Die approbierte Originalversion dieser Diplom-/Masterarbeit ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Erweiterung einer Software zur Berechnung von Schraubenmaschinen

ausgeführt zum Zwecke der Erlangung des akademischen Grades Master of Science unter der Leitung von (Betreuer)

a.O. Univ.-Prof. Dipl.-Ing. Dr. techn. **Manfred Grafinger**
Institutsnummer E 307
Institut für Konstruktionswissenschaften und Technische Logistik

eingereicht an der Technischen Universität Wien
Fakultät für Maschinenwesen und Betriebswissenschaften

von (Verfasser)

BSc **Rainer Riegler**
e1127060 (066 445)
Roseldorf 3, 2002 Großmugl

Wien, im Dezember 2016

Inhaltsverzeichnis

1. Eidesstattliche Erklärung **1**

2. Kurzfassung **1**

3. Abstract **1**

4. Schraubenkompressoren **2**

5. Geometrieerzeugung **4**

5.1. Bezierkurven 4
5.2. optimale Approximation 6
5.2.1. einfache Aproximation 6
5.2.2. Optimierung mithilfe Kurvenfunktion 6
5.3. B-Splines 12
5.4. Bezierkurven als B-Splines 14
5.4.1. Nachweis dass Bezierkurven als spezielle B-Splines dargestellt werden können 14
5.5. Ableitungen von B-Splines und Bezierkurven 16
5.6. Algorithmus zur Berechnung von Helixkurven 16
5.7. Lösen von linearen Gleichungssystemen mit Matrix in tridiagonalform - Thomas Algorithmus 19
5.8. Übergang zu Helixkurven 20
5.9. B-Splineflächen 21
5.10. Implementierung in stp-files 23
5.10.1. Header und allgemeine Einstellungen 23
5.10.2. Grund- und Deckfläche 24
5.10.3. Mantelfläche 25
5.10.4. Objektdefinition 26
5.11. Erzeugung von STL-files 27
5.11.1. Aufbau 27
5.11.2. Triangulation der Oberfläche 28
5.11.3. Triangulation der Grund und Deckflächen 30

6. Flächenverlauf **32**

6.1. Berührpunktermittlung 32
6.2. Flächensuche 37

6.3. Volumenverlauf 41
6.4. Spaltverläufe 42
6.4.1. Eingriffspalt 43
6.4.2. Kopfspalt 45
6.4.3. Stirnspalt 47
6.5. Kräfteberechnung 48

A. Step-file allgemeine Einstellungen **49**

B. Implementierung des Glättungsalgorithmus zur Berührpunktermittlung **51**

Abbildungsverzeichnis

1. Darstellung eines Blaslochs [5] 2
2. Beispiel eines Schraubenkompressors 3
3. Bernsteinpolynome 3.ten Grades 5
4. Illustration der einfachen Bezierkurvenoptimierung 7
5. Ergebnis der einfachen Interpolation 7
6. Illustration der Bezierkurvenoptimierung mithilfe der Kurvenfunktion . . 8
7. Illustration der Radiusdifferenz und der benutzten Fehlergröße 9
8. Illustration der Optimierung mithilfe der Kurvenfunktion 10
9. Illustration der Fehlerfläche mit dem Kreisbogenwinkel und Winkelfaktor 10
10. Illustration der B-Spline Basisfunktion in Bezierdarstellung 15
11. Illustration der B-Spline-Surface Erstellung 22
12. Illustration der Triangulation 29
13. Illustration der Triangulation in die Grundebene projiziert 29
14. Illustration der Triangulation 31
15. Illustration der Berührpunkte 32
16. Illustration des Glättungsalgorithmusses 34
17. diskretisierte parallele Geraden 34
18. Abstandsvariation paralleler Geraden 35
19. Illustration der Flächensuche 38
20. Illustration der benutzten Winkel 39
21. Illustration des resultierenden Flächenverlaufs 40
22. Illustration des resultierenden Volumenverlaufs 41
23. Illustration der Spaltarten 42
24. Illustration des Eingriffspaltverlaufs 43
25. Illustration der räumlichen Eingriffslinie 44
26. Illustration des Kopfspaltverlaufs 46
27. Illustration des Stirnspaltverlaufs 47

1. Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Roseldorf, 31. Dezember 2016

2. Kurzfassung

Gegenstand dieser Arbeit ist die Erweiterung einer bestehenden Software zur Berechnung von Schraubenmaschinen. Die zusätzlichen Module beeinhalten die Erzeugung von 3D Modellen der Schraubenmaschinen als stl und stp Dateien und die Erzeugung von Flächenverläufen von Geometrien mit mehr als 3 Berührpunkten. Zur Geometrieerzeugung sind verschiedene Algorithmen zur Approximation von Kreisbögen bzw. Helixkurven gezeigt.

3. Abstract

Subject of this work is the extensions of exisiting software to calculate screw-type compressors. The additional modules are to create 3d models of the runners as stl and stp files. The creation of geometric area functions of geometries with more than 3 touching points is added as well. The geometry generation is explained with different algorithms to approximate arcs and helices.

4. Schraubenkompressoren

Schraubenkompressoren zählen zu den Verdrängungsmaschinen und sind eine Spezialform von Zahnradpumpen. Die Zahnräder sind im wesentlichen weitaus länger und haben gleichzeitig sehr ausgeprägte Schrägungswinkel. Mehr als eine Umdrehung Umschlingungswinkel sind dabei durchwegs geläufig.
Es entsteht durch die Berührung des Haupt- und Nebenläufers eine Berührlinie, die von der Saugseite des Gehäuses bis zur Druckseite durchgängig ist. Durch diese sind die beiden Bereiche, unterschiedlichen Drucks, stets voneinander getrennt. In der Praxis kommt es durch die nicht perfekte Geometrie zu sogenannten Blaslöchern (Verrundungen am Läuferkopf führen zu einer Eingriffslinie, die nicht bis zur Verschneidungskante der Läufer reicht, siehe Abb. 1). Solche Bereiche, in denen es zu ungewolltem Druckausgleich aneinandergrenzender Zahnlücken kommt, kommen sowohl druck- als auch saugseitig vor. Allerdings ist nur das druckseitige Blasloch von Bedeutung, da dort höhere Druckunterschiede zwischen den Kammern und dementsprechend hohe Leckspaltströme vorkommen, die den Wirkungsgrad der Maschine senken. Wie groß dieser ist, lässt sich geometrisch nur abschätzen. Für eine genauere Aussage müsste eine Strömungsanalyse durchgeführt werden.

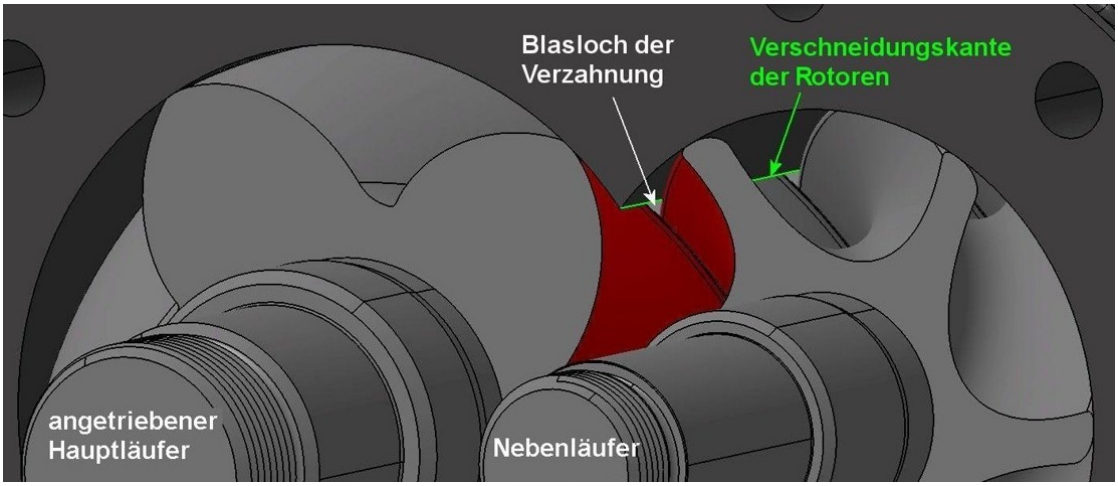


Abbildung 1: Darstellung eines Blaslochs [5]

Durch Drehung der Läufer, wird die in Abbildung 2 rot dargestellte Zahnlücke vergößert und es wird ein Medium angesaugt. Gleichzeitig wird auf der abgewandten Seite eine andere Zahnlücke verkleinert und Medium ausgestoßen bzw. verdichtet.

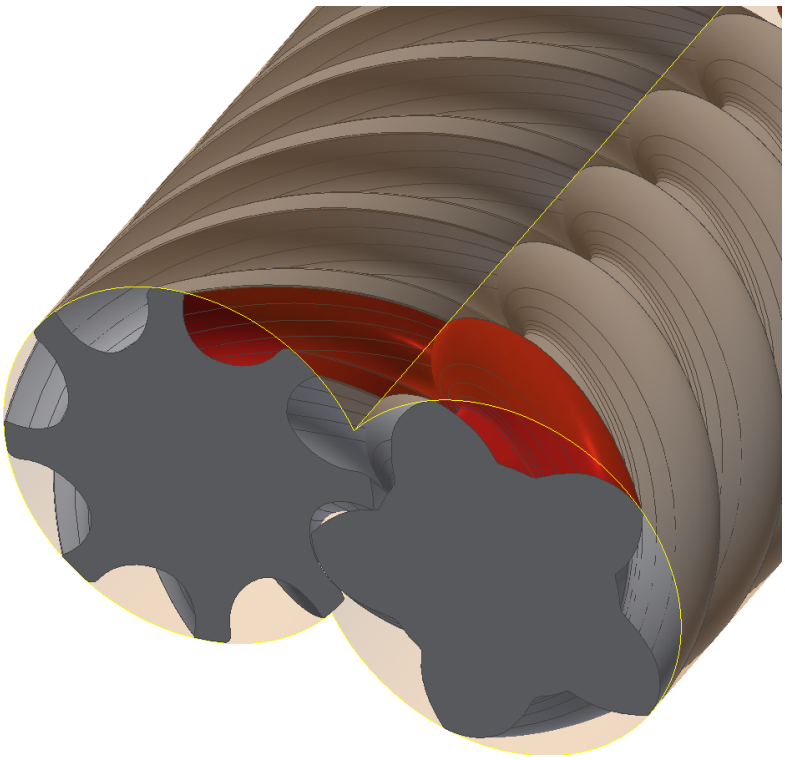


Abbildung 2: Beispiel eines Schraubenkompressors

5. Geometrieerzeugung

Die Geometrie wird im folgenden durch Bezierkurven, B-Splines und B-Splineflächen approximiert. Dabei wird zuerst die Möglichkeit der abschnittsweisen Ermittlung gezeigt, um die Lösung eines Gleichungssystems zu vermeiden und schon im Vorhinein sagen zu können, wie gut die Approximation ist, bzw. welche Winkelschritte nötig sind um im Rahmen der Fehlertoleranz zu bleiben.
NURBS könnten die zu erzeugende Geometrie exakt darstellen, allerdings ist die Kompatibilät mit kommerzieller CAD Software nicht sichergestellt.

5.1. Bezierkurven

Bézierkurven sind parametrische Kurven, die durch eine bestimmte Anzahl an Kontrollpunkten definiert werden. Dabei hat die Kurve den Grad n und wird durch n+1 Kontrollpunkte gesteuert. Die Kurve wird durch die Gleichung 1 definiert, wobei es sich bei P_i um die Kontrollpunkte und bei $b_{i,n}(t)$ um die Bernsteinpolynome n-ten Grades laut Gleichung 2 handelt.

$$\vec{C}\,(t) = \sum_{i=0}^{n} B_{i,n}\,(t)\,\vec{P}_i \;\; 0 \leq t \leq 1 \tag{1}$$

$$B_{i,n}\,(t) = \binom{n}{k} t^i (1-t)^{n-1} \;\; i = 0, \ldots, n \tag{2}$$

Die Summe der Bernsteinpolynome ist stets 1. Speziell für Bezierkurven 3.ten Grades ergeben sich die in Abbildung 3 ersichtlichen Funktionen.
Die Kontrollpunkte bilden einen Polygonzug (welcher als Kontrollpolygon bezeichnet wird). Die Kurve muss durch den Anfangs- und Endpunkt gehen und in diesen tangential an das Kontrollpolygon anliegen. Die Bezierkurve approximiert die anderen Kontrollpunkte und diese haben globalen Einfluss auf die Bezierkurve. Der Einfluss bleibt weitestgehend durch die Form der Basisfunktionen lokal begrenzt.

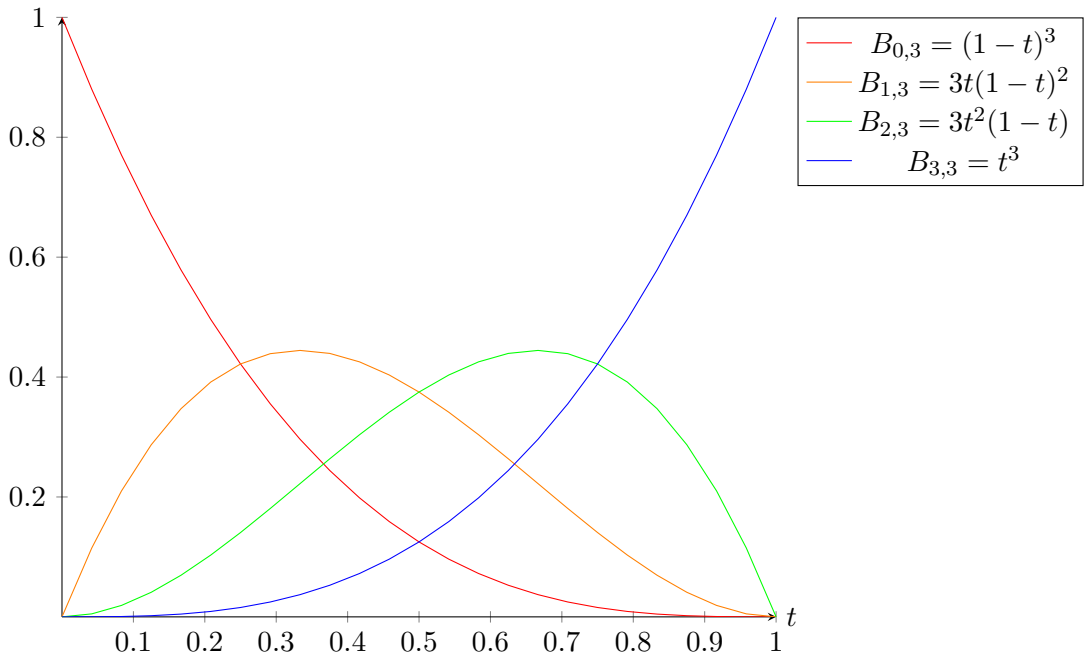


Abbildung 3: Bernsteinpolynome 3.ten Grades

5.2. optimale Approximation

Für diese Arbeit werden Bezierkurven zur Approximation von Kreisbögen bzw. von Helixkurven verwendet. Die Optimierung lässt sich auf ein Problem mit einem Freiheitsgrad vereinfachen, indem ausgenutzt wird, dass die Anfangs- und Endtangente des Kontrollpolygons tangential zu dem Kreisbogen sein soll. Außerdem ist der Kreisbogen symmetrisch um seine Winkelsymmetrale und dementsprechend muss das Kontrollpolygon ebenfalls symmetrisch um die Winkelsymmetrale sein. Das heißt, dass nur noch ein Freiheitsgrad zur Optimierung übrig bleibt.
Für diese Arbeit wird das Verhältnis zwischen Winkel α und β (Abbildung 4) als zu optimierender Freiheitsgrad gewählt. Für die folgenden Betrachtungen werden Bezierkurven dritten Grades und dementsprechend welche mit 4 Kontrollpunkten betrachtet

5.2.1. einfache Aproximation

Im ersten Schritt soll eine möglichst einfache Approximation gefunden werden. Dazu wird eine zusätzliche Bedingung eingeführt, die die Bezierkurve eindeutig definiert und somit die Optimierung hinfällig macht. Diese Bedingung soll den Kreisbogen relativ gut annähern und deshalb wird gefordert, dass die Approximation den Kreisbogen bei $\alpha/3$ und $2\alpha/3$ schneiden soll. Das führt dazu, dass die Bezierkurve bis zum ersten Schnittpunkt einen zu großen Radius um den Ursprung aufweißt und danach einen zu kleinen. Als Fehler wird die Radiusdifferenz bei $\alpha/2$ herangezogen, als e bezeichnet und auf den Radius des Kreisbogens bezogen und lt. Gleichung 4 definiert.

Wie in Abbildung 5 ersichtlich, ergibt sich ein maximal möglicher Kreisbogenwinkel, um den erlaubten Fehler nicht zu überschreiten.

$$\frac{e}{R} = c\alpha^b \tag{3}$$

$$\alpha = \left(\frac{e}{cR}\right)^{\frac{1}{b}} \tag{4}$$

Mit Gleichung 4 lässt sich der maximal zulässige Kreisbogenwinkel bei gegebenem Kreisbogen und erlaubtem absolutem Fehler errechnen.

5.2.2. Optimierung mithilfe Kurvenfunktion

Für eine betrachtete Bezierkurve 3ten Grades lässt sich die Ortskurve in karthesischen Koordinaten laut Gleichung 5 darstellen. Dazu wird ein Kreisbogen mit dem Winkel α

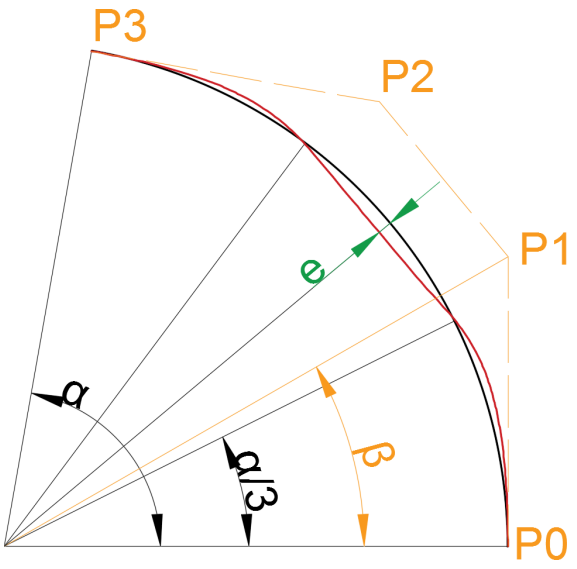


Abbildung 4: Illustration der einfachen Bezierkurvenoptimierung (stark übertrieben)

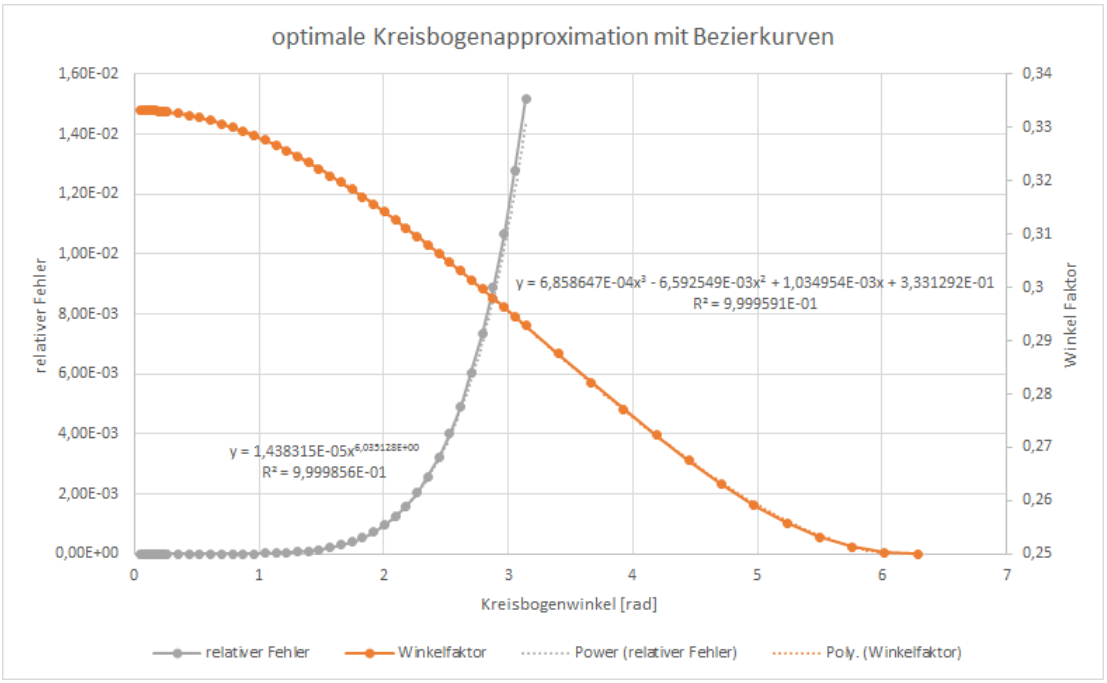


Abbildung 5: Ergebnis der einfachen Interpolation

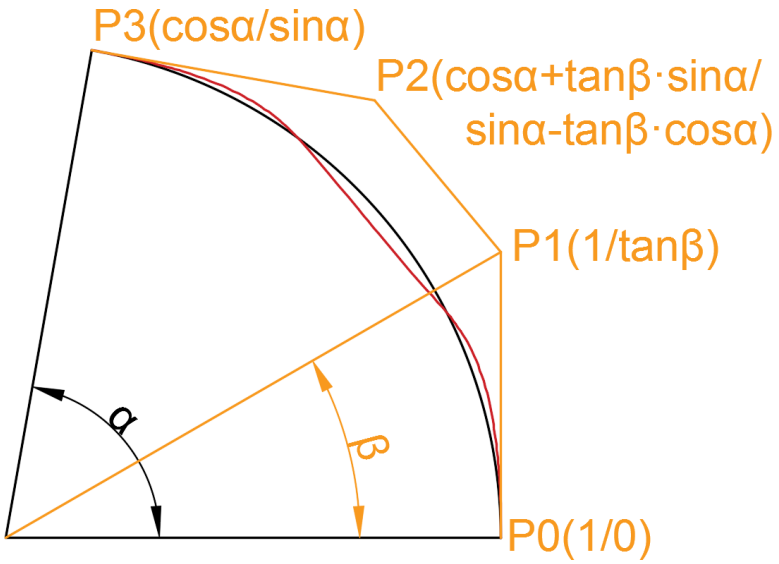


Abbildung 6: Illustration der Bezierkurvenoptimierung mithilfe der Kurvenfunktion

und einem Radius von 1 betrachtet. In Abbildung 6 sind die Anfangs- und Endpunkte P_0 & P_3 mit den inneren Kontrollpunkten P_1 & P_2 auf der Anfangs- und Endtangente dargestellt. Für die Optimierung wird der Radiusunterschied laut Gleichung 6 zu einem Kreisbogen betrachtet.

$$\vec{C}(t) = \left(1-t^3\right)\begin{pmatrix}1\\0\end{pmatrix} + 3\cdot t\,(1-t)^2\cdot\begin{pmatrix}1\\tan\,(\alpha b)\end{pmatrix} + \\ + \,3\cdot t^2\,(1-t)\begin{pmatrix}cos\alpha + tan\,(\alpha b)\cdot sin\alpha\\ sin\alpha - tan\,(\alpha b)\cdot cos\alpha\end{pmatrix} + t^3\cdot\begin{pmatrix}cos\alpha\\sin\alpha\end{pmatrix} \tag{5}$$

$$R(t) = \left|\vec{C}(t)\right| - 1 \tag{6}$$

Die Gleichung 6 hat für einen bestimmten Kreisbogen den Faktor $a = \beta/\alpha$ als einzigen freien Parameter, den es zu optimieren gilt. Die Optimierung soll dafür sorgen, dass die Fehleramplitude, wie in Abbildung 7 dargestellt, minimal wird.

Analytisch müssten dafür die Minima und Maxima im Parameterbereich durch partielles Ableiten nach dem Parameter t ermittelt und nach dem Nullsetzen rückeingesetzt werden. Allerdings ergibt sich durch das Berechnen der Radiusdifferenz lt. Gleichung 6 die Wurzel aus einem Polynom 6ter Ordnung. Diese Gleichung lässt sich zwar durch

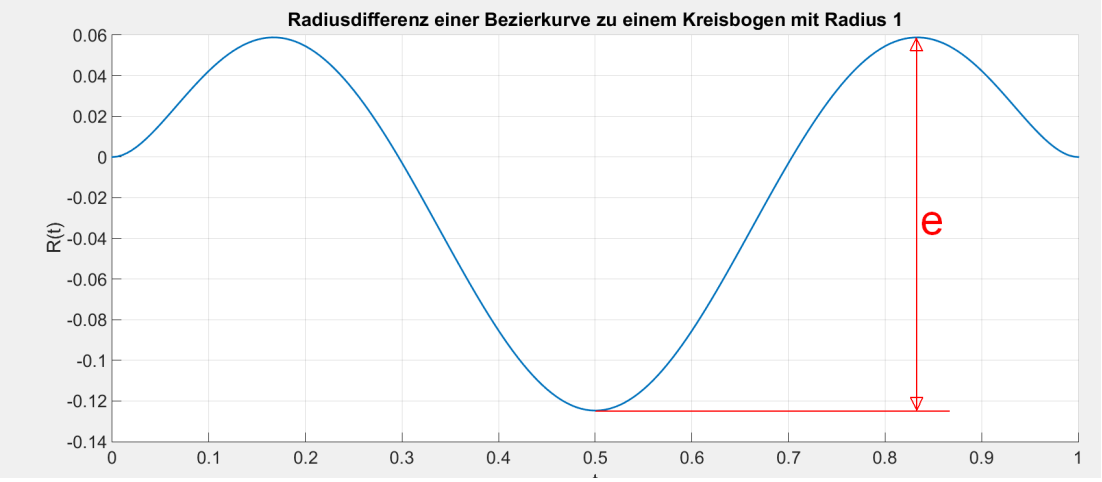


Abbildung 7: Illustration der Radiusdifferenz und der benutzten Fehlergröße

Umformen auf den quadratischen Radius etwas vereinfachen aber analytisch lässt sich kein brauchbares Ergebnis erzielen.
Deshalb wurde die weitere Analyse numerisch durchgeführt. Damit lassen sich relativ einfach brauchbare Parameter finden. Dazu wird die Gleichung 6 betrachtet und für einen gewissen Parameterbereich die Funktion berechnet. Die Differenz aus dem Maximum und dem Minimum der Funktion ergibt lt. Abbildung 7 den Fehler.

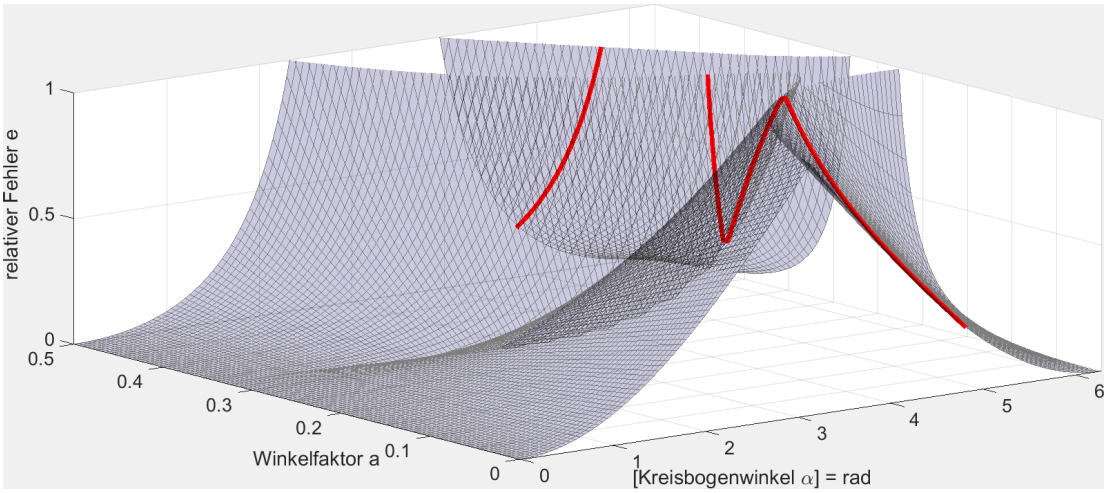


Abbildung 8: Illustration der Optimierung mithilfe der Kurvenfunktion

In Diagramm 8 ist der Fehler auf der z-Achse aufgetragen. Für einen bestimmten Winkel des Kreisbogens lässt sich ein Minimum finden, welches durch partielles Ableiten nach dem Parameter a ermittelt wird. Dadurch erhält man das Diagramm 9.

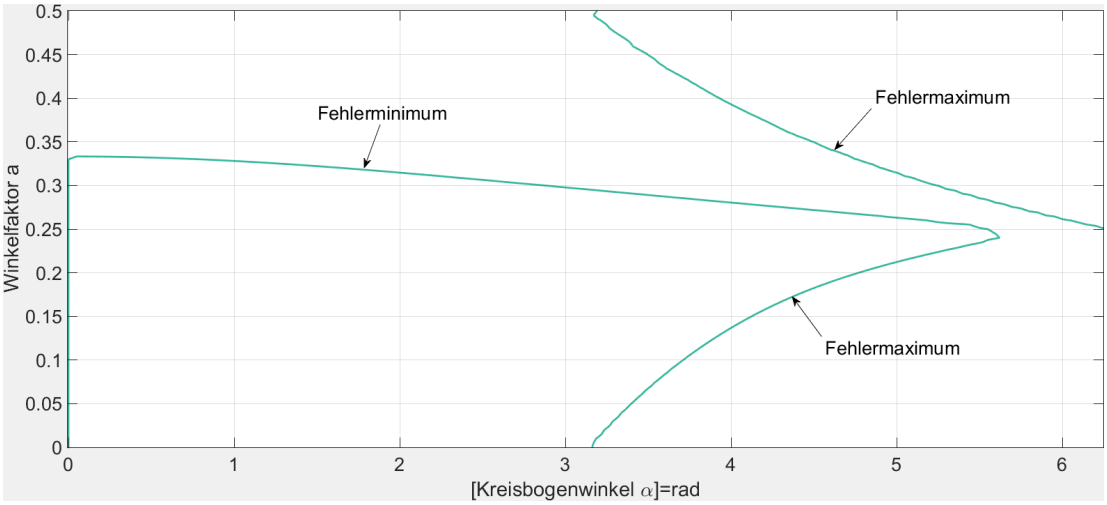


Abbildung 9: Illustration der Fehlerfläche mit dem Kreisbogenwinkel und Winkelfaktor

In Diagramm 8 und Diagramm 9 erkennt man außer der Kurve für den minimalen Fehler auch, dass es ab einem gewissen Winkel besser ist $a = 0$ zu wählen (und damit eine Gerade durch Anfangs- und Endpunkt des Kreisbogens zu legen), als eine Approximation mit der richtigen Anfangs- und Endtangente durchzuführen. Diese Eigenschaft ist durch die rote Kurve ($\alpha = 4.75$) dargestellt. Bei dieser sieht man, dass das lokale Minimum bei

$a \approx 0.3$ einen höheren Fehler aufweist, als die Randwerte dieser Kurve.
Diagramm 9 stellt die lokalen Extremwerte bei fixen Kreisbogenwinkel dar. Dadurch wird gezeigt, dass die qualitativ gleichen Ergebnisse wie in Diagramm 5, der einfachen Interpolation, erhalten werden
Außerdem zeigt Diagramm 9 die von dem lokalen Minimum abweichenden Zweige, welche die Parameterwerte für maximale Abweichung darstellen.

5.3. B-Splines

B(asis)-Splines sind ähnlich wie Bezierkurven, Kurven, die durch ein Kontrollpunktset $\vec{P}_i$ (mit n+1 Kontrollpunkten) und den jeweils zugehörigen Basisfunktionen definiert sind. B-Splines sind lt. Gleichung 7 als die Summe des Produkts der Kontrollpunkte und der jeweiligen Basisfunktion definiert. Die Basisfunktionen sind von den Knoten t im Knotenvektor und dem Grad der Kurve abhängig.
Im Unterschied zu Bezierkurven beeinflusst nicht jeder Kontrollpunkt die gesamte Kurve, sondern abhängig vom Grad der Kurve und dem gewählten Knotenvektor nur einen gewissen Bereich.
Die Basisfunktionen sind in Gleichung 8 und 9 definiert und hängen vom Knotenvektor und dem Grad der Kurve ab. Der Grad der Kurve wird mit p bezeichnet. Man erkennt, dass die Basisfunktionen jeweils eine Linearkombination von Basisfunktionen niedriger Ordnung sind. Somit ist auch klar, dass die Knotenpunkte eines B-Splines höheren Grades Einfluss auf einen größeren Bereich der Kurve haben, als die Knotenpunkte eines B-Splines niedrigeren Grades.
Der Kurvengrad p führt zu C^{p-1} Stetigkeit zwischen den Kurvensegmenten. Vielfachheit der Knoten bewirkt eine Reduktion der Stetigkeit zwischen den benachbarten Basisfunktionen in dem jeweiligen Knoten, und entsprechend hat es den selben Effekt auf die Kurve. Von Vielfachheit wird gesprochen, wenn aufeinanderfolgende Knoten im Knotenvektor den selben Wert aufweisen.

$$\vec{C}(t) = \sum_{i=0}^{n} N_{i,p}(t)\,\vec{P}_i \qquad 0 \le t \le 1 \tag{7}$$

$$N_{i,0}(t) = \begin{cases} 1 & \text{if } t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$N_{i,p+1}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,p}(t) + \frac{t_{i+n} - t}{t_{i+n} - t_{i+1}} N_{i+1,p}(t) \tag{9}$$

Zudem wird stets die Gleichung 10 erfüllt, wobei m+1 die Anzahl an Knoten im Knotenvektor lt. Gleichung 11, n+1 die Anzahl an Kontrollpunkten und p der Kurvengrad ist. Die Elemente des Knotenvektors sind stets monoton steigend und falls der Abstand zweier Elemente im Knotenvektor stets gleich ist, spricht man von kardinalen B-Splines. Diese werden üblicherweise zur Interpolation verwendet. Die Kurvenform lässt sich über

die Abstände der Knoten im Knotenvektor beeinflussen.
Eine Möglichkeit ist es die Wahl des Knotenvektors abhängig vom Kontrollpunkteset zu machen. Dazu kann das Kontrollpolygon betrachtet werden und der Wert des i-ten Knoten als Summe der Längen der ersten i Geradenstücke des Kontrollpolygons festgelegt werden. Solch eine Wahl führt üblicherweise zu "glatteren"Kurven, kann aber auch zu nicht erwarteten Kurvenformen führen.

$$m = p + n + 1 \tag{10}$$

$$\boldsymbol{t} = \underbrace{0, \ldots, 0}_{p+1}, \underbrace{a_1, \ldots, a_{n-p-1}}_{n-p-1}, \underbrace{1, \ldots, 1}_{p+1} \tag{11}$$

$$a_{i-1} \leq a_i \qquad 0 \leq a_i \leq a_m$$

Um die Kurve durch den Anfangs- und Endpunkt gehen zu lassen, müssen die ersten und letzten p+1 Elemente im Knotenvektor den selben Wert aufweisen. Außerdem wird üblicherweise, aber nicht notwendigerweise, der erste Knoten auf 0 und der letzte auf 1 gesetzt. Die Kurvenform wird nur durch die relative Größe der Knoten bestimmt.

5.4. Bezierkurven als B-Splines

B-Splines lassen sich durch eine bestimmte Wahl des Knotenvektors so zusammensetzen, sodass die Teilstücke der Kurve Bezierkurven sind. Dazu werden die Knoten so gewählt, dass sie jeweils eine Vielfacheit von p+1 aufweisen. Damit weißt die Kurve keine Stetigkeit mehr zwischen den Kurvensegmenten auf. Die Form des Knotenvektors ist in Gleichung 12 dargestellt. Die Stetigkeit muss deshalb durch die Positionierung der Kontrollpunkte geschehen, indem zumindest die End- und Anfangspunkte der Teilbezierkurven deckungsgleich sind. Weiters lässt sich eine tangenstetige Kurve leicht konstruieren, indem die End- und Anfangstangenten aufeinanderfolgender Teilkurven zusätzlich zueinander parallel gewählt werden.

$$\boldsymbol{t} = \underbrace{\underbrace{0,\ldots,0}_{p+1},\underbrace{1,\ldots,1}_{p+1},\ldots,\underbrace{\frac{n+1}{p+1},\ldots,\frac{n+1}{p+1}}_{p+1}}_{n+p+2} \tag{12}$$

Diese Vorgehensweise führt zu einer Kurve, die aus Bezierkurven zusammengesetzt ist und damit der Einfluss der jeweiligen inneren Kontrollpunkte nur auf die jeweilige Bezierkurve beschränkt bleibt. Allerdings wird auch die automatische mehrfache Stetigkeit von B-Splines an den Verbindungsstellen der Teilkurven aufgehoben, sodass u.U. die Kurve an den Verbindungsstellen sogar springen kann. Die Kurve ließe sich noch vereinfachen, indem die übereinanderliegenden Punkte entfernt würden, und durch eine entsprechende Wahl des Knotenvektors die Kurve so verändert würde, dass die Kurve weiterhin durch den Punkt geht. Darauf wird in dieser Arbeit verzichtet.

5.4.1. Nachweis dass Bezierkurven als spezielle B-Splines dargestellt werden können

B-Splines und Bezierkurven unterscheiden sich, wie in Gleichung 1 und 7 ersichtlich ist, nur durch die unterschiedlichen Basisfunktionen. Das heißt, die Basisfunktionen eines B-Splines müssen identisch zu den Bernsteinpolynomen gleicher Ordnung sein, um mit gleichen Kontrollpunkten eine identische Bezierkurve erzeugen zu können. Dazu werden Beispielhaft die Basisfunktionen einer B-Splinekurve 3ten Grades berechnet und mit den Bernsteinpolynomen verglichen. Es wird der Knotenvektor lt. Gleichung 13 aufgestellt und für die Berechnung benutzt.

$$\boldsymbol{t} = 0,0,0,0,1,1,1,1 \tag{13}$$

$$p = 3 \quad m = 8 \quad n = 3 \tag{14}$$

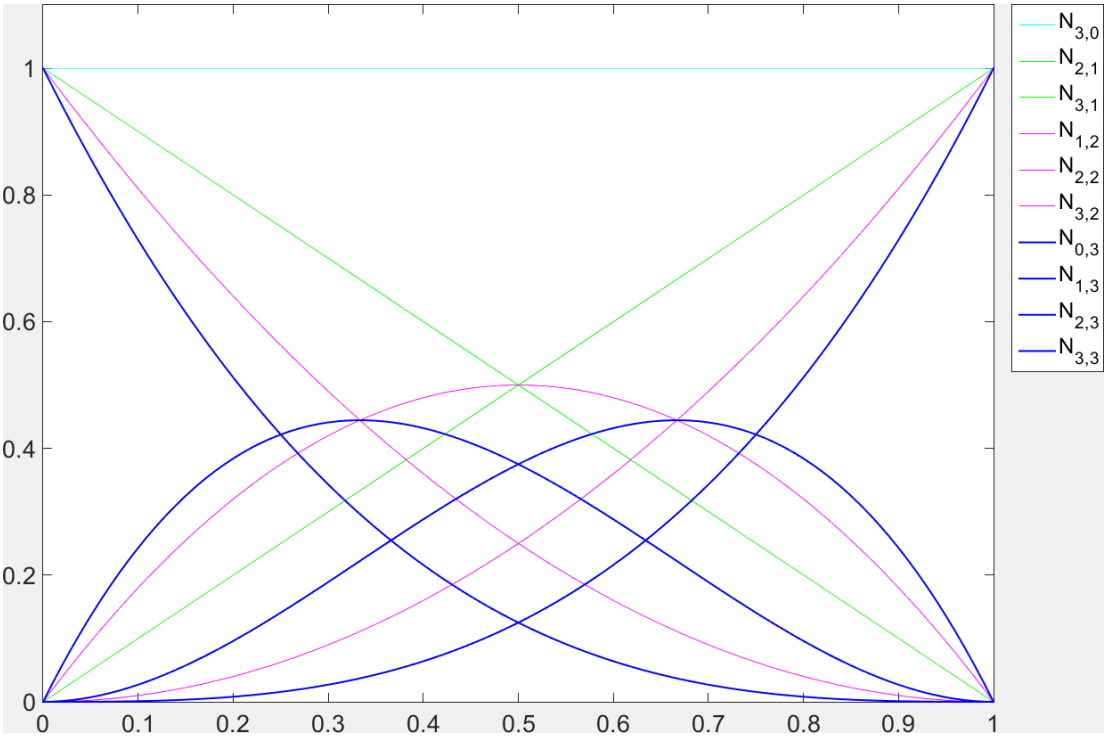


Abbildung 10: Illustration der B-Spline Basisfunktion in Bezierdarstellung

Die damit erzeugten Basisfunktionen lauten lt. Gleichung 8 wie folgt und sind in Abbildung 10, mit jeweils unterschiedlicher Farbe für unterschiedliche Ordnung der Basisfunktionen dargestellt, die nicht 0 sind:

$$
\begin{array}{llll}
N_{0,3} = (1-t)^3 & N_{0,2} = 0 & N_{0,1} = 0 & N_{0,0} = 0 \\
N_{1,3} = 3 \cdot t(1-t)^2 & N_{1,2} = (1-t)^2 & N_{1,1} = 0 & N_{1,0} = 0 \\
N_{2,3} = 3 \cdot t^2(1-t) & N_{2,2} = 2 \cdot t\,(1-t) & N_{2,1} = (1-t) & N_{2,0} = 0 \\
N_{3,3} = t^3 & N_{3,2} = t^2 & N_{3,1} = t & N_{3,0} = 1 \\
 & N_{4,2} = 0 & N_{4,1} = 0 & N_{4,0} = 0 \\
 & & N_{5,1} = 0 & N_{5,0} = 0 \\
 & & & N_{6,0} = 0
\end{array}
$$

Man erkennt sofort, dass die Basisfunktionen identisch, zu denen in Abbildung 3 sind und dass tatsächlich die Basisfunktion 0ter Ordnung konstant 1 bzw. 0 ist. Außerdem lässt

sich erkennen, dass die Basisfunktionen jeweils lineare Interpolationen niedrigerer Ordnung sind. Außerdem sind auch die Basisfunktionen niedriger Ordnung Bernsteinpolynome und dementsprechend sind B-Splines mit einem Knotenvektor der Form 12 stets aus Bezierkurvenstücken zusammengesetzt.

5.5. Ableitungen von B-Splines und Bezierkurven

Die Ableitung eines B-Splines der Ordnung p ergibt sich als B-Spline von Ordnung p-1 und berechnet sich lt. Gleichung 15

$$\frac{dN_{i,n}(t)}{dt} = (n-1)\left(\frac{N_{i,n-1}(t)}{t_{i+k-1}-t_i} - \frac{N_{i+1,n-1}(t)}{t_{i+n}-t_{i+1}}\right) \tag{15}$$

$$\frac{d\,\vec{C}(t)}{dt} = \sum_{i=0}^{n-1} \frac{(n-1)\cdot\left(\vec{P}_{i+1}-\vec{P}_i\right)}{t_{i+n}-t_{i+1}} N_{i,p-1}(t) \qquad 0 \le t \le 1 \tag{16}$$

Außerdem müssen der erste und letzte Knoten entfernt werden, um die Bedingung 10 zu erfüllen. Damit hat die Ableitung die Form eines B-Splines der Ordnung p-1 mit einem neuen Kontrollpolygon, das jeweils aus der Differenz 2er aufeinanderfolgender Kontrollpunkte berechnet wird. [1]

5.6. Algorithmus zur Berechnung von Helixkurven

B-Spline Kurven ohne stetigen Übergang zwischen den Abschnitten durch vielfache innere Knoten der Form der Gleichung 12 können in kommerzieller Software zu unvorhergesehenem Verhalten führen. Deshalb wird ein einfacher Algorithmus zur guten Approximation von Kreisbögen beliebiger Umschlingungswinkel vorgestellt. Dazu wird speziell ein B-Spline 3ter Ordnung betrachtet.

$$\boldsymbol{t} = \underbrace{0,\ldots,0}_{p+1}, 1, 2, \cdots, \frac{n+1}{p+1}, \underbrace{\frac{n+1}{p+1},\ldots,\frac{n+1}{p+1}} \tag{17}$$

Zur Festlegung der Anzahl an Kontrollpunkten wird Gleichung 4 herangezogen und pro nötigem Winkelschritt 4 Kontrollpunkte benutzt. Das führt auf das Gleichungssystem 18. Dieses hat Tridiagonalform und ist damit mit relativ niedrigem Rechenaufwand lösbar. Um die Kurve vollständig zu definieren wird bei t=0.3 ein zusätzlicher Stützpunkt eingeführt. Dies führt zum Verlust der Tridiagonalform und lässt sich beheben, indem näherungsweise der kleinste Parameter zu Null gesetzt wird. Der Knotenvektor 17 wird

benutzt. Außerdem führt die so getroffene Wahl zu einem Radiusfehler im Anfangsbereich der Kurve, der vergleichbar ist mit dem des Rests der Kurve. Als Alternative dazu, ist es möglich die Anfangstangente festzulegen. Mit dieser Herangehensweise hat man allerdings Probleme den vorgegebenen Radiusfehler einzuhalten.

$$K(t) \cdot \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_n \end{bmatrix} = \begin{bmatrix} A_0 \\ A_1 \\ \vdots \\ A_n \end{bmatrix} \tag{18}$$

$$K(t) = \begin{bmatrix}
N_{0,0} & N_{1,0} & N_{2,0} & N_{3,0} & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\
N_{0,0} & N_{1,0} & N_{2,0} & N_{3,0} & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\
0 & N_{1,1} & N_{2,1} & N_{3,1} & N_{3,0} & \cdots & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_{2,2} & N_{3,2} & N_{3,1} & \cdots & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & N_{3,3} & N_{3,2} & \cdots & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & \cdots & N_{3,2} & N_{3,1} & N_{3,0} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & N_{3,3} & N_{3,2} & N_{3,1} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & N_{3,3} & N_{3,2} & N_{5,0} & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & N_{3,3} & N_{4,2} & N_{5,1} & N_{6,0} \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & N_{3,3} & N_{4,2} & N_{5,1} & N_{6,0}
\end{bmatrix}$$

Die sich durch den Knotenvektor 17 ergebenden Basisfunktionen sind im folgenden dargestellt.

$$
\begin{aligned}
&N_{0,0} = -t^3 + 3t^2 - 3t + 1 = (1-t)^3 \\
&N_{1,0} = 5/4t^3 - 9/2t^2 + 3t && N_{1,1} = -1/4t^3 + 3/4t^2 - 3/4t + 1/4 \\
&N_{2,0} = -11/12t^3 + 3/2t^2 && N_{2,1} = 7/12t^3 - 5/4t^2 + 1/4t + 7/12 \\
&N_{2,2} = -1/6t^3 + 1/2t^2 - 1/2t + 1/6 \\
&N_{3,0} = 1/6t^3 && N_{3,1} = -1/2t^3 + 1/2t^2 + 1/2t + 1/6 \\
&N_{3,2} = 1/2t^3 - t^2 + 2/3 && N_{3,3} = -1/6t^3 + 1/2t^2 - 1/2t + 1/6 \\
&N_{4,0} = 1/6t^3 && N_{4,1} = -7/12t^3 + 1/2t^2 + 1/2t + 1/6 \\
&N_{4,2} = 11/12t^3 - 5/4t^2 - 1/4t + 7/12 \\
&N_{5,0} = 1/4t^3 && N_{5,1} = -7/4t^3 + 3/4t^2 + 3/4t + 1/4 \\
&N_{6,0} = t^3
\end{aligned}
$$

Die Matrix K wird explizit berechnet indem Zeilenweise $0, 0.3, 0, \cdots, 0, 0.7, 1$ für t eingesetzt wird. In der zweiten (und vorletzten) Zeile wird der kleinste Wert näherungsweise zu Null gesetzt (und die Summe der anderen 3 wird auf 1 normiert) um tridiagonalform zu erreichen. Zur effizienten Speicherung wird die tridiagonalisierte Matrix in eine nx3 Matrix umgeschrieben.

$$
K = \begin{bmatrix}
1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
0.34 & 0.54 & 0.11 & 0.005 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0.25 & 0.58 & 0.17 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0 & 0.17 & 0.67 & \cdots & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 0.67 & 0.17 & 0 & 0 \\
0 & 0 & 0 & 0 & \cdots & 0.17 & 0.67 & 0.25 & 0 \\
0 & 0 & 0 & 0 & \cdots & 0.005 & 0.11 & 0.54 & 0.34 \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1
\end{bmatrix}
\overset{Tri}{=}
\begin{bmatrix}
0 & 1 & 0 \\
0.34 & 0.54 & 0.11 \\
0.25 & 0.58 & 0.17 \\
0.17 & 0.67 & 0.17 \\
\vdots & \vdots & \vdots \\
0.17 & 0.67 & 0.25 \\
0.11 & 0.54 & 0.34 \\
0 & 1 & 0
\end{bmatrix}
\tag{19}
$$

Damit kann das zu lösende Gleichungssystem aufgestellt und mit einem geeigneten Algorithmus gelöst werden:

$$K \cdot \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_n \end{bmatrix} = \begin{bmatrix} \vec{A}\,(0) \\ \vec{A}\,(0.3) \\ \vec{A}\,(1) \\ \vec{A}\,(2) \\ \vdots \\ \vec{A}\,(n) \end{bmatrix} \qquad \vec{A}\,(t) = r \cdot \begin{pmatrix} cos\left(\frac{\phi}{n} \cdot t\right) \\ sin\left(\frac{\phi}{n} \cdot t\right) \end{pmatrix} \tag{20}$$

5.7. Lösen von linearen Gleichungssystemen mit Matrix in tridiagonalform - Thomas Algorithmus

Zur Lösung von linearen Gleichungssystemen mit tridiagonaler Koeffizientenmatrix, wie es auch in Gleichung 20 vorkommt, eignet sich der Thomas Algorithmus. Dieser spezialisiert im wesentlichen den Gauß Algorithmus auf Gleichungssystemen mit tridiagonaler Koeffizientenmatrix und löst solche Gleichungssysteme in 2 Durchgängen. Das Gleichungssystem wird für jede Komponente in $\vec{A}$ einzeln gelöst. Die Komponenten in $\vec{A}$ werden im weiteren mit d_i bezeichnet, um für den Thomas Algorithmus übliche Bezeichnungen zu verwenden. Dazu wird die Matrix K, wie schon in Gleichung 5.6 durchgeführt wurde, in eine nx3 Matrix umgeschrieben.

$$K = \begin{bmatrix} 0 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \\ \vdots & \vdots & \vdots \\ a_{n-2} & b_{n-2} & c_{n-2} \\ a_{n-1} & b_{n-1} & c_{n-1} \\ a_n & b_n & 0 \end{bmatrix} \tag{21}$$

Im ersten Durchgang werden die Koeffizienten c_i und d_i rekursiv nach folgendem

Schema modifiziert:

$$c'_i = \begin{cases} \frac{c_1}{b_1} & \text{if } i = 1 \\ \frac{c_i}{b_i - c'_{i-1} a_i} & \text{if } 1 < i \leq n-1 \end{cases} \tag{22}$$

$$d'_i = \begin{cases} \frac{d_1}{b_1} & \text{if } i = 1 \\ \frac{d_i - d'_{i-1} a_i}{b_i - c'_{i-1} a_i} & \text{if } 1 < i \leq n \end{cases} \tag{23}$$

Mit dem ersten Durchgang wurden die Koeffizienten a_i eliminiert und es muss zur vollständigen Lösung des Gleichungssystems nur noch rückwärts eingesetzt werden.

$$P_i = \begin{cases} d'_i & \text{if } i = n \\ d'_i - c'_i P_{i+1} & \text{if } n > i \geq 1 \end{cases} \tag{24}$$

5.8. Übergang zu Helixkurven

Bei Helixkurven handelt es sich um Kreisbögen, die um eine zusätzliche Koordinate erweitert wurden. Dabei steigt die zusätzliche Koordinate linear mit dem Kreisbogenwinkel. Somit haben Helixkurven die Form 25

$$\vec{H}\,(t) = \begin{pmatrix} cos\left(\frac{\phi}{n} \cdot t\right) \\ sin\left(\frac{\phi}{n} \cdot t\right) \\ h \cdot t \end{pmatrix} \tag{25}$$

Die Approximation mit einem B-Spline funktioniert genauso wie die eines Kreisbogens, mit dem Unterschied, dass die dritte Koordinate mitapproximiert wird. Das Ergebnis ist dabei immer eine lineare Verteilung der Kontrollpunkte in der neuen Koordinate.

5.9. B-Splineflächen

B-Splineflächen setzen sich, wie in Definition 26 erkennbar, aus dem Produkt 2er B-Spline Kurven zusammen. Das führt dazu, dass bei Veränderung von nur einem Parameter r oder t die Figur eine B-Spline Kurve wird.

$$\vec{S}\,(t) = \sum_{i=0}^{m} \sum_{j=0}^{n} N_{i,p}\,(r)\, N_{j,q}\,(t)\, \vec{P}_{i,j} \qquad 0 \leq r, t < 1 \tag{26}$$

Diese Eigenschaft lässt sich zur einfachen Ermittlung der Kontrollpunkte von verschraubten Flächen ausnutzen. Die Begrenzungen von einem Patch sind jedenfalls B-Spline Kurven, die in dieser Arbeit Kreisbögen bzw. Helixkurven approximieren. Damit lassen diese sich leicht nach den zuvor vorgestellten Methoden berechnen. Da die Anfangs- und Endkurve der verschraubten Fläche bis auf eine Verdrehung Deckungsgleich sind, und alle dazu parallelen Kurven dieselbe Form aufweisen sollen, müssen die inneren Kontrollpunkte dieselbe relative Winkelpositionierung aufweisen,wie die Grund und Deckfläche. Außerdem gilt das gleiche Prinzip mit den begrenzenden Helixkurven. Damit ergeben sich die inneren Kontrollpunkte durch die Erzeugung durch die inneren Kontrollpunkte der Helixkurven. Damit erhält man im wesentlichen um die Drehachse skalierte B-Splinekurven.
Diese Vorgehensweise lässt sich sehr gut bei Verwendung von B-Splines als Bezierkurven implementieren. Allerdings kann es dabei durch die Multiplizität der Knoten und daraus folgendem Verlust der Stetigkeit zu Problemen bei der Interpretation durch kommerzielle Software kommen. Das Problem lässt sich beheben, indem der in 5.6 vorgestellte Algorithmus modifiziert verwendet wird.
Dazu werden wie gehabt die Kreisbögen und Helixkurven der Grund und Deckflächen, sowie der verschraubten Verbindung durch den Algorithmus erzeugt. Weiters wird wie vorhin ausgenutzt, dass die B-Spline-Flächen Definition nur die Basisfunktionen multipliziert und damit ähnliche Kurven erzeugt werden. Die inneren Kontrollpunkte der Fläche werden durch die inneren Kontrollpunkte der begrenzenden Helixkurven mit dem Algorithmus 5.6 erzeugt. Diese B-Splines erzeugen nach Multiplikation mit der anderen Basisfunktion eine B-Splinekurve, die in der verschraubten Fläche liegt, und somit eine Approximation darstellt, die den Fehler in der verschraubten Fläche klein hält.
Der Fehler in der Approximation bestimmt sich nur durch das Produkt des Fehlers der B-Splinekurven in beide Parameterrichtungen. Nachdem der Kreisbogenwinkel der Grund- und Deckkurven im Vergleich zum Umschlingungswinkel klein ist, wird dieser in dieser Arbeit vernachlässigt und nur der Fehler für den Umschlingungswinkel betrachtet. Der Umschlingungswinkel wird so gewählt, dass der maximale Fehler weniger als die Wurzel

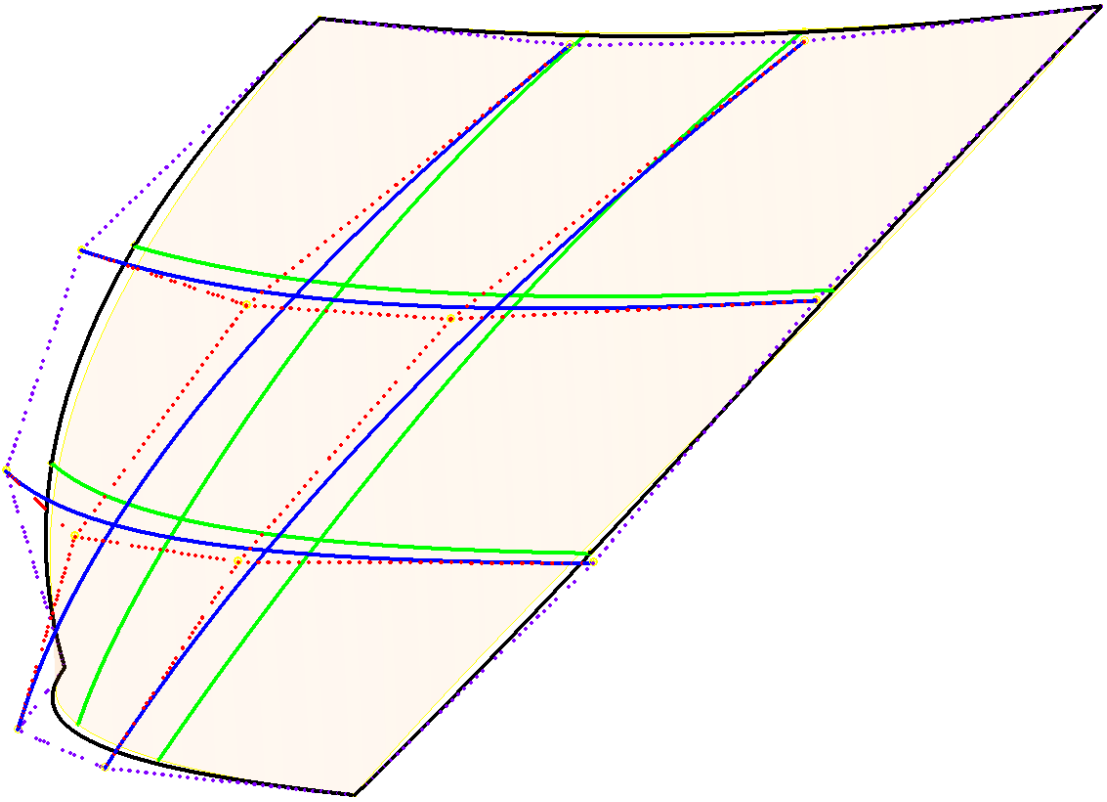


Abbildung 11: Illustration der B-Spline-Surface Erstellung

des zulässigen Fehlers beträgt, und der Gesamtfehler dadurch kleiner als der zulässige Fehler beträgt.

In Abbildung 11 ist die Vorgenhensweise dargestellt. Dazu ist ein Stück einer Zylinderfläche und deren Flächenapproximation gezeigt. In schwarz durchgezogenen Linien sind die B-Spline Kurven der Berandung dargestellt. Die Kontrollpolygone von diesen sind in punktierten magentafarbenen Linien gezeichnet. Von diesen ausgehend sind in blauen durchgezogenen Linien die B-Spline Kurven mit deren Kontrollpolygonen in roten punktierten Linien dargestellt. Die damit in die Fläche projizierten Kurven sind grün.

5.10. Implementierung in stp-files

STEP ist ein neutrales Datenformat nach ISO 10303, das zum Austausch von CAD Daten dient. Diese besitzen ASCII Struktur und sind damit leicht lesbar. Es wird jeweils eine Instanz pro Zeile geschrieben. Jede Zeile ist nummeriert und es wird auf die Instanzen über die Zeilennummer verwiesen. Eine Instanz endet stets mit ;
Für diese Arbeit wird das Step-File im wesentlichen in 4 Teile unterteilt, die im folgenden beschrieben werden. Kommentare in Step-Files werden durch /* */ gekennzeichnet. Strings werden durch ' ' als solche deklariert.

5.10.1. Header und allgemeine Einstellungen

Im Header werden allgemeine Informationen gespeichert, wie der Dateiname, Versionsnummern, usw. Diese Informationen werden nicht auf Richtigkeit überprüft.

```
ISO-10303-21;
HEADER;

FILE_DESCRIPTION(
/* description */ (''),
/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'E:\Dateiname.stp',
/* time_stamp */ '2016-04-21T10:03:19+02:00',
/* author */ ('rarie'),
/* organization */ (''),
/* preprocessor_version */ 'SV',
/* originating_system */ 'SV',
/* authorisation */ '');

FILE_SCHEMA (('AUTOMOTIVE_DESIGN { 1 0 10303 214 3 1 1 }'));
ENDSEC;
```

Der nächste Abschnitt in Step-Files ist der wichtige Datenbereich. Darin werden allgemeine Einstellungen und Informationen zu den benutzen Einheiten, welche Typen von Objekten benutzt werden und welche Toleranzen benutzt werden angegeben. Der

genaue Aufbau davon ist im Anhang zu finden. Der Datenabschnitt wird durch folgende Zeile aufgerufen.

```
DATA;
```

5.10.2. Grund- und Deckfläche

Die Grund und Deckflächen bestehen aus aneinandergereihten Kreisbögenstücken. Diese Kreisbögen werden einzeln durch folgende Zeilen erzeugt:

```
#50=ORIENTED_EDGE('',*,*,#51,.T.);
#51=EDGE_CURVE('',#52,#54,#56,.T.);
#52=VERTEX_POINT('',#53);
#53=CARTESIAN_POINT('',(36.7,-24.2,0.));
#54=VERTEX_POINT('',#55);
#55=CARTESIAN_POINT('',(38.7,-23.3,0.));
#56=CIRCLE('',#57,2.1);
#57=AXIS2_PLACEMENT_3D('',#58,#59,#60);
#58=CARTESIAN_POINT('Origin',(38.4,-25.4,0.0));
#59=DIRECTION('center_axis',(0.,0.,-1.));
#60=DIRECTION('ref_axis',(1.,0.,0.));
```

Die Zeile #50 erzeugt eine Kante, die durch Kurve #51 beschrieben wird, welche die Punkte #52 und #54 als Anfangs- und Endpunkt benutzt und die Kurvenform #56 besitzt. Die Zeile #56 beinhaltet den Radius des Kreisbogens und verweist auf die Achse #57, die den Mittelpunkt des Kreisbogens beschreibt. Die Achse #57 geht durch den Punkt #58 und zeigt in Richtung #59, wobei eine Refernzrichtung #60 angegeben werden muss, die in dieser Arbeit stets in Richtung der x-Achse zeigt.
Die Kreisbögen werden anschließend mit diesem Code zu einer Fläche zusammengesetzt.

```
#42=ADVANCED_FACE('',(#48),#43,.F.);
#43=PLANE('',#44);
#44=AXIS2_PLACEMENT_3D('',#45,#46,#47);
#45=CARTESIAN_POINT('Origin',(0.,0.,0.));
#46=DIRECTION('center_axis',(0.,0.,1.));
#47=DIRECTION('ref_axis',(1.,0.,0.));
#48=FACE_OUTER_BOUND('',#49,.F.);
#49=EDGE_LOOP('',(#50,#61,...,#919));
```

Die Flächenerzeugung geschieht mit #42, die auf die begrenzende Kontur #48 und die Fläche #43 verweist. In diesem Fall ist die Fläche eine Ebene, die eindeutig durch die Achse #44 beschrieben wird. Die Achse wird wie schon zuvor definiert. Die Begrenzungskurve #48 setzt sich aus mehreren Kurvenstücken zusammen, und ruft deshalb die Funktion für geschlossene Kurven #49 auf. In diese werden die Zeilennummern für die orientierten Kurven #50,$\cdots$ eingetragen. Die Richtung der Kurvenstücke muss so sein, dass jeweils die Koordinaten des Endpunktes einer Kurve zu den Koordinaten des Anfangspunktes der nächsten Kurve passt. Prinzipiell sind insgesamt mehr Informationen als nötig vorhanden und es hängt stark von den verwendeten Software ab, welche benutzt und welche verworfen werden. Deshalb ist es anzuraten immer korrekte und nicht widersprüchliche Daten einzutragen.

5.10.3. Mantelfläche

Die Mantelfläche besteht aus verschraubten Flächen, die durch die Verschraubung der Kreisbögen der Grundfläche erzeugt werden. Die Approximation für die Begrenzungskurven und die der dabei entstehenden Flächen wurden zuvor berechnet. Die Definition der Fläche wird wie in #42 als advanced face erzeugt.
Die Kreisbögen werden zuerst, wie in Abschnitt 5.10.2 erzeugt. Die Helixkurven werden durch folgenden Code erzeugt.

```
#1832=ORIENTED_EDGE('',*,*,#1833,.T.);
#1833=EDGE_CURVE('',#1834,#1836,#1838,.T.);
#1834=VERTEX_POINT('',#1835);
#1835=CARTESIAN_POINT('',(38.7,-23.3,0.));
#1836=VERTEX_POINT('',#1837);
#1837=CARTESIAN_POINT('',(38.7,-23.3,250.));
#1838=B_SPLINE_CURVE_WITH_KNOTS('',3,(#1839,#1840,...,#1870),
.U.,.F.,.F.,(4,1,...,1,4),(0.,1.,...,28.,29.),.U.);
#1839=CARTESIAN_POINT('Ctrl Pts',(38.7,-23.3,0.));
...
```

Die Kante wird wie gehabt über #1832 definiert, mit dem Unterschied dass die Kurve #1833 jetzt auf eine B-Spline-Kurve mit Knotenvektor #1838 verweist. Diese beinhaltet den Kurvengrad 3, den Verweis auf die Kontrollpunkte, die anschließend nacheinander aufgelistet werden. Das .U. sagt aus, dass die Kurvenform nicht genauer spezifiziert wird. Die nächsten beiden .F. sagen, dass es sich um keine geschlossene Kurve handelt und dass sich die Kurve nicht selbst schneidet. Der Vektor (4,1,...,1,4) gibt an, welche Vielfachheit

die Knoten des Knotenvektors haben. Der im Anschluss angegebene Knotenvektor besitzt keine vielfachen Knoten mehr, da diese Information schon im Vektor zuvor angegeben wurde. Im nächsten Feld ist der Knotentyp mit .U. angegeben, da dazu keine genauere Information in diesem Fall möglich ist.
Um eine abgeschlossene Fläche zu erzeugen, muss noch die B-Spline Fläche angegeben werden. Diese wird damit erzeugt:

```
#1921=B_SPLINE_SURFACE_WITH_KNOTS('',3,3,((#1922,...,#1953),...,
(#2018,...,#2049)),.U.,.F.,.F.,.F.,(4,4),(4,1,...,1,4),(0.,1.),
(0.,1.,...,28.,29.),.U.);
#1922=CARTESIAN_POINT('Ctrl Pts',(36.7,-24.2,0.));
...
```

Im wesentlichen ist so eine Fläche wie eine Kombination aus 2 B-Spline Kurven. Deshalb wird jede Eigenschaft für 2 Richtungen angegeben. Die Fläche hat in beiden Richtungen den Grad 3. Die Knotenpunkte werden jeweils in Sets in eine Richtung angegeben. In diesem Fall sind jeweils die Knoten in Längsrichtung zu insgesamt 4 Sets zusammengefasst. Die nächsten Eigenschaften beschreiben eine unspezifizierte Kurvenform und dass die Fläche in beide Richtungen nicht geschlossen ist. Mit der nächsten Option wird gesagt, dass die Fläche sich nicht selbst schneidet. Die Knotenmultiplizitäten und die Knotenvektoren werden in den nächsten 4 Feldern beschrieben. Im Anschluss werden wie gehabt die Kontrollpunkte definiert.

5.10.4. Objektdefinition

Zum Abschluss der Geometriedefionition müssen die zuvor definierten Flächen zusammengefügt werden. Dieser Schritt wird über eine geschlossene Schale, die die Flächen aufnimmt durchgeführt.

```
#20378=MECHANICAL_DESIGN_GEOMETRIC_PRESENTATION_REPRESENTATION(
'',(#20381),#13);
#20379=SHAPE_REPRESENTATION_RELATIONSHIP('SRR','None',#22,#20380);
#20380=ADVANCED_BREP_SHAPE_REPRESENTATION('',(#20382),#12);
#20381=STYLED_ITEM('',(#35),#20382);
#20382=MANIFOLD_SOLID_BREP('Solid1',#20383);
#20383=CLOSED_SHELL('',(#42,....,#20146));

ENDSEC;
END-ISO-10303-21;
```

Die Zeile #20378 beschreibt den Körper, der im wesentlichen durch #20383 als geschlossene Schale definiert wird. Die Mantel- und Deckflächen werden durch diese Zeile zusammengefügt. Die nicht aufgeführten Zeilen #13 bis #41 sind die allgemeinen Eigenschaften, auf die nicht näher eingegangen wird. Die letzten beiden Zeilen schließen den Datenabschnitt und die Datei ab. etwaige Daten unterhalb werden ignoriert.

5.11. Erzeugung von STL-files

STL-Files beschreiben die triangulierte Oberfläche eines Objektes. Dazu wird jedes einzelne Dreieck davon durch die kartesischen Koordinaten der Eckpunkte und der Richtung der Flächennormalen gespeichert. Es gibt neben binären STL-Dateien auch ASCII-STL Dateien, die in dieser Arbeit benutzt wurden.
ASCII-STL sind im Klartext gespeichert und können somit von praktisch jeder Textverarbeitungssoftware gelesen werden. STL-Dateien besitzen einen vergleichsweise hohen Speicherverbrauch durch den Klartextcharakter und dass jeder Eckpunkt der Dreiecke der Oberfläche zumindest von 3 Facetten benutzt wird und in jeder Facette gespeichert wird.

5.11.1. Aufbau

ASCII-STL fangen mit dieser Zeile an, wobei name optional ist.

```
solid name
```

Im folgenden wird ein Dreieck nach dem anderen unstruktiert aufgelistet. Die Länge des Normalenvektors ist unerheblich und die Eckpunkte der Dreiecke dürfen nicht negativ sein.

```
facet normal ni nj nk
        outer loop
                vertex v1x v1y v1z
                vertex v2x v2y v2z
                vertex v3x v3y v3z
        endloop
endfacet
```

Nach der Auflistung wird die Datei mit der Zeile beendet.

```
endsolid name
```

Der Aufbau ließe auf die Möglichkeit von anderen Facettenformen schließen, allerdings werden in der Praxis nur Dreiecke unterstützt. Außerdem erkennt man die große Menge an "Füllwörtern", die nur der Lesbarkeit dienen. Dieser Eindruck bestätigt sich durch die Effizienz einfacher gebräuchlicher Komprimierungsverfahren (.zip und .7z). Dabei sind im Praxistest Speicherplatzeinsparungen von etwa 90% bis 96% beobachtet worden.

5.11.2. Triangulation der Oberfläche

Bei der Oberfläche handelt es sich um eine verschraubte Fläche, deren Querschnitt punktweise als Kontur vorliegt. Die Punkte sind dabei bereits gereiht. Zur Berechnung der jedenfalls nötigen Punkte zur Einhaltung der Toleranz wird eine Gerade zwischen 2 Punkten P1 und P2 gebildet und der Normalabstand lt. Gleichung 27 aller Punkte P3 dazwischen berechnet. Falls kein Normalabstand gefunden wurde, der größer als die Toleranz ist, wird für den Punkt P2 der nächste in der Liste gewählt und der Normalabstand zu der Geraden für alle Punkte dazwischen berechnet. Sobald der Normalabstand größer als die Toleranz ist, wird der Punkt P2 in eine neue Liste zwischengespeichert. P2 wird zu P1 und für P2 wird wie gehabt der nächste Punkt der Kontur gewählt. Dies wird solange wiederholt, bis die gesamte Kontur vereinfacht wurde.

$$
\begin{aligned}
e =& abs\left(\frac{dy_2 \cdot dx_3 - dx_2 \cdot dy_3}{\sqrt{dx_2 \cdot dx_2 + dy_2 \cdot dy_2}}\right) && (27)\\
dx_2 &= P2_x - P1_x & dy_2 &= P2_y - P1_y\\
dx_3 &= P3_x - P1_x & dy_3 &= P3_y - P1_y
\end{aligned}
$$

Um eine möglichst einfache Berechnung zu ermöglichen, wird die Gesamtlänge in gleich große Abschnitte unterteilt. Dadurch werden die im vorherigen Schritt ermittelten Punkte bis auf eine Drehung um die Mittelachse wiederbenutzt. Die Abschnittshöhe wird wieder über die maximale Toleranz ermittelt. Von den vorhin berechneten Punkten geht jeweils eine Helixkurve aus, die durch Geradenstücke approximiert werden soll. Die Berechnung des Normalabstands zwischen diesen muss nur in der Projektion in die Grundebene durchgeführt werden, da der Normalabstand durch diese Projektion nicht verzerrt wird. In Abbildung 12 und 13 ist dies exemplarisch dargestellt, wobei die schwarz gezeichnete Kontur die triangulierte Fläche, die rot gepunktetete Geraden die Triangulation darstellt und die blauen Linien die Projektion darstellt.

Somit ist der Normalabstand einfach die Höhe des entsprechenden Kreissegments und unter Berücksichtigung der maximalen Toleranz lässt sich lt. Gleichung 29 der maximale Kreisbogenwinkel berechnen, wobei für den Radius der maximale zu benutzen ist.

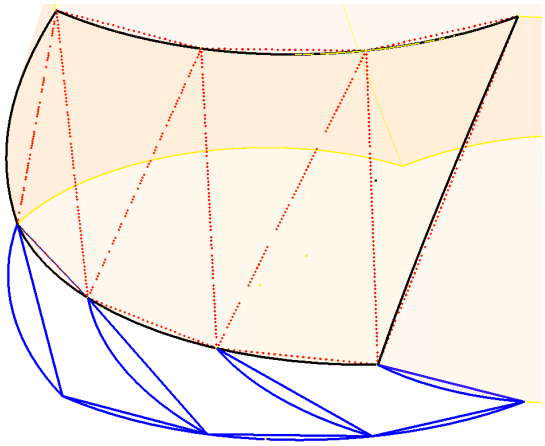


Abbildung 12: Illustration der Triangulation

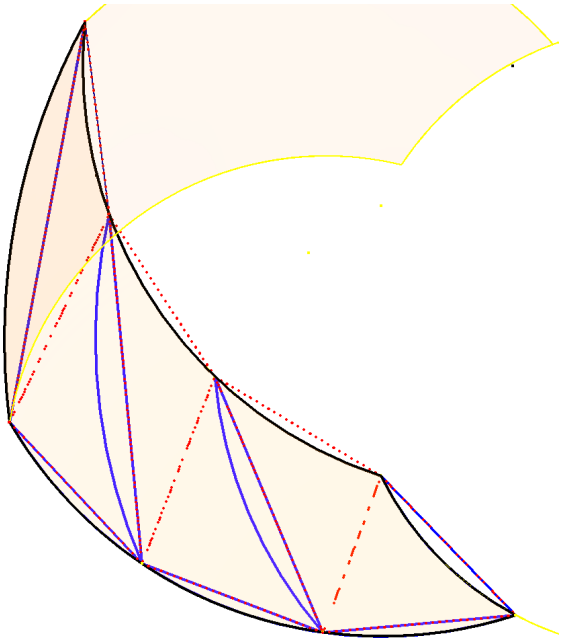


Abbildung 13: Illustration der Triangulation in die Grundebene projiziert

$$e = r \cdot \left(1 - \cos\left(\frac{\alpha}{2}\right)\right) \tag{28}$$

$$\alpha = 2 \cdot acos\left(1 - \frac{e}{r}\right) \tag{29}$$

Über den Steigunswinkel der Helixkurve kann man damit den zulässigen Höhenschritt berechnen. Der Normalabstand von der dritten Kante lässt sich nicht wie bei den anderen Kanten berechen, allerdings ist der Fehler jedenfalls kleiner als der an den anderen beiden.

5.11.3. Triangulation der Grund und Deckflächen

Um den Körper abzuschließen müssen die Grund und Deckflächen ebenso trianguliert werden. Dazu können die, bereits im vorigen Schritt, triangulierten Punkte benutzt werden. Bei einfachen Geometrien, können für die Dreiecke jeweils 2 aufeinanderfolgende Punkte und der Mittelpunkt benutzt werden, ohne sich überlappende Dreiecke zu erzeugen.
Vor allem für Nebenläufer ist oft eine komplexere Geometrie erforderlich, deren Zähne Hinterschnitte aufweisen. Solche Geometrien können mit folgender Vorgehensweise trianguliert werden: Zuerst wird der Punkt mit dem geringsten Radius gesucht und von diesem ausgehend werden mit aufeinanderfolgenden Punkten und dem Mittelpunkt Dreiecke erzeugt. Dabei wird zu jedem Punkt der Winkel für beide Punkte berechnet. Sobald die Differenz der Winkel das Vorzeichen wechselt, ist die Kontur hinterschnitten.
Damit wird die Dreieckserzeugung unterbrochen und der nächste Punkt, der zu dem ersten Punkt eine Winkeldifferenz, wie das Dreieck zuvor aufweißt wird gesucht. Ist dieser gefunden, wird die Dreieckserzeugung mit aufeinanderfolgenden Punkten fortgesetzt, und statt dem Mittelpunkt wird der gefundene Punkt benutzt.
Sobald der Punkt doppelt benutzt werden würde, wird die ursprüngliche Dreieckserzeugung fortgesetzt.
Zu beachten ist dabei, dass Hinterschneidungen in beide Richtungen auftreten können. Dieses Problem lässt sich verhindern, indem zuvor alle Vorzeichenwechsel der Winkeldifferenz ermittelt werden, und gleichzeitig die Punktliste in jene Bereiche unterteilt wird, die kein Überlappen erzeugen.
Dieses Verfahren ist in Abbildung 14 Beispielhaft übertrieben dargestellt. In Schwarz ist die hinterschnittene Kontur, mit den Hinterschneidungsgrenzen strichliert gezeichnet.

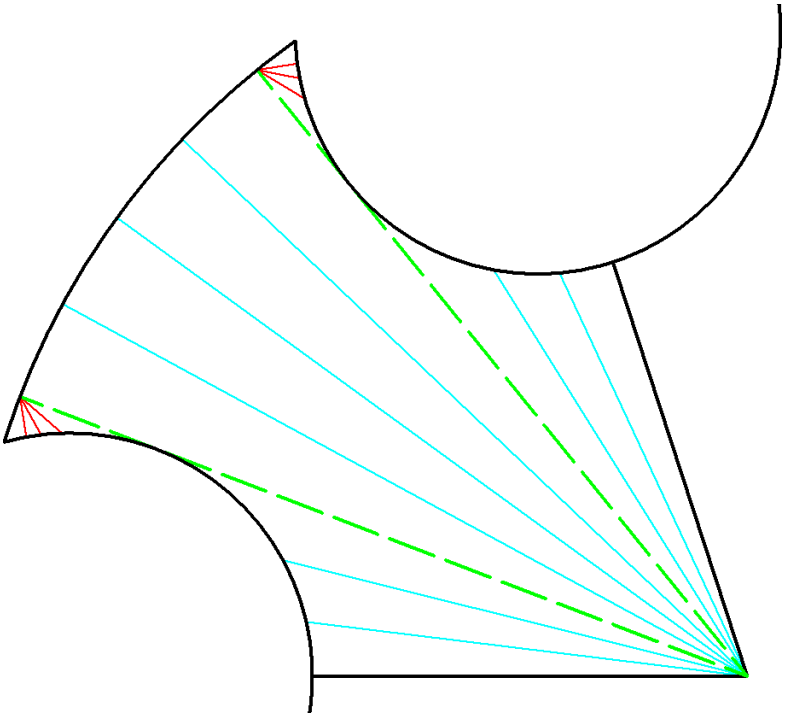


Abbildung 14: Illustration der Triangulation

6. Flächenverlauf

6.1. Berührpunktermittlung

Die Berührpunkte zwischen den Läufern und dem Gehäuse dienen der Begrenzung der Saug- und Kompressionsflächen und sind exemplarisch in Abbildung 15 dargestellt. Für den benutzten Algorithmus ist es unerheblich, wieviele Berührpunkte gleichzeitig vorkommen. Der Algorithmus erfordert einen ersten und letzten Berührpunkt (Punkte A,B und E,F), die als jene Punkte auf der Läuferkontur definiert sind, welche auf dem Kopfkreis des Läufers liegen und bezogen auf den Läufermittelpunkt einen größeren (bzw. kleineren) Winkel als der ± Gehäusewinkel aufweisen. Da die Läuferkontur als Kontur eines Zahns vorliegt, wird die Nummer des entsprechenden Punktes und der Rotation auf die Zahnposition gespeichert.
Zuerst wird ein Punkt gesucht, der zwischen den Gehäusekanten liegt. Dazu wird der Winkel jedes Punktes eines Zahns berechnet und solange auf den nächsten Zahn rotiert, bis der Winkel des Punkts zwischen den Gehäusekanten liegt.

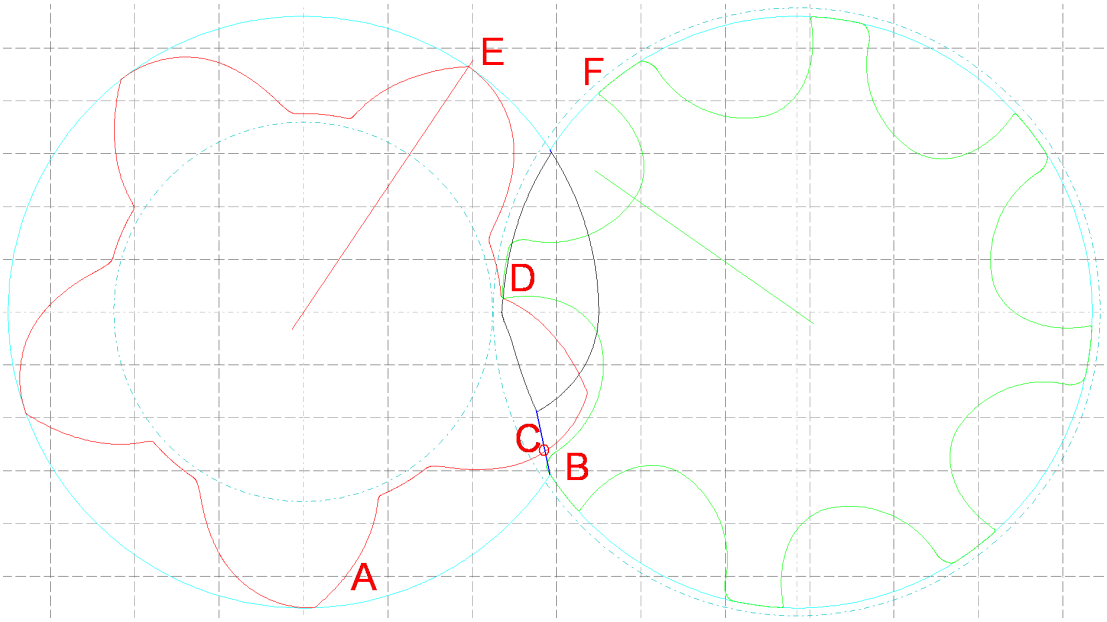


Abbildung 15: Illustration der Berührpunkte

Zur Gehäuseberührpunkteberechnung wird anschließend von dem Startpunkt in beide Richtungen gesucht, bis der erste Punkt außerhalb der Gehäusekanten liegt, und der Radius dem Gehäuseradius entspricht. Um etwaige Ungenauigkeiten auszugleichen, reicht

es für den Algorithmus, wenn die Radiusdifferenz weniger als 0.05% beträgt. Dies wird für beide Läufer ausgeführt.
Um alle Kandidaten für innere Berührpunkte (D) zu finden, wird zu jedem Punkt der Eingriffslinie (in Abbildung 15 schwarz dargestellt) der Abstand zum entsprechenden Punkt am Haupt- und Nebenläufer berechnet und addiert. Dieser Wert gibt näherungsweise an, wie weit die entsprechenden Haupt- und Nebenläuferpunkte voneinanander entfernt sind.
Außerdem kann die Situation vorkommen, dass der erste Punkt auf der Eingriffslinie ein Berührpunkt ist. Dies wird überprüft indem der Abstand klein sein soll, und gleichzeitig eine ansteigende Tangente aufweist. Wie klein der Abstand sein muss, hängt vom Diskretisierungsgrad der Kontur ab. Dabei wird die Distanz zwischen zwei aufeinanderfolgender Punkte zueinander als zulässiger Abstand interpretiert (näheres im nächsten Unterkapitel).
Berührpunkte werden für diesen Abschnitt gefunden, indem die Abstandsliste auf Minima untersucht wird. Die Abstände können von dem einen Punktpaar zum nächsten relativ stark durch die Diskretisierung variieren. Das folgende Schema ist in Abbildung 16 dargestellt. Es wird dazu ein Variationsfenster der Größe H gebildet, das seine Position solange beibehält, bis der Abstand aus dem Fenster herausfällt. Dann wird das Fenster durch minimale Bewegung soweit verschoben, bis der Abstand wieder im Fenster liegt (1). Diese Vorgehensweise wird realisiert, indem bei überschreiten der Obergrenze, diese auf den neuen Wert gelegt wird, und anschließend die Untergrenze um H unter die Obergrenze gelegt wird. Zur Berührpunktesuche wird die Abstandsliste mit dem Fenster solange durchgegangen, bis ein Abstand die Fensterobergrenze übersteigt, wenn davor zumindest ein Punkt das Fenster zu niedrigeren Abständen verschoben hat (2). Anschließend wird jener Abstand in die andere Listenrichtung gesucht, der die Obergrenze überschreitet (3). Als Berührpunkt wird jener Punkt gewählt, der in der Mitte der beiden Obergrenzüberschreitungen liegt (4). Dadurch kann es zwar zum Verschmelzen von Berührpunkten kommen, allerdings ist der Effekt jedenfalls sehr klein, da dies nur bei kleinen Abständen vorkommt und dementsprechend der Einfluss auf die Flächen dazwischen klein ist. Das nächste Minimum wird somit erst gefunden, wenn zumindest ein Punkt die Obergrenze überschreitet (5)
Durch die Bedingung, dass Berührpunkte über Abstandsminima definiert wurden, werden jene Berührpunkte gelöscht, deren Abstand relativ groß ist und damit kein Berührpunkt sein kann.
Im Saug- und Blaslochbereich wird jeweils ein Berührpunkt (C) erzeugt, wenn der Hauptläufer eine Gerade von der Eingriffslinie zur Gehäusekante schneidet. Falls ein

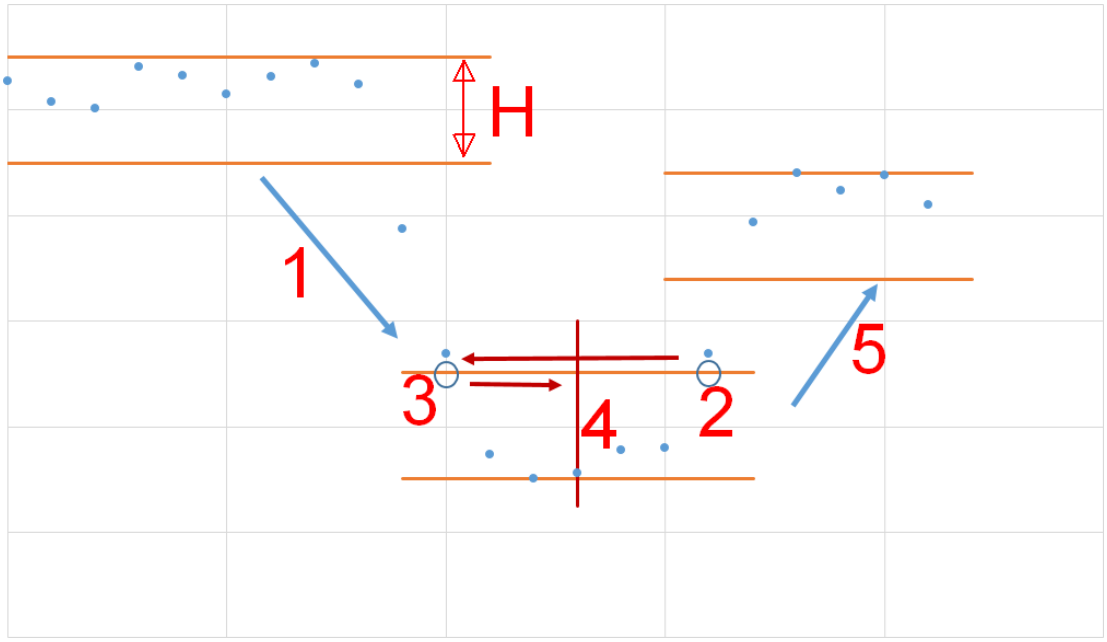


Abbildung 16: Illustration des Glättungsalgorithmusses

Schnittpunkt gefunden wurde, aber der Gehäuseberührpunkt näher der Gehäusekante liegt, wird der Schnittpunkt wieder verworfen. Der zugehörige Nebenläuferpunkt ist jener mit dem geringsten Abstand zum Schnittpunkt.

Abstandsvariation zweier paralleler diskretisierter Linien

Die Abstandsberechnung zwischen zwei unterschiedlich diskretisierter Linien kann relativ stark variieren, auch wenn die Linien annähernd parallel sind. Dies soll durch

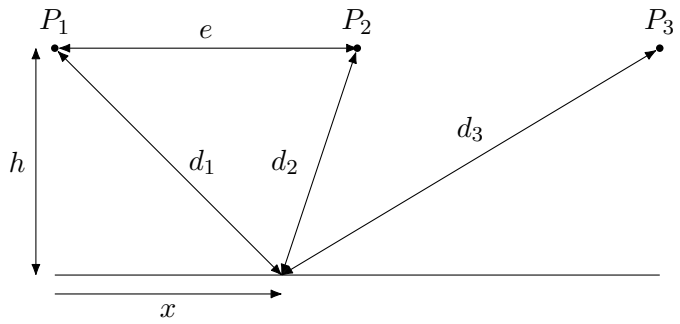


Abbildung 17: diskretisierte parallele Geraden

die Betrachtung zweier paralleler Geraden nach Abbildung 17 gezeigt werden. Dazu wird der Abstand von jedem Punkt einer diskretisierten Geraden zu einer parallelen Geradenfunktion berechnet. Der minimale Abstand ist der Normalabstand, während der größte dort vorliegt, wo der Abstand zum nächsten Punkt gleich groß ist. Dementsprechend ergeben sich folgende Abstände: Der Abstand ist genau zwischen zwei Punkten am

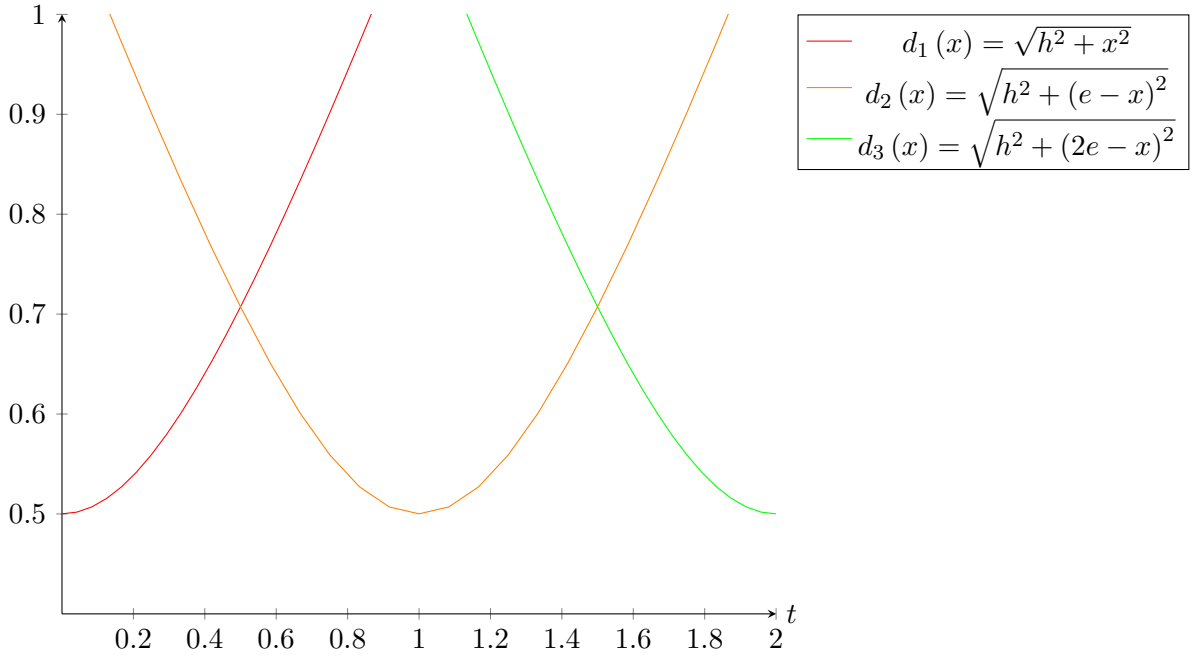


Abbildung 18: Abstandsvariation paralleler Geraden

höchsten und bei den Punkten jeweils am niedrigsten. Damit lässt sich die Variation folgendermaßen berechnen:

$$H = d_1\left(x = \frac{e}{2}\right) - d_1(x = 0) \tag{30}$$

$$= \sqrt{h^2 + x^2} - \sqrt{h^2 + (e-x)^2} \tag{31}$$

$$\lim_{h \to 0} H = x - (x - e) = e \tag{32}$$

$$\lim_{h \to \infty} H = 0 \tag{33}$$

Damit nimmt die maximale Variation den Abstand zwischen zwei Punkten an und ist maximal bei 2 Geraden, die übereinanderliegen. Bei größerem Abstand der Geraden wird die Variation immer kleiner, bis sie 0 wird.

Flächenberechnung Nachdem die Berührpunkte gefunden wurden, wird die freie Fläche zwischen diesen ermittelt. Diese wird auf Haupt- und Nebenläuferflächen aufgeteilt und die gemeinsame Fläche jeweils proportional dem Anteil an der Gesamtfläche zugeschlagen. Die Flächenberechnung an sich ist über eine Summierung von Kreisringsektoren implementiert. Dazu wird der Winkel jedes Punktes zu einer horizontalen berechnet, und mit der Trapezregel numerisch integriert. Die Fläche des Hauptläufers wird dabei durch den Nebenläufer reduziert. Es wird nur jener Anteil des Nebenläufers berücksichtigt, der in den Kopfkreis des Hauptläufers hineinragt.

$$\Delta A_i = \frac{\left(R_{i+1}^2 - R_i^2\right) \cdot \left(\phi_{i+1} - \phi_i\right)}{2} \tag{34}$$

Weil die Berührpunkte zwischen Haupt- und Nebenläufer nicht jeweils die gleichen Punkte sein müssen, wird zu der Integration noch eine Ausgleichsfläche nach der selben Art und Weise wie die Teilflächen berechnet.
Die Kopfkreise des Haupt- und Nebenläufers überschneiden sich und bilden einen linsenförmigen Bereich in dem sich Haupt- und Nebenläuferflächen überdecken. Die genaue Aufteilung der doppelt gerechneten Fläche lässt sich nur über eine Strömungssimulation feststellen. Näherungsweise wird der überdeckte Bereich entsprechend dem Anteil an der Gesamtfläche aufgeteilt.

6.2. Flächensuche

Die für jeden Winkelschritt berechneten Flächen sind in Druck- und Saugflächen zu unterteilen, um einen Flächenverlauf über die Läuferlänge zu erhalten. Jene Flächen, die den unteren Gehäuseberührpunkt beinhalten werden als Saugflächen bezeichnet, und jene die den oberen Gehäuseberührpunkt beinhalten werden als Druckflächen bezeichnet. Im ersten Schritt wird jeweils der Winkel der größten Druck- und Saugfläche ermittelt, indem für jede Läuferstellung der Winkel der Gehäuseberührpunkte des Hauptläufers berechnet wird. Die Läuferstellung, bei der der Winkel maximal wird, ist die Startstellung des Läufers. Dabei ist für die Druck- und Saugseite im Allgemeinen eine andere Startstellung gegeben (1 und 6). Nachdem die Startstellung für den Hauptläufer gesucht wird, kann es sein, dass die korrespondierende Lücke des Nebenläufers vollständig im Gehäuse liegt. Für jene Fläche wird die Fläche der vollständigen Nebenläuferlücke hinzuaddiert.
Die Lücken können sich während einer Läuferdrehung aufteilen. Deshalb wird bei jedem Winkelschritt überprüft, ob sich die Lücke sprunghaft auf weniger als 90% einer linearen Extrapolation der beiden vorherigen Lücken entwickelt hat (bei 4). Ist dies der Fall werden solange die nebenliegenden Teillücken zur Ausgangslücke hinzugefügt, bis mehr als 90% der Extrapolation erreicht werden. Für die weiteren Schritte wird jeweils die Gesamtlückenfläche benutzt. Dieser Vorgang wird solange wiederholt, bis der Ausgangswinkel erreicht wird.
Ab diesem Winkel treten zugleich 2 Flächenabschnitte auf (2). Ab diesem Abschnitt werden die einzelnen Teilflächen nach der Größe der vorherigen Fläche zugeordnet und überprüft ob eine Extrapolation der vorherigen Teilflächen zu einem guten Ergebnis kommt. Dies wird solange fortgeführt, bis die Gesamtfläche weniger als 0.1% der maximalen Zahnlücke ist. Weiters wird die Berechnung für Druckflächen wiederholt. (4-6)
Der Flächenverlauf wird zuerst getrennt für den Haupt- und Nebenläufer ermittelt. Dazu werden zuerst jeweils die Saugflächen in eine Liste geschrieben, sodass die kleinsten Saugflächen zuerst in der Liste stehen. Anschließend bleibt die Fläche gleich groß, da die Zahnlücke vollständig im Gehäuse ist. Der Winkel bis die Fläche wieder aus dem Gehäuse austritt wird für den Haupt- und, mit den Größen für den Nebeläufer, für den Nebenläufer mit Abbildung 20 wie folgt ermittelt:

$$\alpha_{konst} = 2\pi - 2\varphi_{max} - \beta_n + \alpha_{rmax} \tag{35}$$

Dabei bleibt die Zahnlücke konstant. Das Ende des Flächenverlaufs wird durch die Druckflächen gebildet, die in die Liste geschrieben wird. Diese Schema wird für den Neben-

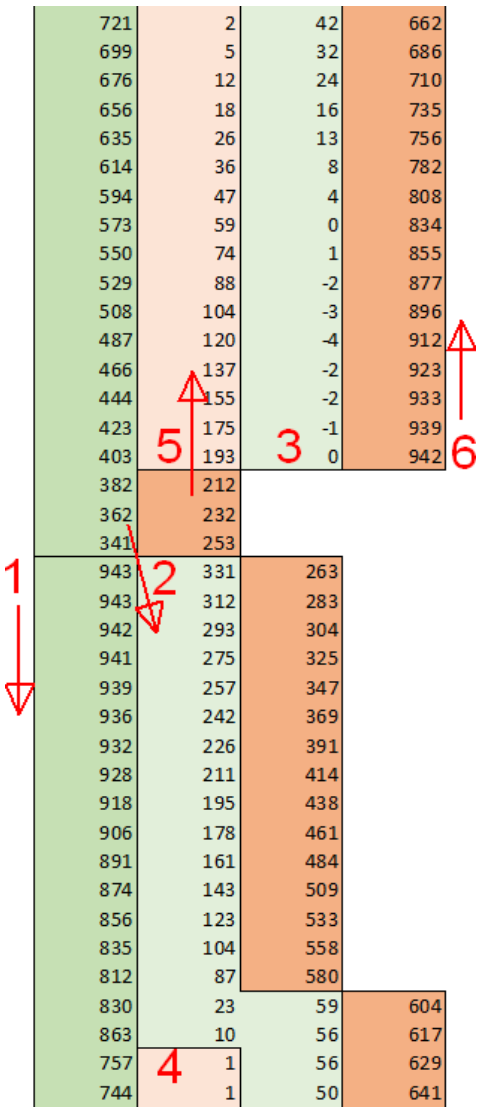


Abbildung 19: Illustration der Flächensuche

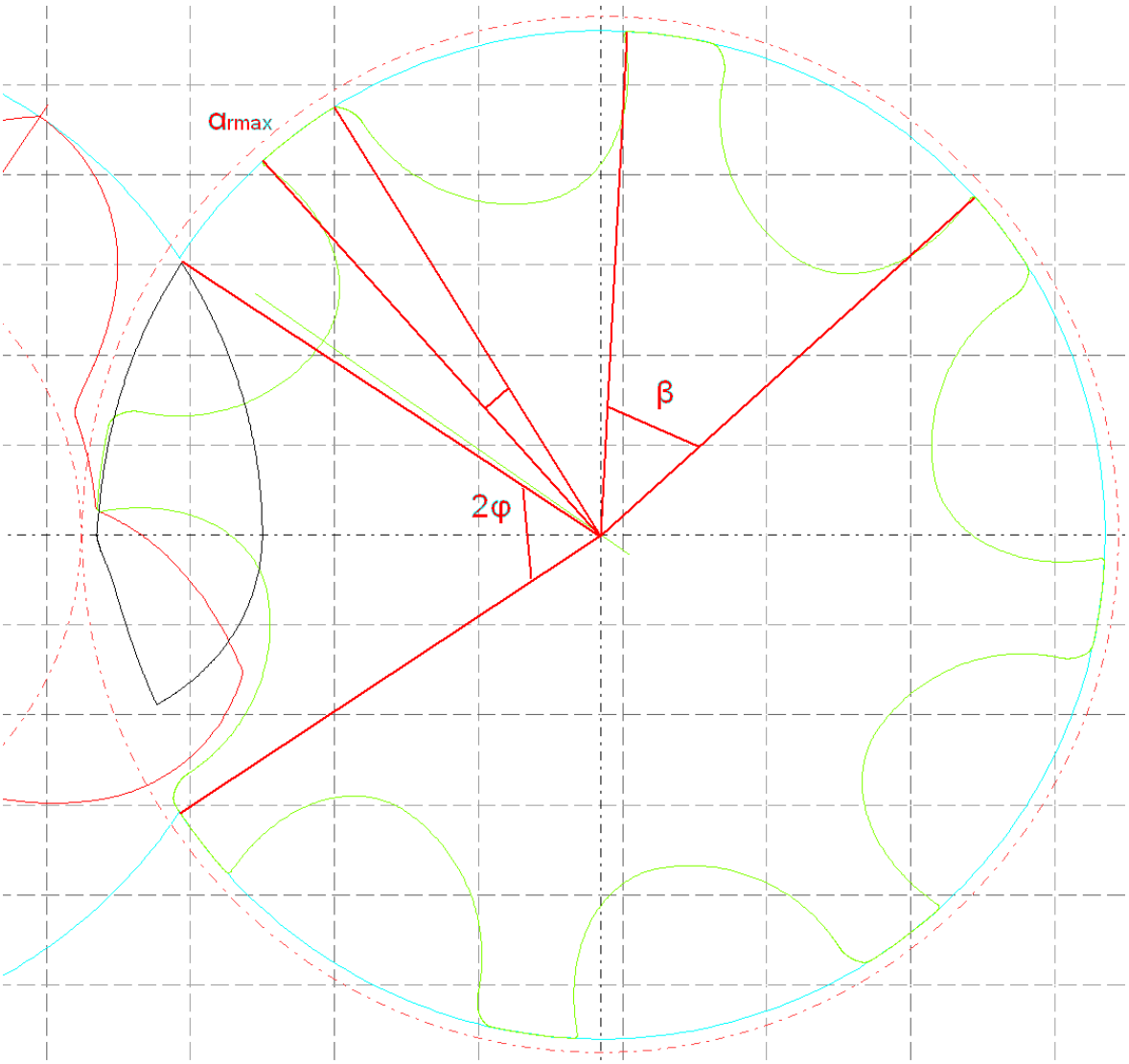


Abbildung 20: Illustration der benutzten Winkel

läufer wiederholt. Beide Listen werden anschließend addiert um den Gesamtflächenverlauf zu erhalten. Dabei ist zu beachten, dass es bei Zähnezahldifferenzen zwischen Haupt- und Nebenläufer, die größer als 1 sind, zu einer Horizontalen (Kante A in Abbildung 21) im Verlauf kommt, da die Zahnlücke länger für den einen Läufer konstant bleibt, da der Flächenverlauf auf die Hauptläuferdrehung bezogen ist und eine interne Übersetzung vorliegt.

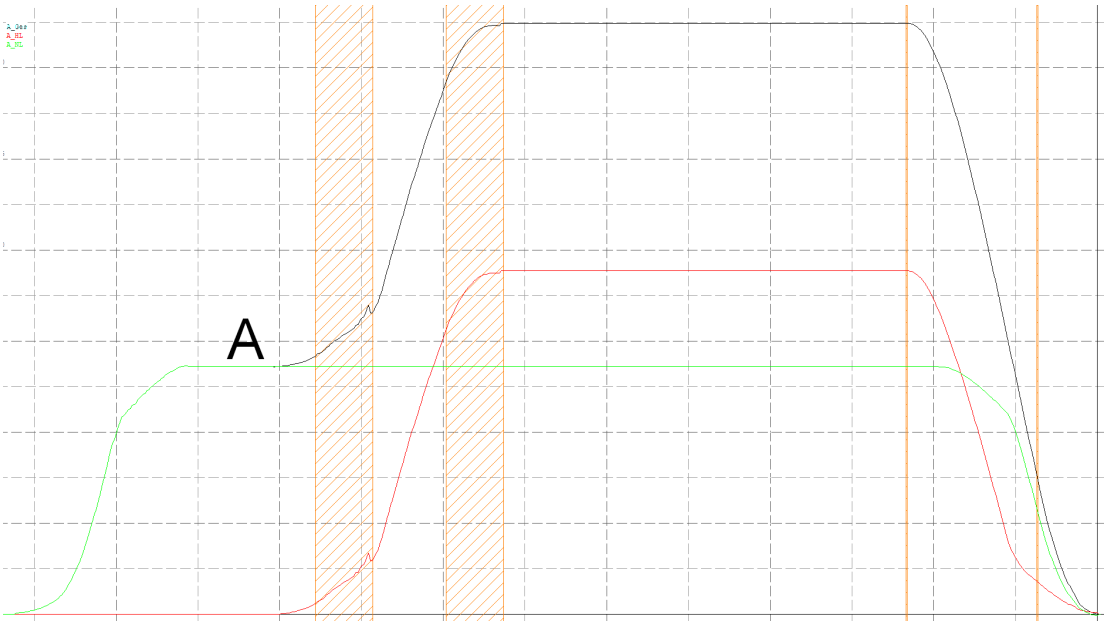


Abbildung 21: Illustration des resultierenden Flächenverlaufs, schwarz: Gesamtfläche, rot: Hauptläuferfläche, grün: Nebenläuferfläche

Abbildung 21 zeigt das Resultat der in diesem Abschnitt gezeigten Flächensuche. Die grüne Kurve zeigt den Verlauf der Nebenläuferflächen, die rote Kurve den Verlauf der Hauptläuferflächen und die schwarze den Gesamtflächenverlauf. Bei diesem Beispiel hat der Nebenläufer eine um mehr als einen Zahn höhere Zähnezahl. Dementsprechend weist der Nebenläuferflächenverlauf eine Horizontale auf, die länger als die des Hauptläuferflächenverlaufs ist. Außerdem weißt der Gesamtflächenverlauf bei A eine Kante auf.

6.3. Volumenverlauf

Um die im Schraubenkompressor wirkenden Drücke und den zu erwartenden Volumenstrom zu ermitteln wird das durch die Zahnlücken erzeugte Volumen bestimmt. Um den Volumenverlauf einer Zahnlücke zu berechnen wird mithilfe der Trapezregel numerisch der Flächenverlauf des Haupt- und Nebenläufers integriert und beide anschließend addiert. Die untere Grenze der Integration ist zu Beginn die Druckseite. Die obere Grenze wandert in 1 Grad Schritten (dadurch ergibt sich jeweils ein z-Schritt von $\Delta z = \frac{ll}{hlwin}$) zur Saugseite.
Wird die obere Grenze größer als die Maximale Ausdehnung der Lücke verschiebt sich die untere Grenze mit der oberen Grenze mit. Das heißt, dass das Zahnlückenvolumen ab diesem Zeitpunkt konstant ist. Allerdings kann die obere Grenze schon vorher das Läuferende erreichen. Dadurch ergibt sich als Obergrenze der Integration die Läuferlänge. Nach Addition der beiden Volumenverläufe ergibt sich aus dem Flächenverlauf Abbildung 21 der Volumenverlauf Abbildung 22.

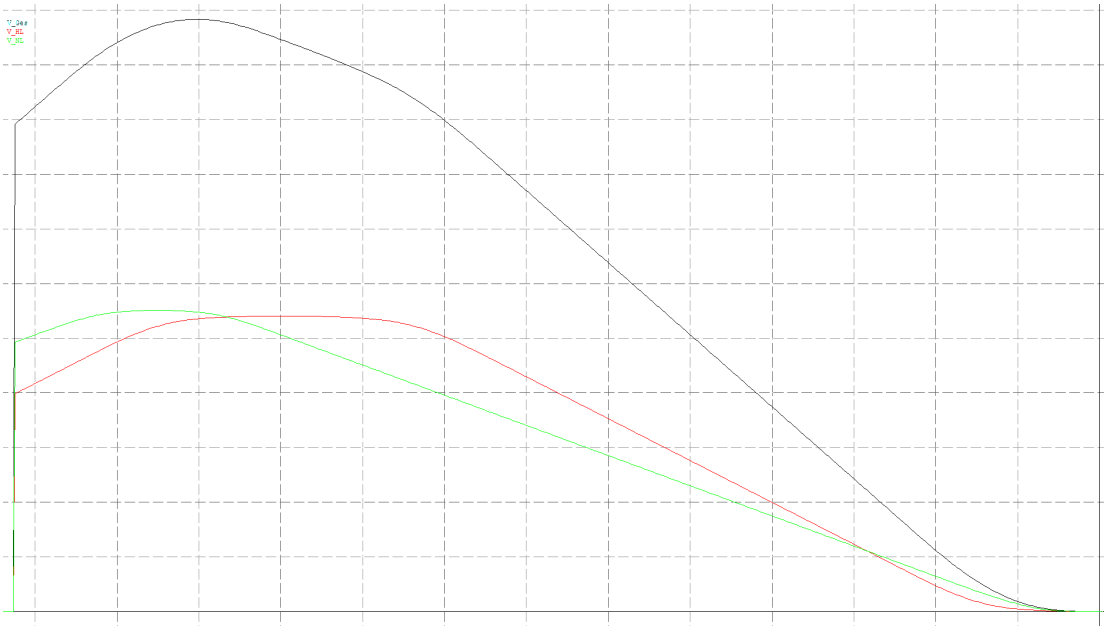


Abbildung 22: Illustration des resultierenden Volumenverlaufs, schwarz: Gesamtvolumen, rot: Hauptläufervolumen, grün: Nebenläufervolumen

6.4. Spaltverläufe

Das Zahnvolumen wird durch Berandungen am Gehäuseumfang, den Läufern und an der Druck- und Saugseite begrenzt. An den Stoßstellen dieser Flächen kommt es zu einem Gasaustausch, da Druckunterschiede vorliegen und gewisse Spalte an den Stoßstellen vorhanden sind. Die Spaltfläche wird näherungsweise durch das Produkt der vorhandenen Spaltlänge mit der anzugebenden Spaltbreite ermittelt. Aus diesen Spaltgrößen wird ein Leckspaltparameter bestimmt werden, der ohne Strömungsuntersuchung eine näherungsweise Aussage zum Gasaustausch liefert.
Die jeweiligen Spalte sind in Abbildung 23 dargestellt.

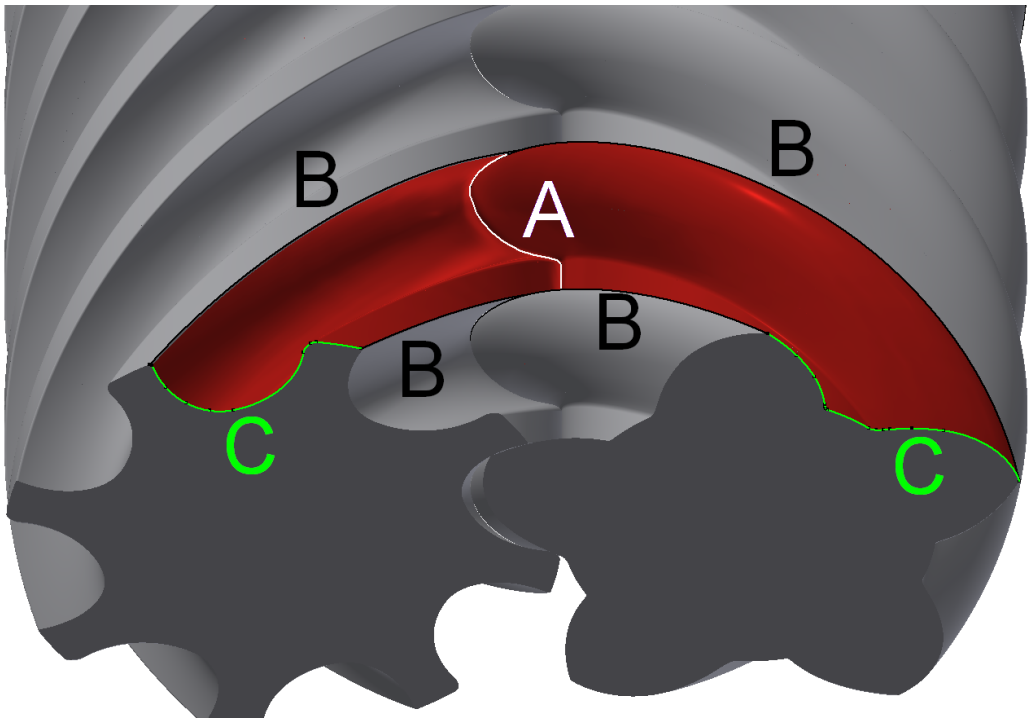


Abbildung 23: Illustration der Spaltarten

6.4.1. Eingriffspalt (A)

Der Eingriffsspalt ergibt sich durch die Berührlinie der beiden Läufer und ist damit die Verbindung der Berührpunkte und ist in Abbildung 24 dargestellt. Die Länge des Eingriffspalts wird durch Hypothenusenbildung zweier zueinander zugehöriger Berührpunkte und Summierung dieser über die Zahnlückenlänge berechnet. Jede Teilfläche kann beliebig viele Berührpunkte gleichzeitig aufweisen. Dementsprechend kann es beliebig viele Segmente des Eingriffspalts geben. Z.b. in Abbildung 25 hat man teilweise 5 Berührlinien gleichzeitig. Die momentanen Berührpunkte werden durch die in Abbildung 19 gefundenen Flächen definiert.

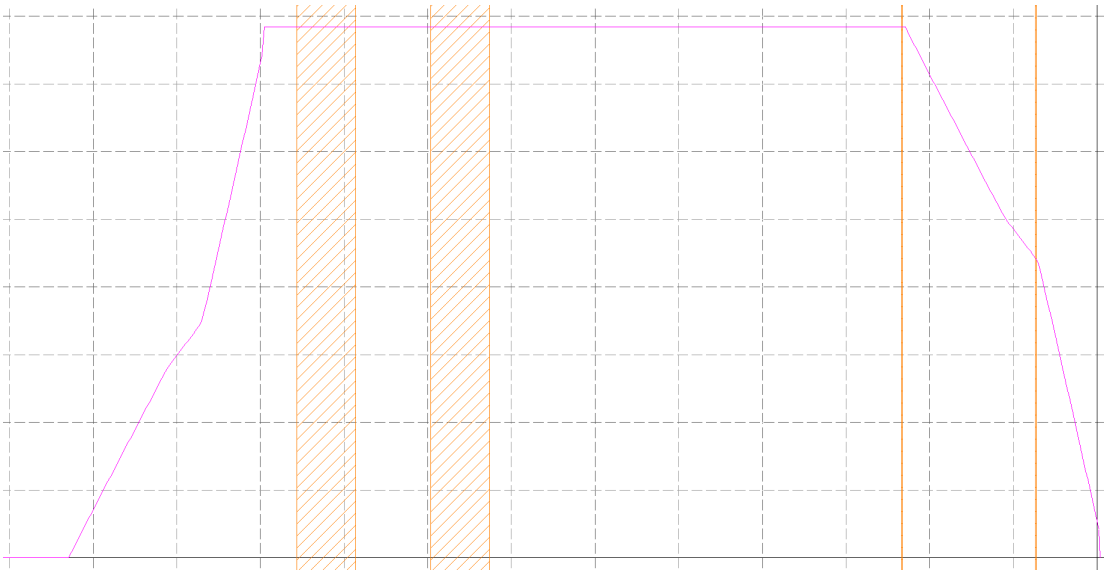
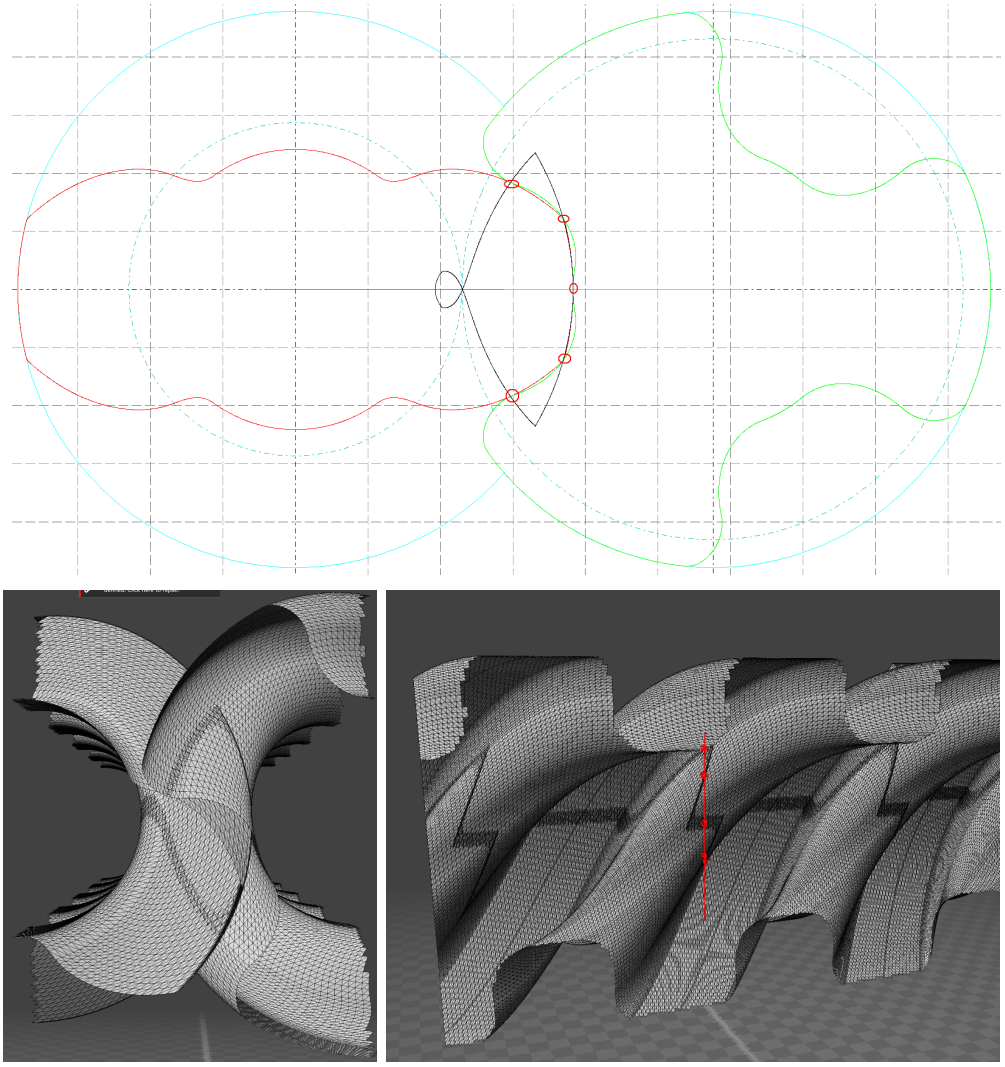


Abbildung 24: Illustration des Eingriffspaltverlaufs

Die Eingriffslinie wird üblicherweise im Stirnschnitt dargestellt. Allerdings ist diese eine räumliche Kurve, die die Druck- und Saugseite trennt. In den folgenden Bildern ist diese näherungsweiße gezeigt.



(a) im Stirnschnitt (b) im Querschnitt

Abbildung 25: Illustration der räumlichen Eingriffslinie mit markierten Berührpunkten

6.4.2. Kopfspalt (B)

Der Kopfspalt bestimmt sich durch die Berührung der Läufer mit dem Gehäuse am Kopfkreisradius und die sich ergebenden Kopfspaltverläufe sind in Abbildung 26 dargestellt. Die Berührpunkte zwischen den Läufern liegen nie am Kopfkreisradius und dementsprechend kann die Kopfspaltlänge ohne numerische Integration ermittelt werden. Zur Berechnung wird zuerst bestimmt ab welchem Winkel der Zahnlücke ein Berührpunkt am Gehäuse vorliegt. Ab diesem Zeitpunkt liegt ein Kopfspalt vor, der die Form einer Helixkurve hat. Da die Geometrie der Helixkurve durch den Kopfkreisradius und der Läuferlänge pro Umschlingungswinkel eindeutig definiert ist, lässt sich die Länge durch 6.4.2 berechnen. $\Delta\phi$ ist der Winkel, den die Zahnlücke beim betrachteten Läuferwinkel berührt.

$$LK = \sqrt{(\Delta\phi r_k)^2 + \left(\frac{ll}{hlwin} \cdot \Delta\phi\right)^2} \tag{36}$$

$$= \Delta\phi \cdot \sqrt{(r_k)^2 + \left(\frac{ll}{hlwin}\right)^2} \tag{37}$$

Somit ergibt sich für den Kopfspaltverlauf eine aus Geraden zusammengesetzt Kurve. Die selbe Vorgehensweise wird für den zweiten Berührpunkt der Zahnlücke mit dem Gehäuse und für den anderen Läufer herangezogen und die Ergebnisse addiert.

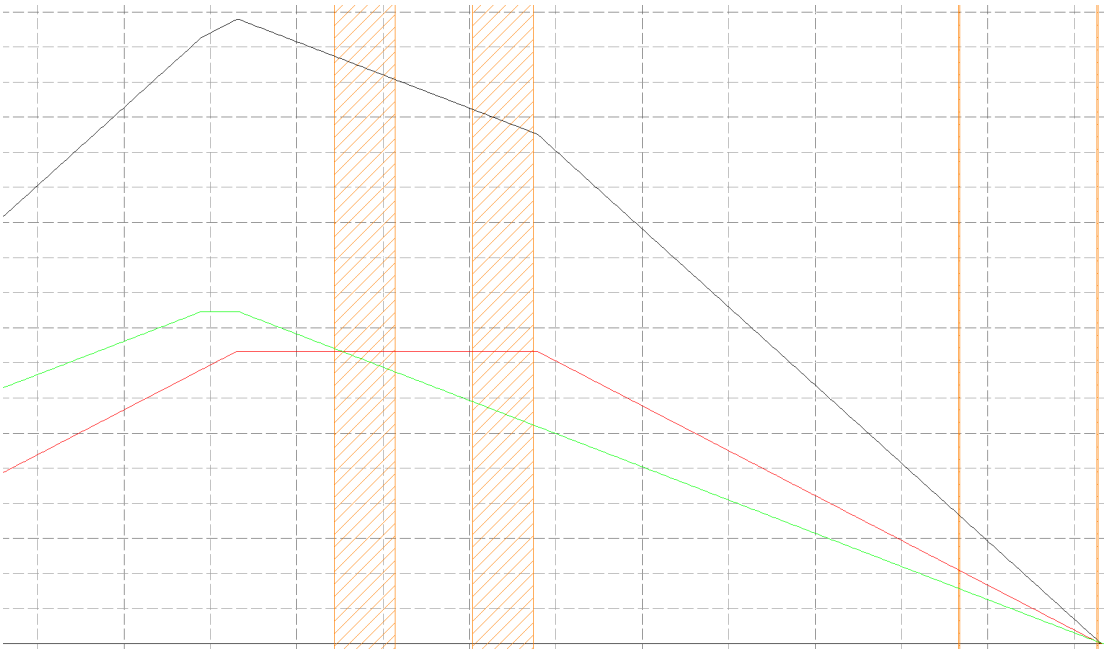


Abbildung 26: Illustration des Kopfspaltverlaufs, schwarz: Gesamkopfspaltverlauf, rot: Hauptläuferkopfspaltverlauf, grün: Nebenläuferspaltverlauf

6.4.3. Stirnspalt (C)

Die Stirnspaltverläufe sind beispielhaft in Abbildung 27 dargestellt. Die Läufer müssen zum Gehäuse an den Stirnflächen einen Spalt aufweisen um u.a. Wärmedehnungen im Betrieb ausgleichen zu können. Dementsprechend wird der Stirnspalt durch die Konturlänge einer Zahnlücke im Stirnschnitt gebildet. Zwischen der Druck- und Saugseite herrscht eine größere Druckdifferenz als zwischen 2 Zahnlücken. Um diesem Umstand in der Rechnung berücksichtigen zu können, wird die Stirnspaltlänge näherungsweise in eine Spaltlänge zur nächsten Lücke und eine zur Saugseite aufgeteilt.
Die Spaltlänge zur Saugseite kann erst existieren sobald die nächste Lücke einen Teil der Saugfläche bildet. Das heißt, dass die Spaltlänge zur Saugseite erst existieren kann, wenn die Zahnlücke aus dem Gehäuse austritt und sich gleichzeitig eine neue Saugfläche ausbildet. Das führt dazu, dass sich der Verlauf in diesem Bereich sprunghaft ändert, da zuvor keine Aufteilung erfolgt, danach hingegen schon.
Die Stirnspaltlänge zur Saugseite wird durch den maximalen Abstand der Kontur zum Lagerdurchmesser begrenzt. Die Restlänge wird der Spaltlänge zur nächsten Lücke zugeordnet.

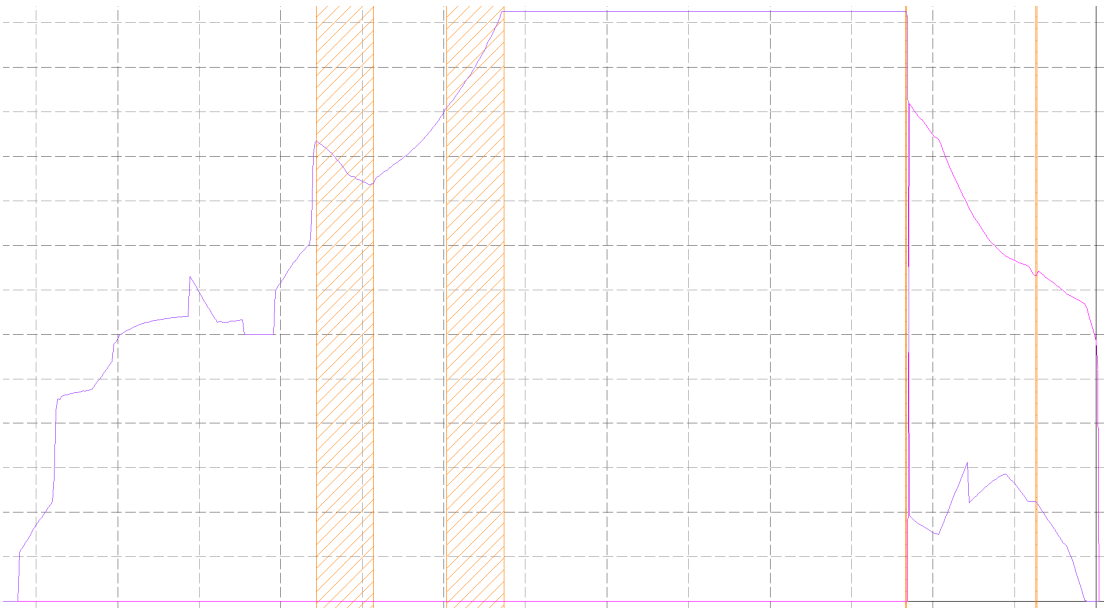


Abbildung 27: Illustration des Stirnspaltverlaufs, violett: Stirnspalt zur nächsten Lücke, pink: Stirnspalt zur Saugseite

6.5. Kräfteberechnung

Die vorhandene Kräfteberechnung ist nur für Schraubenkompressoren bei denen der Haupt- und Nebenläufer unterschiedliche Zähnezahlen aufweisen und deren Querschnitte maximal 3 innere Berührpunkte haben. Der neue Algorithmus ist für beliebige Zähnezahlen und Berührpunkteanzahl modifiziert. Die wesentliche Annahme bei der Berechnung ist, dass der Druck in einer Zahnlücke konstant ist.
Die Berechnung nimmt eine Liste der Berührpunkte über dem Hauptläuferwinkel entgegen. Diese Liste beinhaltet die nötigen Informationen, um alle Konturpunkte zwischen den Berührpunkten berechnen zu können (Indizes der Berührpunkte auf der Kontur, auf welcher Zahnnummer der Berührpunkt liegt und welche Läuferstellung vorliegt). Mit dieser Kontur wird die projizierte Fläche in alle 3 Koordinatenrichtungen und den Schwerpunkten der Flächen berechnet. Damit werden die Kraftkomponenten durch Multiplikation der projizierten Flächen mit dem Druck ermittelt und über eine Zahnlücke numerisch integriert. Außerdem wird angenommen, dass auf die Stirnflächen jeweils der Druck der Saug- bzw. Druckseite herrscht. Mit diesen Kräften lassen sich die Kraftwirkungen auf die Lager und das nötige Drehmoment über die Lebensdauer einer Zahnlücke berechnen. Dies erfolgt jeweils für den Haupt- und Nebenläufer.
Der Unterschied zum alten Algorithmus besteht in der Handhabung der Zahnlückenflächen. Die neue Variante kann pro Zahnlückenwinkel beliebig viele Teilflächen übergeben und die benötigten Größen berechnen. Die Größen der Teilflächen werden in eine entsprechende Größe pro Zahnwinkel umgerechnet. Dadurch kann die vorhandene Kräfteberechnung de facto komplett übernommen werden.
Um die Gesamtlagerkräfte bestimmen zu können müssen alle Zähne der Läufer berücksichtigt werden. Die gesamte Kraftwirkung ergibt sich durch die Addition der unter Druck stehenden Zahnlücken. Somit werden nur Lücken berücksichtigt bis der Zahnlückenwinkel den Umlaufwinkel des Haupt- und Nebenläufers übersteigt.

$$F_{iges} = \sum_{0}^{N} F_i \left(\phi + n \cdot \beta_h\right) \tag{38}$$

$$N = \frac{\phi_{max} - \phi}{\beta_h} \tag{39}$$

Durch die getrennte Berechnung der Kräfte für den Haupt- und Nebenläufer lässt sich der unterschiedliche Winkel, den die beiden für eine komplette Lebensdauer einer Zahnlücke benötigen berücksichtigen. Damit erübrigt sich jede Beschränkung auf die Läuferzähnezahlen.

A. Step-file allgemeine Einstellungen

Die allgemeinen Einstellungen haben auf die Geometrie praktisch keinen Einfluss. Sie dienen im wesentlichen zur Erfüllung der Norm und zur Definition welcher Objekttypen es sich handelt. Außerdem werden die Einheiten der Maße angegeben und wo der Ursprung liegt. Außerdem werden hier die Darstellungsfarben angegeben.

```
#10=UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(0.01),#14,
'DISTANCE_ACCURACY_VALUE',
'Maximum model space distance between geometric entities at asserted c
onnectivities');
#11=UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(1.E-6),#14,
'DISTANCE_ACCURACY_VALUE',
'Maximum model space distance between geometric entities at asserted c
onnectivities');
#12=(
GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#10))
GLOBAL_UNIT_ASSIGNED_CONTEXT((#14,#17,#15))
REPRESENTATION_CONTEXT('','3D')
);
#13=(
GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#11))
GLOBAL_UNIT_ASSIGNED_CONTEXT((#14,#17,#15))
REPRESENTATION_CONTEXT('','3D')
);
#14=(
LENGTH_UNIT()
NAMED_UNIT(*)
SI_UNIT(.MILLI.,.METRE.)
);
#15=(
NAMED_UNIT(*)
SI_UNIT($,.STERADIAN.)
SOLID_ANGLE_UNIT()
);
```

```
#16=DIMENSIONAL_EXPONENTS(0.,0.,0.,0.,0.,0.,0.);
#17=(
CONVERSION_BASED_UNIT('degree',#19)
NAMED_UNIT(#16)
PLANE_ANGLE_UNIT()
);
#18=(
NAMED_UNIT(*)
PLANE_ANGLE_UNIT()
SI_UNIT($,.RADIAN.)
);
#19=PLANE_ANGLE_MEASURE_WITH_UNIT(PLANE_ANGLE_MEASURE(0.01745329252),#18);
#20=SHAPE_DEFINITION_REPRESENTATION(#21,#22);
#21=PRODUCT_DEFINITION_SHAPE('',$,#28);
#22=SHAPE_REPRESENTATION('',(#23),#12);
#23=AXIS2_PLACEMENT_3D('placement',#24,#25,#26);
#24=CARTESIAN_POINT('',(0.,0.,0.));
#25=DIRECTION('axis',(0.,0.,1.));
#26=DIRECTION('refdir',(1.,0.,0.));

#27=DESIGN_CONTEXT('part definition',#32,'design');
#28=PRODUCT_DEFINITION('Part1','Part1',#29,#27);
#29=PRODUCT_DEFINITION_FORMATION('',$,#34);
#30=PRODUCT_RELATED_PRODUCT_CATEGORY('Part1','Part1',(#34));
#31=APPLICATION_PROTOCOL_DEFINITION('international standard',
'ap203_configuration_controlled_3d_design_of_mechanical_parts_and_asse
mblies_mim_lf',2004,#32);
#32=APPLICATION_CONTEXT(
'Configuration Controlled 3D Design of Mechanical Parts and Assemblies');
#33=MECHANICAL_CONTEXT('part definition',#32,'mechanical');
#34=PRODUCT('Part1','Part1',$,(#33));
#35=PRESENTATION_STYLE_ASSIGNMENT((#36));
#36=SURFACE_STYLE_USAGE(.BOTH.,#37);
#37=SURFACE_SIDE_STYLE('',(#38));
#38=SURFACE_STYLE_FILL_AREA(#39);
#39=FILL_AREA_STYLE('',(#40));
```

```
#40=FILL_AREA_STYLE_COLOUR('',#41);
#41=COLOUR_RGB('',0.749019607843137,0.749019607843137,0.749019607843137);
```

B. Implementierung des Glättungsalgorithmus zur Berührpunktermittlung

Der Algorithmus ist in Abbildung 16 dargestellt und die Implementierung wird hier beispielhaft gezeigt. Es liegt der Abstand zwischen korrespondierenden Punkte auf dem Haupt- und Nebenläufer in dist[][4] vor. Es werden die Indizes und die Zahnnummer zusätzlich zum Abstand gespeichert. Außerdem liegt die aus dem maximalen Punktabstand berechnete Fenstergröße vor.
Zeile 1 - 3 initialisiert die Variablen disttmp, die das Verhalten des Punktverlaufs beschreiben. disttmp 1 und 3 sind die momentanen und in der vorherigen Schleife vorhandenen Abstandswerte. Diese geben allerdings nicht die Abstandswerte der Liste wieder, sondern beschreiben die Fensterposition. Das heißt, wenn der momentane Abstand im Fenster liegt, kommt es zu keiner Änderung der Größen disttmp1 und 3. disttmp2 beschreibt, ob in der zuvor durchgeführten Schleife das Fenster zu höheren (>0) oder zu kleineren Werten verschoben wurde (<=0).
In Zeile 3 wird die Schleife gestartet und so oft wiederholt wie Werte in dist[] gespeichert sind. Das sind die Anzahl an Punkten auf der Läuferkontur zwischen jenen Berührpunkten, die auf dem Kopfkreisradius liegen. Im ersten Schritt wird in Zeile 5 der Distanzwert gespeichert um die zu überprüfenden Grenzwerte in Zeile 6-10 festzulegen. Falls der momentane Abstand größer als die Fensterobergrenze ist wird das Fenster um den Differenzbetrag nach oben verschoben, bzw. wenn die Untergrenze unterschritten wird um den Differenzbetrag nach unten.
Die Unterscheidung in Zeile 11 überprüft ob die Liste einen Abwärtstrend aufweist. Berührpunkte gibt es nur, wenn der berechnete Abstand auf einem Minimum liegt. Also müssen die Werte in der Liste zuerst kleiner werden, bevor ein Minimalwert vorliegen kann. Die nächste Zeilen 12 und 14 überprüfen ob das Fenster zu höheren Werten verschoben wurde. Falls dies der Fall ist, wird mit Zeile 15-30 die Liste in die entgegengesetzte Richtung durchgegangen, bis der erste Wert die Fenstergrenze übersteigt.
Der Berührpunkt wird nun als jener Punkt definiert, der in der Mitte der beiden Schnittpunkte mit dem Fenster liegt. Dabei wird nur der Index der Liste gemittelt und nicht der physikalische Koordinatenmittelwert gebildet. Dies ist nur ein kleiner Nachteil, da in diesem Bereich die Kontur der beiden Läufer annähernd parallel sind.

Mit der Zeile 32 und den Zeilen 35 - 38 wird die Verlaufsrichtung für den nächsten Schleifendurchlauf aktualisiert.

```
1  disttmp1 = dist[0][4];
2  disttmp2 = -1;
3  disttmp3 = 0;
4  for (i=1; i<=n; i++){
5    disttmp3=disttmp1;
6    if (dist[i][4]>disttmp1+dist[i][5]/2){
7      disttmp1=dist[i][4]-dist[i][5]/2;
8    } else if (dist[i][4]<disttmp1-dist[i][5]/2){
9      disttmp1=dist[i][4]+dist[i][5]/2;
10   }
11   if (disttmp2<0){
12     if (disttmp1<=disttmp3){
13       disttmp2=-1;
14     } else if((disttmp1>disttmp3)){
15       for (m = i-1; m>=inde; m--){
16         if (m<0){
17           l=m+inde;
18         } else {
19           l=m;
20         }
21         if (dist[l][4]>disttmp1+dist[i][5]/2){
22           itmp1=(i+l)/2;
23           eptemp[k][j][ 1] = dist[itmp1][0];
24           eptemp[k][j][ 2] = dist[itmp1][1];
25           eptemp[k][j][ 7] = dist[itmp1][2];
26           eptemp[k][j][ 8] = dist[itmp1][3];
27           eptemp[k][j][13] = j;
28           j++;
29           break;
30         }
31       }
32       disttmp2 = +1;
33     }
34   } else {
35     if (disttmp1>=disttmp3){
36       disttmp2 = +1;
37     } else if (disttmp1<disttmp3){
38       disttmp2 = -1;
39     }
40   }
41 }
```

Literatur

[1] *B-spline Basis Functions: Definition.* http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/B-spline/bspline-basis.html

[2] Boor, C. de: *A Practical Guide to Splines.* Springer-Verlag, 1978. – ISBN 3–540–90356–9

[3] Farin, G. : *NURB Curves and Surfaces, from Projective Geometry to practical use.* 2nd edition. AK Peters, Ltd., 1999. – ISBN 1–56881–084–9

[4] Gallier, J. : *Curves and Surfaces in Geometric Modeling: Theory and Algorithms.* 1999

[5] Grafinger, M. : *Die computerunterstützte Entwicklung der Flankenprofile für Sonderverzahnungen von Schraubenkompressoren.* Shaker, 2010. – ISBN 978–3–8322–9215–7

[6] Piegl, W. Les; T. Les; Tiller: *The NURBS Book.* 2nd edition. Springer, 1997. – ISBN 3–540–61545–8