DLR-IB-AT-KP-2017-160

Parametrische Modellierung der Mantelflächen von Propellerblättern zur weiteren Verarbeitung in CAD-Systemen

Bachelorarbeit

Christian Palmberg

Deutsches Zentrum
DLR für Luft- und Raumfahrt

Abstract deutsch

Diese Bachelorarbeit befasst sich mit der Aufgabenstellung der Implementierung einer Software-Bibliothek zur parametrischen Modellierung von Propellerblättern zur weiteren Verarbeitung in CAD-Systemen. Die modellierten Propellerschaufeln sollen zukünftig dazu dienen, CFD-Rechnungen an den generierten Modellen auszuführen. Die Ergebnisse dieser Rechnungen sollen genutzt werden, um die implementierten Verfahren zur Strömungsrechnung von Propster, einem DLR-internen Software-Tool, zu validieren. Für die Generierung der Mantelflächen wurden zwei Bibliotheken mit der entsprechenden Funktionalität in Betracht gezogen. Für die beiden Bibliotheken Open Cascade Technology (open source) und BladeGenerator (DLR-interne Software) wurde jeweils das Modell eines Prototyps ausgearbeitet, welches Vor- und Nachteile des jeweiligen Verfahrens aufzeigte. Als zielführend wurde die Implementierung der Generierung von Mantelflächen von Propellerschaufeln mit der Bibliothek Open Cascade Technology betrachtet. Als Ergebnis liegt eine Bibliothek in der Programmiersprache C++ vor, die in der Lage ist, aus einer Parametrisierung eines Propellers, die im Rahmen dieser Arbeit vorgestellt wird, eine Mantelfläche der Propellerschaufel zu generieren und diese für die weitere Verarbeitung in CAD-Systemen durch die Datenformate STEP und IGES bereitzustellen. Die Bibliothek bietet Funktionen zur Generierung von Propellerschaufeln mit und ohne Pfeilung und Neigung. Profile einer Propellerschaufel können bei der Generierung der Mantelfläche über die Sehne, den Flächenschwerpunkt oder über eine Auffädellinie gefädelt werden. Zur Rekonstruktion einer Auffädellinie stehen verschiedene Funktionen bereit. Die Präzision bei der Rekonstruktion der Propellermantelfläche aus der Parametrisierung wurde durch diverse Anwendungstests nachgewiesen. Als Anwendungsfälle dienten die Propeller SRII und SRIII der NASA. Mit Hilfe der Bibliothek besteht nun die Möglichkeit, Mantelflächen von Propellerschaufeln zu generieren, die anschließend für CFD-Rechnungen genutzt werden können, sodass mit den Ergebnissen dieser Rechnungen eine Validierung der Strömungsrechnungsverfahren von Propster stattfinden kann.

Abstract english

This bachelor thesis is concerned with the task of implementing a software library, which provides functionalities to parametrically model propeller blades for further processing in CAD-systems. Those modeled propeller blades will serve for CFD-calculations in the future. Results of those calculations will be used to validate implemented processes of aerodynamic fluid flow calculations in Propster, an internal software tool of the German Aerospace Centre. For the generation of skin surfaces of the propeller blades, two libraries have been considered. For both libraries, Open Cascade Technology (open source) and BladeGenerator (internal software of the DLR) a model of a prototype was worked out, which pointed out advantages and disadvantages of each process. The implementation of the generation of propeller blade skin surfaces was considered as purposeful by using Open Cascade Technology. As a result a c++ software library is available, which is able to generate a propeller blade skin surface out of a given parametrization, which is discussed in this thesis. The implemented library provides functionalities to create files in formats STEP and IGES, which represent modeled skin surfaces and can be used for further processing in CAD-systems. A generation of propeller blades with or without sweep and lean is possible with the implemented library. Airfoils of a propeller blade can be stacked along the chord, the center of gravity or a sweep line, during the process of generating a propeller blade skin surface. For reconstructing sweep lines, several functions are provided. The precision of the generation of a propeller blade skin surface out of a given parametrization was proved by various operating tests. The propellers SRII and SRIII of the NASA were considered as use cases. By using the library it is now possible to generate propeller blade skin surfaces, which can be used in CFD-calculations afterwards, so that the aerodynamic fluid flow calculations in Propster can be validated with the results of those CFD-calculations.

Eidesstattliche Erklärung

gemäß § 5 (3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 29. September 2015.

Ich erkläre, dass ich die Bachelorarbeit ohne fremde Hilfe angefertigt habe und nur die im Literaturverzeichnis angeführten Quellen und Hilfsmittel benutzt habe. Entlehnungen aus fremden Quellen habe ich in jedem Fall als Zitate kenntlich gemacht.

Köln, den 8. September 2017

Inhaltsverzeichnis

Nomenklatur

Zeichen	Bedeutung	Einheit
A	Flächeninhalt	[m^2]
b	Sehnenlänge	[m]
CLD	Auftriebsbeiwert	[$-$]
$diff$	radialer Abstand zwischen zwei Punkten der Auffädellinie	[m]
D	Propellerdurchmesser	[m]
n	Anzahl von Punkten	[$-$]
p	Grad eines B-Splines	[$-$]
r	radiale Koordinate	[m]
r_{circ}	Radius eines Kreises	[m]
R	Propellerradius	[m]
t	Dicke eines Profils	[m]
u	Laufvariable eines B-Splines / einer B-Spline Fläche	[$-$]
v	Laufvariable einer B-Spline Fläche	[$-$]
β	Anstellwinkel	[$Grad$]
$\Delta\beta$	Blattverstellwinkel	[$Grad$]
Λ	Pfeilwinkel	[$Grad$]
φ	Mittelpunktwinkel eines Kreises	[$Grad$]

Abkürzungs- und Fremdwortverzeichnis

Indizes

Indize	Bedeutung
c	Skelettlinie
chord	Sehne
circ	Kreis
COG	Flächenschwerpunkt (center of gravity)
first	Erster
HK	Hinterkante
Hub	Nabe
last	Letzter
maxThick	Maximale Dicke
M	Mittelpunkt eines Kreises
rot	Rotiert
stack	Gefädelt
S	Schnittpunkt
Tip	Spitze eines Propellerblattes
VK	Vorderkante

Abkürzungen

Abkürzung	Bedeutung
B-Spline	**B**ézier **Spline**
CA	**C**ircular **A**rc
CAD	**C**omputer **A**ided **D**esign
CFD	**C**omputational **F**luid **D**ynamics
COG	**C**enter **o**f **G**ravity
CR	**C**ontractor **R**eport
CSV	**C**omma **S**eperated **V**alues
DLR	**D**eutsches Zentrum für **L**uft- und **R**aumfahrt
GUI	**G**raphical **U**ser **I**nterface
IGES	**I**nitial **G**raphics **E**xchange **S**pecification
InSim	**In**tegrierte **Sim**ulation von effizienten Antriebskonzepten
MTU	**M**otoren- und **T**urbinen-**U**nion
NACA	**N**ational **A**dvisory **C**ommittee for **A**eronautics
NASA	**N**ational **A**eronautics and **S**pace **A**dministration
NURBS	**N**on **U**niform **R**ational **B**ézier **S**pline
PCA	**P**itch **C**hange **A**xis
STEP	**St**andard for the **E**xchange of **P**roduct Model Data
TM	**T**echnical **M**anual
XML	E**x**tensible **M**arkup **L**anguage

Fremdworte

Fremdwort	Bedeutung
airfoil	Profil
bladeangle	Anstellwinkel
camber line	Skelettlinie
center of gravity	Geometrischer Schwerpunkt, Flächenschwerpunkt
chord	Sehne
circular arc	Kreisbogen
hub	Nabe
leading edge	Vorderkante
lean	Neigung eines Propellers
maximum thickness	Maximale Dicke
pitch change axis	Verstellachse des Anstellwinkels
pressure side	Druckseite eines Profils
stacked view	gefädelte Ansicht
suction side	Saugseite eines Profils
sweep (aerodynamic/ manufacture)	Pfeilung eines Propellers (aerodynamisch/ gefertigt)
sweepangle	Pfeilwinkel
tip	Spitze eines Propellerblattes
trailing edge	Hinterkante

1 Einleitung

Die Analyse von aktuellen und zukünftigen Triebwerkskonzepten ist ein wesentliches Arbeitsfeld am Institut für Antriebstechnik in der Abteilung Triebwerk des Deutschen Zentrums für Luft- und Raumfahrt. Im Rahmen des in Kooperation mit der MTU geführten Projekts InSim wurde das Software-Tool Propster entwickelt. Dieses stellt Datenstrukturen bereit, um die Geometrie eines Propellers zu parametrisieren. Der hauptsächliche Anwendungsfall dieses Tools besteht darin, die implementierten Funktionen zur Strömungsberechnung zu nutzen und die Ergebnisse zur Analyse von Triebwerken mit Propellern auszuwerten. Die Strömungsrechnungen, die durch Propster ausgeführt werden, erfolgen im zweidimensionalen entlang Schnittflächen einer Propellerschaufel, den sogenannten Profilen. Die Gesamtperformance des Propellerblattes wird anschließend durch eine Integration ermittelt. Um die Ergebnisse der Strömungsrechnung in Propster validieren zu können, sollen zukünftig dreidimensionale Strömungsrechnungen (CFD-Rechnungen) erfolgen, deren Ergebnisse mit denen der Strömungsrechnung aus Propster abgeglichen werden. Damit die CFD-Rechnungen ausgeführt werden können, ist es notwendig, die dreidimensionale Geometrie der Propellerschaufel, (Propellerblatt) bereitzustellen, deren aerodynamische Eigenschaften durch die Strömungsrechnung bestimmt werden sollen. Propster liefert mit seinen Datenstrukturen eine Parametrisierung für Propellerschaufeln. Das Software-Tool beinhaltet aber keine Funktionalität zur Generierung der dreidimensionalen Geometrie einer Propellerschaufel. Daraus ergibt sich die Aufgabenstellung dieser Bachelorarbeit. Es soll eine Bibliothek in Anlehnung an das Tool Propster implementiert werden, die die Parametrisierung von Propellerschaufeln nutzt, um aus dieser die Mantelfläche der Propellerschaufel zu generieren. Die Mantelfläche soll anschließend zur weiteren Verarbeitung in CAD-Systemen genutzt werden und unter anderem der dreidimensionalen Strömungsrechnung dienen. Im Rahmen dieser Ausarbeitung wird zunächst die erarbeitete Parametrisierung von Propellerschaufeln vorgestellt. Anschließend werden die Bibliotheken Open Cascade Technology (open source) und BladeGenerator (DLR-internes Softwaretool zur Optimierung von Schaufel-Geometrien) vorgestellt. Eine der beiden Bibliotheken soll ausgewählt werden, um mit deren Funktionalität, die Mantelfläche einer Propellerschaufel zu erzeugen. Die Auswahl einer der beiden Bibliotheken erfolgt nach einer Gegenüberstellung der Vor- und Nachteile des jeweiligen Verfahrens. Diese werden durch jeweils einen Prototyp aufgezeigt. Im Anschluss wird die Implementierung des ausgewählten Verfahrens beschrieben. Abschließend werden ausgeführte Anwendungstests betrachtet, die zur Validierung der implementierten Bibliothek

dienen. Als Anwendungsfall wurden die Propeller SRII und SRIII der NASA ausgewählt, deren Mantelflächen im Rahmen eines der durchgeführten Anwendungstests generiert wurden.

2 Geometrischer Aufbau einer Propellerschaufel

In diesem Kapitel wird beschrieben, wie die Geometrie einer Propellerschaufel im Rahmen dieser Arbeit betrachtet wird. Die NASA hat zum Ende der 70er Jahre damit begonnen, Studien zu Propellern durchzuführen. Diese beschäftigten sich mit Optimierungen der Propellerblatt-Geometrien, um effiziente aerodynamische Eigenschaften zu erreichen, die zu einem reduzierten Treibstoffverbrauch führen sollten. Weiterhin wurden Studien ausgeführt, die untersuchten, wie Lärmemissionen durch die Optimierung der Propellerblatt-Geometrie reduziert werden können [1]. Die Ergebnisse dieser wissenschaftlichen Arbeiten liefern eine vollständige Beschreibung von Propellerblatt-Geometrien in Form von einer definierten Parametrisierung. Diese Parametrisierung wurde als zielführend betrachtet, um ein Verfahren zur Generierung der Mantelflächen von Propellern zu entwickeln. Sie soll im weiteren Verlauf dieses Kapitels vorgestellt werden. Weiterhin sollen fundamentale Begriffe, die zur Beschreibung eines Propellerblattes notwendig sind, einführend erläutert werden. Die geometrische Information eines Propellerblattes liegt häufig in Form von zweidimensionalen axialen Schnitten durch die Propellerschaufel, die sogenannten Profile (Airfoils), vor. Deshalb werden zunächst die Profile betrachtet. Anschließend soll erläutert werden, wie diese Schnitte in radialer Richtung zueinander angeordnet werden. Bei dieser Anordnung spricht man auch vom Fädeln der Profile.

2.1 Profile einer Propellerschaufel

Profile sind die aus axialen Schnitten durch die Propellerschaufel resultierenden zweidimensionalen Geometrien, die in radialer Richtung zueinander angeordnet sind. Im Rahmen dieser Arbeit liegen die Profile in radialer Richtung parallel zueinander angeordnet vor (Parallelprofile). Ein Parallelprofil eines Propellerblattes befindet sich in der Ebene, deren radiale Koordinate r konstant ist. Häufig wird diese Koordinate auch prozentual zum Gesamtradius des Propellerblattes R angegeben. Man spricht im diesem Fall vom dimensionslosen Radius des Profils r/R. Abbildung 1 zeigt die Zeichnung einer Propellerschaufel. Zu sehen sind vier Schnitte durch diese Schaufel, sowie die daraus resultierenden Schnittflächen, die Profile.

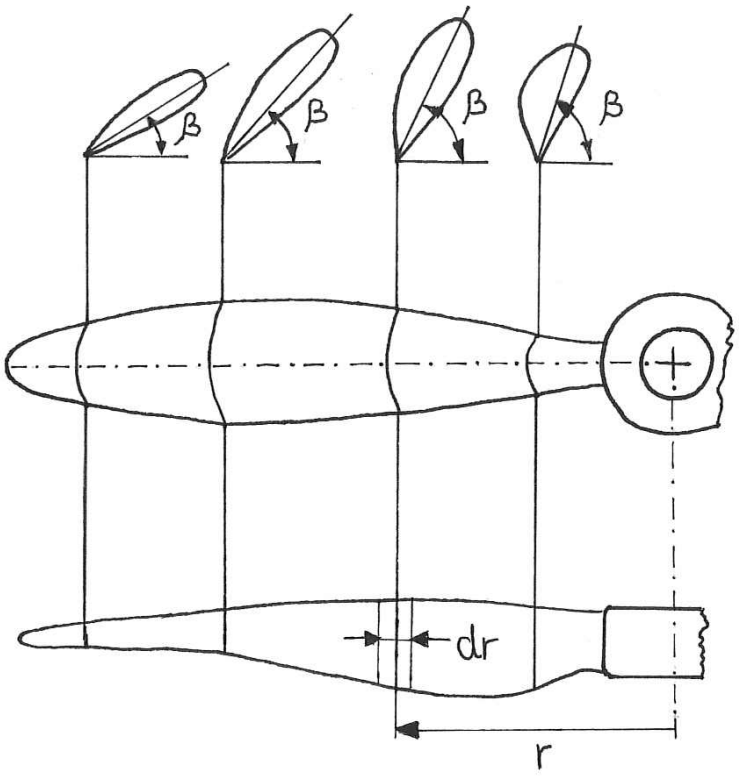


Abbildung 1: Profile einer Propellerschaufel [2]

Ein Profil wird in Zeichnungen in der Regel wie folgt angeordnet. Die Vorderkante (Leading Edge) befindet sich auf der linken Seite der Abbildung und die Hinterkante (Trailing Edge) entsprechend auf der rechten Seite. Die Saugseite (Suction Side) eines Profils befindet sich in der Abbildung oben und die Druckseite (Pressure Side) unten. Außerdem werden die Vorderkante und die Hinterkante durch eine imaginäre Gerade, die sogenannte Sehne (Chord Line), miteinander verbunden. Zusätzlich verbindet eine weitere gekrümmte Kurve die Vorder- und Hinterkante eines Profils. Sie wird als Skelettlinie (Camber Line) bezeichnet [3]. Die Skelettlinie gibt an, wie stark ein Profil gekrümmt ist. Die Druck- und Saugseite eines Profils lassen sich mit Hilfe der sogenannten Dickenverteilung rekonstruieren. Diese gibt an, wie weit Punkte auf der Druck- beziehungsweise Saugseite von der Sehne oder der Skelettlinie entfernt sind. Häufig liegt für die Dickenverteilung eine funktionale Abhängigkeit zwischen der Dicke und dem prozentualen Anteil der Sehnenlänge vor, die als weiteren Parameter die maximale Dicke (Maximum Thickness) eines Profils beinhaltet. Profile werden an der

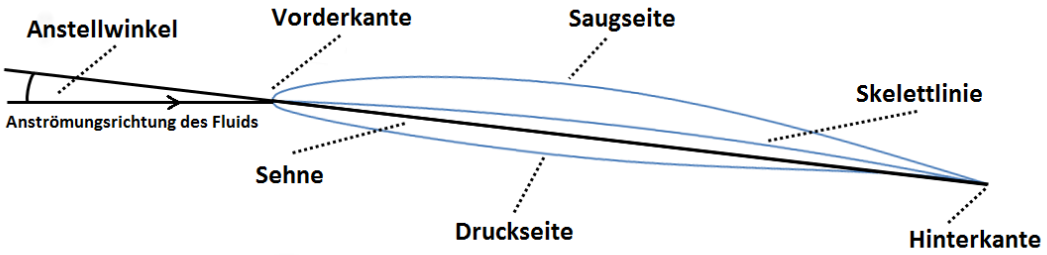


Abbildung 2: Darstellung zur Verdeutlichung fundamentaler Begriffe in Bezug auf ein Profil

Vorderkante von einem Fluid angeströmt. Der Anstellwinkel (Bladeangle) β eines Profils ist definiert als Winkel zwischen der Richtung aus der das Fluid strömt und der Sehne des Profils [4]. Die Abweichung der Anstellwinkel zweier Profile wird auch Blattverstellwinkel $\Delta\beta$ genannt. Häufig wird der Blattverstellwinkel dazu genutzt, die variierenden Anstellwinkel der Profile eines Propellers in einem funktionalen Zusammenhang mit dem dimensionslosen Radius r/R darzustellen. In Abbildung 1 sind die Anstellwinkel der einzelnen Profile eingezeichnet. Abbildung 2 veranschaulicht die Definition des Anstellwinkels und der weiteren erläuterten Begriffe noch einmal.

2.2 Profilparametrisierung

Nachdem im letzten Abschnitt die Profile eines Propellerblattes anhand der Parametrisierung beschrieben wurden, die im Rahmen dieser Arbeit verwendet wird, soll in diesem Abschnitt erläutert werden, wie durch die Kenntnis dieser Parameter die Geometrie eines Profils erzeugt werden kann. Die Beschreibung der Geometrie eines Profils soll von dem Tool Propster übernommen werden. Innerhalb dieses Tools werden Profile durch eine geordnete Anzahl an Punkten beschrieben. Diese Punkte reihen sich von der Hinterkante eines Profils entlang der Saugseite bis zur Vorderkante. Anschließend erfolgt die Anreihung der Punkte entlang der Druckseite zurück zur Hinterkante, sodass die Form eines Profils vollständig beschrieben wird. Im vorigen Abschnitt wurde die Dickenverteilung betrachtet. Soll die Geometrie eines nicht gekrümmten, symmetrischen Profils, welches über keine Skelettlinie verfügt, erzeugt werden, so entsprechen die Punkte der Dickenverteilung bereits der Saugseite eines Profils. Die Druckseite kann bei symmetrischen Profilen durch eine Invertierung der y-Koordinaten

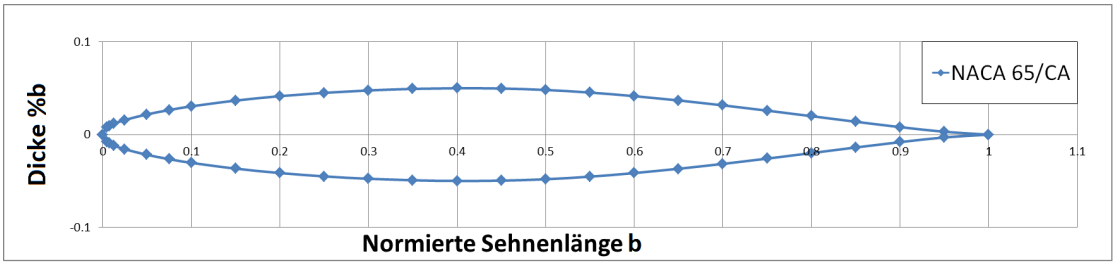


Abbildung 3: Beispielhafte Dickenverteilung für die Druck- und Saugseite eines nicht gekrümmten, symmetrischen Profils der NACA65/CA Serie

aller Punkte der Dickenverteilung erreicht werden. Im Rahmen dieser Arbeit werden lediglich symmetrische Profile betrachtet. Abbildung 3 zeigt die Dickenverteilung der Druck- und Saugseite eines Profils der Serie NACA65/CA in Abhängigkeit von der Sehnenlänge b.

Ist es erforderlich, die Geometrie eines gekrümmten Profils zu erzeugen, so kann die Skelettlinie und die Dickenverteilung genutzt werden. Ein Punkt, der auf der Druck- oder Saugseite eines gekrümmten Profils liegt, kann ermittelt werden, indem von einem Punkt der Skelettlinie $P_c(x_c|y_c)$ aus in Richtung des Normalenvektors im Punkt P_c die Strecke mit einer Länge der Dicke zurückgelegt wird. Die Dicke geht aus dem funktionalen Zusammenhang der Dickenverteilung für die Koordinate x_c hervor. Je nachdem ob ein Punkt der Druck- oder Saugseite erzeugt werden soll, muss die Strecke in Richtung des Normalenvektors in positiver oder negativer Richtung zurückgelegt werden. Abbildung 4 veranschaulicht diesen Zusammenhang noch einmal und zeigt ein gekrümmtes Profil der NACA65/CA Serie sowie dessen Sehne und Skelettlinie. Ein beispielhafter Punkt P_c befindet sich in der Abbildung an der Koordinate der normierten Sehnenlänge 0.2, dessen Normalenvektor ist durch eine Linie in beide Richtungen dargestellt.

2.3 Anordnung der Profile im dreidimensionalen Raum

Im Rahmen dieser Arbeit werden die Flächen der Profile einer Propellerschaufel in Ebenen angeordnet, die parallel zueinander an unterschiedlichen radialen Koordinaten liegen. Man spricht bei dieser Anordnung auch vom Fädeln. Beim Fädeln wird jedes einzelne Profil innerhalb seiner Ebene verschoben, sodass sich ein eindeutig zu bestimmender Punkt jedes Profils an demselben Punkt im zweidimensionalen Koordinatensystem befindet. Ein solcher Punkt könnte die Vorderkante, die Hinterkante, ein Punkt auf der Sehne oder der Flächenschwerpunkt des

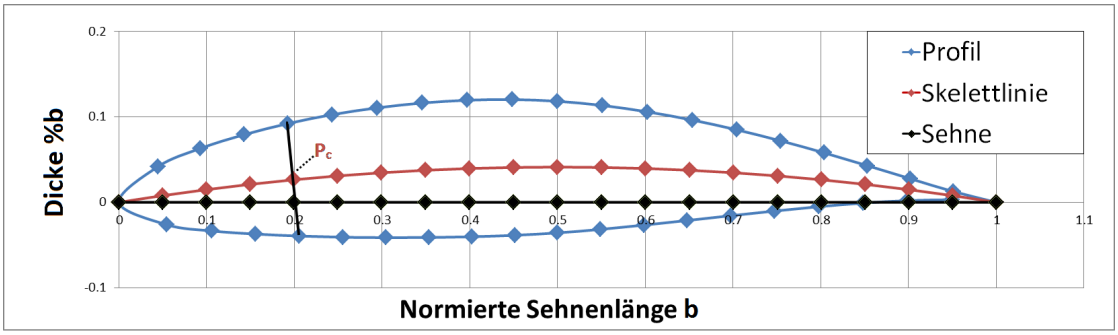


Abbildung 4: Schaubild zur Veranschaulichung der Erzeugung der Druck- und Saugseite eines gekrümmten, symmetrischen Profils

jeweiligen Profils sein. Eine Gerade durch diese Punkte liegt parallel zur radialen Achse im dreidimensionalen Koordinatensystem. Sie wird auch als Auffädelstrahl bezeichnet. Abbildung 5 zeigt die Zeichnung einer Verdichterschaufel, in der auch der Auffädelstrahl eingezeichnet ist.

Handelt es sich bei dem Propellerblatt um eines, welches weder eine Pfeilung noch eine Neigung aufweist, so ist die Anordnung der Profile im dreidimensionalen Raum abgeschlossen. Ein Propellerblatt weist eine Pfeilung (Sweep) auf, wenn dessen Vorderkante aus der Perspektive, bei der die Druckseite frontal betrachtet wird, abgerundet ist. Eine Neigung (Lean) des Propellerblattes liegt vor, wenn die Vorderkante des Propellers aus der Perspektive, bei der die Vorderkante frontal betrachtet wird, eine Rundung aufweist. Die Vorder- und Hinterkante, sowie die Druck- und Saugseite des Propellerblattes befinden sich entsprechend der Ausrichtung aller Profile. Die Pfeilung und Neigung des Propellerblattes wird erreicht, indem die sogenannte Auffädellinie definiert wird. Die Auffädellinie hat den gleichen Ursprung am untersten Profil des Propellerblattes wie der Auffädelstrahl. Allerdings verläuft diese nicht zwingend parallel zur radialen Achse im dreidimensionalen Koordinatensystem, sondern ist bei einer Pfeilung des Propellers rückwärts und bei einer Neigung des Propellers in Umfangsrichtung gebeugt [5]. Abbildung 6 zeigt eine Zeichnung des Propellers SR-3 aus der Perspektive, bei der die Druckseite des Propellers betrachtet wird. Die Pfeilung des Propellers und die Auffädellinie werden durch diese Abbildung ersichtlich. Abbildung 7 zeigt ein dreidimensionales Modell des SR-3 Propellers, aus der Perspektive, bei der die Vorderkante betrachtet wird. Aus dieser Perspektive wird die Neigung des Propellers deutlich.

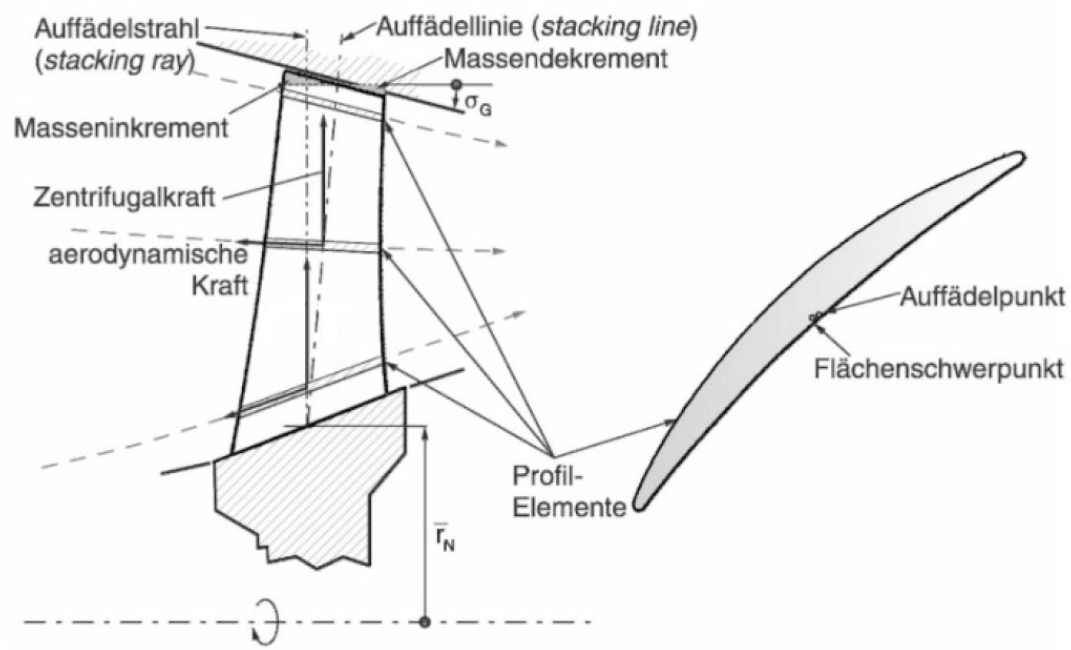


Abbildung 5: Zeichnung einer Verdichterschaufel zur Darstellung des Auffädelstrahls und der Auffädellinie [5]

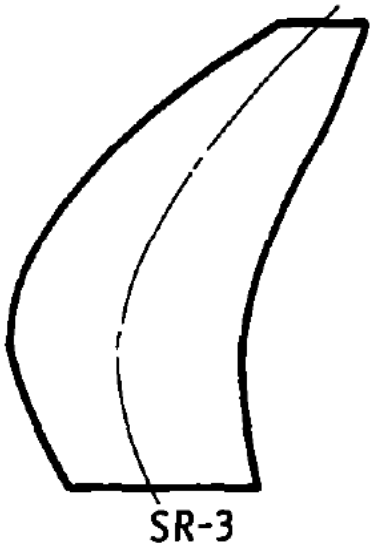


Abbildung 6: Zeichnung des SR-3 Propellers und dessen Auffädellinie [6]

Beschrieben wird die Pfeilung und Neigung des Propellers über den Pfeilwinkel (Sweepangle) Λ. Der Winkel wird in der Regel über eine funktionale Abhängigkeit in Bezug auf den dimensionslosen Radius r/R angegeben, sodass für jedes Profil ein Pfeilwinkel angegeben ist. Der Pfeilwinkel ergibt sich aus der Tangente, die die Auffädellinie am entsprechenden dimensionslosen Radius schneidet und aus der Achse zur Veränderung des Anstellwinkels des Propellers (Pitch Change Axis, PCA) [7]. Diese Achse dient dazu, die Anstellwinkel aller Profile im gleichen Maße während des Fluges zu beeinflussen, um günstige aerodynamische Verhältnisse zu schaffen und beispielsweise für mehr Auftrieb zu sorgen [5]. Abbildung 8 verdeutlicht die Definition des Pfeilwinkels noch einmal. Die Achse zur Veränderung des Anstellwinkels ist in der Abbildung die x-Achse. Die Auffädellinie wird in der Abbildung als Sweepline bezeichnet und ist durch eine Linie mit kurzen, voneinander abgesetzten Strichen dargestellt.

Nachdem die Profile in Abhängigkeit von der Auffädellinie im zweidimensionalen Raum verschoben wurden, ist die Anordnung abgeschlossen. Die Positionen aller Profile beschreiben nun das Blatt des Propellers unter Berücksichtigung der zuvor vorgestellten Parameter. Die Blätter eines Propellers werden bei Turboprop-Triebwerken über eine Nabe mit der Welle einer Turbine verbunden, welche für die Rotation sorgt. Der Abstand von der Rotationsachse

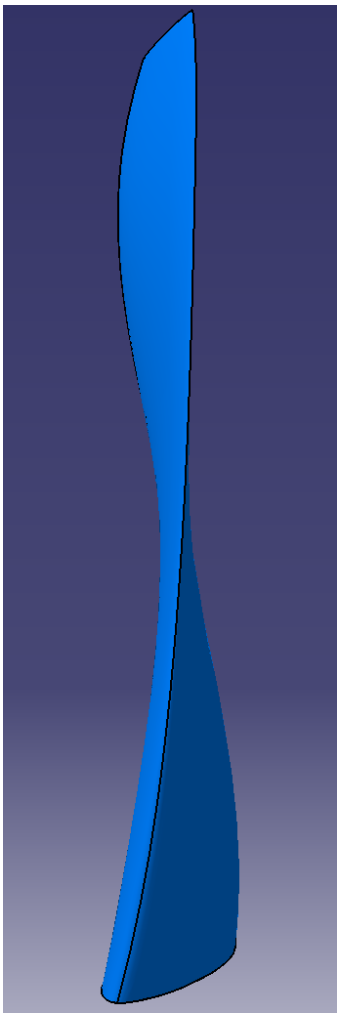


Abbildung 7: CAD-Modell des SR-3 Propellers aus der frontalen Perspektive

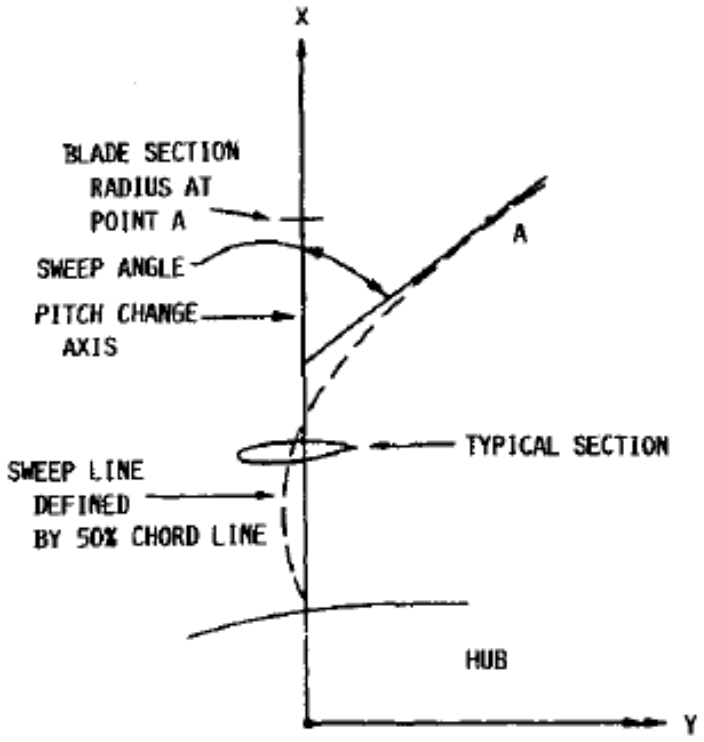


Abbildung 8: Schaubild zur Verdeutlichung des Pfeilwinkels [7]

der Welle in radialer Richtung zur Verbindung des Propellerblattes mit der Nabe (Hub) wird als Hubradius bezeichnet. Die Spitze des Propellerblattes wird auch Tip genannt. Die Bezeichnungen Tip und Hub werden im Folgenden häufiger genutzt.

3 IT-spezifischer Stand der Technik

3.1 CAD-Modellierungssoftware

CAD-Modellierungssoftware dient dazu, bei Praktiken des Computer Aided Designs (CAD) zu unterstützen. Unter Computer Aided Design sind alle Verfahren und Techniken zur Entwicklung und Konstruktion technischer Lösungen mit Hilfe eines Rechners zu verstehen. CAD-Systeme werden zum Design von zwei- und dreidimensionalen Modellen genutzt [8]. Bei einem Entwurf solcher Modelle werden durch entsprechende Parametrisierungen oder graphische Darstellungen Konstruktionen modelliert oder vorhandene Objekte verändert. Standardisierte Elemente und Datenstrukturen sowie die von CAD-Systemen zur Verfügung gestellten Operationen unterstützen in der Regel dabei, komplexere Strukturen mit geringerem Aufwand erstellen zu können. In dem folgenden Abschnitt sollen zwei Softwarebibliotheken vorgestellt werden. Diese stehen im Rahmen dieser Bachelorarbeit zur Auswahl, um bei der Erzeugung der Propellermantelflächen Einfluss zu nehmen, indem sie Funktionen und Datenstrukturen zur Modellierung der Propellerschaufelmantelflächen bereitstellen. Zunächst soll Open Cascade Technology vorgestellt werden.

3.1.1 Open Cascade Technology

Open Cascade Technology ist ein CAD-Software Development Kit von Open Cascade, welches in Softwareprojekte, die in der Programmiersprache C++ geschrieben sind, eingebunden werden kann. Open Cascade Technology steht unter der GNU Lesser General Public License. Zusammengefasst gewährt diese Lizenz jedem Nutzer die Freiheit, den Code der Software einsehen, nutzen und erweitern zu können. Sollte jedoch eine Weitergabe von veränderter Software an Dritte geschehen, so sind den Empfängern die gleichen Freiheiten (Einsehbarkeit, Nutzbarkeit, Erweiterbarkeit) des veränderten Codes zu gewähren [9]. Open Cascade Technology stellt Strukturen für grundlegende geometrische Objekte zur Verfügung. Zum Beispiel sind fundamentale Strukturen der CAD-Modellierung, wie Punkte, Geraden, Flächen oder Körper nutzbar. Auch komplexere Objekte, wie Bézier- oder NURBS-Kurven und -Flächen, werden bereitgestellt. Zusätzlich stehen für die Modellierung von zusammengesetzten Modellen topologische Strukturen zur Verfügung, die beispielsweise dazu dienen können, um aus mehreren Kurven ein Kantenmodell eines Körpers zu erzeugen. Außerdem liefert die Software geometrische und topologische Algorithmen, wie Bool'sche Operationen oder

Approximationen [10]. Für die Generierung von entsprechenden CAD-Outputformaten, wie zum Beispiel STEP oder IGES, stehen ebenfalls entsprechende Funktionen bereit [11]. Im Anschluss an diesen Abschnitt soll nun die zweite zur Auswahl stehende Software vorgestellt werden.

3.1.2 BladeGenerator

Der BladeGenerator ist ein DLR-internes Softwarewerkzeug zur Erzeugung von Schaufel-Geometrien, welches in der Sprache C++ im Institut für Antriebstechnik in der Abteilung Fan und Verdichter entwickelt wurde [12]. Um Schaufel-Geometrien zu erzeugen, benötigt das Werkzeug einige Inputparameter, die über entsprechend formatierte Dateien geliefert werden müssen. Der Hauptanwendungsfall des BladeGenerators besteht darin, eine Optimierung von Schaufel-Geometrien vorzunehmen, um bestimmte aerodynamische Eigenschaften dieser Schaufeln zu erreichen. Die Parametrisierung der Inputdaten ist deshalb auf die Optimierung ausgelegt. Der BladeGenerator liefert die Funktionalität, Profile einer Schaufel zu konstruieren und diese im dreidimensionalen Raum entlang der radialen Achse zu fädeln. Anschließend besteht die Möglichkeit, die Mantelfläche des entstandenen dreidimensionalen Modells mit einer B-Spline Tensorproduktfläche zu erzeugen. Die Mantelfläche kann danach in das Outputformat STEP oder TEC überführt werden [13]. TEC ist das Datenformat, welches von der Visualisierungssoftware Tecplot verwendet wird.

3.2 Schnittstelle zu den Geometrie-Informationen eines Propellers von Propster

Das Tool Propster beinhaltet neben den Verfahren zur Berechnung der zweidimensionalen Strömung entlang einer Propellerschaufel einige Klassen zur Strukturierung der dafür benötigten Daten. Daten, die die Geometrie der Schaufel eines Propellers beschreiben, sollen dazu genutzt werden, um die Propellermantelfläche zu generieren, die zukünftig dazu dienen soll, eine dreidimensionale Strömungsrechnung des entsprechenden Propellerblattes ausführen zu können. In dem folgenden Kapitel soll die Schnittstelle von Propster betrachtet werden, die es ermöglicht, an geometrische Informationen eines Propellers zu gelangen, die für die spätere Modellierung der Mantelfläche eines Propellerblattes benötigt werden. Die Klassenstrukturen des Tools Propster, die als Schnittstelle zu den Geometrischen Daten dienen, werden nach

dem Ansatz Top Down beschrieben. Die oberste Hierarchiestufe der Schnittstelle liegt durch die Klasse `Propeller` vor. Ein Objekt der Klasse `Propeller` beinhaltet als Membervariablen unter anderem den Namen des Propellers, die Anzahl der Propellerschaufeln und den Durchmesser. Außerdem beinhaltet ein Objekt der Klasse `Propeller` einen Zeiger auf ein Objekt der Klasse `Blade`. Die Klasse `Blade` repräsentiert eine Propellerschaufel. Im Kapitel 2 wurde erwähnt, dass das Blatt eines Propellers durch eine Menge an Profilen beschrieben wird. Diese Art der Beschreibung ist auch in der Klasse `Blade` berücksichtigt. Sie besitzt als Membervariable einen Vektor, der eine Anzahl an Zeigern auf Objekte der Klasse `Station` beinhaltet. Die Klasse `Station` repräsentiert eine Querschnittsfläche durch das Propellerblatt. Sie liefert durch Membervariablen wichtige geometrische Informationen, wie den dimensionslosen Radius, die Sehnenlänge sowie den Anstell- und Pfeilwinkel eines durch den Schnitt resultierenden Profils. Das Profil wird durch die Klasse `Airfoil` beschrieben. Die Klasse `Station` besitzt einen Zeiger als Membervariable, der auf ein Objekt der Klasse `Airfoil` verweist. Die Klasse `Airfoil` beinhaltet sowohl geometrische als auch aerodynamische Informationen des entsprechenden Profils. Die geometrischen Informationen sind in der Klasse `AirfoilGeometry` hinterlegt, auf die ein Zeiger verweist, der als Membervariable in der Klasse `Airfoil` hinterlegt ist. Die Geometrie des Profils liegt als Vektor mit einer geordneten Anzahl von Punkten vor. Abbildung 9 zeigt das Klassendiagramm, welches die in diesem Kapitel vorgestellte Klassenstruktur noch einmal darstellt [14] [15].

3.3 Propster Experiment Files

Im vorigen Abschnitt wurden die Datenstrukturen in Propster vorgestellt, die die geometrischen Informationen eines Propellers beinhalten. Im folgenden Abschnitt soll nun erläutert werden, wie diese Datenstrukturen mit Informationen gefüllt werden können. Zur Überführung der geometrischen Daten eines Propellers in die dafür vorgesehenen Datenstrukturen dienen die sogenannten Experiment Files. In einem Experiment File werden im XML-Datenformat die nötigen Informationen bereitgestellt. Wird ein solches Experiment File an Propster übergeben, so wird eine Output Datei zurückgeliefert. Diese wiederum kann jederzeit in Propster eingelesen werden, um die Datenstrukturen mit den geometrischen Informationen zu füllen. Die vorgesehene Strukturierung eines Experiment Files wird nun vorgestellt. Zuvor soll das XML-Datenformat noch einmal kurz betrachtet werden, um Grundbegriffe, die in diesem Abschnitt erwähnt werden, zu definieren. Innerhalb eines Experiment XML-Files werden drei

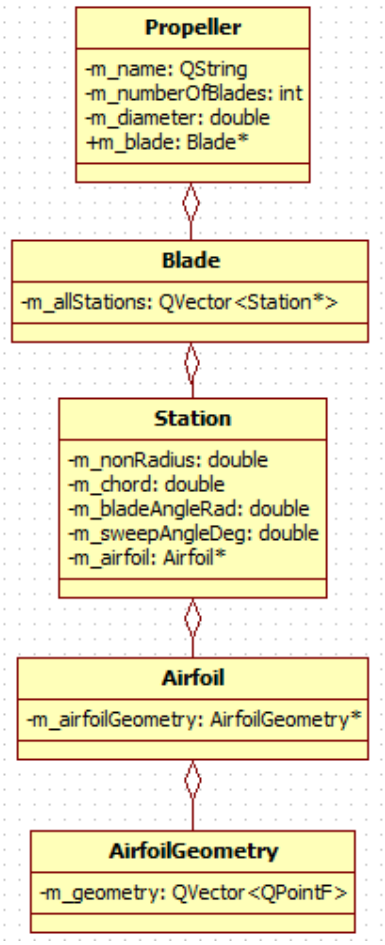


Abbildung 9: Klassenstruktur der Schnittstelle zur geometrischen Information eines Propellers in Propster

Strukturen genutzt, um die geometrischen Informationen strukturiert darzustellen. Dazu gehören Elemente, Attribute und Werte. Ein Element kann weitere Elemente, Attribute und einen Wert beinhalten. Attribute haben lediglich einen Wert. Werte können beispielsweise Zahlen oder Zeichenketten sein. Ein kurzer Abschnitt von XML-Code soll den Zusammenhang zwischen Elementen, Attributen und Werten verdeutlichen.

```
<Element1 attribut1= "Wert eines Attributes" attribut2= "2.222" >
    <Element2 attribut3= "Wert von Attribut 3" >Wert des Elements 2 </Element2>
    <Element3 attribut4= "Wert von Attribut 4" >Wert des Elements 3 </Element3>
</Element1>
```

In dem Code ist das Element `Element1` dargestellt, welches zwei Attribute (`attribut1` und `attribut2`) besitzt. `Element1` beinhaltet keinen Wert, aber zwei weitere Elemente (`Element2` und `Element3`). `Element2` besitzt das Attribut `attribut3` und einen Wert. `Element3` besitzt das Attribut `attribut4` und einen Wert. Auch jedes der vier Attribute hat jeweils einen Wert zugeordnet bekommen.

In einem Experiment File müssen zunächst zwei Pfade als Werte von zwei Elementen angegeben werden. Zum einen muss der Pfad, an dem die Output Datei erzeugt wird, übergeben werden und zum anderen muss ein Pfad für Input Dateien gesetzt werden. Welche Dateien als Input für das Experiment File dienen und sich in dem Inputverzeichnis befinden sollten, wird im weiteren Verlauf dieses Abschnitts erwähnt. Innerhalb des Elements `CreateObjects` in dem Experiment XML-File finden sich Elemente der Klassenstrukturen aus Propster wieder, welche die geometrischen Informationen beinhalten. Die Elemente innerhalb der Experiment Files sind wie die Klassenstrukturen in Propster benannt. So werden innerhalb des XML-Elements `CreateObjects` weitere Elemente `Airfoil`, `Station`, `Blade` und `Propeller` definiert. Diese werden innerhalb des Files nach dem Verfahren Bottom Up angegeben. Zunächst werden `Airfoil`-Elemente definiert, die ihren Namen als Attribut besitzen. Außerdem beinhalten diese ein Element, `Geometry File`, welches als Wert den Pfad zu einem solchen File besitzt. Dieses beschreibt die Geometrie eines Profils, indem es die Koordinaten aller Punkte, welche die Druck- und Saugseite eines Profils beschreiben, beinhaltet. Die Geometry Files aller `Airfoil`-Elemente sollten sich innerhalb des Input Verzeichnisses befinden, welches zuvor im Experiment File angegeben wurde. Ein `Blade`-Element besitzt neben seinem Namen

den dimensionslosen Hubradius als Attribut. Innerhalb eines `Blade`-Elements werden weitere `Station`-Elemente angegeben. Diese besitzen als Attribute den dimensionslosen Radius des Profils, welches sie beschreiben, und den Namen des `Airfoils`, welches die Geometrie dieses Profils beschreibt. Dieses `Airfoil` sollte zuvor in dem Experiment File als Element definiert worden sein. Zusammengefasst sollte sich innerhalb der Definition eines `Blade`-Elements eine feste Anzahl an `Station`-Elementen befinden, die gleich der Anzahl an zuvor festgelegten `Airfoil`-Elementen ist, sodass jedes `Airfoil`-Element genau einem `Station`-Element als Attribut zugeordnet werden kann. Ein `Propeller`-Element besitzt als Attribut den Namen des Propellers und die Anzahl an Profilen, die pro Propellerschaufel innerhalb der Datenstruktur generiert werden soll. Diese Anzahl muss nicht der Anzahl der im Experiment File definierten `Station` beziehungsweise `Airfoil`-Elemente entsprechen. Dies liegt daran, dass bei der Erzeugung der Output Datei zwischen den im Experiment File definierten Geometrien der Profile interpoliert wird, um die als Attribut des `Propeller`-Elements angegebene Anzahl an Profilen zu erreichen. Die durch die Interpolation erzeugten Profile liegen in gleichmäßigen radialen Abständen vor. Zusätzlich besitzt ein `Propeller`-Element einige weitere Elemente. Definiert werden muss ein Element für den Durchmesser, welches als Attribut die SI-Einheit und den Datentypen sowie den Wert des Durchmessers beinhaltet. Ein Element für die Anzahl der Schaufeln muss angegeben werden, welches als Attribut den Datentypen und als Wert die Anzahl der Schaufeln besitzt, sowie eines für die Schaufel selbst. In diesem Element wird als Wert der Name des Schaufel-Elements angegeben, welches zuvor im Experiment File definiert worden ist. Außerdem muss ein Element für die Nabe erzeugt werden sowie eines für die dreidimensionale Geometrie. Da die Nabe im Rahmen dieser Arbeit nicht betrachtet wird, sollen die Attribute auch nicht betrachtet werden. Das `Geometrie`-Element besitzt als Wert den Namen des Geometry Files, welches weitere Parameter des Propellers entlang der radialen Achse liefert. Dieses sollte sich ebenfalls im Verzeichnis des zuvor gesetzten Input Pfades befinden. Das Geometry File ist tabellarisch und im CSV-Datenformat aufgebaut. Es besitzt jeweils eine Spalte, in der die Sehnenlänge, der Anstellwinkel, und der Pfeilwinkel einer Propellerschaufel angegeben werden. Diese drei Parameter sind über den dimensionslosen Radius aufgetragen. Auch zwischen den im Geometry File angegebenen Werten wird bei der Erzeugung der Profile interpoliert, um die drei Parameter für den entsprechenden dimensionslosen Radius des jeweiligen Profils zu ermitteln. Nachdem mit den Elementen `Airfoil`, `Station`, `Blade` und `Propeller` alle geometrischen Informationen hinterlegt worden sind, muss in einem letzten Schritt im Experiment File innerhalb des Elements `Export` der Name der Output Datei festgelegt werden. Dazu wird innerhalb des

Elements `Export` ein Element mit dem Namen `XML` angelegt. Dieses wiederum beinhaltet ein `PropellerFile`-Element, welches als Attribute den Namen des zuvor definierten `Propeller`-Elements beinhaltet, sowie den Namen der Output Datei, welche im XML-Format erzeugt wird. Für die aerodynamische zweidimensionale Berechnung von Strömungen sind innerhalb eines Experiment Files weitere Elemente oder Attribute vorgesehen. Diese können auch innerhalb der in diesem Abschnitt beschriebenen Strukturen vorkommen. Da diese jedoch nicht von weiterer Bedeutung für diese Arbeit sind, wurde auf eine genauere Betrachtung verzichtet. Im Anhang A.1 dieser Arbeit befindet sich ein beispielhaftes Experiment File des SRII-Propellers.

3.4 Modultests mit Google Test

In diesem Abschnitt soll das Testing Framework Google Test vorgestellt werden, welches im Rahmen dieser Bachelorarbeit dazu genutzt wurde, um Unit Tests zu implementieren. Google Test ist ein plattformunabhängiges C++ Testing Framework zur Integration von Modultests, welches in der Abteilung Triebwerk des DLR genutzt wird. Modultests dienen dazu abgrenzbare Komponenten einer Software, wie zum Beispiel eine Klasse und deren Funktionen, zu testen. Das Testziel lautet, diese Komponenten auf deren funktionelle Korrektheit zu prüfen, um mögliche Fehler innerhalb der getesteten Funktionen frühzeitig innerhalb des Softwareentwicklungsprozesses auszumachen und zu beheben. Weiterhin dienen solche Tests dazu, die Robustheit des Codes in Bezug auf die Absturzsicherheit zu prüfen. Deshalb werden Modultests in der Regel bereits während der Implementierungsphase erstellt [16]. Ein Test von Google Test nutzt sogenannte Assertions, um zu beurteilen, ob bestimmte Bedingungen von den getesteten Komponenten erfüllt werden. Als Resultat einer Assertion sind drei mögliche Varianten zu nennen.

1. Erfolg (success)
2. Misserfolg (non-fatal failure)
3. Fataler Misserfolg (fatal failure)

Sind als Resultate aller Assertions innerhalb eines Tests lediglich Erfolge auszumachen, so ist auch der gesamte Test als erfolgreich festzuhalten. Sobald jedoch in Folge einer Assertion ein Misserfolg resultiert, schlägt der gesamte Test fehl. Dies gilt sowohl für Misserfolge, als auch für fatale Misserfolge. Der Unterschied zwischen einem Misserfolg und einem fatalen Misserfolg

besteht darin, dass ein fataler Misserfolg dazu führt, dass der Test umgehend abgebrochen wird, während ein Misserfolg einen möglichen weiteren Verlauf des Tests nicht beeinflusst. Assertions, welche mit Expect beginnen, führen zu einem Erfolg oder einem Misserfolg. Assertions, welche mit Assert beginnen, führen zu einem Erfolg oder einem fatalen Misserfolg. Es ist somit möglich, innerhalb eines Tests mittels der Wahl vom entsprechenden Assertions Abhängigkeiten beziehungsweise Unabhängigkeiten zu schaffen. Assertions vergleichen in der Regel zwei Werte des gleichen Datentyps miteinander. Dabei ergibt sich ein Wert aus der zu testenden Komponente der Software. Es könnte sich hier beispielsweise um einen Rückgabewert einer Funktion handeln. Dieser Wert wird abgeglichen mit einem Wert, den man im Rahmen dieses Anwendungsfalles als korrekten Rückgabewert erwarten würde. Diverse Assertions bieten die Möglichkeit solche Vergleiche beliebig zu gestalten. So können verschiedene Datentypen auf Gleichheit oder Ungleichheit geprüft werden. Vergleiche von Integer- und Gleitkommazahlen sind außerdem durch die üblichen Operatoren ($<$, $>$, $<=$, $>=$) möglich. Auf eine Darstellung weiterer Assertions wird an dieser Stelle verzichtet. Zur Strukturierung der einzelnen Tests bietet Google Test die Möglichkeit Test Cases zu definieren. Test Cases können mehrere Tests beinhalten. Die Test Cases sollten die Struktur des zu testenden Codes wiederspiegeln. So könnten beispielsweise mehrere Tests dieselbe Funktion abdecken. Diese könnten allesamt innerhalb eines Test Cases angeordnet werden, sodass eine Übersicht gewährleistet ist und Code, welcher noch nicht von Modultests abgedeckt ist, leichter erkannt werden kann. Test Cases können zu einem Testprogramm zugeordnet werden. Eine Unabhängigkeit einzelner Tests wird dadurch gewährleistet, dass ein fehlgeschlagener Test auf weitere Tests innerhalb und außerhalb eines Test Cases keine Auswirkungen hat. Ein fehlgeschlagener Test kann zusätzlich in Isolation wiederholt werden und das Debugging erleichtern. Häufig nutzen einige Tests gleiche Daten, welche zunächst erzeugt werden müssen, bevor diese in entsprechenden Tests berücksichtigt werden können. Damit die Generierung der Daten nicht innerhalb jedes Tests erneut in Form von Code auftritt, können Test Fixture Klassen definiert werden. Diese beinhalten neben den Tests eine `Setup` Funktion, in der die Generierung der Daten einmalig beschrieben werden kann. Vor dem Durchlaufen jedes Tests innerhalb der Test Fixture Klasse wird die `Setup` Funktion durchlaufen. Im Anschluss wird zusätzlich eine Funktion (`TearDown`) zum Bereinigen des Speichers ausgeführt. Mit Abschluss dieses Abschnittes kann festgehalten werden, dass Google Test mit den vorgestellten Funktionalitäten, die Möglichkeit bietet, Modultests zu schreiben, die die implementierten Funktionen, welche im Rahmen dieser Bachelorarbeit entstehen, auf deren korrekte Funktionalität prüfen [17] [18].

4 Auswahl eines Verfahrens zur Mantelgenerierung von Propellerschaufeln

Im Rahmen dieser Bachelorarbeit wurden die zuvor vorgestellten Softwares Open Cascade Technology und BladeGenerator auf ihre Eignung überprüft, die Mantelflächen der Propellerschaufeln in Form von B-Spline- beziehungsweise NURBS-Flächen zu generieren. Die Prüfung sollte durch jeweils einen Prototypen für beide Verfahren realisiert werden, die aufzeigen, welche Arbeitsschritte bei der Implementierung des entsprechenden Verfahrens zu tätigen sind und welche Vor- und Nachteile die Nutzung des jeweiligen Verfahrens hat. Im weiteren Verlauf dieses Kapitels sollen die Prototypen der beiden Verfahren vorgestellt werden, indem die zu tätigenden Iterationsschritte für die Mantelflächengenerierung erwähnt werden. Anschließend werden beide Verfahren miteinander verglichen und eines der beiden Verfahren unter Berücksichtigung der Vor- und Nachteile für die Implementierung der Propellermantelflächengenerierung ausgewählt. Zunächst soll der Prototyp für die Mantelgenerierung mit Hilfe des Software Development Kits Open Cascade Technology vorgestellt werden.

4.1 Prototyp zur Propellermantelflächengenerierung mit Open Cascade Technology

Bevor mit Hilfe der Bibliothek von Open Cascade Technology die Propellermantelfläche erzeugt werden kann, müssen die dafür notwendigen Daten bereitgestellt werden. Dazu dient die zuvor vorgestellte Schnittstelle zu den Geometrischen Daten eines Propellers von Propster. Ein Objekt der Klasse `Propeller` liefert alle geometrischen Daten, die für die vollständige Beschreibung der Mantelfläche eines Propellerblattes erforderlich sind. Neben einigen Parametern liefert das Objekt die geometrische Beschreibung von einer festen Anzahl an Profilen eines Propellerblattes in Form von geordneten Punktelisten, die die Druck- und Saugseite eines Profils definieren. Die Profile liegen jedoch normiert im zweidimensionalen Raum vor. Damit die Mantelfläche des Propellerblattes originalgetreu erzeugt werden kann, müssen die Daten durch Transformationen aufbereitet werden. Die Normierung der Profile zeichnet sich unter anderem dadurch aus, dass diese allesamt eine Sehnenlänge von einem

Meter aufweisen. Die Profile müssen somit in einem ersten Schritt der Aufbereitung auf ihre tatsächliche Sehnenlänge skaliert werden. Weiterhin zeichnet sich die Normierung der Profile dadurch aus, dass der Anstellwinkel der Profile nicht berücksichtigt wurde. Daraus ergibt sich, dass eine Rotation um den definierten Anstellwinkel des jeweiligen Profils erfolgen muss. Zusätzlich müssen die Punkte zur geometrischen Beschreibung der Profile vom zweidimensionalen in den dreidimensionalen Raum transformiert werden. In einem weiteren Schritt müssen die Profile im dreidimensionalen Raum gefädelt werden. Diese Iterationsschritte zur Aufbereitung der geometrischen Daten sind für die Mantelflächengenerierung eines Propellerblattes zu implementieren. Anschließend können die aufbereiteten Daten in Datenstrukturen von Open Cascade Technology überführt werden, bevor Funktionen der Bibliothek genutzt werden können, um eine Mantelfläche zu generieren. Die Mantelfläche wird in einem letzten Schritt in die CAD-Datenformate STEP und IGES exportiert, sodass eine Visualisierung mit entsprechenden CAD-Programmen erfolgen kann. Die Implementierung der beschriebenen Iterationsschritte zeigte auf, dass sich Open Cascade Technology für die Generierung der Mantelfläche von Propellerblättern eignet. Auf die Implementierung des Prototypen soll nicht weiter eingegangen werden, da das Verfahren zur Mantelgenerierung, welches im weiteren Verlauf dieses Kapitels für die Bewältigung der Aufgabenstellung dieser Arbeit ausgewählt wird, zu einem späteren Zeitpunkt beschrieben wird. Im folgenden Abschnitt soll der Prototyp zur Propellermantelflächengenerierung mit BladeGenerator vorgestellt werden.

4.2 Prototyp zur Propellermantelflächengenerierung mit BladeGenerator

Die Software BladeGenerator ist darauf ausgelegt Schaufel-Geometrien und deren Profile zu optimieren. Außerdem ist die Software in der Lage die optimierten Geometrien im Dreidimensionalen als Mantelflächen darstellen zu können. Die Parametrisierung, welche die Informationen für die Generierung einer Mantelfläche bereitstellt, ist an den Optimierungsprozess angepasst. Dadurch wird ermöglicht, dass durch die Veränderung weniger Parameter eine beachtliche Veränderung der Profil-Geometrie erfolgen kann. Die Parametrisierung eines Profils in BladeGenerator erfolgt über vier B-Spline Kurven dritten Grades. Die Kurven beschreiben die Druck- und Saugseite, sowie die Vorder- und Hinterkante des Profils. Der Verlauf von B-Spline Kurven ergibt sich neben dem Grad unter anderem auch aus der Position der Kontrollpunkte einer Kurve. Eine detailliertere Beschreibung von B-Splines ist im

zweiten Kapitel des Buches The NURBS Book zu finden [19]. Die Kontrollpunkte der Kurven der Druck und Saugseite, sowie die Steigungen am Start- und Endpunkt der Druck- und Saugseite sind als Parameter von jedem Profil an die Software zu übergeben, um diese zu definieren. Zur Definition der Kurven der Vorder- und Hinterkante eines Profils sind weitere Parameter zu übergeben. Bei diesen Parametern handelt es sich um Abstände zwischen bestimmten Punkten [13]. Auf eine detaillierte Beschreibung der Parameter soll an dieser Stelle verzichtet werden. Die Parametrisierung wird bei der Auswahl eines der beiden Verfahren im folgenden Abschnitt noch einmal angesprochen. Um die Parameter zur Definition der einzelnen Profile bereitstellen zu können, müssen die normierten Profil-Geometrien, welche über die Schnittstelle von Propster in einem Propeller Objekt übergeben werden, zunächst analog zum ersten Iterationsschritt des Prototyps von Open Cascade Technology auf deren Sehnenlänge skaliert werden. Anschließend muss festgelegt werden, an welcher Stelle des Profils die Vorder- und Hinterkante, sowie die Druck- und Saugseite beginnt und endet, damit eine B-Spline Approximation der Druck- und Saugseite, sowie der Vorder- und Hinterkante erfolgen kann. Die zu übergebenden Parameter stehen in Folge der Approximation bereit oder müssen durch Berechnungsverfahren, die im Rahmen dieser Arbeit zu implementieren wären, ermittelt werden. Die Approximation der B-Splines wird durch eine bereits vorhandene, DLR-interne Bibliothek ausgeführt. Programmintern in der Software BladeGenerator werden die vier B-Spline Kurven krümmungsstetig ineinander überführt. Als Zwischenergebnis liegt dann eine stetige Kurve vor, die das entsprechende Profil beschreibt. Als weitere Parameter müssen dem BladeGenerator die Anstellwinkel der Profile übergeben werden, damit diese bei der Generierung der Mantelfläche des Propellerblattes berücksichtigt werden. Außerdem muss der Software die Information übergeben werden, wie die zweidimensionalen Profile entlang der radialen Achse im dreidimensionalen Raum gefädelt werden sollen. Diese Iterationsschritte werden von BladeGenerator automatisiert ausgeführt, sodass keine Implementierung von Funktionen, die diese Aufgaben übernehmen würden, erforderlich ist. Nachdem die Profile im dreidimensionalen Raum angeordnet wurden, kann aus dieser Anordnung programmintern eine Mantelfläche (Tensorproduktfläche) generiert werden. Zur Visualisierung der Mantelfläche stehen die Ausgabeformate STEP und TEC zur Verfügung, welche von BladeGenerator generiert werden können [13]. Nachdem von beiden Verfahren zur Generierung von Mantelflächen einer Propellerschaufel die Prototypen vorgestellt wurden, sollen die in der Implementierungsphase der Prototypen gewonnenen Erkenntnisse genutzt werden, um die beiden Verfahren zu vergleichen sowie Vor- und Nachteile der Verfahren gegenüberzustellen. Eines der Verfahren wird anschließend ausgewählt, um in der im Rahmen dieser Arbeit zu implementierenden

Bibliothek dazu zu dienen, die Mantelflächen von Propellerblatt-Geometrien zu erzeugen.

4.3 Auswahl eines Verfahrens

Bei der Auswahl des Verfahrens zur Mantelflächengenerierung wurden mehrere Faktoren berücksichtigt, die nun betrachtet werden sollen. Ein wesentlicher Faktor bei der Auswahl des Verfahrens sind die Parameter, die als Input an die Bibliothek zu liefern sind und von dieser zur Generierung der Mantelfläche eines Propellerblattes genutzt werden. Diese Inputparameter sind aus der im Kapitel 2 vorgestellten Parametrisierung, die im Rahmen dieser Arbeit genutzt und durch die Schnittstelle von Propster bereitgestellt wird, zu übernehmen oder mit Hilfe dieser zu errechnen. Bei der Implementierung des Prototypen zur Propellermantelflächengenerierung mit Open Cascade Technology konnten alle notwendigen Parameter, die für die Aufbereitung der geometrischen Information benötigt werden, um aus dieser die Mantelfläche zu generieren, aus der Schnittstelle zu Propster direkt entnommen oder mit Hilfe der gelieferten Daten errechnet werden. Bei der Aufbereitung der Daten, die zur Erzeugung der Mantelfläche mit Open Cascade Technology genutzt werden, ist es erforderlich die normierten, von Propster gelieferten geometrischen Informationen eines Propellerblattes in die Originalgröße und Form zu transformieren. Die für die Transformation genutzten Parameter sind beispielsweise die Sehnenlänge, der Anstell- und Pfeilwinkel und der dimensionslose Radius jedes Profils, sowie der Radius des gesamten Propellerblattes. Die Parameter, welche an BladeGenerator übergeben werden und die geometrische Information des Propellerblattmantels beinhalten, sind im Vergleich zu dem Verfahren mit Open Cascade Technology mit größerem Aufwand zu ermitteln. Unter anderem muss eine Approximation der Profil-Geometrien erfolgen. Durch die Approximation sind geringfügige Abweichungen von den Original-Geometrien nicht zu vermeiden. Auch die spätere Überführung der vier B-Spline Kurven in eine Kurve birgt Potential für weitere Abweichungen. Zu erwähnen ist an dieser Stelle noch einmal, dass BladeGenerator das Ziel verfolgt, Profil-Geometrien zu optimieren. Im Rahmen dieser Arbeit sollen jedoch bereits vorhandene Geometrien von Propellerschaufeln exakt nachmodelliert werden. Eine Optimierung soll nicht stattfinden. In dem Anwendungsfall der Optimierung werden Profil-Geometrien verändert, sodass eine exakte Darstellung der ursprünglichen Geometrien nicht als primäres Ziel in BladeGenerator verfolgt wird und die Software somit nicht direkt für den Anwendungsfall dieser Bachelorarbeit konzipiert ist. Gleichzeitig soll aber auch betont werden, dass die Generierung der Mantelflächen von Propellerblättern trotzdem

möglich ist. BladeGenerator benötigt zur Erstellung der Kurven, welche die Vorder- und Hinterkante der Profile definieren, weitere Parameter. Die exakte Ermittlung dieser Parameter aus den vorhandenen geometrischen Informationen stellte sich als schwierig heraus. Ohne eine exakte Ermittlung dieser Parameter ist es jedoch nicht möglich, die Geometrie des Propellers in eine originalgetreue Mantelfläche zu überführen. Eine Mantelfläche, die nicht dem Original entspricht, obwohl dieses durch die von Propster gelieferte Parametrisierung vollständig definiert wird, wäre das Ergebnis. Open Cascade Technology weist in Bezug auf die Bereitstellung der Parameter somit weniger Potential für Fehler und Abweichungen auf. Als weiterer Faktor, der auf die Auswahl der Bibliothek zur Mantelflächengenerierung Einfluss genommen hat, ist der Implementierungsaufwand zu nennen. Nachdem die Inputparameter an die Software BladeGenerator geliefert wurden, sind dieser nur wenige weitere Informationen zu übermitteln, damit die Mantelfläche generiert werden kann. Dazu gehören beispielsweise die Anstell- und Pfeilwinkel der Profile und die Weise, wie die zweidimensionalen Profile entlang der radialen Achse im dreidimensionalen Raum gefädelt werden sollen. Die Anordnung der Profile geschieht anschließend automatisiert entsprechend der Parameterübergabe. Zusätzlich wird auch die Mantelfläche der Propellerschaufel erzeugt, deren Geometrie in Form von Dateien in den CAD-Formaten STEP und TEC beschrieben wird. Die Dateien werden automatisiert generiert. Der Implementierungsaufwand bei der Nutzung von BladeGenerator ist somit gering, da die Funktionalität zur automatisierten Mantelflächengenerierung bereits gegeben ist. Im Vergleich dazu ist die Nutzung der Bibliothek Open Cascade Technology mit mehr Implementierungsaufwand verbunden. Die Bibliothek ist zwar darauf ausgelegt CAD betreiben zu können, jedoch liefert diese nicht wie der BladeGenerator die Funktionalität, automatisiert eine Schaufel-Geometrie zu erzeugen. Somit ist bereits die Aufbereitung der normierten Geometrie-Informationen, die von der Schnittstelle von Propster bereitgestellt werden, mit mehr Aufwand verbunden. Im Vergleich zur Nutzung von BladeGenerator müssen die normierten Profil-Geometrien nicht nur auf die Sehnenlänge skaliert werden, sondern auch um den Anstellwinkel rotiert werden. Auch die Funktionalität zum Fädeln der Profile entlang der radialen Achse müsste implementiert werden. In dem Kapitel 2.3 wurden einige Möglichkeiten aufgezeigt, wie Profile einer Propellerschaufel gefädelt werden können. Um die Mantelgenerierung verschiedenster Propellerblatt-Geometrien zu ermöglichen, müsste die Routine, die die Funktionalität des Fädelns der Profile bereitstellt, flexibel gestaltet werden. Sie müsste die verschiedenen Varianten des Fädelns berücksichtigen und eine Erweiterung um nicht berücksichtigte Arten des Fädelns ermöglichen. Nachdem die Profile rotiert, skaliert und gefädelt wurden, können die aufbereiteten geometrischen Informationen an Datenstrukturen

von Open Cascade Technology übergeben werden. Die geometrischen Informationen, die als geordnete Punktelisten vorliegen und jeweils ein Profil beschreiben, werden anschließend in eine Mantelfläche übertragen. Die Funktionalität zur Adaption der Daten in ein kompatibles Format von Open Cascade Technology und die anschließende Erzeugung einer Mantelfläche muss bei einer Auswahl des Verfahrens implementiert werden. Auch Funktionen zur Generierung von CAD-Output Dateien, welche es ermöglichen, den erstellten Propellerblattmantel zu visualisieren, müssten eigenständig bereitgestellt werden. Open Cascade Technology stellt jedoch für diesen Anwendungsfall Klassenstrukturen zur Verfügung, die es ermöglichen, aus einer Geometrie im Open Cascade Technology Format, wie zum Beispiel einer Fläche, Dateien im STEP- oder IGES-Format zu erzeugen, welche die Geometrien beschreiben und zur Visualisierung dieser genutzt werden können.
Zusammengefasst kann festgehalten werden, dass der Aufwand zur Bereitstellung der Inputparameter unter Verwendung der über die Schnittstelle von Propster gelieferten normierten, geometrischen Informationen bei der Nutzung von Open Cascade Technology geringer ist als bei BladeGenerator. Außerdem ist das Potential für Fehler und Ungenauigkeiten bei der Nutzung von Open Cascade Technology geringer. Auch bei der Nutzung von Open Cascade Technology werden die geometrischen Daten bei der Generierung der Mantelfläche durch eine Approximation geringfügige Abweichungen in Bezug auf die Original-Geometrie aufweisen. Jedoch ist dieses Potential bei der Nutzung von BladeGenerator größer, da nicht nur die vier B-Splines zur Beschreibung der Druck- und Saugseite sowie der Vorder- und Hinterkante approximiert werden, sondern auch eine Überführung der vier Kurven in einen B-Spline mit möglichen Abweichungen verbunden ist. Zuletzt können weitere Abweichungen durch die Erzeugung der Mantelfläche aus den B-Splines entstehen. Die Beschaffung der Parameter zur Beschreibung der Vorder- und Hinterkante eines Profils stellte sich bei dem Verfahren mit BladeGenerator als schwierig dar und führte ebenfalls zu Abweichungen der resultierenden Profil-Geometrien in Bezug auf die Original-Geometrie. Der Implementierungsaufwand bei der Verwendung von Open Cascade Technology übersteigt den Aufwand bei der Verwendung von BladeGenerator deutlich, da BladeGenerator die Funktionalität besitzt, aus einer gegebenen Parametrisierung die Mantelfläche einer Schaufel-Geometrie zu erzeugen und in ein geeignetes CAD-Outputformat zu konvertieren. Bei der Verwendung von Open Cascade Technology steht ein mächtiges CAD-Werkzeug zur Verfügung, welches alle Möglichkeiten liefert die Mantelfläche einer Schaufel-Geometrie zu modellieren. Allerdings müssen die gelieferten normierten, geometrischen Daten vollständig durch implementierte Funktionen transformiert werden, sodass diese die Schaufel in Originalform und Originalgröße beschreiben. Auch für

die Konvertierung der Daten in ein geeignetes Format von Open Cascade Technology, sowie für die anschließende Erzeugung der Mantelfläche aus diesen Daten, sind Funktionen zu implementieren. Zuletzt ist es erforderlich, die Funktionalität zur Generierung von Dateien in CAD-Outputformaten bereitzustellen, welche die erzeugte Mantelfläche eines Propellerblattes beschreibt.
Da das primäre Ziel dieser Arbeit lautet, die Mantelfläche einer Propellerschaufel möglichst exakt abzubilden, wurde die Entscheidung getroffen, Open Cascade Technology für die Generierung der Mantelflächen zu nutzen. Die resultierenden Mantelflächen sollen dazu genutzt werden, dreidimensionale Strömungsrechnungen (CFD-Rechnungen) auszuführen. Die damit gewonnenen Ergebnisse sollen wiederum der Validierung der bereits in Propster implementierten zweidimensionalen Strömungsrechnung dienen. Eine exakte Modellierung der Original-Geometrie ist von großer Bedeutung. Abweichungen von der Original-Geometrie würden eine Validierung der zweidimensionalen Strömungsrechnung erschweren und zu verfälschten Ergebnissen führen. Mit der Auswahl von Open Cascade Technology kann das Gesamtziel der Arbeit somit besser erreicht werden.

5 Implementierung des Verfahrens zur Mantelgenerierung von Propellerschaufeln

Im vorangegangenen Kapitel wurde mit Open Cascade Technology die Software zur Generierung der Mantelfläche ausgewählt. Es wurde aufgezeigt, welche Iterationsschritte zur Aufbereitung der geometrischen Daten, die von der Schnittstelle von Propster geliefert werden, zu implementieren sind. In dem folgenden Kapitel sollen diese Schritte genauer betrachtet werden. Zunächst sollen die Algorithmen des implementierten Verfahrens zur Propellermantelflächengenerierung vorgestellt werden. Anschließend soll die IT-spezifische Umsetzung beschrieben werden, indem die implementierte Klassenstruktur und die Schnittstellen des Verfahrens vorgestellt werden.

5.1 Algorithmen des implementierten Verfahrens

In dem folgenden Abschnitt sollen die implementierten Algorithmen der einzelnen Iterationsschritte, die für die Erzeugung einer Propellermantelfläche aus den gelieferten Daten der Schnittstelle von Propster erforderlich sind, betrachtet werden. In diesem Kapitel werden häufig die Profile einer Propellerschaufel erwähnt. Die Geometrie eines Profils wird jeweils durch eine geordnete Punkteliste beschrieben. Wird in diesem Kapitel die Geometrie eines Profils mittels einer Transformation verändert, so findet diese Transformation für jeden Punkt innerhalb dieser Liste statt. Dies ist im Verlauf dieses Kapitels zu berücksichtigen. In dem Abschnitt 3.2 wurde beschrieben, wie die geometrischen Daten bereitgestellt werden. Damit die Mantelfläche eines Propellers mit Hilfe von Open Cascade Technology erzeugt werden kann, ist es zunächst erforderlich, die normierten Profil-Geometrien aufzubereiten. Danach werden die aufbereiteten geometrischen Informationen in ein geeignetes Format von Open Cascade Technology überführt, sodass aus diesem Format anschließend eine Mantelfläche generiert werden kann. Die folgenden Abschnitte dieser Arbeit befassen sich mit den implementierten Algorithmen, die die Geometrien aufbereiten und anschließend in eine Mantelfläche überführen.

5.1.1 Aufbereitung der normierten geometrischen Daten im Zweidimensionalen

Die über die Schnittstelle von Propster gelieferten normierten Profil-Geometrien, die sich im zweidimensionalen Raum befinden, müssen in einem ersten Iterationsschritt auf die Sehnenlänge skaliert werden. Da die Sehnenlänge bei den normierten Profilen immer einen Meter beträgt, ergibt sich der Skalierungsfaktor aus der tatsächlichen Sehnenlänge des Profils in Metern. Die x- und y-Koordinaten der Punkte, welche die Profil-Geometrie beschrieben, sind mit dem Faktor zu multiplizieren. Anschließend liegen die Profile skaliert in Originalgröße vor. Die skalierten Profile müssen im folgenden Iterationsschritt um den Anstellwinkel rotiert werden. Die Rotation erfolgt im Uhrzeigersinn für jeden Punkt, welcher die Geometrie eines Profils beschreibt. Die Koordinaten eines um den Anstellwinkel β rotierten Punktes $P_{rot}(x_{rot}|y_{rot})$ ergeben sich aus der Multiplikation mit der Rotationsmatrix aus Gleichung 1.

$$\begin{pmatrix} x & y \end{pmatrix} * \begin{pmatrix} \cos(\beta) & -\sin(\beta) \\ \sin(\beta) & \cos(\beta) \end{pmatrix} = \begin{pmatrix} x_{rot} & y_{rot} \end{pmatrix} \tag{1}$$

Nachdem die Profile um den Anstellwinkel rotiert wurden, können sie im dreidimensionalen Raum entlang der radialen Achse gefädelt werden. Im folgenden Abschnitt sollen die im Rahmen dieser Arbeit implementierten Verfahren zum Fädeln von Profilen vorgestellt werden.

5.1.2 Fädeln der Profile

Für das Fädeln von Profilen einer Propellerschaufel gibt es verschiedene Möglichkeiten. Im Rahmen dieser Arbeit wurden die verbreitetsten Verfahren zum Fädeln implementiert. Bevor die entstandenen Algorithmen vorgestellt werden, sollen zunächst noch Allgemeingültigkeiten beim Fädeln angesprochen werden. Die Profile, die im Abschnitt zuvor skaliert und rotiert wurden, befinden sich im zweidimensionalen Raum und sollen nun im dreidimensionalen Raum entlang der radialen Achse gefädelt werden. Die Profile und deren Punkte müssen deshalb aus dem zweidimensionalen in den dreidimensionalen Raum transformiert werden, indem diesen jeweils eine radiale Koordinate zugeordnet wird. Die radiale Koordinate jedes Punktes, welche die Geometrie desselben Profils beschreibt, ist identisch. Die radiale Koordinate r ergibt sich aus dem dimensionslosen Radius r/R des Profils multipliziert mit dem Radius R des Propellerblattes. Das Profil befindet sich somit in der xy-Ebene an der radialen Koordinate

r. Beim Fädeln wird jedes Profil nun innerhalb dieser Ebene verschoben, sodass sich die gewünschte Form des Propellerblattes ergibt. Bei gepfeilten und geneigten Propellerblättern dient die radial verlaufende Auffädellinie dazu, zu definieren, wie die Profile zu verschieben sind. In den folgenden Abschnitten werden die verschiedenen Varianten des Fädelns beschrieben. Zunächst soll das Fädeln über einen Punkt auf der Sehne prozentual zur Sehnenlänge betrachtet werden.

5.1.2.1 Fädeln über die Sehne

Bei dem Fädeln über die Sehne wird die Propellerschaufel weder gepfeilt noch geneigt, sodass keine Auffädellinie benötigt wird. Als Parameter ist anzugeben, über welchen prozentualen Anteil p_{chord} der Sehnenlänge b der einzelnen Profile gefädelt werden soll. Von jedem Profil wird dann der Punkt $P_{chord}(x_{chord}|y_{chord})$ bestimmt, der sich auf der Sehne befindet und dessen Abstand zur Vorderkante des Profils dem definierten prozentualen Anteil der Sehnenlänge entspricht. Die Geometrie der Profile wird durch eine geordnete Punkteliste beschrieben, deren erster Punkt $P_{first}(x_{first}|y_{first})$ immer die Hinterkante des Profils ist. Die Vorderkanten der normierten Profile, die von der Schnittstelle von Propster geliefert werden, liegen allesamt im Ursprung. Die zuvor ausgeführte Skalierung und Rotation ändert an dieser Gegebenheit nichts. Mit der Erkenntnis, dass sich die Vorderkante jedes Profils im Ursprung befindet und die Koordinaten der Hinterkante im ersten Punkt der Punktelisten, die die Profil-Geometrien beschreiben, hinterlegt sind, kann der gesuchte Punkt P_{chord} bestimmt werden (siehe Gleichung 2).

$$x/y_{chord} = x/y_{first} * p_{chord} \tag{2}$$

Nachdem der Punkt P_{chord} jedes Profils ermittelt wurde, sollten die Profile so angeordnet werden, dass sich dieser Punkt jedes Profils an den gleichen x- und y-Koordinaten befindet, um das Fädeln auszuführen. In dem implementierten Algorithmus wurden die Profile derart verschoben, dass sich der Punkt P_{chord} jedes Profils im Anschluss an die Verschiebung im Ursprung befindet. Die Verschiebung jedes Punktes, der die Geometrie eines Profils beschreibt, erfolgt in die Richtung des Vektors $\vec{V}\begin{pmatrix} -x_{chord} \\ -y_{chord} \end{pmatrix}$. Abbildung 10 veranschaulicht den Vorgang des Fädelns über die Sehne noch einmal anhand eines Profils.

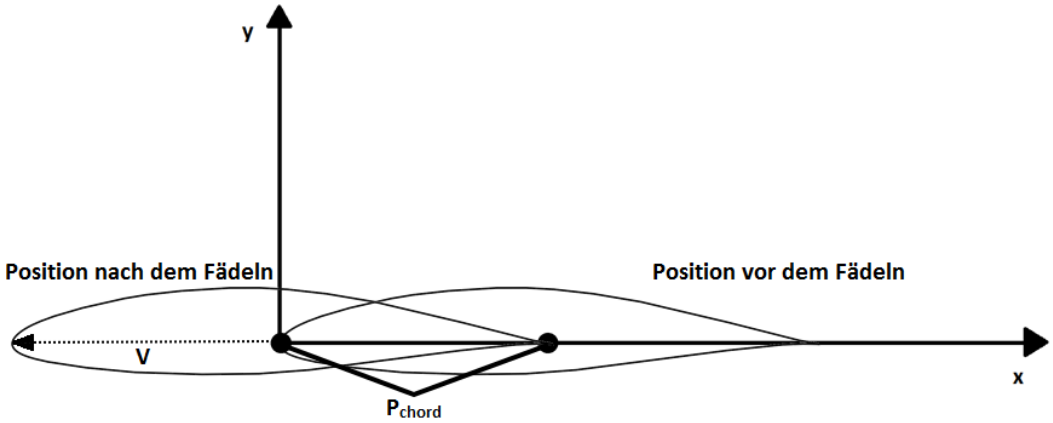


Abbildung 10: Schaubild zum Vorgang des Fädelns über die Sehne

Im folgenden Abschnitt soll das Fädeln über den Flächenschwerpunkt genauer betrachtet werden.

5.1.2.2 Fädeln über den Flächenschwerpunkt

Bei dem Fädeln über den Flächenschwerpunkt wird die Propellerschaufel, wie auch bei dem Fädeln über die Sehne, weder gepfeilt noch geneigt. Der Vorgang des Fädelns über den Flächenschwerpunkt ist dem Fädeln über die Sehne sehr ähnlich. Der einzige Unterschied besteht darin, dass nicht über einen Punkt auf der Sehne P_{chord} gefädelt wird, sondern über den Flächenschwerpunkt (geometrischer Schwerpunkt einer Fläche), der von jedem Profil errechnet werden muss. Vereinfacht wird dazu angenommen, dass es sich bei den Profilen um geschlossene Polygone handelt. Polygone sind Vielecke, deren Eckpunkte durch einen geschlossenen Kantenzug (Polygonzug) miteinander verbunden sind. Ein Kantenzug ist definiert als eine Kette von Liniensegmenten, die wiederum als Verbindungsgerade zwischen zwei Punkten beschrieben werden kann [20] [21]. Da die geometrische Information eines Profils durch eine geordnete Punkteliste vorliegt, deren Punkte von der Hinterkante entlang der Saugseite in Richtung Vorderkante verlaufen und von dort aus über die Druckseite zurück in Richtung Hinterkante verlaufen, liegt prinzipiell ein Polygon vor. Abbildung 11 zeigt einen beispielhaften Polygonzug eines Profils.

Ein geschlossenes Polygon zeichnet sich dadurch aus, dass der erste Punkt des Polygonzuges

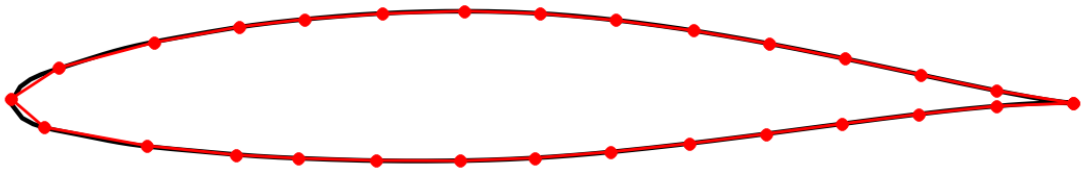


Abbildung 11: Polygonzug eines Profils

gleich dem letzten Punkt innerhalb des Polygonzuges ist. Liegt ein geschlossenes Polygon, welches durch n Punkte definiert wird, vor, so können die Koordinaten des Flächenschwerpunktes über die Trapezformel aus Gleichung 3 und 4 errechnet werden [22].

$$x_{COG} = \frac{1}{6A} \sum_{i=0}^{n} (x_i + x_{i+1})(x_i\, y_{i+1} - x_{i+1}\, y_i) \tag{3}$$

$$y_{COG} = \frac{1}{6A} \sum_{i=0}^{n} (y_i + y_{i+1})(x_i\, y_{i+1} - x_{i+1}\, y_i) \tag{4}$$

Zur Berechnung der Koordinaten ist es erforderlich den Flächeninhalt A des Polygons anzugeben. Dieser lässt sich mit Hilfe von Gleichung 5 ermitteln [23].

$$2A = \sum_{i=0}^{n} (x_i\, y_{i+1} - x_{i+1}\, y_i) \tag{5}$$

Nachdem die Koordinaten des Flächenschwerpunktes für jedes Profil errechnet wurden, werden die Punkte, welche die Geometrie des Profils beschreiben analog zu dem Fädeln über die Sehne in Richtung des Vektors $\vec{V} \begin{pmatrix} -x_{COG} \\ -y_{COG} \end{pmatrix}$ verschoben. Danach ist der Vorgang des Fädelns über den Flächenschwerpunkt abgeschlossen.

5.1.2.3 Fädeln entlang einer Auffädellinie

Bei dem Fädeln von Profilen entlang einer Auffädellinie, die keine Gerade ist, welche parallel zur radialen Achse verläuft, wird das Propellerblatt mit einer Pfeilung und Neigung versehen.

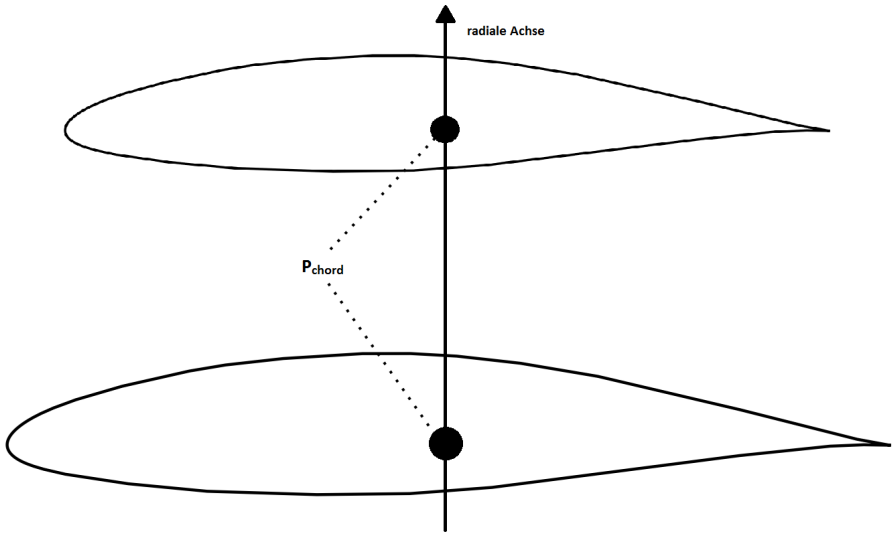


Abbildung 12: Schaubild zur Darstellung der Auffädellinie beim Fädeln über die Sehne

Prinzipiell werden die Profile beim Fädeln über die Sehne oder den Flächenschwerpunkt ebenfalls über eine Auffädellinie gefädelt. Diese ist jedoch die Radiale Achse, sodass das Propellerblatt weder mit einer Pfeilung noch mit einer Neigung versehen wird. Abbildung 12 zeigt ein Schaubild mit zwei Profilen, die über 50% der Sehnenlänge gefädelt wurden, sowie die dazugehörende Auffädellinie (radiale Achse).

Im weiteren Verlauf dieses Abschnitts wird die Auffädellinie als Kurve im zweidimensionalen Raum innerhalb der xz-Ebene betrachtet, die nicht parallel zur z-Achse (radiale Achse) verläuft. Abbildung 13 zeigt beispielhaft die auf den Radius von einem Meter normierte Auffädellinie der Schaufel des NASA SRIII-Propellers.

Die Auffädellinien, die in dieser Arbeit betrachtet werden, beginnen an der radialen Koordinate des Hubs. Bei der Auffädellinie aus Abbildung 13 liegt der Hub bei einem dimensionslosen Radius von 0,239. Die x-Koordinate jeder Auffädellinie beträgt am dimensionslosen Hubradius innerhalb dieser Arbeit immer Null. Bei dem Fädeln über eine Auffädellinie muss zunächst definiert werden, über welchen Punkt des Profils entlang der Auffädellinie gefädelt werden soll. Ein solcher Punkt P_{stack} könnte die Vorder- oder Hinterkante des Profils sein oder auch der in den vorherigen Abschnitten erwähnte Punkt auf der Sehne P_{chord} beziehungsweise der Flächenschwerpunkt. Das Profil wird in diesem Fall jedoch nicht derart verschoben, dass sich P_{stack} im Ursprung befindet. Das Profil wird lediglich in x-Richtung verschoben. Die

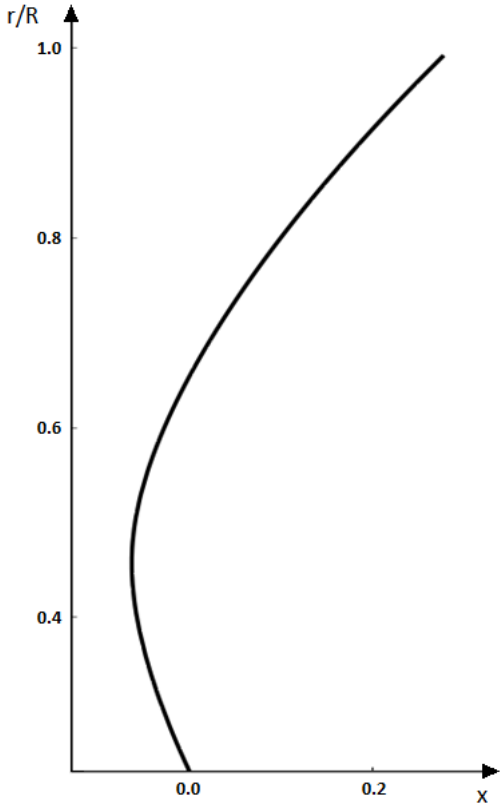


Abbildung 13: Normierte Auffädellinie der Schaufel des SRIII-Propellers

Verschiebung (sweep) für ein Profil, welches sich an der Radialen Koordinate r befindet, ergibt sich aus der Differenz der x-Koordinate von P_{stack} (Minuend) und der x-Koordinate der Auffädelline an der radialen Koordinate r (Subtrahend). Die Auffädellinie wird innerhalb der implementierten Funktionen durch eine geordnete Punkteliste von Hub zu Tip definiert. Liegt kein Punkt mit der gesuchten radialen Koordinate r innerhalb dieser Liste vor, so wird dieser durch eine lineare Interpolation ermittelt. Soll das Propellerblatt lediglich mit Pfeilung versehen werden, so muss lediglich die besagte Verschiebung in x-Richtung (sweep) ausgeführt werden. Der Vektor für die Verschiebung lautet also $\vec{V}\begin{pmatrix} -sweep \\ 0 \end{pmatrix}$. Soll das Propellerblatt neben der Pfeilung auch mit einer Neigung versehen werden, so ist der Vektor $\vec{V}$ noch um den Anstellwinkel des Profils zu rotieren (siehe Gleichung 1). Der daraus resultierende Vektor $\vec{V_{rot}}\begin{pmatrix} x_{rot} \\ y_{rot} \end{pmatrix}$ gibt die Verschiebung in x- und y-Richtung für alle Punkte vor, die die Geometrie des entsprechenden Profils beschreiben. Damit ist die Vorstellung der Algorithmen zum Fädeln von Profilen abgeschlossen. Häufig liegt für Propeller mit Pfeilung und Neigung keine Beschreibung der Auffädellinie in Form einer definierten Kurve oder Punkteverteilung vor. Diese wird im Rahmen dieser Arbeit jedoch benötigt, um die Profile wie zuvor beschrieben zu fädeln. In dem folgenden Abschnitt sollen deshalb zwei implementierte Algorithmen vorgestellt werden, die mittels der Parametrisierung eines Propellerblattes, welche im Rahmen dieser Arbeit betrachtet wird, eine Auffädellinie generieren.

5.1.3 Generierung von Auffädellinien

Im Kapitel 2.3 wurde der Zusammenhang zwischen dem Pfeilwinkel Λ und der Auffädellinie beschrieben. Die Parametrisierung zur Beschreibung der Geometrie eines Propellers, welche über die Schnittstelle von Propster durch ein Objekt der Klasse `Propeller` übergeben wird, beinhaltet auch den Pfeilwinkel Λ von jedem Profil, welches übergeben wird. Im Rahmen dieser Arbeit wurden zwei Algorithmen entwickelt, mit denen es möglich ist, anhand der Pfeilwinkel die Auffädellinie des Propellerblattes, welches durch die gelieferte Parametrisierung beschrieben wird, zu rekonstruieren. Die beiden Algorithmen unterscheiden sich dahingehend, dass ein Algorithmus darauf abzielt, die Auffädellinie möglichst genau zu rekonstruieren, während der andere Algorithmus mit wenigen Inputparametern eine Auffädellinie erzeugt. Letzterer Algorithmus kann verwendet werden, wenn nur sehr wenige Informationen über ein Propellerblatt und dessen Auffädellinie vorhanden sind. Außerdem ist dieser Algorithmus dazu

geeignet, um die Auffädellinie durch die Veränderung weniger Parameter schnell anzupassen. Dies könnte in der Optimierung von Propellerschaufeln von großem Nutzen sein. Obwohl die Optimierung von Propellerschaufeln kein definiertes Ziel dieser Arbeit ist, wurde eine Implementierung des zweiten Verfahrens als nützlich angesehen, da die Beschaffung der notwendigen Parameter für das erste Verfahren nicht immer glückt und somit eine Alternative zur Verfügung steht. Zunächst soll der Algorithmus vorgestellt werden, der die Rekonstruktion der Auffädellinie möglichst präzise vollzieht.

5.1.3.1 Präzise rekonstruktion einer Auffädellinie

Der entwickelte Algorithmus zur präzisen Rekonstruktion der Auffädellinie eines Propellerblattes benötigt einige Inputparameter. Dazu gehören die Pfeilwinkel der Profile und deren dimensionslose radiale Koordinate. Außerdem wird der dimensionslose Hubradius benötigt. Da es sich bei dem zu beschreibenden Algorithmus um ein iteratives Verfahren handelt, muss die Anzahl der Iterationen i vorgegeben werden. Die Präzision der Rekonstruktion der Auffädellinie steigt mit der Erhöhung der Anzahl der Iterationen. Zu einem späteren Zeitpunkt wird auf die Auswirkung einer variierenden Anzahl an Iterationen auf die rekonstruierte Affädellinie eingegangen. Zunächst soll jedoch das Verfahren genauer betrachtet werden. Innerhalb des nun vorgestellten Algorithmus wird eine normierte Auffädellinie erzeugt, die von dem dimensionslosen Hubradius in Richtung des Tips erzeugt wird. Der dimensionslose Tipradius ist eins. Mit jeder Iteration wird ein weiterer Punkt mit steigender radialer Koordinate, der die Auffädellinie definiert, erzeugt. Der radiale Abstand ($diff$) dieser Punkte ergibt sich aus der Differenz vom dimensionslosen Tip- und Hubradius geteilt durch die Anzahl an Iterationen subtrahiert mit eins.

$$diff = \frac{r/R_{Tip} - r/R_{Hub}}{i - 1} \tag{6}$$

Die Pfeilwinkel der Profile und deren dimensionslose Radiale Koordinaten können in einen funktionellen Zusammenhang gebracht werden, sodass der Pfeilwinkel sich aus einer Abhängigkeit des dimensionslosen Radius ergibt. Von dieser Funktion liegen lediglich die Punkte an den dimensionslosen radialen Koordinaten der Profile vor. Für die Generierung der Auffädellinie ist es jedoch erforderlich, von einer beliebigen dimensionslosen radialen Koordinate zwischen Hub und Tip den Pfeilwinkel bestimmen zu können. Da eine möglichst präzise Rekonstruktion der

Auffädellinie gewährleistet werden soll, sind auch die Pfeilwinkel an einer beliebigen dimensionslosen radialen Koordinate möglichst exakt zu bestimmen. Die Bestimmung der Pfeilwinkel über eine lineare Interpolation wurde deshalb nicht in Betracht gezogen. Stattdessen wurde eine DLR-interne Bibliothek verwendet, um eine B-Spline Kurve aus den gegebenen Punkten zu approximieren. Mit Hilfe des Brent-Verfahrens kann anschließend über die Laufvariable u des B-Splines iteriert und ein Punkt mit bestimmter dimensionsloser radialer Koordinate auf dem B-Spline ermittelt werden. Bei dem Brent-Verfahren handelt es sich um einen Algorithmus aus der numerischen Mathematik zur Bestimmung von Nullstellen. Bei der Iteration über die Laufvariable u eines B-Splines wird die x-Koordinate des Punktes auf dem Spline x(u) gesucht, dessen Differenz zur gesuchten dimensionslosen radialen Koordinate gleich Null ist und somit die Nullstelle ergibt. Prinzipiell wird die Funktion, welche durch den Kurvenverlauf des B-Splines beschrieben wird, um die gesuchte dimensionslose radiale Koordinate verschoben. Die daraus resultierende Funktion besitzt dort eine Nullstelle (x(u) = 0). In diesem Fall ist es legitim nach einer Nullstelle für x(u) zu suchen und nicht, so wie es für explizite Funktionen üblich ist, nach einem y(u), da sowohl x als auch y von der Laufvariablen u abhängig sind (Parametrische Funktion). Von dem ermittelten Punkt kann dann die y-Koordinate y(u) bestimmt werden, welche den gesuchten Pfeilwinkel an der dimensionslosen radialen Koordinate beinhaltet.

Mit Hilfe des Brent-Verfahrens ist es nun möglich, für jede radiale Koordinate zwischen Hub und Tip einen präzisen Wert des Pfeilwinkels zu bestimmen. In dem Kapitel 5.1.2.3 wurde erwähnt, dass der Verlauf der Auffädellinien, die in dieser Arbeit betrachtet werden, immer am dimensionslosen Hubradius mit der x-Koordinate null beginnt. Der erste Punkt, der die Auffädellinie definiert, ist somit gegeben als $P_{Hub}(0.0|r/R_{Hub})$. Von diesem Punkt aus beginnt der iterative Abschnitt dieses Verfahrens. Der Pfeilwinkel Λ an der Stelle r/R_{Hub} ist bekannt oder mit dem Brent-Verfahren zu ermitteln. Der Pfeilwinkel an der radialen Koordinate r ist definiert als Winkel zwischen der PCA und der Tangente der Auffädellinie. Die PCA kann mit der radialen Achse gleichgestellt werden. Der radiale Abstand $diff$ soll zwischen den Punkten, die die Auffädellinie definieren, eingehalten werden. Die radiale Koordinate des Punktes, welcher auf P_{Hub} zur Definition der Auffädellinie folgt, hat somit einen Wert von $r/R_{Hub} + diff$. Über den Abstand $diff$ und den Pfeilwinkel Λ kann dann mit Hilfe von Gleichung 7 die x-Koordinate des Punktes, der auf P_{Hub} folgt, bestimmt werden.

$$\tan\left(|\Lambda| * \frac{\pi}{180}\right) * diff = x \tag{7}$$

Abstrakt formuliert beschreibt Gleichung 8 und 9, wie die x- und y-Koordinate eines Punktes $P_{i+1}(x_{i+1}|y_{i+1})$ über den Punkt $P_i(x_i|y_i)$ und den Pfeilwinkel Λ_i an der radialen Koordinate y_i bestimmt werden kann. Die beiden Punkte beschreiben die zu rekonstruierende Auffädellinie. In der folgenden Iteration dient der ermittelte Punkt P_{i+1} als neuer Ausgangspunkt P_i. Die Iteration erfolgt so oft, bis die dimensionslose radiale Koordinate eins (r/R_{Tip}) erreicht wird.

$$x_{i+1} = x_i \pm \tan\left(|\Lambda_i| * \frac{\pi}{180}\right) * diff \tag{8}$$

$$y_{i+1} = y_i + diff \tag{9}$$

Ist der Pfeilwinkel Λ_i größer Null, so ist in Gleichung 8 eine Subtraktion auszuführen. Andernfalls muss addiert werden. Das Prinzip einer Iteration basiert auf der Winkelfunktion des Tangens. Abbildung 14 verdeutlicht dies noch einmal.

Nachdem die festgelegte Anzahl an Iterationen i erfolgt ist, liegen i+1 Punkte vor, welche die Auffädellinie definieren. Diese ist auf den Radius von eins skaliert. Eine Skalierung der normierten Punkte der Auffädellinie erfolgt, indem sowohl die x-Koordinate als auch die y-Koordinate jedes Punktes mit dem tatsächlichen Radius der Propellerschaufel multipliziert wird. Im folgenden Abschnitt wird der zweite Algorithmus zur Generierung einer Auffädelline vorgestellt, der es ermöglicht, mit einer geringen Zahl an Inputparametern eine solche Linie zu erzeugen.

5.1.3.2 Rekonstruktion einer Auffädellinie anhand weniger Parameter

Der zweite Algorithmus zur Rekonstruktion einer Auffädellinie benötigt deutlich weniger Parameter als Input, um eine Auffädellinie zu generieren. Es werden lediglich die Koordinaten des Startpunktes P_{Hub} und Endpunktes P_{Tip} der zu erzeugenden Auffädellinie benötigt sowie die Pfeilwinkel an den radialen Koordinaten der beiden Punkte (Λ_{Hub} und Λ_{Tip}). Abbildung

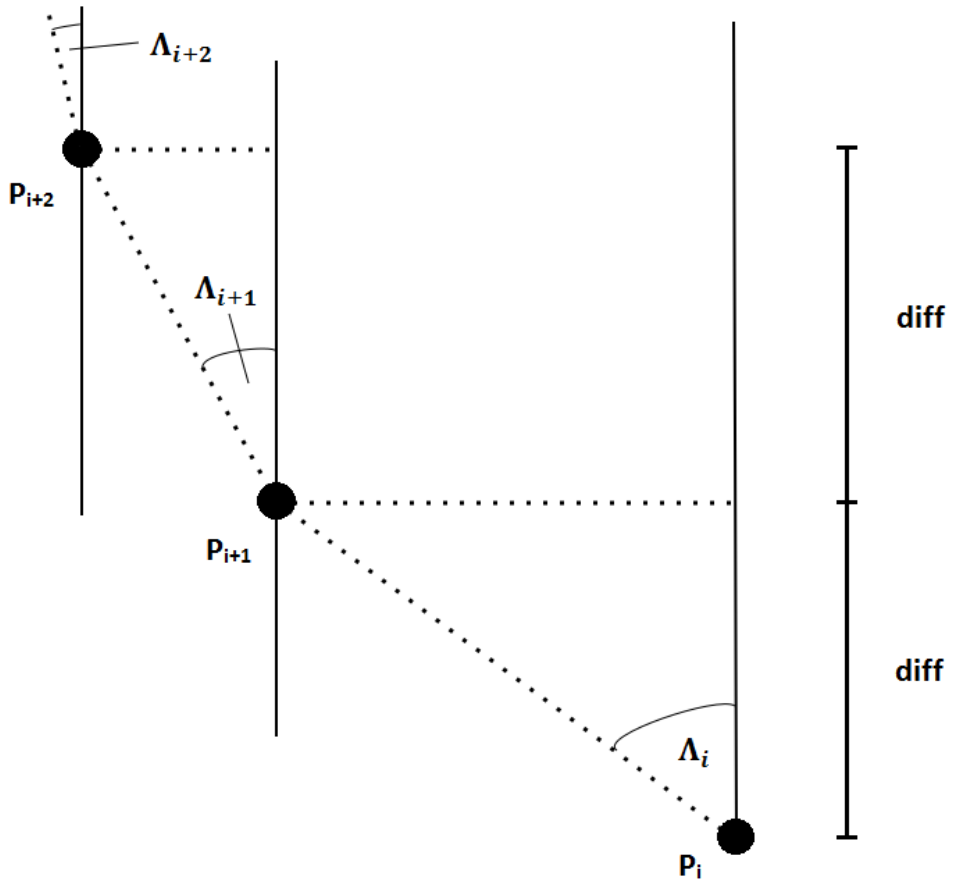


Abbildung 14: Schematische Zeichnung zur Verdeutlichung eines Iterationsintervalls zur Rekonstruktion der Auffädellinie eines Propellerblattes

15 zeigt die benötigten Inputparameter in einem Schaubild mit der aus diesen Parametern generierten Auffädellinie.

Die in Abbildung 15 dargestellte Auffädellinie wird in Form eines eingespannten B-Splines (clamped B-Spline) generiert. Eingespannte B-Splines mit n Kontrollpunkten $[P_1, P_2, \ldots, P_n]$ besitzen eine wesentliche Eigenschaft, die der Algorithmus zur Erzeugung der Auffädellinie ausnutzt. Die Koordinaten des ersten und letzten Kontrollpunktes (P_1 und P_n) sind gleich den Koordinaten des Start- und Endpunktes des B-Splines, welche die Kontrollpunkte definieren. Zusätzlich entsprechen die Verbindungsgeraden vom ersten zum zweiten Kontrollpunkt und vom vorletzten zum letzten Kontrollpunkt den Tangenten des B-Splines im Start- und Endpunkt. Bei dem Algorithmus wird ein eingespannter B-Spline verwendet, der durch drei Kontrollpunkte beschrieben wird. Der erste und dritte Kontrollpunkt wird durch die Inputparameter P_{Hub} und P_{Tip} definiert. Über diese beiden Punkte und die Pfeilwinkel (Λ_{Hub} und Λ_{Tip}) lässt sich der zweite, noch fehlende Kontrollpunkt jedoch errechnen, indem die Vektorgeradengleichungen der Tangenten im Punkt P_{Hub} und P_{Tip} bestimmt und deren Schnittpunkt berechnet wird. Der Schnittpunkt entspricht dem gesuchten Kontrollpunkt und lässt sich durch Gleichung 10 bestimmen:

$$\begin{pmatrix} x_{Hub} \\ y_{Hub} \end{pmatrix} + f * \begin{pmatrix} -\tan\left(\Lambda_{Hub} * \frac{\pi}{180}\right) \\ 1 \end{pmatrix} = \begin{pmatrix} x_{Tip} \\ y_{Tip} \end{pmatrix} + g * \begin{pmatrix} -\tan\left(\Lambda_{Tip} * \frac{\pi}{180}\right) \\ -1 \end{pmatrix} \quad (10)$$

Es lassen sich zwei Gleichungen mit den beiden Unbekannten f und g aufstellen (siehe Gleichung 11 und 12). Löst man f und g auf und setzt die Ergebnisse ein, so lässt sich der gesuchte Schnittpunkt ermitteln.

$$x_{Hub} + f * -\tan\left(\Lambda_{Hub} * \frac{\pi}{180}\right) = x_{Tip} + g * -\tan\left(\Lambda_{Tip} * \frac{\pi}{180}\right) \quad (11)$$

$$y_{Hub} + f = y_{Tip} - g \quad (12)$$

Abbildung 16 veranschaulicht diesen Zusammenhang noch einmal.

Anschließend lässt sich der B-Spline über die Kontrollpunkte, den Grad und den Knotenvektor

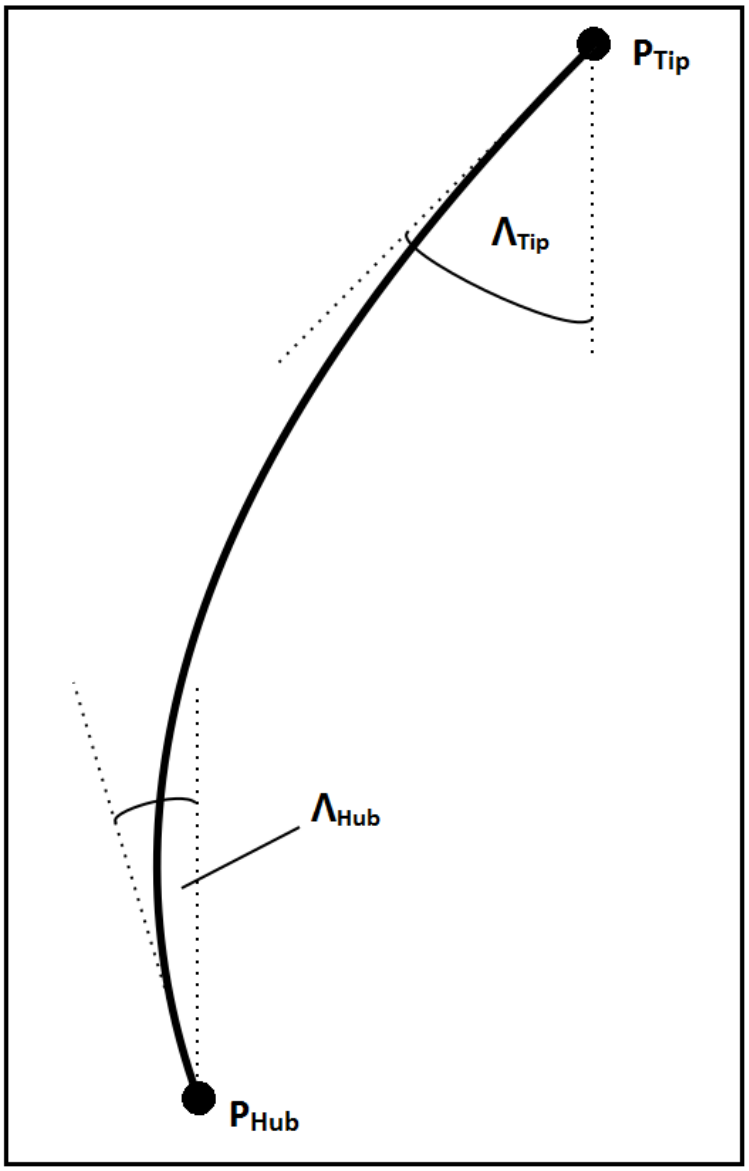


Abbildung 15: Generierte Auffädellinie anhand weniger Parameter

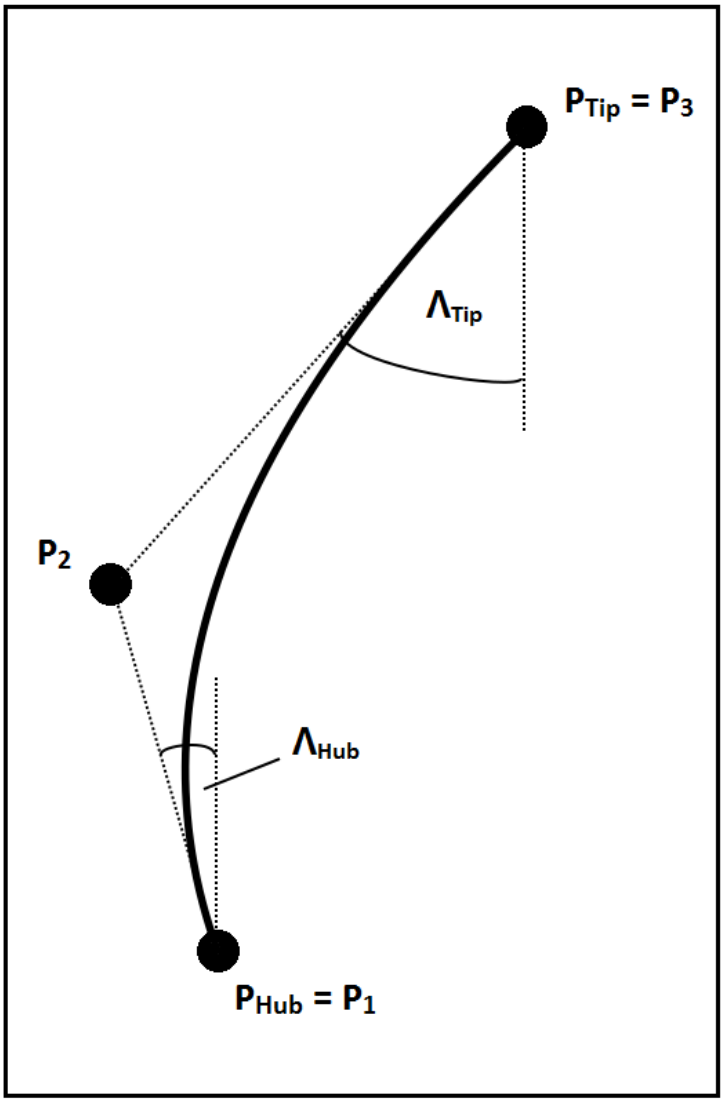


Abbildung 16: Schaubild zur Ermittlung des gesuchten Kontrollpunktes P_2

definieren. Der Grad p des B-Splines ist festgelegt auf zwei und der Knotenvektor liegt vor als $[0, 0, 0, 1, 1, 1]$. Auf eine detaillierte Beschreibung der Parameter zur Definition eines B-Splines wird an dieser Stelle verzichtet. Genauere Informationen können auch in diesem Fall aus [19] entnommen werden. Die Funktionalität zur Erzeugung eines B-Splines wird durch eine interne Bibliothek des DLR bereitgestellt. Diese stellt auch eine Funktion bereit, mit der es möglich ist, einen Punkt auf dem B-Spline über die Laufvariable u des B-Splines zu ermitteln. Die Laufvariable u besitzt den Definitionsbereich $[0, 1]$. Indem in gleichmäßigen Abständen über diesen Bereich iteriert wird, lassen sich eine feste Anzahl an Punkten ermitteln, die auf dem B-Spline liegen und somit die Auffädellinie beschreiben. Damit ist der Algorithmus vollständig beschrieben. Anzumerken ist an dieser Stelle noch, dass eine weitere Funktion implementiert wurde, die es ermöglicht, die drei Kontrollpunkte direkt als Parameter zu übergeben und daraus die Punkte des B-Splines zu generieren, welche die Auffädellinie beschreiben. Diese Funktion bietet Flexibilität bei der Rekonstruktion von Auffädellinien. Im folgenden Abschnitt wird beschrieben, wie mit Hilfe der gefädelten Profile und der Bibliothek Open Cascade Technology eine Mantelfläche generiert wird.

5.1.4 Mantelflächengenerierung mit Open Cascade Technology

Mit Hilfe der zuvor vorgestellten Algorithmen ist es möglich, die normierten Profil-Geometrien, welche von der Schnittstelle von Propster geliefert werden, zu transformieren, sodass die resultierenden Profil-Geometrien die Originalform des Propellerblattes in Originalgröße beschreiben. Die transformierten Profil-Geometrien dienen der Open Cascade Technology Bibliothek anschließend als Inputparameter, um eine Mantelfläche aus diesen geometrischen Informationen zu generieren. In diesem Abschnitt soll nun darauf eingegangen werden, welche Datenstrukturen und Funktionen von Open Cascade Technology in der implementierten Bibliothek zur Mantelflächengenerierung von Propellerschaufeln genutzt werden, um den Mantel der Propellerschaufel aus den transformierten Profil-Geometrien zu erzeugen. Informationen zu den genutzten Klassen von Open Cascade Technology wurden aus der Dokumentation entnommen [24]. Die transformierten Profil-Geometrien liegen in Form von geordneten Punktelisten vor. Diese müssen zunächst in eine Datenstruktur des Open Cascade Technology Datenformats adaptiert werden. Es handelt sich bei der verwendeten Datenstruktur um ein zweidimensionales Array (Klasse `TColgp_Array2OfPnt`), sodass alle Profil-Geometrien innerhalb dieses Arrays abgelegt werden können. Zur Veranschaulichung der Struktur des gefüllten zweidimensionalen Arrays dient die folgende Matrix, welche eine Anzahl m von Profilen mit jeweils n Punkten

beinhaltet. Jedes Profil wird in einer Zeile angeordnet. Eine Zeile besitzt n+1 Spalten, sodass in jeder Spalte ein Punkt, der die Geometrie des Profils beschreibt, hinterlegt wird. In der letzten Spalte mit dem Indizes n+1 wird der erste Punkt der Zeile erneut hinterlegt. Der Grund für dieses Vorgehen wird zu einem späteren Zeitpunkt in diesem Kapitel erwähnt.

$$\begin{pmatrix} P_{1,1} & \dots & P_{1,n+1} \\ \vdots & \ddots & \vdots \\ P_{m,1} & \dots & P_{m,n+1} \end{pmatrix}$$

Nachdem das Array gefüllt wurde, dient dieses als Inputparameter zur Erzeugung einer B-Spline Fläche (Klasse `Geom_BSplineSurface`), die durch einen Approximationsalgorithmus (Klasse `GeomAPI_PointsToBSplineSurface`) erzeugt wird. Diese Fläche könnte anschließend durch einen weiteren internen Algorithmus von Open Cascade Technology (Klasse `BRepBuilderAPI_MakeFace`) in einen sogenannten Shape (`Klasse TopoDS_Shape`) übertragen werden. Shapes dienen in Open Cascade Technology dazu, geometrische Formen zu beschreiben. Von der Klasse `TopoDS_Shape` sind einige Klassen abzuleiten, die für die Generierung der Mantelfläche genutzt wurden. Diese werden in Tabelle 1 aufgelistet und deren Funktion erklärt.

Shape	**OpenCascade Klasse**	**Beschreibung**
Vertex	`TopoDS_Vertex`	Entspricht einem Punkt
Edge	`TopoDS_Edge`	Entspricht einer Kante
Wire	`TopoDS_Wire`	Entspricht einem Teil (Draht) eines Drahtmodells
Face	`TopoDS_Face`	Entspricht einer Fläche
Shell	`TopoDS_Shell`	Entspricht der Ummantelung einer Geometrie
Solid	`TopoDS_Solid`	Entspricht einem Festkörper

Tabelle 1: Shapes der Open Cascade Bibliothek [24]

Mit dem erzeugten Shape, bei dem es sich um eine Fläche (`TopoDS_Face`) handelt, liegt prinzipiell ein Objekt der Klassenstruktur vor, welches für die Generierung des CAD-Outputformats zur Beschreibung der Propellermantelfläche genutzt werden könnte. Die Aufgabenstellung dieser Arbeit wäre somit erfüllt. Es wurde jedoch als sinnvoll erachtet, aus der generierten Fläche einen Festkörper (`TopoDS_Solid`) erzeugen zu können. Mit dem Festkörper könnte die dreidimensionale Strömungsrechnung ebenfalls durchgeführt werden. Zusätzlich ist es

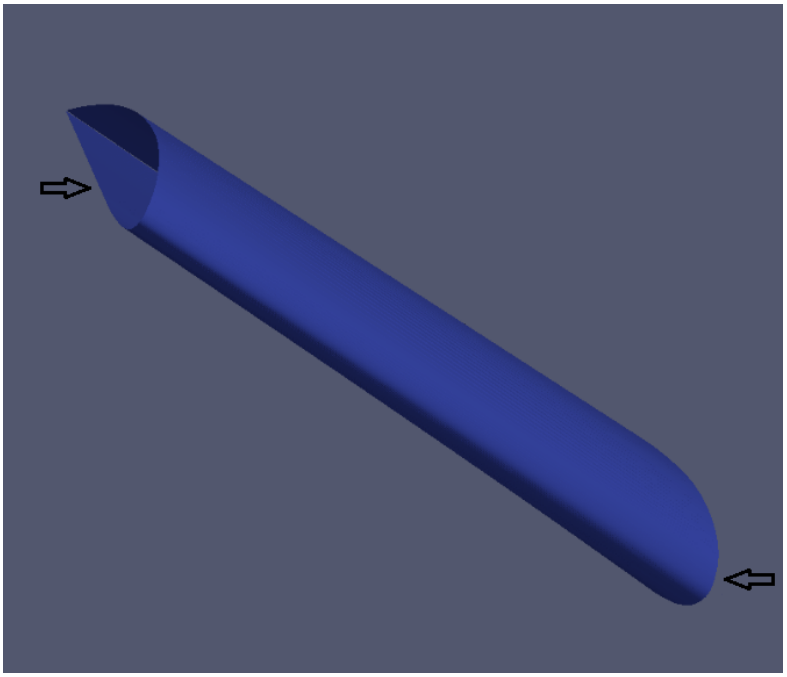


Abbildung 17: Mantelfläche zur Veranschaulichung der Öffnungen an Hub und Tip

möglich, mit Hilfe des Festkörpers an weitere Informationen des Propellers zu gelangen, wie zum Beispiel das Volumen oder Drehmomente. Aus diesem Grund wurde der Algorithmus zur Generierung der Mantelfläche um die Generierung eines Festkörpers erweitert. Zur Generierung eines Festkörpers wird die Mantelfläche des Propellerblattes genutzt. Diese ist an Hub und Tip geöffnet. Abbildung 17 zeigt eine beispielhafte Mantelfläche, die aus einigen Profilen generiert wurde. Die Geometrie in dieser Abbildung dient lediglich zur Veranschaulichung der Öffnungen.

Bei einem Festkörper sollen diese Öffnungen nicht mehr vorliegen. Deshalb sind die Flächen in Form der Öffnungen zu generieren. Anschließend kann eine Shell aus den beiden Flächen in Form der Öffnung und der Mantelfläche erzeugt werden, die anschließend in einen Festkörper umgewandelt werden kann. Die einzelnen Iterationsschritte werden nun vorgestellt. Zur Generierung der Flächen in Form der Öffnungen des Mantels ist es erforderlich, der zuvor generierten B-Spline Fläche (Klasse `Geom_BSplineSurface`) Informationen zu entnehmen. Von der B-Spline Fläche sollen die Kanten übernommen werden, die die beiden Öffnungen umranden. Eine B-Spline Fläche wird über eine Funktion mit zwei Laufvariablen u und v

definiert. In Bezug auf die generierte Mantelfläche einer Propellerschaufel dient eine der beiden Variablen dazu, entlang der radialen Richtung über die Mantelfläche zu iterieren, während die andere Laufvariable dazu dient, an einer radialen Koordinate über den Verlauf eines Profils zu iterieren. Welche der beiden Variablen bei der B-Spline Fläche für den radialen Verlauf oder den Verlauf in Umfangsrichtung des Profils im Zusammenhang steht, ist jedoch zunächst herauszufinden. An dieser Stelle wird nun deutlich warum das zweidimensionale Array eine Zeilenlänge von n+1 besitzt, obwohl lediglich n Punkte ein Profil beschreiben. Der letzte Punkt einer Zeile entspricht dem Ersten. Wird aus dem zweidimensionalen Array eine Fläche approximiert, so wird durch diese Gegebenheit gewährleistet, dass die Fläche an der Hinterkante geschlossen ist. Über eine Funktion der Klasse `Geom_BSplineSurface` kann abgefragt werden, ob die Laufvariable u beziehungsweise v über einen geschlossenen B-Spline iteriert. Dies sollte nur für eine der beiden Variablen zutreffen, sodass die beiden Variablen den entsprechenden Verläufen der Fläche zugeordnet werden können. Über weitere Funktionen der Klasse lässt sich bestimmen, in welchem Definitionsbereich die Laufvariablen u und v operieren. Außerdem steht eine Funktion innerhalb der Klasse `Geom_BSplineSurface` zur Verfügung, die aus der Fläche an der Koordinate der Laufvariablen u oder v den entsprechenden B-Spline (Klasse `Geom_BSpline`) erzeugt. Wird nun von dem kleinstmöglichen und größtmöglichen wählbaren Wert des Definitionsbereichs der Laufvariable, welche im Zusammenhang mit dem radialen Verlauf der B-Spline Fläche steht, jeweils ein B-Spline erzeugt, so beschreiben diese den Verlauf der Kanten, welche die Öffnungen an Hub und Tip abgrenzen. Die Flächen, die von den B-Splines umrandet werden, können über Zwischenschritte generiert werden. Dazu wird mit Hilfe der Klasse `BRepBuilderAPI _MakeEdge` aus jedem B-Spline zunächst ein Objekt der Klasse `TopoDS_Edge` erzeugt. Aus diesen wird dann mittels der Klasse `BRepBuilderAPI_MakeWire` jeweils ein Objekt der Klasse `TopoDS_Wire` generiert. Anschließend können aus diesen Objekten die benötigten Flächen (Klasse `TopoDS_Face`) zum Schließen der Öffnungen der Mantelfläche generiert werden. Die Funktionalität wird durch die Klasse `BRepBuilderAPI_MakeFace` bereitgestellt. Mit Hilfe der Klasse `BRepBuilderAPI_Sewing` kann aus den beiden gewonnenen Flächen und der Mantelfläche ein Objekt der Klasse `TopoDS_Shell` erzeugt werden. Aus diesem Objekt kann anschließend mittels der Klasse `BRepBuilderAPI_MakeSolid` ein Festkörper Objekt (`Klasse TopoDS_Solid`) generiert werden. Dieses Objekt dient als Inputparameter für die Generierung der Dateien in den CAD-Outputformaten STEP und IGES. Die Generierung des Outputs wird in dem folgenden Abschnitt thematisiert.

5.1.5 Generierung der Outputdateien

Die Generierung der Dateien in dem Outputformaten STEP oder IGES, welche ein Objekt der Klasse `TopoDS_Shape` beschreiben, erfolgt über bereitgestellte Klassen von Open Cascade Technology (Klasse `STEPControl_Writer`, `STEPControl_Controller`, `IGESControl_Writer` und `IGESControl_Controller`). Zunächst muss durch die Controller Klassen der Export in das entsprechende Datenformat über die Funktion `Init()` initialisiert werden. Anschließend kann den Writer Klassen der Shape übergeben werden, der von den Outputdateien beschrieben werden soll. Danach kann die Outputdatei erzeugt werden. Mit dem Abschluss dieses Abschnittes sind die Algorithmen, die in der zu erstellenden Bibliothek zur Generierung von Mantelflächen von Propellerschaufeln genutzt werden, allesamt beschrieben worden. In dem folgenden Abschnitt wird die IT-spezifische Umsetzung der Bibliothek genauer betrachtet, indem die Schnittstellen der Bibliothek beschrieben werden und die erzeugte Klassenstruktur der Bibliothek vorgestellt wird.

5.2 IT-spezifische Umsetzung

In dem folgenden Abschnitt wird die IT-spezifische Umsetzung genauer betrachtet. Zunächst sollen die Schnittstellen der Bibliothek beschrieben werden.

5.2.1 Schnittstellen der Bibliothek

Die Bibliothek zur Generierung von Mantelflächen für Propellerschaufeln bietet zwei Schnittstellen. Die erste Schnittstelle dient dazu, die benötigten Daten zur Generierung der Mantelfläche bereitstellen zu können. Vorgestellt wurde bereits die Schnittstelle zu den Geometrie-Informationen eines Propellers von Propster innerhalb des Kapitels 3.2. Diese Schnittstelle von Propster liefert alle benötigten geometrischen Informationen zur Generierung einer Mantelfläche. Somit wurde eine zu der Schnittstelle von Propster kompatible Schnittstelle innerhalb der Bibliothek zur Generierung von Mantelflächen für Propellerschaufeln implementiert, die die Datenstrukturen von Propster mit den benötigten Informationen entgegennimmt. In einem ersten Iterationsschritt werden die normierten, gelieferten Daten zur geometrischen Beschreibung eines Propellerblattes aufbereitet, sodass anschließend ein Propellerblatt in Originalform und Originalgröße vorliegt. Dieser Iterationsschritt geschieht mit Hilfe der Klasse `PropellerBladeSkinGeometryBuilder`. Diese Klasse dient als Schnittstelle für die geome-

trischen Inputdaten, die von Propster geliefert werden, und transformiert diese durch die zuvor vorgestellten Algorithmen. Die Klasse `PropellerBladeSkinGeometryBuilder` baut Objekte der Klasse `PropellerBladeSkinGeometry`. Diese Klasse dient als Datenstruktur, welche die transformierten geometrischen Daten des Mantels der Propellerschaufel in Originalgröße und Originalform beinhaltet. Bei der Klasse `PropellerBladeSkinGeometryBuilder` wurde das Erzeugungsmuster Builder angewandt. Es dient dazu, den Erzeugungsprozess von komplexen Objekten durch den Builder und dessen Repräsentation durch das erzeugte Objekt zu trennen und soll ermöglichen, verschiedene Repräsentationen zu erzeugen. Gleichzeitig ermöglicht das Erzeugungsmuster durch Abstraktion eine Erweiterbarkeit. Ein Nachteil des Erzeugungsmusters ist die Kopplung zwischen dem konkreten Builder und dem gebauten Produkt. Ändert sich etwas in der Struktur des Produktes, so sind auch die Builder anzupassen [25]. Dieser Nachteil wurde jedoch in Kauf genommen, da die Vorteile überwiegen und drastische Änderungen des Produktes nicht zu erwarten sind. Auf mögliche Erweiterungen bezogen auf diesen Anwendungsfall wird im nächsten Abschnitt noch einmal eingegangen, in dem auch eine genauere Beschreibung der angesprochenen Klassen `PropellerBladeSkinGeometry` und deren Builder erfolgt. Zunächst soll jedoch die zweite Schnittstelle der entwickelten Bibliothek zur Generierung von Mantelflächen für Propellerschaufeln beschrieben werden. Diese Schnittstelle dient dazu, die Output Dateien zu generieren, welche der Visualisierung und Beschreibung der Mantelflächen einer Propellerschaufel dienen. Umgesetzt wurde diese Schnittstelle durch die abstrakte Klasse `AbstractExporter`. Sie definiert eine abstrakte Funktion `exportFile()`, durch die eine Datei in einem CAD-Datenformat generiert wird. Als Parameter müssen dieser Funktion ein Objekt der `Klasse TopoDS_Shape` übergeben werden, welches die Mantelfläche der Propellerschaufel repräsentiert, sowie der Pfad unter dem die Datei hinterlegt werden soll. In welchem CAD-Datenformat die Datei erstellt wird, entscheidet sich durch die Nutzung der dafür vorgesehenen erbenden Klassen von `AbstractExporter`. Momentan sind für die Datenformate STEP und IGES die Klassen `STEPExporter` und `IGESExporter` implementiert. Die Erweiterbarkeit ist durch die Vererbungshierarchie gewährleistet, sodass ohne Probleme weitere Exporter Klassen geschrieben und an die aktuelle Struktur der Schnittstelle angebunden werden können. Nachdem die Schnittstellen der Bibliothek zur Generierung von Mantelflächen für Propellerschaufeln vorgestellt wurden, soll nun die gesamte Klassenstruktur der Bibliothek betrachtet werden.

5.2.2 Klassenstruktur der Bibliothek

In diesem Abschnitt soll die Klassenstruktur der Bibliothek zur Mantelflächengenerierung von Propellerschaufeln beschrieben werden. Die Beschreibung erfolgt chronologisch in der Reihenfolge, in der die einzelnen Klassen genutzt werden, um aus den normierten Daten zur geometrischen Beschreibung von Propellerschaufeln eine Mantelfläche zu erzeugen. Zunächst soll deshalb noch einmal die Klasse `PropellerBladeSkinGeometryBuilder` aufgegriffen werden, die als Schnittstelle dient, um die geometrischen Daten, welche von Propster geliefert werden, zu übernehmen und auf die Originalgröße und Originalform des Propellerblattes zu transformieren. Der konkrete Builder der Klasse `PropellerBladeSkinGeometryBuilder` erbt von der Abstrakten Klasse `AbstractPropellerBladeGeometryBuilder`, einem abstrakten Builder (siehe Abb. 18). Diese Klasse definiert eine abstrakte Funktion `buildGeometry()`. Diese Funktion wird von erbenden Klassen überschrieben, sodass Geometrien einer Propellerschaufel erzeugt werden können. Im Anwendungsfall dieser Arbeit wurde die Funktion innerhalb der Klasse `PropellerBladeSkinGeometryBuilder` derart überschrieben, dass die Geometrie der Mantelfläche einer Propellerschaufel erzeugt wird. Prinzipiell ist es jedoch möglich, die Klassenstruktur der Bibliothek dahingehend zu erweitern, weitere Builder Klassen, die von `AbstractPropellerBladeGeometryBuilder` erben, zu implementieren, um weitere Geometrien einer Propellerschaufel zu generieren. So könnten beispielsweise auch Strukturen im Inneren der Propellerschaufel, welche durch die Mantelfläche überdeckt werden, generiert werden. Ein Anwendungsfall, für den die Generierung von inneren Strukturen nötig wäre, ist die Gewichtsabschätzung von Propellerschaufeln. Propellerschaufeln sind in mehreren Schichten aufgebaut, welche sich aus verschiedenen Materialien, wie Leichtbaustoffen, zusammensetzen. Bei einer Gewichtsabschätzung einer Propellerschaufel wären beispielsweise die Volumen der einzelnen Schichten zu ermitteln. Neben den abstrakten und den konkreten Buildern definiert das Builder-Erzeugungsmuster noch einen weiteren Teilnehmer innerhalb des Prozesses zur Erzeugung eines Objektes. Der sogenannte Director übernimmt die Aufgabe, den Erzeugungsprozess anzustoßen, indem diesem ein Objekt eines konkreten Builders übergeben wird. Der Director ist in dem Klassendiagramm nicht hinterlegt. Der Director wurde jedoch implementiert.

Die Funktion `buildGeometry()` liefert eine dreidimensionale Geometrie in Form von geordneten Punktelisten (`QList<QList<QVector3D>>`) zurück. Im Rahmen des Anwendungsfalls dieser Arbeit beschreiben die Punktelisten die Geometrie der Mantelfläche der Propellerschaufel. Jede Punkteliste (`QList<QVector3D>`) beschreibt ein Profil der Propellerschaufel. Als

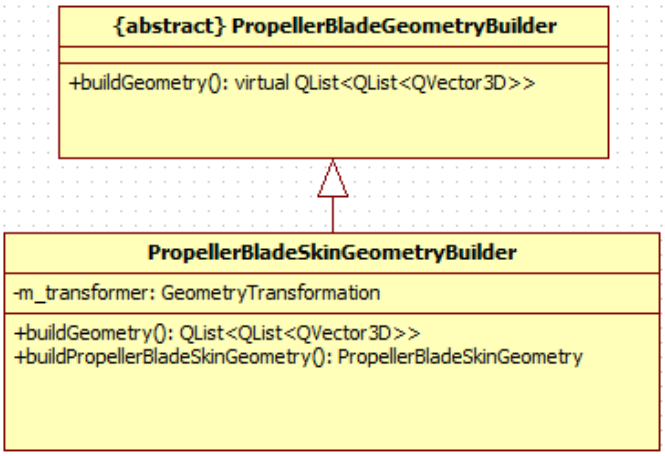


Abbildung 18: Klassendiagramm der Klassen AbstractPropellerBladeGeometryBuilder und PropellerBladeSkinGeometryBuilder

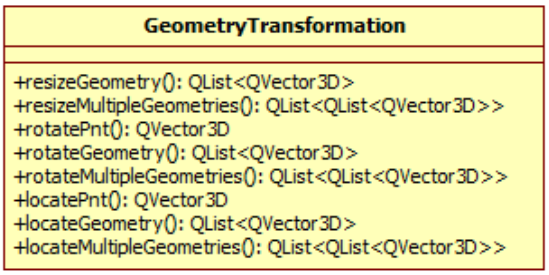


Abbildung 19: Klassendiagramm der Klasse GeometryTransformation

Parameter muss dieser Funktion ein Zeiger auf ein Objekt der Klasse `Propeller` übergeben werden. Das Objekt der Klasse `Propeller` beinhaltet nahezu alle geometrischen Informationen, die für die Erzeugung der Mantelfläche benötigt werden (siehe Kapitel 3.2, insbesondere Abb. 9). Die normierten Daten zur Beschreibung der Geometrie einer Propellerschaufel werden analog zu dem Kapitel 5.1.1 transformiert. Die Transformation der Profile erfolgt mit Hilfe der Klasse `GeometryTransformation`, die Funktionen bereitstellt, um Geometrien, die durch geordnete Punktelisten beschrieben werden, zu skalieren, zu rotieren und zu verschieben (siehe Abb. 19).

Ein Objekt dieser Klasse ist als Membervariable (`m_transformer`) in der Klasse `PropellerBladeSkinGeometryBuilder` hinterlegt, sodass deren Funktionalität bereitgestellt wird (siehe Abb. 18). Im Anschluss an die Transformation der geometrischen Daten liegen die Profile

in Originalgröße und um den Anstellwinkel rotiert vor. Sie müssen danach entlang der radialen Achse gefädelt werden. In dem Kapitel 5.1.2 wurden verschiedene Algorithmen, die die Profile in verschiedener Weise im dreidimensionalen Raum zueinander anordnen, vorgestellt. Welcher dieser Algorithmen für eine vorliegende Propellerschaufel angewandt werden soll entscheidet der Benutzer durch die Übergabe eines weiteren Parameters an die Funktion `buildGeometry()`. Dieser muss ein Objekt, welches von der Klasse `AbstractAirfoilStackingStrategy` erbt, übergeben werden. Die Klasse `AbstractAirfoilStackingStrategy` definiert eine Funktion `stackAirfoils()`, die von den erbenden Klassen überschrieben werden muss. Diese Funktion übernimmt die Aufgabe, die Profile nach einem bestimmten Algorithmus zu fädeln. Jeder Algorithmus, der in dieser Arbeit vorgestellt wurde, wird in der Bibliothek in Form einer Klasse, die von `AbstractAirfoilStackingStrategy` erbt, repräsentiert. So liegen mit den Klassen `ChordStackingStrategy`, `COGStackingStrategy`, `SweepLineStackingStrategy` sowie `NASASRStackingStrategy` vier Klassen vor, deren überschriebene Funktionen `stackAirfoils()` das Fädeln über die Sehne (chord), über den geometrischen Schwerpunkt (center of gravity (COG)) und über eine Auffädellinie vornehmen. Bei dem Fädeln über eine Auffädellinie wird unterschieden zwischen dem Fädeln, bei dem nur eine gepfeilte Propellerschaufel erzeugt wird (`SweepLineStackingStrategy`) und dem Fädeln, bei dem neben der Pfeilung auch eine Neigung der Propellerschaufel erzeugt wird `NASASRStackingStrategy`. Abbildung 20 zeigt das Klassendiagramm der Strategien zum Fädeln der Profile.

Parameter, die die überschriebene Funktion `stackAirfolis()` benötigt, werden über Setter-Methoden übergeben und in entsprechenden Membervariablen hinterlegt. So ist es zum Beispiel erforderlich, der Klasse `ChordStackingStrategy` mitzuteilen, über wie viel Prozent der Sehnenlänge gefädelt werden soll. Über die Setter-Methode der Variable `m_stackAt` wird dies ermöglicht. Den beiden Strategien zum Fädeln über eine Auffädellinie muss eine entsprechende Auffädellinie übergeben werden. Zur Erzeugung einer Auffädellinie nach den Algorithmen, die in Kapitel 5.1.3 beschrieben wurden dient die Klasse `SweepLineBuilder`. Abbildung 21 zeigt das Klassendiagramm dieser Klasse.

Die Klasse `SweepLineBuilder` stellt insgesamt vier Funktionen zur Generierung von Auffädellinien zur Verfügung. Die Funktion `buildSweepLine()` ist dreifach implementiert und wird im Klassendiagramm nur einmal dargestellt. Die ersten beiden Funktionen innerhalb des Klassendiagramms dienen zur Ausführung des Algorithmus, welcher in Kapitel 5.1.3.1 beschrieben wurde. Während die Funktion `buildNormedSweepLine()` eine auf den Radius von eins normierte Auffädellinie generiert, erzeugt die erste der drei Funktionen `buildSweepLine()` eine

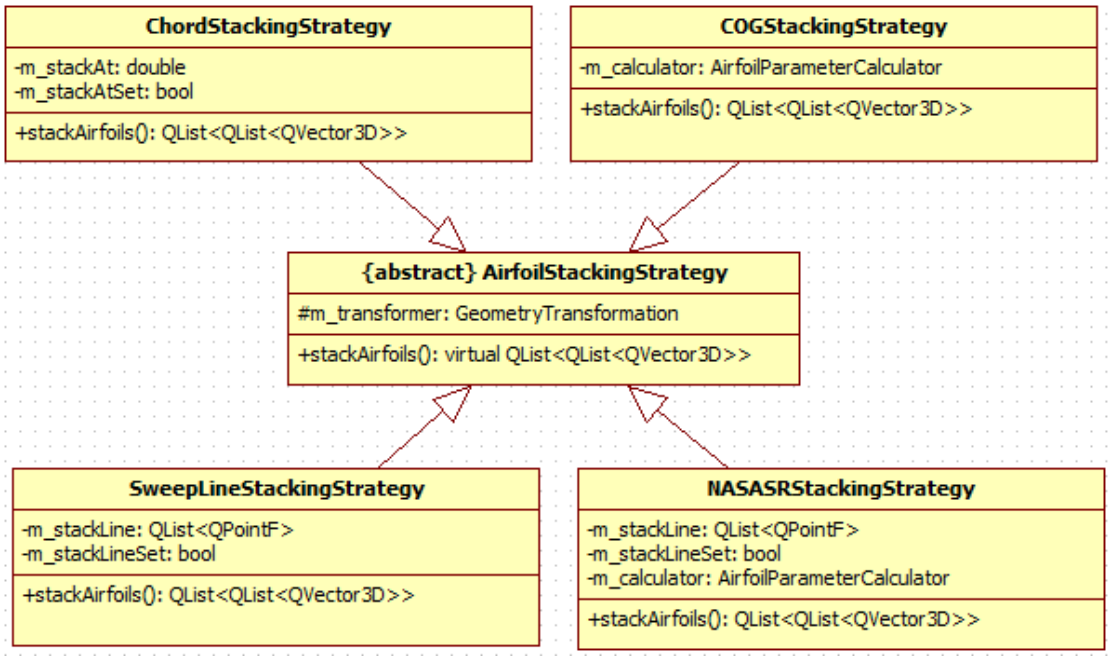


Abbildung 20: Klassendiagramm der Strategien zum Fädeln der Profile

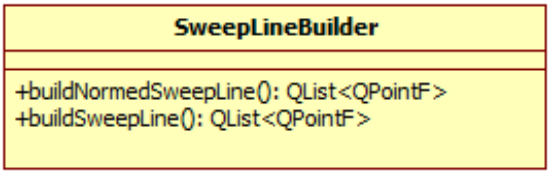


Abbildung 21: Klassendiagramm der Klasse SweepLineBuilder

Auffädellinie nach dem gleichen Algorithmus und skaliert diese im Anschluss auf den Propellerradius. Die Funktionen unterscheiden sich nur in der Skalierung, weshalb diese Funktionen auch die gleichen Parameter übergeben bekommen. Der Funktion `buildSweepLine()` wird zusätzlich der Propellerradius als Parameter übergeben. Beiden Funktionen wird eine geordnete Punkteliste mit steigenden dimensionslosen, radialen Koordinaten (x-Koordinaten der Punkte) übergeben. Sie beinhalten als y-Koordinate den Pfeilwinkel Λ an der entsprechenden radialen Koordinate. Weiterhin wird beiden Funktionen die Anzahl an auszuführenden Iterationen und der dimensionslose Hubradius übergeben. Die beiden anderen Funktionen `buildSweepLine()` repräsentieren die Algorithmen, die im Kapitel 5.1.3.2 vorgestellt wurden. Einer der beiden Funktionen können die drei Kontrollpunkte des B-Splines, der die Auffädellinie beschreibt, übergeben werden, sowie die Anzahl an Punkten , die zur Beschreibung der Auffädellinie zurückgegeben werden. Die andere Funktion benötigt als Eingabe Parameter den Punkt, an dem der Verlauf der Auffädellinie beginnt und endet, sowie die Pfeilwinkel Λ an diesen beiden Punkten. Auch dieser Funktion ist die Anzahl an Punkten zu übergeben, die die generierte Auffädellinie beschreiben. Erst nachdem die Membervariablen der Fädel-Strategien gesetzt wurden, ist es möglich, die Funktion `stackAirfoils()` erfolgreich auszuführen. Sind die entsprechenden Membervariablen zuvor nicht gesetzt worden, so werden die Profile ohne Fädelung in ursprünglicher Anordnung zurückgegeben und der Benutzer über die Kommandozeile darüber informiert, dass ein Setzen der entsprechenden Variablen nicht erfolgte. Für die zu setzenden Membervariablen in den einzelnen Klassen wurden weiterhin Variablen des Datentyps boolean hinterlegt. Diese können über eine Getter-Methode abgefragt werden und liefern true zurück, wenn die entsprechende Membervariable erfolgreich gesetzt wurde. Ansonsten liefern diese false zurück. Getter- und Setter-Methoden werden in den Klassendiagrammen dieser Arbeit nicht berücksichtigt, um diese kompakt zu halten. Sie wurden jedoch implementiert. Die Klassen `COGStackingStrategy` und `NASASRStackingStrategy` beinhalten als Membervariable ein weiteres Objekt der Klasse `AirfoilParameterCalculator`. Diese Klasse stellt Funktionen bereit, um verschiedene Parameter eines Profils zu ermitteln, welches durch eine geordnete Punkteliste beschrieben wird. Abbildung 22 zeigt das Klassendiagramm von `AirfoilParameterCalculator`.

So greift die Klasse `COGStackingStrategy` auf die Funktionen zur Ermittlung des Flächeninhalts und des geometrischen Schwerpunktes zu (siehe Gleichung 3-5). Die Klasse `NASASR-StackingStrategy` nutzt die Funktion zur Ermittlung des Anstellwinkels eines Profils `calculateBladeAngleDeg()`. Die Funktion `stackAirfoils()` der Strategien zum Fädeln

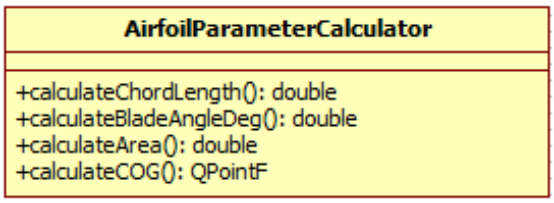


Abbildung 22: Klassendiagramm der Klasse AirfoilParameterCalculator

der Profile wird von der Funktion `buildGeometry()` der Klasse `PropellerBladeSkinGeometryBuilder` aufgerufen. Anschließend liegt die Schaufel-Geometrie in Originalgröße und Originalform vor. Bei der Implementierung der Klassen, die die Algorithmen zum Fädeln der Profile beinhalten, wurde das Strategy-Verhaltensmuster angewandt. Bei diesem Muster wird eine Strategie beziehungsweise ein Algorithmus, in diesem Anwendungsfall die Funktion `stackAirfoils()` zum Fädeln der Profile, an eine Klasse gekapselt. Verschiedene Klassen können diese Funktion beinhalten und eine eigene Strategie zur Lösung eines Problems bereitstellen. In diesem Anwendungsfall sind dies die verschiedenen Algorithmen zum Fädeln der Profile durch die überschriebene Funktion `stackAirfoils()`. Die Klassen, welche diese Funktion überschreiben, erben von einer abstrakten Klasse (abstrakte Strategie), welche die Funktion zur Lösung des Problems definiert. In diesem Fall ist dies `AbstractAirfoilStackingStrategy`. Das Strategy-Verhaltensmuster ermöglicht die Flexibilität eine der Strategien zum Fädeln der Profile auswählen zu können und in der Funktion `buildGeometry()` der Klasse `PropellerBladeSkinGeometryBuilder` zu nutzen. Möglich ist es auch, zur Laufzeit des Programms zu entscheiden, welche Strategie genutzt werden soll. Eine Erweiterung durch andere Strategien zum Fädeln gestaltet sich als unkompliziert, indem eine weitere Klasse, die von `AbstractAirfoilStackingStrategy` erbt, implementiert werden könnte [25]. Die eigentliche Aufgabe der Klasse `PropellerBladeSkinGeometryBuilder` ist es jedoch nicht, die Geometrie der Mantelfläche als geordnete Punktelisten durch die Funktion `buildGeometry()` zu erzeugen, sondern ein Objekt der Klasse `PropellerBladeSkinGeometry`, welches als Datenstruktur zur Bereitstellung der aufbereiteten Geometrie-Informationen dient und die geordneten Punktelisten als Membervariable beinhaltet. Der Builder stellt aus diesem Grund auch eine Funktion `buildPropellerBladeSkinGeometry()` bereit (siehe Abb. 18), die ein Objekt der Klasse `PropellerBladeSkinGeometry` erzeugt und diesem eine Geometrie übergibt, die durch die Funktion `buildGeometry()` erstellt wurde. Die Klasse `PropellerBladeSkinGeometry` erbt von der Klasse `AbstractPropellerBladeGeometry`.

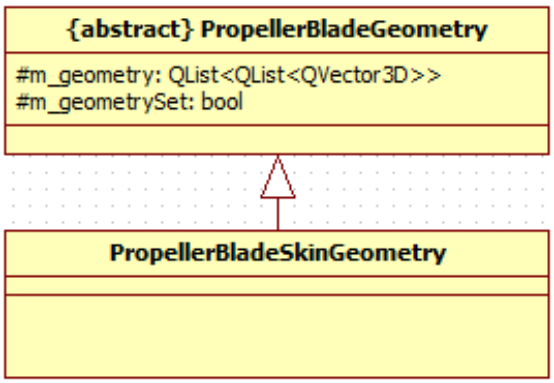


Abbildung 23: Klassendiagramm der Klassen AbstractPropellerBladeGeometry und PropellerBladeSkinGeometry

Zuvor wurde in diesem Kapitel bereits erwähnt, dass eine Erweiterung in Betracht gezogen werden könnte, um innere Strukturen einer Propellerschaufel darstellen zu können. Um für solche Erweiterungen auch entsprechende Datenstrukturen bereitstellen zu können, wurde die Klasse `AbstractPropellerBladeGeometry` implementiert, welche eine Variable zur Beschreibung der entsprechenden Geometrie (`m_geometry`) definiert, die durch die Funktion `buildGeometry()` eines vorgesehenen konkreten Builders erzeugt wird. Da diese Variable als protected definiert wurde, können erbende Klassen auch auf diese zugreifen. Dies hat bei Erweiterungen in Form von weiteren Klassen, die als Datenstruktur für Propellerblatt-Geometrien dienen sollen, einen Vorteil. Solche Klassen erfüllen die Aufgabe, die geometrischen Informationen in Form von geordneten Punktelisten abzuspeichern und bereitzustellen. Bei einer Erweiterung um eine solche Klasse muss nicht darauf geachtet werden, eine Variable zur Speicherung dieser Listen zu implementieren, da diese durch die Vererbung bereits vorliegt. Abbildung 23 zeigt das Klassendiagramm der Klassen, die als Datenstruktur dienen, um die aufbereiteten geometrischen Informationen von Propellerschaufeln bereitzustellen.

Zusammengefasst erzeugt die Schnittstelle `PropellerBladeSkinGeometryBuilder` aus den normierten geometrischen Daten, die von Propster durch ein Objekt der Klasse `Propeller` geliefert werden, die geometrischen Informationen zur Beschreibung des gelieferten Propellers in Originalgröße und Originalform, wenn die richtige Strategie zum Fädeln der Profile vom Benutzer ausgewählt wurde. Die aufbereiteten geometrischen Informationen werden in einem Objekt der Klasse `PropellerBladeSkinGeometry` hinterlegt. Dieses Objekt dient im weiteren Verlauf dazu, die Daten bereitzustellen, um die Mantelfläche der Propeller-

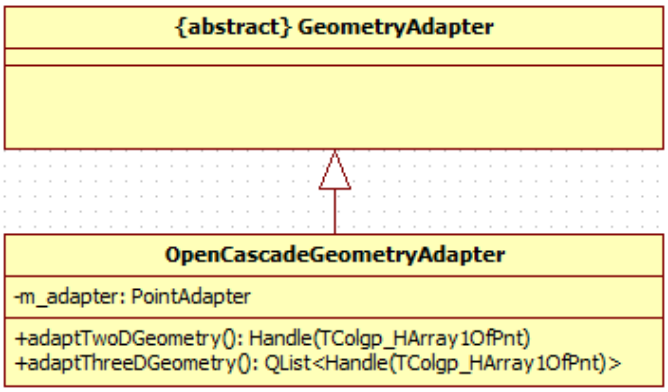


Abbildung 24: Klassendiagramm der Klassen AbstractGeometryAdapter und OpenCascadeGeometryAdapter

schaufel aus diesen zu generieren. Um aus den Daten, welche in einem Objekt der Klasse `PropellerBladeSkinGeometry` hinterlegt werden, analog zu den Algorithmen aus Kapitel 5.1.4 eine Mantelfläche generieren zu können, müssen diese Daten zunächst in ein Datenformat überführt werden, welches kompatibel zu Open Cascade Technology ist. Diese Aufgabe übernimmt die Klasse `OpenCascadeGeometryAdapter`. Das Klassendiagramm ist in Abbildung 24 hinterlegt.

Die Klasse `OpenCascadeGeometryAdapter` stellt zwei Funktionen zur Verfügung, mit denen eine oder mehrere geordnete Punktelisten in entsprechende Arrays des Open Cascade Technology Datenformates (`TColgp_HArray1OfPnt`) übertragen werden können. Die Punktelisten, welche von der Datenstruktur `PropellerBladeSkinGeometry` stammen, beinhalten Punkte der Klasse `QVector3D`. Diese Struktur stammt aus der Bibliothek Qt, welche größtenteils Klassen und Funktionen für die GUI-Programmierung zur Verfügung stellt. Allerdings bietet diese Bibliothek auch einige Containerklassen zur Datenhaltung an, die im Rahmen dieser Arbeit verwendet werden. Dazu gehören unter anderem Listen(`QList`), Vektoren und Arrays(`QVector`) sowie Punkte (`QPointF` (zweidimensional) und `QVector3D` (dreidimensional)). Dreidimensionale Punkte der Bibliothek Open Cascade Technology werden durch die Klasse `gp_Pnt` repräsentiert. Somit werden in dieser Arbeit drei verschiedene Klassen genutzt, die alle die Aufgabe haben, einen Punkt darzustellen. Um die Formate ineinander überführen zu können, wurde die Klasse `PointAdapter` implementiert. Diese besitzt Funktionen, um einen Punkt oder eine geordnete Liste eines der drei Datenformate in einen Punkt oder eine Punkteliste eines anderen Datenformats zu adaptieren. Abbildung 25 zeigt das Klassendiagramm

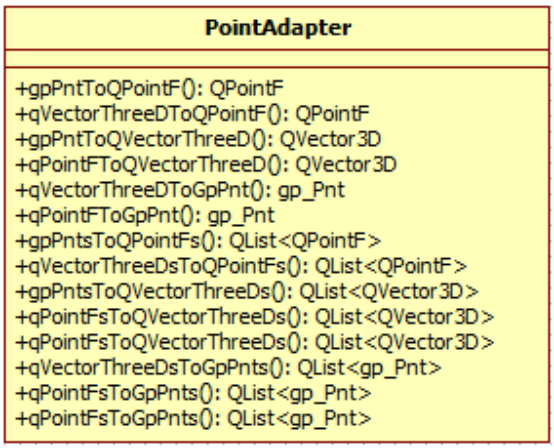


Abbildung 25: Klassendiagramm der Klasse PointAdapter

der Klasse `PointAdapter`.

Den Funktionen werden Punkte oder Punktelisten des ursprünglichen Datenformats übergeben. Soll das Datenformat eines Punktes im zweidimensionalen Raum (`QPointF`) in das Format eines Punktes im dreidimensionalen Raum überführt werden, so ist der Funktion zusätzlich die Fehlende dritte Koordniate z als Parameter zu übergeben. Die Klasse `OpenCascadeGeometryAdapter` nutzt die Funktionalität der Klasse `PointAdapter`, um die gelieferten Punkte (`QVector3D`) der Klasse `PropellerBladeSkinGeometry` in das Open Cascade Format (`gp_Pnt`) zu überführen. Ein Objekt der Klasse `PointAdapter` ist deshalb als Membervariable (`m_adapter`) in der Klasse `OpenCascadeGeometryAdapter` hinterlegt. Sowohl die Klasse `OpenCascadeGeometryAdapter`, als auch die Klasse `PointAdapter` basieren auf dem Prinzip des Adapter-Strukturmusters. Durch das Adapter-Strukturmuster ist es möglich, zwei inkompatible Schnittstellen über ein definiertes Adapter-Interface kompatibel zu gestalten. In diesem Anwendungsfall liegen verschiedene Datentypen zur Beschreibung von Punkten vor. Die Datentypen von Qt, mit denen Propster arbeitet, wurden größtenteils für die Implementierung der Algorithmen, die im Rahmen dieser Arbeit erstellt wurden, genutzt. Diese sind jedoch inkompatibel zu den Funktionen von Open Cascade Technology, die die Datenstruktur `gp_Pnt` nutzt. Das Interface `PointAdapter` beziehungsweise `OpenCascadeGeometryAdapter` ermöglicht eine Überführung des einen Datenformates in das Andere. Ein Nachteil des Adapter-Strukturmusters ist, dass dem Benutzer die Adaption der Daten verborgen bleibt und daraus resultierende Performanceeinbußen für den Benutzer

unerklärlich sein könnten [25]. Die Datenmengen, die mit Hilfe der vorgestellten Klassen adaptiert werden, sind jedoch gering, sodass der Performanceverlust für den Benutzer unauffällig bleibt. Um die Erweiterbarkeit der Bibliothek dahingehend zu gewährleisten, dass zukünftig auch andere Bibliotheken genutzt werden könnten, um die Mantelfläche einer Propellerschaufel zu generieren, wurde eine abstrakte Klasse `AbstractGeometyAdapter` implementiert. Von `AbstractGeometryAdapter` erbende Klassen erfüllen die Aufgabe, die aufbereiteten geometrischen Daten einer Propellerschaufel, welche ein Objekt der Klasse `PropellerBladeSkinGeometry` liefert, in ein gewünschtes Format zu überführen, welches kompatibel zu der jeweiligen Bibliothek ist, die zur Mantelflächengenerierung genutzt werden soll. Da sich diese Formate in Abhängigkeit der Bibliothek unterscheiden, kann jedoch keine abstrakte Funktion innerhalb der Klasse `AbstractGeometryAdapter` definiert werden, welche von erbenden Klassen zu überschreiben wäre. Außerdem ist zu diesem Zeitpunkt nicht bekannt, welche Bibliotheken in Zukunft in Betracht gezogen werden könnten, um Mantelflächen aus den vorhandenen geometrischen Informationen einer Propellerschaufel zu generieren. Aus diesem Grund besitzt die Klasse `AbstractGeometryAdapter` lediglich einen Konstruktor und einen virtuellen Destruktor. Prinzipiell ist diese Klasse nutzlos. Ein imaginärer Nutzen dieser Klasse besteht jedoch darin, dass durch diese Klasse eine Abstraktion geschaffen wird. Sollte zukünftig eine weitere Bibliothek zur Generierung von Mantelflächen genutzt werden, so wird die Adaption der geometrischen Daten durch die Funktionalität einer Klasse geschehen, die von `AbstractGeometryAdapter` erbt. Somit liegen alle Klassen zur Adaption der aufbereiteten geometrischen Daten in ein kompatibles Format der genutzten Bibliotheken zur Generierung von Mantelflächen durch die Vererbung der Klasse `AbstractGeometryAdapter` gebündelt vor. Momentan liegt mit der Klasse `OpenCascadeGeometryAdapter` nur eine Klasse vor, die von `AbstractGeometryAdapter` erbt. Aus den adaptierten Daten kann nun die Mantelfläche in Form der Open Cascade Technology Datenstruktur `TopoDS_Shape` generiert werden. Diese Aufgabe übernimmt die Klasse `PropellerBladeSkinShapeBuilder`. Diese Klasse erbt von der Klasse `AbstractPropellerShapeBuilder`. Abbildung 26 zeigt das entsprechende Klassendiagramm.

Die Klasse `AbstractPropellerShapeBuilder` definiert die abstrakte Funktion `buildShape ()`, die von erbenden Klassen zu überschreiben ist. Dieser Funktion werden die adaptierten geometrischen Daten als Parameter übergeben. Sie liefert eine Geometrie als `TopoDS_Shape` zurück. Die abgeleitete Klasse `PropellerBladeSkinShapeBuilder` beinhaltet neben der zu überschreibenden Funktion noch zwei private Funktionen. Die privaten Funktionen er-

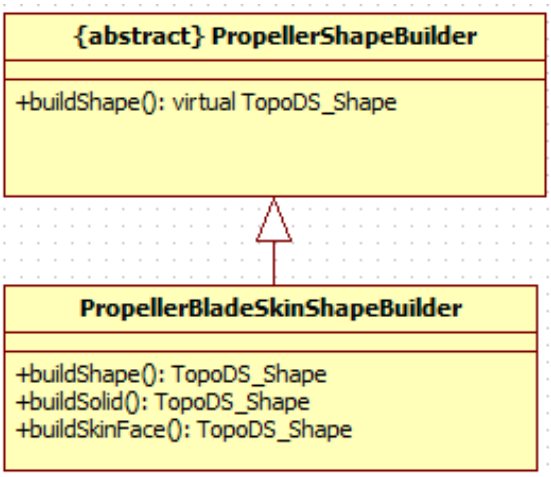


Abbildung 26: Klassendiagramm der Klassen AbstractPropellerShapeBuilder und PropellerBladeSkinShapeBuilder

zeugen aus den adaptierten geometrischen Daten einen Festkörper (`TopoDS_Solid`) oder eine Mantelfläche der Propellerschaufel (`TopoDS_Face`). Die überschriebene Funktion nutzt die Funktion `buildSolid()`. Die Vorteile eines Festkörpers gegenüber der Mantelfläche wurden bereits in Kapitel 5.1.4 beschrieben. In diesem Fall dient die Abstraktion ebenfalls der Erweiterbarkeit. Als Beispiel für eine Erweiterung könnte wieder der Anwendungsfall zur Generierung der inneren Strukturen von Propellerschaufeln dienen. Ein konkreter Builder für diesen Anwendungsfall könnte ohne Probleme zu der vorhandenen Struktur hinzugefügt werden. Die Klassen aus Abbildung 26 wurden nach dem Builder-Erzeugungsmuster implementiert. Auch in diesem Klassendiagramm wurde der Director bewusst nicht dargestellt. Dieser wurde jedoch implementiert. Abschließend sollen nun noch die Klassen vorgestellt werden, die aus der generierten Geometrie `TopoDS_Shape` Dateien in CAD-Outputformaten erzeugen. Um auch die Schnittstelle für die Ausgabedateien erweiterbar und modular zu gestalten, wurde mit `AbstractShapeExporter` eine weitere abstrakte Klasse definiert. Implementiert wurde der Export eines Objekts der Klasse `TopoDS_Shape` in die Datenformate STEP und IGES. Somit stehen mit `STEPShapeExporter` und `IGESShapeExporter` zwei erbende Klassen von `AbstractShapeExporter` zur Verfügung. Die Klasse `AbstractShapeExporter` definiert eine virtuelle Funktion `exportFile()`, der als Parameter das Objekt der Klasse `TopoDS_Shape` und ein String übergeben werden muss, der den Pfad beinhaltet, an dem die Datei generiert werden soll. Abbildung 27 zeigt das Klassendiagramm der Export-Klassen.

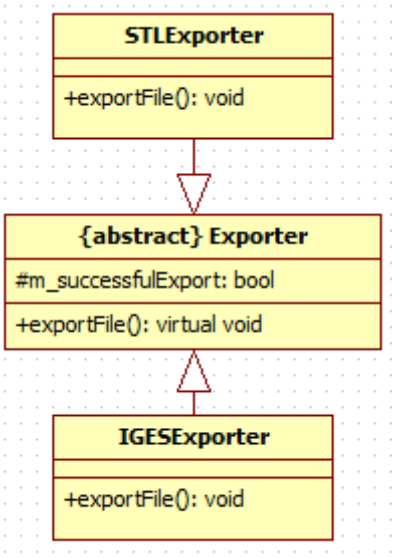


Abbildung 27: Klassendiagramm der Exporter-Klassen

Neben der Funktion `exportFile()` definiert die Klasse `AbstractExporter` noch eine Variable (`m_successfulExport`) des Datentyps boolean, die auf true gesetzt wird, wenn der letzte Export erfolgreich ausgeführt wurde. Andernfalls ist diese auf false gesetzt. Die Variable ist für erbende Klassen ebenfalls verfügbar.

Die Klassenstruktur der implementierten Bibliothek und die damit verbundenen Algorithmen wurden damit vollständig beschrieben. Im Anhang A.2 repräsentiert ein Flussdiagramm den Programmablauf für die Erzeugung der Mantelfläche einer Propellerschaufel. Die roten Pfade stellen den erfolgreichen Durchlauf bei der Generierung dar. Die in Kapitel 5.1 betrachteten Algorithmen werden von den Prozessen verwendet, die in dem Flussdiagramm dargestellt werden. Auch einige vorgestellte Klassen dieses Abschnittes sind in dem Flussdiagramm zu finden. Nachdem die Klassenstruktur vollständig beschrieben wurde, sollen nun die Tests des implementierten Verfahrens zur Generierung von Mantelflächen genauer betrachtet werden.

6 Tests in Bezug zum implementierten Verfahren

In dem folgenden Kapitel sollen die ausgeführten Tests der implementierten Softwarebibliothek vorgestellt werden. Für die Bibliothek wurde eine Unittestsuite eingerichtet, die die Funktionalität und Robustheit der Funktionen aller implementierten Klassen prüft. Weiterhin wurden diverse Anwendungstests ausgeführt, in deren Rahmen die implementierten Funktionen mit den geometrischen Informationen der Propeller SRII und SRIII der NASA geprüft wurden. Zunächst soll die eingerichtete Unittestsuite betrachtet werden.

6.1 Unit Tests

Mit Hilfe des Testing Frameworks Google Test wurden die implementierten Klassen und deren Funktionen mit Unittests versehen. Insgesamt wurden 40 Tests in 18 Test Cases implementiert. Jede implementierte Klasse beinhaltet ein eigenes Test Case. Ausgenommen sind einige abstrakte Klassen, die lediglich dazu dienen abstrakte Funktionen, die von erbenden Klassen zu überschreiben sind, zu definieren. Ein wesentliches Testziel bestand darin, die Funktionen auf deren Robustheit gegenüber fehlerhaften oder unerwarteten Parametereingaben zu prüfen. Im Rahmen der Implementierung dieser Tests wurden für einige Parameter Definitionsbereiche festgelegt, da Parametereingaben außerhalb dieser Bereiche zu fehlerhaften Ausführungen der Algorithmen oder sogar zu einem Programmabsturz führen. Damit es bei Parametereingaben außerhalb des Definitionsbereiches nicht zu einem Programmabsturz kommt, werden diese in entsprechenden Funktionen zunächst geprüft. Liegt ein Parameter nicht in dem definierten Bereich, so wird die Funktion nicht standardmäßig ausgeführt. Stattdessen wird der Benutzer über die Konsole darüber informiert, welche Parametereingabe fehlerhaft war und was von der Funktion als Wert zurückgeliefert wird. Weiterhin wurden einige Tests geschrieben, die Funktionen auf deren korrekte Funktionalität prüfen. Im Rahmen dieser Tests konnten mögliche Fehlerquellen entdeckt und behoben werden. Auch ergaben sich durch diese Tests Definitionsbereiche für Parameter, die zuvor anders eingeschätzt wurden. Zusammengefasst kann festgehalten werden, dass durch die Implementierung von Unittests kleinere Fehlerquellen im Code identifiziert und behoben werden konnten. Die Tests werden allesamt erfolgreich durchlaufen, sodass die korrekte Funktionalität und die Robustheit aller implementierten

Funktionen gewährleistet werden kann.

6.2 Anwendungstests

Neben den implementierten Unittests wurden auch einige Anwendungstests ausgeführt, die zur Validierung der implementierten Bibliothek zur Generierung der Mantelfläche einer Propellerschaufel und deren Funktionen dienen sollen. In diesem Abschnitt sollen die ausgeführten Anwendungstests beschrieben und diskutiert werden. Als Anwendungsfall für diese Tests dienen die Propeller SRII und SRIII der NASA, zu deren Schaufeln detaillierte Beschreibungen der Geometrie in der Parametrisierung, die im Kapitel 2 vorgestellt wurde, veröffentlicht wurden. Im Rahmen dieser Arbeit wurden vier Testreihen durchgeführt. Die erste Testreihe befasst sich mit der Generierung der Auffädellinie nach dem Algorithmus, der in Kapitel 5.1.3.1 beschrieben wurde. Da die Schaufeln des SRII-Propellers keine Pfeilung und Neigung aufweisen, wurde für diese Testreihe lediglich der SRIII-Propeller als Anwendungsfall ausgewählt [6]. Die zweite Testreihe befasst sich mit der Hauptaufgabe der implementierten Bibliothek. In diesen Tests werden die Mantelflächen des SRII- und SRIII-Propellers generiert und die Ergebnisse diskutiert. Die dritte Testreihe befasst sich damit, die verschiedenen Algorithmen zum Fädeln der Profile auf deren Robustheit zu prüfen. Die Profile der Anwendungsfälle SRII und SRIII wurden in dieser Testreihe nach allen implementierten Varianten des Fädelns angeordnet, um zu testen ob die Routinen die Robustheit aufweisen Profile zu fädeln, obwohl diese nicht für entsprechende Varianten des Fädelns vorgesehen sind. Weiterhin sollte mit dieser Testreihe festgestellt werden, ob die resultierenden Daten zur Generierung von Mantelflächen genutzt werden können. Die Robustheit der Funktionen zur Mantelflächengenerierung wurde somit ebenfalls getestet. In einer letzten Testreihe wurden die Algorithmen abgeglichen, die dazu dienten Profil Geometrien des SRII- und SRIII-Propellers zu generieren, die als Inputdaten für die Experiment Files dienen. Zunächst soll die Testreihe beschrieben werden, die sich mit der Generierung der Auffädellinie befasst.

6.2.1 Generierung der Auffädellinie des SRIII-Propellers

In der Testreihe, die sich mit der Generierung von Auffädellinien nach dem Algorithmus, der im Kapitel 5.1.3.1 beschrieben wird, befasst, soll ermittelt werden, wie viele Iterationen notwendig sind, um eine normierte Auffädellinie möglichst genau zu rekonstruieren. Der Algorithmus ist derart konzipiert, dass eine größere Anzahl von Iterationen zu einer genaueren

Rekonstruktion der Auffädellinie führt. Gleichzeitig erfordert eine größere Anzahl an Iterationen auch mehr Rechenzeit, die möglichst gering ausfallen soll. Aus diesem Grund wurden innerhalb dieser Testreihe Auffädellinien mit Hilfe des Algorithmus aus Kapitel 5.1.3.1 generiert, die sich lediglich durch eine unterschiedliche Anzahl an Iterationen unterscheiden. Die anderen Parameter unterscheiden sich innerhalb dieser Testreihe nicht. Als Anwendungsfall dient der SRIII-Propeller der NASA. Der Algorithmus zur Generierung der Auffädellinie benötigt neben der Anzahl an Iterationen weitere Parameter, die zu übergeben sind. So benötigt er eine geordnete Punkteliste, die den Pfeilwinkel (y-Koordinaten der Punkte) des Propellers in einem Funktionalen Zusammenhang mit dem dimensionslosen Radius (x-Koordinaten) beschreibt. Außerdem wird der dimensionslose Hubradius des entsprechenden Propellers benötigt. Bevor die Auffädellinien erzeugt werden konnten, mussten diese Parameter ermittelt werden. Dieser Vorgang soll zunächst beschrieben werden. Anschließend erfolgen die Darstellung der generierten Auffädellinien und eine Diskussion der Ergebnisse. Der Funktionale Zusammenhang zwischen dem Pfeilwinkel und dem dimensionslosen Radius des Propellers konnte aus der Quelle [26] entnommen werden. Abbildung 28 stellt den Funktionalen Zusammenhang aus dieser Quelle dar.

In Abbildung 28 sind mit dem aerodynamischen und dem gefertigten Pfeilwinkel zwei Parameter in Abhängigkeit des dimensionslosen Radius aufgetragen. Die beiden Pfeilwinkel unterscheiden sich dadurch, dass der gefertigte Pfeilwinkel die Pfeilung des Propellerblattes im Ruhezustand beschreibt. Der aerodynamische Pfeilwinkel definiert hingegen die Pfeilung des Propellerblattes, während dieses mit der standartmäßigen Drehzahl im Betrieb rotiert. Die Propellerschaufel verformt sich somit während des Fluges, sodass der Propeller im Ruhezustand und im Betrieb differenziert betrachtet werden muss. Im Kapitel 2.3 wurde eine Zeichnung des Propellerblattes vom SRIII-Propeller inklusive dessen Auffädellinie in Abbildung 6 dargestellt. Die Zeichnung liegt als Parallelprojektion vor. Die generierten Auffädellinien dieser Testreihe sollen mit der Auffädellinie aus Abbildung 6 abgeglichen werden. Da in der Quelle der Zeichnung nicht eindeutig hervorging ob die Zeichnung die Propellerschaufel und deren Auffädellinie mit aerodynamischer oder gefertigter Pfeilung darstellt, wurde zunächst für beide funktionale Abhängigkeiten aus Abbildung 28 eine Testreihe ausgeführt. Um dieses Kapitel kompakt zu halten, soll bereits an dieser Stelle erwähnt werden, dass sich aus den Testergebnissen ergab, dass die Propellerschaufel des SRIII-Propellers in der Zeichnung mit gefertigter Pfeilung abgebildet wird. Deshalb wird in diesem Kapitel auch nur die Testreihe mit Berücksichtigung der gefertigten Pfeilung betrachtet. Um an die Daten des gefertigten

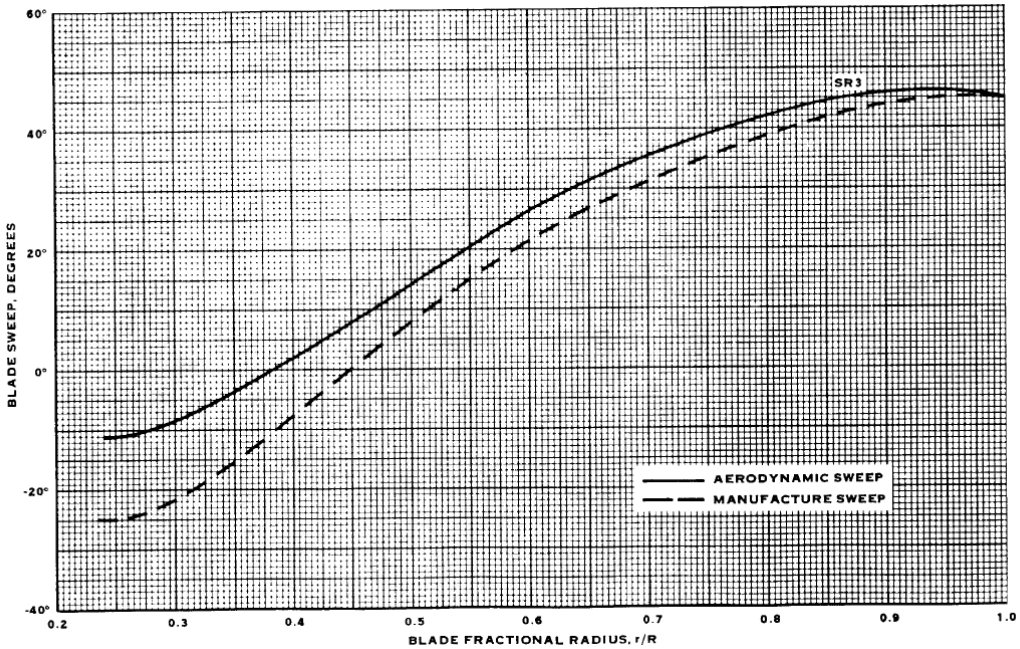


Abbildung 28: Aerodynamischer und gefertigter Pfeilwinkel in Abhängigkeit des dimensionslosen Radius beim SRIII-Propeller [26]

Pfeilwinkels in Abhängigkeit des dimensionslosen Radius mit einem möglichst geringen Ablesefehler zu gelangen, wurden diese mit Hilfe des Open Source Programms Engauge Digitizer digitalisiert und anschließend für die Testreihe eingelesen. Der dimensionslose Hubradius des SRIII-Propellers wurde in der Quelle [27] mit einem Wert von 0.239 beschrieben. Somit wurden alle benötigten Parameter zur Generierung der Auffädellinie des SRIII-Propellers ermittelt, sodass die Generierung der Auffädellinien für die Testreihe ausgeführt werden konnte. In der Testreihe wurden Auffädellinien in 10 bis 100.000 Iterationen generiert. Abbildung 29 zeigt die generierten Auffädellinien des SRIII-Propellers, die mit 10, 100, 1000 und 10000 Iterationen erzeugt wurden. Aus der Abbildung geht lediglich durch Farbunterschiede und verschiedene Linienstile hervor, dass innerhalb dieser vier Auffädellinien dargestellt werden. Eindeutig zu erkennen ist lediglich, dass der Verlauf der grauen Auffädellinie links, die mit 10 Iterationen generiert wurde, sich von den anderen drei Verläufen deutlich unterscheidet. Dass die Verläufe der anderen drei Auffädellinien geringfügig voneinander abweichen, wird erst durch eine Vergrößerung eindeutig ersichtlich.

Bevor die Abweichungen zwischen den einzelnen Auffädellinien diskutiert werden, sollen diese mit der Auffädellinie der Zeichnung aus Abbildung 6 verglichen werden. Abbildung 30 zeigt den Vergleich zwischen der Zeichnung und den vier Auffädellinien, die mit 10, 100, 1000 und 10000 Iterationen generiert wurden.

Der Vergleich zeigt auf, dass bereits bei der Generierung einer Auffädellinie mit 1000 Iterationen ohne Vergrößerung kein Unterschied zu der Zeichnung zu erkennen ist. Gleiches gilt für die Auffädellinie, die mit 10000 Iterationsschritten generiert wurde. Lediglich bei den Auffädellinien, die mit 10 und 100 Iterationen erzeugt wurden, ist eine Differenz zur Zeichnung zu erkennen. Es kann somit festgehalten werden, dass diese beiden Auffädellinien eine ungenügende Präzision aufweisen, und somit nicht für die Generierung einer originalgetreuen Mantelfläche des SRIII-Propellers genutzt werden können. Um festzustellen, ob bereits 1000 Iterationen für die Generierung einer präzisen Rekonstruktion der Auffädellinie genügen, reicht ein Vergleich mit der Zeichnung nicht aus. Der Vergleich bestätigte die Annahme, dass mit dem im Kapitel 5.1.3.1 vorgestellten Algorithmus die Rekonstruktion einer Auffädellinie erfolgen kann. Um zu ermitteln, wie viele Iterationen zur präzisen Rekonstruktion der Auffädellinie nötig sind, wurden die Auffädellinien, die mit 1000, 10000 und 100000 Iterationen generiert wurden, noch einmal gesondert betrachtet. An dieser Stelle soll noch einmal erwähnt werden, dass der Algorithmus zur Generation der Auffädellinien mit mehr Iterationsschritten immer ein genaueres Ergebnis liefert. Die Auffädellinie mit 100000 Iterationen liegt somit am nächsten am

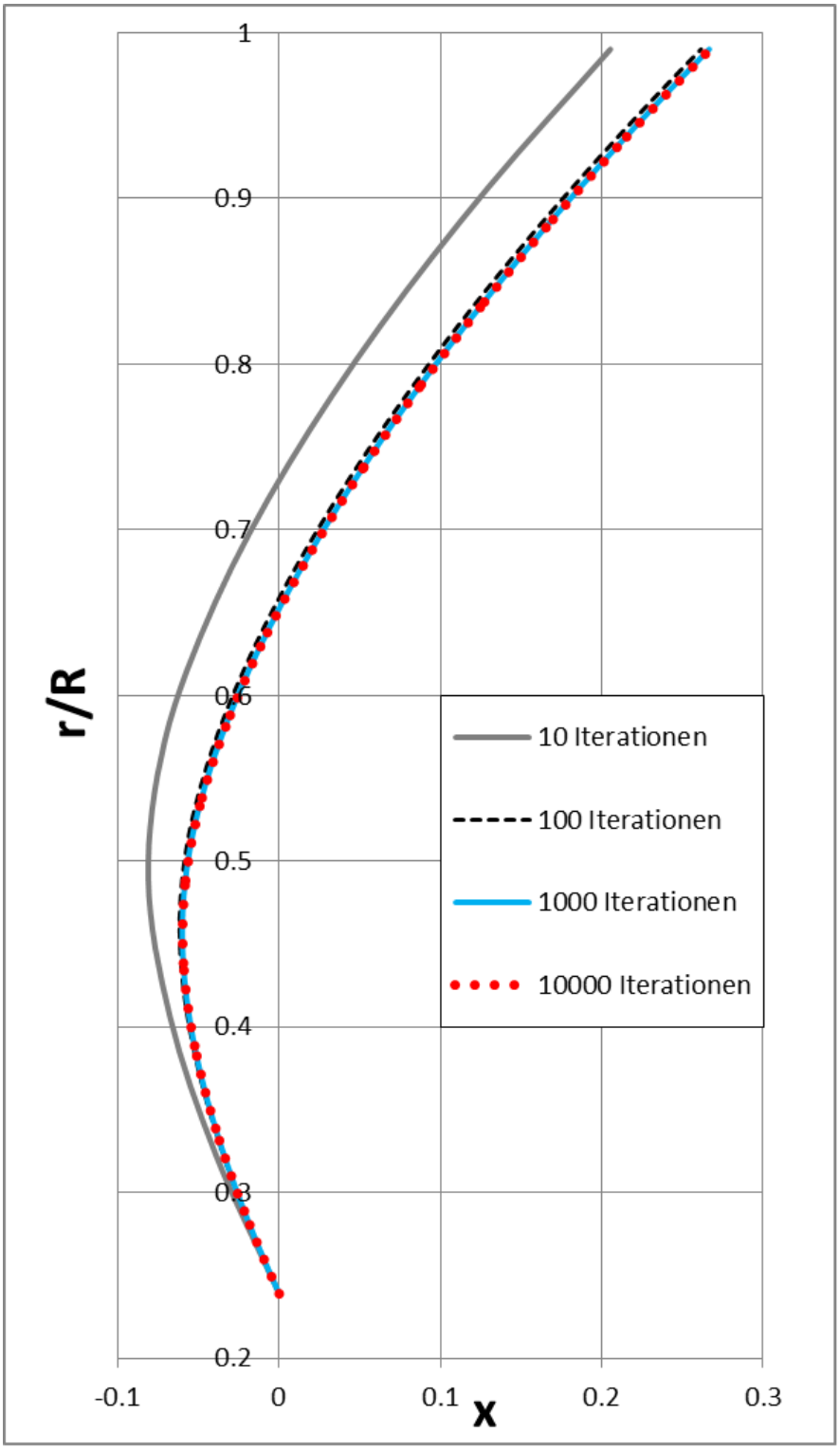


Abbildung 29: Auffädellinien des SRIII-Propellers mit unterschiedlicher Anzahl an Iterationen

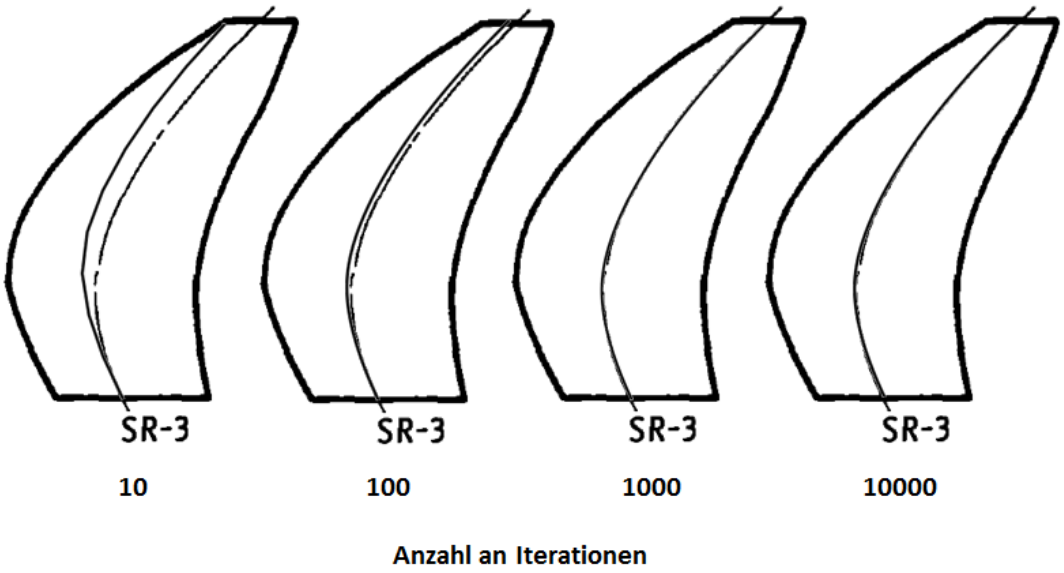


Abbildung 30: Vergleich der Zeichnung der Auffädellinie des SRIII-Propellers mit vier generierten Auffädellinien mit einer unterschiedlichen Anzahl an Iterationen

Original. Da innerhalb intensiver Literaturrecherchen jedoch keine Beschreibung der originalen Auffädellinie aufzufinden war, wurde für die weitere Betrachtung die generierte Auffädellinie mit 100000 Iterationen als Referenz gewählt. Die Abweichungen zum Original wurden als zu vernachlässigen eingeschätzt. Diese Einschätzung sollte sich nach der weiteren Betrachtung der Auffädellinien bestätigen. Die Auffädellinien mit 1000 und 10000 Iterationen wurden in einem nächsten Schritt auf deren Abweichungen mit der Auffädellinie, die in 100000 Iterationen erzeugt wurde, verglichen. Der maximale Abstand ergibt sich bei diesen Auffädellinien immer am dimensionslosen Radius von 1,0 (Tip). Betrachtet wird der Abstand in x-Richtung an gleichen dimensionslosen radialen Koordinaten (y-Koordinaten). Eine maximale Abweichung von 0,01% wird toleriert und gilt als genügend präzise. Da es sich bei den generierten Kurven um normierte Auffädellinien mit einem Radius von eins handelt, lassen sich die Abstände in x-Richtung zwischen den Auffädellinien mit wenig Aufwand in die prozentuale Abweichung umrechnen. Die maximale Differenz zwischen der Auffädellinie mit 1000 Iterationen lag bei 0,055% und somit oberhalb der Toleranz. Mit einer maximalen Abweichung von 0,005% wurde die Auffädellinie mit 10000 Iterationsschritten als ausreichend präzise eingestuft. Durch die geringe Abweichung zwischen den beiden Auffädellinien, die mit 10000 und 100000 Iterationen generiert wurden, lässt sich folgern, dass durch eine Generierung der Auffädellinie mit mehr als

100000 Iterationen keine deutliche Präzisionssteigerung gewonnen werden könnte. Dadurch bestätigt sich auch die Annahme, dass die Auffädellinie, die mit 100000 Iterationen erzeugt wurde, zu vernachlässigende Abweichungen zu der Originalkurve aufweist. Zusammengefasst führt eine Generierung der Auffädellinie mit 10.000 Iterationen zu einem Ergebnis, welches den Anforderungen genügt. Mit dieser Kenntnis lassen sich die Auffädellinien nun präzise genug generieren, ohne dabei mehr Iterationsschritte als nötig ausführen zu müssen. Weiterhin ergab sich aus der Übereinstimmung der generierten Auffädellinien mit der Zeichnung aus Abbildung 6, dass der Algorithmus aus Kapitel 5.1.3.1 dazu geeignet ist, Auffädellinien präzise zu rekonstruieren.

6.2.2 Generierung der Mantelflächen von Schaufeln des SRII- und SRIII-Propellers

In diesem Kapitel soll die Testreihe zur Generierung von Mantelflächen mit Hilfe der im Rahmen dieser Arbeit implementierten Bibliothek beschrieben werden. In dieser Testreihe soll die Bibliothek, so eingesetzt werden, wie es in Zukunft geplant ist. Dazu gehört, dass mit Hilfe der Experiment Files von Propster, die in Kapitel 3.3 beschrieben wurden, die Schnittstelle zu den geometrischen Daten mit den benötigten Parametern zur Mantelgenerierung gefüllt wird und mit diesen anschließend durch die Bibliothek eine Mantelfläche erzeugt wird. Als Anwendungsfall dienen die Propeller SRII und SRIII, die in Quellen der NASA mit der Parametrisierung, die in dieser Arbeit genutzt wird, um Propeller-Geometrien zu definieren (siehe Kapitel 2), beschrieben werden. In der Testreihe sollen die generierten Mantelflächen mit den originalen Propellerschaufeln abgeglichen werden.

6.2.2.1 Prozesskette der Generierung

Die geometrischen Daten der Propeller sollen über die Schnittstelle von Propster an die Bibliothek übergeben werden. Diese Daten mussten für die Testreihe zunächst über Experiment Files bereitgestellt werden. Wie die Experiment Files mit Daten gefüllt wurden, um die Schnittstelle von Propster mit den geometrischen Daten zu versorgen, soll nun beschrieben werden. Die Experiment Files müssen mit einigen Parametern der Propeller gefüllt werden, damit aus diesen ein Objekt der Klasse `Propeller` erzeugt werden kann. Zum einen müssen funktionale Abhängigkeiten der Sehnenlänge, des Anstellwinkels und des Pfeilwinkels in einem Geometry File definiert werden. Diese Parameter sind in Abhängigkeit des dimensionslosen

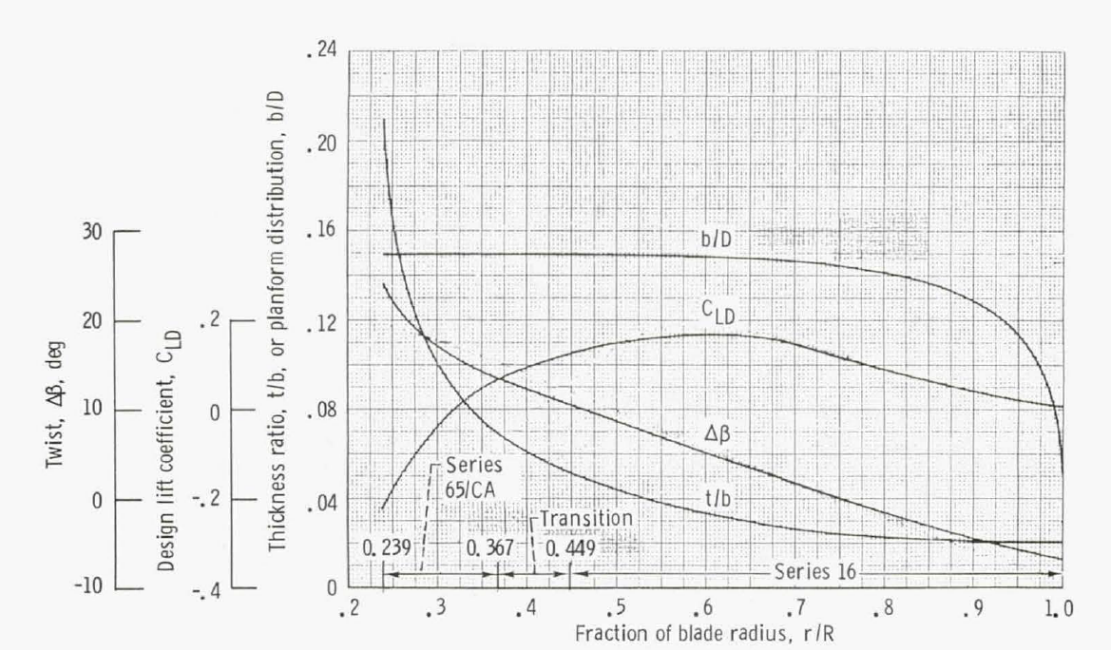


Abbildung 31: Designparameter des SRII-Propellers in funktionaler Abhängigkeit des dimensionslosen Radius [27]

Radius zwischen Hub und Tip anzugeben. Die Informationen zu diesen Parametern sind den Abbildungen 31 und 32 für die beiden Propeller zu entnehmen.

Die Abbildungen stellen vier Design Parameter der Propeller in funktionaler Abhängigkeit dar.

1. Verhältnis der Sehnenlänge b zum Propellerdurchmesser D
2. Design Lift Koeffizient CLD
3. Anstellwinkel β
4. Verhältnis der maximalen Dicke eines Profils t zur Sehnenlänge b

Die Abbildungen sind auch in weiteren Quellen der NASA hinterlegt [28] [29] und deren Daten wurden analog zu den Daten der Pfeilwinkel aus Abbildung 28 mit Hilfe von Engauge Digitizer digitalisiert. Die Daten des Anstellwinkels konnten durch die digitalisierten Daten sofort in das Geometry File integriert werden. Die Daten des Verhältnisses der Sehnenlänge zum Propellerdurchmesser mussten mit dem Durchmesser des Propellers multipliziert werden und konnten anschließend in das Geometry File eingefügt werden. Der Durchmesser des SRII-Propellers ist mit dem Durchmesser des SRIII-Propellers identisch und wird mit 0.622

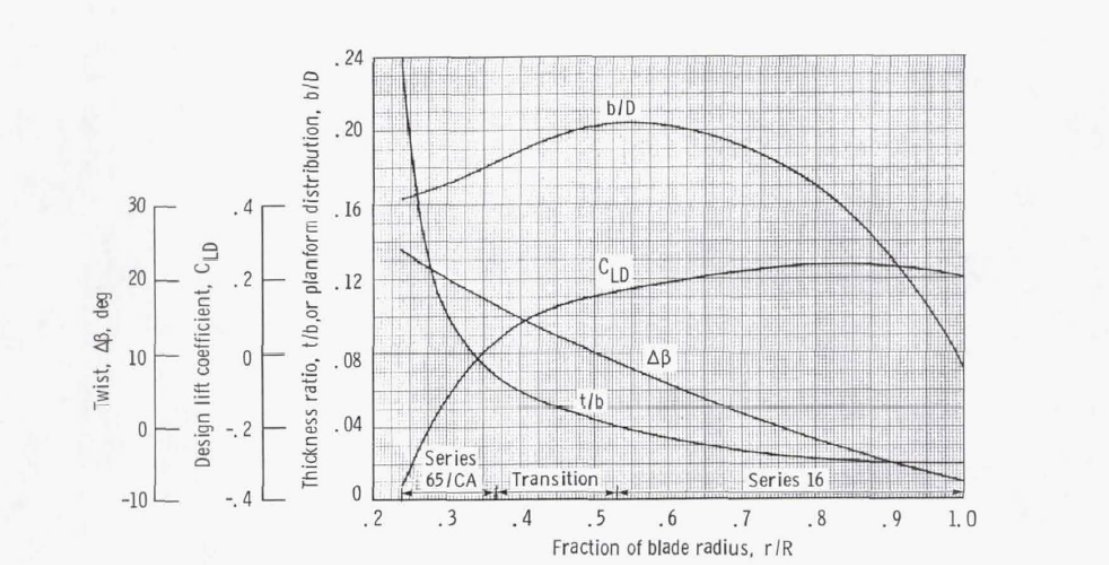


Abbildung 32: Designparameter des SRIII-Propellers in funktionaler Abhängigkeit des dimensionslosen Radius [26]

m angegeben [27] [30]. Um die Daten des Pfeilwinkels in das jeweilige Geometry File zu integrieren, wurden für den SRIII-Propeller die digitalisierten Daten aus Abbildung 28 genutzt. Da zwischen dem aerodynamischen und dem gefertigten Pfeilwinkel unterschieden werden muss, wurden zwei Geometry Files angelegt, die sich nur in den Daten der Pfeilwinkel unterscheiden. So konnte im Rahmen dieser Testreihe die Generierung der Mantelfläche der Propellerschaufel erfolgen, die die Schaufel in ruhendem und rotierendem Zustand darstellt. Der SRII-Propeller besitzt keine Pfeilung, sodass sich der Pfeilwinkel über den gesamten Verlauf des dimensionslosen Radius mit Null erstreckt. Neben dem Geometry File müssen für die Experiment Files der Propeller weitere Daten bereitgestellt werden. Dazu gehört eine Anzahl an normierten Profil-Geometrien, die durch eine feste Anzahl an geordneten Punkten beschrieben werden. Zwischen diesen Profil-Geometrien wird durch eine Funktion von Propster interpoliert, um weitere Profile zur Beschreibung des Propellers zu erhalten. Wie viele Profile den Propeller nach der Interpolation beschreiben, kann in dem Experiment File festgelegt werden. Die Profile liegen nach der Interpolation in gleichmäßigen radialen Abständen vor. Für eine präzise Interpolation ist es ausreichend zwischen 5 und 10 Profil-Geometrien im Experiment File vorzugeben. Aus den Abbildungen 31 und 32 kann man entnehmen, dass für die Propeller SRII und SRIII Profile der Serie NACA 16 und NACA

65/CA genutzt wurden. In [29] und [30] wird dies bestätigt. Diesen Quellen und Abbildung 31 ist zu entnehmen, in welchen Bereichen des dimensionslosen Radius welche Profilserie vorliegt. Mit Hilfe der Designparameter aus den Abbildungen 31 und 32 ist es möglich, die normierten Profil-Geometrien in Anlehnung an Kapitel 2.2 zu generieren. Im Folgenden soll erläutert werden, wie die Profil Geometrien der NACA 16 und NACA 65/CA Serie nach dem Algorithmus aus Kapitel 2.2 erzeugt wurden. Zunächst soll die NACA 16 Profilserie betrachtet werden. Die benötigten Informationen zur Generierung der Profil-Geometrien der NACA 16 Serie werden in [31] beschrieben. In dieser Quelle werden Gleichungen angegeben, mit denen sich die Dickenverteilung und die Skelettlinie konstruieren lassen. Die Dickenverteilung wird über die Gleichungen 13 und 14 beschrieben.

$$\pm y_1 = \frac{t}{b} * 0,989665{x_1}^{\frac{1}{2}} - 0,23925x_1 - 0,041{x_1}^2 - 0,5594{x_1}^3 \tag{13}$$

$$\pm y_2 = \frac{t}{b} * [0,01 + 2,325\,(1 - x_2) - 3,42\,(1 - x_2)^2 + 1,46\,(1 - x_2)^3] \tag{14}$$

$y_{1/2}$ ist hierbei die Dicke eines Profils ohne Skelettlinie ausgehend von der Stelle der auf die Länge von eins normierten Sehne $x_{1/2}$ in Richtung des Normalenvektors. t/b ist das Dickenverhältnis und entspricht der Hälfte der maximalen Dicke eines Profils geteilt durch die Länge der Sehne des Profils. Das Dickenverhältnis für ein NACA 16 Profil des SRII- und SRIII-Propellers an einer bestimmten Stelle des dimensionslosen Radius des Propellerblattes kann aus einer der vier Kurvenverläufe aus Abbildung 31 beziehungsweise 32 entnommen werden. Zu beachten ist, dass Gleichung 13 für die Dickenverteilung benutzt wird bevor die maximale Dicke aus Richtung der Vorderkante des Profils im Punkt $P_{maxThick}(x_{maxThick}|y_{maxThick})$ erreicht wird ($x_1 \leqslant x_{maxThick}$). Anschließend wird Gleichung 14 für die restliche Dickenverteilung bis zur Hinterkante genutzt ($x_2 \geqslant x_{maxThick}$). Nachdem die Dickenverteilung für die Profil Serie NACA 16 erläutert wurde, kann die Generierung der Skelettlinie für diese Profil Serie betrachtet werden. Der Kurvenverlauf der normierten Skelettlinie der NACA 16 Profil Serie ist durch folgende Gleichung beschrieben.

$$y_c = -0,079577\;cld\;[x_c \ln(x_c) + (1 - x_c)\ln(1 - x_c)] \tag{15}$$

Der entsprechende Design Lift Coefficient CLD kann an der entsprechenden Stelle des dimensionslosen Radius jeweiligen Propellers aus einer der vier Kurvenverläufe aus Abbildung 31 und 32 beziehungsweise deren digitalisierten Daten entnommen werden. Die Ableitung der Gleichung 15 lautet wie folgt.

$$\frac{dy_c}{dx_c} = -0,079577\ cld\ [\ln(x_c) - \ln(1 - x_c)] \tag{16}$$

Die Gleichung 16 wird benötigt um die Koordinaten des Normalenvektors an den Punkten der Skelettlinie aus Gleichung 15 zu bestimmen. Dazu muss der Steigungsvektor um $90°$ nach links gedreht werden. Dies wird erreicht indem die Koordinaten des Steigungsvektors $\vec{S}\begin{pmatrix} 1 \\ \frac{dy_c}{dx_c} \end{pmatrix}$ vertauscht werden und die neue x-Koordinate invertiert wird. Daraus ergibt sich der Normalenvektor $\vec{N}\begin{pmatrix} -\frac{dy_c}{dx_c} \\ 1 \end{pmatrix}$. Mit Hilfe dieser Informationen und dem Algorithmus, der in Kapitel 2.2 vorgestellt wurde, lässt sich ein normiertes Profil der NACA 16 Serie modellieren. Nun soll die Modellierung eines Profils der NACA 65/CA Serie beschrieben werden. Auch nach intensiver Literaturrecherche konnte keine Gleichung gefunden werden, die die Dickenverteilung der NACA 65/CA Serie beschreibt. Allerdings ist die Dickenverteilung der Geometrie eines NACA 65/CA Profils in Abhängigkeit der prozentualen Sehnenlänge durch eine Punktewolke in der Literaturquelle [32] beschrieben (siehe Tab. 2). Abbildung 3 zeigt das daraus resultierende Profil.

% chord	NACA 65 Half thickness (% chord)	Camber line (% chord) CLD = 1.0
0	0	0
0.5	0.772	0.25
0.75	0.932	0.35
1.25	1.169	0.535
2.5	1.574	0.93
5.0	2.177	1.58
7.5	2.647	2.12
10	3.04	2.585
15	3.666	3.365
20	4.143	3.98
25	4.503	4.475
30	4.76	4.86
35	4.924	5.15
40	4.996	5.355
45	4.963	5.475
50	4.812	5.515
55	4.53	5.475
60	4.146	5.355
65	3.682	5.15
70	3.156	4.86
75	2.584	4.475
80	1.987	3.98
85	1.385	3.365
90	0.81	2.585
95	0.306	1.58
100	0	0

Tabelle 2: Beschreibung der Dickenverteilung eines Profils der Serie NACA65/CA und dessen Skelettlinie in Abhängigkeit des prozentualen Anteils der Sehnenlänge [32]

Das Profil aus Abbildung 3 und Tabelle 2 wird als Standard dazu genutzt, um die NACA 65/CA

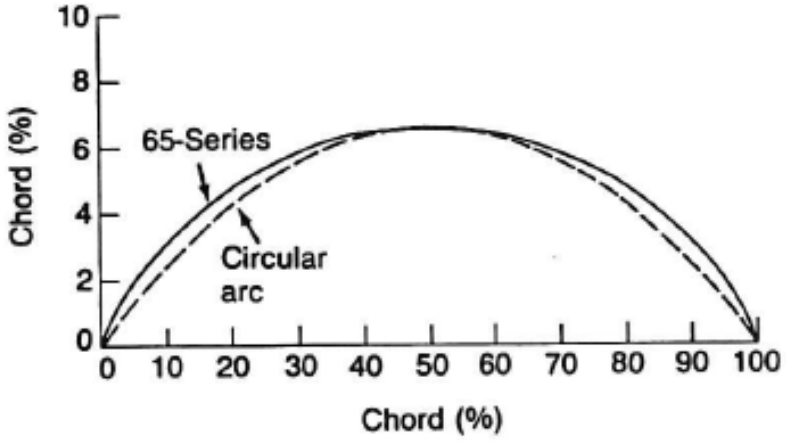


Abbildung 33: Skelettlinien der Profilserien NACA 65 und NACA 65/CA (CLD = 1,0) [32]

Profile zu erzeugen, welche für die Modelle des SRII- und SRIII-Propellers verwendet werden. Dieses auf die Sehnenlänge eins normierte Standardprofil muss lediglich auf die maximale Dicke t skaliert werden. Die maximale Dicke kann durch das Verhältnis t/b, welches auch in Abbildung 31 beziehungsweise 32 durch einen Kurvenverlauf dargestellt wird, ermittelt werden, indem das Verhältnis an der entsprechenden Stelle des dimensionslosen Radius bestimmt und mit der Sehnenlänge b multipliziert wird. Man erhält die maximale Dicke t in Prozent des Profils in Metern. Die maximale Dicke tritt bei Profilen der NACA 65/CA Serie bei 40 % der Sehnenlänge auf [32]. Bei dem Standardprofil beträgt sie 0,04996 Meter (vergleiche Tab. 2). Der Faktor yScal mit dem die y-Koordinaten des Standardprofils somit skaliert werden müssen ergibt sich aus folgender Gleichung.

$$yScal = \frac{0,01 * t}{0,04996} \tag{17}$$

Somit lassen sich die Dickenverteilungen eines beliebigen Profils der Serie NACA 65/CA bestimmen. Die Abkürzung CA in dem Namen der Profilserie steht für Kreisbogen (circular arc). In [32] ist beschrieben, dass die Skelettlinie dieser Profilserie durch einen Kreisbogen beschrieben wird. Abbildung 33 zeigt eine Zeichnung, in der die Skelettlinie der Profilserie NACA 65 mit der Skelettlinie der Profilserie NACA65/CA verglichen wird.

Wie der Kreisbogen, der als Skelettlinie der NACA 65/CA Serie dient, generiert werden kann,

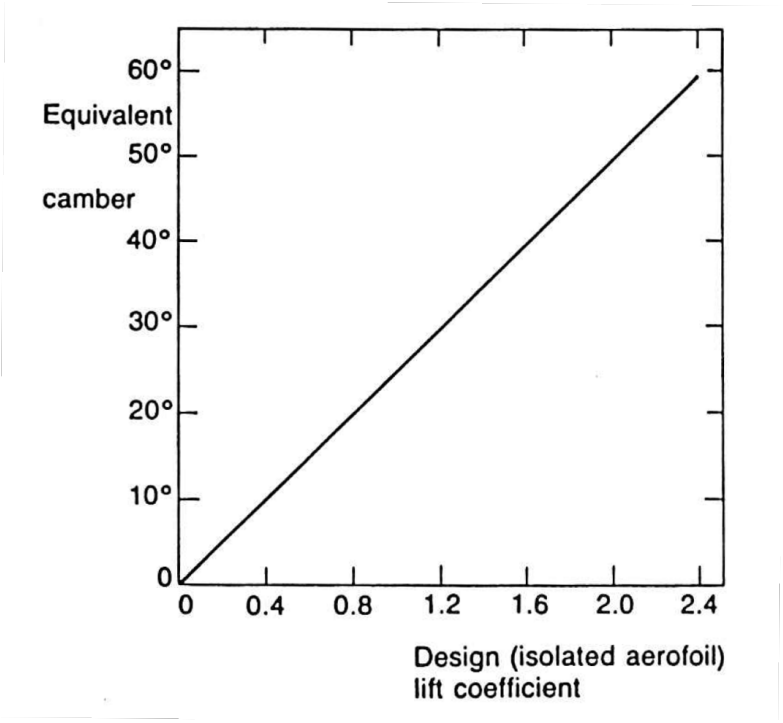


Abbildung 34: Funktionaler Zusammenhang zwischen dem Mittelpunktwinkel des Kreisbogens für die Skelettlinie der NACA 65/CA Profil Serie und dem Design Lift Coefficient CLD [32]

wird nun beschrieben. Die Skelettlinie der NACA 65/CA Profilserie wird in Abhängigkeit des Design Lift Koeffizienten CLD generiert. Abbildung 34 zeigt einen funktionalen Zusammenhang zwischen dem Mittelpunktwinkel des zu generierenden Kreisbogens und dem Design Lift Koeffizienten CLD.

Aus Abbildung 34 geht hervor, dass es sich um einen linearen Zusammenhang zwischen dem Mittelpunkt Winkel des Kreisbogens für die Skelettlinie und dem Design Lift Koeffizienten handelt. Um Ablesefehler zu vermeiden, wurde nach einer mathematischen Bestimmung der linearen Gleichung gesucht. Die Kenntnis über die lineare Funktion, welche diesen Zusammenhang beschreibt wird durch folgende Überlegungen erlangt. In Abbildung 33 ist zu sehen, dass sich die beiden Kurven der Camber Line der NACA 65 Serie und des Kreisbogens bei 50 % der Sehnenlänge schneiden. Der Schnittpunkt $S(50|5,515)$ geht aus Tabelle 2 hervor. Aus Abbildung 33 lassen sich zusätzlich die Punkte $P_{VK}(0|0)$ und $P_{HK}(100|0)$ ablesen, durch welche der Kreisbogen verläuft. Der Kreisbogen ist nun durch drei Punkte definiert. Dadurch

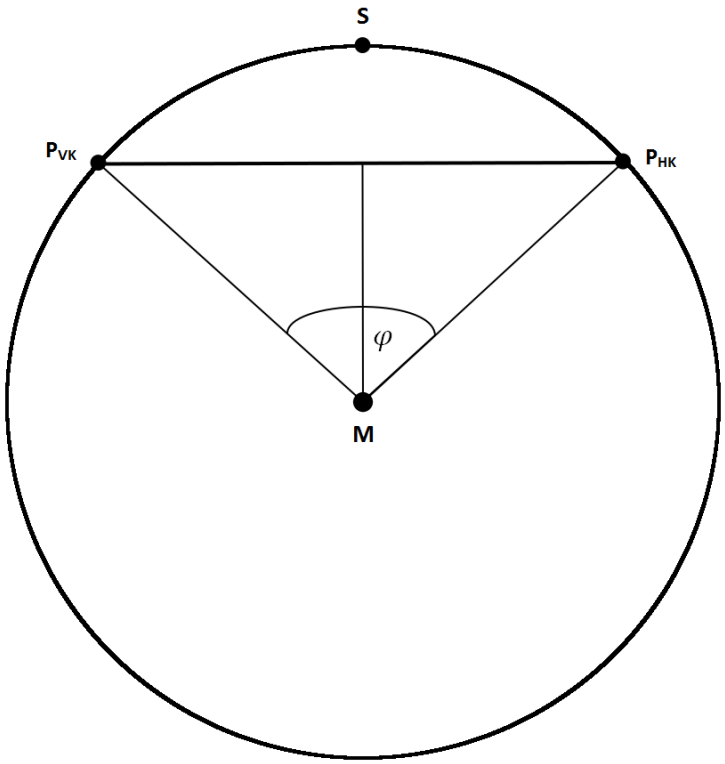


Abbildung 35: Schematisches Schaubild zur Bestimmung des Winkels φ

lässt sich der Mittelpunkt $M(50|-223,897078)$ des Kreises eindeutig bestimmen. Auf die einzelnen Rechenschritte wird an dieser Stelle verzichtet. Der Radius r_{circ} des Kreises ergibt sich somit aus der Summe des Betrags der y-Koordinaten von M und S.

$$r_{circ} = |y_M| + |y_S| \tag{18}$$

Bestimmt werden kann dadurch auch der Mittelpunkt Winkel φ (siehe Abb. 35, sowie Gleichung 19 und 20).

$$\frac{\varphi}{2} = \arcsin\left(\frac{x_{P_{HK}} - x_S}{r_{circ}}\right) = \arcsin\left(\frac{50}{229,412078}\right) = 12,58856^\circ \tag{19}$$

$$\varphi = 25,17712^\circ \tag{20}$$

Der Mittelpunkt Winkel φ beträgt demnach für einen Kreisbogen als Skelettlinie für NACA 65/CA Profile mit einem Design Lift Koeffizienten CLD von eins $25,17712^\circ$. Damit ergibt sich folgende funktionale Abhängigkeit (vgl. Abb. 34).

$$f(cld) = 25,17712 * cld = \varphi \tag{21}$$

Unter Angabe des Design Lift Koeffizienten CLD kann nun der Mittelpunkt Winkel für jeden Kreisbogen als Skelettlinie der NACA 65/CA Profil Serie bestimmt werden. Mit Hilfe dieses Winkels lässt sich anschließend der Radius r_{circ} des Kreises für den entsprechenden Kreisbogen errechnen.

$$r_{circ} = \frac{x_{P_{HK}} - x_S}{\sin\left(\frac{\varphi}{2}\right)} = \frac{50}{\sin\left(\frac{\varphi}{2}\right)} \tag{22}$$

Danach kann auch die y-Koordinate des Mittelpunktes y_M bestimmt werden. Die x-Koordinate des Mittelpunktes des Kreises für jeden Kreisbogen ist konstant 50.

$$y_M = \frac{x_{P_{HK}} - x_S}{\tan\left(\frac{\varphi}{2}\right)} = \frac{50}{\tan\left(\frac{\varphi}{2}\right)} \tag{23}$$

Gleichung 23 gilt lediglich für positive CLD. Bei einem negativen Design Lift Koeffizienten muss die Gleichung invertiert werden, da der Kreisbogen dann unterhalb der x-Achse verläuft. Im nächsten Schritt wird der Kreisbogen generiert. Dabei wird dieser zunächst mit dem Radius r_{circ} und dem Mittelpunkt im Ursprung erzeugt. Auch hier muss zwischen positivem

und negativem Design Lift Koeffizienten unterschieden werden. Bei einem positivem Design Lift Koeffizienten verläuft der Kreisbogen oberhalb der x-Achse nach unten geöffnet. Bei einem negativen Design Lift Koeffizienten verläuft der Bogen unterhalb der x-Achse nach oben geöffnet. Die Punkte werden in gleichmäßigen Abständen zueinander verteilt. Die x- und y-Koordinaten der Punkte sind auch gleichzeitig die Koordinaten des Normalenvektors an dem entsprechenden Punkt. Danach werden alle Punkte des Kreisbogens verschoben, sodass sich der Mittelpunkt $M(50|y_M)$ ergibt. Aus Abbildung 33 geht hervor, dass der Kreisbogen prozentual zu der Sehnenläge skaliert ist. Somit müssen alle x- und y-Koordinaten der Punkte des Kreisbogens mit dem Faktor 0,01 multipliziert werden, da die Dickenverteilung, welche im Kapitel zuvor erzeugt wurde, auf die normierte Sehnenlänge von eins skaliert wurde. Anschließend liegen sowohl die Dickenverteilung, als auch die Skelettlinie vor, sodass mit Hilfe des Algorithmus aus Kapitel 2.2 auch die Profile der NACA 65/CA Serie generiert werden können.

Für die Generierung der Profile der Serien NACA 16 und NACA 65/CA wurde jeweils eine Klasse implementiert. Die Klassen beinhalten Funktionen zur Generierung von Profil-Geometrien, Dickenverteilungen und Skelettlinien der entsprechenden Serien. Bei der Generierung der Profil-Geometrien für die Propeller SRII und SRIII muss darauf geachtet werden, in welchen Bereichen entlang des dimensionslosen Radius welche Profilserie genutzt wurde. Aus Abbildung 31 geht hervor, dass die NACA 65/CA Profilserie bei dem Propeller SRII zwischen dem dimensionslosen Hubradius, der bei 0.239 liegt, und dem dimensionslosen Radius von 0.367 genutzt wurde. Die NACA 16 Profilserie wurde zwischen den dimensionslosen Radien von 0.449 und 1.0 (Tip) genutzt. Über den Bereich zwischen den dimensionslosen Radien von 0.367 und 0.449 erstreckt sich die Transitionszone, in der ein Übergang von der Profilserie NACA 65/CA zur NACA 16 Profilserie stattfindet. Auch für den Propeller SRIII wird eine ähnliche Verteilung der Profilserien in [30] beschrieben. Die Verteilung der NACA65/CA Profilserie liegt im Bereich der dimensionslosen Radien von 0.239 bis 0.37 vor. Die Profilserie NACA 16 wird in dem Bereich von 0.53 bis 1.0 genutzt. Zwischen diesen beiden Bereichen befindet sich ebenfalls eine Transitionszone. Für die drei verschiedenen Propeller SRII und SRIII (aerodynamische und gefertigte Pfeilung) wurden Profile der NACA 16 und 65/CA Serie für die jeweiligen Bereiche erzeugt. Dazu wurden die in Abbildung 31 und 32 dargestellten Design Parameter berücksichtigt. Die Transitionszone wird durch die Interpolation von Propster ebenfalls mit Profilen belegt. Die generierten Profile wurden in Dateien hinterlegt, die dem Format entsprechen, welches mit den Experiment Files kompatibel ist. Abschließend muss in dem

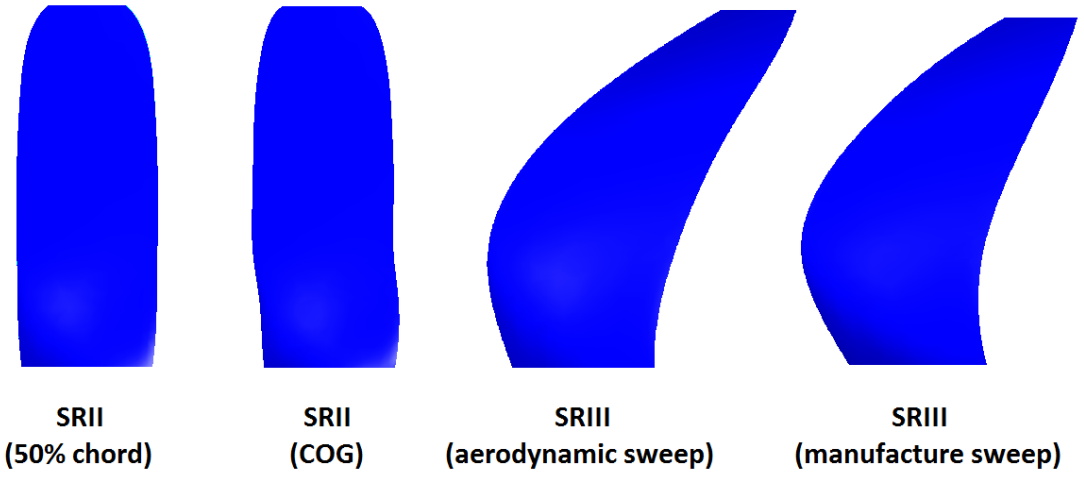


Abbildung 36: Generierte Mantelflächen der Schaufeln der Propeller SRII und SRIII

Experiment File angegeben werden, wie viele Profile nach der Interpolation in gleichmäßigen radialen Abständen vorliegen sollen. Als weitere Information muss festgelegt werden, über wie viele Schaufeln der entsprechende Propeller verfügt. In [1] ist beschrieben, dass sowohl der SRII-Propeller als auch der SRIII-Propeller acht Schaufeln besitzt. Anschließend liegen alle benötigten Daten vor, sodass mit Hilfe der Experiment Files ein Propeller Objekt des entsprechenden Propellers erzeugt werden kann. Mit diesem kann dann unter der Wahl einer entsprechenden Strategie zum Fädeln der Profile eine Mantelfläche der Schaufel mit der implementierten Bibliothek generiert werden. Für den SRII-Propeller liegen trotz intensiver Literaturrecherchen keine Informationen über die Art der Fädelung der Profile vor. Die einzige Information, die aus den vorhandenen Quellen in Bezug auf das Fädeln der Profile des SRII-Propellers gewonnen werden konnte, ist, dass die Schaufeln dieses Propellers keine Pfeilung und Neigung aufweisen. Im Rahmen der Testreihe wurde aus diesem Grund ein Fädeln über 50% der Sehne und über den Flächenschwerpunkt (center of gravity, COG) ausgeführt, da diese Verfahren häufig für Propellerschaufeln ohne Pfeilung und Neigung genutzt werden. Für den SRIII-Propeller wird mit Hilfe des Algorithmus aus Kapitel 5.1.3.1 eine Auffädellinie generiert, sodass die Profile entlang dieser gefädelt werden und eine Propellerschaufel mit Pfeilung und Neigung entsteht. Abbildung 36 zeigt die generierten Mantelflächen der Schaufeln des SRII- und SRIII-Propellers.

6.2.2.2 Vergleich der generierten Mantelflächen

Nachdem die Mantelflächen der Propellerschaufeln generiert wurden, wurden diese verglichen. Zum Vergleich dienten die Modelle, die im Rahmen anderer wissenschaftlicher Arbeiten generiert wurden, sowie Abbildungen und Zeichnungen aus Quellen der NASA. Zunächst sollen die getätigten Vergleiche des Modells der Schaufel des SRII-Propellers vorgestellt werden. Um Herauszufinden, ob die Mantelfläche des SRII-Propellers tatsächlich mit einer der beiden angewandten Strategien zum Fädeln der Profile zu generieren ist, wurden diese mit zwei öffentlich zugänglichen Quellen abgeglichen, die den Propeller SRII geometrisch beschreiben. Eine dieser Quellen stammt von der NASA [33]. In dieser Quelle ist eine Tabelle abgebildet (siehe Tab. 3), die mehrere Profile des SRII-Propellers inklusive einiger Design Parameter beschreibt. Jedes dieser Profile wird in einer Zeile beschrieben.

Non-dimensional Radius r/R	**Blade-angle** $\Delta\beta$ **(Deg)**	**Chord Length b**		**Thickness to Chord Ratio t/b**	**Design Lift Coefficient CLD**	**Airfoil Type**
		m	**ft**			
0.2392	25.25	0.0872	0.2863	0.212	-0.367	NACA 65
0.2857	18.14	0.0884	0.2903	0.111	-0.263	
0.3673	12.22	0.0908	0.2980	0.07	-0.060	
0.449	10.37	0.0915	0.3004	0.045	0.075	Transition
0.5306	7.685	0.0907	0.2977	0.04	0.170	
0.6112	3.88	0.0891	0.2926	0.033	0.188	NACA 16
0.6939	1.54	0.0881	0.2891	0.028	0.160	
0.7755	-1.05	0.0862	0.2829	0.025	0.115	
0.8571	-3.33	0.0822	0.2697	0.022	0.064	
0.9388	-5.17	0.0718	0.2357	0.021	0.025	
0.9796	-6.31	0.0577	0.1895	0.0205	0.01	
1.0	-6.87	0.0310	0.1017	0.02	0.008	

Tabelle 3: Beschreibung mehrerer Profile des SRII-Propellers anhand geometrischer Daten und Design Parameter [33]

Erwähnt werden sollte an dieser Stelle noch einmal, dass Abbildung 31 und Tabelle 3 jeweils aus einem Paper stammt, das von der NASA veröffentlicht wurde. Betrachtet man die erste Zeile aus Tabelle 3 genauer, so fällt Folgendes auf: Teilt man die angegebene Sehnenlänge (Spalte 4) durch den Durchmesser des Propellers, so ergibt sich der Design Parameter b/D, der in Abbildung 31 durch einen der vier Kurvenverläufe dargestellt wird, für die entsprechende Stelle des dimensionslosen Radius. In diesem Fall ist der dimensionslose Radius gleich 0,2392. Der Durchmesser des Propellers beträgt 0,622 m. Dementsprechend ist der Design Parameter b/D in dieser Quelle am dimensionslosen Radius von 0,2392:

$$\frac{b}{D} = \frac{0,0872}{0,622} \approx 0,1402 \tag{24}$$

Aus den digitalisierten Daten von Abbildung 31 ergibt sich der Wert b/D am dimensionslosen Radius von 0,2392 zu etwa 0,15. Die relative Abweichung beträgt somit fast 7%. Diese Abweichung liegt bei allen Profilen, welche in Tabelle 3 beschrieben werden, zwischen 3% und 7%. Auffällig ist auch, dass die Sehnenlänge der ersten vier Profile in Tabelle 3 stetig zunimmt. Dies müsste auch der Kurvenverlauf des Design Parameters b/D widerspiegeln, indem dieser eine positive Steigung im Bereich des dimensionslosen Radius von 0,2392 und 0,449 aufweist. Eine solche positive Steigung ist in diesem Bereich aus Abbildung 31 jedoch nicht zu entnehmen. Abbildung 31 und Tabelle 3 wiedersprechen sich somit. Da die Generierung der Profile im Rahmen dieser Arbeit auf den Kurvenverläufen von Abbildung 31 aufbauen, ist ein Abgleich mit den widersprüchlichen Daten aus Tabelle 3 hinfällig. Die zweite öffentlich zugängliche Quelle [34] stellt durch mehrere Profile und deren Design Parameter ebenfalls ein Modell des SRII-Propellers zur Verfügung. Im Rahmen der Arbeit [34] sind an den Daten, welche in Abbildung 31 dargestellt sind, Korrekturen vorgenommen worden. Diese Korrekturen basieren auf Informationen aus anderen Quellen [27] [29] (vgl. [34] Folie 16). Eine solche Korrektur sollte im Rahmen dieser Arbeit jedoch nicht stattfinden, weshalb die korrigierten Kurvenverläufe der Quelle [34] von den Kurvenverläufen aus Abbildung 31, auf welchen die Tests basieren, geringfügig abweichen. Die Abbildungen 37 – 40 zeigen die vier Kurvenverläufe der Design Parameter aus Abbildung 31 (rote Kurven) im Vergleich mit den korrigierten Daten (blaue Rauten). Die angepassten Kurvenverläufe der Quelle [34] unterscheiden sich geringfügig von jenen aus Abbildung 31. Dementsprechend sind auch Differenzen zwischen den resultierenden Geometrien zu erkennen. Abbildung 41 stellt das

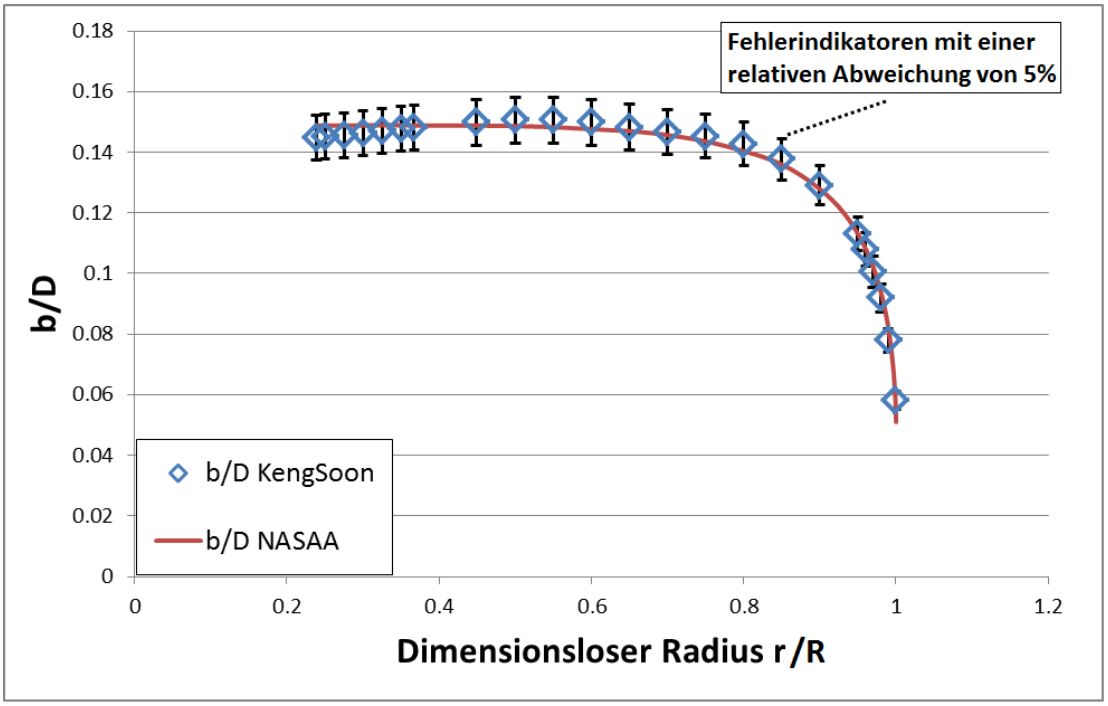


Abbildung 37: Vergleich des Design Parameters b/D des SRII-Propellers aus Abbildung 31 (Linie) mit den Daten aus [34] (Rauten)

resultierende CAD-Modell der Quelle [34] mit einer Zeichnung und einem Foto der Schaufel des SRII-Propellers sowie den resultierenden SRII-Modellen dieses Testfalls gegenüber.

Die Geometrien des CAD-Modells aus [34] und der generierten Mantelfläche aus diesem Testfall, deren Profile über 50% der Sehnenlänge gefädelt wurden, sind sich ähnlich. An den Vorderkanten (leading edges) und Hinterkanten (trailing edges) des Modells aus NASA-CR-4199 [35] und des Modells aus NASA-TM-87030 [27] sind Ein- und Ausbuchtungen zu erkennen, die bei dem CAD-Modell der Quelle und bei dem SRII-Modell, welches innerhalb dieses Testfalls über 50% der Sehnenlänge gefädelt wurde, nicht zu erkennen sind. Diese Ein- und Ausbuchtungen sind ebenfalls bei der generierten Mantelfläche aus diesem Testfall, deren Profile über den Flächenschwerpunkt gefädelt wurden, zu erkennen. Aus der Quelle [34] geht nicht hervor, in welcher Perspektive die gegenübergestellten Propellerschaufeln dargestellt werden. Somit kann nicht eindeutig gewährleistet werden, dass die Schaufeln aus Abbildung 41 in identischer Weise positioniert sind. Eine unterschiedliche Drehung um die radiale Achse der einzelnen Propellerschaufelmodelle in Abbildung 41 ist nicht auszuschließen.

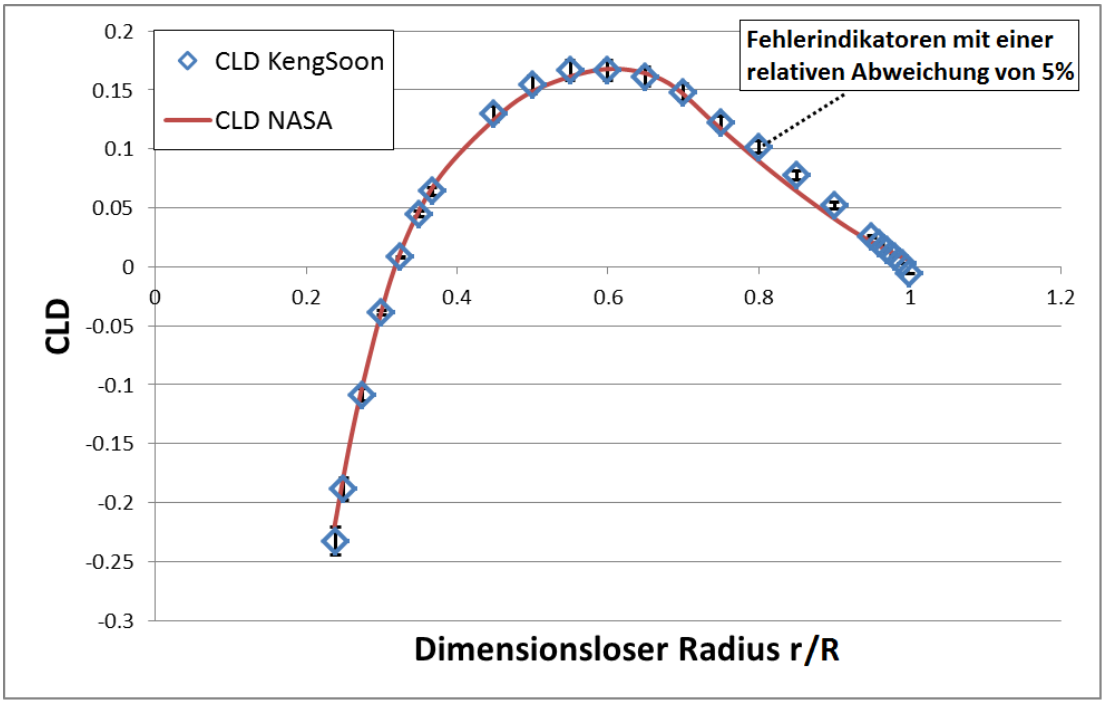


Abbildung 38: Vergleich des Design Parameters CLD des SRII-Propellers aus Abbildung 31 (Linie) mit den Daten aus [34] (Rauten)

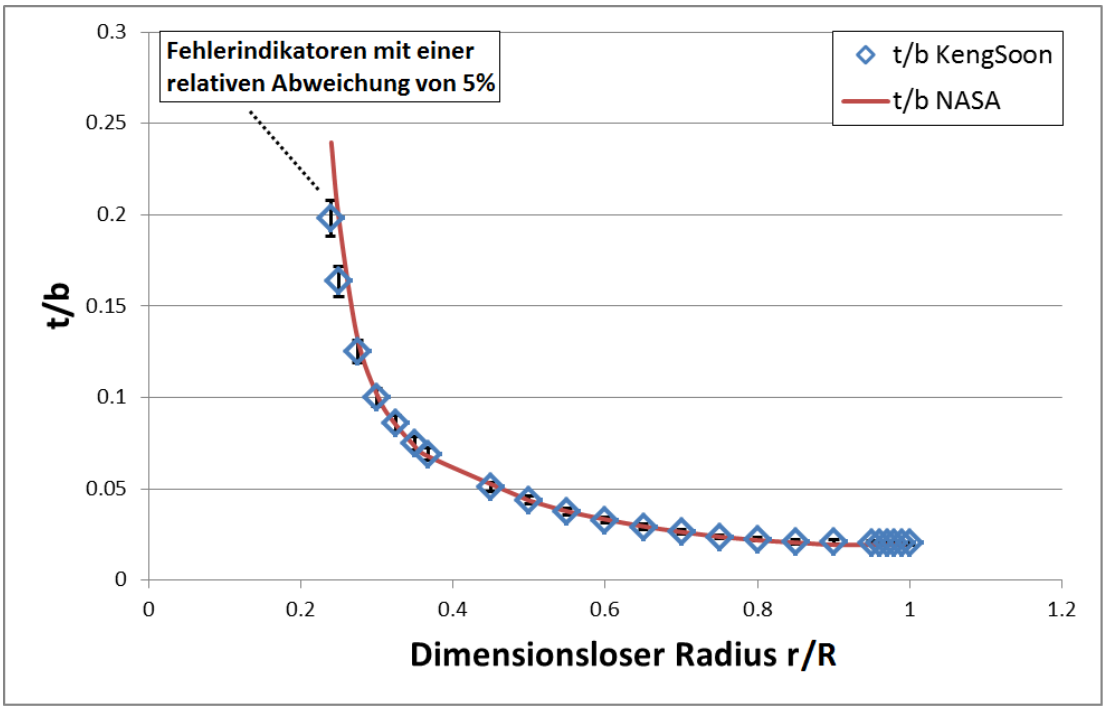


Abbildung 39: Vergleich des Design Parameters t/b des SRII-Propellers aus Abbildung 31 (Linie) mit den Daten aus [34] (Rauten)

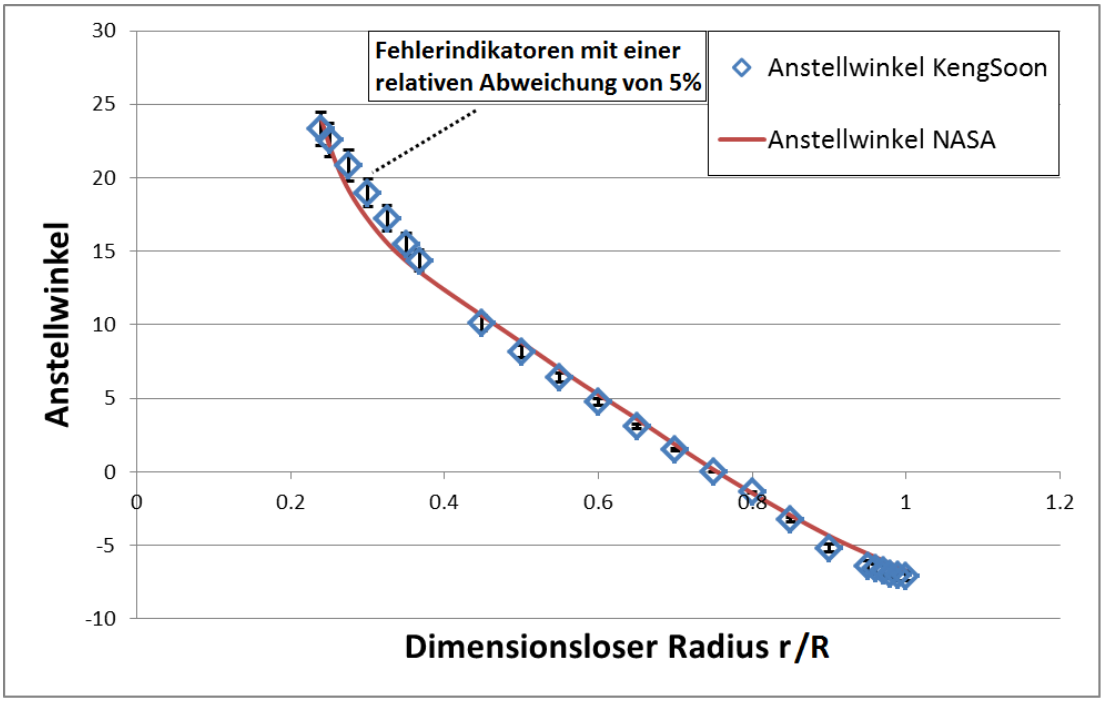


Abbildung 40: Vergleich des Design Parameters β des SRII-Propellers aus Abbildung 31 (Linie) mit den Daten aus [34] (Rauten)

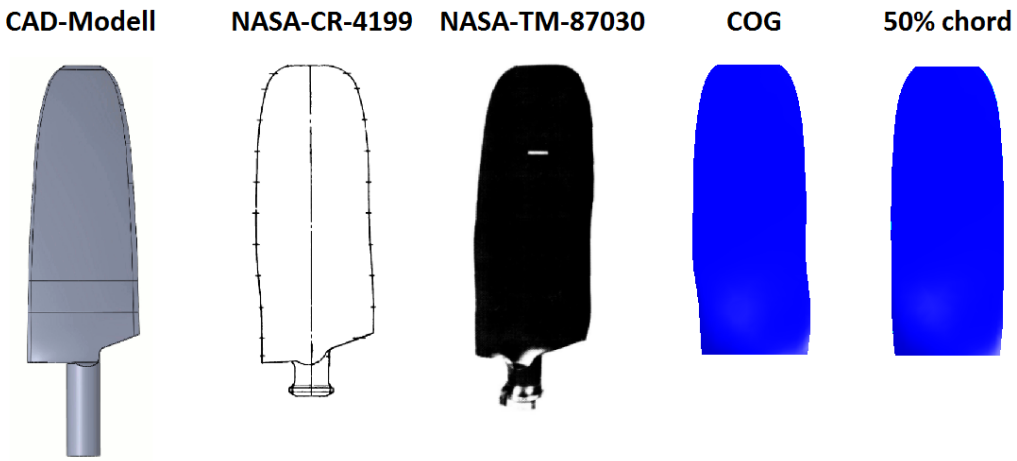


Abbildung 41: Gegenüberstellung mehrerer NASA SRII-Propeller-Geometrien in Anlehnung an [34]

Insgesamt gesehen ist eine große Ähnlichkeit zwischen einigen Modellen gut erkennbar. Es lässt sich schließen, dass die Profile des CAD-Modells von [34] über 50% der Sehnenlänge gefädelt wurden. Abbildung 42, die ebenfalls aus [34] entnommen wurde bestärkt diese Vermutung.

Denkbar wäre auch, dass durch die vorgenommenen Anpassungen der Design Parameter in [34] dem CAD-Modell die Ein- und Ausbuchtungen genommen wurden und auch bei [34] ein Fädeln über den Flächenschwerpunkt stattgefunden hat. Zusammenfassend ist die Ähnlichkeit zwischen den Modellen aus NASA-CR-4199 und NASA-TM-87030 sowie der Mantelfläche dieses Testfalls, deren Profile über den Flächenschwerpunkt gefädelt wurden, deutlich sichtbar. Es liegt also die Vermutung nahe, dass die Profile des SRII-Propellers von der NASA über den Flächenschwerpunkt gefädelt wurden. Bei dem CAD-Modell aus [34] lässt sich keine eindeutige Strategie des Fädelns ausmachen. Eindeutig bestätigen lassen sich die diskutierten Hypothesen aufgrund der nur unzureichenden Beschreibung der Modelle nicht. Die zuvor diskutierten Indizien sprechen jedoch eindeutig dafür, dass der SRII-Propeller über den Flächenschwerpunkt zu fädeln ist, da die Modelle der NASA dafürsprechen. Die NASA hat den Propeller SRII entwickelt. Auch die generierten Mantelflächen des SRIII-Propellers wurden mit Abbildungen aus Quellen der NASA verglichen. Dafür wurde die Zeichnung aus Abbildung

Abbildung 42: Darstellung des Fädelns der Profile des SRII-Propellers in [34]

6 verwendet, die den SRIII-Propeller aus der seitlichen Ansicht in einer Parallelprojektion darstellt. Beide generierten Mantelflächen des SRIII-Propellers mit aerodynamischer und gefertigter Pfeilung wurden mit der Zeichnung abgeglichen. In Abbildung 43 sind die beiden Propellerschaufeln und die Zeichnung gegenübergestellt.

Mit Hilfe dieses Vergleichs bestätigte sich noch einmal, dass es sich bei der Zeichnung um den SRIII-Propeller mit gefertigter Pfeilung handelt. Innerhalb der getätigten Literaturrecherchen wurde keine Abbildung des SRIII-Propellers gefunden, die zu einem Vergleich mit der generierten Mantelfläche mit aerodynamischer Pfeilung hätte dienen können. Der Vergleich zwischen der Zeichnung und der Mantelfläche mit gefertigter Pfeilung bestätigte jedoch, dass die implementierte Bibliothek in der Lage ist eine Propellerschaufel präzise nachmodellieren zu können. Ein weiterer Test sollte dies zusätzlich unterstreichen. Innerhalb [26] befindet sich eine Abbildung, die einige Profile des SRIII-Propellers aus einer Perspektive von oben zeigt, die darstellt, wie die Profile zueinander angeordnet sind (stacked view). Aus den vorliegenden geometrischen Daten des SRIII-Propellers mit gefertigter Pfeilung nach Abschluss der Fädelung wurde ebenfalls versucht, eine solche Anordnung der Profile zu modellieren. Diese wurde anschließend mit der Abbildung verglichen. Abbildung 44 zeigt das Ergebnis.

In Abbildung 44 wird die Ähnlichkeit der beiden Abbildungen der Profile ersichtlich. Von einem Vergleich einzelner Profile ist abzusehen, da sich die radialen Koordinaten der Profile in den beiden Modellen vermutlich unterscheiden. Trotzdem ist die unterschiedliche Form des untersten Profils der beiden Modelle auffallend. Dies könnte daran liegen, dass das unterste Profil derart dargestellt ist, wie es als Verbindung mit der Nabe des Propellers vorliegt. Eine solche Verbindung ist in dem modellierten Modell nicht vorgesehen. Gut verglichen werden

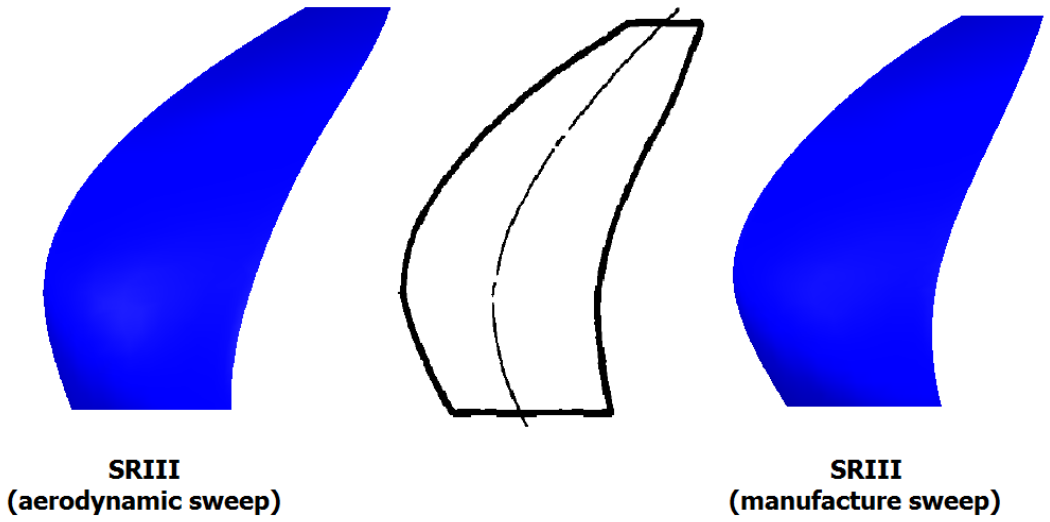


Abbildung 43: Vergleich der generierten Mantelflächen der Schaufeln des SRIII-Propellers mit einer Zeichnung

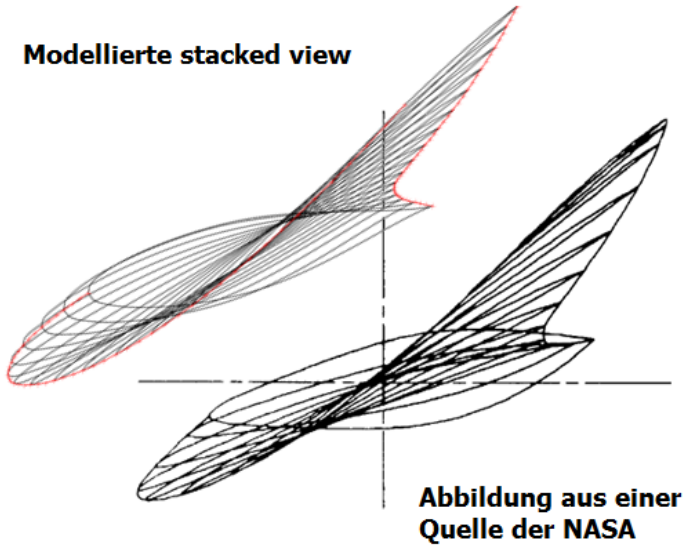


Abbildung 44: Vergleich der gefädelten Profile einer Schaufel des SRIII-Propellers mit gefertigter Pfeilung

kann der Verlauf der Vorder- und Hinterkante der beiden Modelle. Der Verlauf der Vorderkante ist bei beiden Modellen gleich. Unterschiede sind nicht festzustellen. Auch der Verlauf der Hinterkante beider Modelle ist ähnlich. Lediglich der Verlauf im Bereich niedriger radialer Koordinaten unterscheidet sich leicht. Dieser könnte sich durch minimale Ablesefehler bei der Digitalisierung der Design Parameter ergeben haben. Durch eine minimale Änderung des Anstellwinkels oder der Sehnenlänge der unteren Profile könnte bereits ein ähnlicher Verlauf der Hinterkante erreicht werden. Auffallend ist auch, dass der Verlauf der Anstellwinkel der Profile entlang der radialen Achse bei beiden Modellen übereinstimmt. Festzuhalten ist somit, dass bis auf den Verlauf der Hinterkante keine drastischen Unterschiede zwischen den beiden Modellen festzustellen sind. Die minimalen Unterschiede sind vermutlich auf Ablesefehler der Design Parameter aus Abbildung 32 zurückzuführen. Die beiden Vergleiche der Mantelfläche der Schaufel des SRIII-Propellers mit gefertigter Pfeilung mit Abbildungen der Schaufeln aus Quellen der NASA lieferten eine nahezu vollständige Übereinstimmung der verglichenen Modelle. Für das Modell der Mantelfläche des SRIII-Propellers mit aerodynamischer Pfeilung konnte kein vergleichbares Modell gefunden werden. Da sich diese Mantelfläche von der Mantelfläche des SRIII-Propellers mit gefertigter Pfeilung nur durch eine andere Verteilung des Pfeilwinkels entlang der radialen Achse unterscheidet (siehe Abb. 28), ist davon auszugehen, dass dieses Modell die Form der Propellerschaufel im Betrieb ebenfalls präzise beschreibt.

6.2.3 Robustheitsprüfung der Algorithmen

Weiterhin soll getestet werden, wie die Bibliothek damit umgeht, wenn eine der vier implementierten Strategien zum Fädeln der Profile ausgewählt wird, die nicht für das entsprechende Modell der Propellerschaufel vorgesehen ist. Die Strategie zum Fädeln der Profile ist die einzige Parametereingabe, die der Benutzer zusätzlich zu dem Objekt der Klasse `Propeller` an die Bibliothek zur Mantelflächengenerierung liefern muss. Wird von der Schnittstelle von Propster ein Objekt der Klasse `Propeller` geliefert, welches der Beschreibung aus Kapitel 3.2 entspricht, so ist die Wahl der Strategie zum Fädeln der Profile die einzige mögliche Fehlerquelle. Mit Abschluss dieser Testreihe soll das Fehlerpotential eingeschätzt werden können.

Zur Prüfung der Robustheit der Algorithmen zum Fädeln von Profilen und zur Generierung der Mantelflächen wurden die Profile der Anwendungsfälle SRII, SRIII (aerodynamische Pfeilung) und SRIII (gefertigte Pfeilung) mit den vier implementierten Strategien, die im Kapitel 5.1.5 vorgestellt wurden, gefädelt. Mit Abschluss dieser Testreihe soll das Fehlerpotential der

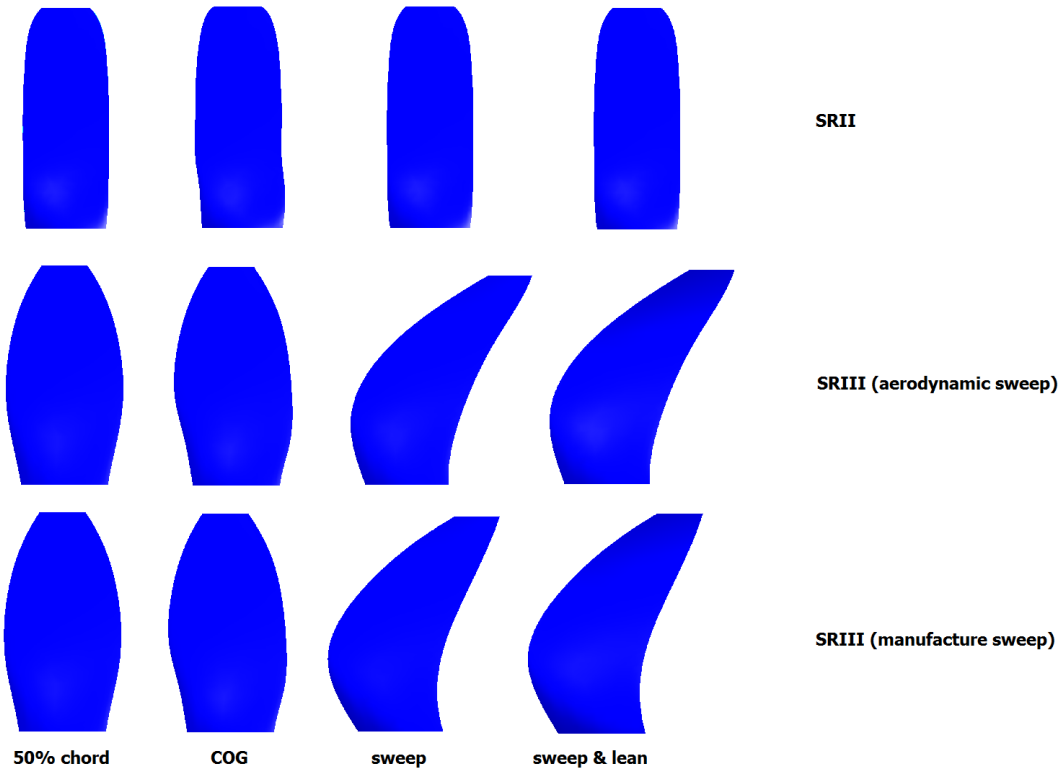


Abbildung 45: Generierte Mantelflächen der Propeller SRII und SRIII mit verschiedenen Varianten des Fädelns

Algorithmen zum Fädeln von Profilen eingeschätzt werden können. Die generierten Daten wurden anschließend dazu genutzt, um die Mantelflächen der Propellerschaufeln zu generieren. Abbildung 45 zeigt die als Ergebnis dieser Testreihe vorliegenden zwölf Mantelflächen.

Die Robustheit der Algorithmen wurde durch den Anwendungsfall aufgezeigt. Alle zwölf Mantelflächen konnten generiert werden, sodass kein Fehlerpotential bei den Algorithmen zum Fädeln von Profilen nachgewiesen werden konnte. Die Mantelflächen des SRIII-Propellers unterscheiden sich zwischen allen vier Varianten des Fädelns. Beim SRII-Propeller sind die Formen der Mantelflächen unter der Nutzung der Strategien des Fädelns über 50% der Sehnenlänge, sowie den beiden Strategien mit einer Auffädellinie (sweep, sweep & lean) identisch. Dies liegt daran, dass die Profile bei den Algorithmen zum Fädeln über eine Auffädellinie zunächst auch über 50% der Sehne gefädelt werden und anschließend entsprechend des Verlaufs der Auffädellinie verschoben werden. Der SRII-Propeller besitzt

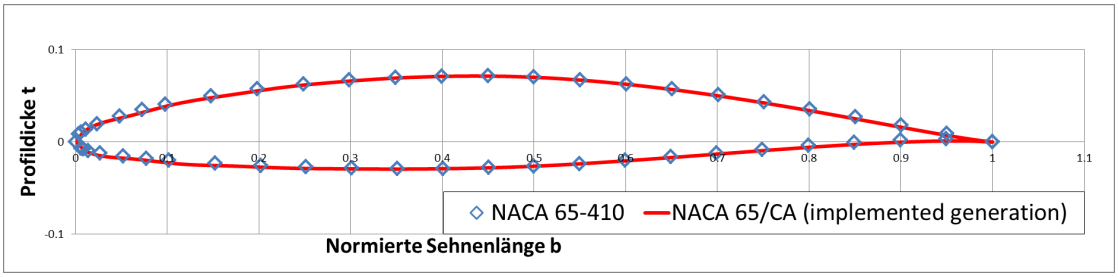


Abbildung 46: Beispielhafter Vergleich eines generierten NACA 65/CA Profils mit dem Datensatz aus [36]

jedoch keine Pfeilung und Neigung. Die Pfeilwinkel aller Profile haben einen Wert von Null. Der Algorithmus zur Generierung einer Auffädellinie erzeugt somit eine Gerade, die parallel zur radialen Achse verläuft. Die Profile werden dadurch nicht weiter verschoben. Als Resultat liegt eine Fädelung vor, die jener über 50% der Sehnenlänge entspricht.

6.2.4 Validierung der Algorithmen zur Generierung von Profil-Geometrien

Neben den Vergleichen verschiedener Modelle der Propeller SRII und SRIII fanden auch Validierungen der Generierungsmethoden der Profil Serien NACA 16 und NACA 65/CA statt. Dazu wurden öffentlich zugängliche Datensätze von Profilen dieser Serien [36] mit generierten Profilen aus den im Kapitel 2.2 und 6.2.2.1 vorgestellten Routinen verglichen. Die Datensätze beinhalten eine geordnete Punkteliste, die die Profil-Geometrie beschreibt, sowie die Design Parameter der maximalen Profildicke t und des Design Lift Coefficients CLD. Die Design Parameter wurden als Eingaben für die entsprechenden Routinen genutzt, sodass die Geometrien generiert und mit den Datensätzen der Quellen abgeglichen werden konnten. Die generierten Geometrien stimmten mit den Datensätzen der Quellen überein. Abbildung 46 zeigt den beispielhaften Vergleich zweier NACA 65/CA Profile mit einer maximalen Profildicke von 10% der Sehnenlänge und einem Design Lift Coefficient von 0,4 NACA 65/CA - 410. Die Markierungen in Form einer blauen Raute wurden dem Datensatz der Quelle entnommen und der rote Kurvenverlauf wurde durch die im Rahmen dieser Arbeit entwickelten Algorithmen generiert.

Insgesamt wurden jeweils sechs Profile der NACA 16 und der NACA 65/CA Serie mit vergleichbaren Datensätzen [36] abgeglichen. Die Ergebnisse der Vergleiche lieferten die Erkenntnis, dass die Algorithmen zur Profil-Generierung zufriedenstellend exakt arbeiten.

Als Fazit des Kapitels 6.2 kann festgehalten werden, dass alle durch die Anwendungstests geprüften Algorithmen präzise arbeiten. Die Propellermantelflächen der Anwendungsfälle SRII und SRIII konnten nachmodelliert werden und weisen eine identische Form zu den in Quellen auffindbaren Abbildungen der Propellerschaufeln auf. Dazu steuern auch die Algorithmen zur Generierung der Auffädellinien und Profil-Geometrien der Serien NACA 16 und NACA 65/CA bei sowie die Routinen zum Fädeln von Profilen, deren Robustheit nachgewiesen werden konnte. Die Validierungsergebnisse dieser Algorithmen sind positiv hervorzuheben.

Zusammenfassend wurde eine Bibliothek zur Generierung von Mantelflächen von Propellerschaufeln implementiert, die in der Lage ist aus der Parametrisierung, die in Kapitel 2 vorgestellt wurde, eine präzise Mantelfläche der Propellerschaufel zu generieren. Die Aufgabenstellung der Bachelorarbeit wurde somit vollständig erfüllt. Die mit Hilfe der Bibliothek generierten Mantelflächen können dazu genutzt werden, dreidimensionale Strömungsrechnungen auszuführen. Die Ergebnisse der in Propster implementierten zweidimensionalen Strömungsrechnung können nun mit den Ergebnissen der dreidimensionalen Strömungsrechnung validiert werden.

7 Zusammenfassung und Ausblick

Zusammenfassend kann festgehalten werden, dass mit Abschluss dieser Arbeit eine Bibliothek vorliegt, die die Aufgabenstellung erfüllt, aus einer gegebenen Parametrisierung eines Propellers die Mantelfläche einer seiner Propellerschaufeln zu generieren. Die Bibliothek ist an das Software-Tool Propster angelehnt, welches Funktionen zur aerodynamischen Analyse von Propellerschaufeln bereitstellt, und soll zukünftig in Propster integriert werden. Sie bietet die Funktionalität, Propellerschaufeln mit oder ohne Pfeilung und Neigung zu modellieren. Profile von ungepfeilten und ungeneigten Propellerschaufeln können über einen Punkt prozentual zur Sehnenlänge oder über den Flächenschwerpunkt gefädelt werden. Um die Profile von gepfeilten und geneigten Propellern zu fädeln, bietet die Bibliothek die Funktionalität, die benötigte Auffädellinie zu rekonstruieren, sodass die Profile anschließend entlang dieser Linie gefädelt werden können. Die ausgeführten Anwendungstests zeigten auf, dass die Bibliothek mit Hilfe der Parametrisierung eine präzise Rekonstruktion der Mantelflächen von Propellerschaufeln ermöglicht. Die Mantelflächen können im Anschluss an diese Arbeit dazu genutzt werden, dreidimensionale numerische Strömungsrechnungen (CFD-Rechnungen) durchzuführen. Die Ergebnisse dieser Rechnungen sollen dazu genutzt werden, die bereits in Propster implementierte zweidimensionale Strömungsrechnung validieren zu können. Um auf die Mantelflächen der Propellerschaufeln zurückgreifen zu können, dienen die CAD-Datenformate STEP und IGES. Die Bibliothek bietet die Funktionalität, die Daten der generierten Mantelflächen in diese Datenformate zu überführen. In näherer Zukunft sind für die Anwendungsfälle der Anwendungstests SRII- und SRIII-Propeller erste CFD-Rechnungen geplant. Die Ergebnisse der Rechnungen sollen anschließend ersten Validierungen der zweidimensionalen Strömungsrechnung von Propster dienen. Mit Hilfe der in den Anwendungstests generierten Experiment Files können die Daten für die zweidimensionale Strömungsrechnung der Propeller SRII und SRIII bereitgestellt werden. Für die CFD-Rechnungen können die CAD-Datenformate genutzt werden. Als Anwendungsfälle für die implementierte Bibliothek dienten bislang lediglich die beiden Propeller der NASA SRII und SRIII. Im Allgemeinen kann unter der Kenntnis der in dieser Arbeit beschriebenen Parametrisierung von Propellern jede beliebige Propellerschaufel modelliert werden. Die einzige Einschränkung besteht in den implementierten Fädelroutinen. Sind die Pfofile eines Propellerblattes, welches modelliert werden soll, nicht mit einer Fädelroutine angeordnet, die in der Bibliothek implementiert ist, so ist eine Erweiterung der Bibliothek erforderlich. Die Architektur der Bibliothek ist auf mögliche

Erweiterungen ausgelegt. Im Verlauf dieser Arbeit wurde erwähnt, dass eine Erweiterung durch die Funktionalität der Generierung von inneren Strukturen der Propellerschaufel in Betracht gezogen werden könnte. Jenes Vorhaben wird eventuell in Zukunft umgesetzt werden. Es wäre dann möglich, die Volumen der einzelnen Teilstrukturen, die aus unterschiedlichen Materialien bestehen, zu ermitteln. Mit der Kenntnis der Volumen und der Dichten der Materialien könnten Gewichtsabschätzungen der Propellerschaufeln erfolgen. Weiterhin bestünde nach der Generierung der inneren Strukturen die Möglichkeit, Trägheitsmomente der Propellerschaufeln zu bestimmen.

A Anhang

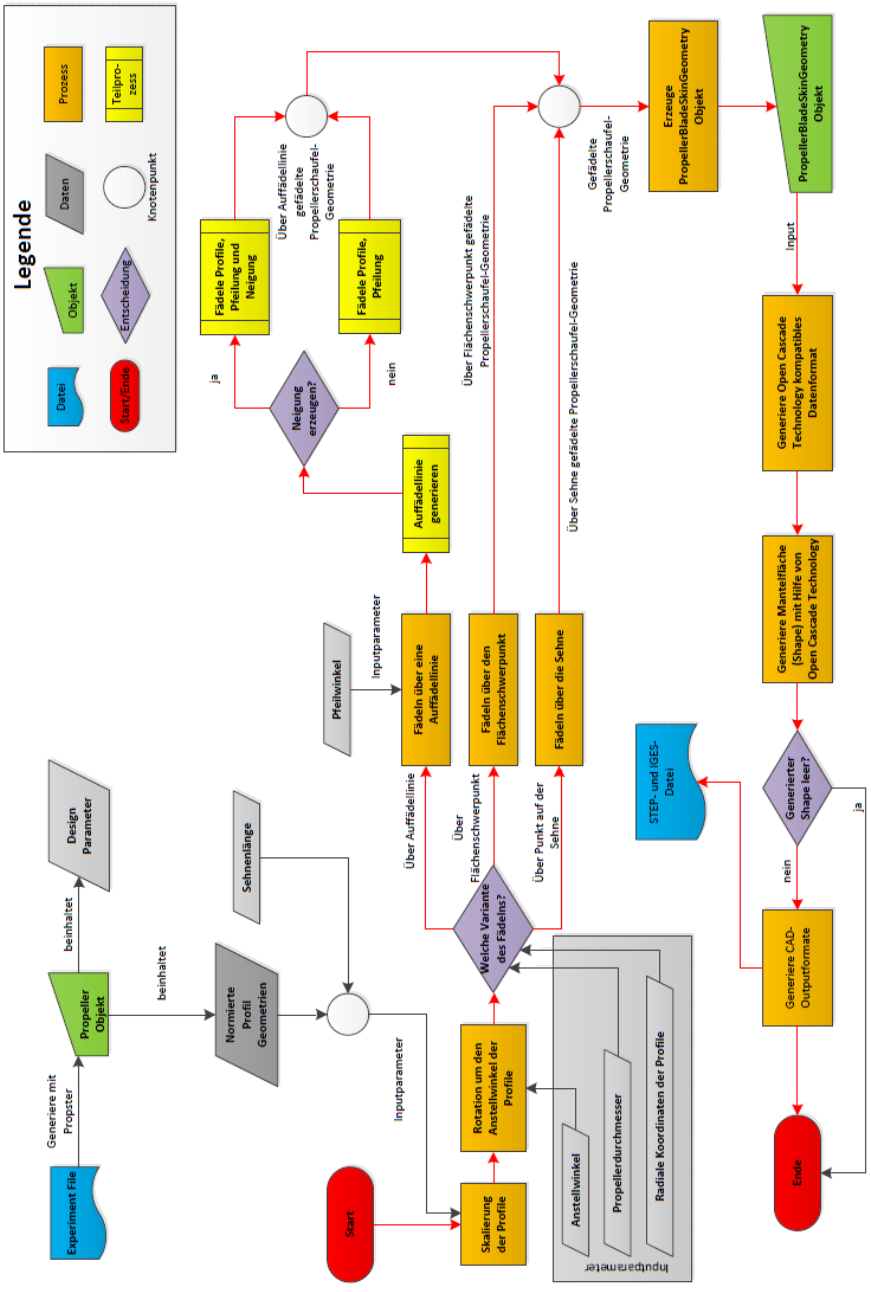
A.1 Experiment File des SRII-Propellers

```
<?xml version="1.0" encoding="utf-8"?>
<AirScrewExperiment version="0">

  <InputDirectoryPath>D:\\data\\PraxisphaseVI\\ExperimentFilesPropster\\
      Input\\</InputDirectoryPath>
  <OutputDirectoryPath>D:\\data\\PraxisphaseVI\\ExperimentFilesPropster\\
      Output\\</OutputDirectoryPath>

  <XFoilPath></XFoilPath>
  <MSESPath></MSESPath>

<GoldsteinData></GoldsteinData>

  <LoadObjects>
  </LoadObjects>

  <CreateObjects>

    <Airfoil name="0.2SRII_NACA65CA">
      <GeometryFile type="String">Profils\\0.2SRII_NACA65CA</GeometryFile
          >
      <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
    </Airfoil>

    <Airfoil name="0.3SRII_NACA65CA">
      <GeometryFile type="String">Profils\\0.3SRII_NACA65CA</GeometryFile
          >
      <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
    </Airfoil>

    <Airfoil name="0.35SRII_NACA65CA">
      <GeometryFile type="String">Profils\\0.35SRII_NACA65CA</
          GeometryFile>
      <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
    </Airfoil>

```

```
32      <Airfoil name="0.45SRII_NACA16">
33        <GeometryFile type="String">Profils\\0.45SRII_NACA16</GeometryFile>
34        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
35      </Airfoil>
36
37      <Airfoil name="0.5SRII_NACA16">
38        <GeometryFile type="String">Profils\\0.5SRII_NACA16</GeometryFile>
39        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
40      </Airfoil>
41
42      <Airfoil name="0.6SRII_NACA16">
43        <GeometryFile type="String">Profils\\0.6SRII_NACA16</GeometryFile>
44        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
45      </Airfoil>
46
47      <Airfoil name="0.7SRII_NACA16">
48        <GeometryFile type="String">Profils\\0.7SRII_NACA16</GeometryFile>
49        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
50      </Airfoil>
51
52      <Airfoil name="0.8SRII_NACA16">
53        <GeometryFile type="String">Profils\\0.8SRII_NACA16</GeometryFile>
54        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
55      </Airfoil>
56
57      <Airfoil name="0.9SRII_NACA16">
58        <GeometryFile type="String">Profils\\0.9SRII_NACA16</GeometryFile>
59        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
60      </Airfoil>
61
62      <Airfoil name="1.0SRII_NACA16">
63        <GeometryFile type="String">Profils\\1.0SRII_NACA16</GeometryFile>
64        <AerodynamicSolver name="XFOIL"></AerodynamicSolver>
65      </Airfoil>
66
67      <Blade name="Blade SRII" nondimensionalHubRadius ="0.239" export="
            false">
68        <station nondimensionalRadius="0.2" airfoilName="0.2SRII_NACA65CA">
              </station>
69        <station nondimensionalRadius="0.3" airfoilName="0.3SRII_NACA65CA">
              </station>
```

```
70        <station nondimensionalRadius="0.35" airfoilName="0.35SRII_NACA65CA
              "></station>
71        <station nondimensionalRadius="0.45" airfoilName="0.45SRII_NACA16">
              </station>
72        <station nondimensionalRadius="0.5" airfoilName="0.5SRII_NACA16"></
              station>
73        <station nondimensionalRadius="0.6" airfoilName="0.6SRII_NACA16"></
              station>
74        <station nondimensionalRadius="0.7" airfoilName="0.7SRII_NACA16"></
              station>
75        <station nondimensionalRadius="0.8" airfoilName="0.8SRII_NACA16"></
              station>
76        <station nondimensionalRadius="0.9" airfoilName="0.9SRII_NACA16"></
              station>
77        <station nondimensionalRadius="1.0" airfoilName="1.0SRII_NACA16"></
              station>
78      </Blade>
79
80      <Propeller name="SRII" numberOfStations="50" export="false">
81        <Diameter type="double" unit="[m]">0.622</Diameter>
82        <NumberOfBlades type="int">8</NumberOfBlades>
83        <BladeName type="String">Blade SRII</BladeName>
84        <GeometryFile type="String">\\Geometry\\Geometry SRII.csv</
              GeometryFile>
85        <EngineNacelle mode="OFF" type="double" unit="[m]" length="3.2"
              radius="1.0287"></EngineNacelle>
86      </Propeller>
87
88    </CreateObjects>
89
90    <Export>
91      <XML>
92        <PropellerFile name="SRII">SRII.xml</PropellerFile>
93      </XML>
94    </Export>
95
96  </AirScrewExperiment>
```

A.2 Flussdiagramm der implementierten Bibliothek



Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Glenn A Mitchell und Daniel C Mikkelson. *Summary and recent results from the NASA advanced High Speed Propeller Research Program*. National Aeronautics and Space Administration, 1982. S.1–2, Abb.6.

[2] Jannik Häßy. *Validierung und Implementierung eines Verfahrens zur Vorauslegung von Propellern auf Basis der Blattelementtheorie, Bachelorarbeit vorgelegt an der Rheinisch-Westfälischen Technischen Hochschule Aachen*. Deutsches Zentrum für Luft- und Raumfahrt, 2016. Abb. 3.4.

[3] Richard Von Mises. *Theory of flight*. Courier Corporation, 1959. Kapitel 6.

[4] Ira Herbert Abbott und Albert Edward Von Doenhoff. *Theory of wing sections, including a summary of airfoil data*. Courier Corporation, 1959. Kapitel 4 & 6.

[5] Willy JG Bräunling. *Flugzeugtriebwerke: Grundlagen, Aero-Thermodynamik, ideale und reale Kreisprozesse, Thermische Turbomaschinen, Komponenten, Emissionen und Systeme, 3.Auflage*. Springer-Verlag, 2015. S.101–104 & 907.

[6] JF Dugan und BA Miller und DA Sagerser. *Status of advanced turboprop technology*. 1978. S.141 & 158.

[7] JG Maser u.a. *Parametric studies of advanced turboprops*. National Aeronautics and Space Administration, 1990. S.58–59.

[8] o.V. *Computerunterstützte Konstruktion*. 2017. http://www.itwissen.info/CAD-computer-aided-design-Computerunterstuetzte-Konstruktion.html, Einsichtnahme: 03.07.2017.

[9] o.V. *Licensing :Open Cascade*. o.J. https://www.opencascade.com/content/licensing, Einsichtnahme: 12.07.2017.

[10] o.V. *Geometrical 3D Modeling :Open Cascade*. o.J. https://www.opencascade.com/content/3d-modeling, Einsichtnahme: 13.07.2017.

[11] o.V. *CAD Data Processing :Open Cascade*. o.J. https://www.opencascade.com/content/cad-data-processing, Einsichtnahme: 13.07.2017.

[12] o.V. *Design und Optimierung :DLR Abteilung Fan und Verdichter*. 2013. http://www.dlr.de/at/desktopdefault.aspx/tabid-9123/, Einsichtnahme: 14.07.2017.

[13] Christian Voß. *BladeGenerator, Schaufel Parametrisierung (Verdichter), DLR-interne Dokumentation*. 2017.

[14] Jannik Häßy. *Propster Source Code, DLR-intern*. 2017. https://svn.dlr.de/Propster, Einsichtnahme: 14.07.2017.

[15] Jannik Häßy. *Propster Software Design Review, DLR-interne Powerpoint Präsentation*. 2017.

[16] Harald Kornmayer. *Software Engineering II, Testen in großen Frameworks, Skript einer Vorlesung an der DHBW Mannheim*. 2016. Folie 173.

[17] o.V. *Primer, Dokumentation :Google Test*. o.J. https://github.com/google/googletest/blob/master/googletest/docs/Primer.md, Einsichtnahme: 11.07.2017.

[18] o.V. *Advanced Guide, Dokumentation :Google Test*. o.J. https://github.com/google/googletest/blob/master/googletest/docs/AdvancedGuide.md, Einsichtnahme: 11.07.2017.

[19] Les Piegl und Wayne Tiller. *The NURBS book, 2nd Edition*. Springer Science & Business Media, 2012.

[20] HW Lang. *Polygon, Konvexe Hülle :FH Flensburg*. 2002. http://www.iti.fh-flensburg.de/lang/algorithmen/geo/polygon.htm, Einsichtnahme: 14.08.2017.

[21] Oliver Karch. *Algorithmische Geometrie, Skript einer Vorlesung an der Martin-Luther-Universität Halle-Wittenberg*. 2000. http://users.informatik.uni-halle.de/∼schenzel/ss07/Uebung-D/vorlesung.pdf, Einsichtnahme: 15.08.2017. S.33.

[22] Paul Bourke. *Calculating the Area and Centroid of a Polygon*. 1988. http://www.seas.upenn.edu/∼sys502/extra_materials/Polygon%20Area%20and%20Centroid.pdf, Einsichtnahme: 15.08.2017.

[23] B Lehmann. *Vermessungskunde I, Skript einer Vorlesung der Hochschule Trier*. 2014. https://www.hochschule-trier.de/fileadmin/groups/11/bauingenieurwesen/Personen/Lehmann/Vermessungstechnik/Skript/skript_verm1.pdf, Einsichtnahme: 15.08.2017. S.13.

[24] o.V. *Dokumentation :Open Cascade Technology*. 2010. https://www.opencascade.com/doc/occt-6.9.1/refman/html/index.html, Einsichtnahme: 14.07.2017.

[25] Karl Eilebrecht und Gernot Starke. *Patterns kompakt, Entwurfsmuster für effektive Software-Entwicklung, 4.Auflage*. Springer Vieweg, 2013. S.29–33 & 66–68 & 77–79.

[26] C Rohrbach u.a. *Evaluation of Wind Tunnel Performance Testings of an Advanced* 45° *Swept Eight-bladed Propeller at Mach Numbers from 0.45 to 0.85*. National Aeronautics and Space Administration, 1962. Abb.16–18.

[27] George L Stefko und Robert J Jeracki. *Wind-tunnel results of advanced high-speed propellers at takeoff, climb, and landing mach numbers*. National Aeronautics and Space Administration, 1985. S.3, Abb.6 & 7, Tab.1.

[28] Lawrence J Bober und Li-Ko Chang. *Factors influencing the predicted performance of advanced propeller designs*. National Aeronautics and Space Administration, 1981. Abb.3.

[29] Daniel C Mikkelson u.a. *Design and performance of energy efficient propellers for Mach 0.8 cruise*. National Aeronautics and Space Administration, 1977. S.8, Abb.13.

[30] Robert J Jeracki und Daniel C Mikkelson und Bernard J Blaha. *Wind tunnel performance of four energy efficient propellers designed for Mach 0.8 cruise*. National Aeronautics and Space Administration, 1979. S.3.

[31] WF Lindsey und DB Stevenson und BN Daley. *Aerodynamics characteristics of 24 NACA 16-series airfoils at Mach numbers between 0.3 and 0.8*. National Advisory Committee for Aeronautics, 1948. S.3–5.

[32] Nicholas A Cumpsty. *Compressor aerodynamics*. Longman Scientific & Technical, 1989. S.478–482.

[33] TSR Reddy und KRV Kaza. *Analysis of an unswept propfan blade with a semiempirical dynamic stall model*. National Aeronautics and Space Administration, 1989. Tab.1.

[34] Voo Keng Soon u.a. *Prediction of noise emission from the NASA SR-2 Propeller*. 2015. S.14–17.

[35] TA Egolf u.a. *An analysis for high speed propeller-nacelle aerodynamic performance prediction. Volume 1: Theory and application*. National Aeronautics and Space Administration, 1988. Abb.30.

[36] o.V. *Airfoil Database :UIUC Applied Aerodynamics Group, NACA16 & NACA65/CA Series*. o.J. https://m-selig.ae.illinois.edu/ads/coord_database.html#N, Einsichtnahme: 10.08.2017.