Bachelorarbeit

# Abstract

# Verzeichnisse

## Nomenklatur

r – Radiale Koordinate eines Profils – [m]

R – Propellerradius – [m]

r/R – dimensionsloser Radius – [-]

b – Sehnenlänge – [m]

β – Anstellwinkel – [m]

t – maximale Dicke eines Profils – [m]

## Abkürzungsverzeichnis

PCA – Pitch Change Axis

CAD – Computer Aided Design

NURBS – Non Uniform Rational B-Spline

STEP/ STP - Standard for the exchange of product model data

IGES/ IGS - Initial Graphics Exchange Specification

GUI – Graphical User Interface

# 1 Einleitung

AT-TWK -> Analyse von Triebwerkskonzepten

InSim -> Strömungsrechnung 2D schnelle Ergebnisse die in Analyse mit einfließen,

Zukünftig 3D um 2D zu validieren und kalibrieren -> Propellermantelfläche benötigt, die als input für 3D Berechnungsverfahren dienen soll

# 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. Die daraus resultierenden Veröffentlichungen der getätigten 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. Diese Parametrisierung soll im weiteren Verlauf dieses Kapitels vorgestellt werden. Weiterhin sollen fundamentale Begriffe, die zur Beschreibung eines Propellerblattes notwendig sind, erläutert werden.

Die geometrische Information eines Propellerblattes liegt häufig in Form von zweidimensionalen axialen Schnitten durch die Propellerschaufel, die sogenannten Profile, vor. Deshalb werden die Profile zunächst 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 eines Propellerblattes

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 X zeigt die Zeichnung einer Propellerschaufel. Zu sehen sind vier Schnitte durch diese Schaufel, sowie die daraus resultierenden Schnittflächen, die Profile.

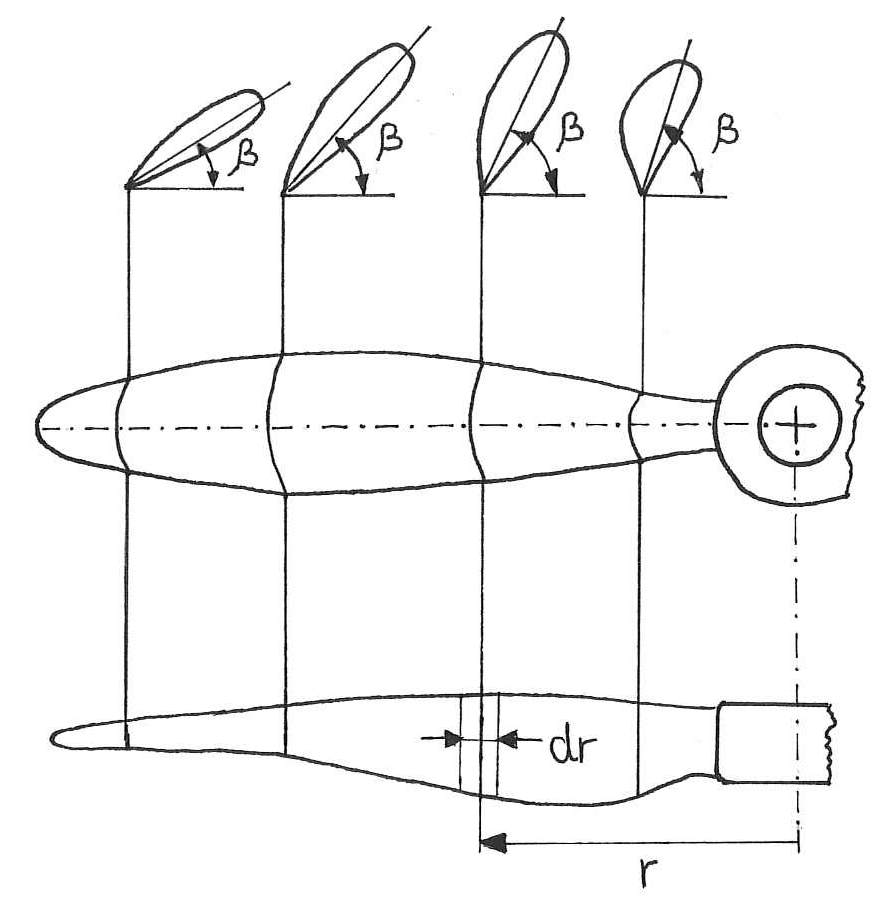


Abbildung X

Profile einer Propellerschaufel

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 (upper surface) eines Profils befindet sich in der Abbildung oben und die Druckseite (lower surface) 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. 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 (max. thickness) eines Profils beinhaltet.

Profile werden an der Vorderkante von einem Fluid angeströmt. Der Anstellwinkel 𝛽 eines Profils ist definiert als Winkel zwischen der Richtung aus der das Fluid strömt und der Sehne des Profils. In Abbildung X-1 sind die Anstellwinkel der einzelnen Profile eingezeichnet. Abbildung X veranschaulicht die Definition des Anstellwinkels und der weiteren erläuterten Begriffe noch einmal.

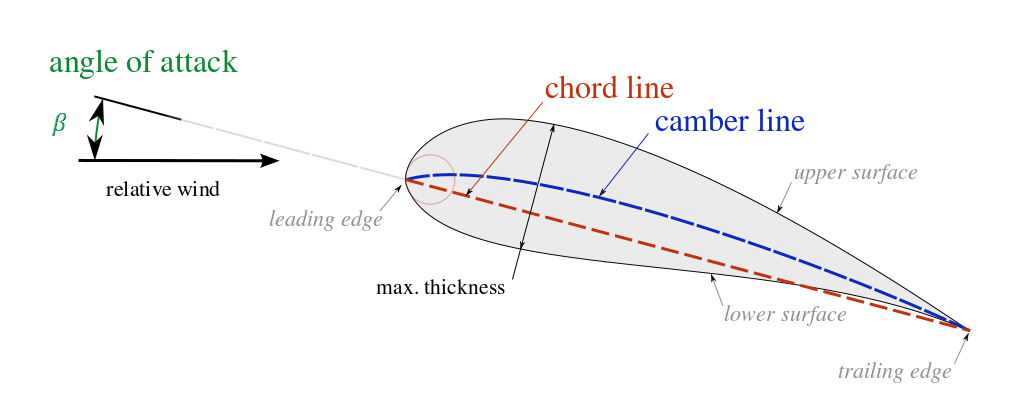


Abbildung X

Darstellung zur Verdeutlichung fundamentaler Begriffe in Bezug auf ein Profil

## 2.2 Algorithmus zur Erzeugung einer Profilgeometrie

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 des Profils. 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 Profils, welches über keine Skelettlinie verfügt, erzeugt werden, so entsprechen die Punkte der Dickenverteilung bereits der Saugseite eines Profils. Die Druckseite kann durch eine Invertierung der y Koordinaten aller Punkte der Dickenverteilung erreicht werden. Abbildung X zeigt die Dickenverteilung eines Profils der Serie NACA65/CA in Abhängigkeit der Sehnenlänge b.

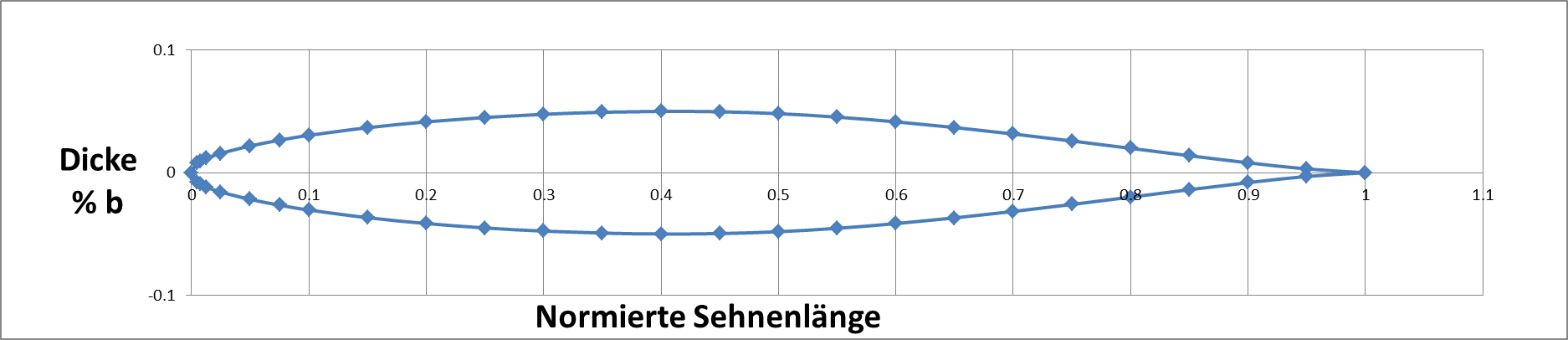
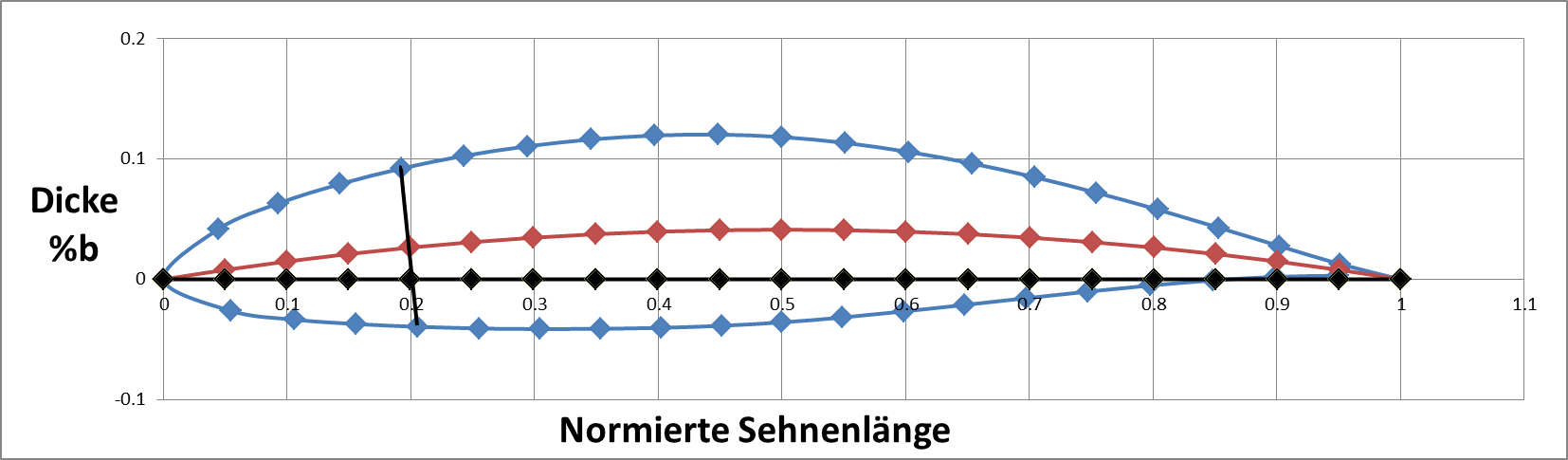


Abbildung X

Beispielhafte Dickenverteilung für die Druck und Saugseite eines nicht gekrümmten Profils der NACA65/CA Serie

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 Ps(xs|ys) aus in Richtung des Normalenvektors im Punkt Ps die Strecke mit einer Läge der Dicke zurückgelegt wird. Die Dicke geht aus dem funktionalen Zusammenhang der Dickenverteilung für die Koordinate xs 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 x veranschaulicht diesen Sachzusammenhang noch einmal und zeigt ein gekrümmtes Profil der NAC65/CA Serie, sowie dessen Sehne und Skelettlinie. Ein beispielhafter Punkt Ps befindet sich in der Abbildung an der Koordinate der normierten Sehnenlänge 0.2 und 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 parallel zueinander in radialer Richtung angeordnet. Die Profile werden in einem ersten Schritt gefädelt. Beim Fädeln wird jedes einzelne Profil im zweidimensionalen Raum 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 jeweiligen Profils sein. Eine Gerade durch diese Punkte liegt parallel zur radialen Achse im dreidimensionalen Koordinatensystem. Sie wird auch als Auffädelstrahl bezeichnet. Abbildung X zeigt die Zeichnung einer Verdichterschaufel, in der auch der Auffädelstrahl eingezeichnet ist.

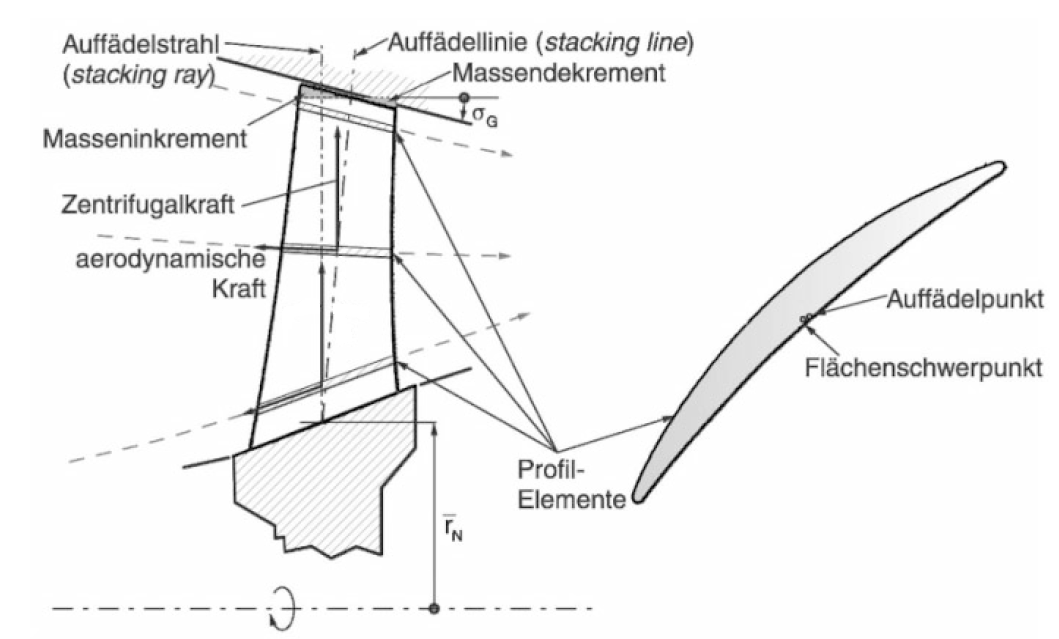


Abbildung X

Zeichnung einer Verdichterschaufel zur Darstellung des Auffädelstrahls und der Auffädellinie

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 auf, wenn dessen Vorderkante aus der Perspektive, bei der die Druckseite frontal betrachtet wird, abgerundet ist. Eine Neigung 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. Abbildung X 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 X 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 ersichtlich.

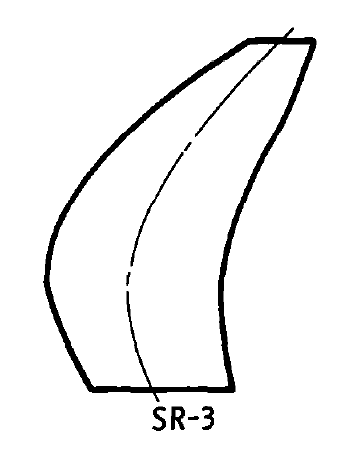


Abbildung X

Zeichnung des SR-3 Propellers und der Auffädellinie

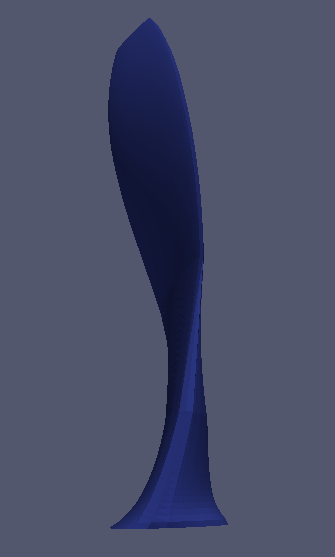


Abbildung X

CAD-Modell des SR-3 Propellers

Beschrieben wird die Pfeilung und Neigung des Propellers über den Pfeilwinkel 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). 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. Abbildung X 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 sweep line bezeichnet und ist durch eine Linie mit kurzen, voneinander abgesetzten Strichen dargestellt.

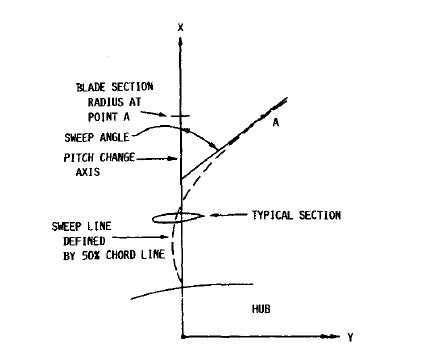


Abbildung X

Schaubild zur Verdeutlichung des Pfeilwinkels

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 über eine Nabe mit der Welle einer Turbine verbunden, welche für die Rotation sorgt. Der Abstand von der Rotationsachse der Welle in radialer Richtung zur Verbindung des Propellerblattes mit der Nabe (Hub) wird als Hubradius bezeichnet. Die Spitze des Propellerblattes wir auch als Tip bezeichnet.

# 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. 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 diese 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.

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. Für die Generierung von entsprechenden CAD Outputformaten, wie zum Beispiel STEP oder IGES, stehen ebenfalls entsprechende Funktionen bereit.

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 Schaufelgeometrien, welches in der Sprache C++ im Institut für Antriebstechnik in der Abteilung Fan und Verdichter entwickelt wurde. Um Schaufelgeometrien zu erzeugen, benötigt das Werkzeug einige Inputparameter, die über entsprechend formatierte Dateien geliefert werden müssen. Der Hauptanwendungsfall des BladeGenerators besteht darin, eine Optimierung von Schaufelgeometrien 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 anschließend in das Outputformat STEP oder TEC überführt werden. TEC ist das Datenformat, welches von der Visualisierungssoftware Tecplot verwendet wird.

## 3.2 Schnittstelle zu den Geometrieinformationen 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 X zeigt das Klassendiagramm, welches die in diesem Kapitel vorgestellte Klassenstruktur noch einmal darstellt.

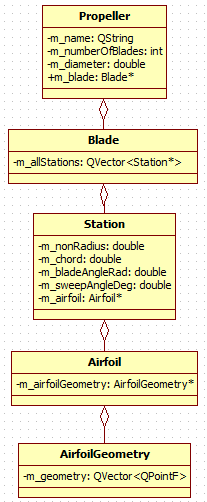


Abbildung X

Klassenstruktur der Schnittstelle zur geometrischen Information eines Propellers in Propster

## 3.3 Propster Experimentfiles

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.

Um die geometrischen Daten eines Propellers in die dafür vorgesehenen Datenstrukturen zu integrieren, 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 Experimentfiles 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 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 Attributs“ 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 definiert.

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 definiert. 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 definierten 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 Attribute die SI-Einheit und den Datentypen sowie den Wert des Durchmessers beinhaltet. Ein Element für die Anzahl der Schaufeln muss definiert 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 dieser Arbeit befindet sich ein beispielhaftes Experiment File.

## 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.

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 hierbei 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 zu prüfen.

# 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 in einem ersten Schritt der Aufbereitung somit 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 für die 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 Blade Generator

Die Software BladeGenerator ist darauf ausgelegt Schaufelgeometrien 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 Profilgeometrie 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. 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. 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 Profilgeometrien, 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 STP und TEC zur Verfügung, welche von BladeGenerator generiert werden können.

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 Propellerblattgeometrien zu erzeugen.

## 4.3 Wahl eines der beiden Verfahren

Bei der Auswahl des Verfahrens zur Mantelflächengenerierung wurden mehrere Faktoren berücksichtigt. Diese sollen nun betrachtet werden.

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 wird 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 werden 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 Profilgeometrien erfolgen. Durch die Approximation sind geringfügige Abweichungen von den Originalgeometrien nicht zu vermeiden. Auch die spätere Überführung der vier B-Spline Kurven in eine Kurve birgt ein Potential zu weiteren Abweichungen. Zu erwähnen ist an dieser Stelle noch einmal, dass BladeGenerator das Ziel verfolgt Profilgeometrien 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 Profilgeometrien 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 Schaufelgeometrie zu erzeugen. Somit ist bereits die Aufbereitung der normierten Geometrieinformationen, die von der Schnittstelle von Propster bereitgestellt werden, mit mehr Aufwand verbunden. Im Vergleich zur Nutzung von BladeGenerator müssen die normierten Profilgeometrien 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 Propellerblattgeometrien zu ermöglichen, müsste die Routine, die die Funktionalität des Fädelns der Profile bereitstellt, flexibel auf die verschiedenen Varianten des Fädelns anpassbar implementiert werden und eine Erweiterbarkeit um weitere Varianten 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 die Generierung von CAD-Output Dateien, welche es ermöglichen, den erstellten Propellerblattmantel zu visualisieren, müsste implementiert 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 beschrieben 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 Originalgeometrie 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 Profilgeometrien in Bezug auf die Originalgeometrie. 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 Schaufelgeometrie 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 Schaufelgeometrie 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 zur Validierung der bereits in Propster implementierten zweidimensionalen Strömungsrechnung dienen. Eine exakte Modellierung der Originalgeometrie ist von großer Bedeutung. Abweichungen von der Originalgeometrie würden eine Validierung der zweidimensionalen Strömungsrechnung erschweren und zu verfälschten Ergebnissen führen. Damit würde das Gesamtziel dieser Arbeit in Gefahr gebracht werden. Dem ist mit der Auswahl von Open Cascade Technology entgegen gewirkt worden.

# 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 Kapitel Schnittstelle zu den Geometrieinformationen eines Propellers von Propster 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 Profilgeometrien 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 Profilgeometrien, 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 Profilgeometrie 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 Prot(xrot|yrot) ergeben sich aus der Multiplikation mit der Rotationsmatrix aus Gleichung 1.

(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 Arten. 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 beschrieben 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 pchord der Sehnenlänge b der einzelnen Profile gefädelt werden soll. Von jedem Profil wird dann der Punkt Pchord(xchord|ychord) 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 Pfirst(xfirst|yfirst) 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 dem Erkenntnis, dass sich die Vorderkante jedes Profils im Ursprung befindet und die Koordinaten der Hinterkante im ersten Punkt der Punktelisten, die die Profilgeometrien beschreiben, hinterlegt sind, kann der gesuchte Punkt Pchord bestimmt werden (siehe Gleichung 2).

(2)

Nachdem der Punkt Pchord 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 Pchord 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. Abbildung X veranschaulicht den Vorgang des Fädelns über die Sehne noch einmal anhand eines Profils.

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

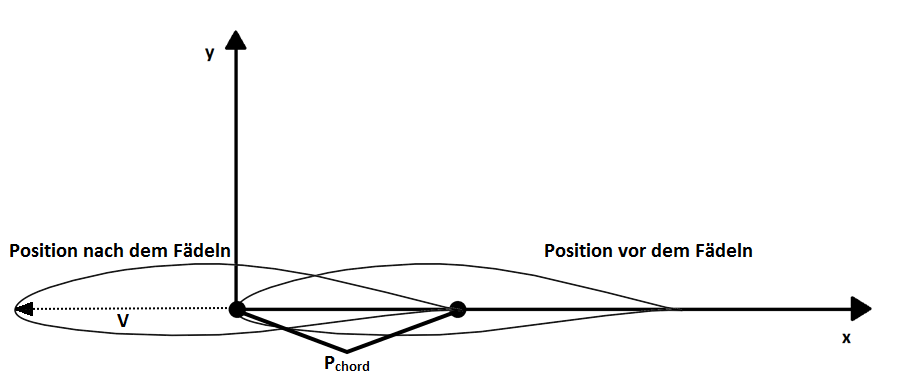


Abbildung X

Schaubild zum Vorgang des Fädelns über die Sehne

#### 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 Pchord 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. 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 X zeigt einen beispielhaften Polygonzug eines Profils.

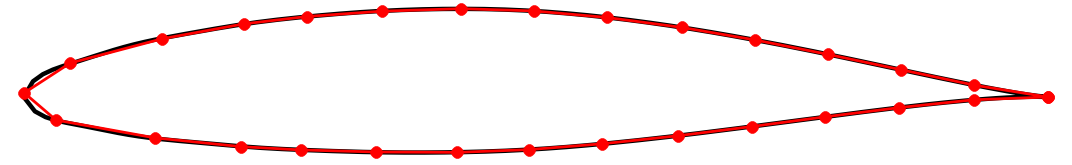


Abbildung X

Polygonzug eines Profils

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

(3)

(4)

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

(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 V 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, die parallel zur radialen Achse verläuft, wird das Propellerblatt mit einer Pfeilung und Neigung versehen. 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 X zeigt ein Schaubild mit zwei Profilen, die über 50% der Sehnenlänge gefädelt wurden, sowie die dazugehörende Auffädellinie (radiale Achse).

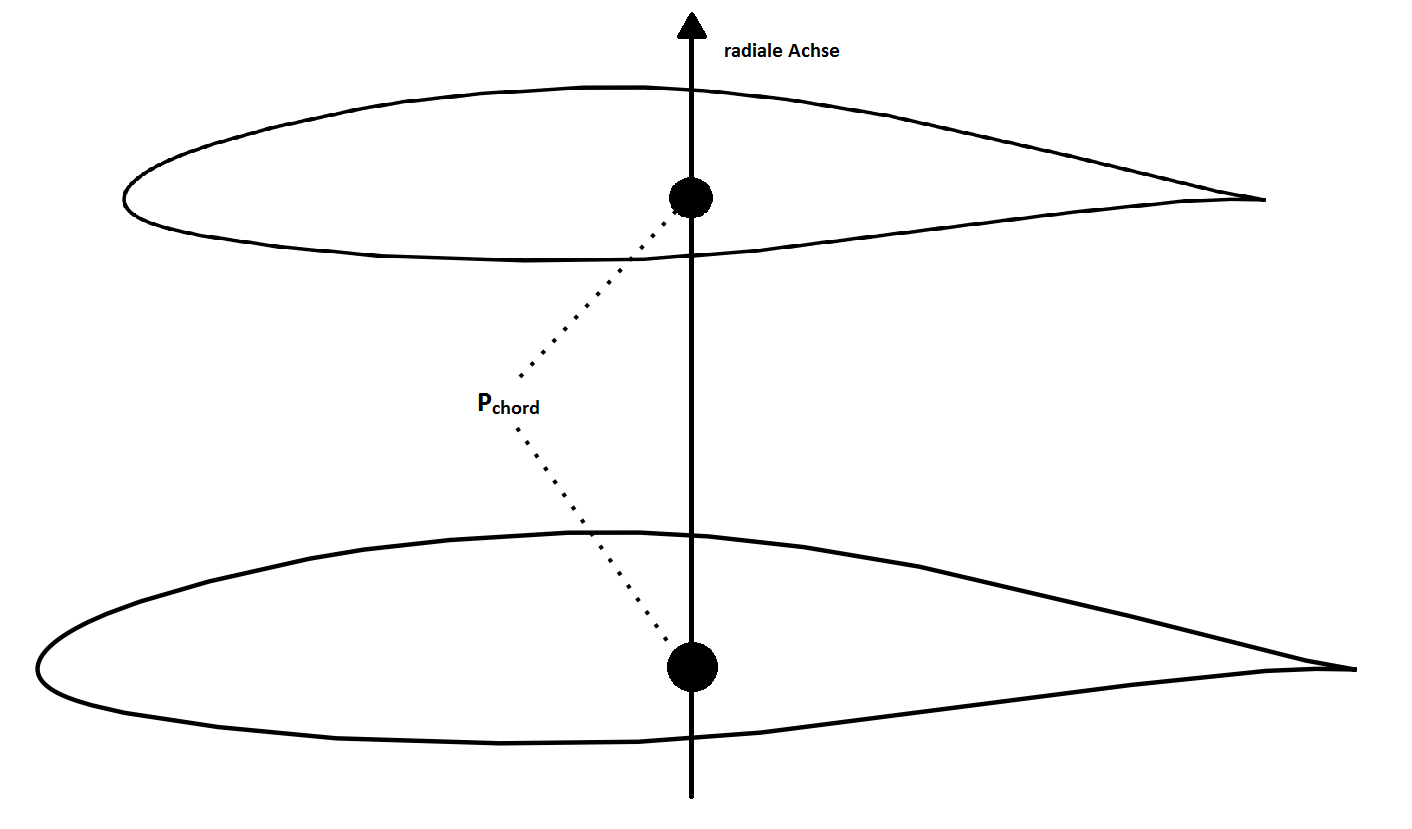


Abbildung X

Schaubild zur Darstellung der Auffädellinie beim Fädeln über die Sehne

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 X zeigt beispielhaft die auf den Radius von einem Meter normierte Auffädellinie der Schaufel des NASA SRIII Propellers.

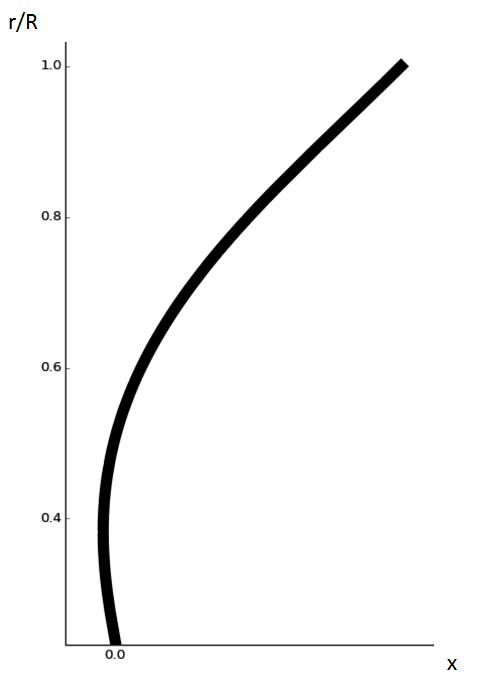


Abbildung X

Normierte Auffädellinie der Schaufel des SRIII Propellers

Die Auffädellinien, die in dieser Arbeit betrachtet werden, beginnen an der radialen Koordinate des Hubs. Bei der Auffädellinie aus Abbildung X liegt der Hub bei dem 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 Pstack könnte die Vorder- oder Hinterkante des Profils sein oder auch der in dem vorigen Abschnitten erwähnte Punkt auf der Sehne Pchord beziehungsweise der Flächenschwerpunkt. Das Profil wird in diesem Fall jedoch nicht derart verschoben, dass sich Pstack im Ursprung befindet. Das Profil wird lediglich in x-Richtung verschoben. Die Verschiebung (sweep) für ein Profil, welches sich an der Radialen Koordinate r befindet, ergibt sich aus der Differenz der x-Koordinate von Pstack (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 V . Soll das Propellerblatt neben der Pfeilung auch mit einer Neigung versehen werden, so ist der Vektor V noch um den Anstellwinkel des Profils zu rotieren (siehe Gleichung 1). Der daraus resultierende Vektor Vrot gibt die Verschiebung in x- und y-Richtung für alle Punkte vor, die die Geometrie des entsprechenden Profils beschrieben. 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 wurde 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 generiert.

### 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.

(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 werden 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. Abbildung X veranschaulicht das Prinzip des Brent-Verfahrens noch einmal.

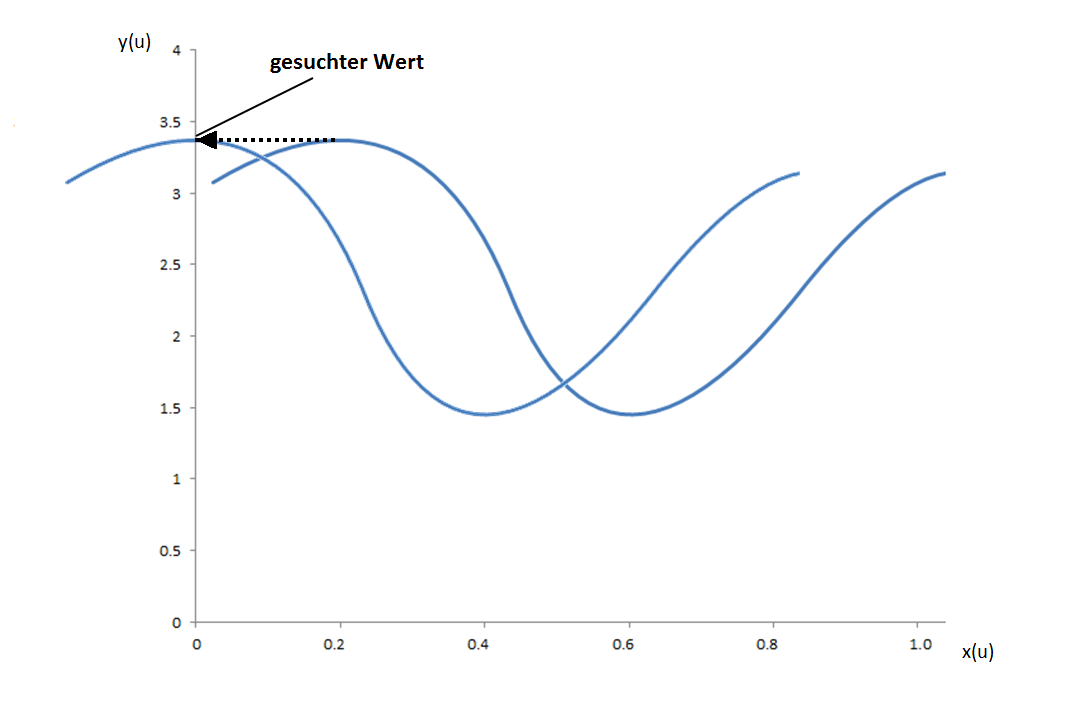


Abbildung X

Schaubild zum Prinzip des Brent-Verfahrens

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 PHub(0.0|r/RHub). Von diesem Punkt aus beginnt der iterative Abschnitt dieses Verfahrens.

Der Pfeilwinkel Λ an der Stelle r/RHub 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 PHub zur Definition der Auffädellinie folgt, hat somit einen Wert von r/RHub + diff. Über den Abstand diff und den Pfeilwinkel Λ kann dann mit Hilfe von Gleichung 7 die x-Koordinate des Punktes, der auf PHub folgt, bestimmt werden.

(7)

Abstrakt formuliert beschreibt Gleichung 8 und 9, wie die x- und y-Koordinate eines Punktes Pi+1(xi+1|yi+1) über den Punkt Pi(xi|yi) und den Pfeilwinkel Λi an der radialen Koordinate yi bestimmt werden kann. Die beiden Punkte beschreiben die zu rekonstruierende Auffädellinie. In der folgenden Iteration dient der ermittelte Punkt Pi+1 als neuer Ausgangspunkt Pi. Die Iteration erfolgt so oft, bis die dimensionslose radiale Koordinate eins (r/Rtip) erreicht wurde.

(8)

(9)

Ist der Pfeilwinkel größer null, so ist in Gleichung 8 eine Subtraktion auszuführen. Andernfalls muss Addiert werden. Das Prinzip einer Iteration basiert auf der Winkelfunktion des Tangens (Gleichung 10). Abbildung X verdeutlicht dies noch einmal.

(10)

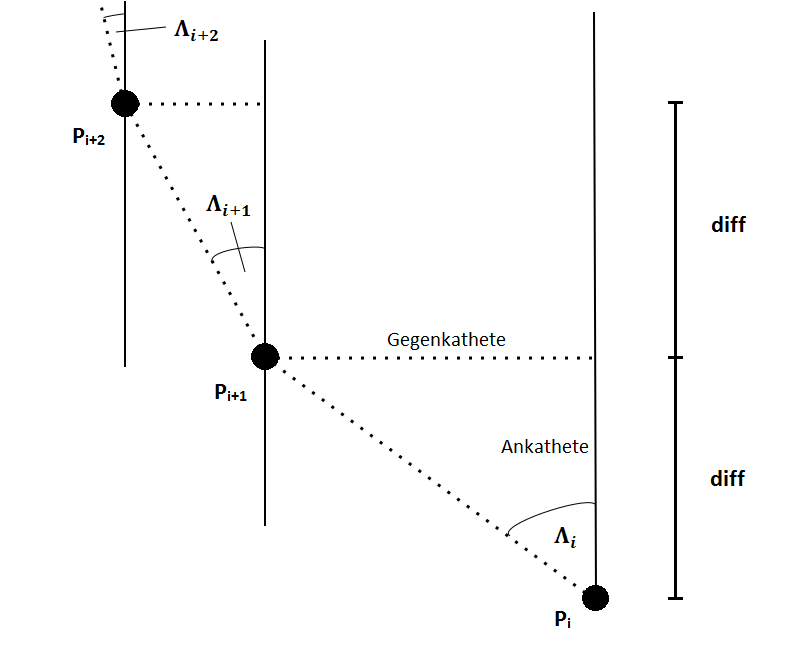


Abbildung X

Schematische Zeichnung zur Verdeutlichung eines Iterationsintervalls zur Rekonstruktion der Auffädellinie eines Propellerblattes

Nachdem die festgelegte Anzahl an Iterationen i erfolgt ist, liegen i+1 Punkte vor, die die auf den Radius von eins skalierte Auffädellinie definieren. 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 PHub und Endpunktes PTip der zu erzeugenden Auffädellinie benötigt, sowie die Pfeilwinkel an den radialen Koordinaten der beiden Punkte (ΛHub und ΛTip). Abbildung X zeigt die benötigten Inputparameter in einem Schaubild mit der aus diesen Parametern generierten Auffädellinie.

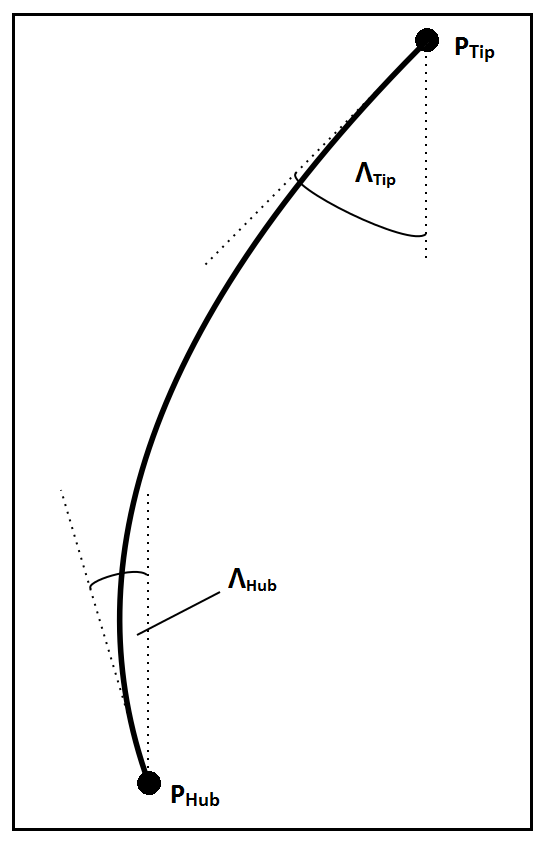


Abbildung X

Generierte Auffädellinie anhand weniger Parameter

Die in Abbildung X dargestellte Auffädellinie wird in Form eines eingespannten B-Splines (clamped B-Spline) generiert. Eingespannte B-Splines mit n Kontrollpunkten [P1, P2, …, Pn] besitzen eine wesentliche Eigenschaft, die der Algorithmus zur Generierung der Auffädellinie ausnutzt. Die Koordinaten des ersten und letzten Kontrollpunktes (P1 und Pn) sind gleich den Koordinaten des Start- und Endpunktes des B-Splines, welchen 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 definiert wird. Der erste und dritte Kontrollpunkt wird durch die Inputparameter PHub und PTip definiert. Über diese beiden Punkte und die Pfeilwinkel (ΛHub und ΛTip) lässt sich der zweite, noch fehlende Kontrollpunkt jedoch errechnen, indem die Vektorgeradengleichungen der Tangenten im Punkt PHub und PTip bestimmt und deren Schnittpunkt berechnet wird. Der Schnittpunkt entspricht dem gesuchten Kontrollpunkt und lässt sich durch Gleichung 11 bestimmen:

(11)

Es lassen sich zwei Gleichungen mit den beiden Unbekannten r und s aufstellen (siehe Gleichung 12 und 13). Löst man auf r und s auf und setzt die Ergebnisse ein, so lässt sich der gesuchte Schnittpunkt ermitteln.

(12)

(13)

Abbildung X veranschaulicht diesen Zusammenhang noch einmal.

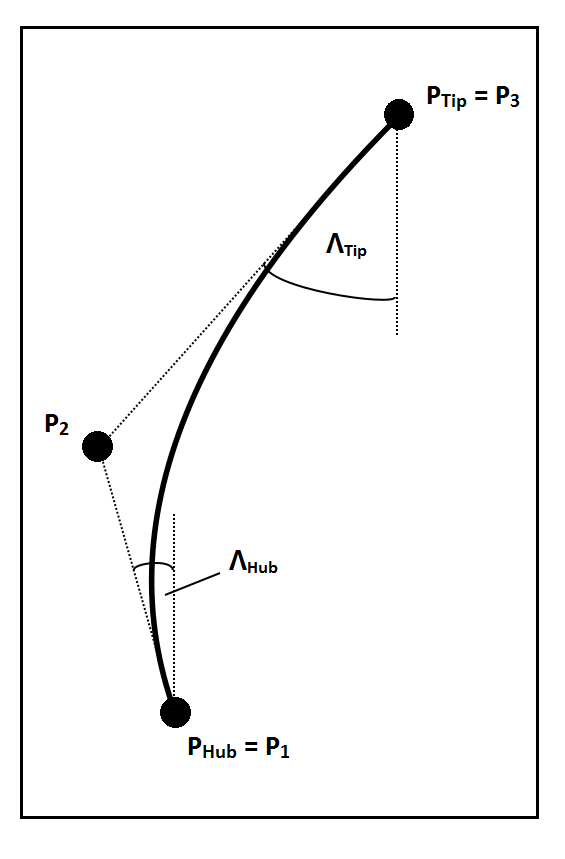


Abbildung X

Schaubild zur Veranschaulichung der Ermittlung des gesuchten Kontrollpunktes P2

Anschließend lässt sich der B-Spline über die Kontrollpunkte, den Grad und den Knotenvektor definieren. Der Grad des B-Splines ist festeglegt 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 [NURBSBOOK] 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. Der Algorithmus ist somit 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 Algortithmen ist es möglich die normierten Profilgeometrien, welche von der Schnittstelle von Propster geliefert werden, zu transformieren, sodass die resultierenden Profilgeometrien die Originalform des Propellerblattes in Originalgröße beschreiben. Die transformierten Profilgeometrien 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 Profilgeometrien zu erzeugen. Informationen zu den genutzten Klassen von Open Cascade Technology wurden aus der Dokumentation entnommen.

Die transformierten Profilgeometrien 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 Profilgeometrien 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 Zeitpunk in diesem Kapitel erwähnt.

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.

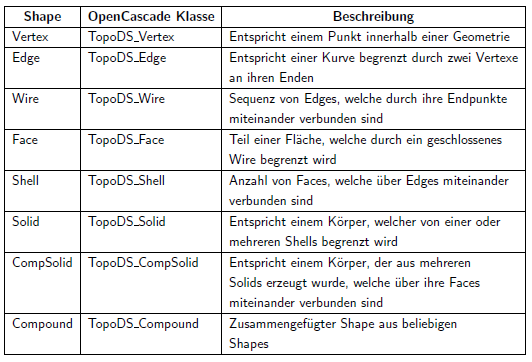


Tabelle 1

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 betrachtet aus der generierten Fläche einen Festkörper (TopoDS\_Solid) erzeugen zu können. Mit dem Festkörper könnte die dreidimensionale Strömungsrechnung ebenfalls durchgeführt werden. Zusätzlich ist es jedoch 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 X zeigt eine beispielhafte Mantelfläche, die aus einigen Profilen generiert wurde. Die Geometrie in dieser Abbildung dient lediglich zur Veranschaulichung der Öffnungen.

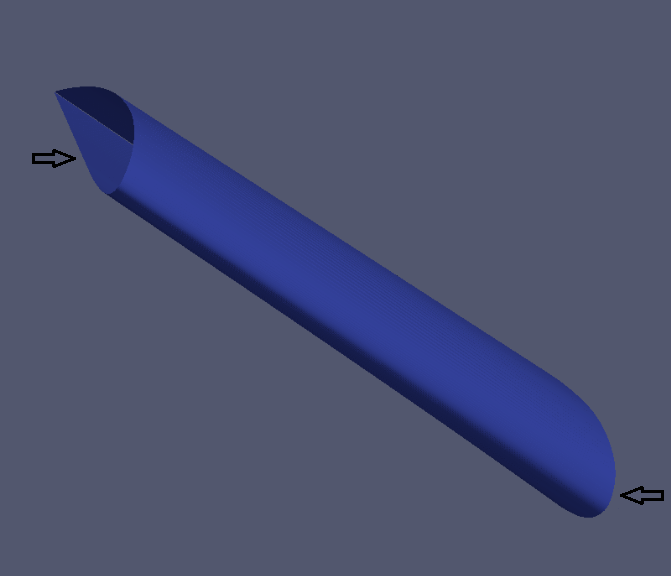


Abbildung X

Mantelfläche zur Veranschaulichung der Öffnungen an Hub und Tip

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 und der letzte Punkt einer Zeile gleich dem ersten Punkt ist. 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 Output Dateien

Für die Generierung der Dateien in dem Outputformat 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 Geometrieinformationen 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 geometrischen 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 die 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. 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 zur 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 werden können 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. X). 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. Propeller schaufeln sind in mehreren Schichten aufgebaut, welche sich aus verschiedenen Materialien, wie zum Beispiel Leichtbaustoffen, zusammensetzen. Bei einer Gewichtsabschätzung einer Propellerschaufel wären beispielsweise die Volumen der einzelnen Schichten zu ermitteln. Neben dem 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.

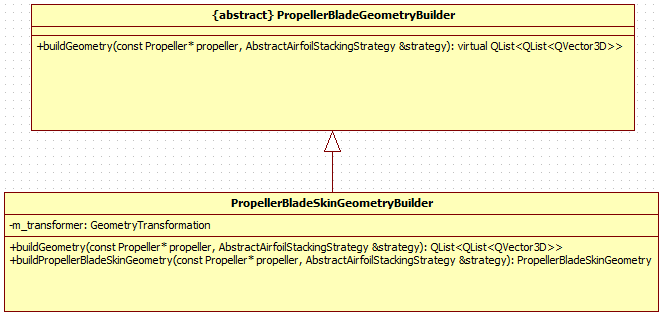


Abbildung X

Klassendiagramm der Klasse AbstractPropellerBladeGeometryBuilder und PropellerBladeSkinGeometryBuilder

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 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. X). 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, rotieren und verschieben (siehe Abb. X).

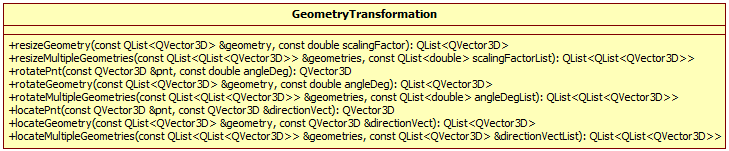


Abbildung X

Klassendiagramm der Klasse GeometryTransformation

Ein Objekt dieser Klasse ist als Membervariable (m\_transformer) in der Klasse PropellerBladeSkinGeometryBuilder hinterlegt, sodass deren Funktionalität bereitgestellt wird (siehe Abb. X). 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 und 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 und dem Fädeln, bei dem neben der Pfeilung auch eine Neigung der Propellerschaufel erzeugt wird. Abbildung X zeigt das Klassendiagramm der Strategien zum Fädeln der Profile

.

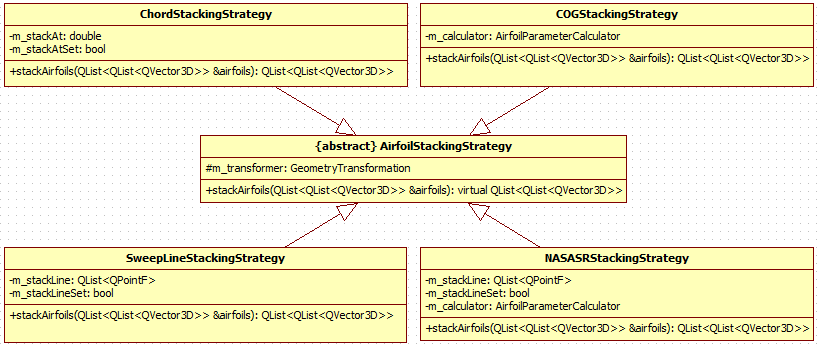


Abbildung X

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 X zeigt das Klassendiagramm dieser Klasse.

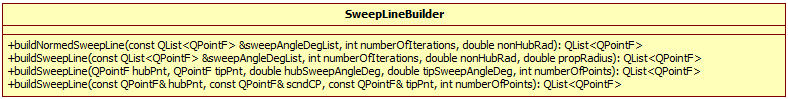


Abbildung X

Klassendiagramm der Klasse SweepLineBuilder

In dem Klassendiagramm ist zu erkennen, dass die Klasse SweepLineBuilder vier Funktionen zur Generierung von Auffädellinien zur Verfügung stellt. 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 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 übergeben und der dimensionslose Hubradius.

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 (hubPnt, scndCp, tipPnt) des B-Splines, der die Auffädellinie beschreibt, übergeben werden, sowie die Anzahl an Punkten (numberOfPoints), 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 liefert 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 X zeigt das Klassendiagramm von AirfoilParameterCalculator.

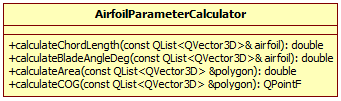


Abbildung X

Klassendiagramm der Klasse AirfoilParameterCalculator

So greift die Klasse COGStackingStrategy auf die Funktionen zur Ermittlung des Flächeninhalts und des geometrischen Schwerpunktes zu (siehe Gleichung 3-5). Die Klasse NASASRStackingStrategy nutzt die Funktion zur Ermittlung des Anstellwinkels eines Profils calculateBladeAngleDeg().

Die Funktion stackAirfoils() der Strategien zum Fädeln der Profile wird von der Funktion buildGeometry() der Klasse PropellerBladeSkinGeometryBuilder aufgerufen. Anschließend liegt die Schaufelgeometrie 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 von weiteren Strategien zum Fädeln gestaltet sich als unkompliziert, indem eine weitere Klasse, die von AbstractAirfoilStackingStrategy erbt, implementiert werden könnte.

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 Geometrieinformationen dient und die geordneten Punktelisten als Membervariable beinhaltet. Der Builder stellt aus diesem Grund auch eine Funktion buildPropellerBladeSkinGeometry() bereit (siehe Abb. X), die ein Objekt der Klasse PropellerBladeSkinGeometry erzeugt und diesem eine Geometrie übergibt, die durch die Funktion buildGeometry() erzeugt wurde.

Die Klasse PropellerBladeSkinGeometry erbt von der Klasse AbstractPropellerBladeGeometry. 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, die 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 Propellerblattgeometrien 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 X zeigt das Klassendiagramm der Klassen, die als Datenstruktur dienen um die aufbereiteten geometrischen Informationen von Propellerschaufeln bereitzustellen.

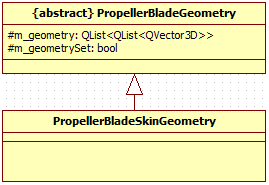


Abbildung X

Klassendiagramm der Klassen AbstractPropellerBladeGeometry und PropellerBladeSkinGeometry

Zuasammengefasst erzeugt die Schnittstelle PropellerBladeSkinGeometryBuilder aus den nomierten 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 die Mantelfläche der Propellerschaufel 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, welches kompatibel zu Open Cascade Technology ist, überführt werden. Diese Aufgabe übernimmt die Klasse OpenCascadeGeometryAdapter. Das Klassendiagramm ist in Abbildung X hinterlegt.

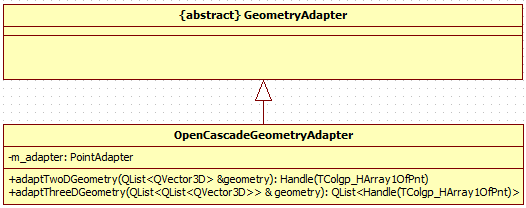


Abbildung X

Klassendiagramm der Klassen AbstractGeometryAdapter und OpenCascadeGeometryAdapter

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 X zeigt das Klassendiagramm der Klasse PointAdapter.

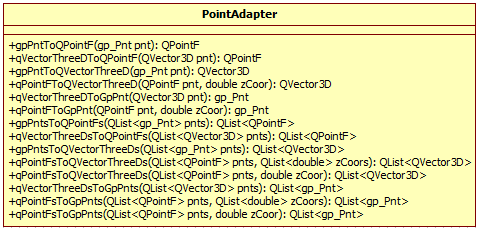


Abbildung X

Klassendiagramm 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 Adapterinterface 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. 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 auch 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 auch 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 X zeigt das entsprechende Klassendiagramm.

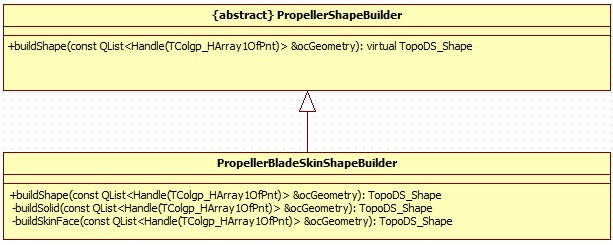


Abbildung X

Klassendiagramm der Klassen AbstractPropellerShapeBuilder und PropellerBladeSkinShapeBuilder

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 erzeugen 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. Auch in diesem Fall die die Abstraktion 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 X 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 auch hier mit AbstractShapeExporter eine 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 auch 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 X zeigt das Klassendiagramm der Export-Klassen.

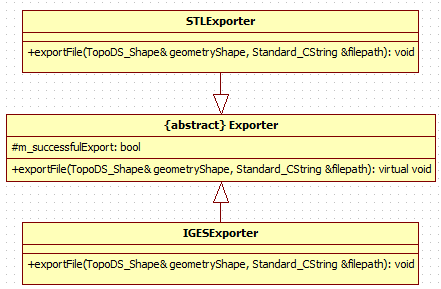


Abbildung X

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.

Nachdem die Klassenstruktur vollständig beschrieben wurde, sollen nun die Tests des implementierten Verfahrens zur Generierung von Mantelflächen genauer betrachtet werden.

# 6 Tests im 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 aller implementierten Klassen prüft. Weiterhin wurden Anwendungstests ausgeführt, in deren Rahmen die Mantelflächen der Propellerschaufeln der Propeller SRII und SRIII der NASA modelliert wurden. Zunächst soll die eingerichtete Unittestsuite betrachtet werden.

## 6.1 Eingerichtete Unittestsuite

Mit Hilfe des Testing Frameworks Google Test wurden die implementierten Klassen und Strukturen mit Unittests versehen. Insgesmat 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 so ausgeführt, wie es vorgesehen ist. 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 weitere mögliche Fehlerquellen entdeckt und behoben werden. Auch kam es dazu, dass sich durch diese Tests Definitionsbereiche für Parameter ergaben, 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 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 zwei 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.

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.

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 der Auffädellinie des SRIII-Propellers 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 sollen Auffädellinien mit Hilfe des Algorithmus aus Kapitel 5.1.3.1 generiert werden, 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 [NASA] entnommen werden. Abbildung X stellt den Funktionalen Zusammenhang aus dieser Quelle dar.

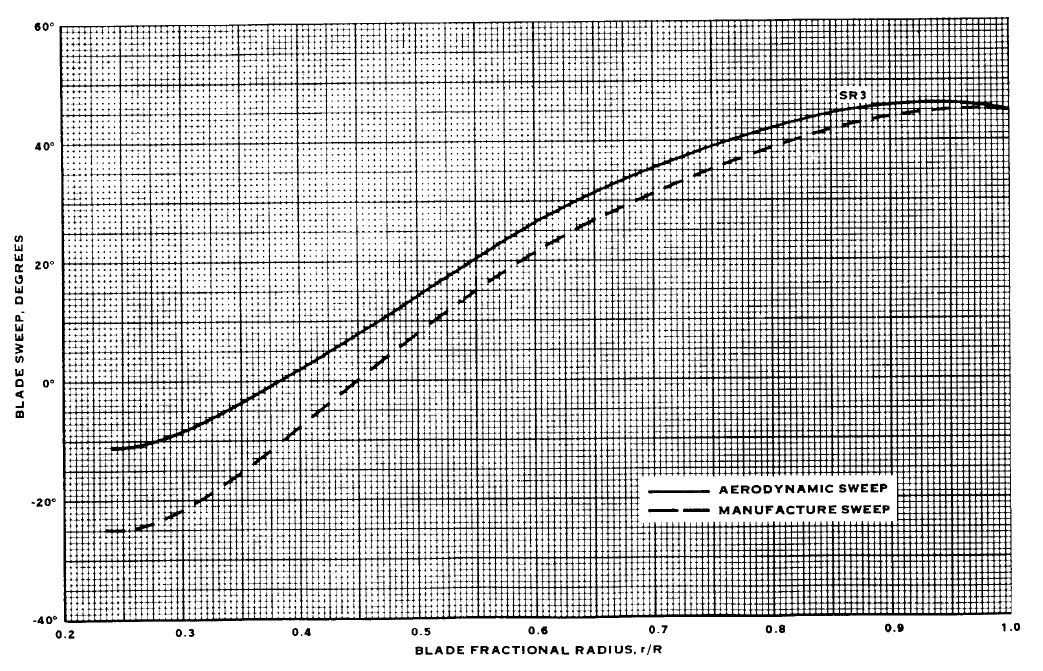


Abbildung X

Aerodynamischer und gefertigter Pfeilwinkel in Abhängigkeit des dimensionslosen Radius beim SRIII-Propeller [NASA]

In Abbildung X 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, während der aerodynamische Pfeilwinkel die Pfeilung des Propellerblattes beschreibt, während dieser 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-Propellers inklusive dessen Auffädellinie in Abbildung X dargestellt. Die Zeichnung liegt als Parallelprojektion vor. Die generierten Auffädellinien dieser Testreihe sollen mit der Auffädellinie aus Abbildung X 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 X 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 Pfeilwinkels in Abhängigkeit des dimensionslosen Radius mit einem möglichst geringen Ablesefehler zu gelangen, wurden diese mit Hilfe des Open Source Programms Enguage Digitizer digitalisiert und anschließend für die Testreihe eingelesen. Der dimensionslose Hubradius des SRIII-Propellers wurde in der Quelle [NASA] 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 X zeigt die generierten Auffädellinien des SRIII-Propellers, die mit 10, 100, 1000, 10000 und 100000 Iterationen erzeugt wurden. Aus der Abbildung geht nicht ersichtlich hervor, dass innerhalb dieser fünf Auffädellinien dargestellt werden. Zu erkennen ist lediglich, dass der Verlauf der Auffädellinie links, die mit 10 Iterationen generiert wurde, sich von den anderen vier Verläufen deutlich unterscheidet. Dass auch die Verläufe der anderen vier Auffädellinien geringfügig voneinander abweichen ist nur durch eine Vergrößerung eindeutig ersichtlich.

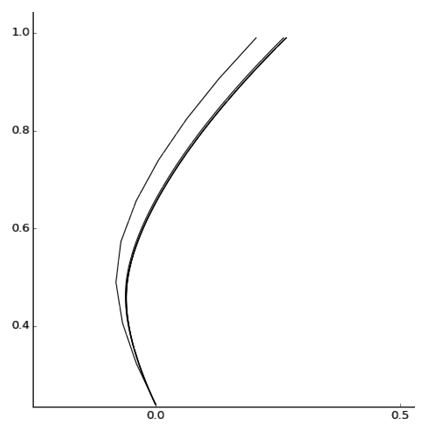


Abbildung X

Auffädellinien des SRIII-Propellers mit unterschiedlicher Anzahl an Iterationen

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

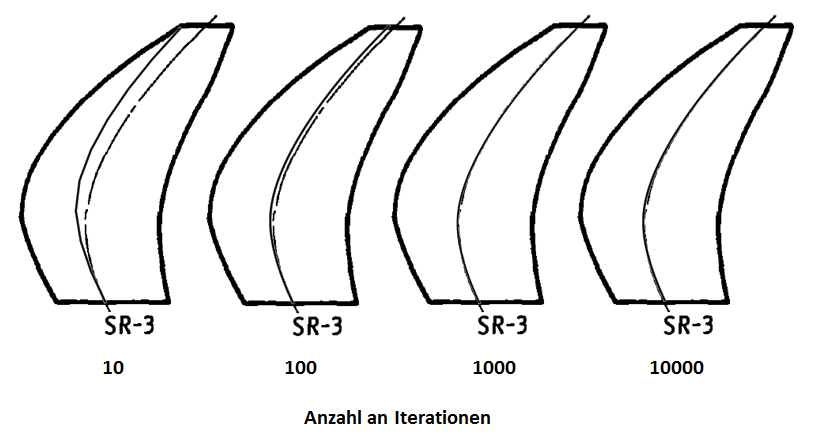


Abbildung X

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

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 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 Übereinstimmung der generierten Auffädellinien mit der Zeichnung aus Abbildung X, 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 Propellergeometrien zu definieren (siehe Kapitel 2), beschrieben werden. In der Testreihe wird die Bibliothek auf zwei Aspekte geprüft. Zum einen sollen die generierten Mantelflächen mit den originalen Propellerschaufeln abgeglichen werden. Zum anderen 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.

#### 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 Radius zwischen hub und tip anzugeben. Die Informationen zu diesen Parametern sind den Abbildungen X und X für die beiden Propeller zu entnehmen.

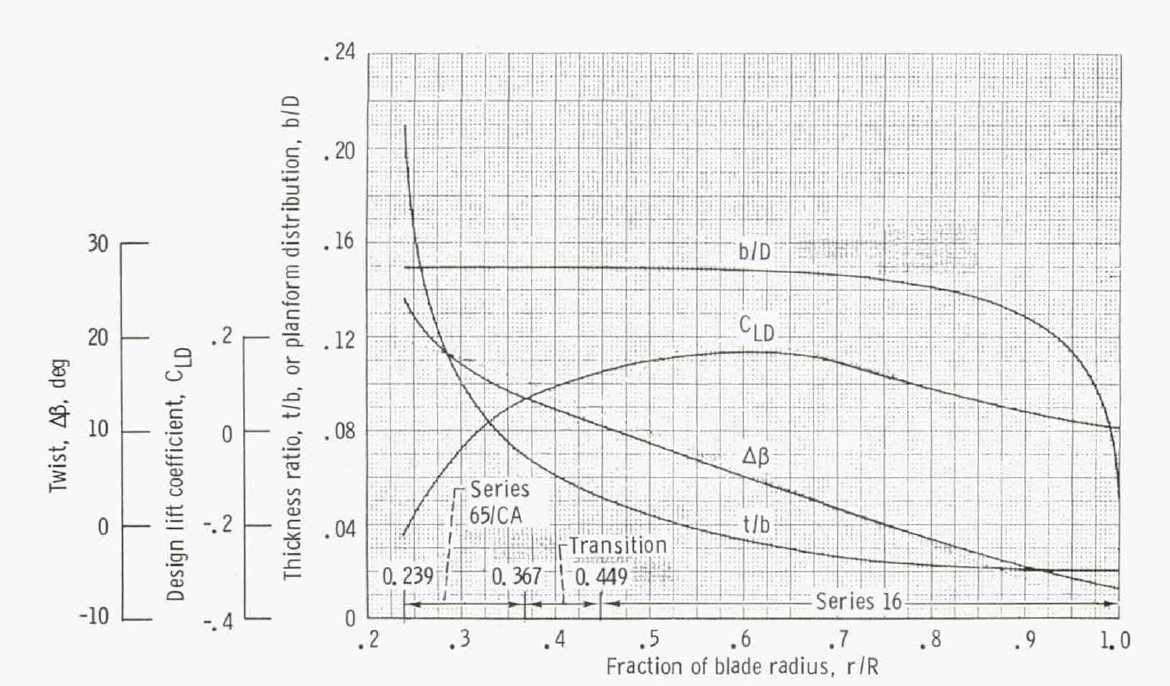


Abbildung X

Designparameter des SRII-Propellers in funktionaler Abhängigkeit des dimensionslosen Radius [NASA]

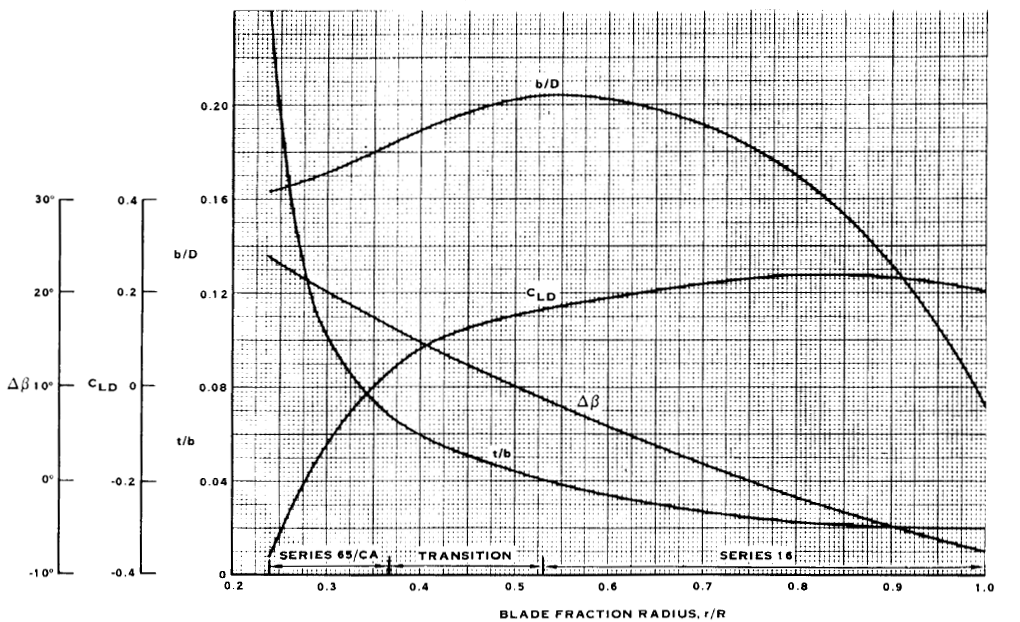


Abbildung X

Designparameter des SRII-Propellers in funktionaler Abhängigkeit des dimensionslosen Radius [NASA]

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
4. Verhältnis der maximalen Dicke eines Profils t zur Sehnenlänge b

Die Abbildungen sind auch in weiteren Quellen der NASA hinterlegt [NASA][NASA]. Die Daten wurden analog zu den Daten der Pfeilwinkel aus Abbildung X mit Hilfe von Enguage 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 m angegeben[NASA][NASA]. Um die Daten des Pfeilwinkels in das jeweilige Geometry File zu integrieren wurden für den SRIII-Propeller die digitalisierten Daten aus Abbildung X 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 Profilgeometrien, die durch eine feste Anzahl an geordneten Punkten beschrieben werden. Zwischen diesen Profilgeometrien 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 Profilgeometrien im Experiment File vorzugeben.

Aus den Abbildungen X und X kann man entnehmen, dass für die Propeller SRII und SRIII Profile der Serie NACA 16 und NACA 65/CA genutzt wurden. In [NASA] und [NASA] wird dies bestätigt. Diesen Quellen und Abbildung X ist zu entnehmen in welchen Bereichen des dimensionlosen Radius welche Profilserie vorliegt. Mit Hilfe der Designparameter aus den Abbildungen X und X ist es möglich die normierten Profilgeometrien in Anlehnung an Kapitel 2.2 zu generieren. Im Folgenden soll erläutert werden, wie die Profilgeometrien 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 Profilgeometrie eines Profils der NACA 16 Serie werden in der Quelle [NACA] beschrieben. In dieser werden Gleichungen angegeben, mit denen sich die Dickenverteilung und die Skelettlinie konstruieren lassen. Die Dickenverteilung wird über die Gleichungen 14 und 15 beschrieben.

(14)

(15)

y1/2 ist hierbei die Dicke eines Profils ohne Skelettlinie ausgehend von der Stelle der auf die Länge eins normierten Sehne *x1/2* in Richtung des Normalen Vektors. *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 Propellers an einer bestimmten Stelle des dimensionslosen Radius des Propellerblattes kann aus einer der vier Kurvenverläufe aus Abbildung X beziehungsweise X entnommen werden. Zu beachten ist, dass Gleichung 14 für die Dickenverteilung benutzt wird bevor die maximale Dicke aus Richtung der Vorderkante des Profils im Punkt PmaxThick(xmaxThick | ymaxThick) erreicht wird (. Anschließend wird Gleichung 15 für die restliche Dickenverteilung bis zur Hinterkante genutzt (.

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.

(16)

Der entsprechende Design Lift Coefficient CLD kann an der entsprechenden Stelle des dimensionslosen Radius jeweiligen Propellers aus einer der vier Kurvenverläufe aus Abbildung X und X beziehungsweise deren digitalisierten Daten entnommen werden. Die Ableitung der Gleichung 16 lautet wie folgt.

(17)

Die Gleichung 17 wird benötigt um die Koordinaten des Normalen Vektors an den Punkten der Skelettlinie aus Gleichung 16 zu bestimmen. Dazu muss der Steigungsvektor um 90° nach links gedreht werden. Dies wird erreicht indem die Koordinaten des Steigungsvektors vertauscht werden und die neue x Koordinate invertiert wird. Daraus ergibt sich der Normalen Vektor.

Mit Hilfe dieser Informationen und dem Algorithmus, der in Kaptel 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.

Innerhalb einer intensiven 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 [] beschrieben (siehe Abb. X). Abbildung X zeigt das daraus resultierende Profil.

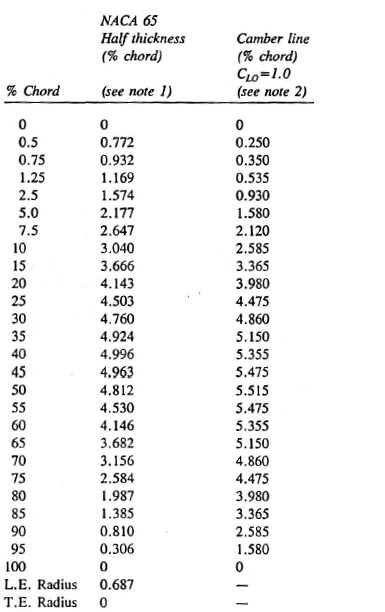


Abbildung X

Beschreibung der Dickenverteilung eines Profils der Serie NACA65/CA und dessen Skelettlinie in Abhängigkeit des prozentualen Anteils der Sehnenlänge

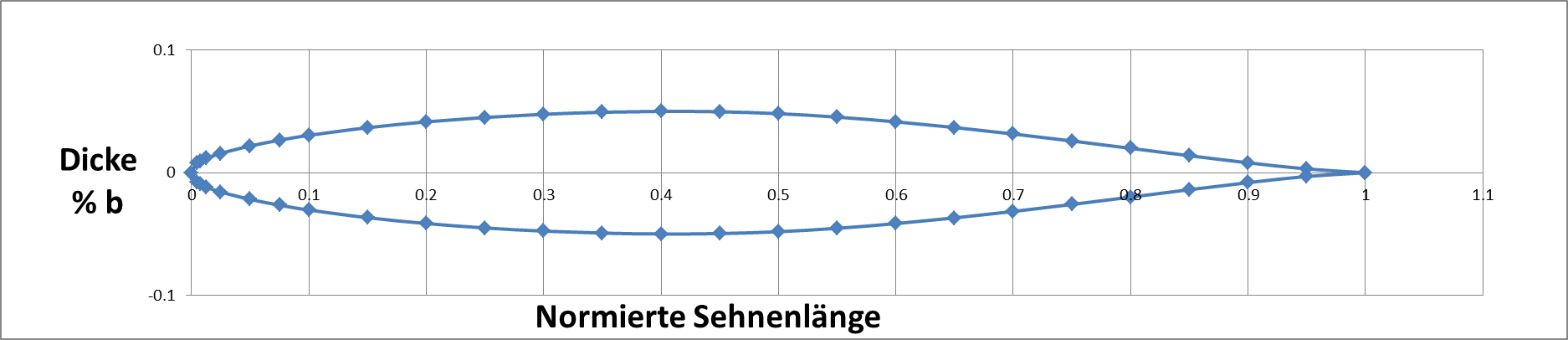


Abbildung X

Normiertes Profil der NACA 65/CA Serie ohne Skelettlinie

Das Profil aus Abbildung X wird als Standard dazu genutzt, um die NACA 65/CA Profile zu erzeugen, welche für das Modell des SRII 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 X beziehungsweise X durch einen Kurvenverlauf dargestellt wird, ermittelt werden, indem das Verhältnis an der entsprechenden Stelle des dimensionslosen Radius bestimmt wird und mit der Sehnenlänge *b* multipliziert wird. Man erhält die maximale Dicke in Prozent *t* des Profils in Metern. Die maximale Dicke tritt bei Profilen der NACA 65/CA Serie bei 40 % der Sehnenlänge auf []. Bei dem Standardprofil beträgt sie 0,04996 Meter (vergleiche Abb. X). Der Faktor *yScal* mit dem die y Koordinaten des Standardprofils somit skaliert werden müssen ergibt sich aus folgender Gleichung.

(18)

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 [] ist beschrieben, dass die Skelettlinie dieser Profilserie durch einen Kreisbogen beschrieben wird. Abbildung X zeigt eine Zeichnung, in der die Skelettlinie der Profilserie NACA 65 mit der Skelettline der Profilserie NACA65/CA verglichen wird.

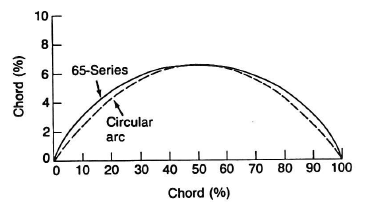


Abbildung X

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

Wie der Kreisbogen, der als Skelettlinie der NACA 65/CA Serie dient, generiert werden kann, wird nun beschrieben. Die Skelettlinie der NACA 65/CA Profilserie wird in Abhängigkeit des Design Lift Koeffizienten *CLD* generiert. Abbildung X zeigt einen funktionalen Zusammenhang zwischen dem Mittelpunktwinkel des zu generierenden Kreisbogens und dem Design Lift Koeffizienten *CLD*.

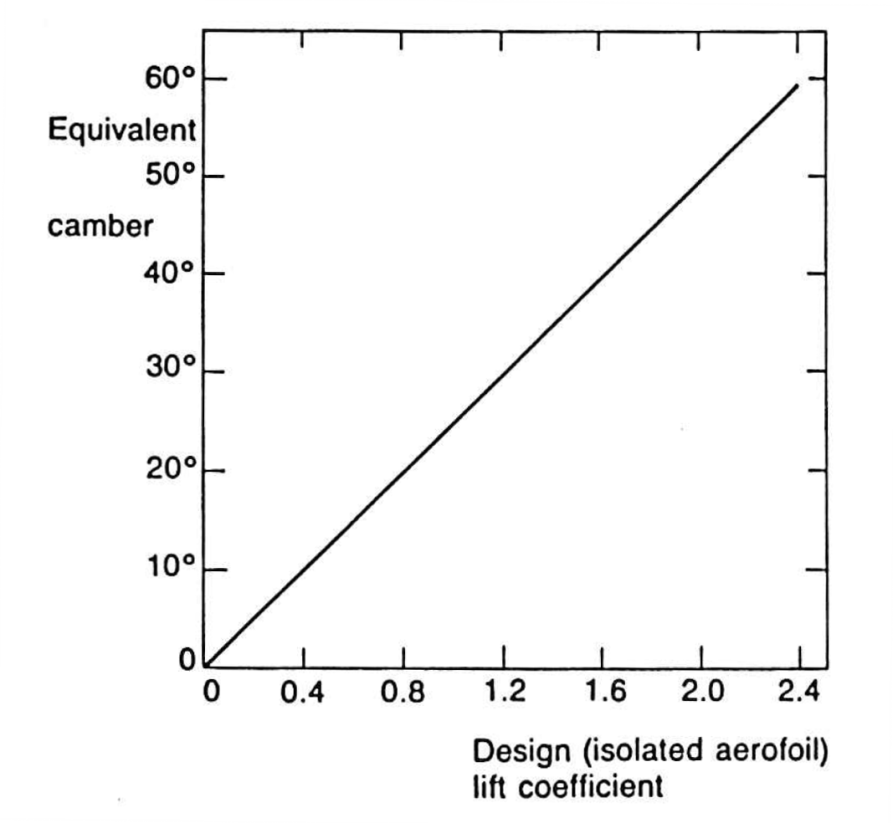


Abbildung X

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

Aus Abbildung X 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 X 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 Abbildung X hervor. Aus Abbildung X lassen sich zusätzlich die Punkte PVK(0 | 0) und PHK(100 | 0) ablesen, durch welche der Kreisbogen verläuft. Der Kreisbogen ist nun durch drei Punkte definiert. Dadurch lässt sich der Mittelpunkt M(50 | -223,897078) des Kreises eindeutig bestimmen. Auf die einzelnen Rechenschritte wird an dieser Stelle verzichtet.

Der Radius des Kreises ergibt sich somit aus der Summe des Betrags der y Koordinaten von M und S.

(19)

Bestimmt werden kann dadurch auch der Mittelpunkt Winkel (siehe Abb. X).

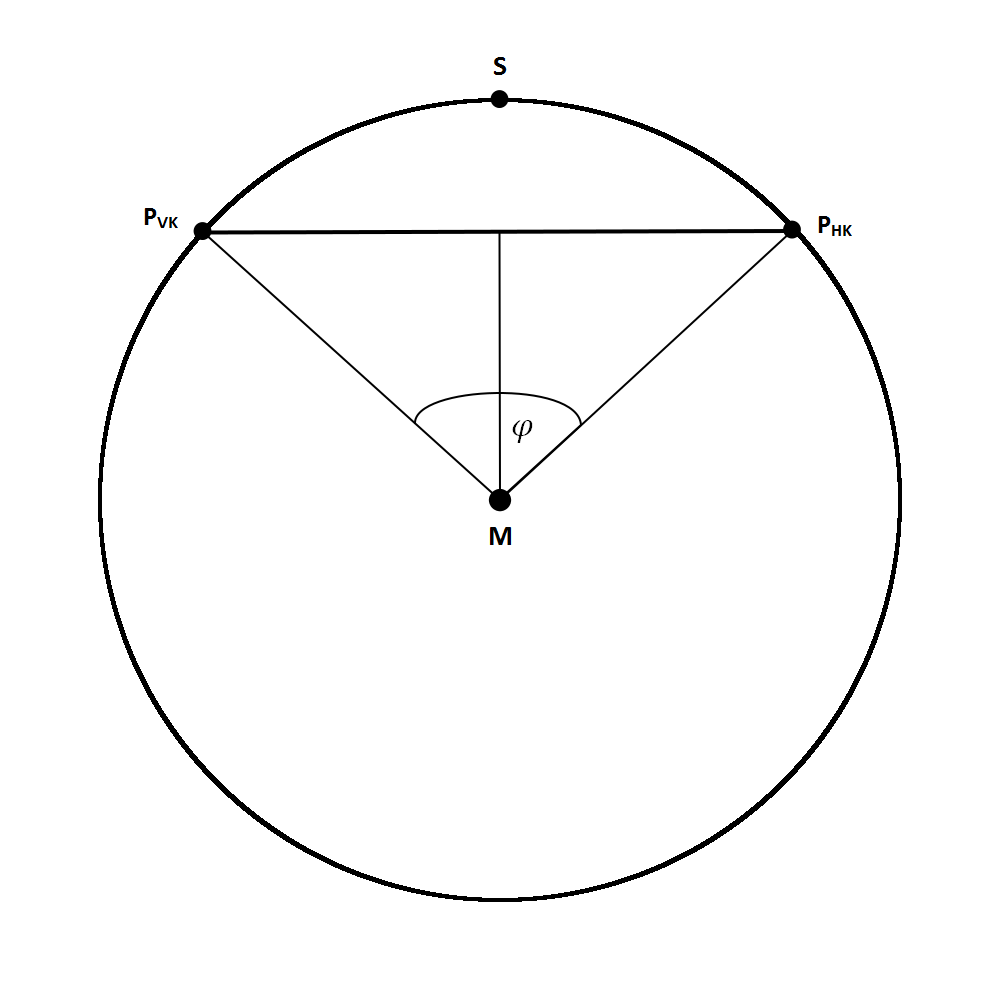


Abb. X

Schematisches Schaubild zur Bestimmung des Winkels

(20)

(21)

Der Mittelpunkt Winkel beträgt demnach für einen Kreisbogen als Skelettlinie für NACA 65/CA Profile mit einem Design Lift Koeffizienten *CLD* von eins 25,17712°. Damit ergibt sich folgende funktionale Abhängigkeit.

(22)

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 des Kreises für den entsprechenden Kreisbogen bestimmen.

(23)

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

(24)

Gleichung 24 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 erzeugt. Dabei wird dieser zunächst mit dem Radius *r* 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 Normalen Vektors an dem entsprechenden Punkt. Danach werden alle Punkte des Kreisbogens verschoben, sodass sich der Mittelpunkt M(50 | ym) ergibt.

Aus Abbildung 31 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 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.

Bei der Generierung der Profilgeometrien für die Propeller SRII und SRIII muss darauf geachtet werden, in welchen Bereichen entlang des dimensionslosen Radius welche Profilserie genutzt wurde. Aus Abbildung X geht hervor, dass die NACA 65/CA Profilserie 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 [NASA] 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 X und X dargestellten Design Parameter berücksichtigt. Die Transitionszone wird durch die Interpolation 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 Experiment File angegeben werden, wie viele Profile nach der Interpolation in gleichmäßigen radialen Abständen vorliegen sollen. Als weitere Information muss angegeben werden, über wie viele Schaufeln der entsprechende Propeller verfügt. In [NASA] ist angegeben, 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 anschließend unter der Wahl einer entsprechenden Strategie zum Fädeln der Profile eine Mantelfläche der Schaufel mit der implementierten Bibliothek generiert werden. Die Profile des SRII-Propellers werden über 50% der Sehnenlänge gefädelt. 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 X zeigt die generierten Mantelflächen der Schaufeln des SRII- und SRIII-Propellers.

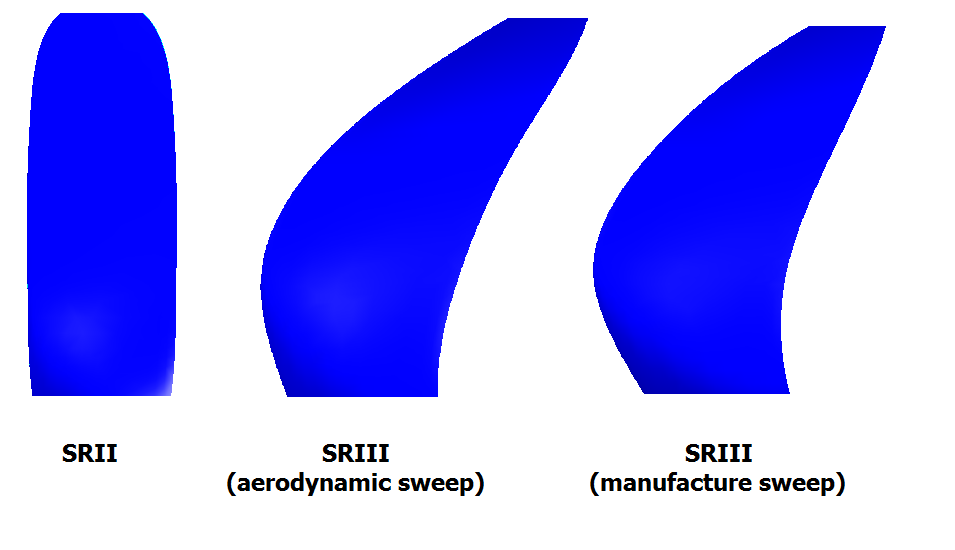


Abbildung X

Generierte Mantelflächen der Schaufeln der Propeller SRII und SRIII (aerodynamischer und gefertigter Pfeilwinkel)

#### 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.

Vergleich Form,

Vergleich Stacked View

Test Matrix

# 7 Zusammenfassung und Ausblick