu können. Dazu kann ein eigenes Skript ausgeführt werden, für welches der Startbefehl anzugeben ist und eine Datei, aus dem das Ergebnis auszulesen ist. Für eine bessere Vergleichbarkeit sollte der Ergebniswert zwischen 0 und 1 liegen. Die entsprechenden Einträge in der *ASCO*-Konfigurationsdatei sind in Bild 28 zu sehen. Jeweils in den geschwungenen Klammern sind die Eingaben zu tätigen. Damit die Parameter aktiv werden, ist bei `VALI_METHOD` die Option `OWNSC` (*own script*) zu wählen.

```
VALI_METHOD                                 {OWNSC}
## OWNSC:
# Command to start your script
COMMAND                                     {}
# File with a single validation value between 0 and 1
VALI_FILE                                   {}
```

Bild 28: Ausschnitt aus der *ASCO*-Konfigurationsdatei zu den Einstellungen für eigene Validierungsskripte

3.5 Workflow zur Erstellung von Submodellen

Je nach vorliegendem Modell, gewünschter Nutzung und Lastfall werden entsprechende Anforderungen an die Software gestellt. Wichtig ist, dass die Software automatisch abläuft und robust bei den verschiedenen Modellen ist. Dazu ist die Software in Module aufgeteilt, die unabhängig voneinander ausgeführt werden können.

ASCO ist in vier Kategorien von Modulen aufgeteilt: Erstellung, Randbedingung, Validierung und Nutzung. Über eine Konfigurationsdatei werden alle Einstellungen getroffen, sodass keine Anpassungen im Code notwendig sind. In dem Modul der Submodellerstellung sind die in Abschnitt 3.2 vorgestellten Verfahren umgesetzt. Ein Teil der Randbedingungen aus Abschnitt 3.3 wurden für die Randbedingungen umgesetzt. Konkret sind DMA, KMA und RIGID umgesetzt worden. Bei den Validierungsverfahren gibt es die vorgefertigten und bereits genannten Verfahren, sowie die Möglichkeit eigene Skripte einzubinden.

Das *ASCO*-Tool besteht initial aus einem Ordner *ASCO*, in dem alle benötigten Dateien und Unterordner liegen (siehe Bild 29 (a)). Eine detaillierte Ordnerstruktur mit allen Unterordnern und Dateien ist im Anhang A.1 dargestellt. Für den Nutzer relevant sind erst einmal nur der Ordner `LOADCASE` und die *ASCO*-Konfigurationsdatei `params.cfg`. In dieser Konfigurationsdatei werden sämtliche Einstellungen getroffen, angefangen bei Pfaden für die benötigte Software, bis hin zu Optimierungsparametern. Alle Optionen der Konfigurationsdatei sind in Anhang 0 dargestellt.

In den Ordner `LOADCASE` ist das Gesamtmodell zu speichern, dieses sollte eine Hauptdatei haben, von welcher aus alle anderen benötigten Modelldaten referenziert werden. Der `SOURCE`-Ordner enthält die Programmdaten und sollte vom Nutzer nicht verändert werden. Ein Ordner, der vom Nutzer bearbeitet werden kann ist `RESPONSES`, dort liegen einige XML-Dateien, in denen die auszuwertenden Strukturantworten für die Optimierungssoftware LS-OPT® gespeichert sind. Für die Submodellerstellung sind diese Daten nicht relevant, erst für eine Optimierung oder einen DoE.

Zum Start von *ASCO* wird die Datei `main.py` in einem Terminal ausgeführt. Der Prozess läuft von hier an automatisch entsprechend der gewählten Konfiguration ab. Informationen über den Status, Warnungen und Fehlermeldungen werden in der Datei `ASCO.log` abgelegt. Es können auch einzelne Module ausgeführt werden, dazu sind weitere Daten notwendig. Ist beispielsweise bereits eine Simulation des Gesamtmodells verfügbar, kann diese in einem Ordner `INITIAL` im Hauptverzeichnis von *ASCO* gespeichert werden. Das Tool erkennt die Simulation dann selbstständig und

überspringt die entsprechenden Schritte. Gleiches funktioniert mit bereits vorhandenen Submodellen, wobei hier die richtige Bezeichnung zu beachten ist. Abhängig vom genutzten Verfahren wird der Submodellordner so bezeichnet, dass die wichtigsten Parameter und das gewählte Verfahren dem Namen zu entnehmen sind. Beispielhaft ist in Bild 29 (b) eine Ordnerstruktur mit erstelltem Submodell aufgeführt. Das Submodell wurde mit dem PBS-Algorithmus erstellt und es wurde die Option „Submodellgröße“ verwendet, mit einer gewünschten relativen Größe von 0,6. Für die anderen Algorithmen wird dort die entsprechende Bezeichnung eingetragen.

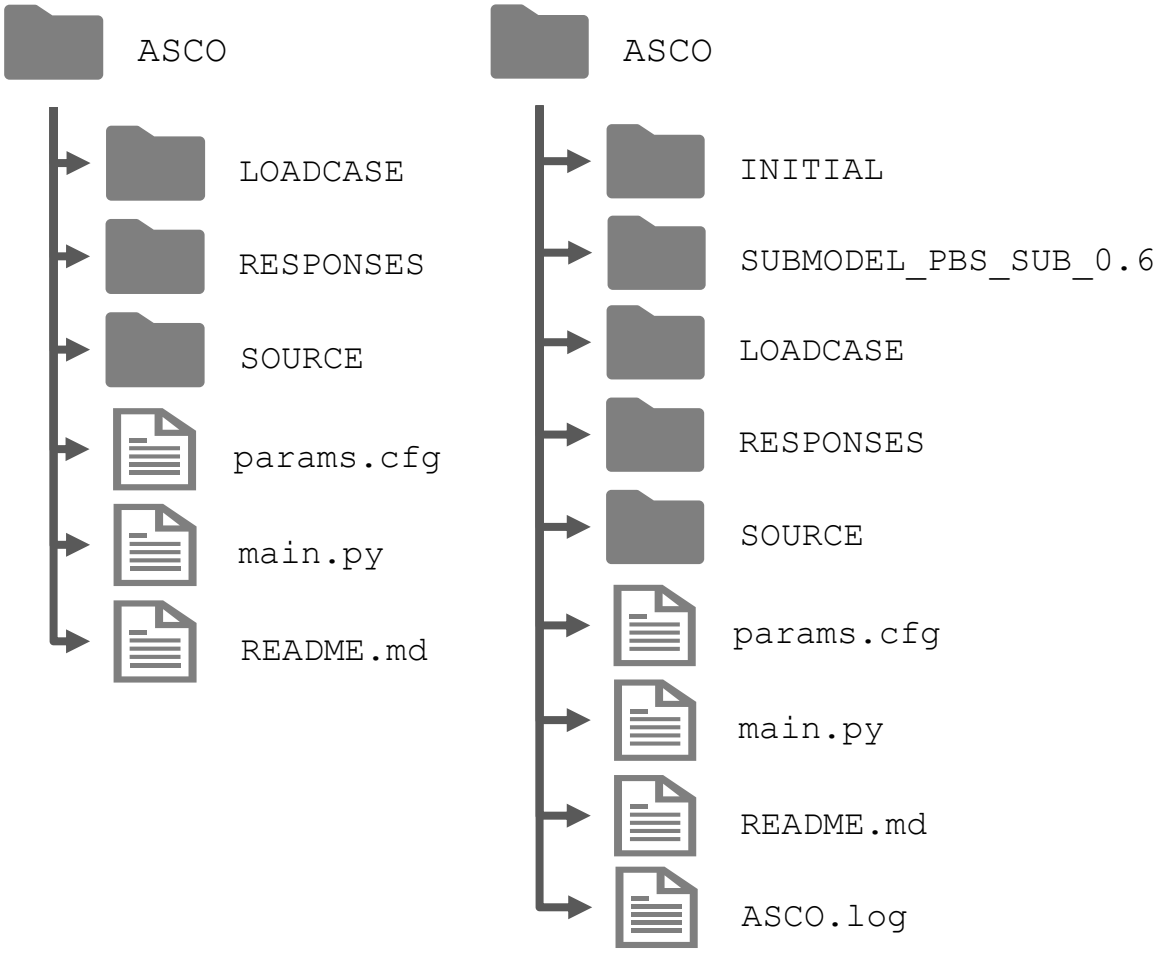


Bild 29: Übersicht der Ordnerstruktur des *ASCO*-Tools im initialen Zustand (a) und nach der Erstellung eines Submodells (b)

Intern wird zuerst geprüft, welche Art des Submodells zu erzeugen ist und ob dazu eine Simulation des Gesamtmodells erforderlich ist. Wird eine Simulation benötigt und ist diese nicht bereits vorhanden, erstellt das Tool einen Ordner `INITIAL` und führt den *Solver* zur Berechnung des Modells aus. Die Ergebnisse der Simulation liegen

dann im `INITIAL`-Ordner. Von hier aus wird die entsprechende Submodellgenerierung gestartet. Im Programmcode der Datei *createsubmodel.py* gibt es eine Klasse für jedes Verfahren, welche bei Bedarf ausgeführt wird. Das erstellte Submodell wird im Ordner „`SUBMODEL`_Verfahren_Parameter“ gespeichert. Bei der Verwendung von Verschiebungsrandbedingungen wird ein zugehöriges Gesamtmodell mit den *Interface Nodes* erzeugt, welches zum Update der Randbedingungen verwendet wird. Es ist möglich, mehrere Submodelle parallel zu generieren, dann werden mehrere Ordner erstellt, die wie oben bezeichnet sind. Im Anschluss an die Erstellung der Submodelle erfolgt die Aufbereitung der *Solverdecks*.

Durch die Änderungen im *Solverdeck* kann es vorkommen, dass das Submodell direkt nach der Erstellung nicht rechenfähig ist. Gründe hierfür sind meistens Verknüpfungen zwischen verschiedenen *Keywords*, die bei der Reduktion nicht entdeckt wurden. Ein Algorithmus (*cleanInputdeck*) sucht Fehler in den Dateien und behebt diese. Es ist nicht immer möglich, alle Fehler zu korrigieren, je nach Komplexität des Modells und ob der entsprechende Fehler bereits implementiert ist. Der *cleanInputdeck*-Algorithmus kann über die Konfigurationsdatei deaktiviert werden (Bild A - 3 im Anhang).

Mit dem korrigierten Submodell wird eine Simulation durchgeführt, anhand der die Güte beurteilt werden kann. Bei einer unzureichenden Validität des Modells oder keinem rechenfähigen *Inputdeck* wird eine Fehlermeldung mit entsprechenden Hinweisen ausgegeben. Für die Prüfung der Validität ist eine Simulation des Gesamtmodells notwendig. Sollte diese noch nicht vorliegen, wird sie vom Tool durchgeführt. Der gewählte Validierungsalgorithmus wird gestartet und zum Schluss wird die Bewertung des Submodells ausgegeben.

Der Nutzer kann dann entscheiden, wie das Submodell genutzt werden soll. Einige Möglichkeiten dazu werden im nachfolgenden Kapitel 4 vorgestellt.

4. Einsatz von Submodellen in der Optimierung

Im Folgenden werden Einsatzgebiete von Submodellen dargestellt, wobei besonderes Augenmerk auf die submodellbasierte Optimierung gelegt wird.

4.1 Ablauf submodellbasierter Optimierungen

Anhand der für die Optimierung benötigten Strukturantworten können Bauteile und Bereiche des Fahrzeugs bestimmt werden, welche zwingend im Submodell enthalten sein müssen. Diese sollten über eine der Funktionen des Tools (z. B. `KEEP_PIDS`, Details in Anhang Bild A - 3) als nicht löschbar definiert werden. Als nächstes ist ein geeigneter Submodellalgorithmus auszuwählen. Die Auswahl sollte zusammen mit den Randbedingungen überlegt werden. Für Probleme, bei denen die Strukturmechanik und das kinematische Verhalten des Modells im Vordergrund stehen, bietet es sich an, eine Ersatzmasse mit dem CBS Algorithmus zu nutzen, da hier die kinematische Flexibilität des Modells erhalten bleibt. Zum Schluss sollte die Submodellgröße abgeschätzt werden. Im Falle eines Hochgeschwindigkeitslastfalls sollte tendenziell eher ein größeres Modell gewählt werden, da auch in weiter hinten liegenden Bereichen des Modells noch Energie aufgenommen wird. Für Lastfälle mit niedriger Geschwindigkeit, wie die Versicherungslastfälle kann das Submodell auch kleiner ausfallen, da im besten Falle nur Strukturen in unmittelbarer Nähe des Hindernisses deformiert werden.

4.2 Sicherstellung der Submodellvalidität während der Optimierung

Es gibt zwei mögliche Ansätze, die Validität des Submodells im gesamten Entwurfsraum zu prüfen. **Möglichkeit Eins** ist die Prüfung vor Start der Optimierung. Dazu wird im Vorfeld der Optimierung ein DoE durchgeführt, mit relativ wenigen Stützstellen, die über den gesamten Entwurfsraum verteilt sind. An den Stützstellen wird sowohl das Submodell, als auch das Gesamtmodell simuliert. Es kann dann eine Validierung an jeder Stützstelle stattfinden und somit die Eignung des Submodells überprüft werden.

Die **zweite Möglichkeit** ist die Validierung während der Optimierung. Dieses Verfahren ist insofern von der Art der Optimierung abhängig, als das bei einer metamodellbasierten Optimierung ausgewählte Stützstellen des DoE durch ein Gesamtmodell zu validieren sind. Hierzu ist ein Verfahren zu entwickeln, welches die zu validierenden Stützstellen bestimmt, im einfachsten Fall kann eine zufällige Auswahl

erfolgen. Für eine direkte Optimierung sollten je nach Anzahl der Stützstellen pro Iteration mindestens der beste Entwurf einer Iteration durch eine Gesamtmodellsimulation überprüft werden. Für eine hohe Anzahl Stützstellen pro Iteration kann es durchaus sinnvoll sein, an mehreren Stützstellen in einer Iteration zu validieren. Hierzu könnte eine Abstandsfunktion der jeweiligen Stützstelle vom Startentwurf der Iteration hilfreich sein, wie sie oben vorgestellt wurde. Als einfachste Möglichkeit wäre eine zufällige Auswahl einsetzbar.

4.3 Weitere Anwendungsbereiche von Submodellen

Bisher lag das Augenmerk vor allem auf der Nutzung von Submodellen in der Fahrzeugcrashoptimierung. Dies ist auch das Hauptaugenmerk dieser Arbeit. Trotzdem soll hier ein kurzer Überblick über andere Anwendungsbereiche für Submodelle folgen.

Gängige Praxis beim Insassenschutz sind sowohl im Experiment, als auch in der Simulation Schlittentests, bei denen der Sitz mit umliegenden Anbindungspunkten und einem Dummy nachgebildet wird. Es gibt standardisierte Testverfahren für solche Schlittentests, wie sie in (EURO NCAP 2021) dargestellt sind. Mit einem Submodell, wie es beispielhaft in Bild 30 dargestellt ist, würde zur Extraktion der Verschiebungen an den *Interface Nodes*, lediglich eine Gesamtfahrzeugsimulation nötig sein.

Eine weitere Idee für Submodelle in der Automobilbranche ist die Herausgabe von Modellen an Externe. Zulieferer benötigen Daten, um ihr Produkt auf das des Fahrzeugherstellers anzupassen. Aus Geheimhaltungsgründen kann oft allerdings kein gesamtes Modell an externe Firmen herausgegeben werden. Die Erstellung rechenfähiger Submodelle, die den für den Zulieferer relevanten Bereich abbilden, ist von Hand aufwendig. Denkbar ist hier die Verwendung von automatisch generierten Submodellen, die auch automatisch validiert wurden. Der Zeitaufwand bei der Modellerstellung würde sich drastisch reduzieren.

Außerhalb der Automobilbranche sind der Anwendung von Submodellen kaum Grenzen gesetzt, solange es sich um nichtlineare Simulationen handelt. Theoretisch wäre die Methodik auch auf lineare Problemstellungen anwendbar. Aufgrund der geringen Komplexität linearstatischer Simulationen und der damit verbundenen geringen Rechenzeit sind Submodelle durch den Aufwand bei der Erstellung und der verringerten Aussagekraft gegenüber dem originalen Modell im linearstatischen nicht sinnvoll einsetzbar. Einsatzbereiche von Submodellen außerhalb der Automobilindustrie sind in der Luftfahrt und im Bauwesen zu finden. Bei einer Notlandung oder einem Absturz gibt es diverse Elemente in einem Flugzeug, welche die Energie aufnehmen müssen.

Neben der grundlegenden Flugzeugstruktur tragen vor allem die Sitze (vgl. YOGANANDAN ET AL. 2015, S. 825 f) und Crashelemente im Boden von Flugzeugen und Hubschraubern (vgl. MCCARTHY ET AL. 2000) zur Energieaufnahme im Falle eines Absturzes bei. Für solche Simulationen wäre es ebenfalls möglich, die Submodelltechnik anzuwenden, sodass z. B. die Strukturen der Sitze im Crashlastfall optimiert werden können, ohne dazu ein gesamtes Verkehrsflugzeug zu simulieren.

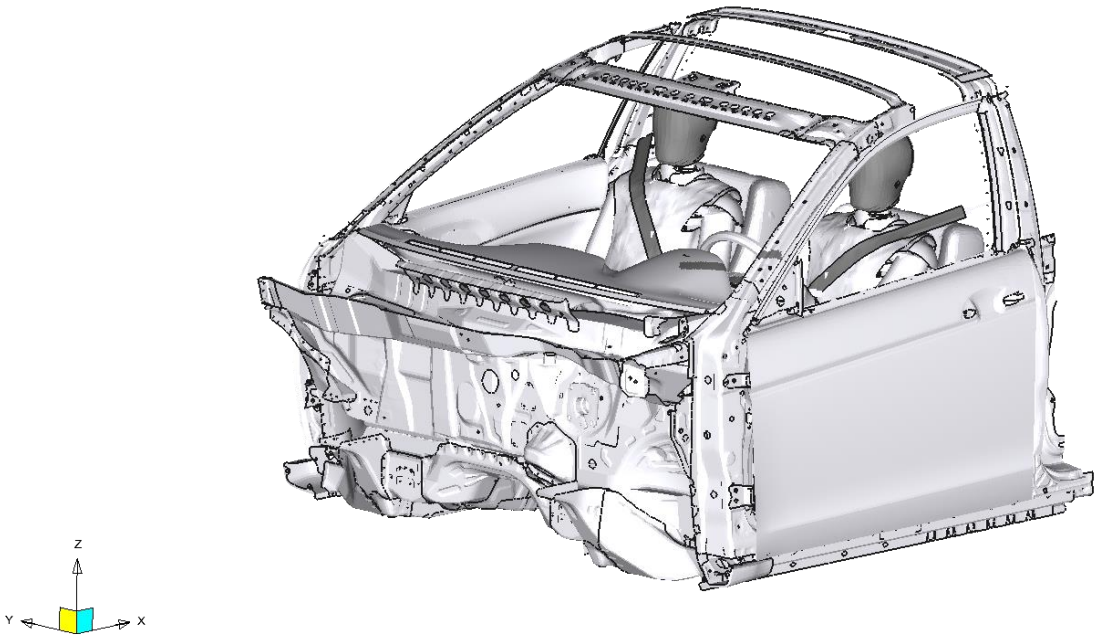


Bild 30: Mögliches Submodell aus dem Honda Accord von (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020) für die Optimierung des Insassenschutzes

5. Praktische Anwendung von Submodellen

Nachdem der theoretische Hintergrund zu Submodellen, deren Erstellung und Einsatz erklärt ist, folgt nun die praktische Anwendung dieser anhand ausgewählter Beispiele. Zuerst wird eine Multi-Level-Optimierung mit einem relativ kleinen Fahrzeugmodell durchgeführt. Damit soll gezeigt werden, wie die Umsetzung einer solchen Optimierungsaufgabe aussehen kann und welche Rolle das Submodell dabei spielt.

In einem zweiten Beispiel wird eine industrienahe Optimierungsaufgabe, mit einem großen Fahrzeugmodell, aufgestellt. Dazu ist ein DoE mit dem Submodell durchgeführt worden, auf Basis dessen die Metamodelle trainiert werden. Anhand dieser Metamodelle erfolgt die Optimierung.

5.1 Submodellbasierte Multi-Level-Optimierung von Fahrzeug 1

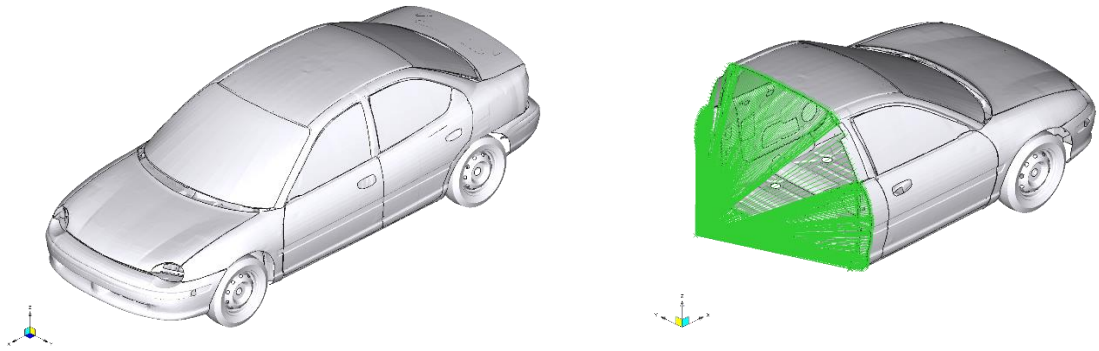
Basis für die erste Optimierung ist ein Modell des Dodge Neon mit 270.000 Elementen, welches von der NHTSA (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020) veröffentlicht wurde.

Für die Optimierung wird ein Frontcrash mit voller Überdeckung untersucht. Dazu kollidiert das Fahrzeug bei 56 km/h mit einer starren Barriere. Dies ist der Standardlastfall des Modells und entspricht dem US-NCAP Frontcrash.

In Bild 31a ist das Gesamtmodell in der isometrischen Ansicht dargestellt. Die Rechenzeit liegt mit 32 CPU[6] bei unter einer Stunde. So können in relativ kurzer Zeit Ergebnisse erzeugt werden, was für die Prinzipuntersuchung der Methodik gut geeignet ist. Das in der Optimierung genutzte Submodell wird mit dem CBS-Verfahren und KMA als Randbedingung automatisch erzeugt und hat in etwa 170.000 Elemente und damit noch 63 % des Gesamtmodells. Die Dauer für eine Simulation verringert sich dabei um ca. 25 %. Als Validierung wurde das Verfahren aus Abschnitt 3.4.2 verwendet, der Ergebniswert liegt bei ca. 0,89.

Die beiden miteinander zu vergleichenden Optimierungen sind eine direkte Optimierung mit dem Gesamtmodell und eine Multi-Level-Optimierung mit Gesamt- und Submodell. In beiden Fällen werden die gleichen Entwurfsvariablen und Restriktionen, sowie die gleiche Zielfunktion verwendet.

[6] Für dieses Beispiel wurde ein AMD EPYC 7452 Prozessor auf dem PLEIADES Rechencluster mit 32 Kernen bei 2,35 GHz verwendet (siehe PLEIADES Hardware 2022).



(a) (b)

Bild 31: Gesamt- (a) und Submodell (b) des Dodge Neon von (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

5.1.1 Optimierungsaufgabe

Es werden sechs Blechdicken im Vorderwagen variiert. Die Entwurfsvariablen sind die Wanddicken der Längsträger, der Stoßstange und der Spritzschutzwand, wie in Bild 32 zu sehen.

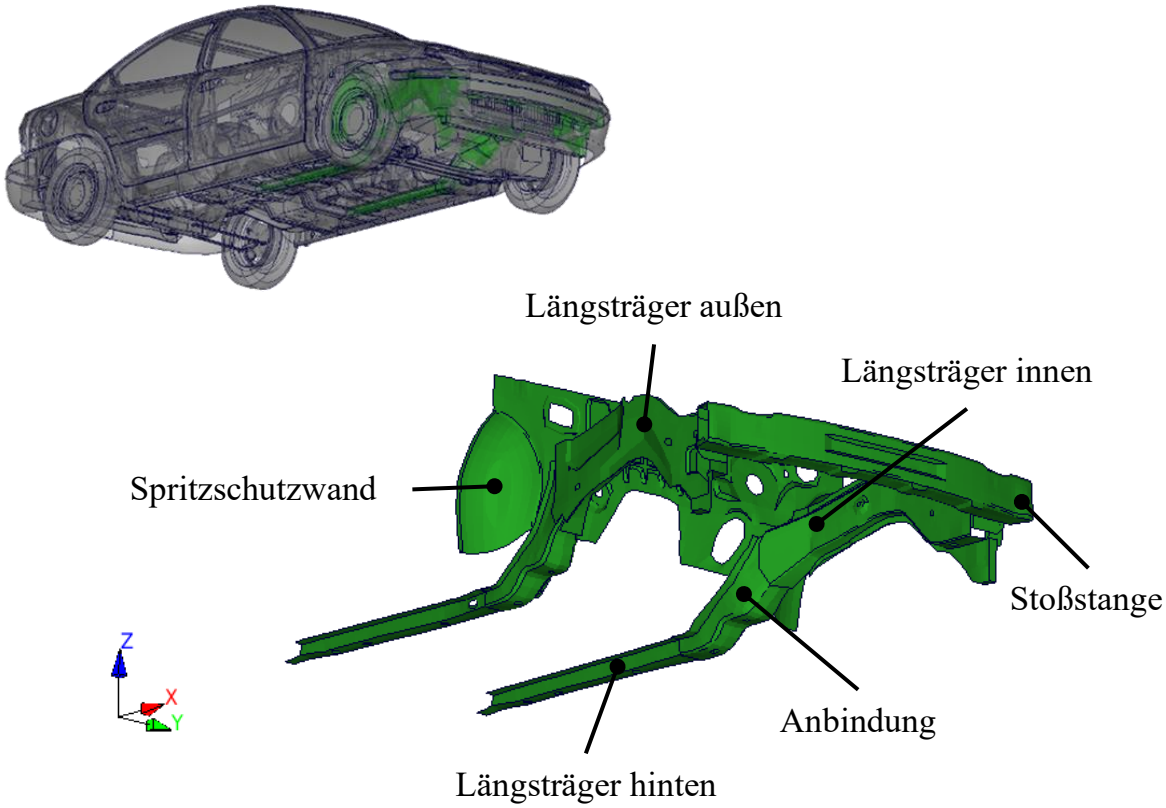


Bild 32: Darstellung der Entwurfsvariablen im Dodge Neon

Ziel der Optimierung ist die Minimierung der maximalen Kraft auf die Barriere unter Berücksichtigung zweier Deformations- und einer Masserestriktion. Die Türausschnitte (siehe Bild 33) dürfen nicht über 50 mm deformiert werden, sodass die Türen weiterhin zu öffnen sind. Ermittelt wird dieser Wert anhand zweier FE-Knoten im Türrahmen. Der Abstand vor und nach dem Crash wird bestimmt. Aus der Differenz ergibt sich die Deformation. Diese werden für die Fahrer- und Beifahrerseite getrennt betrachtet.

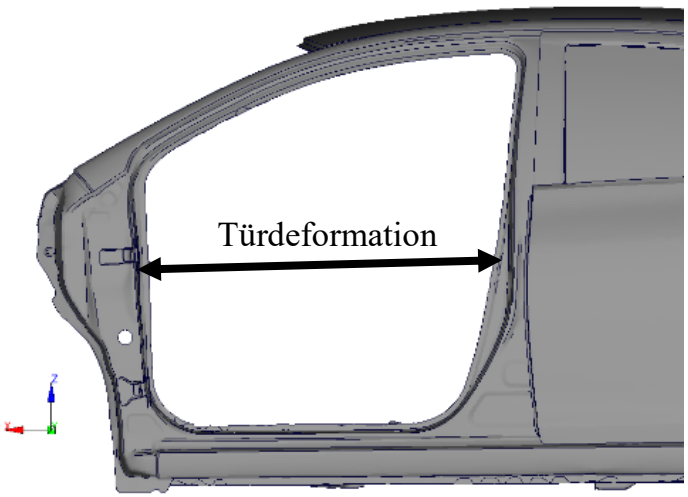


Bild 33: Bestimmung der Türdeformation über den Abstand zweier Knoten in x-Richtung

Die dritte Restriktion ist die Masse, welche sich um ca. 0,5 %, von 1334 auf 1340 kg erhöhen darf. Durch das Zulassen einer geringen Massenerhöhung wird dem Optimierungsalgorithmus mehr Spielraum in der Variation der Entwurfsvariablen ermöglicht.

Der Kraftverlauf des initialen Entwurfs ist in Bild 34 dargestellt. Die dort eingezeichnete gestrichelte Linie stellt den optimalen Verlauf dar, welcher ein Rechteck bildet. So ergibt sich die maximale Fläche unter der Kurve, bei möglichst niedriger maximaler Kraft. In der Realität ist ein senkrechter Kraftanstieg nicht möglich, sodass ein Rechteck nur angenähert werden kann. Durch die Reduktion der Peaks des realen Kraftverlaufs wird die Kurve nach und nach an den optimalen Verlauf angeglichen.

In Tabelle 4 sind die Zielfunktion und Restriktionen, sowie die Entwurfsvariablen zusammengefasst.

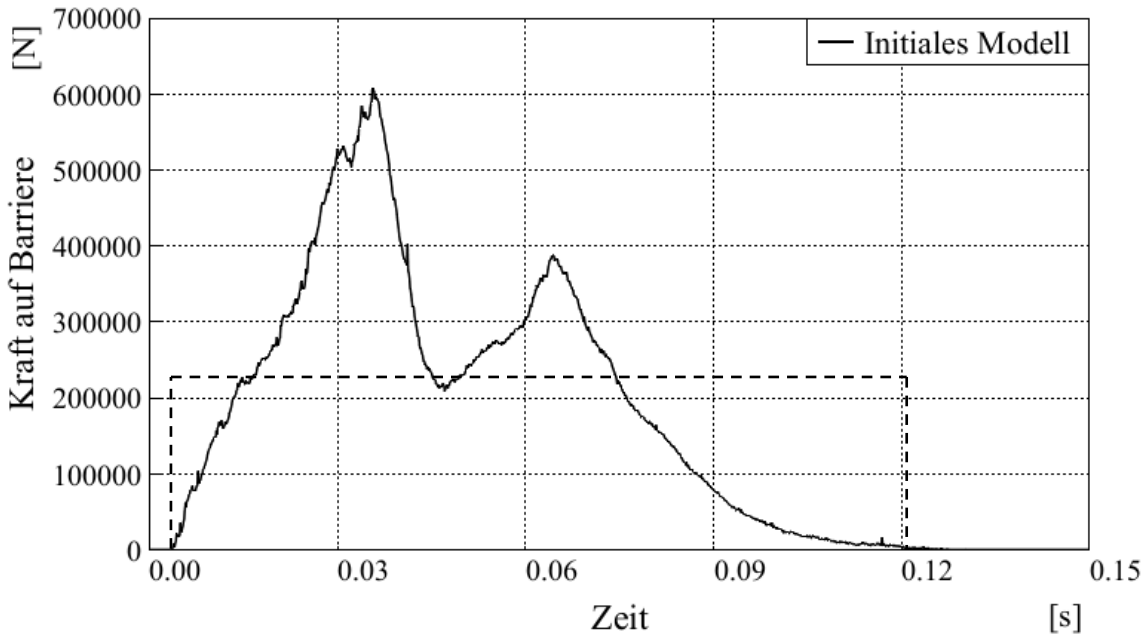


Bild 34: Realer Verlauf der Kraft auf die Barriere (durchgezogene Linie) und optimaler Kraftverlauf (gestrichelte Linie)

Tabelle 4: Übersicht über die Entwurfsvariablen, Zielfunktion und Restriktionen des Dodge Neon

	Bezeichnung	**Initialer Wert**	**Bedingung**
Entwurfs-variablen	Stoßstange	1,956 mm	1,0…3,0 mm
	Spritzschutzwand	0,735 mm	0,2…1,5 mm
	Längsträger außen	1,900 mm	1,0…3,0 mm
	Längsträger innen	1,900 mm	1,0…3,0 mm
	Längsträger hinten	1,156 mm	0,5…2,0 mm
	Anbindung	1,900 mm	1,0…3,0 mm
Zielfunktion	Maximale Kraft auf die Barriere	604270 N	-
Restrik-tionen	Türdeformation Fahrerseite	110,2 mm	< 50 mm
	Türdeformation Beifahrerseite	89,4 mm	< 50 mm
	Masse	1334 kg	< 1340 kg

5.1.2 Validierung des Modells im Entwurfsraum mit DIFFCRASH®

Anhand des hier vorgestellten Modells wird die in Abschnitt 4.2 dargelegte Möglichkeit der Validierung des Modells mithilfe eines DoE getestet. Dazu werden Simulationen des Submodells mit ihrem zugehörigen Gesamtmodell an neun Stellen im definierten Entwurfsraum durchgeführt. In Tabelle 5 sind die Ergebnisse der PCA mit DIFFCRASH® aufgezeigt. Als Maß für die Lage im Entwurfsraum ist links in der

Tabelle der Betrag des Abstands der Entwurfsvariablen aufgeführt. Dieser Abstand d berechnet sich nach der Formel

$$d = \sqrt{\sum_{1}^{n_{DV}} DV_i^2} \tag{2}$$

mit

n_{DV}: Anzahl der Entwurfsvariablen

DV: Wert der Entwurfsvariablen

Die Bewertungen aus DIFFCRASH® bleiben über den Entwurfsraum verteilt bei relativ niedrigen Werten und somit guten Übereinstimmungen der Submodelle zu den jeweiligen Gesamtmodellen.

Tabelle 5: Beispielhafte Validierung der Submodelle durch DIFFCRASH® im Entwurfsraum mit dem Betrag des Abstands der Entwurfsvariablen als Maß für die Lage im Entwurfsraum

Abstandsbetrag	**Submodellbewertung**
0,88	6,60
0,94	5,82
0,99	6,66
1,09	6,54
1,28	4,66
1,34	4,53
1,41	6,90
1,41	6,58
1,51	5,16

5.1.3 Aufbau und Durchführung der Multi-Level-Optimierung

Eine Multi-Level-Optimierung wird wie in 2.6.2 dargestellt aufgebaut. Dazu sind zwei Instanzen der Optimierungssoftware verschachtelt auszuführen. Als Optimierungstool wird LS-OPT® genutzt, da dies die Möglichkeit bietet, ohne weitere Skripte eine Multi-Level-Optimierung durchzuführen. In anderen Optimierungstools ist eine solche Prozedur ebenfalls umsetzbar, allerdings müsste diese manuell erstellt werden, da z. B. Optimus keinen vorgefertigten Optimierungsablauf für eine Multi-Level-Optimierung anbietet. Außerdem bietet LS-OPT® eine direkte Schnittstelle zum FE-*Solver* LS-DYNA®, da beides vom gleichen Hersteller stammt.

In der äußeren Schleife (siehe Bild 35) werden die sechs genannten Entwurfsvariablen definiert und das Gesamtmodell verwendet. Danach wird im *Sampling* die Populationsgröße für eine Iteration definiert, da ein genetischer Optimierungsalgorithmus verwendet wird. Die äußere Schleife dient vor allem zur Validierung des Submodells in regelmäßigen Abständen.

Es folgt der Aufruf der inneren Schleife (*innerLoop*), dazu wird eine weitere Instanz von LS-OPT® gestartet. Das Ergebnis der inneren Optimierungsschleife wird zurück an die äußere Schleife übertragen und durch eine Simulation mit dem Gesamtmodell bestätigt. Es folgt die Überprüfung der Abbruchkriterien und eine neue Iteration wird gestartet oder das Optimum ausgegeben.

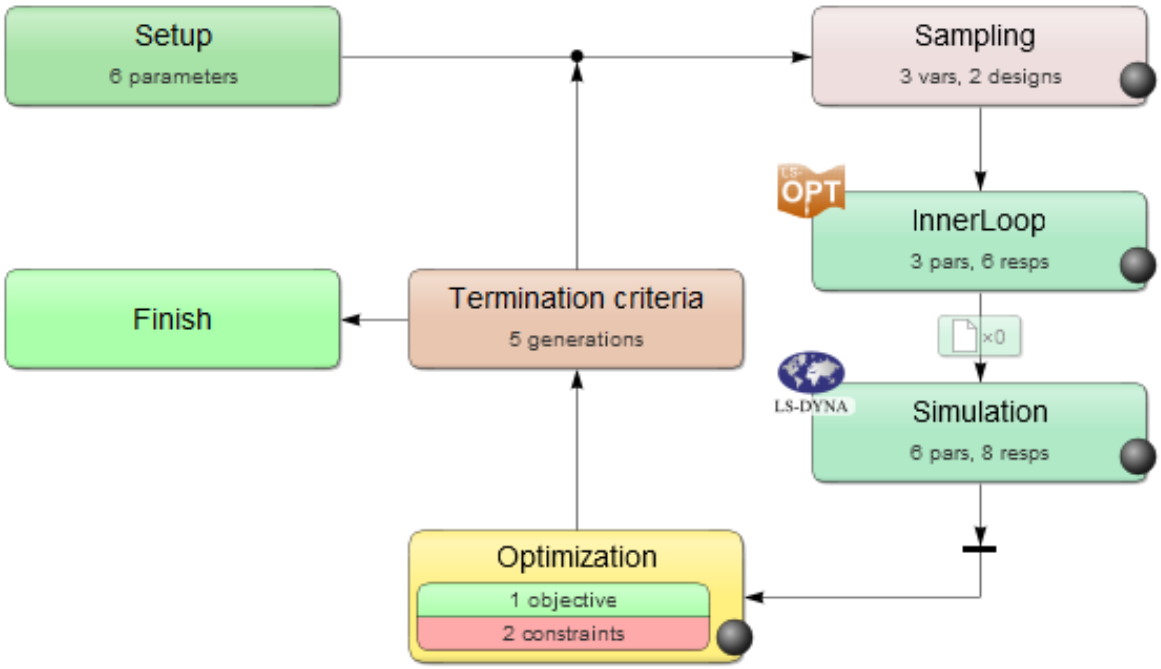


Bild 35: Äußere Schleife der Multi-Level-Optimierung in LS-OPT®

Die innere Optimierungsschleife ist in Bild 36 zu sehen und ähnelt vom Aufbau der äußeren Schleife. Hier wird keine weitere Instanz von LS-OPT® aufgerufen, sondern nur eine Simulation mit einem Submodell. Außerdem erfolgt die Erstellung von Metamodellen, aus den im *Sampling* festgelegten Stützstellen der Iteration. Anhand der Metamodelle wird die Position des Optimums vorhergesagt. Basierend auf den Metamodellen wird durch die in der *Domain Reduction* verwendete *Sequential Response Surface Method* (Kurz: SRSM) der Entwurfsraum eingegrenzt. Der Algorithmus kann dabei die Größe des Entwurfsraums anpassen (*Zoom*), die Position verschieben (*Pan*) oder beides zusammen (*Pan & Zoom*) (vgl. STANDER ET AL. 2020, S. 600 ff).

Wichtig ist, dass in jeder der beiden Schleifen drei der sechs Entwurfsvariablen variiert werden und die übrigen jeweils konstant bleiben. In der inneren Schleife werden die Längsträger innen und außen, sowie die Stoßstange variiert. Die entsprechend anderen Entwurfsvariablen werden in der äußeren Schleife verwendet. Durch dieses Vorgehen werden weniger Stützstellen für die Metamodelle benötigt. In der inneren Schleife werden die Entwurfsvariablen, die unmittelbar an der Barriere liegen variiert, also die Stoßstange und die beiden vorderen Längsträgerbleche.

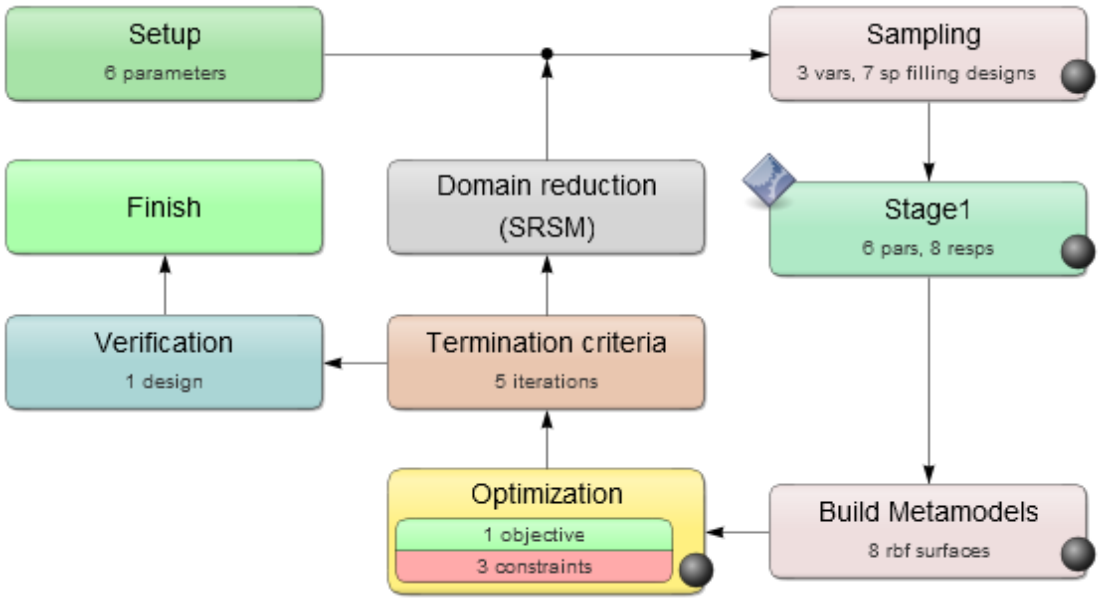


Bild 36: Innere Schleife der Multi-Level-Optimierung mit LS-OPT®

Für die Optimierungsparameter werden die in Tabelle 6 dargestellten Werte verwendet. Dabei sind für die meisten Parameter der gewählten Optimierungsalgorithmen die von der Software vorgeschlagenen Standardwerte in Verwendung. Die maximale Gesamtzahl an Simulationen sollte in etwa mit denen der Vergleichsoptimierung übereinstimmen. Für diese Vergleichsoptimierung mit einer Ebene und dem Gesamtmodell, werden die Einstellungen der inneren Schleife genutzt. Die Iterationen werden auf maximal 30 eingestellt, mit jeweils 10 Stützstellen. Dadurch ergeben sich für die Multi-Level-Optimierung maximal fünf Iterationen in der äußeren Schleife, mit jeweils fünf Iterationen in der Inneren. In der äußeren Schleife werden zwei Stützstellen erstellt, in der inneren sieben. Damit können in der Multi-Level-Optimierung maximal 355 Stützstellen[7] berechnet werden, in der Vergleichsoptimierung sind es maximal 300 Stützstellen[8].

[7] Berechnung: 5 Iterationen außen $*$ 2 Stützstellen $*$ 5 Iterationen innen $*$ 7 Stützstellen $+$ 5 Gesamtmodellsimulationen $= 355$

[8] Berechnung: 30 Iterationen $*$ 10 Stützstellen $= 300$

Tabelle 6: Optimierungsparameter der Multi-Level-Optimierung

Parameter	Wert äußere Schleife	Wert innere Schleife
Metamodell	-	RBFN
Versuchsplan	-	Space Filling
Anzahl Stützstellen	2	7
Anzahl Iterationen	5	5
Optimierungsalgorithmus	Genetic Algorithm	Simulated Annealing
Parameter des Algorithmus	Standardwerte	Standardwerte

5.1.4 Ergebnisse der Optimierung

Bei der Vergleichsoptimierung mit dem Gesamtmodell wurden bis zur Erfüllung des Konvergenzkriteriums 61 Simulationen durchgeführt. Mit der Multi-Level-Optimierung wurden 56 Submodellsimulationen und 4 Gesamtmodellsimulationen benötigt. In der Optimierung wurden bei der Multi-Level-Optimierung ca. 21 % der CPU-Stunden eingespart, gegenüber der Optimierung mit dem Gesamtmodell.

Die Optimierung hat gezeigt, dass mit dem Submodell ein ähnliches Ergebnis zum Gesamtmodell erreicht werden kann (siehe Tabelle 7). Die dort aufgezeigten Ergebnisse sind durch eine Simulation des Gesamtmodells validiert. Bei Betrachtung der Ergebnisse, sind für beide Optimierungen, abgesehen von der Spritzschutzwand und dem äußeren Längsträgerblech, ähnliche Entwurfsvariablen im Optimum gefunden worden. Auch die maximale Kraft auf die Barriere hat in beiden Fällen ein sehr ähnliches Peak. In Bild 37 sind vergleichend die Kraftverläufe dargestellt. Beide Optima haben die maximale Kraft im gleichen Zeitschritt. Das zweite Peak tritt in beiden Fällen früher auf als im initialen Modell. Bei der Vergleichsoptimierung hat es sich auch erhöht, bei der mit dem Submodell verringert.

Beim Vergleich der anderen Ergebnisse in Tabelle 7 fällt weiterhin auf, dass die Restriktion der Türdeformation lediglich beim Submodell auf der Beifahrerseite eingehalten wird. Dafür fällt die Verletzung der Restriktion auf der Fahrerseite etwas höher aus, als bei der Vergleichsoptimierung. In beiden Optima konnte die Türdeformation jedoch deutlich reduziert werden, gegenüber dem initialen Entwurf. Die Masse verletzt in beiden Optimierungen knapp die Restriktion. Dies liegt an dem Optimierungsablauf der inneren Schleife. Nach der letzten Iteration der inneren Schleife wird

das Optimum anhand der aufgebauten Metamodelle vorhergesagt und durch die in Bild 36 als *Verification* dargestellte Simulation überprüft (vgl. STANDER ET AL. 2020, S. 26). Diese neu berechnete Stützstelle kann dann Restriktionen, wie in diesem Falle, verletzten.

Tabelle 7: Ergebnisse der Multi-Level-Optimierung des Dodge Neon, validiert durch eine Gesamtmodellsimulation, im Vergleich zu einer Optimierung mit dem Gesamtmodell. Verletzte Restriktionen sind rot markiert

	Bezeichnung	Initial	Gesamt-modell	Sub-modell
Entwurfsvariablen	Stoßstange [mm]	1,96	3,00	2,90
	Spritzschutzwand [mm]	0,74	1,50	1,29
	Längsträger außen [mm]	1,90	1,00	1,90
	Längsträger innen [mm]	1,90	3,00	2,64
	Längsträger hinten [mm]	1,16	2,00	2,00
	Anbindung [mm]	1,90	1,00	1,00
	Maximale Kraft auf die Barriere [N]	604270	480964	490762
Restrik-tionen	Türdeformation Fahrerseite [mm]	110	65	68
	Türdeformation Beifahrerseite [mm]	89	63	39
	Masse [kg]	1334	1343	1342
	CPU-Stunden	-	2602	2048

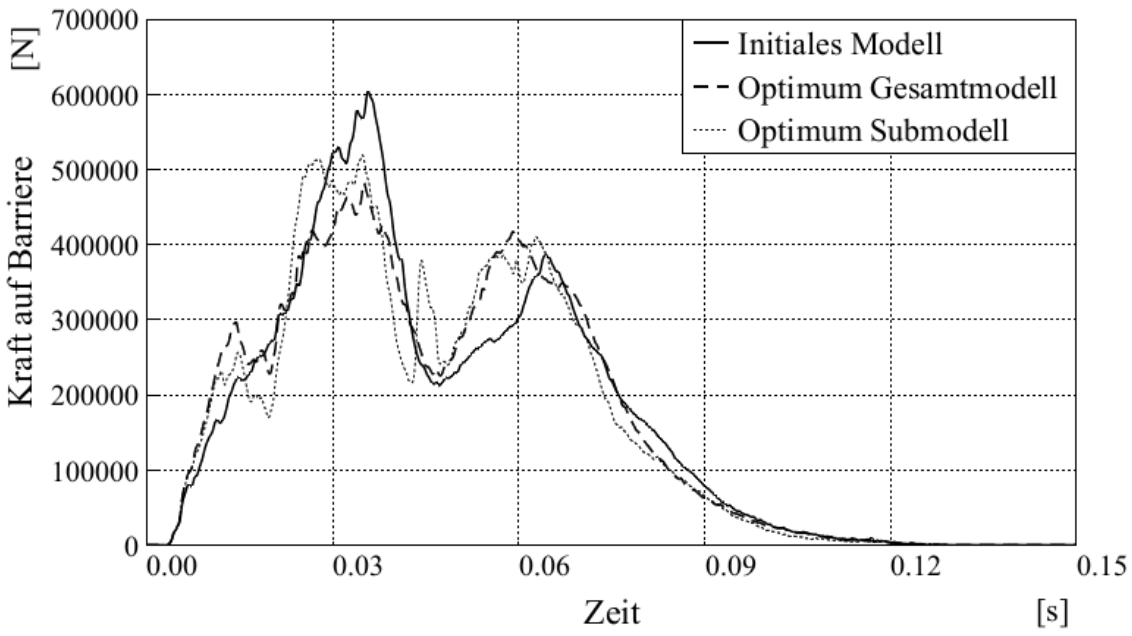
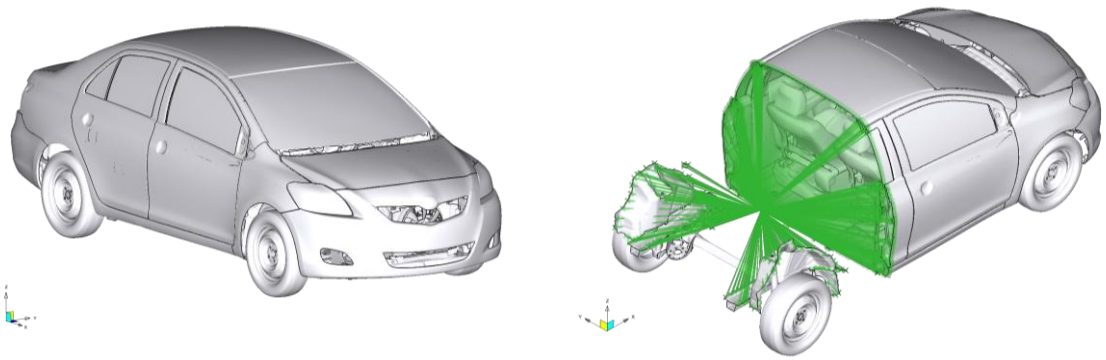


Bild 37: Verläufe der Kraft auf die Barriere des initialen Modells und der beiden Optima. Die Verläufe wurden zur besseren Erkennbarkeit mit einem CFC 600 Filter geglättet

5.2 Submodellbasierte Optimierung von Fahrzeug 2

In einer zweiten Optimierung soll ein industrienahes Problem behandelt werden. Dazu wird das Modell des Toyota Yaris von (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020) verwendet (siehe Bild 38a). Mit ungefähr 1,5 Mio. Elementen ist dieses Modell wesentlich detaillierter als der in Abschnitt 5.1 verwendete Dodge Neon. Das Submodell für Frontalaufpralle für die Optimierung wird im Vorhinein erstellt. Als Generierungsverfahren wird ebenfalls CBS gewählt, mit der Auswahl pro Element. Die Box wird über die gesamte Breite und Höhe des Fahrzeugs erstellt und reicht bis 2000 mm der globalen x-Koordinatenachse des Modells. Zusätzlich sind alle Bauteile des hinteren Fahrwerks in einer Gruppe zusammengefasst und dem Submodell hinzugefügt. Das generierte Submodell mit Ersatzmasse ist Bild 38b zu entnehmen und enthält in etwa 76 % der Elemente des Gesamtmodells.



(a) (b)

Bild 38: Gesamt- (a) und Submodell (b) des Toyota Yaris von (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020)

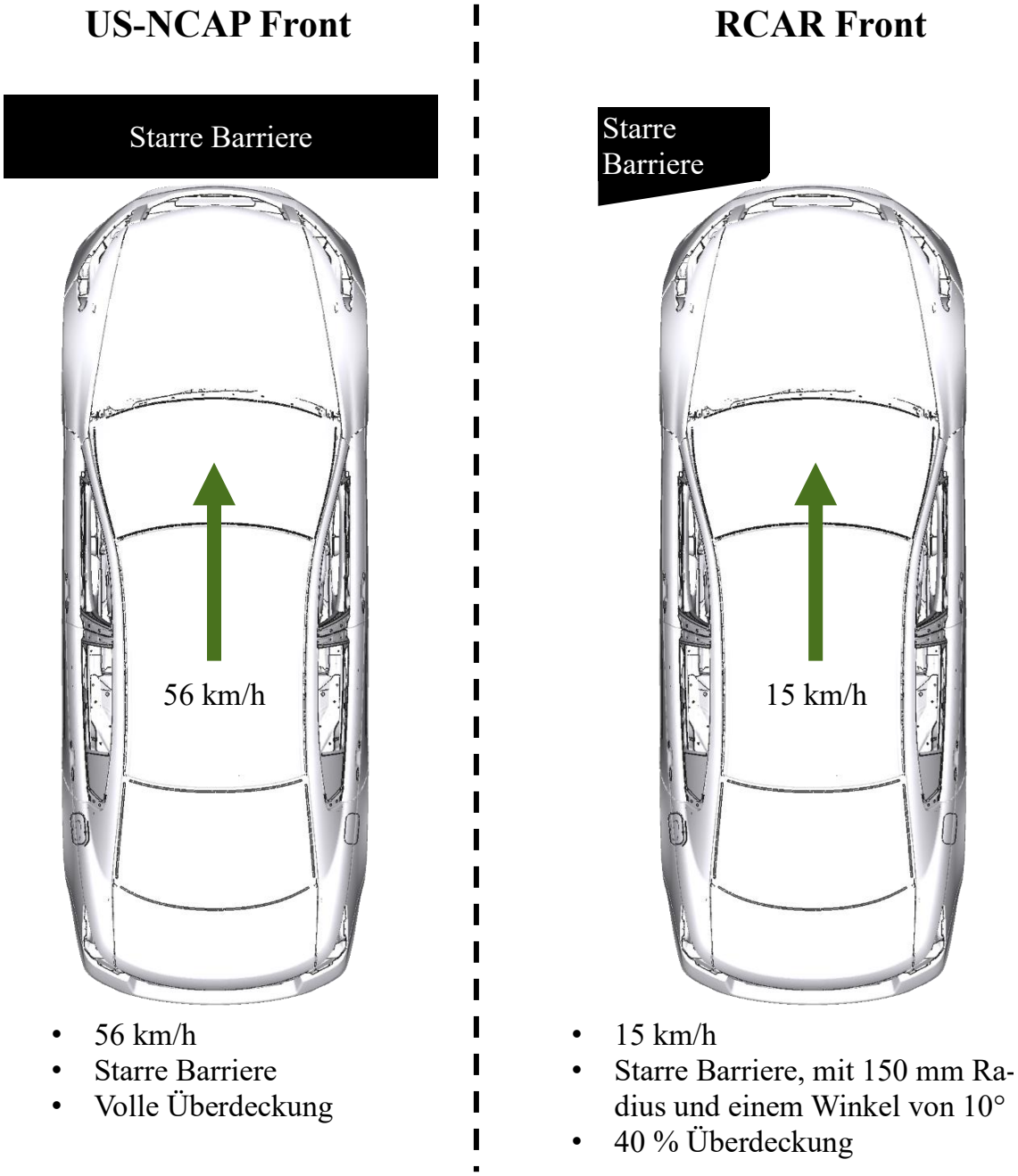
5.2.1 Lastfälle

Es sollen zwei Frontalaufprall-Lastfälle untersucht werden. Dazu werden ein Hochgeschwindigkeits- und ein Versicherungslastfall gewählt. Der Hochgeschwindigkeitslastfall entspricht dem Standardlastfall des Modells, sowie es von (NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION 2020) herunterzuladen ist. Für den US-NCAP Front Lastfall mit voller Überdeckung der Barriere fährt das Modell mit 56 km/h frontal gegen eine starre Barriere (siehe Bild 39a). Beim Versicherungslastfall RCAR Front (RCAR 2011a, S. 11) findet die Kollision mit 15 km/h statt. Die Barriere ist ebenfalls starr und überdeckt zu 40 % mit dem Fahrzeug. Außerdem hat die Kante der Barriere einen Radius von 150 mm und steht mit einem Winkel von 10° zur Fahrzeugfront (siehe Bild 39b).

Wie bereits erwähnt ist der US-NCAP Lastfall der Standardlastfall des Modells und muss daher abgesehen von der Submodellerstellung nicht verändert werden. Für den Versicherungslastfall ist die Geschwindigkeit des Modells anzupassen und die Barriere auszutauschen. Die Abmessungen der Barriere sind in (RCAR 2011a) detailliert dargestellt. Im Gegensatz zur starren Wand aus dem NCAP Lastfall wird die Barriere für den RCAR-Lastfall aus *Shell*-Elementen aufgebaut. Die Elemente erhalten die `MAT_RIGID` Materialkarte, sodass sie nicht verformbar sind. Dadurch, dass hier *Shell*-Elemente verwendet werden, sind im Fahrzeugmodell Anpassungen der Kontaktbedingung notwendig. Im Standardsetup besteht die Barriere aus einer *Rigid Wall*, für diese wird der Kontakt automatisch vom *Solver* LS-DYNA® bestimmt. Die neue RCAR-Barriere muss eine zusätzliche Kontaktebedingung erhalten.

Zuerst wurde die neue Barriere in die allgemeine Kontaktbedingung des Fahrzeugmodells aufgenommen. Dies führte zur Durchdringung der Barriere durch das Modell, wie Bild 40a zeigt. Aufgrund der Durchdringung wird das Abgleiten des Fahrzeugs an der Barriere verhindert und die Ergebnisse verfälscht. Durch Erstellen einer neuen Kontaktbedingung wird diese Problematik verhindert. Der verwendete Kontakt lautet `CONTACT_AUTOMATIC_SURFACE_TO_SURFACE_ID`. Als Gruppen für die Kontaktbedingung wurden eine das Fahrzeug umfassende Gruppe und die Barriere ausgewählt. Wie in Bild 40b zu sehen ist, tritt keine Durchdringung mehr auf und das Modell kann an der Barriere entlanggleiten.

In beiden Lastfällen werden die gleichen Entwurfsvariablen verwendet, sodass die Modelle einander entsprechen. Als Entwurfsvariablen werden die Wanddicken der in Tabelle 8 aufgezeigten Bauteile gewählt. Dort sind ebenfalls die initialen Blechdicken und Nebenbedingungen für die Entwurfsvariablen dargestellt. Bild 41 zeigt die zugehörigen Bauteile im Fahrzeugmodell.



- 56 km/h
- Starre Barriere
- Volle Überdeckung

(a)

- 15 km/h
- Starre Barriere, mit 150 mm Radius und einem Winkel von 10°
- 40 % Überdeckung

(b)

Bild 39: Zu untersuchende Lastfälle nach (CARHS.TRAINING GMBH 2019, S. 22–23) für die Optimierung von Fahrzeug 2. (a) Hochgeschwindigkeitslastfall gegen eine starre Barriere; (b) Versicherungslastfall gegen starre Barriere mit 40% Überdeckung

(a)

(b)

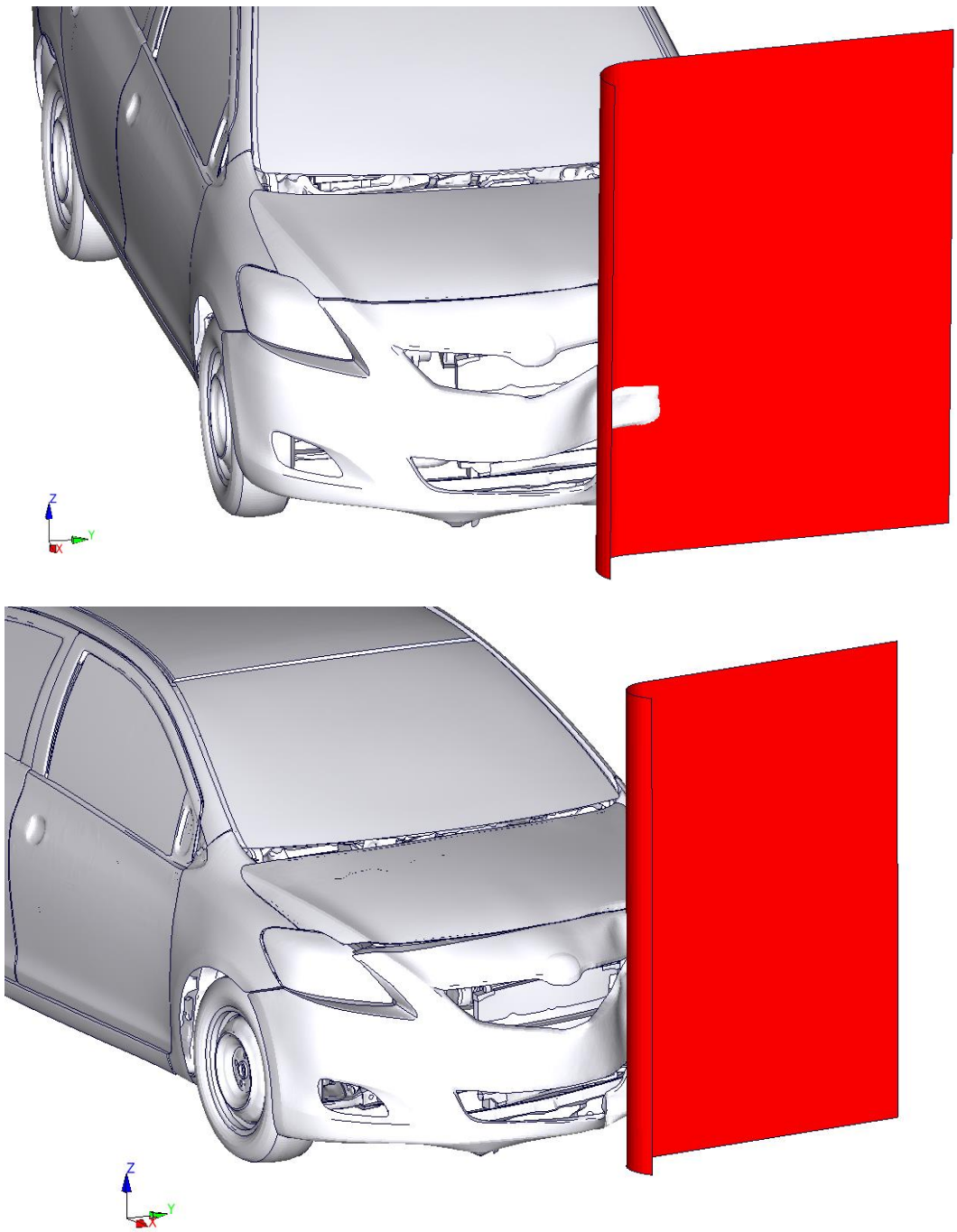


Bild 40: Korrektur der Kontaktbedingungen für den RCAR-Lastfall: (a) Durchdringung der RCAR-Barriere durch das Fahrzeugmodell; (b) korrigiertes Modell ohne Durchdringung

Tabelle 8: Entwurfsvariablen für die Optimierung von Fahrzeug 2 und ihre initialen Werte mit Nebenbedingungen

Gruppe	Bezeichnung	Initial Blechdicke [mm]	Nebenbedingung [mm]
Crashbox	Crashbox außen	1,889	0,6…2,0
	Crashbox innen	1,300	0,6…2,0
Längsträger	Längsträger außen	1,673	1,0…3,0
	Längsträger innen	1,869	1,0…3,0
	Anbauteil 1	2,004	1,0…3,0
	Anbauteil 2	1,737	1,0…3,0
	Anbauteil 3	1,767	1,0…3,0
Rest	A-Säule innen	0,843	0,6…1,5
	A-Säule außen	1,400	1,0…2,0
	Spritzschutzwand	0,747	0,6…1,5

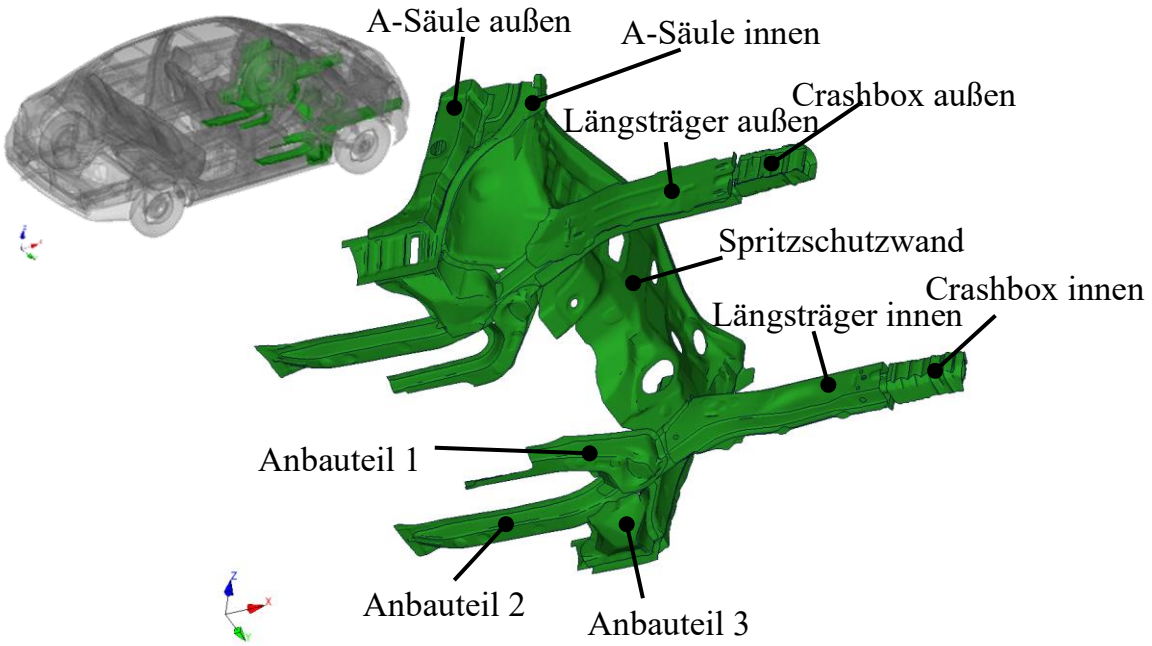


Bild 41: Darstellung der Entwurfsvariablen im Fahrzeugmodell

5.2.2 Optimierungsaufgabe

Die beiden zuvor vorgestellten Lastfälle sollen parallel ausgeführt werden. In beiden Lastfällen sind einige Restriktionen einzuhalten. Eine Übersicht der gewählten Restriktionen ist in Tabelle 9 zu finden. Es erfolgte eine Aufteilung der Restriktionen nach den Lastfällen. Viele der Restriktionen werden getrennt voneinander für die Fahrer- (Kurz: FS) und Beifahrerseite (Kurz: BFS) betrachtet. Wie den initialen Werten zu

entnehmen ist, ist das Modell nicht symmetrisch, wodurch die getrennte Betrachtung der jeweiligen Seite sinnvoll ist, auch wenn die Restriktionen dieselben sind.

Tabelle 9: Restriktionen für die Optimierung von Fahrzeug 2

Lastfall	Bezeichnung	Initialer Wert	Restriktion
	Masse	1253 kg	< 1265 kg
US-NCAP	OLC[a]	38,15 g	< 35 g
	Türdeformation Fahrerseite (FS)	6,12 mm	< 25 mm
	Türdeformation Beifahrerseite (BFS)	5,60 mm	< 25 mm
	Intrusion Spritzschutzwand FS	140 mm	< 100 mm
	Intrusion Spritzschutzwand BFS	131 mm	< 100 mm
RCAR	Max. plast. Dehnung Längsträger	6,3 %	< 3 %
	Max. plast. Dehnung Spritzschutzwand	6,3 %	< 6 %
	Max. plast. Dehnung A-Säule	0,8 %	< 1 %
	Intrusion Spritzschutzwand	|2,67| mm	< 3 mm
	Längsträger Kompression FS	|0,17| mm	< 3 mm
	Längsträger Kompression BFS	|0,07| mm	< 3 mm
	Längsträger Höhendifferenz FS	|2,34| mm	< 3 mm
	Längsträger Höhendifferenz BFS	|5,70| mm	< 3 mm
	Längsträger diagonal FS	|4,92| mm	< 3 mm
	Längsträger diagonal BFS	|12,98| mm	< 3 mm
	Längsträger Distanz	|10,91| mm	< 3 mm

[a] *Occupant Load Criterion nach (KÜBLER ET AL. 2009)*

Bei der Masse wird eine Erhöhung von ca. 1 % gewährt, damit der Optimierungsalgorithmus Raum für Änderungen hat.

Im Hochgeschwindigkeitslastfall wird ein Ersatzkriterium für die auf den Insassen wirkenden Beschleunigungen ausgewertet. Das sogenannte *Occupant Load Criterion* (Kurz: OLC) schätzt die durch die Rückhaltesysteme verringerte Beschleunigung des Insassen anhand von Knotenbeschleunigungen im Modell ab. Dadurch ist es möglich, die Verletzungsgefahr der Insassen abzuschätzen, ohne einen aufwendigen Dummy zu simulieren. Zur Ermittlung des OLC-Wertes wird der Geschwindigkeitsverlauf eines Knotens im Fahrzeug untersucht (siehe Bild 42). Durch Integration des Bereichs zwischen einer Geraden (Bereich I) und der Geschwindigkeitskurve wird der Weg bis zur Ankopplung des Insassen an das Fahrzeug und damit die verzögert wirkende negative Beschleunigung berücksichtigt. Diese verzögerte Ankopplung kommt z. B. durch die Gurtlose zustande. Sie beschreibt die Weglänge, bis der Sitzgurt den Insassen hält. Erreicht das Integral die Länge bis zur Ankopplung, beginnt die Abbremsphase des

Insassen, Bereich II in Bild 42. Dieser wird so gewählt, dass das Integral zwischen ihr und dem Geschwindigkeitsprofil der Weglänge der wirkenden Rückhaltesysteme entspricht. Aus der Steigung der Geraden ergibt sich der OLC-Wert, welcher in Vielfachen der Erdbeschleunigung angegeben wird. (vgl. KÜBLER ET AL. 2009)

Für die Auswertung des OLC-Wertes in dieser Optimierung wird eine Länge bis zur Ankopplung von 65 mm und ein Abbremsweg von 235 mm angenommen, wie in (KÜBLER ET AL. 2009, S. 427) beschrieben ist. Es wird der OLC-Wert an sechs Knoten im Modell (siehe Bild 43) bestimmt und gemittelt.

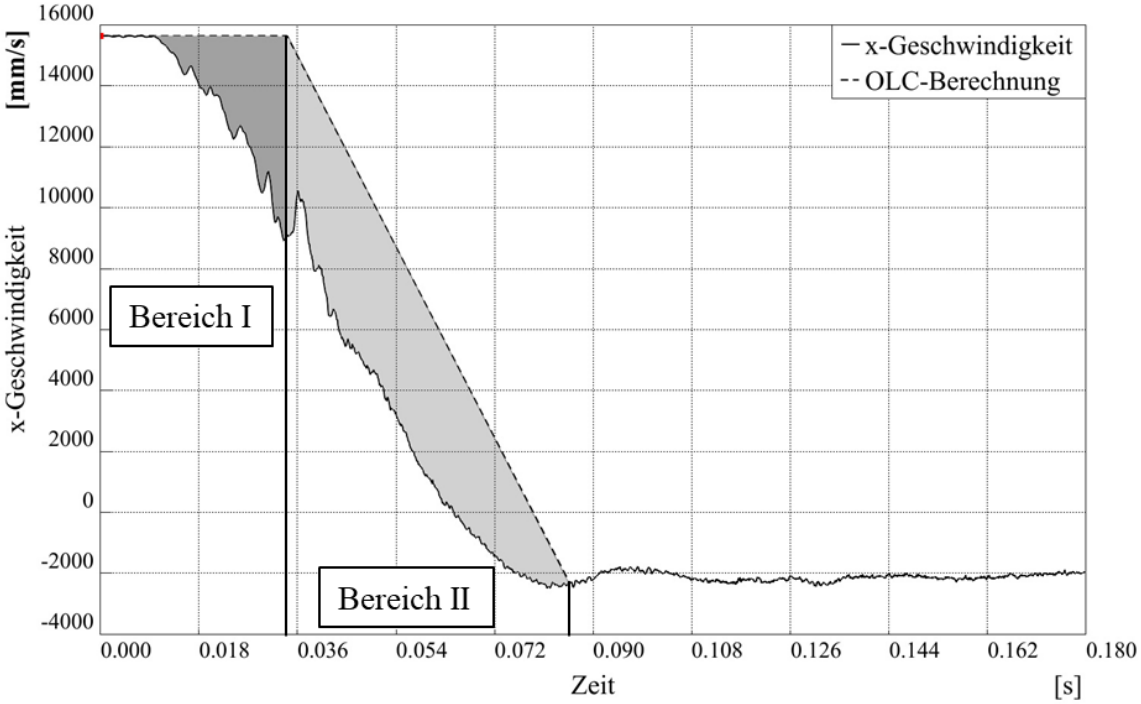


Bild 42: Bestimmung des OLC-Wertes anhand des Geschwindigkeitsverlaufs eines ausgewählten Knotens. Die Steigung der gestrichelten Linie in Bereich II ergibt den OLC-Wert, Bereich I repräsentiert die Verzögerung, bis der Insasse an das Fahrzeug ankoppelt

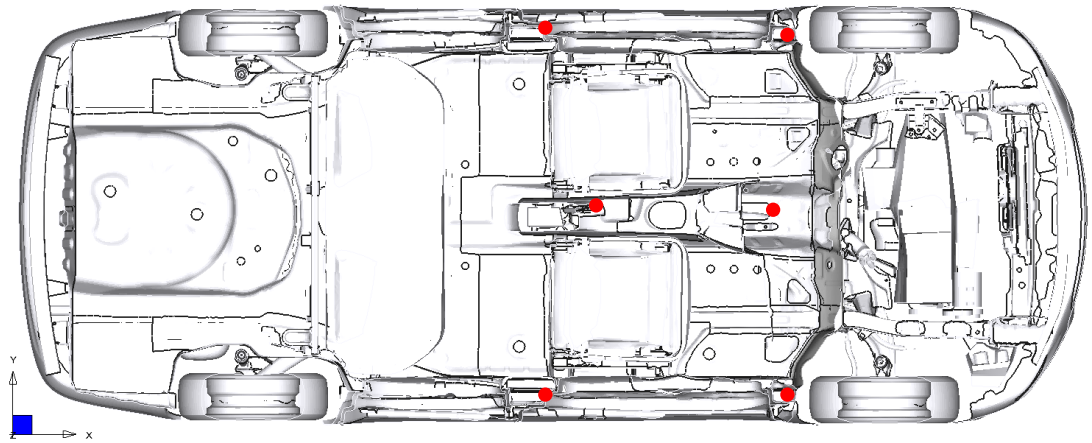


Bild 43: Positionen der Knoten (rot markiert) zur Auswertung der Beschleunigungen

Die zulässige Veränderung des Türrahmens ist so gewählt, dass die Tür nicht eingeklemmt wird und sich auch nach dem Crash noch öffnen lässt. Der Restriktionswert von 25 mm wurde an vorhandenen realen Fahrzeugen gemessen. In der Simulation findet die Auswertung anhand der Abstände zweier Knoten im Türrahmen statt (siehe auch Bild 45 rechts).

Nach (EURO NCAP 2020, S. 6) dürfen sich die Pedale auf der Fahrerseite um bis zu 100 mm nach innen drücken. Bei Überschreiten des Grenzwertes wird die Fahrzeugbewertung reduziert. Bei einer Verformung von 200 mm wird die erhält das Fahrzeug für dieses Kriterium keine Punkte. Daraus wurde die 100 mm Restriktion für die Intrusion I_p der Spritzschutzwand abgeleitet, da an dieser die Pedale befestigt sind. Zur Auswertung dieser Größe wird der Abstand a von allen interessanten Knoten n auf der Wand zu einem Knoten im hinteren Dachbereich vor und nach dem Crash ermittelt. Damit der ausgegebene Wert nicht springen kann, was für den Optimierungsalgorithmus schwierig ist, erfolgt eine Mittelung der Ergebnisse mit der P-Norm. Diese wird für die vorliegende Aufgabe nach der folgenden Gleichung, in Anlehnung an (ALT 2012, S. 14), ermittelt:

$$I_p = \sqrt[p]{\frac{\sum_n \Delta a^p}{n}} \tag{3}$$

Wobei n die Anzahl der Knoten auf der Spritzschutzwand ist und Δa die Änderung des Abstands zwischen den Knoten auf der Spritzschutzwand und dem Knoten im Dach. In Bild 44 sind die ausgewerteten Knoten auf der Spritzschutzwand zu sehen, welche in zwei Gruppen, für den jeweiligen Fußraum, aufgeteilt sind. Als Exponent p wird drei gewählt.

Für den Versicherungslastfall werden drei verschiedene maximale plastische Dehnungen ausgewertet, wie der Tabelle 8 zu entnehmen ist. Die Restriktionen beruhen dabei auf den initialen Werten, da die weiteren Restriktionen indirekt ebenfalls verhindern, dass zu große plastische Dehnungen auftreten. Einzig bei der plastischen Dehnung der Längsträger wird eine niedrigere Grenze gewählt, da diese Bauteile im Versicherungslastfall kritisch sind.

Laut (RCAR 2011b, S. 2) sind plastische Verformungen von bis zu 3 mm zulässig, damit diese nicht als Schaden eingestuft werden. Dieser Restriktionswert wird für alle weiteren geometrischen Restriktionen gewählt. Die Auswertung der Längsträgerverformungen erfolgt wieder über die Abstandsänderungen zweier Knoten, wie sie in Bild 45 dargestellt sind. Für die Höhendifferenz wird die Änderung der z-Komponente

zweier Knoten betrachtet. Anhand der Differenz der Änderung beider Knoten kann die Höhenänderung des Längsträgers bewertet werden.

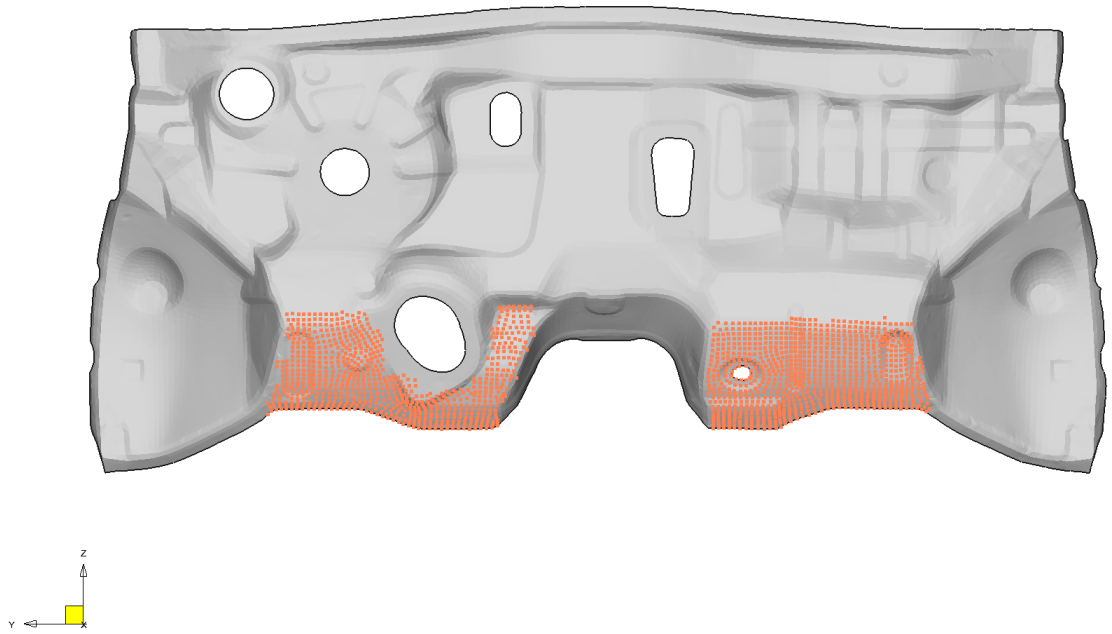


Bild 44: Knoten für die Auswertung der Intrusion der Spritzschutzwand

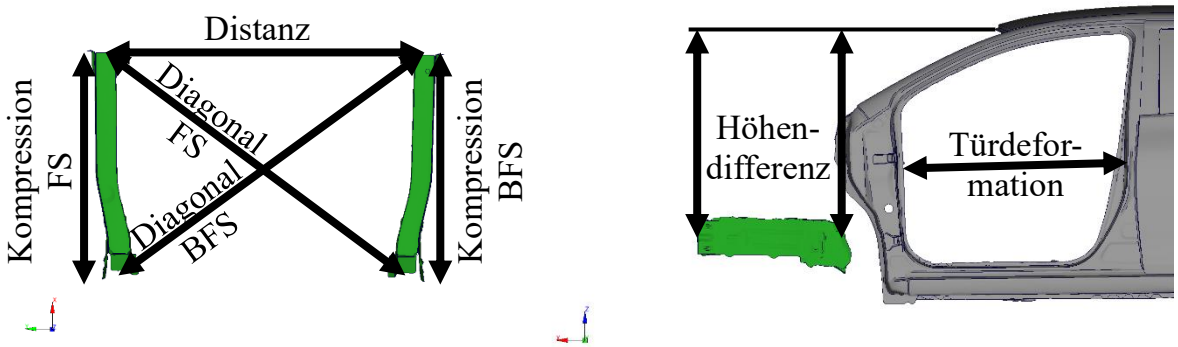


Bild 45: Ermittlung der plastischen Längsträgerverformungen anhand von Knotenabständen

Wie in Tabelle 9 bereits zu sehen ist, verletzt das Modell im initialen Zustand die meisten Restriktionen. Für den Optimierungsalgorithmus sind zuerst alle Restriktionen in den Zielbereich zu bringen, bevor dieser die Zielfunktion reduzieren kann. Um zu prüfen, ob die Restriktionen einzuhalten sind, werden acht Simulationen an verschiedenen Stellen im Entwurfsraum durchgeführt. Dazu werden die Entwurfsvariablen in drei Gruppen eingeteilt und variiert. Die drei Gruppen sind die Entwurfsvariablen der Crashboxen, der Längsträger und die übrigen Entwurfsvariablen in der Restgruppe (siehe Tabelle 8). Nach und nach werden verschiedene Wertekombinationen für die Entwurfsvariablen getestet, wie in Tabelle 10 dargestellt. Dabei bedeutet „Initial“, dass

die Entwurfsvariablen in einer Gruppe jeweils ihren eigenen initialen Wert zugeordnet bekommen. Entsprechend wird bei „Maximal“ der jeweilige maximale Wert eingesetzt und so weiter.

Tabelle 10: Größen der in Gruppen eingeteilten Entwurfsvariablen für die zusätzlichen Simulationen. Der jeweiligen Entwurfsvariablen wird ihr minimaler, initialer oder maximaler Wert zugewiesen

Nr.	Gruppe Crashbox	Gruppe Längsträger	Gruppe Rest
1	Initial	Initial	Initial
2	Minimal	Initial	Initial
3	Minimal	Minimal	Initial
4	Minimal	Minimal	Minimal
5	Maximal	Maximal	Maximal
6	Minimal	Maximal	Initial
7	Minimal	Initial	Minimal
8	Minimal	Maximal	Minimal

Zur Übersichtlichkeit werden in Tabelle 11 die minimalen Ergebnisse der verschiedenen Strukturantworten dargestellt, im Vergleich zur Restriktion. Es ist zu sehen, dass fünf der Restriktionen in keinem der gerechneten Fälle eingehalten werden.

Eine Minimierung der Masse oder einer der vorgestellten Strukturantworten ist aufgrund der vielen verletzten Restriktionen nicht sinnvoll. Stattdessen wird die Minimierung der Restriktionsverletzung angestrebt. Dazu ist eine Formel zu entwickeln, die die Verletzung der Restriktionen zu einem skalaren Wert zusammenfasst.

Problematisch sind die verschiedenen Einheiten und Dimensionen der einzelnen Restriktionen. Die Werte der plastischen Dehnungen sind im Vergleich zur Masse um ein Vielfaches kleiner. Durch eine Normierung der Werte werden die Dimensionen angeglichen und die Einheiten eliminiert. Als Normierungswert wird die dreifache Standardabweichung σ_k der jeweiligen Strukturantwort verwendet. Die Standardabweichung errechnet sich anhand der obigen Simulationen. Wie bereits bei der Intrusion der Spritzschutzwand wird auch hier die P-Norm mit dem Exponenten p zur Verrechnung der einzelnen Differenzen verwendet. Für jede der k Differenzen aus Strukturantwort $x_k$und Restriktionswert r_k wird das Maximum zwischen dieser Differenz und 0 ermittelt. Somit haben erfüllte Restriktionen keinen Einfluss mehr auf die

Zielfunktion. Der minimale Wert der Zielfunktion liegt also bei 0. Es ergibt sich folgende Formel für die Zielfunktion:

$$f = \sqrt[p]{\frac{\sum_k \left(max\left(\frac{x_k - r_k}{3 * \sigma_k}; 0\right)\right)^p}{k}} \tag{4}$$

Somit hat die Optimierung keine direkten Restriktionen, sondern die Erfüllung aller Restriktionen als Zielfunktion.

Tabelle 11: Ergebnisse der Untersuchung zur Einhaltung der Restriktionen, mit in rot markierten nicht erfüllten Restriktionen

Lastfall	Bezeichnung	Min. Wert	Restriktion
	Masse	1242 kg	< 1265 kg
US-NCAP	OLC	38,17 g	< 35 g
	Türdeformation Fahrerseite (FS)	6,25 mm	< 25 mm
	Türdeformation Beifahrerseite (BFS)	5,75 mm	< 25 mm
	Intrusion Spritzschutzwand FS	111 mm	< 100 mm
	Intrusion Spritzschutzwand BFS	130 mm	< 100 mm
RCAR	Max. plast. Dehnung Längsträger	0,33 %	< 3 %
	Max. plast. Dehnung Spritzschutzwand	4,03 %	< 6 %
	Max. plast. Dehnung A-Säule	0,27 %	< 1 %
	Intrusion Spritzschutzwand	|1,48| mm	< 3 mm
	Längsträger Kompression FS	|0,09| mm	< 3 mm
	Längsträger Kompression BFS	|0,00| mm	< 3 mm
	Längsträger Höhendifferenz FS	|0,35| mm	< 3 mm
	Längsträger Höhendifferenz BFS	|4,59| mm	< 3 mm
	Längsträger diagonal FS	|1,94| mm	< 3 mm
	Längsträger diagonal BFS	|3,56| mm	< 3 mm
	Längsträger Distanz	|0,28| mm	< 3 mm

5.2.3 Metamodellbasierte Optimierung

Für diese Optimierung wird keine direkte Optimierungsmethode gewählt, sondern eine metamodellbasierte Optimierung. Hierdurch lassen sich in kurzer Zeit verschiedene Optimierungen auf Grundlage eines DoE durchführen. Außerdem ist es im Nachhinein möglich, die Zielfunktion anzupassen oder Restriktionen zu verändern.

Zu Beginn sind die Daten für das Training der Metamodelle zu erzeugen. Dazu wird ein DoE mit 200 Stützstellen durchgeführt. Dieser wird mit der Software LS-OPT®

erstellt. Als DoE wird der Versuchsplan *Space Filling* gewählt. In (SCHUMACHER 2020, S. 88), wird *Space Filling* als eine übergeordnete Klasse der Versuchspläne dargestellt, unter die z. B. der Versuchsplan *Latin Hypercube* fällt. Bei (STANDER ET AL. 2020, S. 549) hingegen wird *Space Filling* als ein eigener Versuchsplan neben *Latin-Hypercube* vorgestellt und stellt eine Erweiterung von ebendiesem dar. Der *Space Filling* Versuchsplan nach (STANDER ET AL. 2020, S. 549) maximiert die Abstände zwischen den einzelnen Stützstellen und bietet somit eine gute Datenbasis für Approximationsverfahren (vgl. STANDER ET AL. 2020, S. 550).

Für jede Stützstelle werden die beiden oben genannten Lastfälle simuliert und die entsprechenden Strukturantworten aus Tabelle 9 ausgewertet. Die Auswertung findet dabei durch ein übergeordnetes Python-Skript statt, welches Strukturantworten direkt aus den Simulationsdateien extrahiert oder z. B. für die Berechnung des OLC-Wertes den Postprozessor Animator4® der GNS mbH startet und über ein automatisch generiertes Skript den OLC-Wert exportiert. Alle Strukturantworten werden in einer Tabelle gespeichert, sodass die Daten in einer Datei zusammengefasst sind. Das DoE wird sowohl mit dem Gesamtfahrzeug, als auch dem in Bild 38 vorgestellten Submodell durchgeführt.

Als Software zur Approximation und Optimierung wird Optimus® von NOESIS verwendet. Diese ist in ihrer Anwendung flexibler als LS-OPT® und bietet eine größere Auswahl von Metamodellen. In Optimus® wird die mit LS-OPT® erzeugte Tabelle als Datensatz importiert und dient zum Training der Metamodelle. Die Optimierung wird dann anhand der trainierten Metamodelle durchgeführt, sodass hier keine weiteren Simulationen notwendig sind. Im Nachgang sind die Optimierungsergebnisse durch Simulationen zu validieren.

Es bieten sich zwei Möglichkeiten, was den Umgang mit der Zielfunktion für die Optimierung betrifft. Einerseits kann die Zielfunktion direkt approximiert werden. Alternativ dazu ist es möglich alle Strukturantworten einzeln zu approximieren und die Zielfunktion aus den approximierten Werten zu ermitteln. Bei der direkten Approximation der Zielfunktion besteht die Gefahr, durch ein unzureichendes Metamodell keine validen Ergebnisse zu erhalten. Werden Metamodelle für jede Strukturantwort erstellt, hat eine schlecht approximierte einzelne Strukturantwort einen kleineren Einfluss auf die Zielfunktion. Außerdem besteht die Möglichkeit, verschiedene Arten von Metamodellen für unterschiedliche Strukturantworten zu verwenden und somit eine höhere Genauigkeit der Zielfunktion zu erhalten. Aus den genannten Gründen wird

jede Strukturantwort approximiert und die Zielfunktion anhand der Approximationen berechnet.

Die Auswahl der Metamodelle für die Strukturantworten erfolgt manuell, anhand von sogenannten *Model Sections* und *Scatter Plots*. Dies sind Funktionen von Optimus®, mit denen die Metamodelle für eine einzelne Strukturantwort, in Abhängigkeit von den Entwurfsvariablen dargestellt werden. In Bild 46 sind ausgewählte Diagramme für die Türdeformation auf der Beifahrerseite dargestellt. Die drei zur Auswahl stehenden Metamodelle sind *Kriging*, *Radial Basis Function network* (Kurz: RBFN) und ein *Deep Neural Network* (Kurz: DNN). Bei dem DNN fällt auf, dass der Kurvenverlauf scheinbar jeden Datenpunkt trifft und somit *Overfitting* stattfindet. Das bedeutet, dass das Neuronale Netzwerk alle Datenpunkte miteinander verbindet und somit zwar einen hohen Regressionswert hat, jedoch kaum Aussagekraft für die interpolierten Bereiche. Für das RBFN scheint kein *Overfitting* stattzufinden, jedoch weisen die Kurvenverläufe Knicke auf. Beim *Kriging* ergeben sich knickfreie stetige Kurven, weshalb diese Approximation für die Türdeformation der Beifahrerseite ausgewählt wird. Ein ähnliches Vorgehen findet für alle Strukturantworten statt, bis die in Tabelle 12 aufgeführten Metamodelle ausgewählt sind. Dort sind hauptsächlich quadratische Taylorapproximationen und *Kriging* vertreten. Lediglich die Masse wird linear approximiert.

In Optimus ergibt sich für die Optimierung der Workflow aus Bild 47. Dort werden links, im Array *Inputs*, die approximierten Strukturantworten importiert. Damit das Python-Skript `Goalfunction_Eval.py` diese verarbeiten kann, müssen die Eingangsgrößen über das Array *Outputs* in die Datei *Output* ausgegeben werden. Danach folgt die Aktion, das genannte Python-Skript auszuführen, welches die Datei *Goalvalue* erzeugt. In der Datei ist der Wert der Zielfunktion gespeichert, welcher wiederum von Optimus® eingelesen und in der einzelnen Ausgangsvariablen *Goalfunction_new* gespeichert wird.

Es ist ein Optimierungsalgorithmus und seine Parameter festzulegen, um mit dem dargestellten Workflow eine Optimierung durchzuführen. Der Vorteil einer metamodellbasierten Optimierung liegt darin, dass ein Funktionsaufruf der Metamodelle, im Vergleich zu einer Simulation, kaum Ressourcen benötigt. Es wird *Simulated Annealing* als Optimierungsalgorithmus gewählt.

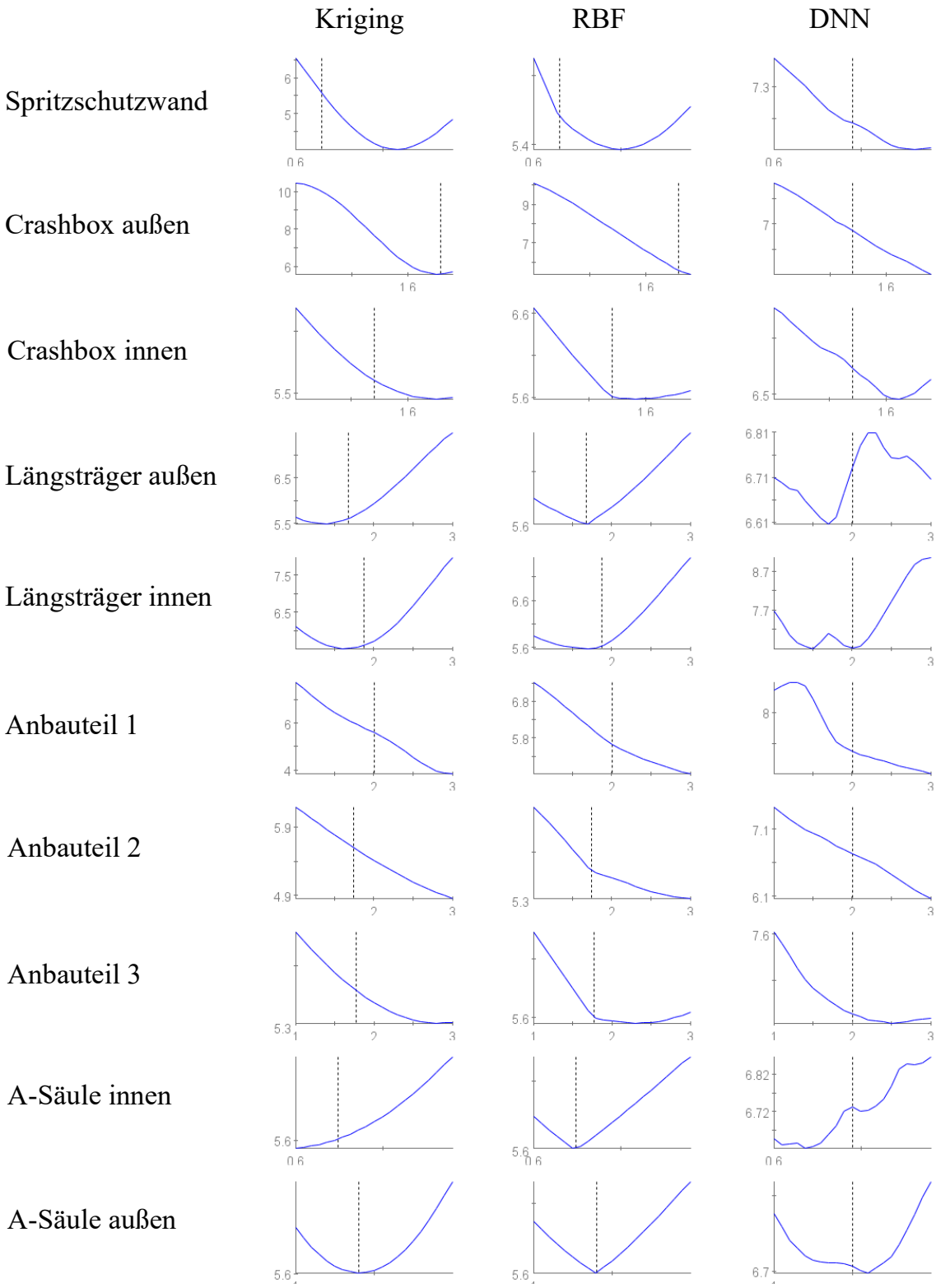


Bild 46: Beispiel von *Model Sections* für die Türdeformation der Beifahrerseite von drei verschiedenen Metamodellen (*Kriging*, *Radial Basis Function Network*, *Deep Neural Network*)

Tabelle 12: Gewählte Metamodelle für die einzelnen Strukturantworten

Lastfall	Bezeichnung	Metamodell
	Masse	linear
US-NCAP	OLC	quadr. Taylor
	Türdeformation Fahrerseite (FS)	quadr. Taylor
	Türdeformation Beifahrerseite (BFS)	Kriging
	Intrusion Spritzschutzwand FS	quadr. Taylor
	Intrusion Spritzschutzwand BFS	quadr. Taylor
RCAR	Max. plast. Dehnung Längsträger	quadr. Taylor
	Max. plast. Dehnung Spritzschutzwand	Kriging
	Max. plast. Dehnung A-Säule	quadr. Taylor
	Intrusion Spritzschutzwand	Kriging
	Längsträger Kompression FS	quadr. Taylor
	Längsträger Kompression BFS	Kriging
	Längsträger Höhendifferenz FS	quadr. Taylor
	Längsträger Höhendifferenz BFS	quadr. Taylor
	Längsträger diagonal FS	quadr. Taylor
	Längsträger diagonal BFS	quadr. Taylor
	Längsträger Distanz	quadr. Taylor

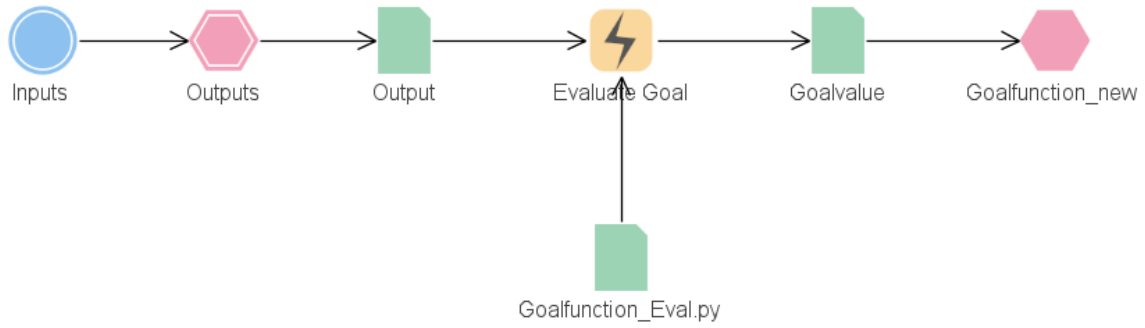


Bild 47: Optimus Workflow für die metamodellbasierte Optimierung mit aus den approximierten Strukturantworten errechneter Zielfunktion

Für das vorliegende Problem werden die Optimierungsparameter aus Tabelle 13 gewählt. Dabei ist der *Random Seed* auf 1 gesetzt. Damit wird der Zufallsgenerator ausgeschaltet, wodurch die gleiche Optimierung mehrfach ausgeführt werden kann. Dies ist notwendig, damit die Optimierung des Gesamtmodells und des Submodells vergleichbar sind. Die anderen Parameter orientieren sich an den im Programm vorgestellten Standardeinstellungen.

Tabelle 13: Optimierungsparameter für den *Simulated Annealing* Algorithmus in Optimus®

Parameter	Wert
Random seed	1
Initial temperature	5
Annealing factor	0,93
Stepwidth adjustment factor	1,2
Number of cycles	5
Iteration at T=const.	5
Average stopping stepwidth	0,01
Maximum number of function evaluations	10000

5.2.4 Optimierungsergebnisse

Durch die Optimierung anhand der Metamodelle ergeben sich für das Submodell und Gesamtmodell verschiedene Optima. Tabelle 14 gibt eine Übersicht über die Ergebnisse der Optimierung. Die Strukturantworten sind anhand von Gesamtmodellsimulationen ermittelt, um diese zu validieren. Beim Blick auf die Entwurfsvariablen haben sieben von zehn den gleichen Trend im Submodell wie im Gesamtmodell. Lediglich die Anbauteile 2 und 3 sowie das äußere Blech der A-Säule verändern sich in verschiedene Richtungen vom initialen Wert aus.

Die Zielfunktion ist im Gesamtmodell etwas kleiner als im Submodell. In beiden Fällen wurde eine Verbesserung erreicht. Masse, Türdeformationen und OLC-Wert liegen bei beiden Optima in der gleichen Größenordnung. Im Falle der Intrusion der Spritzschutzwand des US-NCAP Lastfalls haben sich die Werte des Submodells gegenüber dem initialen Entwurf verschlechtert. Gleiches ist bei der maximalen plastischen Dehnung auf der Spritzschutzwand zu beobachten, wobei sich hier ebenfalls das Gesamtmodell verschlechtert hat. Dem entgegen steht in beiden Fällen eine Reduktion der Intrusion der Spritzschutzwand im RCAR Lastfall. Im direkten Vergleich der plastischen Dehnungen in der Spritzschutzwand (siehe Bild 48) ist zu sehen, dass bei beiden Optima die Dehnung insgesamt reduziert wurde.

Tabelle 14: Optimierungsergebnisse der metamodellbasierten Optimierung validiert durch Gesamtmodellsimulationen. Verletzte Restriktionen sind in rot markiert

	Bezeichnung	Initial	Gesamt-modell	Sub-modell
Entwurfsvariablen	Spritzschutzwand	0,75	1,19	0,92
	Crashbox außen	1,89	1,96	1,97
	Crashbox innen	1,30	1,79	1,75
	Längsträger außen	1,67	2,28	2,78
	Längsträger innen	1,87	2,51	2,64
	Anbauteil 1	2,00	2,99	2,92
	Anbauteil 2	1,74	1,64	1,79
	Anbauteil 3	1,77	1,55	2,23
	A-Säule innen	0,84	1,07	1,16
	A-Säule außen	1,40	1,58	1,12
	Zielfunktion	0,62	0,47	0,51
Strukturantworten	Masse [kg]	1253	1263	1263
	OLC [g]	38,15	36,51	36,09
	Türdeformation Fahrerseite (FS) [mm]	6,12	6,14	7,49
	Türdeformation Beifahrerseite (BFS) [mm]	5,60	7,31	7,30
	Intrusion Spritzschutzwand FS [mm]	140	134	142
	Intrusion Spritzschutzwand BFS [mm]	131	131	138
	Max. plast. Dehnung Längsträger [%]	6,3	7,3	4,2
	Max. plast. Dehnung Spritzschutzwand [%]	6,3	7,5	8,8
	Max. plast. Dehnung A-Säule [%]	0,8	0,5	0,8
	Intrusion Spritzschutzwand [mm]	|2,67|	|2,01|	|1,99|
	Längsträger Kompression FS [mm]	|0,17|	|0,08|	|0,09|
	Längsträger Kompression BFS [mm]	|0,07|	|0,02|	|0,02|
	Längsträger Höhendifferenz FS [mm]	|2,34|	|0,93|	|1,08|
	Längsträger Höhendifferenz BFS [mm]	|5,70|	|5,48|	|5,63|
	Längsträger diagonal FS [mm]	|4,92|	|1,07|	|1,02|
	Längsträger diagonal BFS [mm]	|12,98|	|11,78|	|12,14|
	Längsträger Distanz [mm]	|10,91|	|14,77|	|15,22|
	CPU-Stunden	-	37949	28622

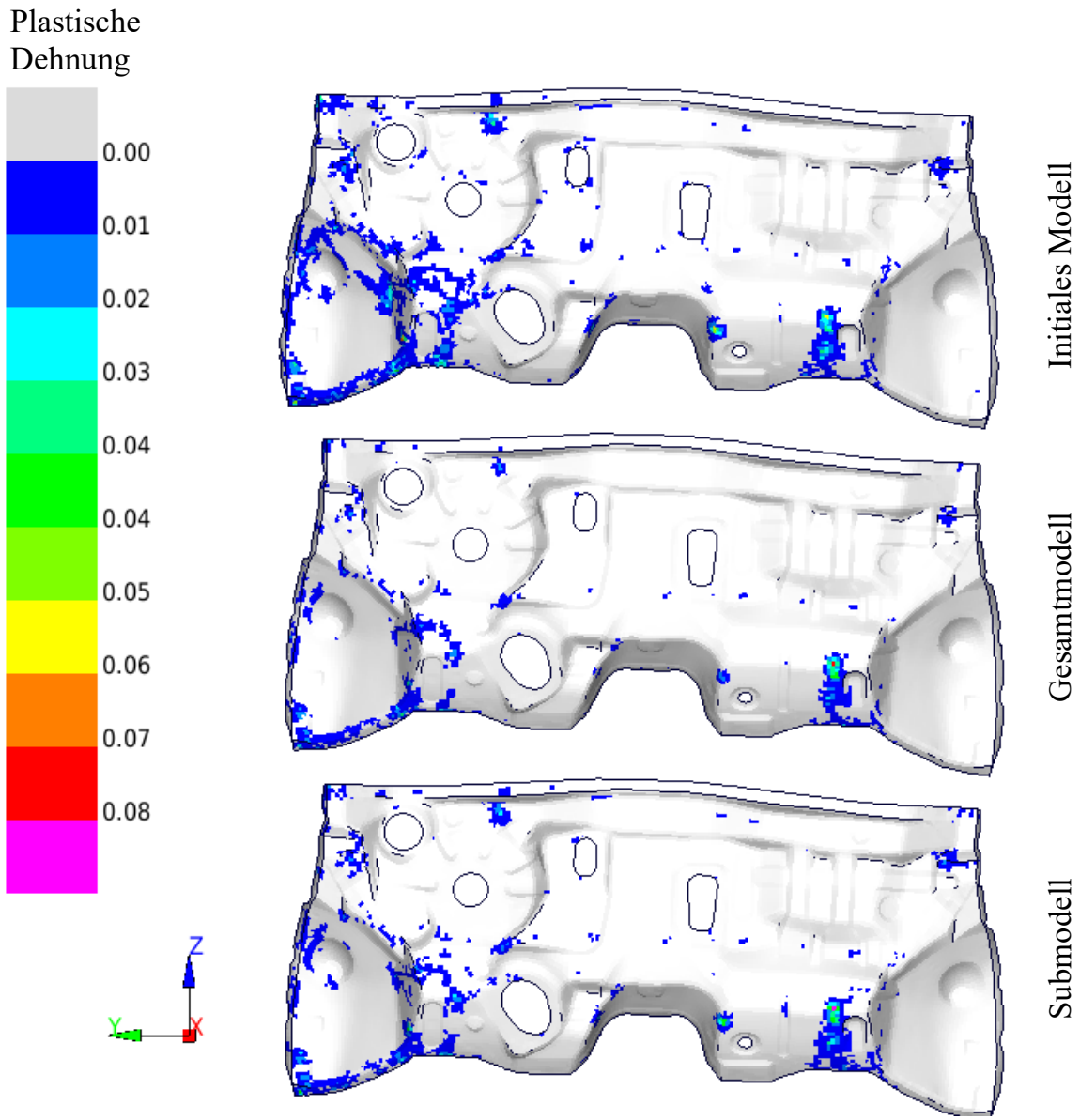


Bild 48: Vergleich von plastischen Dehnungen in der Spritzschutzwand

Bei den Deformationen des RCAR Lastfalls sind alle Werte in den Optima geringer als initial, abgesehen von der Distanz zwischen den Längsträgern. Diese steigt in beiden Optima gegenüber dem Startentwurf an. Der Grund dafür liegt in den Crashboxen. Bereits im initialen Entwurf kommt es beim RCAR Lastfall nur bedingt zum Faltenbeulen, dafür tritt ein Ausknicken der Crashboxen auf. Die Verbindungsstelle zwischen Crashbox und Längsträger versagt, wodurch zuerst die Crashbox auf der Fahrerseite wegknickt und daraufhin durch die Kopplung über den Bumper die Beifahrerseite. In Bild 49 ist der Unterschied zwischen den drei Simulationen zu sehen. Bei beiden Optima hat die Crashbox eine höhere Blechdicke und ist dadurch steifer. Dies hat ein stärkeres Ausknicken der Crashboxen und damit die größere Distanzänderung zwischen den Längsträgern zur Folge.

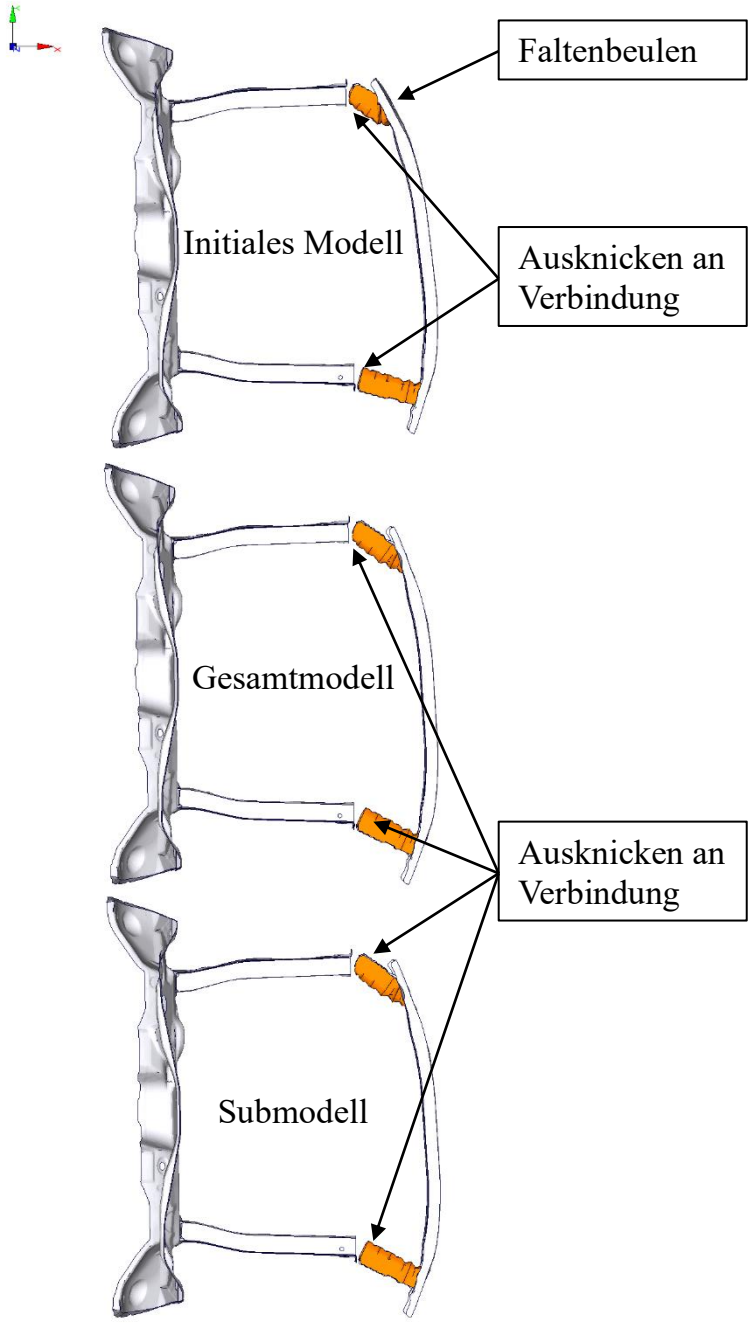


Bild 49: Vergleich der kritischen Bauteile initial und im Optimum für den Lastfall RCAR, in orange markiert sind die Crashboxen

Das Optimierungsergebnis als solches ist nur bedingt nutzbar. Viele der Restriktionen konnten reduziert werden, jedoch erfüllt das Modell noch immer nicht alle. Das Knicken der Crashboxen ist ein unerwünschter Mechanismus, welcher in der Optimierung hätte reduziert, statt gefördert werden sollen. Im Vergleich zwischen Submodell und Gesamtmodell haben sich ähnliche Optima ergeben, bei einer Zeiteinsparung von ca. 25 % des Submodells gegenüber dem Gesamtmodell.

6. Zusammenfassung und Ausblick

Die vorliegende Arbeit zeigt das Potential von Submodellen in der Optimierung. Es werden verschiedene automatische Methoden zur Erstellung von Submodellen und den zugehörigen Randbedingungen gezeigt. Durch beispielhafte Optimierungen wird die Methodik in der Praxis geprüft.

6.1 Ergebnisse

Die physikalische Reduktion von Fahrzeugmodellen zu Submodellen bietet ein hohes Potential, Ressourcen einzusparen und somit Crashoptimierungen sinnvoll in den Fahrzeugentwicklungsprozess zu integrieren. Die vorgestellten Optimierungsbeispiele haben gezeigt, dass durchaus ähnliche Ergebnisse zu erzielen sind wie mit einem Gesamtfahrzeugmodell bei geringerem Ressourcenbedarf. Allerdings liegen die Ressourceneinsparungen im Bereich von 20-30 % und erreichen nicht die gewünschten 50 %. Einerseits sinkt die Genauigkeit der Submodelle, andererseits sind evtl. benötigte Strukturantworten nicht mehr auswertbar durch eine weitere Reduktion des Modells.

Es ist zu beachten, dass die Erstellung von Submodellen automatisch ablaufen sollte, damit die Methodik in bestehende Prozessketten integriert werden kann. Dies ist für die vorliegenden Modelle gegeben, allerdings nicht allgemeingültig. Bei der Erstellung der Submodelle können Fehler in den *Solverdecks* entstehen, die manuell korrigiert werden müssen. Mit jedem neuen Modell werden weitere Schritte automatisiert, jedoch entsteht daraus ein Wettlauf gegen die Entwicklung von FE-Modellen.

Der Anwender sollte möglichst wenig Eingaben tätigen müssen, damit er zur Bedienung nur minimales Wissen über Submodelle benötigt. Das *ASCO*-Tool bietet daher verschiedene Optionen, bei denen z. B. ein einzelner Parameter, die Submodellgröße, ausreicht. Es ist auch möglich, die Parameter der verschiedenen Algorithmen bis ins Detail zu steuern, sodass erfahrene Nutzer Anpassungen nach ihren Wünschen festlegen können.

6.2 Diskussion und Ausblick

Die Generierung von Submodellen sollte für einen reibungslosen Ablauf vollautomatisch ablaufen, dies ist nicht für alle Modelle gegeben. Insbesondere bei aktuellen Fahrzeugmodellen, in denen Umformdaten der Herstellung usw. berücksichtigt werden, funktioniert der automatische Ablauf nicht. Dazu sind weitere Modelle notwendig, mit denen das Tool getestet und erweitert werden kann. Diese Aufgabe

würde eher einem Softwarehersteller bei der Vermarktung des Tools zukommen, als dass es ein eigenes Forschungsthema wäre, da die Problematik hauptsächlich aus programmtechnischen Hürden besteht.

Bei der Anwendung von Submodellen besteht die Gefahr, dass Ergebnisse nicht mit dem Gesamtmodell übereinstimmen. Trotz verschiedener Validierungstechniken ist nicht garantiert, dass das Submodell für jeden Punkt im Entwurfsraum gültig ist. Dadurch ist einerseits der Nutzen des Submodells gegenüber der Gefahr der Ungenauigkeit abzuwägen, auf der anderen Seite sind weitere Validierungsmethodiken zu entwickeln. Hierzu sind Metriken nötig, die den gesamten Entwurfsraum in Betracht ziehen. Dies führt wiederum zu einem hohen Simulationsaufwand. Außerdem sollten umfassendere Validierungsmetriken implementiert werden, welche auch die Kinematik des Modells detailliert validieren. Hierbei könnte das bereits erwähnte Programm DIFFCRASH® unterstützen, da dieses die Verschiebungen jedes Knotens untersucht und somit sowohl die globale als auch lokale Kinematik des Modells abbildet.

Auch die verschiedenen Möglichkeiten der Submodellerstellung und Randbedingungen bieten auf der einen Seite eine Vielfalt an Kombinationsmöglichkeiten, jedoch auch die Gefahr nicht sinnvoll nutzbarer Submodelle. So ist eine Kombination aus einem PBS-Submodell, also abhängig von der Evaluationsfunktion mit einer angebundenen Ersatzmasse nicht sinnvoll. Durch das RBE an der Ersatzmasse und die im gesamten Modell verteilten *Interface Nodes* wird das Modell extrem ausgesteift.

Zukünftig könnte die Auswahl der Kombination von Submodellmethode, Randbedingung und den erforderlichen Parametern durch eine künstliche Intelligenz oder einen Auswahlalgorithmus unterstützt werden. Es müssten dazu eine Reihe von Submodellen für verschiedene Aufgaben und Lastfälle erzeugt werden und daraus die Auswahlmetrik abgeleitet oder die künstliche Intelligenz trainiert werden.

Eine weitere Idee für zukünftige Forschungsarbeiten ist die Kombination der Submodelltechnik mit der Graphen- und Heuristikbasierten Topologieoptimierung (Kurz: GHT), welche z. B. in (BEYER ET AL. 2021), (SCHUMACHER 2020, S. 290 ff.) oder (SCHNEIDER ET AL. 2019) vorgestellt wird. Der Ablauf könnte dann so aussehen, dass durch das *ASCO*-Tool eine Substruktur mit dem Entwurfsraum aus dem Gesamtmodell extrahiert wird, auf die dann die GHT angewendet wird.

Insgesamt bietet die Submodelltechnik viel Potential für weitere Entwicklungen. Sowohl in der Methodik selbst, als auch die Kombination mit anderen Optimierungsverfahren wie der GHT. Dem entgegen stehen die Probleme mit dem Automatisierungsgrad der Software und der Genauigkeit der Submodelle. Auch das

Einsparpotential an Ressourcen liegt noch unterhalb des gewünschten Ziels. Es ist fraglich, inwiefern sich hier noch mehr Rechenzeit einsparen lässt, da bei der weiteren Reduktion des Modells evtl. benötigte Strukturantworten nicht mehr auswertbar sind.

Literaturverzeichnis

ALT, HANS WILHELM (2012): Lineare Funktionalanalysis. Eine anwendungsorientierte Einführung. 6. Aufl. 2012. Berlin, Heidelberg. Springer Berlin Heidelberg.

BAIER, HORST; SEEßELBERG, CHRISTOPH; SPECHT, BERNHARD (1994): Optimierung in der Strukturmechanik. Wiesbaden. Vieweg+Teubner Verlag.

BARBAT, SAEED; FU, YAN; ZHAN, ZHENFEI; YANG, REN-JYE; GEHRE, CHRISTIAN (2013): OBJECTIVE RATING METRIC FOR DYNAMIC SYSTEMS. In: *Proceedings of the 23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*. Online verfügbar unter https://www-esv.nhtsa.dot.gov/Proceedings/23/files/23ESV-000448.PDF, zuletzt geprüft am 11.03.2022.

BENITO CIA, LEYRE; WIELENS, SVEN; SCHÖNE, CLAUDIUS; SCHUMACHER, AXEL (2020): Automatic Submodel-based Multi-Level Optimization of Crash Structures. In: *Proceeding of the Automotive CAE Grand Challenge 2020, 29th and 30th of September 2020*, Hanau, Germany.

BEYER, F.; SCHNEIDER, D.; SCHUMACHER, A. (2021): Finding three-dimensional layouts for crashworthiness load cases using the graph and heuristic based topology optimization. In: *Structural and Multidisciplinary Optimization*, S. 59–73.

BÜTTNER, JANA (2022): Effiziente Lösungsansätze zur Reduktion des numerischen Ressourcenbedarfs für den operativen Einsatz der Multidisziplinären Optimierung von Fahrzeugstrukturen. Dissertation, Bergische Universität Wuppertal.

CARHS.TRAINING GMBH (2019): SAFETY COMPANION 2019. CARHS.TRAINING GMBH. Alzenau.

CHRISTENSEN, PETER W.; KLARBRING, ANDERS (2009): An introduction to structural optimization. Dordrecht, London. Springer (Solid Mechanics and Its Applications, 153).

DIEZ, CONSTANTIN (2019): Process for Extraction of Knowledge from Crash Simulations by means of Dimensionality Reduction and Rule Mining. Dissertation, Bergische Universität Wuppertal.

DUDDECK, FABIAN; WEHRLE, ERICH (2015): Recent Advances on Surrogate Modeling for Robustness Assessment of Structures with respect to Crashworthiness Requirements. In: *Proceedings of 10th European LS-DYNA Conference 2015, Würzburg, Germany*.

DYNAMORE GMBH: plot2bc. Online verfügbar unter https://www.dynamore.de/de/produkte/pre-und-postprozessoren/tools/plot2bc, zuletzt geprüft am 11.01.2022.

EURO NCAP (2020): Assessment Protocol - ADULT OCCUPANT PROTECTION. 2020. Online verfügbar unter https://cdn.euroncap.com/media/58227/euro-ncap-assessment-protocol-aop-v912.pdf, zuletzt geprüft am 25.08.2021.

EURO NCAP (2021): The Dynamic Assessment of Car Seats for Neck Injury Protection Testing Protocol. Online verfügbar unter https://cdn.euroncap.com/media/57828/euro-ncap-whiplash-test-protocol-v41.pdf, zuletzt geprüft am 25.01.2022.

FALCONI D., CARLOS J.; WALSER, ALEXANDER F.; SINGH, HARMAN; SCHUMACHER, AXEL (2018): Automatic Generation, Validation and Correlation of the Submodels for the Use in the Optimization of Crashworthy Structures. In: *Schumacher, A., Vietor, T., Fiebig, S., Bletzinger, K.-U., Maute, K. (Hrsg.): Advances in Structural and Multidisciplinary Optimization, Proceedings of the 12th World Congress of Structural and Multidisciplinary Optimization (WCSMO12)*, Springer Nature, S. 1558–1571.

FANG, JIANGUANG; SUN, GUANGYONG; QIU, NA; KIM, NAM H.; LI, QING (2017): On design optimization for structural crashworthiness and its state of the art. In: *Struct Multidisc Optim.* 55, S. 1091–1119.

KACPRZYK, JANUSZ; JAIN, LAKHMI C.; AGUIAR E OLIVEIRA JUNIOR, HIME; INGBER, LESTER; PETRAGLIA, ANTONIO; REMBOLD PETRAGLIA, MARIANE; AUGUSTA SOARES MACHADO, MARIA (2012): Stochastic Global Optimization and Its Applications with Fuzzy Adaptive Simulated Annealing. Berlin, Heidelberg. Springer Berlin Heidelberg.

KRAMER, FLORIAN (2009): Passive Sicherheit von Kraftfahrzeugen. Biomechanik - Simulation - Sicherheit im Entwicklungsprozess. 3., überarb. Aufl. Wiesbaden. Vieweg + Teubner (ATZ-MTZ-Fachbuch).

KÜBLER, LARS; GARGALLO, SIMON; ELSÄßER, KONRAD (2009): Sicherheit Bewertungskriterien zur Auslegung von Insassenschutzsystemen. In: *Automobiltechnische Zeitschrift: ATZ 2009*, S. 426–433.

LIVERMORE SOFTWARE TECHNOLOGY (LST), AN ANSYS COMPANY (2020): LS-DYNA KEYWORD USER'S MANUAL. Volume 1. LS-DYNA R12.

LIVERMORE SOFTWARE TECHNOLOGY CORPORATION (2006): LS-DYNA. Theory manual. Livermore, Calif.. Livermore Software Technology Corp.

MCCARTHY, M. A.; HARTE, C. G.; WIGGENRAAD, J. F. M.; MICHIELSEN, A. L. P. J.; KOHLGRÜBER, D.; KAMOULAKOS, A. (2000): Finite element modelling of crash response of composite aerospace sub-floor structures. In: *Computational Mechanics*, S. 250–258.

MORTISHED, CHARLES; OLLAR, JONATHAN; BENZIE, PETER; JONES, ROYSTON; SIENZ, JOHANN; TOROPOV, VASSILI (2018): Multidisciplinary optimisation of an automotive body-in-white structure using crushable frame springs and sub space metamodels in trust-regions. In: *Schumacher, A., Vietor, T., Fiebig, S., Bletzinger, K.-U., Maute, K. (Hrsg.): Advances in Structural and Multidisciplinary Optimization, Proceedings of the 12th World Congress of Structural and Multidisciplinary Optimization (WCSMO12)*, Springer Nature, S. 1572–1584.

NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION (2020): Crash Simulation Vehicle Models. NHTSA. Online verfügbar unter https://www.nhtsa.gov/crash-simulation-vehicle-models, zuletzt geprüft am 12.01.2020.

PLASCHKO, PETER; BROD, KLAUS (1995): Nichtlineare Dynamik, Bifurkation und Chaotische Systeme. Wiesbaden. Vieweg+Teubner Verlag.

PLEIADES Hardware (2022). Online verfügbar unter https://pleiades-buw.github.io/PleiadesUserDocumentation/hardware, zuletzt geprüft am 07.03.2022.

RAO, SINGIRESU S. (2009): Engineering optimization. Theory and practice. 4th ed. Hoboken N.J.. John Wiley & Sons.

RCAR (2011a): Information on the implimentation of RCAR crash standards in the German insurance vehicle rating system and information on AEB systems; Document 1. Online verfügbar unter https://www.rcar.org/Papers/Procedures/CrashStandards_GermanRatingSystem.pdf, zuletzt geprüft am 12.01.2021.

RCAR (2011b): Information on the implimentation of RCAR crash standards in the German insurance vehicle rating system and information on AEB systems; Document 2. Online verfügbar unter https://www.rcar.org/Papers/Procedures/CrashStandards_GermanRatingSystem.pdf, zuletzt geprüft am 12.01.2021.

SCHÄFFER, MICHAEL; STURM, RALF; FRIEDRICH, HORST E. (2019a): Automated generation of physical surrogate vehicle models for crash optimization. In: *International Journal of Mechanics and Materials in Design*, S. 43–60.

SCHÄFFER, MICHAEL; STURM, RALF; FRIEDRICH, HORST E. (2019b): Methodological approach for reducing computational costs of vehicle frontal crashworthiness analysis

by using simplified structural modelling. In: *International Journal of Crashworthiness*, S. 39–53.

SCHNEIDER, DOMINIK; SCHUMACHER, AXEL; DONHAUSER, TOBIAS; HUF, ALEXANDER; SCHMEER, SEBASTIAN (2019): Flexible Graph Syntax for the Topology Optimization of Crashworthiness Profile Structures Made from Thermoplastic Composites. In: *Key Engineering Materials*, S. 493–499.

SCHUMACHER, AXEL (2020): Optimierung mechanischer Strukturen. 3. Edition. Springer Berlin Heidelberg.

SCHUMACHER, AXEL; ORTMANN, CHRISTOPHER (2015): Combining state of the art meta-models for predicting the behavior of non-linear crashworthiness structures for shape and sizing optimizations. In: *Proceedings of the 11th World Congress on Structural and Multidisciplinary Optimization, 07th - 12th, June 2015, Sydney, Australia*, S. 477–482.

SCHUMACHER, AXEL; SINGH, HARMAN; WIELENS, SVEN (2019): Submodel-based Multi-Level optimization of crash structures using statistically generated universal correlations of the different levels. In: *Proceedings of the 13th World Congress on Structural and Multidisciplinary Optimization, 20th - 24th, May 2019, Beijing, China*.

SINGH, HARMAN; SCHUMACHER, AXEL; FALCONI D., CARLOS J.; WALSER, ALEXANDER F.; TRENTMANN, SVEN; BENITO C., LEYRE ET AL. (2017): Hierarchical Multi-Level-Optimization of crashworthy structures using automatic generated submodels. In: *11th European LS-DYNA Conference 2017, Salzburg, Austria*.

STANDER, NIELEN; BASUDHAR, ANIRBAN; ROUX, WILLEM; LIEBOLD, KATHARINA; EGGLESTON, TRENT; GOEL, TUSHAR; CRAIG, KEN (2020): LS-OPT® User's Manual. A DESIGN OPTIMIZATION AND PROBABILISTIC ANALYSIS TOOL FOR THE ENGINEERING ANALYST. 2020, Livermore Software Technology Corporation.

THOLE, CLEMANS-AUGUST; NIKITINA, LIALIA; CLEES, TANJA (2010): Advanced Mode Analysis for Crash Simulation Results. Online verfügbar unter https://www.sidact.de/fileadmin/user_upload/Paper_Advanced_Mode_Analysis_LS-DYNA.pdf, zuletzt geprüft am 11.03.2022.

WAGNER, MARCUS (2019): Lineare und nichtlineare FEM. Wiesbaden. Springer Fachmedien Wiesbaden.

WALSER, ALEXANDER F.; SCHUMACHER, AXEL; SINGH, HARMAN; SCHÖNE, CLAUDIUS; BENITO CIA, LEYRE (2018): Verschachtelte Optimierung des Crashverhaltens

von Fahrzeugen mit automatisch erzeugten und bewerteten Submodellen. In: *VDI-Berichte zur 19. VDI-Kongress SIMVEC*.

WEIDER, KATRIN (2021): Topologische Ableitung zur Optimierung crashbelasteter Strukturen. Düren. Shaker Verlag GmbH (Berichte aus dem Maschinenbau).

YOGANANDAN, NARAYAN; NAHUM, ALAN M.; MELVIN, JOHN W. (2015): Accidental Injury. New York, NY. Springer New York.

Anhang – Ergänzende Bilder und Tabellen

A.1 ASCO-Programminfrastruktur

Tabelle A - 1: Dateien und Ordner des *ASCO*-Tools im initialen Zustand. Einträge in Klammern und ihre Unterordner/-Dateien werden während der Submodellgenerierung erstellt

ASCO			
	LOADCASE		
		Modelldateien	
	RESPONSES		
		Expressions	
	(INITIAL)		
		Simulation des Gesamtmodells	
	(SUBMODEL_PBS_SUB_0.6)		
		FULLMODEL	
			Gesamtmodell mit *Interface Nodes*
			deleted_lines.txt
		SUBMODEL	
			Submodell
			deleted_lines.txt
		Weitere Daten, abhängig vom gewählten Verfahren	
	SOURCE		
		GEN	
			prepareModel_utils.inc
			ani_displData_generation.tcl
			CBS_ModelReduction.tcl
			DMA_BoundaryConditions.tcl
			ePBS_ModelReduction.tcl
			gen_subModel_generation.tcl
			GET_InterfaceNodes.tcl
			KMA_BoundaryConditions.tcl
			PBS_ModelReduction.tcl
			process_SubModel_generation.tcl
		RIGID	
			Def2Rigid.py
			MatRigid.py
			count_rigid_elements.tcl
			take_screenschot.tcl
			write_neighbours.tcl
		TEMPLATES	
			session_file___a_pillar_intrusion.ses
			session_file___crashzone_intrusion.ses
			session_file___olc.ses
			get_responses_Template.lsopt
			SSL_TEMPLATE.lsopt

- **UNI**
  - functions.cfg
  - params.inc
  - prepareModel_utils.inc
  - tcl_utils.tcl
  - ani_evaluateFuntion.tcl
  - calculateIntegralProc.tcl
  - compareTwoEvaluations_norm.tcl
  - count_elements.tcl
  - gen_prepareModelProc.tcl
  - prepareModel.tcl
  - START_G4_TCL.tcl
- cleanInputdeck.py
- compare_models.py
- control_optimization.py
- createsubmodel.py
- methods.py
- monitor_opt.py
- optimization.py
- get_responses.sh
- log.cfg
- l2a
- README.md
- main.py
- params.cfg

Tabelle A - 2: Beschreibung der verschiedenen Daten und Skripte des *ASCO*-Tools

Datei	**Beschreibung**
ani_displData_generation.tcl	TCL-Skript zur Extraktion der Verschiebungen aus einer Pamcrash® Simulation
ani_evaluateFuntion.tcl	TCL-Skript zur Evaluation des Modells vor der Submodellerstellung
calculateIntegralProc.tcl	TCL-Skript zur Berechnung eines Integrals mit Animator4®
CBS_ModelReduction.tcl	TCL-Skript für das CBS Reduktionsverfahren
cleanInputdeck.py	Python-Skript zur Korrektur des Solverdecks nach der Submodellerstellung
compare_models.py	Python-Skript für die Validierung des Submodells, enthält die Aufrufe für die verschiedenen Verfahren
compareTwoEvaluations_norm.tcl	TCL-Skript zum Vergleich zweier Evaluationsdateien
control_optimization.py	Python-Skript zur Steuerung innerhalb einer Stützstelle bei der Optimierung mit dem ASCO-Tool
count_elements.tcl	TCL-Skript zur Bestimmung der Anzahl an Elementen in einem Modell mit Generator4®

count_rigid_elements.tcl	TCL-Skript für den RIGID-Ansatz, zur Bestimmung der Anzahl an Rigid-Elementen an einem Bauteil
createsubmodel.py	Python-Skript, welches die Erstellung der Submodelle steuert
Def2Rigid.py	Python-Skript für den RIGID-Ansatz, bei der Verwendung der D2RA Methode
DMA_BoundaryConditions.tcl	TCL-Skript zur Erstellung der DMA Randbedingungen
ePBS_ModelReduction.tcl	TCL-Skript zur ePBS Modellreduktion
Expressions	XML-Dateien für Expressions in LS-OPT bei Verwendung des ASCO-Tools für die Durchführung eines DOE oder Optimierung
functions.cfg	Konfigurationsdatei für die Auszuwertende Evaluationsfunktion durch Animator4®
gen_prepareModelProc.tcl	TCL-Skript zur Modellevaluierung
gen_subModel_generation.tcl	TCL-Skript zur Submodellerstellung
GET_InterfaceNodes.tcl	TCL-Skript zur Ermittlung der *Interface Nodes*
get_responses.sh	Shell-Skript zur Auswertung der Strukturantworten bei einer von ASCO-Tool ausgewerteten Optimierung
get_responses_Template.lsopt	Template einer LS-OPT Datei zur Ermittlung der Strukturantworten
KMA_BoundaryConditions.tcl	TCL-Skript zur Bestimmung der KMA Randbedingungen
l2a	LS-Dyna Tool zur Extraktion von ASCII-Daten aus einer *d3plot*-Datei
log.cfg	Konfigurationsdatei für den Logger des ASCO-Tools
main.py	Hauptprogramm des ASCO-Tools, von dem aus alle Funktionen gesteuert werden (Python-Skript)
MatRigid.py	Python-Skript für den RIGID-Ansatz, bei der Verwendung der MatRigid Methode
methods.py	Python-Bibliothek von häufig verwendeten Methoden im ASCO-Tool
Modelldateien	FE Dateien des Fahrzeugmodells
monitor_opt.py	Python-Skript zur Überwachung einer Optimierung mit dem ASCO-Tool
optimization.py	Python-Skript zur Steuerung einer Optimierung mit dem ASCO-Tool
params.cfg	ASCO-Konfigurationsdatei
params.inc	TCL-Bibliothek zur Extraktion von Daten aus der ASCO-Konfigurationsdatei
PBS_ModelReduction.tcl	TCL-Skript für das PBS Reduktionsverfahren
prepareModel.tcl	TCL-Skript zur Steuerung der initialen Modellevaluierung
prepareModel_utils.inc	TCL-Bibliothek mit Methoden zur Steuerung der initialen Modellevaluierung
process_SubModel_generation.tcl	TCL-Skript zur Steuerung der Submodellerstellung
README.md	Readme Datei mit Anweisungen und Informationen zum ASCO-Tool

session_file___a_pillar_intrusion.ses	Animator4-Sessionfile zur Auswertung der Deformation an der A-Säule
session_file___crashzone_intrusion.ses	Animator4-Sessionfile zur Auswertung der Intrusion der Crashzone
session_file___olc.ses	Animator4-Sessionfile zur Auswertung des OLC-Werts
SSL_TEMPLATE.lsopt	Template einer LS-OPT Datei zur Erstellung einer Single-Level-Optimierung
START_G4_TCL.tcl	TCL-Skript zum Starten von Generator4 bei der RIGID-Methode
take_screenschot.tcl	TCL-Skript zum Erstellen von Screenshots in Generator4 für die RIGID-Methode
tcl_utils.tcl	TCL-Bibliothek mit häufig verwendeten Methoden
write_neighbours.tcl	TCL-Skript zur Ermittlung zusammenhängender Bauteile bei der RIGID-Methode

A.2 ASCO-Konfigurationsdatei params.cfg

```
#############################################################################################
# INITIAL OPTIONS FOR ASCO
#
# STEPS (SUB: Submodel generation, SLO: Single Level Optimization,
#        MLO: Multi Level Optimization)
STEP                                    {}
# Define the evaluation mode (local/hybrid/cluster; pleiades;
PAG_local/PAG_queue/PAG_hww)
EVALUATION_MODE                         {}
#
#############################################################################################
# PATH INFO FOR DIFFERENT PROGRAMS ON LOCAL MACHINE, CLUSTER OR HLRS
#
# Executable path for python on local computer
LOCAL_PATH_PYTHON                       {}
# Executable path for animator on local computer
LOCAL_PATH_A4                           {}
# Executable path for generator on local computer
LOCAL_PATH_G4                           {}
# Executable path for generator on local computer
LOCAL_PATH_LSOPT                        {}
# Executable path for ls-dyna on local computer
LOCAL_PATH_LSDYNA                       {}
# Executable path for python on cluster
CLUSTER_PATH_PYTHON                     {}
# Executable path for animator on cluster
CLUSTER_PATH_A4                         {}
# Executable path for generator on cluster
CLUSTER_PATH_G4                         {}
# Executable path for LS-OPT on cluster
CLUSTER_PATH_LSOPT                      {}
# Executable path for ls-dyna on cluster
CLUSTER_PATH_LSDYNA                     {}
#
#
# Extra call options for A4 and G4   (i.e. special versions)
A4_CALL_OPT                             {}
G4_CALL_OPT                             {}
# GNS software run mode
G4_BATCH_MODE                           {}
A4_BATCH_MODE                           {}
#
#############################################################################################
```

Bild A - 1: Initiale Optionen, Programmpfade und Aufrufoptionen für die zusätzliche Software

```
##########################################################################################
# CONNECTION INFO AND PARAMETERS FOR PROCESSING ON CLUSTER
#
# Username on the  cluster for ssh connection
USER                                    {}
# IP-Address of the cluster for ssh connection
CLUSTER                                 {}
# Number of processor for lsdyna simulation on local computer
PROCESSORS_NUMBER_LOCAL                 {}
# Number of nodes and processors for full model simulations
PROCESSORS_FULL_CLUSTER                 {}
# Number of nodes and processors for submodel simulations
PROCESSORS_SUB_CLUSTER                  {}
# Number of processors for all the small scripts to be run on cluster
PROCESSORS_CMD                          {}
# Maximum time limit defined in the PBS-Skript for cluster
MAX_TIME                                {}
#
##########################################################################################
# REFERENCE MODEL PARAMETERS
#
# Used solver (Pamcrash / Dyna)
SOLVER                                  {}
# Reference Model location. (Full or relative to working Directory Path)
LOCATION                                {}
# Input file to start the simulation
RUN_NAME                                {}
# FE-Model file which is modified to create submodel.
MOD_INCLUDE_NAME                        {}
# Output format (Only necessary for PamCrash: DSY / DSY.fz / )
RESULTS_FILETYPE                        {}
#
##########################################################################################
###
# SIMULATION PARAMETERS
#
# Solver options for running simulation on cluster
SOLVER_OPTIONS                          {}
#
##########################################################################################
```

Bild A - 2: Einstellungen für die Ressourcen, Modell- und Solveroptionen

```
################################################################################
# PARAMETERS FOR SUBMODEL GENERATION
#
# Debug mode: one step after another, you have to accept every step in the console
DEBUG_MODE                                  {}
# Select submodel generation algorithm
#                                       Options:
#                                       PBS: performance based submodel generation;
#                                       CBS: coordinate based submodel generation;
#                                       EPBS: extended PBS, combination of PBS and CBS;
#                                       RIGID: set crash irrelevant parts rigid)
GEN_ALGORITHM                               {}
#
# Method used to add extra Boundary conditions to the reduced model
#                                           Options:
#                                           DMA: Displacements Model Aproach
#                                           KMA: kinematic mass approach
BC_APROACH                                  {}
#
# decide if a script should clean potential errors from the submodels FE-inputdeck
CLEAN_FE_DECK                               {}
# max number of parallel submodel generations (geometry) and validations
# needs a lot of RAM, do not choose too many at once
MAX_PARALLEL_SUBMODEL_GENERATIONS           {}
# max number of parallel submodel simulations
MAX_PARALLEL_SUBMODEL_SIMULATIONS           {}
# Decide if models created during the process should be stored
SAVE_MILESTONES                             {}
# Maximum allowed time limit to create submodel
TIME_LIMIT                                  {}
# PIDs to block in the submodel | Also used to reserve PIDs which are to be optimized
#                                 using the submodel (PBS. ePBS)
# Multiple PIDs can be seperated by space (PID1 PID2 PID3 ...)
KEEP_PIDS                                   {}
# Group of Objects to block in the submodel | Objects included will be kept
#                                             at the submodel. (PBS, ePBS, CBS).
# This is an independent group. Not taken into consideration during Island evaluation.
# Thus MAY LEAD TO MODELS WITH ISLANDS.
# Multiple PIDs can be seperated by space (GROUP1 GROUP2 GROUP3 ...)
KEEP_GROUP                                  {}
# PIDs where the bead is inserted are blocked in submodel
BLOCK_BEAD_PIDs                             {}
# PIDs to overlook at the process (Avoid evaluation of certain PIDs - PIDs directly
erased
#                                 from model)
# Multiple PIDs can be seperated by space (PID1 PID2 PID3 ...)
OVERLOOK_PIDS                               {}
# PIDs representing special connection modelling (Thus not being detected as connector
#                                                 entities in G4)
SPECIAL_CON_PIDS                            {}
# PIDs where the creation of boundary conditions should be avoided
# Multiple PIDs can be seperated by space (PID1 PID2 PID3 ...)
BC_FREE_PIDS                                {}
################################################################################
```

Bild A - 3: Optionen für die Submodellgenerierung

```
##########################################################################################
# PBS/EPBS PARAMETER
#
# Model Discipline assuming that the x-coordinate is based on the vehicle orientation.
# Used for ePBS reduction Method. (FRONT / REAR / SIDE)
DISCIPLINE                                {}
#submodelsize (SUB) or threshold (THR): if sub is selected, the threshold doesn't need
# to be filled in
# if thr is picked the SUB_SIZE can be left empty
GEN_METHOD                                {}
# Determines the submodel size by finding the appropriate threshold value. Only used
# with GEN_METHOD "sub"
SUB_SIZE                                  {}
# Parameters for Connecting Island: (THRESHOLD, CI_VALUE), vary submodelsize, only used
# with GEN_METHOD "thr"
# THRESHOLD: Varies submodel size. Determines which parts will be deleted
# CI_VALUE: Determines distance to connect islands.
CI_PARAMETERS                             {}
# Evaluation function for submodel generation (1-plastic strain; 2-internal energy;
#                                              3-max. v. Mises)
EVAL_FUNCTION                             {}
#
##########################################################################################
# CBS PARAMETER
#
# Object type considered as basic entity for the reduction. (ELE: Element-Based;
#                                                             PID: Property based)
CBS_BASIC_ENTITY                          {}
# uses 2 sets of coordinates for box. Submodel will be generated inside the box.
BOUNDING_BOX                              {}
#
##########################################################################################
```

Bild A - 4: Optionen der einzelnen Verfahren zur Erstellung der Submodelle

```
#########################################################################################
# PARAMETERS FOR SUBMODEL VALIDATION
#
# Validation methods:
#                   - CURVE: analysing multiple responses like internal energy with an
#                            ISO curve rating
#                   - NODAL: calculating the differences between all nodes of the
#                            submodel and the fullmodel
#                   - DIFFC: using DIFFCRASH
#                   - OWNSC: using your own script
# Note: a '+' behind the method saves ratings of all methods, but decides based on the
#       chosen method, e.g.
# CURVE+ -> the submodels will be rated with the CURVE-method, but all
# resultfiles will be generated
VALI_METHOD                                 {}
# Additional parameters regarding the VALI_METHOD:
#
## CURVE and NODAL:
# Number of PIDs to be used to calculate mean validation parameter sorted by internal
# energy of PID
N_PID_VALIDATION                            {}
#
## NODAL:
# Exponent to amplify the differences in displacements
AMPLIFICATION                               {}
## DIFF:
# Path to Diffcrash 'bin' directory:
DIFF_PATH                                   {}
# License file of DIFFCRASH
DIFF_LIC                                    {}
## OWNSC:
# Command to start your script
COMMAND                                     {}
# File with a single validation value between 0 and 1
VALI_FILE                                   {}
#########################################################################################
```

Bild A - 5: Einstellungen der Submodellvalidierung

```
#################################################################################################
# PARAMETERS FOR OPTIMIZATION WITH LS-OPT
#
# Example: Edit MOD_INCLUDE_NAME File as ls-opt input file
# OPT_PID_ALL                                 {}
#
# Optimization parameters | LS-OPT Parameters to add in ls-opt input file
# Note: doubled DV-names are possible, but with same limits and initial values!!
OPT_L1_PID_1                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_2                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_3                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_4                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_5                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_6                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_7                                {} # DV-name, PartID, initial, lower, upper
OPT_L1_PID_8                                {} # DV-name, PartID, initial, lower, upper
# Optimization target
OPT_TARGET                                  {}
# Optimization goal (minimize or maximize)
GOAL                                        {}
# Number of iterations
ITERATIONS                                  {}
# Number of simulation points per iteration
SIM_POINTS                                  {}
# Number of parallel simulations per iteration
PARALLEL_SIMULATIONS                        {}
# Constraints for level 1 ("<" for upper and ">" for lower boundary)
L1_CONSTRAINTS                              {}
# If you enable the following option, substitute functions will be used, instead of the
# L1_RESPONSES
SUBSTITUTE_FUNCTION                         {}
# Files with list of structural responses
L1_RESPONSES                                {}
L2_RESPONSES                                {}
EXPRESSIONS                                 {}
#
#################################################################################################
```

Bild A - 6: Optimierungsparameter

```
##############################################################################################
# SESSION FILE NODES
#
OPTFOLLOWID1                                {}
OPTFOLLOWID2                                {}
OPTFOLLOWID3                                {}
B_SAEULE_UNTEN_LINKS                        {}
B_SAEULE_UNTEN_RECHTS                       {}
B_SAEULE_SCHLOSS_FS                         {}
B_SAEULE_SCHLOSS_BFS                        {}
KAROSSERIE_VORNE                            {}
RAHMENTUNNEL_MITTE                          {}
SCHWELLER_VORN_LINKS                        {}
SCHWELLER_VORN_RECHTS                       {}
TUERAUS_FS_UNTEN                            {}
TUERAUS_FS_OBEN                             {}
TUERAUS_BFS_UNTEN                           {}
TUERAUS_BFS_OBEN                            {}
ZAE                                         {}
##############################################################################################
```

Bild A - 7: Optionen für die Auswertung von Fahrzeugmodellen