Contents

1 Die Porsche AG

Porsche wurde 1931 von Ferdinand Porsche gegründet. Seit 1945 stellt die Firma Sportwagen und Luxusautos her.
Im Finanzjahr 2018 erzielte der Konzern einen Umsatz von 25,784 Mrd. Euro.
Die aktuelle Produktpalette der AG umfasst für Serienfahrzeuge:

- Macan (2013)
- Cayenne (2002)
- 911 (1963)
- Boxter (1996)
- Panamera (2009)

Weissach

Seit 1971 werden diese Fahrzeuge am Standort Weissach entwickelt. Das sogenannte "Entwicklungszentrum Weissach", kurz EZW, auf dem die Porsche AG ein Fahrzeug entwickeln, testen und justieren kann, bevor es in Serienproduktion in Zuffenhausen oder einem anderen Standort geht.
Alleine an diesem Standort beschäftigt Porsche um die 6500 Mitarbeiter und Mitarbeiterinnen.
Zuletzt wurde hier auch der Porsche 918 entickelt: Ein Supersportwagen mit hybridem Antrieb und einem Listenpreis von mehr als 800 000 Euro.
Das in Weissach entwickelt wird, sieht man schon an der Anzahl an Patenten die hier jährlich angemeldet werden. Knapp 400 pro Jahr sind es aktuell, mit mehr als 7000 bereits zugelassenen Patenten.
Die Signifikanz des Standortes Weissach wird immer wieder durch spezielle Produktvarianten oder Ausstattungen hervorgehoben. Ein Beispiel hierfür ist das Weissach-Paket, welches man für den 911 bestellen kann. Teil hiervon sind typischerweise die Farben Grün und Weiß, zusammen mit besonderen Aero-Elementen.

2 Das Praktikum

2.1 Fachpraktikum

Für das Fachpraktikum mussten noch 14 Wochen absolviert werden. Alle Bereiche des Grundpraktikums wurden bereits im Vorpraktikum abgearbeitet, weshalb sich das Fachpraktikum nur auf das in den Praktikumsrichtlinien der RWTH Aachen genannte Fachpraktikum Teil A und Teil B bezieht.
Die 8 Wochen aus Teil B konnten alle mit FP7 abgegolten werden, da in Weissach ausschließlich Fahrzeugentwicklung betrieben wird. Die übrigen sechs Wochen aus Teil A wurden mit FP4 - **Messen, Prüfen, Qualitätskontrolle** - mit einem Anteil von drei (3) Wochen und FP6 - **Montage** - mit weiteren drei Wochen ausgeführt.

2.2 Fachbereich Porsche

Bei Porsche wurde das Praktikum im Fachbereich **Aerodynamik und Thermomanagement** absolviert.
Diese Abteilung ist für die thermische Absicherung der Fahrzeuge und die aerodynamische Auslegung der Neuentwicklungen zuständig.
Neben Versuchen im Klimawind- oder Windkanal werden auch Erprobungsfahrten, Simulationen und andere Mess- und Testmethoden verwendet um die hohen Standards bei Porsche aufrecht zu erhalten.
Im engen Austausch mit den anderen Abteilungen und Bauteilverantwortlichen wird so jedes Fahrzeug und Einbauteil erprobt und entsprechend der geltenden Normen und Rechtssprechungen, sowie der Anforderungen die Porsche an sich selbst stellt, entwickelt und zur Serienreife gebracht.

3 Praktikumsbericht

3.1 FP7 - Entwicklung (8 Wochen)

Der Anfang des Praktikums wurde mit Einarbeiten verbracht.
Zu den verwendeten Tools gehören unter anderem PowerFlow, PowerTherm, ANSA, Star CCM+ und MATLAB.

Abb. 3.1: PowerFlow von 3DS

Vor allem PowerFlow spielt eine große Rolle, da es für Fahrzeugsimulationen geeignet ist. Das Programm baut auf der **Lattice-Boltzmann-Methode** auf.

3.1.1 Die Lattice-Boltzmann-Methode (LBM)

Die Lattice-Boltzmann-Methode beruht auf einer Vereinfachung der Euler-Gleichung und ist Aufgrund geringer Speicheranforderungen pro Zelle geeigenet um komplexe Geometrien zu berechnen.
Da Fahrzeuge nur selten bis in den kompressiblen Machzahlbereich von

$$Ma > 0.3 \tag{3.1}$$

ausgelegt werden, können die getroffenen Vereinfachungen angewendet und dadurch die **LBM** für die Fahrzeugauslegung verwendet werden.
Da die LB-Methode leicht für die Wärmeübertragung erweitert werden kann, spielt bei der Auslegung der Fahrzeuge auch PowerTHERM eine große Rolle für die Bauteilsicherung.

3.1.2 Einarbeitung

Für die Einarbeitung in PowerTherm und PowerFlow wurden Tutorien durchgearbeitet.
Um sich mit den Funktionalitäten des Programms vertraut zu machen, wurde zuerst eine einfache NACA-Flügelgeometrie geladen und diese dann mit den entsprechenden Netzen und Berechnungsvolumen ausgestattet um eine Berechnung zu ermöglichen.
Anschließend mussten die Randbediungen an den Simulationsvolumengrenzen definiert werden und die Strömungskonditionen definiert werden. Hierzu zählen die Reynoldszahl, Temperatur, Strömungsgeschwindigkeit und Vieles mehr.

Nachdem die Berechnung abgeschlossen ist, wird dann PowerCASE, ein Programm zur Darstellung der Simulationsergebnisse, verwendet um sich verschiedene relevante Parameter oder Geschwindigkeiten, Drücke und vieles mehr anzeigen zu lassen.

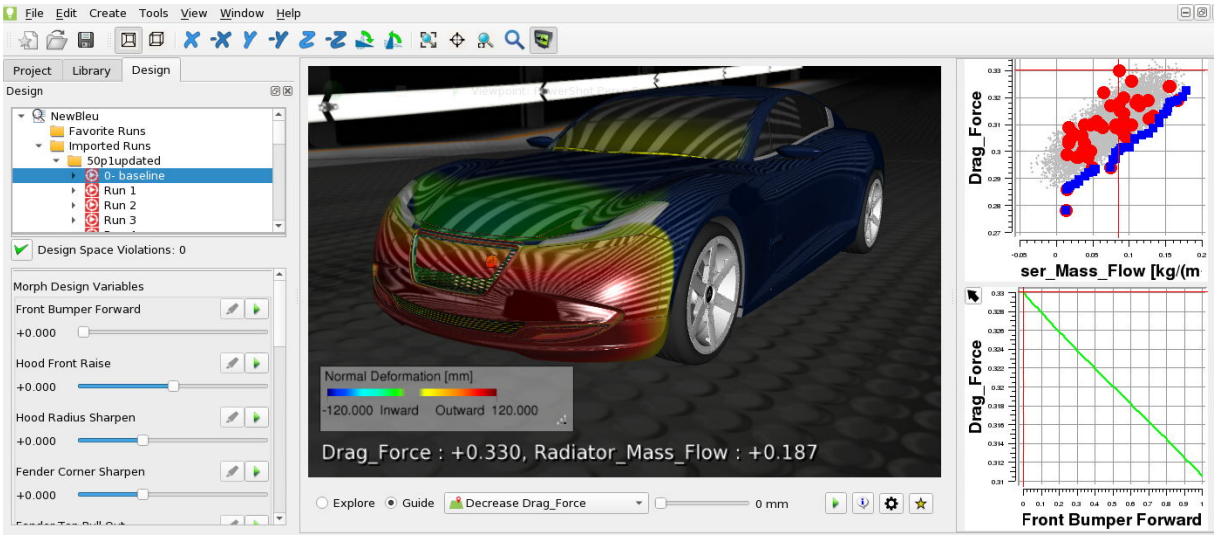

Abb. 3.2: PowerFlow User-Interface

Nach dem NACA-Profil wurde als nächstes mit ANSA, einem Pre-Processing Tool, gearbeitet. In diesem können Geometrien für die Simulationen aufarbeitet und vernetzt werden.

Abb. 3.3: ANSA Logo

Abgesehen von der recht veralteten Benutzeroberfläche ist ANSA was die Geometrievorbereitung für PowerFlow gut geeignet, da die Geometriegüte in PowerFlow für verwertbare Ergebnisse nicht perfekt sein muss.
In ANSA wurde nun eine einfache Geländewagengeometrie aufgebaut, die dann in PowerFlow geladen und mit einem einfachen Simulationsvolumen berechnet wurde. Anschließend wurden die Ergebnisse wieder in PowerCASE ausgewertet und mit anderen Praktikanten verglichen.

Wenn man PowerFlow mit Star CCM+ vergleicht kommt heraus, dass CCM+ zwar etwas genauere Ergebnisse liefert. Dafür benötigt es jedoch eine sehr viel höhere Güte an eingeladener Geometrie, was es für die aktuelle Anwendung zum Teil ungeeignet macht.

Am Ende der Einarbeitung wurde noch eine Einweisung in die Fahrzeughandhabung auf dem Werksgelände durchgeführt. Da die Fahrzeuge selbst auf der niedrigsten Austattungsstufe mehr **kW** entwickeln als die meisten andere Kraftwagen, ist hier besondere Vorsicht geboten. Vor allem, wenn man Fahranfänger ist.

3.1.3 Datenauswertung in Excel

Als erstes Projekt wurde eine Datenauswertung in Excel zugewiesen.
Um mit den großen Datenmengen umzugehen wurde in Excel mit Hilfe des integrierten VBA-Editors ein Makro geschriebe. Dieser kann nach minimaler Vorarbeit des Anwenders automatisch die Datenmengen auswerten und das entsprechende Kollektiv erstellen.
Die hinterlegte Logik und Datenstrukturen mussten zuerst entworfen und dann implementiert und getestet werden.
Da VBA (Virtual Basic) eine free-type Programmiersprache ist, können Datentypen direkt im Code gecasted und definiert werden. Dies ist im Vergleich zu einer Sprache wie JAVA zwar schneller zum Tippen, ist aber bei der Code-Compilation und späteren Ausführung langsamer, da der Interpreter die Datentypen bei der Ausführung zuweisen muss.
Ein weiteres Problem, das speziell bei der Anwendung von VBA in der Microsoft Office-Umgebung entsteht, ist die extrem langsame Zugriffszeit von dem VBA Editor in das entsprechende Dokument in z.B. Excel.

3.1.3.1 Makroversion 1.0

In der ersten Version des Makros wurde in den entsprechenden Iterationsschleifen direkt auf das jeweilige Worksheet in Excel zugegriffen um die Messdaten auszulesen und vergleichen zu können.
Die grundliegende Logik ist in Abbildung **??** zu sehen.

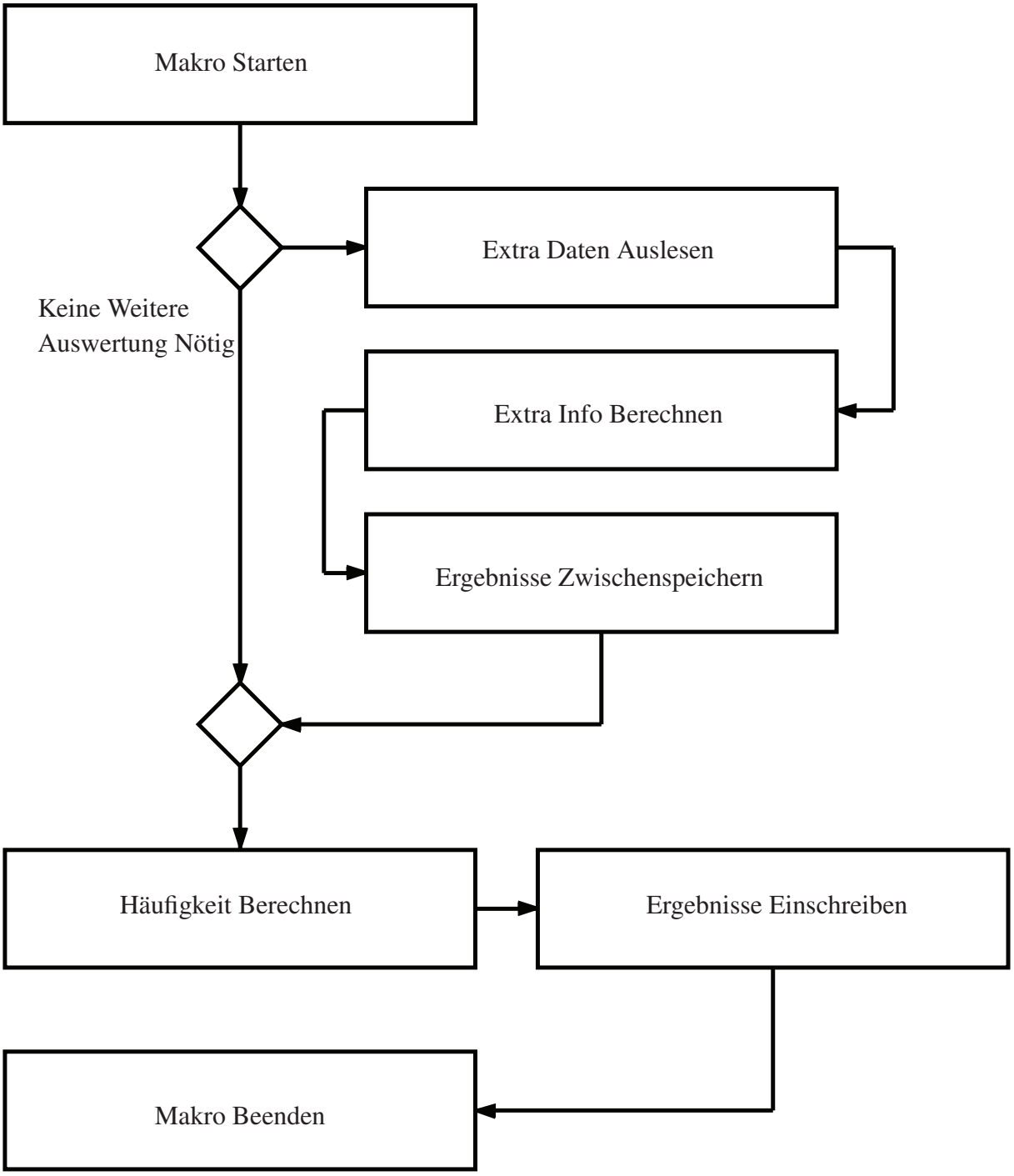

Abb. 3.4: Makro-Logik

Da die Messdatenmengen des Versuchs aus mehreren tausend Messpunkten besteht, brauchte der Makro für die Iteration über alle Datenpunkte geschätzte 120 Stunden. Diese relativ lange

Bearbeitungszeit war zwar zum Teil auf die Hardware zurückzuführen, wurde jedoch vor allem von dem oben beschriebenen Problem hervorgerufen.
Bei einer Messungsreihe mit 5 verschiedenen Messumgebungen die aus je 12 Messtemperaturen bestehen wird die Datenmenge schnell sehr groß. Die Messdaten wurden auf eine uniforme Zeitbasis von 1 Hz bestimmt. Je nach Messumgebung variierte auch die Messdauer. In der Testumgebung wurden daher nur etwa 20 000 Datenpunkte ausgewertet.

Da selbst in der Testumgebung die Berechnungsdauer bei mehreren Stunden lag, musste die Berechnungszeit verkürzt werden.

3.1.3.2 Makroversion 1.5

Nach erweiterte Recherche konnte die Berechnungszeit verkürzt werden, indem die Datenauslegung am Anfang der Iteration einmal durchgeführt wurde und dann nur noch mit den eingelesenen Daten direkt in der VBA Umgebung gerechnet wurde.
Hier war nun vor allem die Indizierung der Matrizen eine Herausforderung. Je nachdem ob bestimmte Messinformationen mit ausgewertet werden mussten, mussten die Indizes und die Iterationsschrittweite angepasst werden.
Dies wurde durch eine VBA Userform gelöst, in welcher der Benutzer angeben konnte ob die Information benötigt wird. Im Code wurde basierend auf dieser Auswahl eine Iterationsvariable und die Iterationsgrenzen angepasst.
Nach weiterem Testen war diese Art der Auswertung erfolgreich.

Nun musste nur noch das Problem der Matrixindizierung gelöst werden. Um den Makro möglichst benutzerfreundlich zu gestalten, mussten die Matrixgrenzen so einprogrammiert werden, dass die Auswertung auch funktioniert wenn der Benutzer die Daten falsch in Excel einfügt oder Daten fehlen. Hierfür mussten in die Iterationsschleifen Logikabfragen implementiert werden, da die Fehlerbehandlung in VBA fragwürdig ist. Diese Maßnahme verlangsamten die Auswertung wieder etwas. Jedoch war es möglich durch Deaktivieren der Bildschirmanpassung und Hintergrundberechnung von Excel selbst diese Rechenzeit wieder auszugleichen.

3.1.3.3 Makroversion 2.0

Nachdem die Matrixauswertung erfolgreich war, mussten die Auswertungsergebnisse noch aufgearbeitet werden. Die Ergebnismatrix wurde dann in ein eigenes Worksheet geschrieben. In diesem Worksheet ist Code hinterlegt, der die Matrixdaten in einer 3D-Grafik wiedergibt.
Mit dieser Matrix und dem entsprechenden Graphen können dann die Gespräche mit den Zulieferern und Lebenszeitexperten geführt werden im das Produkt möglichst optimiert zu entwickeln.

Durch die oben genannten Optimierungen konnte die Berechnungsdauer in der finalen Version des Makros auf etwa 20 Minuten reduziert werden. Diese Zeit wurde in der echten Auswertungsumgebung mit ca. 70 000 Datenpunkten erreicht.

Ein weiterer Grund warum diese Auswertung jedoch in Excel nur bedingt Effizient ist, ist der pre-processing Aufwand um verwertbare Ergbenisse zu erlangen. Die Daten müssen zuerst mit anderen Programmen aus der Messdatei ausgelesen und dann in Excel exportiert werden. Aus dieser Excel-Datei müssen die relevanten Daten dann in das Excel-Workbook mit dem hinterlegten Makro übertragen werden. Bei insgesamt 60 Messläufen ist dieser manuelle Aufwand wesentlich größer als die später benötigte Zeit um die Auswertung durchzuführen.

3.1.3.4 Das Excel-Workbook

Das Excel-Workbook in welchem der Makro hinterlegt ist wurde als Vorlage (Template) erstellt und in einem entsprechendem Ordner abgespeichert um den restlichen Abteilungsmitgliedern den Zugriff auf das Workbook zu ermöglichen.

Da in der Fahrzeugherstellung viele verschiedene Umweltzustände und dadurch Temperaturen dargestellt und getestet werden müssen, sind in dem Excel-Workbook verschiedene Worksheets hinterlegt, in denen die relevanten Umgebungsdaten hinterlegt sind.
Um diese in der Berechnung automatisch mit zu Berücksichtigen müssen diese vor Starten des Makros entsprechend über eine User-Form ausgewählt werden. Um die Bearbeitung zu erleichtern ist als Standard-Wert der am häufigsten verwendete Fall hinterlegt.

Die Lebensdauerfaktoren werden in dem ersten Worksheet festgelegt und berechnet. Der Makro liest sich diese Werte entsprechend aus und verwendet sie. Daher ist es wichtig, dass das Makro-Template immer den neuesten Werten entsprechend angepasst wird. Diese Arbeit muss leider weiterhin manuell ausgeführt werden, da die Lebensdauerfaktorenberechnung oft in Arbeitskreisen durchgeführt wird und diese Faktoren dann nicht in einer zentralen Datenbank hinterlegt werden.

(Notiz)

In diesem Bereich ist nach Meinung des Authors weitere Entwicklung gewünscht um Arbeitsprozesse zu Streamlinen und die gesamte Entwicklung effizienter zu gestalten.

(Notiz-Ende)

Für die Weiterverarbeitung der berechneten Daten ist es nun günstiger, die resultierenden Matrizen aus dem Makro-Worksheet in ein Worksheet ohne hinterlegte Makros und Berechnungsparametern zu transferieren. Dies resultiert aus der Geheimhaltungsklausel der Porsche AG. Die Makro-Datei hat unter anderem konzern-interne Daten hinterlegt, die nicht von Zulieferern oder Dritten Parteien eingesehen werden dürfen.
Hier könnte der Makro erweitert werden, damit die Ergebnisse automatisch in ein neues Excel-Workbook übertragen werden.

Ein weiterer Mangel an dem Marko-Workbook besteht darin, dass während der Makro läuft Excel unbenutzbar auf diesem Rechner wird. Der von Excel blockierte Thread wird komplett von dem Makro übernommen und lässt das Betriebssystem Excel als ein Programm ohne aktive Rückmeldung anzeigen.
Dies muss leider mit der aktuellen Version des Makros hingenommen werden.
Aufgrund des Wunsches diesen Workflow in MATLAB effizienter zu gestalten wurde an dieser Stelle der Makro nicht mehr weiterentwickelt.

3.1.3.5 Endresultat

Wenn der Makro erfolgreich durchlaufen ist, werden je nach Benutzereingabe eine oder zwei Matrizen ausgegeben.
Die erste Matrix wird immer erstellt und besteht aus dem Temperaturkollektiv der Messdateien. die zweite Matrix wird nur erstellt, wenn die Information in der Messung enthalten und diese Information für die weitere Entwicklung benötigt wird. Da diese Matrix den Großteil der Berechnungsdauer in Anspruch nimmt, wäre es redundant diese Berechnung durchzuführen wenn sie nicht benötigt wird.
Da dieser Gesamtprozess wie oben aufgeführt ineffizient ist, bestand einer der weiteren Aufgaben darin diesen Prozess in MATLAB zu verbessern und den Workflow damit komplett zu überarbeiten.

Nach der Erfahrung des Authors nimmt beim Entwicklungsprozess die Datenaufarbeitung einen großen Anteil der gesamt benötigten Zeit und damit einen Großteil der Kosten auf sich. Die Kosten stellen sich aus Lizenzkosten der benötigten Programme und den direkten Personalkosten zusammen.
Daher kann mit einer entsprechenden Workflow-Verbesserung auch eine Kostensenkung erwirkt werden.

3.1.4 Aktueller Workflow der Messungsauswertung

Der in **Datenauswertung in Excel** [3.1.3] beschrieben Workflow beinhaltet nur einen Bruchteil der benötigten Arbeitsschritte um verwertbare Ergebnisse aus dem Gesamtprozess zu gewinnen.

Da die Messdaten nur der letzte Teil einer Arbeitskette darstellen, wird nun auf den gesamtem Prozess eingegangen. Dieser wird erläutert, die Problemstellen hervorgehoben und eine Lösung in MATLAB für den Workflow vorgestellt.

3.1.4.1 Messstellenplan

Der Messstellenplan oder **(MSP)** wird von den Messstellentechnikern und dem zuständigen Mitarbeiter erstellt und in Zusammenarbeit mit den relevanten Bauteilverantwortlichen und anderen Parteien kontinuierlich angepasst.
Die pro Versuchsreihe oftmals eine signifikante Anzahl von Messungen gefahren werden ist es recht aufwändig den Messstellenplan jedes Mal an die neue Messung anzupassen.
Der Messstellenplan wird über konzern-eigene Datenverwaltungstools für jeden Wagen gepflegt. Dieser kann dann in Excel exportiert werden und für jede relevante Messung referenziert werden.
Hierbei entstehen schnell eine unübersichtliche Anzahl von Excel Dateien mit zum Teil extensiven Bezeichnungen um eine genaue Zuordnung zu ermöglichen.

3.1.4.2 Grenztemperaturen

Die Grenztemperaturen werden in einer seperaten Datei für jede Messstelle, insofern diese Messstelle eine Grenztemperatur hat, festgelegt.
Analog zum **MSP** werden diese Grenztemperaturpläne in Excel exportiert.
Aktuell werden diese Dateien dann in einer seperaten Excel-Datei mit hinterlegten Makros eingelesen und ausgewertet. Da VBA eine recht alte und langsame Programmiersprache ist, ist diese Auswertung oft zeitaufwändig oder stürzt einfach ab.

3.1.4.3 Messungen

In der Excel-Datei können die Messungen aus ihrem nativen MDF-Format nicht eingelesen werden.
Daher müssen um die Messungen zuzuordnen erst die relevanten Kanäle mit Hilfe eines Porsche-internen Tools ausgelesen und in Excel exportiert werden.
Diese exportierten Excel-Dateie sind regelmäßig weiter über 100mb groß und benötigen dementsprechend lang um eingelesen zu werden.
Bei mehreren Messungen pro Auswertung ist hier der Aufwand schnell groß, da jede Messung einzeln verarbeitet werden muss.

3.1.4.4 Simulationen

In dem aktuellen Workflow gibt es keine Möglichkeit Simulationsergebnisse der Auswertung anzufügen.
Da diese aber oft für Erstauslegungen oder Konzept-Bestätigung verwendet werden, wäre es von Vorteil diese direkt mit einsehen zu können.

3.1.4.5 Ergebnisausgabe

Die Ergebnisse werden entweder als PDF oder als Excel Workbook ausgegeben.
Diese werden dann oft in Powerpoint oder anderen Medien weiterverwendet um entsprechende Maßnahmen abzuleiten.

3.1.5 Workflow in MATLAB

Um diesen Workflow effizienter zu gestalten wurde er in MATLAB übertragen und dort in einem Stand-alone Programm entworfen.

Der erste Arbeitschritt bestand darin, sich mit der MATLAB-App Designer Umgebung vertraut zu machen. Da MATLAB sehr effizient große Matrizen verarbeiten kann, bot sich das Programm für diese Art der Auswertung an.
Nach Durcharbeitung der Tutorials wurde als nächstes der Workflow entworfen.
Dieser sollte idealerweise die vorher genannte Auswertung in Excel und anderen Programmen komplett ersetzen.
Damit das Programm auch von den Kollegen eingesetzt wird, wurden die relevanten Parteien von Anfang an in den Entwicklungsprozess miteingebunden. Hierfür wurden wöchentliche Meetings angesetzt.
Im Laufe der Entwicklung des Workflows und Programms wurde schnell offensichtlich, wie wichtig diese andauernde Rückmeldung mit den anderen Teammitgliedern für eine erfolgreiche Implementierung ist.

3.1.5.1 Vorarbeit

Um den Workflow zu entwerfen musste erst der gesamte Auswertungsprozess verstanden und nachvollzogen werden.
Um dies zu erreichen wurden kleine Aufgaben entlang des gesamten Prozesses übernommen und selbstständig bearbeitet. Bei Unklarheiten wurden die relevanten Kollegen hinzugezogen.

Als nächstes wurde der Ablauf ausgelegt und dann Überlegungen angestellt, wo er beschleunigt und effizienter durhgeführt werden kann.

3.1.5.2 Verbesserungen

Die einfachste Verbesserung wurde im Bereich der Messstellenpläne implementiert. Diese können nun aus der exportierten Excel-Datei direkt in das Programm geladen werden.
Sollte eine neue Messung einen veränderten Messstellenplan haben, kann diese Veränderung in den Messstellenplan der bereits eingelesen ist eingefügt werden.
Um das weitere Vorgehen zu erleichtern und das Vergleichen von Messungen zu ermöglichen, werden neue Messstellen an den alten Plan angefügt und dann einsortiert. Keine Messstellen werden entfernt, damit alle vorhergehenden Messungen ihre Aussagekraft bewahren.

Des Weiteren wird der Grenztemperaturplan nur temporär eingelesen und die Werte für die relevanten Messstellen ausgelesen, wonach der Grenztemperaturplan wieder aus dem Arbeitsspeicher gelöscht wird.
Analog kann, sollte der Messstellenplan angepasst werden, die Grenztemperatur für die neuen Messstellen dann hinzugefügt werden. Sollte sich eine Grenztemperatur für eine bestehende Messstelle geändert haben, wird die alte Grenztemperatur automatisch überschrieben. Diese Maßnahme soll der Benutzerfreundlichkeit des Programms dienen.

Um die Messungen einzulesen, bedarf es nun keinem Dritt-Programm mehr. Das MATLAB-Programm gleicht den eingelesenen Messstellenplan in MDF-Format mit der gesamten Messung ab und lädt nur die Messkanäle die für den Plan von Relevanz sind.
Sollten weitere Kanäle wie zum Beispiel die GPS-Geschwindigkeit oder Ähnliches wichtig sein, können diese Messstellen manuell in den Messstellenplan in MATLAB eingefügt werden. Hierdurch werden diese Kanäle dann mitverwertet.
Genaueres hierzu in Kapitel 3.1.5.3.

Die Simulationsergebnisse liegen normalerweise in **.cvs** oder **.xml** Format vor.
Da MATLAB diese Formate nicht nativ einlesen kann, muss für diese Funktion noch eine Routine geschrieben oder eine verfügbare Funktion gefunden werden.
Aufgrund der wichtigeren Grundfunktion des Programmes wurde diese Funktion aber auf die 'Nice-To-Have'-Liste gesetzt. Daher wird an dieser Funktion nur gearbeitet, wenn der Rest des Programms zufriedenstellend läuft.

Das Ziel der Verbesserungen ist eine signifikante Zeiteinsparung und Vereinfachung des Workflows der Messungsauswertung und Ergebnisinterpretation.

3.1.5.3 Implementierung und Probleme in der Entwicklung

Version 0.1

In der ersten Version des Programms wurde der Fokus darauf gesetzt die Grundfunktionen der Auswertung linear zu implementieren.
Dies sollte auch helfen den Grundprozess besser zu verstehen. Das Programm selbst sollte dann iterativ verbessert unde ausgebaut werden bis hin zur finalen Version.

Die ersten Probleme bei der Messstellenplanverwertung traten recht früh auf. Einige Messstellenpläne aus dem alten und dem neuen Datenverwaltungstool von Porsche waren zwar alles Excel-Dateien, aber mit verschiedenen Datentypen abgespeichert.
Zwischen diesen Datentypen gab es bei der Spaltennummerierung und -benennung unterschiede, was das auslesen der korrekten Information erschwert.
Nach Rücksprache mit dem Betreuer ergab sich, dass in kurzer Zeit das alte Datenverwaltungstool komplett durch das neue ersetzt werden würde. Daher wurde das Problem teilweise gelöst, jedoch bestand weiterhin in der Nomenklatur eine Diskrepanz.
Weiter Maßnahmen um die Spaltenbenennung zu vereinheitlichen sind daher nötig.
Meiner Meinung nach würde dies den Workflow verbessern, da man bei der Auswertung, ob manuell oder per Programm, immer mit der gleichen Informationsstruktur konfrontiert ist.
Die endgültige Lösung des Problems bestand dann darin, die Spalten mit den entsprechenden Namen einzulesen und in einer vordefinierten Reihenfolge zu speichern.
Als weitere gewünschte Funktion sollte der MSP direkt in MATLAB bearbeitet werden können. Da die grafische Umgebung von MATLAB in diesem Fall eher langsam arbeitet, wurde abgeraten dies im Programm zu machen.
Die Möglichkeit Messstellen umzubenennen, zu löschen oder hinzuzufügen wurden dennoch grafisch implementiert.

Um den Grenztemperaturplan einzulesen, der auch in Excel exportiert wird, konnten die bereits implementierten Methoden wiederverwendet werden.
Analog zum Messstellenplan ist das hinzfügen, löschen und umbenennen oder editieren der Grenztemperaturen und Messstellen hier möglich. Dafür wird die eingelesene Information in einer Tabelle dargestellt.

Die signifikanteste Zeiteinsparung wird bei dem Auswerten der Messungen erzielt.
Um die Messungen einzulesen wurden Porsche-interne Routinen verwendet. Diese lesen aus den **.mf4**-Messdateien die relevanten Informationen wie Zeitskala, Werte, Messstellennamen und -info aus.
Diese Informationen werden in einem Cell-Array gespeichert und können so abgerufen werden.
Dann gleicht eine Funktion die Messstellennamen mit dem Messstellenplan und Grenztemperaturplan ab und speichert alle gefundenen Übereinstimmungen in ein weiteres Cell-Array, in dem auch der MSP mit den Grenztemperaturen ist, ab.
Dieses Array ist vordefiniert als 3-D Array, welches in Richtung der Messstellen, oder der ersten Dimension, dynamisch erweitert werden kann.
Obwohl das dynamische Erweitern von Arrays in MATLAB die Rechendauer verlängert, war dies die effizienteste Lösung.
Die zweite Dimension ist definiert und beinhaltet alle relevanten Daten der einzelnen Messstellen, sowie die jeweiligen Messstellen.

Die dritte Dimension beinhaltet die Messdaten der Messstellen. Diese Dimension wurde als k vordefiniert mit

$$k = 100 \tag{3.2}$$

Da MATLAB sonst die zum Teil aus mehreren Hundert Messstellen bestehenden Messungen dynamisch hinzufügen müsste, was zu einer signifikanten Verlängerung der Einlesezeit geführt hätte.
Vor dem Einlesen jeder Messung wird der Benutzer gefragt, ob die Messung neue Messstellen beinhaltet. Diese können dann bei Bedarf aus Excel eingelesen und zu den existierenden Messstellen hinzugefügt werden.
Die Messinformation der Messungen werden in einem zweiten Array abgespeichert, dass mit den gleichen Indizes arbeitet. So können jederzeit alle Information zusammen mit den Messungen abgerufen werden.

3.1.5.4 Auswertung

Für die Auswertung müssen nach dem einlesen aller Daten noch einige Parameter vom Benutzer angepasst werden, je nachdem welche Informationen betrachtet werden sollen.
Dann wird aus den Daten eine PDF-Datei ertellt, die alle Messkurven mit den Informationen abbildet.

3.1.5.5 Schwierigkeiten in MATLAB

Das größte Problem mit MATLAB ist die langsame grafische Umgebung.
Da jedoch die Konsolenumgebung schwieriger zu bedienen ist, wurde die grafische Lösung gewählt.

Eine weitere Schwierigkeit stammt aus dem Erstellen einer Speicherungsdatei. Um nicht bei jeder Teilauswertung alle Dateien neu einlesen zu müssen, ist ein weiterer Schritt die Erstellung einer solchen Speicherdatei zu ermöglichen.
Hierfür müssen alle Daten die in der GUI hinterlegt sind ausgelesen und abgespeichert werden. Dies wird bearbeitet, wenn am Ende des Projekts noch Zeit übrig ist.

3.2 FP4 - Messen, Prüfen, Qualitätskontrolle (3 Wochen)

3.2.1 Messstellenpläne

Wie in Abschnitt 3.1 ausgeführt, war die Auswertung der Messergebnisse nur ein Teil der Arbeit.
Um die Messergebnisse für die Auswertung zu gewinnen, mussten Vorarbeit und vor allem Messungen getätigt werden.

Bei dem Erstellen von Messstellenplänen war vor allem das Auswählen der Messstellennamen von großer Bedeutung.
Die Namen müssen eindeutig und leicht Verständlich gewählt werden um sicherzustellen, dass auch Dritte den Plan schnell verstehen können.

Die Bearbeitung und Erstellung dieser Pläne, die oftmals mehrere hundert Messstellen beinhalten, ist eine langsame und schlecht automatisierbare Tätigkeit.

3.2.2 Messungen

Ein Teil der Messungen kann auf der firmeneigenen Teststrecke in Weissach durchgeführt werden.

Abb. 3.5: Die Teststrecke in Weissach

Die Rundstreckentests werden zum Teil vor Ort durchgeführt. Wenn ein wärmeres Klima oder andere Streckenbedingungen erforderlich sind, wird auf eine andere Teststrecke im südlichen Teil von Europa ausgewichen.

Literaturverzeichnis

[1] R. OLSSON. Mass criterion for wave controlled impact response of composite plates. In: Composites Part A: Applied Science and Manufacturing 31 (2000), pp. 879–887.

[2] H. HERTZ. Über die Berührung fester elastischer Körper. In: Journal für die reine und angewandte Mathematik 92.92 (1881), pp. 156–171.

[3] K. KARAS. Platten unter seitlichem Stoß. In: Ing. Arch 10 (1939), pp. 237–250.

[4] S. TIMOSHENKO. Über die Biegung der allseitig ünterstützten rechteckigen Platte unter Wirkung einer Einzellast. In: Der Bauingenieur 2.2 (1922), pp. 51–54.

[5] C-T. WANG. Nonlinear large-deflection boundary-value problems of rectangular plates. In: National Advisory Committee for Aeronautics, Washington D.C. (1947).

[6] K. N. SHIVAKUMAR, W. ELBER and W. ILLG. Prediction of Impact Force and Duration Due to Low-Velocity Impact on Circular Composite Laminates. In: Transactions of the ASME 52 (1985), pp. 674–680.

[7] G. M. KOLVIK. Higher Order Shear Deformation Plate Theory. Master's Thesis. Oslo: University of Oslo, May / 2012.

[8] E. ONATE. Thick/Thin Plates. Reissner-Mindlin Theory. In: Structural Analysis with the Finite Element Method, pp. 291–381.

[9] H. DJOJODIHARDJO and A. S. MAHMUD. Computational Modeling, Simulation, and Tailoring of Nonpenetrating Impact on a Generic Structure. In: Journal of Aerospace Engineering (2015), pp. 04015061-1 - 04015061–9.

[10] W. MINGYU. Development of a semi-analytical approach for the application of large deflection plate theory on arbitrary load positions. Mini-Thesis. Aachen: RWTH Aachen, 2019.

[11] G. JACOBS. Maschinengestaltung III. 04/2019. Aachen: Druck & Verlagshaus Mainz, 2019.

[12] Structural Analysis with the Finite Element Method.

[13] C. M. HARRIS and A. G. PIERSOL. Harris' Shock and Vibration Handbook. 5th ed. McGraw-Hill, 2002. ISBN: 0-07-137081-1.

[14] P. A. KELLY. Mechanics Lecture Notes: An Introduction to Solid Mechanics. Auckland, NZ, 2020.