Numerisch stabile Berechnungen auf Kreisbögen

Masterarbeit

an der Fakultät für Informatik und Mathematik
der Universität Passau

Prüfer:

Prof. Dr. Tomas Sauer

Zweitprüferin:

Prof. Dr. Brigitte Forster-Heinlein

Günther Reitberger

Sommersemester 2016

Zusammenfassung

Die folgende Ausarbeitung beschäftigt sich mit numerischen Schwächen der Kreisdarstellung durch Mittelpunkt und Radius, wenn der Radius groß wird. Es ist ein Ziel dieser Arbeit, ein Gefühl zu vermitteln, warum diese numerischen Fehler auftreten und ab welcher Größe des Radius. Es werden Ansätze besprochen, die diesen Effekt umgehen wollen. In diesem Sinne wird versucht Kreisprobleme mittels alternativer Berechnungsmethoden und Kreisdarstellungen zu formulieren. Die Effektivität der vorgestellten Ansätze wird in praktischen Tests gegenübergestellt.

Abstract

The following work addresses the numerical weaknesses of the representation of circles using center and radius when there is a large radius. This paper wants to give an impression why these numerical issues appear and in what way the size of the radius is involved. Approaches are discussed that aim to circumvent those effects.
In this spirit, an attempt is made to reformulate certain calculations on circles by an alternative way of calculation, or, by a different way of representing a circle.
The use and effects of the new approaches are compared using practical testing.

Inhaltsverzeichnis

1. Einleitung

1.1. Einordnung der Masterarbeit

Diese Arbeit entstand im Rahmen des Projekts „Methoden für den Einsatz von Kreisbogensplines zur Darstellung hochgenauer digitaler Karten“, das von der Deutschen Forschungsgemeinschaft[1] gefördert wurde, ist jedoch in seiner Aussage auf keinen Fall darauf beschränkt. Das Projekt basiert auf der Promotion von Georg Maier, der in seinem SMAP(Smooth Minimum Arc Path)-Algorithmus einen Kreisbogenspline verwendet, um eine gegebene Menge an Punkten durch eine Kurve zu approximieren, die neben einem glatten Anschluss der einzelnen Segmente eine maximale Abweichung erfüllen und dabei in seiner Segmentzahl minimal sein muss. Ein Kreisbogenspline ist eine Kurve, die aus Kreisbögen und Geradenstücken zusammengesetzt ist und an den Übergangsstellen glatt sein muss. Eine genauere Definition und eine ausführlichere Beschreibung der Arbeit findet sich in [Mai10].
Im Rahmen des beschriebenen DFG-Projekts war der Autor dieser Masterarbeit damit beschäftigt, den SMAP basierend auf der Grundlage von Georg Maier weiterzuentwickeln. Die intuitive Darstellung eines Kreises ist gegeben durch den Mittelpunkt M und den Radius r. Damit verbunden ist die Darstellung der Punkte auf dem Kreis durch die Parametrisierung $P(\phi) = M + r * (\cos(\phi), \sin(\phi)),\ \phi \in [0, 2\pi]$, zumindest im zweidimensionalen Raum, der für diese Arbeit die Grundlage bildet. Diese Charakterisierung hat zwei schöne Eigenschaften. Zum einen ist die Parametrisierung eine Bogenlängenparametrisierung, d.h. dass der Punkt $P(2 * \phi)$ bezüglich der Bogenlänge doppelt so weit von $P(0)$ entfernt ist, wie $P(\phi)$ für beliebige Winkel $0 < \phi < \pi$. Zum anderen ist es ausreichend, sich den Abstand eines Punktes zum Kreismittelpunkt zu errechnen, um mit einem Vergleich mit dem Radius entscheiden zu können, ob sich der Punkt innerhalb des Kreises befindet, oder nicht.
In der softwaretechnischen Realität ist aufgefallen, dass Kreisbogensplines basierend auf dieser Kreisdarstellung auch numerische Schwächen haben. Ist der Radius des zu einem Kreisbogen gehörigen Kreises sehr groß, so ist der Kreisbogen lokal gesehen einer Strecke sehr nah. Will man nun den Abstand eines Punktes zu diesem Segment berechnen, so gibt es zwei Möglichkeiten. Einerseits könnte man einfach annehmen, dass es sich um eine Strecke handelt, den Punkt auf die Strecke projizieren und den Abstand dazu ausrechnen. Andererseits kann man bei der Kreisdarstellung bleiben und über die Methode „Abstand des Punktes zum Mittelpunkt minus dem Radius“ zum Ziel kommen. Nun haben leider beide Varianten ihre Nachteile. Da der Kreisbogen eben noch nicht ganz einer Strecke entspricht, macht man bei der Annahme, es sei eine

[1] Fördernummer MA 5834/1-1, URL: http://gepris.dfg.de/gepris/projekt/225923277 (besucht am 16.6.2016)

Strecke, offensichtlich einen Fehler. Für den Nachteil der zweiten Variante muss kurz etwas ausgeholt und der numerische Hintergrund beleuchtet werden.

1.2. Numerische Aspekte

Bekanntlich ist die Genauigkeit auf Rechnern beschränkt. Man hat nur eine begrenzte Anzahl an Stellen zur Verfügung, um Zahlen darzustellen. Deshalb treten bei den meisten Rechenoperationen Fehler auf. Die Darstellung von Zahlen ist eine sogenannte normierte Gleitkommadarstellung. Dies bedeutet, dass die Zahlen stets in der Form $.d_1 * ... * d_m \ * \ 2^e$ behandelt werden. So würde etwa die Zahl 7 (binär: 111) als $.111 * 2^3$ dargestellt werden. Dabei werden die Ziffern $d_1 ... d_m$ als Mantisse bezeichnet und e als der Exponent. Auf aktueller Hardware in double-Genauigkeit ist eine Mantissenlänge von $52 + 1$ und Exponentenlänge von 11 üblich[2].
Dadurch ergibt sich, dass die kleinste Zahl, die zu 1 addiert werden kann und das Ergebnis sich von 1 unterscheidet, die $2^{-53} \approx 1.11 * 10^{-16}$ ist. Kleinere Zahlen fallen wegen der begrenzten Mantissenlänge unter den Tisch. Außerdem ist der Zahlenbereich beschränkt und es können natürlich nicht alle reellen Zahlen dargestellt werden. Bei Rechnungen, die eine höhere Genauigkeit als 16 Stellen benötigen würden, tritt deswegen ein relativer Fehler von $\frac{1}{2}2^{1-m}$ allein auf Grund der Rundung auf die letzte Stelle auf. Wenn zwei ähnlich große Zahlen voneinander abgezogen werden, entstehen beim Subtraktionsprozess führende Nullen. Da die Zahlen auf dem Rechner stets normiert gespeichert werden, wird das berechnete Ergebnis so lange geshiftet, bis an der ersten Stelle eine von 0 verschiedene Ziffer steht. Dabei wird mit Nullen von hinten aufgefüllt. Dies sind natürlich reine Phantasieziffern, die die Rechnung mit einem teils sehr großen relativen Fehler behaften. Dieser Effekt wird Auslöschung genannt. Mehr dazu kann in [Sauer13] nachgelesen werden.

1.3. Numerik im SMAP-Projekt

Kehren wir zu dem Kreisbogenbeispiel mit großem Radius zurück. Ist der Radius r im Bereich von 10^{10} oder größer, so ist der Abstand von dem gegebenen Punkt P zum Mittelpunkt M ebenfalls mindestens 10^{10}, wenn man annimmt, dass der Punkt außerhalb des Kreises liegt. Bei der Subtraktion von $\overline{MP}$ mit r sind demnach bereits mindestens 10 Stellen der Mantisse durch den Radius und den großen Abstand von P vom Mittelpunkt besetzt und das Ergebnis kann, da wie oben beschrieben nur 16 Stellen zur Verfügung stehen, nur etwa auf 6 Stellen nach dem Komma exakt bestimmt werden. In der SMAP-Anwendung ist eine maximale Genauigkeit von 6 Stellen, ohne

[2] Das +1 bei der Mantissenlänge rührt von einem sogenannten 'guard digit', das bei Gleitkommaoperationen für zusätzliche Genauigkeit sorgt und damit den Fehler der Operation deutlich verringert.

Rechenfehler zu berücksichtigen, nicht tolerierbar. Eine ausreichende Genauigkeit kann basierend auf diesem Algorithmus folglich nicht eingehalten werden.

1.4. Anforderung an diese Masterarbeit

Die Anforderung an diese Arbeit besteht darin, die Ursachen für das oben geschilderte Problem herauszuarbeiten und sich ein Verfahren zu überlegen, das die numerische Herausforderung bewältigt und trotz großer Radien ein ausreichend exaktes Ergebnis liefert. Ist das Verfahren erfolgreich und kann die Projektion eines Punktes auf einen Kreis mit großem Radius, bzw. die Abstandsberechnung besser als mit dem bisherigen Ansatz bewältigt werden, so ist eine Ausweitung auf andere Methoden des Interfaces für Kreisberechnungen in dem bestehenden Code gewünscht. Berechnungen wie der Schnittpunkt von zwei Kreisbögen sind bei großen Radien ähnlichen Herausforderungen ausgesetzt wie die Abstandsberechnung.
In der folgenden Ausarbeitung wird zunächst das Defizit der aktuellen Implementierung spezifiziert und an Beispielen konkretisiert. Daraufhin werden zwei Verfahren eingeführt, die auf Grund ihrer Konstruktion eine höhere Genauigkeit in den angedeuteten Problemfällen versprechen. Mit einer Reihe von Tests werden die verschiedenen Implementierungen verglichen. Am Ende wird eine Übersicht über den erstellten Code gegeben und die Integration in das SMAP-Projekt vorgestellt, bevor ein Resümee die Arbeit abschließt.

2. Aktuelle Implementierung und Problemspezifikation

Bevor die Problemspezifikation konkretisiert wird und sich die Arbeit an Hand der Lösungsversuche entfaltet, soll an dieser Stelle kurz eingeführt werden, wie zwei zentrale Operationen auf Kreisen bzw. Kreisbögen in der aktuellen Implementierung nach der Mittelpunkt-Radius Darstellung eines Kreises gehandhabt werden. Die beiden Operationen sind die Projektion eines Punktes auf einen Kreis und die Berechnung der Schnittpunkte zweier Kreise. Diese beiden werden sich wie ein roter Faden durch die Arbeit ziehen und als Vergleichsobjekte der verschiedenen Verfahren dienen.

2.1. Punktprojektion

Eigentlich sollte man noch vor der Punktprojektion beginnen. Zum Erstellen eines Kreisbogensplines müssen zunächst die Kreise erzeugt werden, die jeweils unter anderem durch drei Punkte aufgespannt werden. In Abbildung 1 ist der Kreis c durch die Punkte A, B und C definiert. Geometrisch lässt sich der Mittelpunkt M durch den Schnitt zweier Mittelsenkrechten der drei gegebenen Punkte bestimmen. Etwas genauer wird dies im Appendix unter A.1 beschrieben. Im weiteren Verlauf der Arbeit wird es die Situation geben, dass zwei Punkte und die Tangente an einem der beiden Punkte gegeben ist und daraus ein Kreis erstellt werden muss. Die Vorgehensweise dazu ist in A.2 aufgeführt. Für einen allgemeineren Hintergrund zum Bestimmen von Kreisbögen auf numerisch stabile Art und Weise wird auf [Rei14] verwiesen. Jetzt ist es ausreichend, den Kreis c durch die Punkte A, B und C bestimmt zu haben.

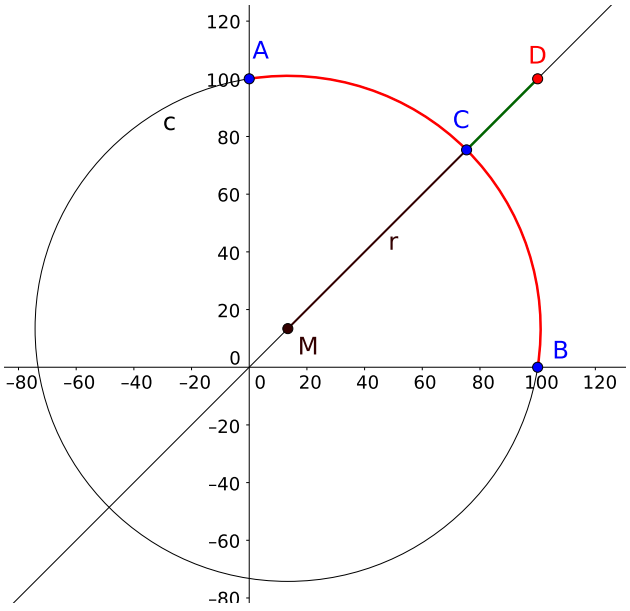

Abb. 1: *Testfall zur Problemspezifikation - moderater Radius*

Wie in der Abbildung zu sehen ist, soll der Punkt D auf den Kreis c mit Mittelpunkt M und Radius r projiziert werden. Dies geschieht durch folgende Formel:

$$P_D = M + \frac{D - M}{\|D - M\|_2} * r$$

Der Richtungsvektor von M zu D wird normiert und mit r multipliziert. Damit erhält man einen Vektor, dessen Länge r beträgt und von M aus gesehen Richtung D zeigt. Zu M addiert ergibt dies den gewünschten Projektionspunkt.

2.2. Schnittpunktberechnung

Nun wollen wir uns mit der Berechnung der Schnittpunkte zweier Kreisbögen beschäftigen. Im regulären Fall sind dies maximal zwei. Fälle, in denen die beiden Kreisbögen identisch sind, oder zu demselben Kreis gehören, werden hier der Einfachheit halber nicht betrachtet. Hier wird die Schnittberechnung nur schematisch behandelt. Unter A.3 ist die Version weiter ausgeführt und näher an der Implementierung.

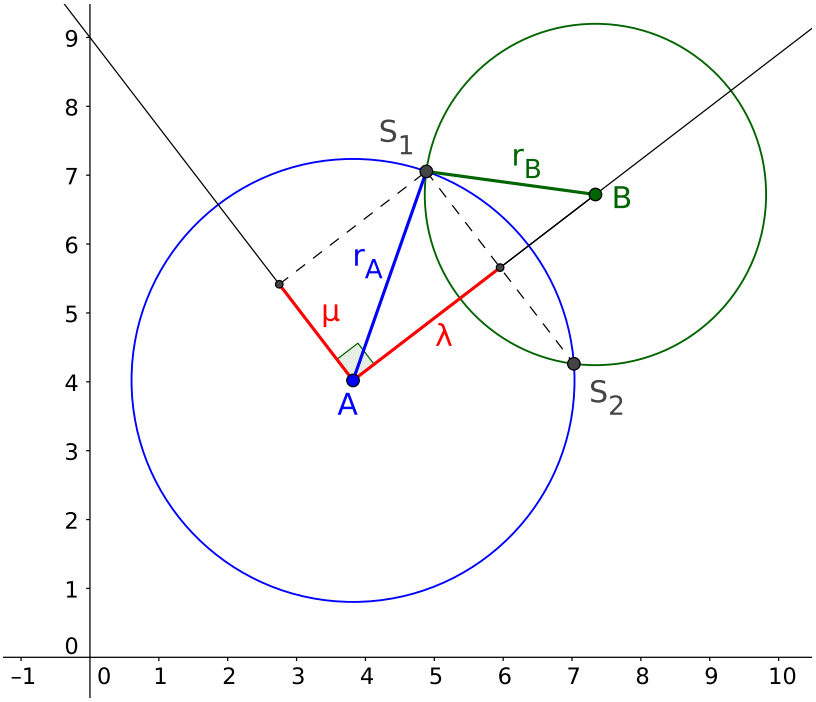

Abb. 2: *Schnittpunkte zweier Kreise*

Um, wie in Abbildung 2 zu sehen, die Schnittpunkte S_1 und S_2 von zwei Kreisen mit Mittelpunkten A und B und Radien r_A und r_B zu berechnen, wird gerne ein lokales Koordinatensystem durch eines der beiden Zentren angelegt. In der Abbildung ist das Koordinatensystem durch die beiden Halbgeraden mit Zentrum A dargestellt. Mit $d := \|A - B\|_2$ und dem lokalen Koordinatensystem vereinfachen sich die Kreisgleichungen

zu

$$\lambda^2 + \mu^2 = r_A^2$$
$$(d - \lambda)^2 + \mu^2 = r_B^2.$$

Da ein Schnittpunkt $S = (\lambda, \mu)$ auf beiden Kreisen liegen muss, ist ein Schnittpunkt die Lösung des Gleichungssystems. Nach einigen Umformungen erhält man

$$\begin{aligned} \lambda &= \frac{r_A^2 - r_B^2 + d^2}{2 * d} \\ \mu &= \sqrt{r_A^2 - \lambda^2}. \end{aligned} \tag{1}$$

Damit hat man jedoch nur eine Lösung im lokalen Koordinatensystem. Für das Standard-Koordinatensystem ist eine Transformation nötig:

$$S_1 = \begin{pmatrix} A_x + \lambda\frac{B_x - A_x}{d} - \mu\frac{B_y - A_y}{d} \\ A_y + \lambda\frac{B_y - A_y}{d} + \mu\frac{B_x - A_x}{d} \end{pmatrix}$$

$$S_2 = \begin{pmatrix} A_x + \lambda\frac{B_x - A_x}{d} + \mu\frac{B_y - A_y}{d} \\ A_y + \lambda\frac{B_y - A_y}{d} - \mu\frac{B_x - A_x}{d} \end{pmatrix}$$

2.3. Problemstellung

Um das in der Einleitung angedeutete numerische Problem bei großen Radien explizit darzustellen, soll der grün eingezeichnete Abstand des Punktes D vom Kreis c in Zeichnung 1 berechnet werden. Der Kreis c wird dabei durch die drei Punkte A, B und C definiert, wobei A die Koordinaten $(0, 100)$ und B $(100, 0)$ hat. C bewegt sich auf der Winkelhalbierenden der Koordinatenachsen im 1. Quadranten. Diese Konstruktion hat zwei Vorteile. Zum einen kann der Abstand des Kreises zum Punkt D durch den Abstand der Punkte C und D sehr exakt ermittelt werden. Zum anderen kann durch die Bewegung des Punktes C auf der Winkelhalbierenden in Richtung E der Radius des Kreises c erhöht werden. Dies kann in Zeichnung 3 gesehen werden. E ist mit den Koordinaten $(50, 50)$ der Mittelpunkt der Sehne $[A, B]$. Im Laufe des Tests wird C immer näher an E herangeführt und beobachtet, wie exakt die Abstandsberechnung von D bezüglich c bleibt.
Beim Testen wurden die Punkte A, B und D festgehalten und C hatte initial die Koordinaten $(50.9, 50.9)$. Bei verschiedenen Testdurchläufen wurde die Anzahl der Nullen nach dem Komma bei C variiert, von $(50.9, 50.9)$ zu $(50.09, 50.09)$ usw. Damit das

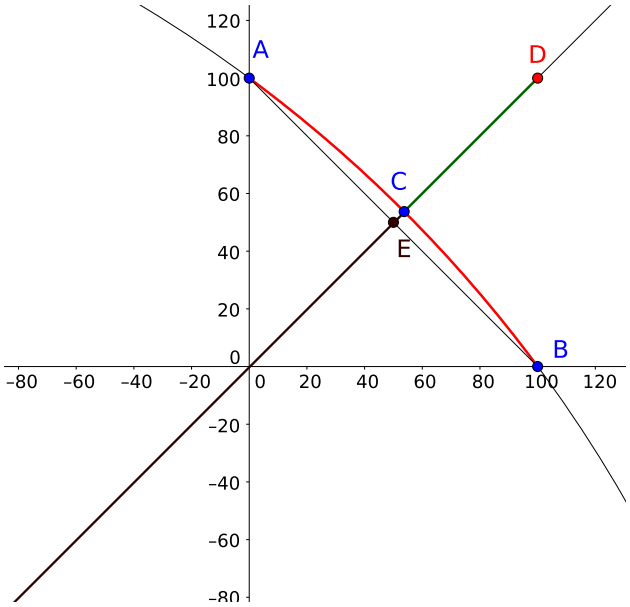

Abb. 3: *Testfall zur Problemspezifikation - großer Radius*

Testszenario in etwas allgemeinerer Lage ist, wurde das komplette Setting um einen konstanten Vektor verschoben. Dieser betrug etwa $(51.212..., 51.212...)$. Es wurden viele Nachkommastellen gewählt, damit die Koordinaten der Punkte in der Mantisse mehr von 0 verschiedene Einträge haben und numerische Effekte besser sichtbar sind. Hat eine Zahl nur 0-Einträge nach dem Komma, so spielt Auslöschung keine Rolle. Hat die Zahl von 0 verschiedene Nachkommastellen, so sind für die exakte Darstellung auch die Stellen nach dem Komma wichtig.
In den folgenden Tests werden drei Ansätze zur Abstandsberechnung verglichen. Die erste Methode berechnet den Abstand von D zum Mittelpunkt M und subtrahiert den Radius r. Die Zweite projiziert zunächst D auf c wie oben beschrieben. Der Abstand von D zu diesem Punkt ist der zweite Vergleichswert. Die letzte Methode zieht eine Gerade durch die Punkte A und B ein und berechnet den Abstand von D zu der Geraden. Damit die Werte besser verglichen werden können, wurde von allen Ergebnissen, der erwartete Abstand $\overline{CD}$ abgezogen.

c	50.9	$50 + 9 * 10^{-5}$	$50 + 9 * 10^{-8}$	$50 + 9 * 10^{-10}$	$50 + 9 * 10^{-13}$
Radius	$2 * 10^4$	$2 * 10^8$	$2 * 10^{11}$	$2 * 10^{13}$	$2 * 10^{16}$
Direkt	$5.7 * 10^{-14}$	$1.5 * 10^{-9}$	$1.1 * 10^{-7}$	$1.5 * 10^{-5}$	$4.0 * 10^{-2}$
Projektion	$1.6 * 10^{-13}$	$4.4 * 10^{-9}$	$9.1 * 10^{-7}$	$8.0 * 10^{-5}$	$5.3 * 10^{-2}$
Gerade	1.3	$1.2 * 10^{-4}$	$1.3 * 10^{-7}$	$1.2 * 10^{-9}$	$1.3 * 10^{-12}$

Tab. 1: *Fehler bei der Punktprojektion für verschiedene Methoden*

In der Tabelle 1 sind nach rechts die einzelnen Tests abgetragen. Wie oben beschrieben unterscheiden sie sich an der Anzahl der Nullen nach dem Komma bei C. In den Zeilen

ist zunächst der Radius des Kreises abgetragen, anschließend die verschiedenen Berechnungsmethoden in obiger Reihenfolge. Man sieht zum einen, dass die Ergebnisse, der ersten und zweiten Methode sehr nah beieinander liegen. Mit größer werdendem Radius nimmt die Genauigkeit der ersten (und zweiten) Methode ab, wohingegen die dritte Methode immer genauer wird. Dies ist nicht verwunderlich, da der Kreis sich mit wachsendem Radius immer mehr der Gerade annähert und so der Fehler der Approximation geringer wird. Man sieht zudem, dass die Summe der Beträge des Radius und der ersten Methode stets etwa 10^{18} ergeben, was auf die Mantissenlänge (mit dem Zusatzbit) zurückzuführen ist. Man sieht, dass die begrenzte Anzahl an Stellen dafür verantwortlich ist, dass die erste Methode immer ungenauer wird. Bei $c = 50 + 9 * 10^{-8}$ haben alle drei Methoden etwa den gleichen Fehler. Geht man nach der Taktik vor, dass man stets die bessere der Methoden als Lösung wählt, bzw. wenn der Radius 10^{11} übersteigt, man auf die Geradenapproximation wechselt, so macht man in dem Bereich mindestens einen Fehler von 10^{-7}.
In der aktuellen Implementierung von Kreisberechnungen im SMAP-Projekt ist dieser Fehler unumgänglich und führt zu Fehlern, wenn eine höhere Genauigkeit eingehalten werden soll. Exakte Ergebnisse sollen auch in etwas komplizierteren Algorithmen, wie zum Beispiel Schnittberechnungen erreicht werden können. Nun erzeugt jedoch schon diese einfache Punktprojektion, bzw. Abstandsberechnung einen kritischen Fehler. Deshalb besteht die Notwendigkeit, sich eine Methode zu überlegen und zu implementieren, die mit Kreisen mit großen Radien besser umgehen kann, als die vorgestellte Berechnung. Dies ist die zentrale Herausforderung an diese Masterarbeit.
Im Folgenden werden zwei Ansätze vorgestellt. Der erste Ansatz behält die Mittelpunkt-Radius Darstellung bei und versucht über eine andere Berechnungsmethode diesen numerischen Effekt zu umgehen. Der Zweite beruht auf einer Kreisdarstellung, die nicht auf Mittelpunkt und Radius basiert, sondern lokal einen Kreisbogen aufspannt. Damit soll vermieden werden, dass der unverhältnismäßig große Radius in die Rechnung miteinfließt, obwohl sich das Problem in einem sehr kleinen Abschnitt des Koordinatensystems abspielt.

3. Standard-Kreisdarstellung mit numerischem Augenmerk

Um die in der Problemstellung vorgestellte numerische Herausforderung zu bewältigen, wird zunächst die Kreisdarstellung beibehalten und versucht den Einfluss des Radius harmlos zu gestalten.
In dem Artikel [IGW09] wird ebenfalls eine sehr genaue und unter großen Radien stabile Punktprojektion benötigt. In deren Einleitung wird auf die Möglichkeit der Verwendung eines Newton-Verfahrens basierend auf einer NURBS-Darstellung hingewiesen. Dies entspricht den Ausführungen im anschließenden Kapitel. Die Autoren haben sich jedoch gegen einen solchen Ansatz entschieden, da sie die numerische Zuverlässigkeit hinsichtlich des Vermeidens von Nulldivision, negativer Wurzeln und Stellenauslöschung bei Subtraktion großer Zahlen, sowie deterministischer Antwortzeiten nicht ausreichend erfüllt sahen. Deshalb haben sie sich für einen direkten Ansatz nach der traditionellen Kreisdarstellung entschieden, wobei die Lösungsformel für die Punktprojektion umformuliert wird. Diese interessante Berechnungsweise der Projektion wird im Folgenden vorgestellt. Der Vergleich der Verfahren bezüglich der numerischen Zuverlässigkeit wird im Kapitel 5 mittels Tests vollzogen.

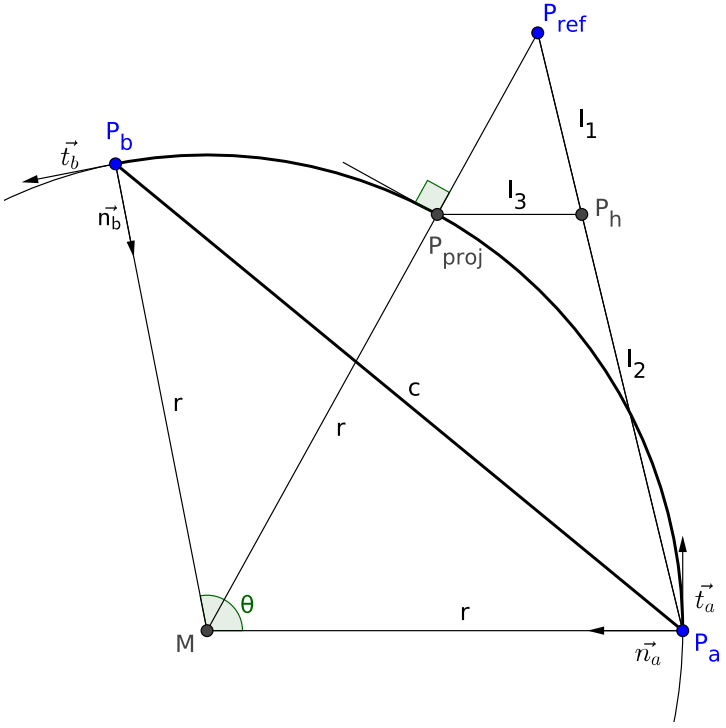

Abb. 4: *Punktprojektion mittels Strahlensatz*

Wie in Abbildung 4 zu sehen ist, soll ein Punkt $\mathbf{P}_{ref}$ auf einen Kreisbogen, der durch die Punkte $\mathbf{P}_a$ und $\mathbf{P}_b$ beschränkt und zu dem Kreis mit Mittelpunkt $\mathbf{M}$ und Radius

r gehört, projiziert werden. Die Standard-Berechnungsmethode dafür lautet:

$$\mathbf{P}_{poj} = \mathbf{M} + \frac{\mathbf{P}_{ref} - \mathbf{M}}{\|\mathbf{P}_{ref} - \mathbf{M}\|_2} * r$$

Die Grundidee des Ansatzes in [IGW09] versucht statt dem Radius r die gerichtete Krümmung κ mit $|\kappa| = \frac{1}{r}$, $\kappa > 0$ für Links- und $\kappa < 0$ für Rechtskurven in die Formeln zu integrieren. Dazu wird der Hilfspunkt $\mathbf{P}_h$ eingeführt, wie es in Abbildung 4 dargestellt ist. Definiert ist dieser durch

$$\mathbf{P}_{proj} = \mathbf{P}_a + (\mathbf{P}_h - \mathbf{P}_a) + (\mathbf{P}_{proj} - \mathbf{P}_h) = \mathbf{P}_a + \frac{l_2}{l_1 + l_2}(\mathbf{P}_{ref} - \mathbf{P}_a) + l_3 \operatorname{sign}(\kappa)\vec{n_a}. \quad (2)$$

Damit ist $\mathbf{P}_{proj}\mathbf{P}_h$ parallel zu $\mathbf{M}\mathbf{P}_a$ und es kann der erste Strahlensatz angewendet werden:

$$\begin{aligned} \gamma := \frac{l_2}{l_1 + l_2} &= \frac{r}{\|\mathbf{P}_{ref} - \mathbf{M}\|_2} = \frac{1}{|\kappa|\|(\mathbf{P}_{ref} - \mathbf{P}_a) + (\mathbf{P}_a - \mathbf{M})\|_2} \\ &= \frac{1}{\||\kappa|(\mathbf{P}_{ref} - \mathbf{P}_a) - |\kappa| \operatorname{sign}(\kappa) r\vec{n_a}\|_2} = \frac{1}{\||\kappa|(\mathbf{P}_{ref} - \mathbf{P}_a) - \operatorname{sign}(\kappa)\vec{n_a}\|_2} \\ &= \frac{1}{\|\kappa(\mathbf{P}_{ref} - \mathbf{P}_a) - \vec{n_a}\|_2} \end{aligned} \quad (3)$$

Dadurch treten weder unendlich große Werte noch eine Division durch 0 auf, außer es ist $\mathbf{P}_{ref} \approx \mathbf{M}$, was in der Praxis nicht sehr relevant ist oder abgeprüft werden kann. Mithilfe des zweiten Strahlensatzes erhält man

$$\frac{l_3}{r} = \frac{l_1}{l_1 + l_2} = 1 - \frac{l_2}{l_1 + l_2} = 1 - \gamma = \frac{\gamma(\frac{1}{\gamma} - 1)(\frac{1}{\gamma} + 1)}{(\frac{1}{\gamma} + 1)} = \frac{\gamma^2}{1 + \gamma}\left(\frac{1}{\gamma^2} - 1\right). \quad (4)$$

Multipliziert man in (4) beide Seite mit $r * \operatorname{sign}(\kappa)$,

$$l_3 \operatorname{sign}(\kappa) = \frac{\gamma^2}{1 + \gamma}\left(\frac{1}{\gamma^2} - 1\right) r * \operatorname{sign}(\kappa),$$

und setzt anschließend $\frac{1}{\gamma^2} = \|\kappa(\mathbf{P}_{ref} - \mathbf{P}_a) - \vec{n_a}\|_2^2$ aus (3) ein, so erhält man:

$$\begin{aligned} l_3 \operatorname{sign}(\kappa) &= \frac{\gamma^2}{1 + \gamma}\left(\|\kappa(\mathbf{P}_{ref} - \mathbf{P}_a) - \vec{n_a}\|_2^2 - 1\right) r * \operatorname{sign}(\kappa) \\ &= \frac{\gamma^2}{1 + \gamma}\left(\kappa\|\mathbf{P}_{ref} - \mathbf{P}_a\|_2^2 - 2\vec{n_a}^T(\mathbf{P}_{ref} - \mathbf{P}_a)\right). \end{aligned}$$

Hier soll ebenfalls darauf hingewiesen werden, dass weder unendlich große Werte noch eine Nulldivision auftritt. Die Krümmung κ kann mit Hilfe des Sehnenlängensatz durch

den Zentriwinkel θ und der Sehnenlänge c berechnet werden. Die dazugehörigen Formeln lauten:

$$\theta = \arctan_2(t_{b_y}, t_{b_x}) - \arctan_2(t_{a_y}, t_{a_x})^3$$

$$\kappa = 2\,\mathrm{sign}(\theta)\frac{|\sin(\frac{\theta}{2})|}{\|\mathbf{P}_b - \mathbf{P}_a\|_2}$$

Damit können γ, $l_3\,\mathrm{sign}(\kappa)$ und κ bestimmt werden und die Projektion setzt sich aus

$$\mathbf{P}_{proj} = \mathbf{P}_a + \gamma(\mathbf{P}_{ref} - \mathbf{P}_a) + l_3\,\mathrm{sign}(\kappa)\vec{n_a}$$

zusammen, wobei weder die Berechnung des Mittelpunktes noch die des Kreisradius nötig ist.

Worauf in [IGW09] nicht eingegangen wird, ist, dass trotzdem numerische Probleme auftreten können. Betrachtet man etwa die Berechnung von γ, so wird im Nenner $\|\kappa(\mathbf{P}_{ref} - \mathbf{P}_a) - \vec{n_a}\|_2$ bestimmt. Das Verfahren ist darauf ausgerichtet, dass kein Wert unendlich groß wird, jedoch ist die relative Größe entscheidend. So kann besagte Berechnung numerisch kritisch werden, wenn der Radius immer größer wird (dadurch κ immer kleiner), $\mathbf{P}_{ref}$, $\mathbf{P}_a$ und $\vec{n_a}$ jedoch zugleich (fast) unverändert bleiben. Damit wird $\kappa(\mathbf{P}_{ref} - \mathbf{P}_a)$ im Vergleich zu $\vec{n_a}$ sehr klein, wodurch es zu Stellenauslöschungen bei der Subtraktion kommen kann. Dieser Effekt wird im Kapitel 5 mittels Tests praktisch untersucht.

Nun könnte man annehmen, dass der gewählte Algorithmus nicht gut genug ist und es trotz der traditionellen Kreisdarstellung ein Verfahren gibt, das das numerische Problem großer Radien umgehen kann. Dass dieser Ansatz es nicht schafft, alle Probleme zu umgehen, ist natürlich kein Beweis dafür, dass es überhaupt nicht möglich ist, jedoch wird verdeutlicht, wie leicht es ist, in ein solches zu geraten. Die Division oder die Multiplikation mit dem Radius oder der Krümmung ist stabil, subtrahiert oder addiert man einen Term, der annähern konstant ist, oder nicht in derselben Art vom Radius oder der Krümmung abhängt, so erhält man einen Problemfall. Eine Berechnungsmethode, die dies bei einem direkten Ansatz gänzlich umgeht ist dem Autor nicht bekannt.

Ein weiterer Nachteil dieses Vorgehens besteht darin, dass es sehr aufwändig ist, eine numerisch stabile, vom Radius eingeschränkt abhängige Methode zur Berechnung der Punktprojektion zu finden. Dabei ist das Punktprojektionsproblem relativ einfach. Umso schwieriger wird es wohl sein, ein entsprechendes Verfahren für die Berechnung der Schnittpunkte zweier Kreisbögen zu finden. Darüber hinaus sind diese Algorithmen sehr speziell auf die Anforderung angepasst, weshalb sie zum Beispiel für die Berechnung ähnlicher Kreisprobleme nicht zu gebrauchen sind.

[3] $arctan_2$ bezeichnet die quadrantenbezogene Arcustangens-Funktion.

4. Kreisdarstellung durch eine rationale quadratische Bézierkurve

Um die Hindernisse des Ausgangsproblems umgehen zu können, war eine Idee, eine Kreisdarstellung, bzw. Kreisbogendarstellung zu finden und verwenden, die nicht auf Mittelpunkt und Radius basiert, da diese offensichtlich die kritische Größe im obigen Beispiel waren. Mit Hilfe von rationalen quadratischen Bézierkurven lassen sich Kreisbögen exakt darstellen. Dabei benötigt man weder den Radius noch den Mittelpunkt. Diese alternative Kreisdarstellung wird zunächst vorgestellt und es werden einige interessante Eigenschaften angesprochen, die theoretisch die Herausforderungen im Ausgangsproblem überwinden sollten. Dabei wird vor allem den Ausführungen des Buchs „The NURBS Book“ [PT97] gefolgt.

4.1. Eigenschaften rationaler quadratischer Bézierkurven

Polynome sind ausführlich studierte Funktionen zur Modellierung von Kurven und besitzen auch viele vorteilhafte Eigenschaften. Um jedoch Kreise, Ellipsen oder Hyperbeln exakt darstellen zu können, sind sie nicht ausreichend. Ein Beweis dafür, dass der Einheitskreis nicht durch Polynomfunktionen repräsentiert werden kann, ist in [PT97, 25 f.] aufgeführt. Es können jedoch alle konische Kurven[4] mit Hilfe von rationalen Funktionen repräsentiert werden. Diese sind wie folgt definiert:

$$x(u) = \frac{X(u)}{W(u)} \qquad y(u) = \frac{Y(u)}{W(u)} \tag{5}$$

Dabei sind $X(u)$, $Y(u)$ und $W(u)$ von dem Parameter u abhängige Polynome und $C(u) := (x(u), y(u))$ ist die Parametrisierung der in unserem Fall konischen Kurve.
Es würde nun ausreichen, passende Polynome X, Y und W zu finden und damit in obiger Form die Kreisbögen exakt darzustellen. Nun haben jedoch Polynome in der Standard-Potenz-Basis aus Sicht der geometrischen Modellierung einige Nachteile [PT97, S. 9]. Die Koeffizienten sagen aus geometrischer Sicht wenig über die Form der Kurve aus. Beim Gestalten von Kurven stellt man gerne Bedingungen an den Start und das Ende. Bei der Potenz-Basis ist das nur für den Start möglich. Darüber hinaus weist diese Basis auch numerische Schwächen auf. Deshalb wird im weiteren Verlauf auf eine andere Basis gewechselt.
Um in diesem Unterkapitel besser zwischen Skalar und mehrdimensionalen Punkten, Vektoren oder Funktionswerten unterscheiden zu können, werden diese fett gedruckt.

[4] Konische Kurven oder auch Kegelschnitte ergeben sich, wenn man die Oberfläche eines Doppelkegels mit einer Ebene schneidet. Die entstehenden Kurven sind Ellipsen, Parabeln oder Hyperbeln, bzw. Spezialfälle davon, wie z.B. ein Kreis.

4.1.1. Einführung von (nicht-rationalen) Bézierkurven

Eine spezielle parametrische Polynomkurve ist die Bézierkurve. Die Bézier-Methode schafft es, den eben genannten Nachteilen Abhilfe zu schaffen. Eine Bézierkurve n-ten Grades ist derart definiert:

$$\mathbf{B}(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i \qquad 0 \leq u \leq 1$$

Die Basis-Funktionen $\{B_{i,n}(u)\}$ sind die Bernstein-Polynome n-ten Grades:

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i}$$

Die geometrischen Koeffizienten oder auch Kontrollpunkte $\{\mathbf{P}_i\}$ sind in unserem Fall aus dem $\mathbb{R}^2$. Die Bernstein-Polynome haben folgende Eigenschaften, die im Verlauf benötigt werden:

(B1) Nichtnegativität: $B_{i,n}(u) \geq 0$ für alle i, n und $0 \leq u \leq 1$

(B2) Teilung der Eins: $\sum_{i=0}^{n} B_{i,n}(u) = 1$ für alle $0 \leq u \leq 1$

(B3) $B_{0,n}(0) = B_{n,n}(1) = 1$

(B4) rekursive Definition: $B_{i,n}(u) = (1-u)B_{i,n-1}(u) + uB_{i-1,n-1}(u)$ mit $B_{i,n}(u) := 0$ für $i < 0$ oder $i > n$

(B5) Ableitungen: $B'_{i,n}(u) = \dfrac{dB_{i,n}(u)}{du} = n(B_{i-1,n-1}(u) - B_{i,n-1}(u))$

Mit der Eigenschaft *(B5)* kann nun leicht die Ableitung der Bézierkurve berechnet werden:

$$\begin{aligned}\mathbf{B}'(u) &= \frac{d\left(\sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i\right)}{du} = \sum_{i=0}^{n} B'_{i,n}(u)\mathbf{P}_i \\ &= \sum_{i=0}^{n} n(B_{i-1,n-1}(u) - B_{i,n-1}(u))\mathbf{P}_i = n\sum_{i=0}^{n-1} B_{i,n-1}(u)(\mathbf{P}_{i+1} - \mathbf{P}_i)\end{aligned}$$

Für den Fall $n = 2$, d.h. einer quadratischen Bézierkurve, gilt:

$$\begin{aligned}\mathbf{B}(u) &= \sum_{i=0}^{2} B_{i,2}(u)\mathbf{P}_i = (1-u)^2\mathbf{P}_0 + 2u(1-u)\mathbf{P}_1 + u^2\mathbf{P}_2 \\ &= (1-u)((1-u)\mathbf{P}_0 + u\mathbf{P}_1) + u((1-u)\mathbf{P}_1 + u\mathbf{P}_2)\end{aligned} \tag{6}$$

Damit gilt für ein fixiertes $u = u_0$, $\mathbf{P}_{1,0} := (1-u_0)\mathbf{P}_0 + u_0\mathbf{P}_1$, $\mathbf{P}_{1,1} := (1-u_0)\mathbf{P}_1 + u_0\mathbf{P}_2$ und $\mathbf{P}_{2,0} = (1-u_0)\mathbf{P}_{1,0} + u_0\mathbf{P}_{1,1}$, dass $\mathbf{B}(u_0) = \mathbf{P}_{2,0}$ ist. Graphisch bedeutet dies,

dass man mit wiederholter linearer Interpolation nach dem Teilungsverhältnis u eine Punktauswertung vornehmen kann, wie es in Zeichnung 5 zu sehen ist. Die schwarze, dicke Kurve ist eine quadratische Bézierkurve mit den Kontrollpunkten $\mathbf{P}_0$, $\mathbf{P}_1$ und $\mathbf{P}_2$ an der Auswertungsstelle $u = 0.4$.

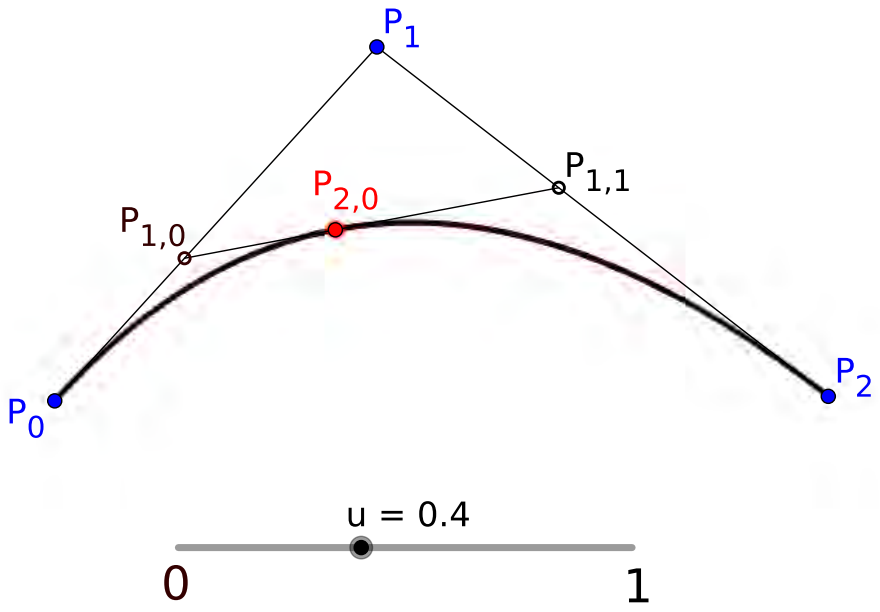

Abb. 5: *Auswertung einer Bézierkurve*

Die Hilfspunkte, die zur Auswertung verwendet wurden, können ebenfalls auf Grund der rekursiven Eigenschaft *(B4)* für ein fixes u_0 formalisiert werden:

$$\mathbf{P}_{i,k}(u_0) = (1-u_0)\mathbf{P}_{i,k-1}(u_0) + u_0\mathbf{P}_{i+1,k-1}(u_0) \quad \text{für} \begin{cases} i = 1, ..., n \\ k = 0, ..., n-i \end{cases} \tag{7}$$

Diese rekursive Formel ist auch besser bekannt als der *de Casteljau-Algorithmus* zur Auswertung einer Bézierkurve.

4.1.2. Einführung in rationale Bézierkurven

Nun haben wir einige wichtige Eigenschaften der Bézierpolynome beschrieben. Wir können sie auswerten, kennen die Ableitung und haben Wissen über die Parametrisierung. Nun wenden wir uns den rationalen Bézierkurven zu. Sie entsprechen der Form in (5), in die polynomiale Bézierkurven eingesetzt wurden. Eine rationale Bézierkurve n-ten Grades ist derart definiert:

$$\mathbf{R}(u) = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i \mathbf{P}_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i} = \sum_{i=0}^{n} R_{i,n}(u)\mathbf{P}_i \qquad 0 \le u \le 1$$

mit

$$R_{i,n}(u) = \frac{B_{i,n}(u)w_i}{\sum_{j=0}^{n} B_{j,n}(u)w_j} \tag{8}$$

Dabei sind wie oben $\mathbf{P}_i = (x_i, y_i)$ die Kontrollpunkte aus dem $\mathbb{R}^2$. Die Skalare w_i werden als „Gewichte“ bezeichnet und spielen im Verlauf eine entscheidende geometrische Rolle. Damit ist $W(u) = \sum_{i=0}^{n} B_{i,n}(u)w_i$ die gemeinsame Nenner-Funktion. Die Werte für die Gewichte werden im Folgenden genauer betrachtet, jedoch kann vorerst angenommen werden, dass $w_i > 0$ für alle i und damit $W(u) > 0$ für alle $u \in [0, 1]$ gilt. Die $R_{i,n}(u)$ sind die rationalen Basisfunktionen. Für diese werden ebenfalls zentrale Eigenschaften notiert, die auf die entsprechenden Eigenschaften der $B_{i,n}(u)$ und die Gleichung (8) zurückzuführen sind:

(R1) Nichtnegativität: $R_{i,n}(u) \geq 0$ für alle i, n und $0 \leq u \leq 1$

(R2) Teilung der Eins: $\sum_{i=0}^{n} R_{i,n}(u) = 1$ für alle $0 \leq u \leq 1$

(R3) $R_{0,n}(0) = R_{n,n}(1) = 1$

(R4) $B_{i,n}(u)$ ist ein Spezialfall von $R_{i,n}(u)$; für $w_i = 1$ für alle i gilt Gleichheit

Dies führt zu folgenden geometrischen Eigenschaften rationaler Bézierkurven:

(R5) Konvexe Hülle Eigenschaft: die Kurven sind in der konvexen Hülle ihrer definierenden Kontrollpunkte $\mathbf{P}_i$ enthalten

(R6) Transformationsinvarianz: Rotationen, Translationen und Skalierungen werden auf die Kurve angewandt, indem man sie auf die Kontrollpunkte anwendet

(R7) Endpunktinterpolation: $\mathbf{R}(0) = \mathbf{P}_0$ und $\mathbf{R}(1) = \mathbf{P}_n$

(R8) die k-te Ableitung an $u = 0$ ($u = 1$) ist abhängig von den ersten (letzten) $k + 1$ Kontrollpunkten und Gewichten; im Besonderen sind **R'**(0) und **R'**(1) parallel zu $\mathbf{P}_1 - \mathbf{P}_0$ bzw. $\mathbf{P}_n - \mathbf{P}_{n-1}$

(R9) polynomiale Bézierkurven sind ein Spezialfall von Rationalen

Der Zusammenhang in *(R9)* kann weiter ausgebaut werden, indem man *homogene Koordinaten* verwendet. Damit kann eine rationale Kurve im n-dimensionalen Raum als eine polynomiale Kurve im $n + 1$-dimensionalen Raum dargestellt werden. Beginnt man mit einem Punkt $\mathbf{P} = (x, y)$ aus dem $\mathbb{R}^2$, so wird $\mathbf{P}$ als $\mathbf{P}^w = (wx, wy, w) = (X, Y, W)$ im Dreidimensionalen geschrieben für $w \neq 0$. Umgekehrt kann $\mathbf{P}$ wieder aus $\mathbf{P}^w$ gewonnen werden, indem man X und Y durch W teilt, d.h. den Punkt $\mathbf{P}^w$ auf die

Hyperebene $W = 1$ projiziert:

$$\mathbf{P} = H\{\mathbf{P}^w\} = H\{(X, Y, W)\} = \begin{cases} \left(\frac{X}{W}, \frac{Y}{W}\right), & falls\ W \neq 0 \\ (X, Y), & falls\ W = 0 \end{cases}$$

Nun kann man für eine gegebene Menge von Kontrollpunkten $\{\mathbf{P}_i\}$ und Gewichten $\{w_i\}$ die gewichteten Kontrollpunkte $\mathbf{P}_i^w = (w_i x_i, w_i y_i, w_i)$ erzeugen und damit die polynomiale Bézierkurve im Dreidimensionalen aufstellen:

$$\mathbf{R}^w(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i^w \tag{9}$$

Diese projektive Darstellung kann in Algorithmen vorteilhaft sein. So kann man etwa die Punktauswertung, die in (6) und (7) für den polynomialen Fall angesprochen wurden, analog auf die rationalen Bézierkurven ausweiten, indem man die rationale Kurve zunächst in die Form, die in (9) angenommen wird, überführt und anschließend das Ergebnis durch die Abbildung H zurück in den $\mathbb{R}^2$ zurücktransformiert.

4.1.3. Ableitungen von rationalen Bézierkurven

Ableitungen sind eine sehr wichtige Eigenschaft von Kurven. Sie geben Informationen über das Verhalten und sind Bestandteil effizienter Algorithmen. Deshalb wollen wir uns in diesem Abschnitt der Berechnung der Ableitungen der eben aufgestellten rationalen Bézierkurven widmen. Ausführlicheres kann in [Flo92] nachgelesen werden.
Zunächst wird analog zu der Idee in (7) der de Casteljau-Algorithmus verwendet, um dazwischenliegende Gewichte[5] zu berechnen. Definiert man die Gewichte

$$w_{i,k}(u) := \sum_{j=0}^{k} B_{j,k}(u) w_{i+j},$$

so können diese mit dem de Casteljau-Algorithmus wie folgt berechnet werden:

$$w_{i,k} = (1 - u)w_{i,k-1} + u w_{i+1,k-1} \tag{10}$$

Die dazwischenliegenden Punkte $\mathbf{P}_{i,k}(u)$ werden nach

$$\mathbf{P}_{i,k}(u) = \frac{\sum_{j=0}^{k} B_{j,k}(u) w_{i+j} \mathbf{P}_{i+j}}{B_{j,k}(u) w_{i+j}} \tag{11}$$

[5] Im Englischen werden diese als „intermediate weights“ bezeichnet und beschreiben die zu den im de Casteljau-Algorithmus berechneten „dazwischenliegenden“ Punkten gehörigen Gewichte.

definiert. Dies ist äquivalent dazu, die Kontrollpunkte $\mathbf{P}_i$ zunächst mit homogenen Koordinaten darzustellen, den de Casteljau-Algorithmus aus (7) anwenden und abschließend in den $\mathbb{R}^n$ zurückprojizieren.
Aus (11) folgt, dass $\mathbf{P}_{i,0}(u) = \mathbf{P}_i$ und $\mathbf{P}_{i,n}(u) = \mathbf{P}_n$ gilt. Farin hat in [Far83] gezeigt, dass

$$w_{i,k}\mathbf{P}_{i,k} = (1-u)w_{i,k-1}\mathbf{P}_{i,k-1} + uw_{i+1,k-1}\mathbf{P}_{i+1,k-1} \tag{12}$$

gilt. Mit (10) und (12) kann $\mathbf{R}(u)$ rekursiv ausgewertet werden.
Da $\mathbf{R}(u)$ eine rationale Funktion ist, kann sie als $\mathbf{R}(u) = \dfrac{a(u)}{b(u)}$ geschrieben werden. Aus $b\mathbf{R} = a$ folgt für die Ableitung nach der Produktregel $b'\mathbf{R} + b\mathbf{R}' = a'$, bzw.

$$\mathbf{R}'(u) = \frac{a'(u) - b'(u)\mathbf{R}(u)}{b(u)}. \tag{13}$$

Nun werden die Funktionen $a(u)$ und $b(u)$ mit Hilfe der Eigenschaft *(B5)* abgeleitet. Damit ergibt sich:

$$a'(u) = \sum_{i=0}^{n} B'_{i,n}(u)w_i\mathbf{P}_i = n\sum_{i=0}^{n-1} B_{i,n-1}(u)(w_{i+1}\mathbf{P}_{i+1} - w_i\mathbf{P}_i) \tag{14}$$

und

$$b'(u) = n\sum_{i=0}^{n-1} B_{i,n-1}(u)(w_{i+1} - w_i) \tag{15}$$

Setzt man nun die beiden Gleichungen in (13) ein, hätte man bereits die Ableitung von $\mathbf{R}(u)$ mit

$$w_{0,n}\mathbf{R}' = n(w_{1,n-1}\mathbf{P}_{1,n-1} - w_{0,n-1}\mathbf{P}_{0,n-1}) - n(w_{1,n-1} - w_{0,n-1})\mathbf{P}_{0,n}$$

gefunden, jedoch ist diese geometrisch wenig aussagekräftig. Deshalb wendet man nun (10) und (12) an und erhält folgendes Endergebnis:

$$\mathbf{R}'(u) = n\frac{w_{0,n-1}(u)w_{1,n-1}(u)}{w^2_{0,n}(u)}(\mathbf{P}_{1,n-1}(u) - \mathbf{P}_{0,n-1}(u)) \tag{16}$$

Die Berechnung der ersten Ableitung funktionierte relativ problemlos und war bezüglich des Aufwandes überschaubar. Dies ändert sich jedoch schnell mit höheren Ableitungen. Die Herleitung der zweiten Ableitung wird hier deshalb nicht mehr ausgeführt,

sondern dafür nur noch auf die Literatur verwiesen. Die zweite Ableitung beträgt:

$$\begin{aligned}\mathbf{R}'' = n\frac{w_{2,n-2}}{w_{0,n}^3}(2nw_{0,n-1}^2 - (n-1)w_{0,n-2}w_{0,n} - 2w_{0,n-1}w_{0,n})(\mathbf{P}_{2,n-2} - \mathbf{P}_{1,n-2}) \\ - n\frac{w_{0,n-2}}{w_{0,n}^3}(2nw_{1,n-1}^2 - (n-1)w_{2,n-2}w_{0,n} - 2w_{1,n-1}w_{0,n})(\mathbf{P}_{1,n-2} - \mathbf{P}_{0,n-2})\end{aligned} \tag{17}$$

Mit Hilfe der ersten und zweiten Ableitung lässt sich die Krümmung der Kurve an der Stelle u berechnen:

$$\kappa(u) = \frac{\|\mathbf{R}'(u) \times \mathbf{R}''(u)\|_2}{\|\mathbf{R}'(u)\|_2^3}$$

4.1.4. Rationale quadratische Bézierkurven zur Repräsentation von Kegelschnitten

Wir werden zeigen, dass eine quadratische rationale Bézierkurve ausreicht, um konische Kurven exakt zu beschreiben. Diese sieht in allgemeiner Form wie folgt aus:

$$\begin{aligned}\mathbf{C}(u) &= \frac{B_{0,2}(u)w_0\mathbf{P}_0 + B_{1,2}(u)w_1\mathbf{P}_1 + B_{2,2}(u)w_2\mathbf{P}_2}{B_{0,2}w_0 + B_{1,2}w_1 + B_{2,2}w_2} \\ &= R_{0,2}(u)\mathbf{P}_0 + R_{1,2}(u)\mathbf{P}_1 + R_{2,2}(u)\mathbf{P}_2 \\ &= \frac{(1-u)^2w_0\mathbf{P}_0 + 2u(1-u)w_1\mathbf{P}_1 + u^2w_2\mathbf{P}_2}{(1-u)^2w_0 + 2u(1-u)w_1 + u^2w_2}\end{aligned} \tag{18}$$

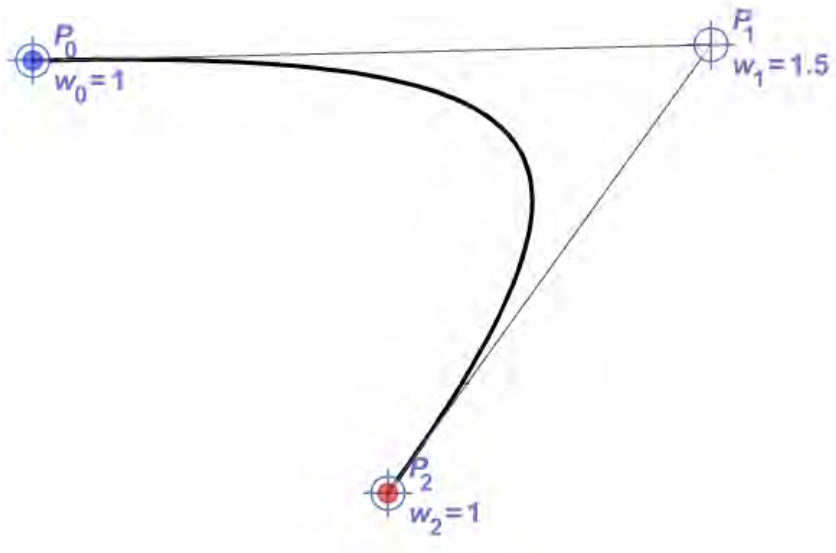

Abb. 6: *Beispiel einer rationalen Bézierkurve* [6]

[6] Die Zeichnung wurde erstellt mit der „nurbs demo v.1.0.1.en“ von Jan Foretnik (http://geometrie.foretnik.net/files/NURBS-en.swf).

Ein graphisches Beispiel einer rationalen quadratischen Bézierkurve ist in Abbildung 6 zu sehen. Die Gewichte sind ebenfalls eingezeichnet. Diese sind geometrisch sehr bedeutend. Beginnen wir damit, zu zeigen, dass (18) eine konische Kurve beschreibt. Dazu wird folgendes lokale Koordinatensystem aufgespannt:

$$\{\mathbf{P}_1, \mathbf{S}, \mathbf{T}\}, \qquad \text{mit} \quad \mathbf{S} = \mathbf{P}_0 - \mathbf{P}_1, \quad \mathbf{T} = \mathbf{P}_2 - \mathbf{P}_1$$

Auf Grund der konvexen Hülle Eigenschaft liegt $\mathbf{C}(u)$ für beliebiges u in dem Dreieck $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$ und kann somit geschrieben werden als

$$\mathbf{C}(u) = \mathbf{P}_1 + \alpha(u)\mathbf{S} + \beta(u)\mathbf{T} = \alpha(u)\mathbf{P}_0 + (1 - \alpha(u) - \beta(u))\mathbf{P}_1 + \beta(u)\mathbf{P}_2. \tag{19}$$

Vergleicht man dies mit Gleichung (18), so erkennt man, dass

$$\alpha(u) = R_{0,2}(u) \quad \text{und} \quad \beta(u) = R_{2,2}(u)$$

gilt. Verwendet man die aus der Definition der Bernstein-Polynome folgende Identität

$$B_{0,2}(u)B_{2,2}(u) = \left(\frac{B_{1,2}(u)}{2}\right)^2$$

und die Gleichung (19) und benennt die Funktion im Nenner von (18) mit $w(u)$, so erhält man:

$$\alpha(u)\beta(u) = R_{0,2}(u)R_{2,2}(u) = \frac{w_0 w_2 B_{0,2}(u)B_{2,2}(u)}{(w(u))^2} = w_0 w_2 \frac{(B_{1,2}(u))^2}{4(w(u))^2} = \frac{w_0 w_2}{4w_1^2}(R_{1,2}(u))^2$$

Setzt man

$$k = \frac{w_0 w_2}{w_1^2}, \tag{20}$$

und benutzt den Zusammenhang $R_{1,2}(u) = 1 - R_{0,2}(u) - R_{2,2}(u)$, der aus der Definition der $R_{i,2}$ folgt, so ergibt sich:

$$\alpha(u)\beta(u) = \frac{1}{4}k(1 - \alpha(u) - \beta(u))^2 \tag{21}$$

Diese Gleichung ist die implizite Gleichung eines Kegelschnitts in dem oben gewählten Koordinatensystem $\{\mathbf{P}_1, \mathbf{S}, \mathbf{T}\}$ [Lee87, S. 5 f.]. Damit ist gezeigt, dass die quadratische rationale Bézierkurve Kegelschnitte beschreibt.
Die Konstante k ist der sogenannte *Formfaktor*. Gleichung (21) besagt, dass k den Kegelschnitt bestimmt. Falls die drei Gewichte jedoch so geändert werden, dass sich k nicht ändert, dann ändert sich damit nur die Parametrisierung, jedoch nicht die Kurve. Die Art des Kegelschnitts kann dadurch bestimmt werden, dass man den Nenner $w(u)$

von Gleichung (18) untersucht. Die Funktion $w(u)$ kann wie folgt umgeformt werden:

$$w(u) = (1-u)^2 w_0 + 2u(1-u)w_1 + u^2 w_2 = (w_0 - 2w_1 + w_2)u^2 + 2(w_1 - w_0)u + w_0$$

Die Nullstellen dieser Gleichung sind

$$u_{1,2} = \frac{w_0 - w_1 \pm w_1\sqrt{1-k}}{w_0 - 2w_1 + w_2},$$

wobei k der Formfaktor ist. Aus Gleichung (20) geht hervor, dass zwei der drei Gewichte beliebig gewählt werden dürfen. Standardmäßig wird $w_0 = w_2 = 1$ gesetzt und diese Form als *normalisierte Parametrisierung* bezeichnet. Mit Gleichung (18) ergibt sich folglich:

$$\mathfrak{C}(u) = \frac{(1-u)^2\mathbf{P}_0 + 2u(1-u)w_1\mathbf{P}_1 + u^2\mathbf{P}_2}{(1-u)^2 + 2u(1-u)w_1 + u^2} \tag{22}$$

Mit ein paar Umformungen und etwas Wissen über Kegelschnitte kann man folgende Fallunterscheidung nach der Art des Kegelschnitts vornehmen:

- $w_1^2 < 1 \Rightarrow$ *Ellipse*
- $w_1^2 = 1 \Rightarrow$ *Parabel*
- $w_1^2 > 1 \Rightarrow$ *Hyperbel*

Das Beispiel in obiger Abbildung 6 ist in normalisierter Parametrisierung und hat für den Parameter w_1 den Wert 1.5, weshalb eine Hyperbel als Kurve dargestellt wird.
Für w_1 gibt es keine Einschränkung. Für $w_1 = 0$ erhält man die Strecke zwischen $\mathbf{P}_0$ und $\mathbf{P}_2$. Auch negative w_1 sind erlaubt. Wenn $w_1 < 0$ gilt, so wird der komplementäre Kreisbogen beschrieben, der in inverser Orientierung durchlaufen wird [Lee87, S. 7]. Damit ist jedoch die konvexe Hülle Eigenschaft nicht mehr erfüllt. Die ist aus geometrischer Sicht nicht immer wünschenswert, da man sich nicht darauf verlassen kann, dass sich die Kurve innerhalb des Dreiecks, das von den drei Kontrollpunkten aufgespannt wird, befindet. Für verschiedene Gewichte w_1 sind Beispiele von rationalen Bézierkurven in normalisierter Form in Abbildung 7 zu sehen. Wie erwähnt komplettiert die Kurve zu dem Gewicht -0.3 das Ellipsenstück, das von dem Gewicht 0.3 aufgespannt wird.
Was in Abbildung 7 ebenfalls eingezeichnet ist, sind die Scheitelpunkte der Kegelschnitte. Der Scheitelpunkt ist der Punkt, der am weitesten von der Geraden durch $\mathbf{P}_0$ und $\mathbf{P}_2$ entfernt ist. Man sieht, dass die Scheitelpunkte auf der Geraden durch $\mathbf{P}_1$ und dem Mittelpunkt $\mathbf{M}$ der Strecke $[\mathbf{P}_0\mathbf{P}_2]$ liegen. Das heißt, man kann die Punkte $\mathbf{S}$ durch ein s mit

$$\mathbf{S} = \mathbf{M} + s(\mathbf{P}_1 - \mathbf{M}) = (1-s)\mathbf{M} + s\mathbf{P}_1 \tag{23}$$

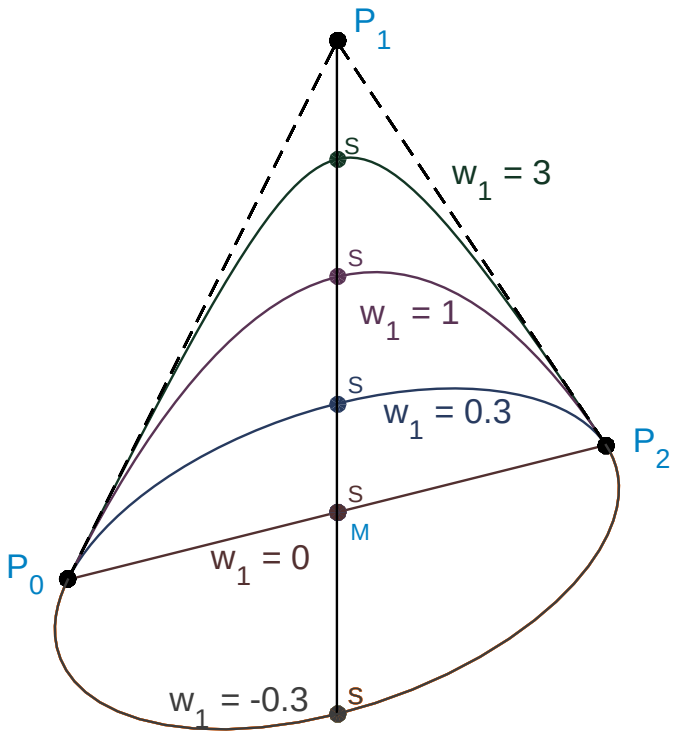

Abb. 7: *Rationale Bézierkurven in normalisierter Form zu verschiedenen w_1 Gewichten*

charakterisieren.

Die Ableitung der rationalen quadratischen Bézierkurve nach Gleichung (22) an der Stelle $u = \frac{1}{2}$ beträgt $\mathfrak{C}'(\frac{1}{2}) = \mathbf{P}_2 - \mathbf{P}_0$. Damit ist die Tangente an die Kurve bei $u = \frac{1}{2}$ parallel zu der aufspannenden Sehne $[\mathbf{P}_0\mathbf{P}_2]$. Da die Kurven Kegelschnitte repräsentieren, ist es nur an extremalen Stellen der Fall, dass die Ableitung parallel zur Sehne ist. Deshalb ist $\mathfrak{C}(\frac{1}{2})$ der Punkt auf der Kurve, der am weitesten von $\mathbf{P}_0\mathbf{P}_2$ entfernt ist. Da dieser Punkt für $w_1 \neq 0$ eindeutig ist, entspricht $\mathfrak{C}(\frac{1}{2})$ dem Scheitelpunkt $\mathbf{S}$. Damit gilt:

$$\mathbf{S} = \mathfrak{C}(\frac{1}{2}) = \frac{\frac{1}{4}\mathbf{P}_0 + \frac{1}{2}w_1\mathbf{P}_1 + \frac{1}{4}\mathbf{P}_2}{\frac{1}{4} + \frac{1}{2}w_1 + \frac{1}{4}} = \frac{\frac{1}{2}\mathbf{P}_0 + \frac{1}{2}\mathbf{P}_2}{1 + w_1} + \frac{w_1\mathbf{P}_1}{1 + w_1} = \frac{1}{1 + w_1}\mathbf{M} + \frac{w_1}{1 + w_1}\mathbf{P}_1$$

Aus dieser Gleichung und mit (23) folgt somit

$$s = \frac{w_1}{1 + w_1}, \quad \text{bzw.} \quad w_1 = \frac{s}{1 - s}. \tag{24}$$

Der Parameter s ist ein guter Design-Faktor. Mit ihm kann man den Scheitelpunkt von $\mathbf{M}$ nach $\mathbf{P}_1$ wandern lassen und somit die Fülle der Kurve bestimmen. Für $s = 0$ erhält man eine Strecke, $0 < s < \frac{1}{2}$ eine Ellipse, $s = \frac{1}{2}$ eine Parabel und $\frac{1}{2} < s < 1$ eine Hyperbel.

4.1.5. Kreis(bogen)darstellung durch rationale quadratische Bézierkurven

Da ein Kreis ein Spezialfall einer Ellipse ist, können mit (22) auch Kreisbögen dargestellt werden. Dies war auch das Ziel dieser Einführung in die Bézierdarstellung. Aus Symmetriegründen muss dafür das Dreieck $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$ gleichschenklig sein mit $\mathbf{P}_1$ als Spitze. Dies lässt sich auch dadurch erklären, dass die Tangenten über der Sehne eines Kreises ein gleichschenkliges Dreieck aufspannen.

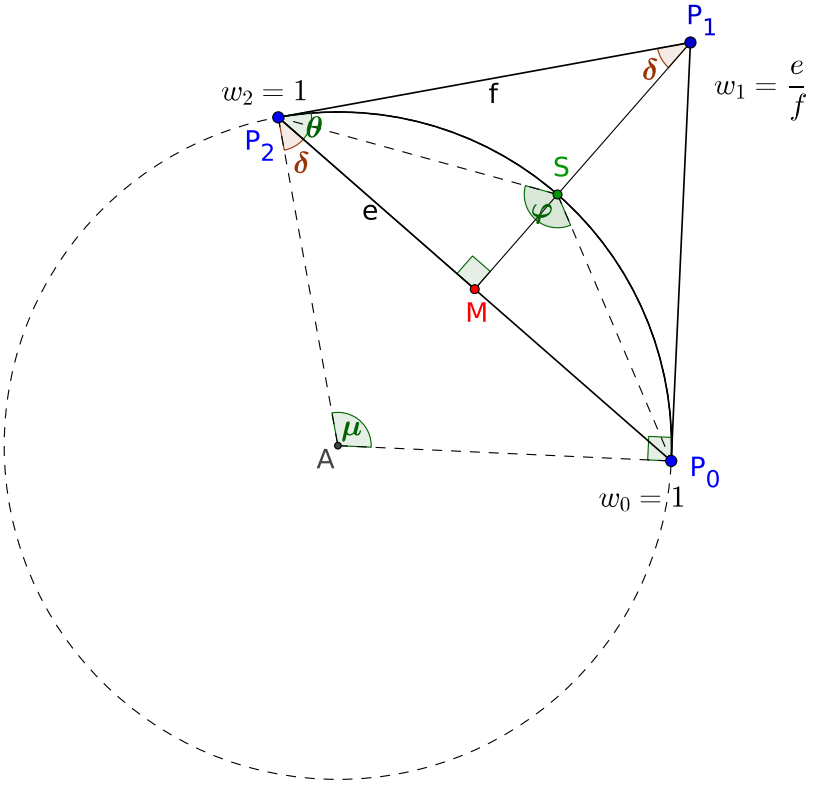

Abb. 8: *Kontrollpolygon und rationale Bézierkurve zur Darstellung eines Kreisbogens*

In Abbildung 8 ist das Kontrollpolygon $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$ über dem gewünschten Kreisbogen von $\mathbf{P}_0$ bis $\mathbf{P}_2$ zu sehen. Der Kreisbogen ist Teil des gestrichelten Kreises. Aus der Zeichnung ist zu sehen, dass mit dieser Methode nur Kreisbögen mit einem Öffnungswinkel[7], der weniger als 180° beträgt, dargestellt werden können. Will man den komplementären Kreisbogen darstellen, d.h. den Kreisbogen, der sich von $\mathbf{P}_0$ bis $\mathbf{P}_2$ im Uhrzeigersinn erstreckt, so muss man das Gewicht w_1 negieren. Die Gewichte sind in der Abbildung ebenfalls eingetragen. Die rationale Bézierkurve ist hier, wie im Folgenden, normiert. Mit Gleichung (24) folgt:

$$w_1 = \frac{s}{1-s} = \frac{\overline{\mathbf{MS}}}{\overline{\mathbf{SP}_1}}$$

Seien $\theta = \angle\mathbf{M}\mathbf{P}_2\mathbf{P}_1$, $\varphi = \angle\mathbf{P}_2\mathbf{S}\mathbf{P}_0$ und $\mu = \angle\mathbf{P}_0\mathbf{A}\mathbf{P}_2$. Auf Grund der Gleichschenk-

[7]Mit dem Öffnungswinkel eines Kreisbogens wird der Mittelpunktswinkel bezeichnet, den das zu dem Kreisbogen gehörige Kreissegment aufspannt. In Abbildung 8 wäre das der Winkel μ.

ligkeit des Kontrolldreiecks $\mathbf{P_0P_1P_2}$ und der tangentialen Eigenschaft der rationalen Bézierkurven am Start- und Endpunkt, sind $\angle\mathbf{P_1MP_2}$ und $\angle\mathbf{P_1P_0A}$, bzw. $\angle\mathbf{AP_2P_1}$ rechtwinklig. Aus den Grundlagen der Geometrie weiß man über die Winkel φ und μ, dass $2 * \varphi + \mu = 360°$, bzw.

$$\varphi + \frac{\mu}{2} = 180° \tag{25}$$

gilt. Mit den beiden Zusammenhängen $\theta + \angle\mathbf{P_2P_1M} = 90°$ und $\theta + \angle\mathbf{AP_2P_0} = 90°$, die wegen der Winkelsumme in dem rechtwinkligen Dreieck $\mathbf{MP_2P_1}$ bzw. des 90°-Winkels $\angle\mathbf{AP_2P_1}$ gelten, erhält man $\angle\mathbf{AP_2P_0} = \angle\mathbf{P_2P_1M} =: \delta$. Für das gleichschenklige Dreieck $\mathbf{AP_2P_0}$ gilt $\mu + 2 * \delta = 180°$. Setzt man darin den Zusammenhang $\theta + \delta = 90°$ ein, so ergibt sich

$$\theta = \frac{\mu}{2}. \tag{26}$$

Setzt man nun (26) in (25) ein, so ergibt sich

$$\varphi + \theta = 180°. \tag{27}$$

Da der Scheitelpunkt S in der Mitte des Kreisbogens liegt, ist das Dreieck $\mathbf{P_0P_2S}$ ebenfalls gleichschenklig und der Basiswinkel $\angle\mathbf{P_0P_2S}$ gleich $\frac{\theta}{2}$. Mit anderen Worten ist die Gerade $\mathbf{P_2S}$ die Winkelhalbierende von θ. Für w_1 bedeutet dies:

$$w_1 = \frac{\overline{\mathbf{MS}}}{\overline{\mathbf{SP}_1}} = \frac{\overline{\mathbf{MP}_2}}{\overline{\mathbf{P_1P_2}}} = \frac{e}{f} = \cos(\theta) \tag{28}$$

Damit kann nun das zu dem Kreisbogen und Kontrolldreieck gehörige w_1 bestimmt werden.

Wie bereits erwähnt, ist das Kontrolldreieck mit positivem Gewicht w_1 nur für Kreisbögen geeignet, die einen Öffnungswinkel von weniger als 180° haben. Um einen Vollkreis, oder einen Kreisbogen mit größerem Öffnungswinkel zu beschreiben, muss auf andere Weise vorgegangen werden. Eine ausführlichere Übersicht über die möglichen Ansätze kann in [PT97, S. 298 ff.] oder [PT89] nachgelesen werden.

Eine Methode wurde schon angedeutet. Um den durchgezogenen Kreisbogen in Abbildung 9 darstellen zu können, wird das Kontrolldreieck über dem komplementären, gestrichelt dargestellten Kreisbogen, der einen Öffnungswinkel von unter 180° hat, aufgespannt. Das Gewicht w_1 wird ebenfalls bzgl. dem gestrichelten Bogen berechnet und für den Durchgezogenen negiert.

Es ist bewiesen, dass es unmöglich ist, einen Vollkreis mit Hilfe eines non-uniform rational B-Splines (NURBS) mit positiven Gewichten darzustellen, ohne doppelte innere Knoten zu verwenden. Auf B-Splines wird hier nicht weiter eingegangen und für die B-Spline Ansätze mit doppelten Knoten wird ebenfalls auf die Literatur verwiesen.

Der Ansatz, der hier weiterverfolgt wird, ist ein sehr Intuitiver und Geometrischer. Wie

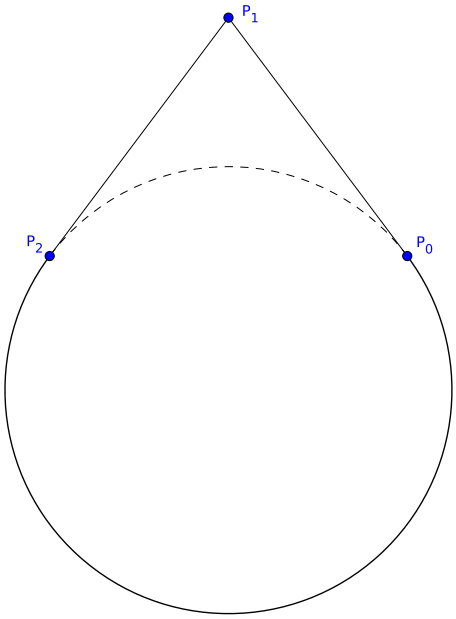

Abb. 9: *Darstellung von Kreisbögen mit Öffnungswinkel von mehr als* 180° *durch negatives* w_1

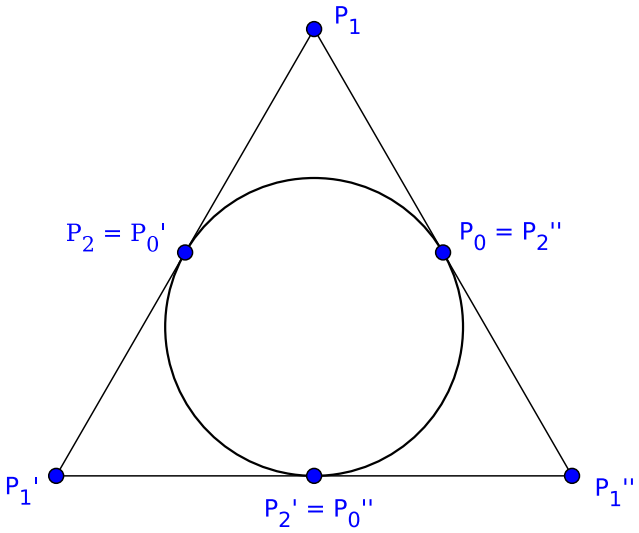

Abb. 10: *Ein Vollkreis zusammengesetzt aus drei kongruenten Kreisbögen*

man in Abbildung 10 sehen kann, wird der Vollkreis aus drei kongruenten Kreisbögen zusammengesetzt. Jeder Kreisbogen hat einen Öffnungswinkel von 120° und kann somit mit der Standardkonstruktion und positivem w_1 dargestellt werden. Die Kontrolldreiecke werden wie in der Skizze derart aneinandergefügt, dass der Endpunkt des vorherigen Kontrollpolygons der Startpunkt des folgenden ist. Dabei wird die Parametrisierung so gewählt, dass sich $\mathbf{P}_0$ an $u = 0$, $\mathbf{P}_2$ an $u = \frac{1}{3}$, $\mathbf{P}_2'$ an $u = \frac{2}{3}$ und $\mathbf{P}_2''$ an $u = 1$ befinden. Es kann gezeigt werden, dass $\mathfrak{C}(u)$ an $u = \frac{1}{3}$ und $u = \frac{2}{3}$ C^1-stetig ist. Graphisch ist das leicht einzusehen.

4.2. Implementierungsdetails zur Konstruktion eines Kreisbogens

Es ist kein zentraler Punkt dieser Arbeit, das Aufstellen einer rationalen Bézierkurve zur Beschreibung eines Kreisbogens ausführlich zu besprechen. Dennoch ist ein Blick auf die Implementierung wichtig, da sie bei den Testfällen eine Rolle spielt. Deshalb sollen hier die beiden grundlegenden Konstruktoren vorgestellt werden. Auf die weitere Implementierung wird auf das gesonderte Kapitel 6 verwiesen.

4.2.1. Konstruktion aus drei Punkten

Gegeben sind drei Punkte. Der erste, S, markiert den Startpunkt des Kreisbogens, der zweite, E, das Ende und B ist ein Punkt, der zwischen S und E auf dem Kreisbogen liegt. Zunächst soll der Öffnungswinkel berechnet werden:

$$\mu = 2 * \left(\arccos \left(- \frac{S - B}{\|S - B\|_2} \frac{E - B}{\|E - B\|_2} \right) \right)$$

An Hand von μ wird entschieden, in wie viele Teile der Kreisbogen zerlegt wird.
Wenn der Winkel kleiner als 180° ist, so wird der Bogen durch eine rationale Bézierkurve beschrieben. Dafür muss das Kontrolldreieck aufgespannt werden. Um die Berechnung numerisch stabiler zu gestalten, wird das Problem in eine einheitliche Form transformiert. Zunächst werden die drei Punkte so verschoben, dass S auf dem Ursprung liegt. Die neuen Punkte seien S', E' und B'. Anschließend werden E' und B' derart um S' gedreht, dass die Sehne $S'E'$ auf der x-Achse liegt und E' eine positive x-Koordinate besitzt. Damit ist die Kongruenzabbildung abgeschlossen und die Punkte sollen nun S'', E'' und B'' heißen.
In Abbildung 11 ist dies zu sehen. Für die weitere Berechnung ist anzumerken, dass alle Winkel über eine Sehne gleich sind. In diesem Fall heißt dies, dass der Winkel φ am Scheitelpunkt gleich $\angle S''B''E''$ gilt. Nun wird w_1 bestimmt:

$$w_1 = \frac{\|M - S''\|_2}{\|P_1 - S''\|_2} = \cos(\theta) = \cos(180° - \varphi) = - \frac{B''}{\|B''\|_2} \frac{E'' - B''}{\|E'' - B''\|_2}$$

Die ersten beiden Gleichheiten folgen aus (28). Die Dritte stammt von dem Zusammenhang in (27). Die letzte Umformung gilt durch eine Kosinusregel und der Darstellung des Kosinus eines Winkels durch das Skalarprodukt.
Ein Vorteil der Normierung ist, dass die Orientierung des Kreisbogen unmittelbar an der y-Koordinate von B'' abgelesen werden kann. Ist $B''.y > 0$, so ist der Kreisbogen im Uhrzeigersinn orientiert. Die Orientierung spielt in der weiteren Implementierung eine Rolle. Die Berechnung der Kontrollpunkte gestaltet sich ebenfalls einfacher in

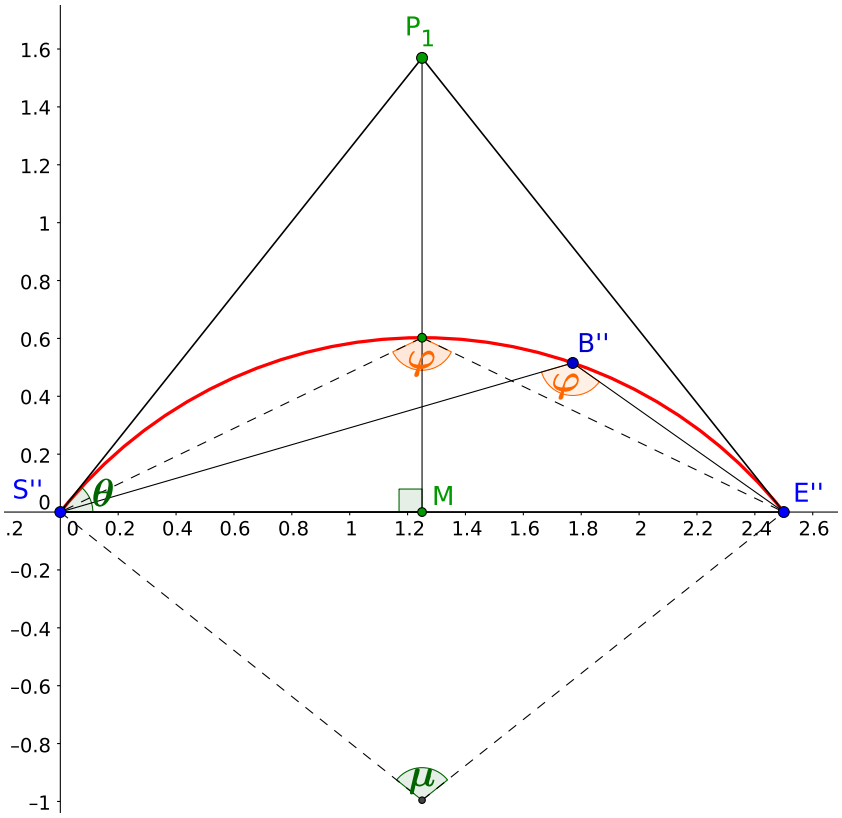

Abb. 11: *Normierte Lage eines Kreisbogens*

dieser Form. P_0 und P_2 sind trivial durch die Endpunkte gegeben. Für P_1 wird als Zwischenschritt M durch $M = E''/2$ berechnet. Nun kann P_1 etwa durch

$$P_1 = M + \tan(\arccos(w_1)) * M.x * \begin{pmatrix} 0 \\ \text{sign}(B''.v) \end{pmatrix}$$

bestimmt werden. Dabei formuliert der Tangens die absolute Steigung von $S''P_1$ und der Vektor am Ende das Vorzeichen. Damit sind alle Größen bestimmt, die für die Darstellung des Bogens als rationale Bézierkurve nötig sind.

Bei der Berechnung von P_1 kann es zu numerischen Ungenauigkeiten vor allem durch die Steigungsberechnung kommen. Deshalb wird mit Hilfe von nichtlinearer Optimierung versucht, P_1 zu verbessern. Da P_1 auf der Mittelsenkrechten über M liegt, reicht es, die y-Koordinate anzupassen. Da von einem geänderten P_1 w_1 abhängt, wird dieses anschließend durch $w_1 = \dfrac{M.x}{\|P_1\|_2}$ nach (28) angepasst. Dadurch erhält man ein konsistentes Kontrolldreieck mit w_1 und es kann mit der im folgenden Abschnitt aufgeführten Methode der Punktprojektion der Abstand zu B'' berechnet werden. Da dieser bei optimalem P_1 und w_1 Null sein sollte, dient dieser Abstand als Fehlerfunktion der Optimierung.

Ist die Iteration mit einer genügend kleinen Abweichung zum Ende gekommen, müssen

die Kontrollpunkte mit den jeweiligen inversen Transformationen zurück in die Ausgangsform gebracht werden, um zu dem ursprünglichen Problem zu passen. Damit ist die Konstruktion der Parameter für kleine Öffnungswinkel abgeschlossen.
Ist der Öffnungswinkel annähernd 180° oder mehr, bzw. beinahe 360°, so wird der Kreisbogen in zwei, bzw. drei äquidistante Stücke unterteilt und wie schon in 4.1.5 beschrieben für jeden Abschnitt das Kontrolldreieck aufgespannt. Für die äquidistante Unterteilung, wird die Situation erneut zunächst in normierte Lage wie eben beschrieben transformiert. Anschließend werden der Mittelpunkt C und der Radius r des zu den drei gegebenen Punkten gehörigen Vollkreises auf dieselbe Art und Weise wie bei der traditionellen Kreisdarstellung berechnet. Es wird ein lokales Koordinatensystem mit Ursprung C angelegt, bei dem die x-Achse Richtung $(0, 0)$ zeigt und die y-Achse orthogonal dazu ist. Die Orthogonale ist so orientiert, wie es der Kreisbogen ist, d.h. wenn der Kreisbogen gegen den Uhrzeigersinn orientiert ist, so wird für die y-Achse die orthogonale gegen den Uhrzeigersinn gewählt. Die beiden orthonormalen Richtungen sollen $\vec{a}$ und $\vec{b}$ heißen. Für den Fall, dass der Kreisbogen in d Kreisbögen unterteilt werden soll, werden die Punkte Q_i durch

$$Q_i = C + r * \cos(\frac{i}{2d} * \mu)\vec{a} + r * \sin(\frac{i}{2d} * \mu)\vec{b}$$

für $i \in \{0, 1, ..., 2d\}$ berechnet. Diese sind äquidistant auf dem Kreisbogen verteilt. Die oben beschriebene Konstruktion für drei Punkte mit einem Öffnungswinkel, der weniger als 180° beträgt, wird nun jeweils für die Punkte $\{Q_0, Q_1, Q_2\}$, $\{Q_2, Q_3, Q_4\}$, usw. aufgerufen. Hiermit ist die Konstruktion für den Fall großer Öffnungswinkel ebenso abgehandelt.

4.2.2. Konstruktion aus zwei Punkten und einer Tangente

Eine weitere Möglichkeit, die rationale Bézierkurve aufzuspannen, soll durch den Startpunkt S, den Endpunkt E und wahlweise die Tangente am Start, oder am Ende gegeben sein. Hier wird nur auf den Fall mit der Tangente am Start $\vec{t}$ eingegangen. Im Folgenden soll angenommen werden, dass der Tangentenvektor normiert ist. Der Algorithmus ist analog zu dem obigen mit drei Punkten. Der Öffnungswinkel lässt sich durch

$$\mu = 2 * \arccos\left(\vec{t} * \frac{E - S}{\|E - S\|_2}\right)$$

bestimmen. Danach wird unterschieden, wie viele Teilkreisbögen es geben soll. Zunächst wird die Berechnung für einen Öffnungswinkel unter 180° betrachtet. Nachdem in die

normierte Lage transformiert wurde, wird w_1 durch

$$w_1 = \vec{t''} * \frac{E''}{\|E''\|_2}$$

berechnet. Durch die Tangente ist die Steigung zur Ermittlung von P_1 bereits gegeben und die Darstellung vereinfacht sich zu

$$P_1 = \begin{pmatrix} M.x \\ \frac{\vec{t''}.y}{\vec{t''}.x} * M.x \end{pmatrix}.$$

Die Division stellt kein Problem dar, da $\vec{t''}.x$ nur für einen Öffnungswinkel von 180° Null ergibt. Damit sind die Parameter bestimmt. Die Nachoptimierung funktioniert ebenfalls wie oben, mit dem Unterschied, dass die Fehlerfunktion durch die Differenz der Tangente zur Ableitung am Start gebildet wird.
Für Öffnungswinkel, die annähernd 180° sind oder mehr, wird wie oben ein lokales Koordinatensystem angelegt und mit Hilfe von äquidistanten Punkten und dem Konstruktor für drei Punkte vorgegangen. Dafür muss der Mittelpunkt C berechnet werden, was durch

$$C = \begin{cases} M & , falls\ \vec{t''}.y \approx 0 \\ \begin{pmatrix} M.x \\ -\vec{t''}.x/\vec{t''}.y * M.x \end{pmatrix} & , sonst \end{cases}$$

geschieht. Damit lässt sich für einen Kreisbogen, der mit Hilfe von zwei Punkten und der Tangente am Start gegeben ist, die rationalen Bézierkurven formulieren, die ihn darstellen.

4.3. Algorithmische Lösung zweier Kreisprobleme

Nachdem die Theorie der rationalen quadratischen Bézierkurven dargelegt und einige wichtige Eigenschaften aufgezeigt wurden, wird nun auf die Punktprojektion und Schnittberechnung konkret eingegangen. Eine parametrische Darstellung einer Kurve ist nur dann wirklich nützlich, wenn man mit ihr vernünftig rechnen kann.

4.3.1. Projektion eines Punktes auf einen Kreisbogen

Die Aufgabe besteht darin, zu einem gegebenen Kreisbogen k und zu projizierendem Punkt P den Punkt auf dem Kreisbogen zu finden, der den geringsten Abstand zu P hat. Anders formuliert versucht man den Parameter $u^* \in [0, 1]$ zu finden, der das

Skalarprodukt

$$f(u^*) = \mathfrak{C}'(u^*) * (\mathfrak{C}(u^*) - P)$$

minimiert.

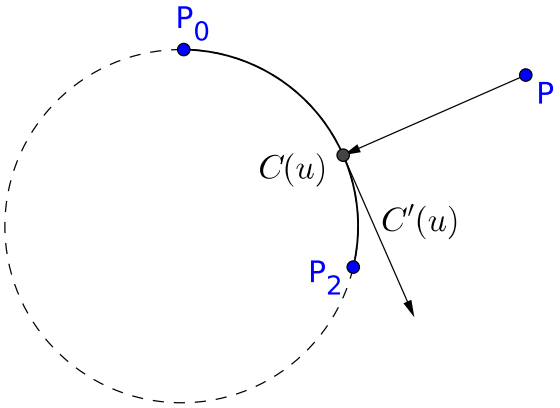

Abb. 12: *Projektion eines Punktes auf einen Kreisbogen.*

Veranschaulicht wird dies in Abbildung 12. Am gewünschten Punkt $\mathfrak{C}(u^*)$ steht der Richtungsvektor zum Punkt P senkrecht auf der Ableitung, d.h. das Skalarprodukt ist 0. Nun gilt es zwei Anmerkungen zu machen. Zum einen kann es sein, dass der Punkt, an dem das Skalarprodukt 0 ist, nicht auf dem Kreisbogen liegt. Dann ist einer der Endpunkte der nächste Punkt. Ein weiteres Problem ist in Abbildung 12 zu erahnen. Es gibt auf dem, zu dem Kreisbogen gehörigen Kreis zwei Punkte, an denen das Skalarprodukt 0 ergibt. Der eine ist der Punkt, der am nächsten zu dem gegebenen Punkt P gelegen ist, der andere, der am entferntest gelegene. Dies gilt es in dem Algorithmus zu beachten.
Als Algorithmus wurde Newton-Iteration gewählt, so wie es auch in [PT97, S. 230 ff.] angedeutet wird. Auf die theoretischen Hintergründe des Verfahrens wird hier nicht explizit eingegangen. Eine Ausführung kann man etwa in [Sauer13, S. 184 ff.] nachlesen. Das Newton-Verfahren ist eine nichtlineare Optimierungsmethode, die quadratisches Konvergenzverhalten an den Tag legt, sofern die Startlösung nahe genug am Ziel liegt. Das heißt insbesondere, dass man eine Startlösung finden muss. Dafür wird an äquidistanten Parameterwerten die Entfernung zu P berechnet und das u_0 als Startwert gewählt, sodass $\mathfrak{C}(u_0)$ von den gewählten Sampling-Werten den geringsten Abstand zu P hat. Dabei ist zu beachten, dass man genügend genau abtastet, sodass das Newton-Verfahren garantiert gegen die gewünschte Lösung konvergiert. Man kann sich überlegen, dass die Lösung eindeutig ist, wenn etwa die Abtastwerte nicht weiter als 90°, vom Mittelpunkt aus gesehen, voneinander entfernt sind. Jedoch kann eine höhere Abtastrate vorteilhaft sein, weil das Newton-Verfahren nur lokal, nahe bei u^* quadratisch konvergiert.

Hat man seinen Startwert u_0 bestimmt, so kann nun mit der Iteration begonnen werden. Dabei erhält man den nächsten Wert nach folgender Vorschrift:

$$u_{i+1} = u_i - \frac{f(u_i)}{f'(u_i)} = u_i - \frac{\mathfrak{C}'(u_i) * (\mathfrak{C}(u_i) - P)}{\mathfrak{C}''(u_i) * (\mathfrak{C}(u_i) - P) + \|\mathfrak{C}'(u_i)\|_2^2} \tag{29}$$

Mit den Formeln (16) und (17) für die erste und zweite Ableitung einer rationalen Bézierkurve kann jeder Newton-Schritt berechnet werden. Wenn, wie in diesem Fall, die erste und zweite Ableitung in relativ überschaubarer, geschlossener Form existieren, so bietet es sich an, das Newton-Verfahren zu verwenden. Durch die Eigenschaft der quadratischen Konvergenz reichen meist wenige Iterationen[8], um ein numerisch stabiles und exaktes Ergebnis zu erhalten.
Für jede Iteration benötigt man selbstverständlich ein Abbruchkriterium. Eine Bedingung stellt die Maximalzahl an Iterationen dar. Da, wie gerade beschrieben, das Verfahren nur sehr wenige Iterationen benötigt, kann diese Zahl sehr niedrig gewählt werden. Darüber hinaus wird in jeder Iteration abgeprüft, ob der Punkt P mit dem aktuellen Kandidaten für die Projektion $\mathfrak{C}(u_i)$ auf eine gewähltes numerisches Toleranzniveau ϵ übereinstimmt:

$$|\mathfrak{C}(u_i) - P| \leq \epsilon$$

Falls dies der Fall ist, ist die Projektion gefunden und die Iteration kann abgebrochen werden. In diesem Fall liegt der Punkt P auf dem Kreisbogen. Falls dies nicht der Fall ist, wird getestet, ob das normierte Skalarprodukt 0 ergibt:

$$\frac{\mathfrak{C}'(u^*) * (\mathfrak{C}(u^*) - P)}{|\mathfrak{C}'(u^*)||\mathfrak{C}(u^*) - P|} \leq \epsilon$$

Ist diese Bedingung erfüllt, ist mit $\mathfrak{C}(u_i)$ ebenfalls die Projektion gefunden worden und die Iteration kann beendet werden. Die beiden ϵ-Werte können natürlich auch verschieden gewählt werden. Wenn die beiden Kriterien nicht erfüllt sind, so kann ein neuer Wert u_{i+1} nach (29) berechnet werden. Dabei ist zu beachten, dass u_{i+1} innerhalb der definierten Grenzen bleibt. Für unsere Anwendung folgt daraus:

$$\begin{aligned} falls\ (u_{i+1} < 0): &\qquad u_{i+1} = 0 \\ falls\ (u_{i+1} > 1): &\qquad u_{i+1} = 1 \end{aligned}$$

Die letzte Abbruchbedingung testet, ob sich der Parameter zum vorhergehenden signifikant verändert hat:

$$|(u_{i+1} - u_i)\mathfrak{C}'(u_i)| \leq \epsilon$$

[8] Bei den Tests, die am Ende durchgeführt wurden, war eine maximale Iterationszahl von 5 ausreichend.

Ist keine signifikante Veränderung mehr eingetreten, so wird die Iteration abgebrochen und $\mathfrak{C}(u_{i+1})$ bzw. $\mathfrak{C}(u_i)$ stellt die Lösung dar. Andernfalls wird eine weitere Iteration durchlaufen.

4.3.2. Schnittpunkte zweier Kreisbögen

Für die Punktprojektion ist das Newton-Verfahren die Standard-Lösung, da sie einfach zu implementieren ist, numerisch stabil ist und eine gute Performanz zu erwarten ist. Für die Schnittpunktberechnung gibt es eine Reihe verschiedener Ansätze.
Für die Untersuchung von Vorgehensweisen und der Auswahl der Methode der Wahl wurde sich an [Sed14, S. 84 ff.] orientiert. Darin werden der Bézier-Subdivision-Algorithmus [LR80], die Intervall-Subdivision [KM83], Implizitierung [SP86] und Bézierclipping [SN90] kurz vorgestellt und anschließend bezüglich der Ausführungszeit verglichen.

4.3.2.1. Bézier-Subdivision

Bézier-Subdivision verwendet die konvexe Hülle-Eigenschaft und den de Casteljau-Algorithmus, um die Schnittpunkte zweier Kurven zu berechnen. Der Algorithmus vergleicht die konvexen Hüllen von zwei Kurven. Wenn sie nicht überlappen, schneiden sich die Kurven nicht. Falls sie überlappen, werden die Kurven halbiert und die beiden Hälften der einen Kurve werden auf Überlappung mit den beiden Hälften der anderen Kurven bzw. deren konvexen Hüllen überprüft. Da diese Vorgehensweise fortgeführt wird, werden in jedem Iterationsschritt Teile der Kurven ausgeschlossen, die keine Schnittpunkte enthalten können. Wenn ein Paar von Kurven genau genug unterteilt wurde, sodass die Kurve bis auf eine vorgegebene Toleranz durch eine Strecke approximiert werden kann, dann wird ein Schnittpunkt der Kurven durch den Schnittpunkt der Strecke bestimmt. Abbildung 13 illustriert das Vorgehen.

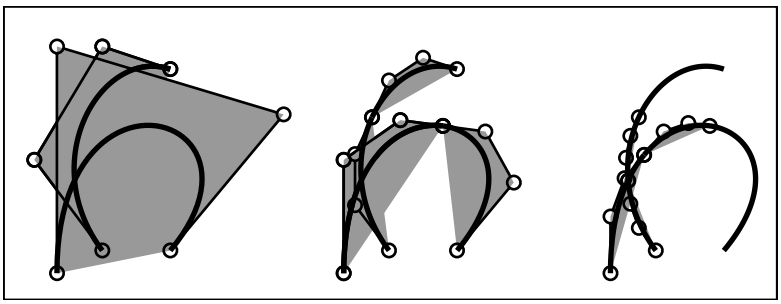

Abb. 13: *Drei Iterationen des Bézier-Subdivision Algorithmus*

4.3.2.2. Intervall-Subdivision

Die Intervall-Subdivision Methode ist der Bézier-Subdivision sehr ähnlich. Hierbei werden für jede Kurve zunächst die horizontale und vertikale Tangente bestimmt und in

„Intervalle“ unterteilt, die diese Tangenten, falls überhaupt, an den Endpunkten haben. Mit dieser Konstruktion erhält man die praktische Eigenschaft, dass innerhalb eines jeden solchen Intervalls ein Rechteck, dessen Diagonale durch zwei beliebige Punkte der Kurve bestimmt ist, die Kurve zwischen den beiden Punkten komplett einbeschreibt. Die Mächtigkeit dieser Methode liegt darin, dass die Subdivision nun hauptsächlich dadurch bestimmt werden kann, dass man die Kurve auswertet. Darüber hinaus ist die Bounding-Box trivial zu bestimmen. Die Bounding-Boxen sind in der Abbildung 14 zu sehen. Die Iteration läuft analog zur Bézier-Subdivision ab.

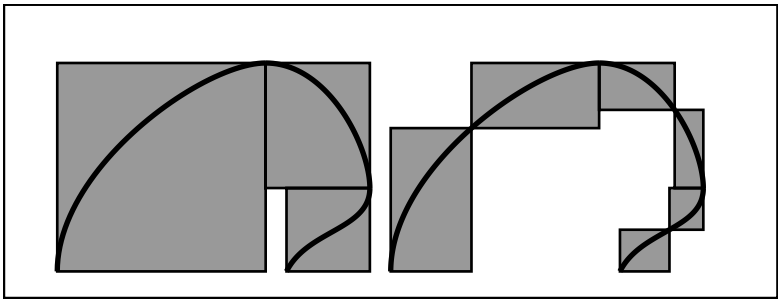

Abb. 14: *Bestimmung der Intervalle und erste Subdivisionsschritte*

4.3.2.3. Implizitierung

Es gibt prinzipiell zwei Arten, eine planare Kurve darzustellen: parametrisch und implizit. Bisher wurde etwa in (22) nur die parametrische Form einer rationalen Bézierkurve vorgestellt. Eine implizite Darstellung ist von der Form $F(x, y) = 0$. Alle Punkte, die die Gleichung erfüllen, liegen auf der Kurve. Den Vorgang, eine Kurve von ihrer parametrischen Form in die Implizite zu überführen, heißt Implizitierung. Dies ist für jede parametrische Form möglich. Dabei muss man jedoch etwas vorsichtig sein, da bei der Implizitierung die Einschränkung des Parameters, wie in unserem Fall $t \in [0, 1]$ verloren geht und die implizite Kurve die parametrische Kurve für $t \in \mathbb{R}$ darstellt. Das Problem, zwei Kurven in parametrischer Form zu schneiden, kann nun dadurch gelöst werden, indem man eine Kurve implizitiert und die andere Kurve in diese implizite Form einsetzt. Der Algorithmus in [SP86] geht so vor, dass mit Hilfe der Resultante und dem Aufstellen einer symbolischen Determinante die implizite Form erstellt wird. Hat man die zweite parametrische Kurve eingesetzt, so erhält man ein univariates Polynom. Die Nullstellen dieses Polynoms stellen die Parameter der Schnittpunkte dar.

4.3.2.4. Bézier-Clipping

Die letzte Methode, die hier kurz eingeführt wird, ist das Bézier-Clipping. Es kann als eine Art Intervall-Newton Methode angesehen werden, da es quadratische Konvergenz besitzt, jedoch robust alle Schnittpunkte findet.

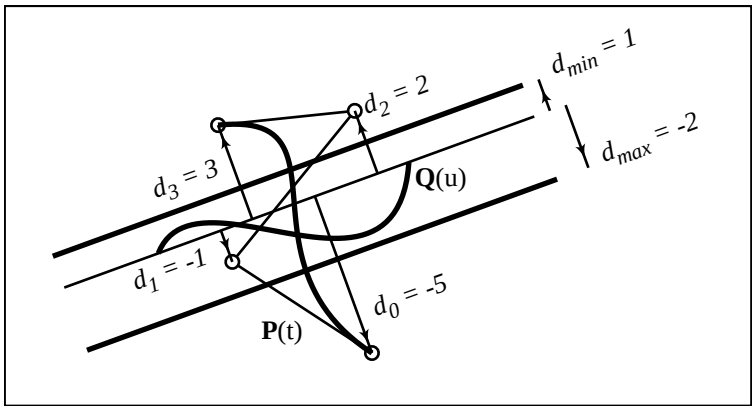

Abb. 15: *Schnittpunktberechnung durch Fat Lines*

In Abbildung 15 ist ein Beispiel einer sogenannten „Fat Line“ zu sehen, die im Bézier-Clipping-Algorithmus verwendet wird, um das mögliche Schnittintervall einzugrenzen. Die Bézierkurven in der Abbildung sind jedoch nicht-rational. Für den rationalen Fall ist die Idee der Fat Lines nicht so leicht umzusetzen und es muss auf eine etwas andere Methode eine solche Fat Line gefunden werden. Für weitere Informationen soll an dieser Stelle an die Literatur verwiesen werden.

4.3.2.5. Vergleich der Schnittmethoden

Die vorgestellten Methoden wurden in [Sed14, S. 84 ff.] bezüglich ihrer relativen Ausführungszeit verglichen. In Tabelle 2 sind die Ergebnisse zu sehen.

Grad	Clip	Impl	Int	Sub
3	2.5	1	10	15
3	1.8	1	5	6
5	1	1.7	3	5
10	1	na	2	4

Tab. 2: *Relative Ausführungszeit der Schnittmethoden abhängig vom Grad der Kurve*

Dabei steht *Clip* für Bézier-Clipping, *Impl* für Implizitierung, *Int* für Intervall-Subdivision und *Sub* für Bézier-Subdivision. *Grad* gibt an, von welchem Grad die Kurven in dem jeweiligen Beispiel waren. Es wurde double-Genauigkeit Verwendet und die Ergebnisse wurden auf 8 Stellen genau berechnet. Allgemein ist die Implizitierung nur für Kurven mit einem Grad von bis zu 5 zuverlässig. Für höheren Grad ist es möglich, dass der Polynomgrad degeneriert.
Für kleineren Grad (bis 4) ist jedoch die Implizitierung zuverlässig und ist Tabelle 2 nach die schnellste Methode. Allgemein werden auf Implizitierung basierende Methoden für kleinen Grad schneller als andere Schnittalgorithmen eingeschätzt [MD92]. Dies beinhaltet auch schnellere Subdivisionsmethoden wie Bézier-Clipping. Da in un-

serer Anwendung nur Kurven von kleinem Grad relevant sind, wurde die Implizitierung ausgewählt und wird im Folgenden genauer betrachtet.

4.3.2.6. Explizite Ausführung des Implizitierung-Ansatzes

Für die Implementierung wurde nicht die Standardmethode zur Implizitierung nach Sederberg [Sed14] gewählt, sondern eine Erweiterung davon. Es wird nicht die symbolische Determinante, sondern eine Matrix-Determinante aufgestellt. Die Parametrisierung wird anschließend in die Matrix-Formel eingesetzt. Aus der resultierenden Matrix wird eine numerische Matrix erzeugt, sodass sich die Schnittpunkte aus der Eigenwertzerlegung derselbigen ergeben [MD92]. Der Vorteil dieser Vorgehensweise besteht darin, dass Algorithmen für die Berechnung von Eigenwerten und Eigenvektoren numerisch stabil sind und es schnelle Implementierungen in Bibliotheken wie z.B. LAPACK gibt. Damit ist das Verfahren effizient, robust und numerisch stabil. Die für die Implementierung relevanten Schritte werden nun explizit ausgeführt. Für den kompletten Hintergrund wird auf die Literatur verwiesen.
Für die implizite Matrixdarstellung einer rationalen Bézierkurve wurde der Vorgehensweise in [Bus14] gefolgt. Darin wird beschrieben, wie allgemein rationale Bézierkurven von einer Parameterdarstellung in eine implizite Form überführt werden. Hier wird die Vorgehensweise konkret auf die normierte quadratische Bézierkurve nach (22) angewandt. Sei nun

$$\phi : t \in \mathbb{R} \to \frac{\sum_{i=0}^{2} B_{i,2}(t) w_i \mathbf{P}_i}{\sum_{i=0}^{2} B_{i,2}(t) w_i} =: \left(\frac{f_1(t)}{f_0(t)}, \frac{f_2(t)}{f_0(t)} \right)$$

die Parametrisierung des Kreisbogens, bzw. des zu dem Kreisbogen gehörigen Kreises. Nun wird ein nicht-negativer Grad ν gewählt und eine Matrix $\mathbb{M}_\nu(\phi)$ wie folgt erstellt. Betrachte ein 3-Tupel von Polynomen $(g_0(t), g_1(t), g_2(t))$, sodass

$$deg(g_i(t)) \leq \nu \quad \text{und} \quad \sum_{i=0}^{2} g_i(t) f_i(t) \equiv 0 \tag{30}$$

erfüllt ist. Dies ist ein Vektorraum. Will man eine Basis $L_1, ..., L_{r_\nu}$ aufstellen, so führt dies zu dem Problem, ein lineares Gleichungssystem zu lösen. Jedes L_j ist ein 3-Tupel von Polynomen $(g_0(t), g_1(t), g_2(t))$ und es kann durch

$$L_j(t, X, Y) := g_0(t) + X g_1(t) + Y g_2(t)$$

dargestellt werden. Darüber hinaus besteht die Freiheit, eine beliebige Basis von Polynomen des Grades $\leq \nu$ zu wählen, um die g_i zu beschreiben. Sei $\mathfrak{B}'_\nu = \{\psi_1(t), ..., \psi_{m_\nu}(t)\}$

eine solche Basis, so ergibt sich

$$L_j(t,X,Y) = \left(\sum_{i=1}^{m_\nu} \lambda_{0,i}^{(j)}\psi_i(t)\right) + \left(\sum_{i=1}^{m_\nu} \lambda_{1,i}^{(j)}\psi_i(t)\right) X + \left(\sum_{i=1}^{m_\nu} \lambda_{2,i}^{(j)}\psi_i(t)\right) Y,$$

wobei die $\lambda_{k,i}^{(j)}$ reelle Zahlen sind. Durch Umformungen dieser Gleichung erhält man

$$L_j(t,X,Y) = \sum_{i=1}^{m_\nu} \left(\lambda_{0,i}^{(j)} + \lambda_{1,i}^{(j)}X + \lambda_{2,i}^{(j)}Y\right)\psi_i(t) = \sum_{i=1}^{m_\nu} \Lambda_{i,j}(X,Y)\psi_i(t).$$

Dabei ist $\Lambda_{i,j}(X,Y)$ ein lineares Polynom in $\mathbb{R}[X,Y]$. Die Matrix $\mathbb{M}_\nu(\phi)$ ist definiert als eine $m_\nu \times r_\nu$ - Matrix, deren Einträge die lineare Polynome $\Lambda_{i,j}(X,Y)$ sind:

$$\mathbb{M}_\nu(\phi) := \begin{pmatrix} \Lambda_{1,1} & \Lambda_{1,2} & \dots & \Lambda_{1,r_\nu} \\ \Lambda_{2,1} & \Lambda_{2,2} & \dots & \Lambda_{2,r_\nu} \\ \vdots & \vdots & & \vdots \\ \Lambda_{m_\nu,1} & \Lambda_{m_\nu,2} & \dots & \Lambda_{m_\nu,r_\nu} \end{pmatrix}.$$

Die Abhängigkeit von ϕ wird im Folgenden der Einfachheit halber weggelassen. Interpretiert man diese Konstruktion geometrisch, so definiert $L_j(t_0,X,Y) = 0$ für ein bestimmtes t_0 eine Gerade im $\mathbb{R}^2$. Wenn man den Parameter t variiert, ändert sich die Gerade ebenfalls. Darüber hinaus gilt nach der Definition von L_j:

$$L_j\left(t, \frac{f_1(t)}{f_0(t)}, \frac{f_2(t)}{f_0(t)}\right) = 0. \tag{31}$$

Dies bedeutet, dass die Gerade, die zu dem gegebenen Parameter t_0 gehört, immer durch den Punkt $\phi(t_0)$ geht, der auf der durch ϕ parametrisierten Kurve liegt, zumindest für alle t_0 für die $f_0(t_0) \neq 0$ gilt. Für $L_1, ..., L_{r_\nu}$ bedeutet dies, dass sich für ein bestimmtes t_0 die r_ν dazugehörigen Geraden alle in dem Punkt $\phi(t_0)$ schneiden.
Nach Definition existieren drei Matrizen $M_{\nu,i},\ i = 0,1,2$, deren Einträge reelle Zahlen sind und

$$\mathbb{M}_\nu(X,Y) = M_{\nu,0} + XM_{\nu,1} + YM_{\nu,2}$$

gilt. Daraus folgt, dass die Matrix $\mathbb{M}_\nu := \mathbb{M}_\nu(X,Y)$ leicht an jedem Punkt $\mathbf{P} \in \mathbb{R}^2$ ausgewertet werden kann. Sei $\mathbf{P} \in \mathbb{R}^2$ ein derartiger Punkt, dass $\mathbf{P} = \phi(t_0)$ für einen Parameterwert t_0 gilt. Nach Konstruktion von $\mathbb{M}_\nu$ und (31) erhält man

$$[\psi_1(t_0)\dots\psi_{m_\nu}(t_0)] \times \mathbb{M}_\nu(\mathbf{P}) = [L_1(t_0,\mathbf{P})\dots L_{r_\nu}(t_0,\mathbf{P})] = [0\dots 0].$$

Daraus folgt, dass eine nicht-triviale lineare Kombination der Reihen von $\mathbb{M}_\nu$ entsteht, wenn man an einem Punkt $\mathbf{P}$ auswertet, der zum Bild von ϕ gehört, da die Funktionen

ψ_i nicht alle simultan verschwinden. Es stellt sich heraus, dass es einen kritischen Grad ν_0 gibt, dass für alle Matrizen $\mathbb{M}_\nu$ mit $\nu \geq \nu_0$ die obige Eigenschaft die Punkte $\mathbf{P}$ als zum (algebraischen) Abschluss des Bildes von ϕ gehörig charakterisiert. Für rationale Bézierkurven vom Grad d liegt der kritische Grad ν_0 bei $\nu_0 := d-1$ [Bus14]. Für unseren Fall bedeutet dies, dass $\nu_0 = 1$ gilt. Die natürlichen Zahlen r_ν und m_ν bezeichnen die Anzahl an Zeilen und Spalten der Matrix $\mathbb{M}_\nu$. Es lässt sich zeigen, dass für $\nu \geq \nu_0$ $r_\nu \geq m_\nu$ gilt und damit für den Rang $rank(\mathbb{M}_\nu(\phi)(\mathbf{P})) \leq m_\nu$ für alle $\mathbf{P} \in \mathbb{R}^2$ erfüllt ist. Es ist sogar belegt, dass $rank(\mathbb{M}_\nu(\phi)(\mathbf{P})) < m_\nu$ genau dann gilt, wenn $\mathbf{P}$ aus dem Abschluss des Bildes von ϕ stammt. Die dazugehörigen Beweise sind in [BB10] ausformuliert. Diese Eigenschaften führen dazu, dass die Matrix $\mathbb{M}_\nu$ für alle $\nu \geq \nu_0$ als implizite Repräsentation der Parametrisierung ϕ verwendet werden kann.
Nun soll die Matrix $\mathbb{M}_\nu$ explizit für unsere quadratische Bézierkurve aufgestellt werden. Wie bereits erwähnt, ist $\nu_0 = 1$. Da wir natürlich die kleinstmögliche passende Matrix suchen, wird $\mathbb{M}_1$ bestimmt, d.h. $\nu = \nu_0 = 1$ gesetzt. Nach der theoretischen Einführung, müssen 3-Tupel (g_0, g_1, g_2) gefunden werden, die (30) erfüllen und die Bernstein-Polynome als Basis haben, also von folgender Form sind:

$$g_j(t) = \sum_{i=0}^{\nu} \alpha_{j,i} B_{i,\nu}(t) = \sum_{i=0}^{1} \alpha_{j,i} B_{i,1}(t), \qquad \alpha_{j,i} \in \mathbb{R} \tag{32}$$

Zu diesem Zweck wird eine Matrix $\mathbb{S}_\nu \equiv \mathbb{S}_1$ wie folgt aufgestellt: die erste Spalte wird mit den Koeffizienten von $B_{0,1}(t)f_0(t)$ in der Bernstein-Basis $\mathfrak{B}_{\nu+d} \equiv \mathfrak{B}_3 = \{B_{0,3}(t), ..., B_{3,3}(t)\}$ gefüllt und die zweite Spalte mit den Koeffizienten von $B_{1,1}(t)f_0(t)$. Anschließend werden zwei weitere solche Blöcke von Spalten angehängt bezüglich den Polynomen $f_1(t)$ und $f_2(t)$. Die Matrix $\mathbb{S}_1$ hat damit die Dimensionen $(d+\nu+1) \times 3(\nu+1) \equiv 4 \times 6$ und erfüllt die Gleichheit

$$[B_{0,3}(t), ..., B_{3,3}(t)]\mathbb{S}_1 = [B_{0,1}(t)f_0(t), B_{1,1}(t)f_0(t), ..., B_{0,1}(t)f_2(t), B_{1,1}(t)f_2(t)].$$

Konkret nach (22) ist $\mathbb{S}_1$ von folgender Form:

$$\mathbb{S}_1 = \begin{pmatrix} 1 & 0 & x_0 & 0 & y_0 & 0 \\ \frac{2}{3}w_1 & \frac{1}{3} & \frac{2}{3}w_1x_1 & \frac{1}{3}x_0 & \frac{2}{3}w_1y_1 & \frac{1}{3}y_0 \\ \frac{1}{3} & \frac{2}{3}w_1 & \frac{1}{3}x_2 & \frac{2}{3}w_1x_1 & \frac{1}{3}y_2 & \frac{2}{3}w_1y_1 \\ 0 & 1 & 0 & x_2 & 0 & y_2 \end{pmatrix}$$

Nach der Definition von $\mathbb{S}_1$ ist jeder Vektor in seinem Kern von der Form

$$[\alpha_{0,0}, \alpha_{0,1}, ..., \alpha_{2,0}, \alpha_{2,1}]^T$$

und ergibt ein 3-Tupel von Polynomen (32), die (30) erfüllen. Deshalb entspricht der Kern von $\mathbb{S}_1$ einer $6 \times r_\nu$-Matrix der Form

$$Ke(\mathbb{S}_1) = \begin{bmatrix} M_{1,0} \\ M_{1,1} \\ M_{1,2} \end{bmatrix}, \tag{33}$$

wobei die Blockmatrizen $M_{1,0}$, $M_{1,1}$ und $M_{1,2}$ von der Dimension $2 \times r_\nu$ sind. Nach der Definition von $\mathbb{M}_1$ erhält man

$$\mathbb{M}_1 = M_{1,0} + XM_{1,1} + YM_{1,2}.$$

Für die explizite Bestimmung der Blockmatrizen und deren Anzahl von Spalten r_ν muss der Kern der Matrix $\mathbb{S}_1$ berechnet werden. Dies wird mittels einer singulären Wertezerlegung (SVD) durchgeführt.
Eine SVD zerlegt eine Matrix $A \in \mathbb{R}^{m \times r}$ mit $m \leq r$ in $A = U\Sigma V^T$, wobei $U \in \mathbb{R}^{m \times m}$ und $V = [v_1, ..., v_r] \in \mathbb{R}^{r \times r}$ orthogonale Matrizen sind und die Matrix Σ von der Form

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & \sigma_m & 0 & \dots & 0 \end{bmatrix} \in \mathbb{R}^{m \times r}$$

nicht-negative Diagonaleinträge, die sogenannten Singulärwerte, in absteigender Ordnung $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m \geq 0$ enthält. Um den Bezug der Singulärwerte zu der Matrix A besser darzustellen, wird oft $\sigma_i(A)$ statt σ_i geschrieben. Der Rang einer Matrix lässt sich durch den Zusammenhang $rank(A) = \max\{i : \sigma_i(A) \neq 0\}$ mittels der SVD bestimmen. Falls $\sigma_i(A)$ „klein“ ist im Vergleich zu $\|A\|_2 = \sigma_1(A)$, so ist der dazugehörige rechte Singulärvektor v_i „fast“ ein Nullvektor für A. Der numerische Rang der Matrix A ist definiert durch $rank(A, \epsilon) := \max\{k : \sigma_k(A) > \epsilon\}$. Die Bestimmung des numerischen Rangs r_ϵ von A mit der Toleranz ϵ führt zu einem numerischen Kern von A, der von den Vektoren $v_{r_\epsilon+1}, ..., v_r$ aufgespannt wird. Diese numerischen Überlegungen sind wichtig, da bei der praktischen Berechnung der SVD Fehler gemacht werden und Matrizen etwa dazu tendieren, vollen Rang zu besitzen.
Basierend auf diesen Überlegungen folgt, dass $\mathbb{M}_1$ von der SVD der Matrix $\mathbb{S}_1$ abgelesen werden kann. Mit dem geeigneten ϵ, bringt die SVD einen numerischen Rang r_ϵ hervor. Damit beschreiben die letzten $r - r_\epsilon$ Spalten der Matrix V den numerischen Kern von $\mathbb{S}_1$, ist von der Form (33) und stellt eine numerische Approximation von $\mathbb{M}_1$ durch $M_{1,0} + XM_{1,1} + YM_{1,2}$ dar. Insgesamt ist nun eine implizite Matrixdarstellung der ursprünglichen Parametrisierung hergestellt worden.

Seien die quadratischen rationalen Bézierkurven $\mathbf{P}(t)$ und $\mathbf{Q}(u)$ gegeben. Zunächst wird nach obigem Vorgehen die implizite Matrixdarstellung von $\mathbf{P}(t)$ erzeugt, die in $\mathbb{M}_1 = M_{1,0} + XM_{1,1} + YM_{1,2}$ resultiert. Die zweite Parametrisierung $\mathbf{Q}(u) = (\frac{\overline{x}(u)}{\overline{w}(u)}, \frac{\overline{y}(u)}{\overline{w}(u)})$ wird mittels $X = \frac{\overline{x}(u)}{\overline{w}(u)}$ und $Y = \frac{\overline{y}(u)}{\overline{w}(u)}$ in die implizite Form eingesetzt. Die resultierende Matrix wird mit $\overline{w}(u)$ multipliziert, um eine Matrix mit polynomialen Einträgen zu erhalten. Die resultierende Matrix wird mit $\mathbf{M}(u)$ bezeichnet und setzt sich derart zusammen:

$$\begin{aligned}\mathbf{M}(u) = &\left((1-u)^2 + 2\overline{w_1}u(1-u) + u^2\right) M_{1,0} + \left(\overline{x_0}(1-u)^2 + 2\overline{w_1x_1}u(1-u) + \overline{x_2}u^2\right) M_{1,1} \\ &+ \left(\overline{y_0}(1-u)^2 + 2\overline{w_1y_1}u(1-u) + \overline{y_2}u^2\right) M_{1,2}\end{aligned}$$

Auf Grund der Eigenschaften der impliziten Darstellung und der Resultante folgt, dass die Schnittpunkte den Nullstellen der Determinante von $\mathbf{M}(u)$ entsprechen. Das heißt, es gilt

$$det(\mathbf{M}(u)) = 0$$

zu lösen, was auf ein Eigenwertproblem zurückgeführt werden kann, wie es in [MD92] aufgeführt ist.
Die Matrix $\mathbf{M}(u)$ wird dazu zunächst in Potenzbasis-Schreibweise umformuliert:

$$\begin{aligned}\mathbf{M}(u) &= u^2(M_{1,0} + \overline{x_2}M_{1,1} + \overline{y_2}M_{1,2}) + u(1-u)(2\overline{w_1}M_0 + 2\overline{w_1x_1}M_1 + 2\overline{w_1y_1}M_2) \\ &\quad + (1-u)^2(M_0 + \overline{x_0}M_1 + \overline{y_0}M_2) \\ &=: u^2N_2 + u(1-u)N_1 + (1-u)^2N_0\end{aligned}$$

Division durch $(1-u)^2$ ergibt auf der rechten Seite:

$$N_2\left(\frac{u}{1-u}\right)^2 + N_1\left(\frac{u}{1-u}\right) + N_0 \tag{34}$$

Substituiert man $s := \frac{u}{1-u}$, so erhält man die angestrebte Potenzbasis:

$$\mathbf{L}(s) = N_2s^2 + N_1s + N_0$$

Ursprünglich war man an den Nullstellen der Determinante von $\mathbf{M}(u)$ im Bereich $[0,1]$ interessiert. Nach der Reparametrisierung sind die Nullstellen der Determinante von $\mathbf{L}(s)$ in dem Bereich $[0,\infty]$ gesucht. Die Nullstellen können durch ein Eigenwertproblem gelöst werden. Dieses benötigt jedoch eine Umformung, die die Matrix N_2^{-1} beinhaltet. Die Berechnung der Inversen ist numerisch instabil, wenn die Nullstellen sehr groß werden, bzw. im Ausgangsproblem der Schnittpunkt auf $\mathbf{Q}(u)$ nahe am Endpunkt ist.

In solchen Fällen ist N_2 beinahe singulär. Die Vorgehensweise nach dem Eigenwertproblem ist in [MD92] zu finden. Hier wird jedoch der Ansatz, der ein verallgemeinertes Eigenwertproblem behandelt und in [BB10, S. 15 ff.] aufgeführt ist, vorgestellt. Dieser vermeidet die Berechnung der Inversen von N_2.

Theorem 4.1. *Gegeben sei das Matrix-Polynom* $\mathbf{L}(s)$. *Die Nullstellen der Determinante des Polynoms entsprechen den Eigenwerten des verallgemeinerten Systems* $A - sB$, *wobei*

$$A = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ N_0 & N_1 \end{bmatrix} \quad \textit{und} \quad B = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -N_2 \end{bmatrix}$$

gilt und $\mathbf{0}$ *bzw.* $\mathbf{I}$ *die Null- bzw. Einheitsmatrizen sind.*

Der Beweis wird über Ähnlichkeitstransformationen geführt und kann in [GLR82] nachvollzogen werden. Ein verallgemeinerter Eigenwert der Matrizen A und B von der Größe $m \times n$ ist ein Wert in der Menge

$$\lambda(A, B) := \{s \in \mathbb{R} : rank(A - sB) < \min\{m, n\}\}.$$

Die Implementierung zur Berechnung der verallgemeinerten Eigenwerte geht nach dem sogenannten QZ-Algorithmus vor, der orthogonale Transformationen auf A und B anwendet, um A auf Hessenberg-Form A' zu reduzieren und B auf obere Dreiecksform B'. Anschließend wird implizit der QR-Algorithmus auf $B'^{-1}A'$ angewandt, ohne es explizit zu formulieren. Dazu wurde die Implementierung aus LAPACK verwendet.
Damit sind die gewünschten Eigenwerte berechnet und die Schnittpunkte auf $\mathbf{Q}(u)$. Jedoch sind wir nur an den Parameterwerten $u \in [0, 1]$ interessiert, d.h. negative Eigenwerte werden verworfen. Für einen Eigenwert $s_0 \in [0, \infty]$ erhält man den entsprechenden Parameter auf $\mathbf{Q}$ durch die Resubstitution $u_0 := \dfrac{s_0}{1 + s_0}$. Dies ergibt eine Liste von Schnittpunkten $\mathbf{Q}(u_0)$ mit $u_0 \in [0, 1]$.
Nichtsdestotrotz können diese Punkte auf $\mathbf{P}(t)$ nicht in dem Intervall $t \in [0, 1]$ liegen, da bei der Implizitierung diese Einschränkung vernachlässigt wurde. Deshalb muss abschließend mit dem Punkt-Projektion-Algorithmus aus dem vorherigen Abschnitt der Parameter t_0 bestimmt werden, für den $\mathbf{P}(t_0) = \mathbf{Q}(u_0)$ gilt. Ist $t_0 \in [0, 1]$, so ist der Punkt $\mathbf{P}(t_0)$ bzw. $\mathbf{Q}(u_0)$ ein echter Schnittpunkt der beiden Kreisbögen, die durch $\mathbf{P}(t)$ bzw. $\mathbf{Q}(u)$ parametrisiert werden.
Zu dem vorgestellten Verfahren gilt es noch zwei Anmerkungen zu machen. Bei der Division durch $(1 - u)$ in (34) verliert man je nach Implementierung die mögliche Lösung $u = 1$, da der entsprechende Eigenwert $s = \infty$ nur theoretisch gut dargestellt werden kann. Das praktische Verfahren behebt diesen Makel dadurch, dass die Parametrisierung von $\mathbf{Q}(u)$ durch Vertauschen des Startpunktes $\mathbf{Q}_0$ und Endpunktes $\mathbf{Q}_2$ umgekehrt wird und das Verfahren erneut aufgerufen wird, wenn die Anzahl der Lösungen kleiner

als zwei ist und somit der Verlust einer Lösung möglich sein könnte.
Bei der Implizitierung verlässt man die Einschränkung von t auf $[0,1]$, wie es in der Parametrisierung noch gilt, und stellt implizit den gesamten Kreis dar. Dies führt zu der Frage, ob die implizite Matrixdarstellung, die den gesamten Kreis definiert, nicht ähnlichen numerischen Problemen unterliegt, wie die traditionelle Kreisdarstellung. Die Matrix $\mathbb{S}_1$ setzt sich aus den Werten des Kontrolldreiecks und dem Gewicht w_1 so zusammen, dass die Einträge der Matrix im Bereich der Eingabedaten bleiben. Es könnte sich jedoch ein Problem bei der Berechnung der SVD ergeben, wenn die $r - r_\epsilon$ letzten Spalten der Matrix V numerisch ungünstige Einträge erhalten, da daraus die Koeffizienten des Matrix-Polynoms gebildet werden. Jedoch bilden diese Spalten eine orthonormale Basis des Kerns und sind daher bei vernünftigen Algorithmen zur Berechnung einer SVD numerisch stabil. In [CD06] sind einige Algorithmen aufgeführt, die etwa nur mit unitären Transformationen arbeiten. Dadurch ist sichergestellt, dass zur Bestimmung der Matrizen $M_{1,i}$ keine numerischen Herausforderungen hinzukommen. Anschließend werden die Matrizen N_i berechnet, die aus Linearkombinationen der Matrizen $M_{1,i}$ bestehen. Die Koeffizienten ergeben sich aus dem Kontrollpolygon und dem Gewicht $\overline{w}_1$ des parametrisierten Kreisbogens. Sind diese Parameter wie in dem angestrebten Anwendungsfall nicht überaus groß, so sind auch diese Berechnungen unabhängig von einem großen Radius. Der Übergang zu einem Eigenwertproblem bedingt die Transformation des Parameters u zu s, die nur dann problematisch ist, wenn u nahe bei 1 ist. Dies wurde bereits behandelt. Somit ist der Übergang von dem Schnittproblem zweier Kreisbögen, über eine Parametrisierung durch rationale Bézierkurven und eine Implizitierung hin zu der Formel $\mathbf{L}(s) = N_2 s^2 + N_1 s + N_0$ theoretisch numerisch stabil und ohne weiteren Einfluss eines großen Radius. $\mathbf{L}(s)$ kann abschließend mit einem numerisch stabilen Verfahren aus der linearen Algebra gelöst werden. Ob sich diese theoretische Überlegung auch in der Praxis durchsetzt, wird mittels Tests im weiteren Verlauf herausgearbeitet.

4.3.2.7. Nachoptimierung eines Schnittpunkts

Das Ergebnis der Schnittpunktberechnung unterliegt trotz der gewählten stabilen Methoden einem gewissen Fehler aufgrund von Rechenungenauigkeiten. Da die Kurven parametrisiert sind und zudem die Ableitungen zur Verfügung stehen, bietet sich hier erneut das Newton-Verfahren an, um ein Ergebnis im Nachgang zu verbessern. Seien die beiden Kurven

$$\mathbf{P}(t) = \left(\frac{x(t)}{w(t)}, \frac{y(t)}{w(t)}\right) =: (\mathbf{P}(t).x, \mathbf{P}(t).y) \quad \text{und}$$
$$\mathbf{Q}(u) = \left(\frac{\overline{x}(u)}{\overline{w}(u)}, \frac{\overline{y}(u)}{\overline{w}(u)}\right) =: (\mathbf{Q}(u).x, \mathbf{Q}(u).y)$$

und die Lösung des Schnittalgorithmus $\mathbf{S}$ gegeben. Die Projektionen von $\mathbf{S}$ auf $\mathbf{P}(t)$ bzw. $\mathbf{Q}(u)$ seien $\mathbf{P}(t_0)$ bzw. $\mathbf{Q}(u_0)$. Es werden hier die Projektionen angegeben, da $\mathbf{S}$ wegen Ungenauigkeiten der Berechnungen nicht exakt (bzw. bis auf ein kleines ϵ) auf beiden Kreisbögen liegen muss. Sei dies der Fall, so wäre die Nachoptimierung an dieser Stelle beendet.
Das Ziel der Optimierung ist es, die Parameter (t^*, u^*) zu finden, an denen

$$F(t^*, u^*) := \mathbf{P}(t^*).x - \mathbf{Q}(u^*).x = 0$$
$$G(t^*, u^*) := \mathbf{P}(t^*).y - \mathbf{Q}(u^*).y = 0$$

gilt. Ein Iterationsschritt sieht folgendermaßen aus:

$$\begin{aligned}\begin{pmatrix} t_{i+1} \\ u_{i+1} \end{pmatrix} &= \begin{pmatrix} t_i \\ u_i \end{pmatrix} - \begin{pmatrix} F_t(t_i, u_i) & F_u(t_i, u_i) \\ G_t(t_i, u_i) & G_u(t_i, u_i) \end{pmatrix}^{-1} \begin{pmatrix} F(t_i, u_i) \\ G(t_i, u_i) \end{pmatrix} \\ &= \begin{pmatrix} t_i \\ u_i \end{pmatrix} - \begin{pmatrix} \mathbf{P}'(t_i).x & -\mathbf{Q}'(u_i).x \\ \mathbf{P}'(t_i).y & -\mathbf{Q}'(u_i).y \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{P}(t_i).x - \mathbf{Q}(u_i).x \\ \mathbf{P}(t_i).y - \mathbf{Q}(u_i).y \end{pmatrix}\end{aligned}$$

Dabei sind F_t und G_t die Richtungsableitungen nach t und $\mathbf{P}'$ und $\mathbf{Q}'$ die Ableitungen nach (16). Es wird solange iteriert, bis

$$\left\| \begin{pmatrix} F(t_i, u_i) \\ G(t_i, u_i) \end{pmatrix} \right\|_2 < \epsilon$$

erfüllt ist.

5. Vergleich der Ansätze durch Testen

In den vorherigen drei Abschnitten wurde beschrieben, welche Probleme auftreten, wenn der Radius eines Kreises groß wird. Dazu wurden die Algorithmen zur Berechnung der Projektion eines Punktes auf einen Kreisbogen nach der standardmäßigen Methode mit der Mittelpunkt-Radius Darstellung eines Kreises (*Standard*), nach einer ausgefeilteren Überlegung ebenfalls mittels der Mittelpunkt-Radius Darstellung (*Krümmung*) und basierend auf der Kreisbogendarstellung durch rationale Bézierkurven (*RatBez*) vorgestellt. Darüber hinaus wurden Algorithmen zur Berechnung von Schnittpunkten zweier Kreisbögen für den Standard- und RatBez-Ansatz aufgezeigt. Auf theoretischer Ebene wurden etwaige Schwachpunkte der einzelnen Verfahren bereits angesprochen. Nun werden diese anhand praktischer Tests gegenübergestellt.

5.1. Test der Punktprojektion

5.1.1. Konstruktion aus drei Punkten

Der erste Test ist analog zu dem Testfall zur Problemspezifikation im zweiten Abschnitt. Dieser wird hier noch einmal kurz vorgestellt. Der Kreisbogen wird durch drei Punkte aufgespannt. Der Startpunkt $A(0, 100))$ und der Endpunkt $B(100, 0)$ sind fix, wohingegen der Punkt dazwischen, C, sich auf der Winkelhalbierenden der Achsen im 1. Quadranten bewegt. Dieser liegt initial bei $(50.9, 50.9)$ und nähert sich im Verlauf des Tests $(50, 50)$ an. Damit wandert der Mittelpunkt des Kreises auf der Winkelhalbierenden immer weiter nach außen und der Radius wird größer. Berechnet werden soll der Abstand des Punktes $D(100, 100)$ zum Kreisbogen, was nach der Konstruktion des Beispiels dem Abstand von C zu D entspricht. Das Setting ist in Abbildung 1 aufgezeichnet. Um die Situation in allgemeinere Lage zu bringen, wurde jeder Punkt um den Vektor $(51.212..., 51.212...)$ verschoben. Die Tabelle der Testergebnisse sieht wie folgt aus:

c	50.9	$50 + 9 * 10^{-5}$	$50 + 9 * 10^{-8}$	$50 + 9 * 10^{-10}$	$50 + 9 * 10^{-13}$
Radius	$2 * 10^4$	$2 * 10^8$	$2 * 10^{11}$	$2 * 10^{13}$	$2 * 10^{16}$
Standard	$1.6 * 10^{-13}$	$4.4 * 10^{-9}$	$9.1 * 10^{-7}$	$8.0 * 10^{-5}$	$5.3 * 10^{-2}$
RatBez*	$4.3 * 10^{-14}$	$3.8 * 10^{-10}$	$1.3 * 10^{-7}$	$1.3 * 10^{-9}$	$1.3 * 10^{-12}$
RatBez	$2.8 * 10^{-14}$	0.0	$2.8 * 10^{-14}$	$2.8 * 10^{-14}$	0.0
Krümmung	$2.6 * 10^{-13}$	$2.6 * 10^{-9}$	$6.2 * 10^{-7}$	$7.4 * 10^{-7}$	$7.4 * 10^{-7}$
Gerade	1.3	$1.2 * 10^{-4}$	$1.3 * 10^{-7}$	$1.2 * 10^{-9}$	$1.3 * 10^{-12}$

Tab. 3: *Abstandsberechnung zwischen einem Punkt und einem durch drei Punkte gegebenen Kreisbogen im Vergleich*

In den Spalten der Tabelle sind die Werte von c als Koordinaten von C abgetragen.

Es wurde etwa in der zweiten Spalte mit dem Punkt $(50.00009, 50.00009)$ getestet. In der zweiten Reihe ist der jeweilige Radius angegeben, den der zu Grunde liegende Vollkreis besitzt. RatBez* ist die auf der rationalen Bézierkurve basierenden Methode, die im Gegensatz zur RatBez Zeile auf die Nachoptimierung beim Erstellen verzichtet. Krümmung bezeichnet die alternative Kreisdarstellung, die mit der Krümmung anstatt großen Radien operiert. In der letzten Zeile ist der Fehler aufgeführt, den man macht, wenn man eine Gerade durch A und B annimmt. Die Standard-Methode ist die Projektions-Methode aus der Problemstellung in 2.3. Genauso ist die Geradenapproximation aus Tabelle 1 übernommen.
In Tabelle 3 sieht man bei RatBez*, dass diese Vorgehensweise in den ersten beiden Tests noch sehr gut ist, bei Radius 10^{11} relativ schlecht ist und sich ab Radius 10^{11} kaum von dem Geradenansatz hinsichtlich des Fehlers unterscheidet. Dies liegt daran, dass die Berechnung von w_1 numerisch ungenau wird und stets 1 liefert. D.h. die rationale Bézierkurve stellt eine Strecke dar. Die deutlich überlegene Methode ist RatBez. Sie liefert in jedem Test einen Fehler, der der numerischen Null zugeordnet werden kann. Man muss jedoch anmerken, dass dieser Ansatz in diesem Testsetting bevorzugt wird, da bei der Optimierung zur Berechnung der Parameter für die Bézierkurve diese so angepasst werden, dass der Abstand der Kurve zu dem mittleren Punkt[9] minimiert wird. Der mittlere Punkt ist hier C. Nach der Optimierung liegt C sehr genau auf der Kurve. Das Ergebnis der Punktprojektion soll C liefern. Dieser wird offensichtlich sehr zuverlässig gefunden, jedoch ist das nur so gut möglich, da C exakt auf der Kurve liegt. Damit soll jedoch nicht verhehlt werden, dass das RatBez-Verfahren sehr gut funktioniert und unabhängig von Radius exakte Ergebnisse liefert für den Anwendungsfall, dass die gegebenen Punkte nicht zu weit voneinander und von der 0 entfernt sind. Der Test wäre jedoch etwas aussagekräftiger, wenn der Projektionspunkt nicht einer der Punkte wäre, der verwendet wird, um den Kreisbogen aufzustellen.
Die Krümmungsmethode ist in der ersten Spalte noch sehr exakt, dann wird sie jedoch etwas fehleranfälliger. Im Gegensatz zur Standardmethode wird der Fehler nicht größer als 10^{-7}. Das deutet schon darauf hin, dass dieser Ansatz besser ist, als der Standardansatz, obwohl er auf derselben Kreisdarstellung arbeitet. Hinzu kommt, dass die Krümmungsmethode primär auf zwei Punkten und der Tangente arbeitet und in diesem Testfall dadurch benachteiligt ist. Im folgenden Test wird deshalb etwas genauer auf die Krümmungsmethode eingegangen.

[9] Mit mittlerem Punkt wird vereinfacht der Punkt bezeichnet, der bei der Konstruktion mit drei gegebenen Punkten auf dem Kreisbogen zwischen Start und Ende liegt.

5.1.2. Konstruktion aus zwei Punkten und einer Tangente

In diesem Testsetting sind zwei Punkte und die Tangente am Start gegeben. Der Startpunkt S hat die Koordinaten $(0, -1)$, der Endpunkt E $(0, 1)$. Die Tangente t an S hat die Koordinaten $(\mathit{1}, \mathit{d})$. Kursive Zahlen bedeuten hier und im Folgenden, dass für den Testfall etwa die x-Koordinate von t nicht exakt 1 betrug, sondern einige Nachkommastellen hinzugefügt wurden, um zu vermeiden, dass numerische Effekte durch nicht Vorhandensein von Nachkommastellen unterschlagen werden. Für den Testfall an sich sind diese Nachkommastellen jedoch unerheblich und lenken hier in der Ausarbeitung nur vom Wesentlichen ab, weshalb sie durch diese Notation weggelassen wurden.
Die y-Koordinate von t, d, wird im Laufe des Tests verschieden gesetzt. Je größer d ist, desto flacher wird der Kreisbogen und desto größer wird der dazugehörige Kreis. Der Punkt T befindet sich auf der x-Achse und soll auf den Kreisbogen projiziert werden. Im Testverlauf wird T auf der x-Achse bewegt. Die Situation ist in Abbildung 16 dargestellt. Wie im vorigen Test wird auch hier das Szenario in allgemeinere Lage versetzt,

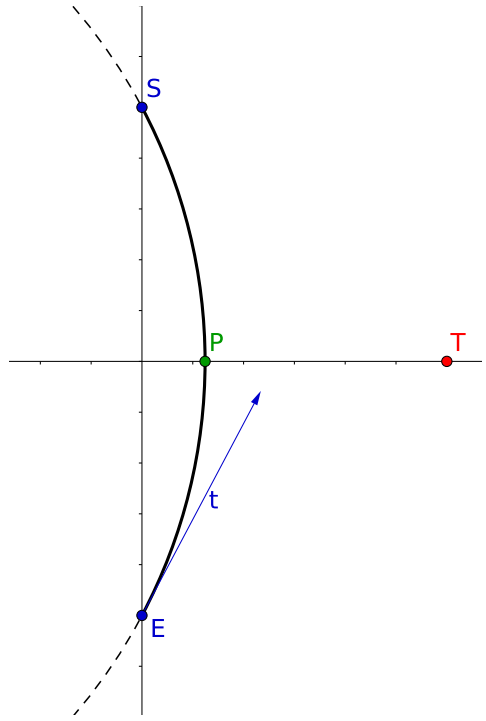

Abb. 16: *Test zur Punktprojektion mit Tangente*

indem jeder Punkt um den Vektor $V = (\mathit{512}, \mathit{108})$ verschoben wurde. Es wird darauf hingewiesen, dass die Koordinaten kursiv geschrieben sind, demzufolge im Vergleich mit dem tatsächlichen Test Nachkommastellen fehlen.
Im Gegensatz zum vorigen Test, ist hier die Projektion P nicht gegeben und kann auch nicht ohne Weiteres exakt bestimmt werden. Dafür ist RatBez, wie oben beschrieben, nicht bevorteilt. Im Folgenden wird die Genauigkeit des Verfahrens dadurch geprüft,

Steigung	$6.1 * 10^2$	$6.1 * 10^6$	$6.1 * 10^{11}$
Radius	$5.4 * 10^2$	$5.4 * 10^6$	$5.4 * 10^{11}$
$T.x = 5.5 * 10^2$			
Standard	$(9.2 * 10^{-4}, 2.8 * 10^{-14})$	$(9.2 * 10^{-8}, 0)$	$(1.7 * 10^{-5}, 0)$
RatBez	$(9.2 * 10^{-4}, 0)$	$(9.2 * 10^{-8}, 1.4 * 10^{-14})$	$(9.1 * 10^{-13}, 0)$
Krümmung	$(9.2 * 10^{-4}, 1.4 * 10^{-14})$	$(9.2 * 10^{-8}, 0)$	$(9.1 * 10^{-13}, 1.4 * 10^{-14})$
$T.x = 5.1 * 10^7$			
Standard	$(9.2 * 10^{-4}, 2.8 * 10^{-14})$	$(9.2 * 10^{-8}, 1.4 * 10^{-14})$	$(1.7 * 10^{-5}, 0)$
RatBez	$(9.2 * 10^{-4}, 0)$	$(9.2 * 10^{-8}, 1.4 * 10^{-14})$	$(9.1 * 10^{-13}, 0)$
Krümmung	$(9.2 * 10^{-4}, 1.4 * 10^{-14})$	$(9.0 * 10^{-8}, 2.7 * 10^{-11})$	$(1.9 * 10^{-9}, 1.7 * 10^{-9})$

Tab. 4: *Projektion zweier Punkte T nach der Standard-, RatBez- und Krümmungsmethode für verschiedene Tangentensteigungen*

dass die y-Koordinate von P Null ist und sich P mit wachsendem d an den Ursprung annähert. Für diese Überprüfungen wurde von der Lösung der Vektor V wieder abgezogen.

In Tabelle 4 sind die Projektionen von T in zwei Testreihen aufgetragen. Die erste wurde für den Punkt $T = (5.5 * 10^2, 0)$, die zweite für $T = (5.1 * 10^7, 0)$ durchgeführt. In den Spalten wurde die Tangentensteigung verändert. Der dazugehörige Radius des Vollkreises ist ebenfalls dargestellt. Dazu lässt sich anmerken, dass die Nullstellen der Tangentengeraden $1.8 * 10^{-3}$, $1.8 * 10^{-7}$ und $1.8 * 10^{-12}$ nach wachsender Steigung geordnet betragen. Das sind die Punkte auf der x-Achse, an denen die Verlängerungen von t in 16 dieselbige schneiden. Diese Punkte stellen für große Tangentensteigungen eine gute Annäherung an P dar. Damit stellen diese Werte ein weiteres Kriterium an die Güte der Lösung dar.

Betrachtet man die erste Testreihe, so fällt auf, dass bei RatBez und Krümmung stets der Projektionspunkt bis auf die numerische Null auf der x-Achse liegt und auch der Abstand zum Ursprung geringer wird. Die x-Werte sind stets etwas kleiner als die Nullstellen der Tangentengeraden, was der Vorstellung sehr nahe kommt. Man sieht auch, dass die Standardmethode für die ersten beiden Spalten vergleichbare Ergebnisse wie die anderen beiden Methoden liefert, beim letzten Test bei einem Radius von etwa 10^{11} jedoch einen großen Fehler macht. Die x-Koordinate liegt bei 10^{-5}, sollte jedoch in jedem Fall kleiner als $1.8 * 10^{-12}$ betragen.

Im Theorieteil zur Krümmungsmethode wurde angesprochen, dass eine kleine Krümmung für die Berechnung kritisch sein könnte. Die ist in der ersten Testreihe nicht zu erkennen. Die theoretische Überlegung ist korrekt, jedoch wirkt sie sich hier nicht oder zumindest nicht merklich aus.

In der zweiten Testreihe wurde der zu projizierende Punkt T mit der x-Koordinate $5.1 * 10^7$ weiter entfernt vom Kreisbogen gewählt. Ein robustes Verfahren sollte unab-

hängig von der Entfernung des zu projizierenden Punktes zum Objekt die Projektion berechnen können. Beim Standard- und RatBez-Ansatz erhält man erwartungsgemäß dieselben Resultate wie in der vorigen Testreihe. Bei der Krümmungsmethode zeigen sich bei einer Steigung von $6.1 * 10^6$ und $6.1 * 10^{11}$ deutliche Unterschiede. Der Projektionspunkt liegt nicht mehr auf der x-Achse und die x-Koordinate ist in der letzten Spalte größer als die Nullstelle der Tangentengerade.
Das liegt daran, dass die Berechnung nach der Krümmungsmethode durch den Strahlensatzansatz in (2) und (3) abhängig von der Entfernung des zu projizierenden Punktes ist. Dies ist ein Nachteil. Bei der Standard- und RatBez-Methode ist hingegen nur die Richtung, in der T liegt, entscheidend. Ist die Krümmung klein und der Abstand zu T groß, so sind die numerischen Effekte merklich und können in Tabelle 4 gesehen werden.

5.1.3. Projektion auf Kreisbögen mit großem Öffnungswinkel

Dieser Test soll überprüfen, wie gut die RatBez-Methode funktioniert, wenn der Kreisbogen beinahe ein Vollkreis ist. Die Erwartung ist, dass die RatBez nicht mehr so gut funktioniert, wie bei kleinem Öffnungswinkel, da allein zur Erstellung des Kreisbogens auf die Kreisdarstellung mit Mittelpunkt und Radius zurückgegriffen wird. Darüber hinaus ist es nicht zu vermeiden, dass einige Kontrollpunkte große Koordinaten haben, wenn beinahe ein Vollkreis mit großem Radius dargestellt werden soll. Dadurch erhält man dieselben numerischen Probleme wie bei der Standard-Kreisdarstellung.

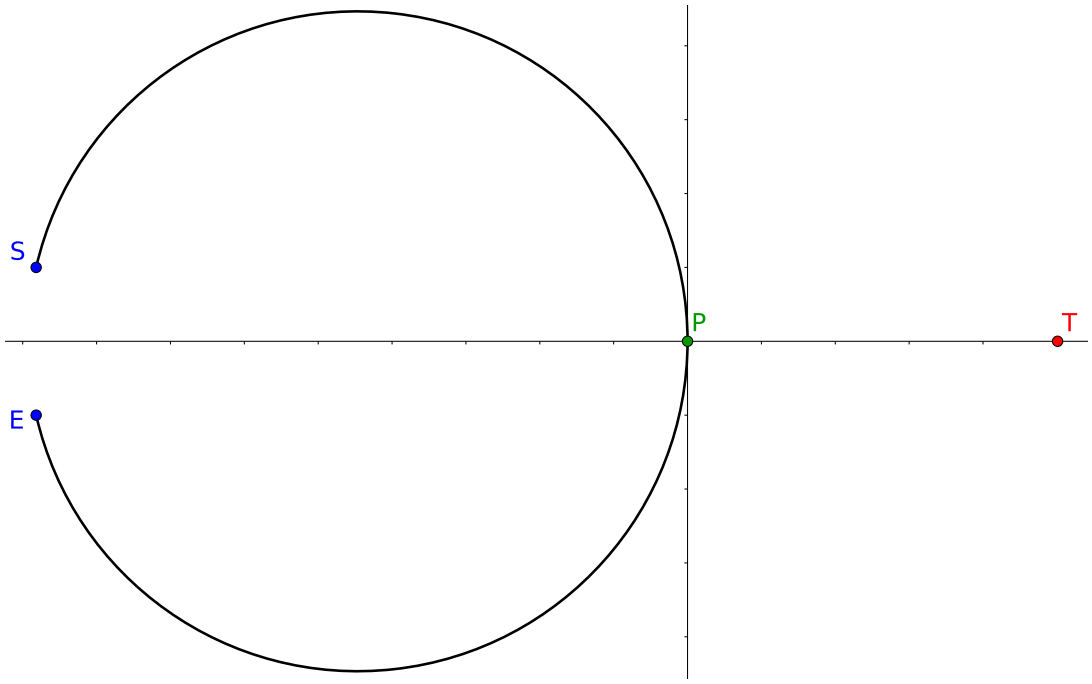

Abb. 17: *Test zur Punktprojektion mit beinahe einem Vollkreis*

Die folgende Testreihe baut auf dem Projektionstest mit drei Punkten von oben auf. Der Verschiebungsvektor in allgemeinere Lage ist wie oben $(51, 51)$. Die Punkte $S = (d, 3)$ und $E = (d, -3)$ werden im Laufe des Tests simultan parallel zur x-Achse bewegt. Da sie symmetrisch um die x-Achse liegen, befindet sich der Mittelpunkt des dazugehörigen Vollkreises ebenfalls auf der x-Achse. Der mittlere Punkt auf dem Kreisbogen wird im Ursprung fixiert. Da der Testpunkt T ebenfalls auf der x-Achse gewählt wird, fallen der mittlere Punkt und die Projektion P zusammen. Eine Übersicht liefert Abbildung 17. Tabelle 5 visualisiert die Testergebnisse.

d	$-1.5 * 10^2$	$-1.5 * 10^6$	$-1.5 * 10^{10}$	$-1.5 * 10^{12}$
Radius	77	$7.7 * 10^5$	$7.7 * 10^9$	$7.7 * 10^{11}$
Standard	$1.4 * 10^{-14}$	$2.6 * 10^{-10}$	$5.6 * 10^{-7}$	$5.7 * 10^{-5}$
RatBez	$2.8 * 10^{-14}$	$2.8 * 10^{-10}$	$2.3 * 10^{-7}$	$5.7 * 10^{-5}$

Tab. 5: *Abstandsberechnung zwischen einem Punkt und einem durch drei Punkte gegebenen Kreisbogen mit großem Öffnungswinkel im Vergleich*

In der Tabelle sind in der ersten Zeile die x-Koordinaten d von S und E angegeben und in der Zweiten die Radien der dazugehörigen Vollkreise. Im Anschluss werden die Standard- und RatBez-Methode verglichen. Im Test wurden jeweils die Abstände von T zum Kreisbogen berechnet. Nach Aufgabenstellung weiß man, dass dieser 100 betragen muss. Die Abweichung zu 100 ist für die jeweiligen Testfälle und Methoden aufgetragen.
Man sieht relativ schnell, dass die Vorüberlegung richtig war und der RatBez-Ansatz in diesem Szenario etwa die gleichen Fehler macht, wie die Standard-Kreisdarstellung. Mal ist das eine Verfahren leicht besser, mal das andere, jedoch nie um eine Größenordnung. Beide sind deutlich von der Größe des Radius abhängig.
Die Krümmungsmethode konnte hier nicht direkt angewandt werden, da die Strahlensatz-Konstruktion in der angegebenen Form nur für Öffnungswinkel von weniger als 180° geeignet ist.

5.2. Test der Schnittpunktberechnung

Neben der Punktprojektion soll auch die Schnittpunktberechnung ausführlicher getestet werden. Im Folgenden wird die Standardmethode dem RatBez-Ansatz gegenübergestellt.

5.2.1. Testszenario 1

In Abbildung 18 ist das erste Testszenario schematisch dargestellt. Der erste Kreisbogen wird durch den Startpunkt S_1, den Endpunkt E_1 und C_1 aufgespannt. C_1 bleibt dabei

über den gesamten Test hinweg fest bei $(\cos(0.5), \sin(0.5))$. $S_1 = (\cos(\pi + d_1), \sin(\pi + d_1))$ und $E_1 = (\cos(-d_1), \sin(-d_1))$ werden im Laufe des Tests so bewegt, dass sie symmetrisch zur Winkelhalbierenden des ersten Quadranten sind und der Kreisbogen mindestens den ersten Quadranten enthält, d.h. es gilt $d_1 \geq 0$. Da die drei Punkte allesamt auf dem Einheitskreis liegen, bildet der erste Kreisbogen auch einen Ausschnitt des Einheitskreises. Der zweite Kreisbogen wird durch die Punkte S_2, E_2 und C_2 aufgespannt. $S_2 = (0, 1)$ und $E_2 = (1, 0)$ sind fest, wohingegen $C_2 = (d_2, d_2)$ im Laufe des Testes auf der Winkelhalbierenden des ersten Quadranten wandert. Liegt C_2 nahe bei $(0.5, 0.5)$, so ist der Radius des zum zweiten Kreisbogen gehörigen Kreises groß.

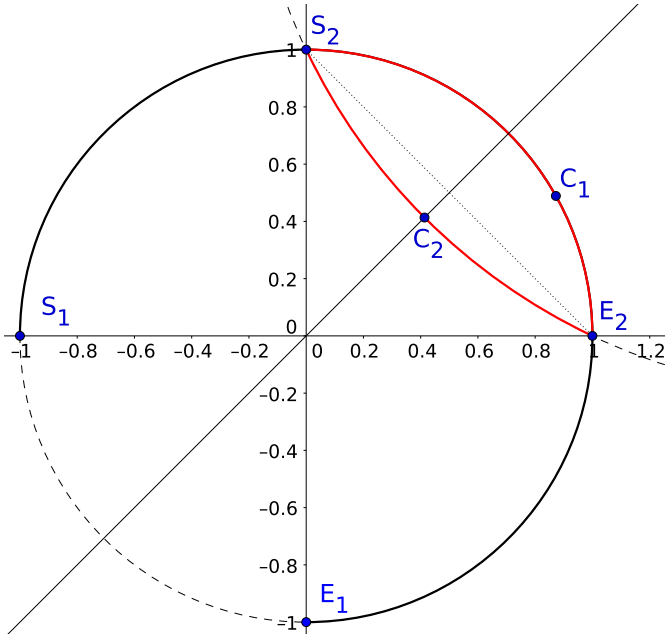

Abb. 18: *Graphische Darstellung des ersten Testszenarios zur Schnittberechnung*

Schneidet man die beiden Kreisbögen, so sollten in jedem Fall S_2 und E_2 als Schnittpunkte zurückgegeben werden. Im Folgenden wird in jedem Testfall überprüft, ob zwei Lösungen berechnet wurden und als Qualitätskriterium der Lösungen wird der Abstand zu $(1, 0)$ und $(0, 1)$ gemessen.
Für den Parameter d_1 wurden die Werte $\{0.0, \text{FEPS}, 1.0, \frac{\pi}{4} - \text{FEPS}, \frac{\pi}{4}, \frac{\pi}{4} + 1, \frac{\pi}{2}, \frac{3\pi}{4} - 0.1, \frac{3\pi}{4} - \text{FEPS} * 10\}$ getestet. FEPS entspricht etwa $1.2 * 10^{-7}$. Die Werte wurden so gewählt, damit der erste Kreisbogen verschiedene Öffnungswinkel von Viertelkreis über Halbkreis bis beinahe Vollkreis annimmt. Für d_2 und damit die Beschaffenheit des zweiten Kreisbogens wurden drei Testreihen durchgeführt, die verschiedene Klassen von Kreisbögen abtesten sollen.

- Testreihe 1
  Bei der ersten Testreihe wird der Motivationsfall dieser Arbeit abgehandelt. Der zweite Kreisbogen hat einen kleinen Öffnungswinkel und der dazugehörige Voll-

kreis einen großen Radius. Dazu wird d_2 nahe an $(0.5, 0.5)$ heranbewegt. Die Testwerte sind aus der Menge
$\{0.5 - 10^{-i} | 1 \leq i < 15, i \in \mathbb{N}\} \cup \{0.5 - 5 * 10^{-i} | 1 \leq i < 15, i \in \mathbb{N}\}$.
Insgesamt wurden dadurch in der ersten Testreihe 252 Tests durchlaufen. Dia-

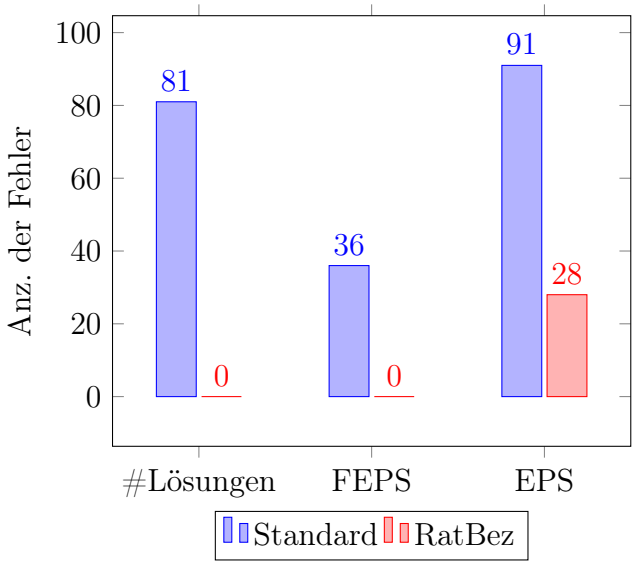

Abb. 19: *Testszenario 1 - Testreihe 1*

gramm 19 zeigt die Resultate. Getestet wurde, ob jeweils zwei Lösungen berechnet werden, ob der Abstand einer Lösung zu dem Entsprechenden wahren Schnittpunkt kleiner als FEPS bzw. kleiner als EPS $\approx 2.2 * 10^{-13}$ ist. Dem Säulendiagramm kann man entnehmen, dass es dem Standardansatz in 81 Fällen nicht gelungen ist, zwei Lösungen zurückzugeben. Dies geschah in Testdurchläufen, bei denen d_2 etwa bei $0.5 - 10^{-10}, 0.5 - 10^{-10}$ bzw. näher an $(0.5, 0.5)$ lag. Durch den daraus resultierenden großen Radius und der numerischen Ungenauigkeit ergab sich bei der Berechnung von μ in (1) stets 0, weshalb es nur eine Lösung gab. Zu den FEPS- und EPS-Fehlern muss angemerkt werden, dass jeder FEPS-Fehler folgerichtig auch ein EPS-Fehler ist. Darüber hinaus wurden diese sowohl für $(0, 1)$ als auch $(1, 0)$ berechnet, d.h. es könnte maximal doppelt so viele FEPS-Fehler wie Testdurchläufe geben. Sind bei einem Testfall nicht zwei Lösungen berechnet worden, so wurde dies zur Anzahl der Lösungs-Fehler hinzugezählt, anschließend jedoch nicht mehr auf FEPS oder EPS überprüft. Man könnte folgerichtig die Fehler bezüglich der Lösungsanzahl zu den FEPS- und EPS-Fehlern hinzuzählen. Im Diagramm sieht man, dass der RatBez-Ansatz stets mindestens auf FEPS Genauigkeit die Schnittpunkte berechnen konnte, wohingegen die Standardmethode bereits 36 FEPS-Abweichungen verzeichnete. An EPS-Fehlern hat die Standardmethode ebenfalls mehr zu verzeichnen. Die 28 EPS-Fehler der RatBez-Methode sind allesamt für $d_1 = \frac{3\pi}{4} - \text{FEPS} * 10$ zu verzeichnen. Hier liegt die Vermutung

nahe, dass dies daran liegt, dass der erste Kreisbogen fast einem Vollkreis entspricht und die Punkte S_1 und E_1 so nahe beieinander liegen, dass der Kreisbogen beinahe entartet und so nicht mit ausreichender Genauigkeit aufgespannt werden kann.
Man kann zusammenfassen, dass erwartungsgemäß der RatBez-Ansatz für die erste Testreihe bessere Ergebnisse liefern konnte.

- Testreihe 2
  In der zweiten Testreihe wurde d_2 aus der Menge $\{0.5 - i * 2 | 1 \leq i \leq 100, i \in \mathbb{N}\}$ gewählt. Dies ist ein Bereich, in dem die Kreise moderate Radien haben und exakte Berechnungen auch in der Mittelpunkt-Radius Darstellung möglich sein sollten. Abbildung 20 zeigt die Resultate. Die Gesamtzahl der Tests belief sich auf 900.

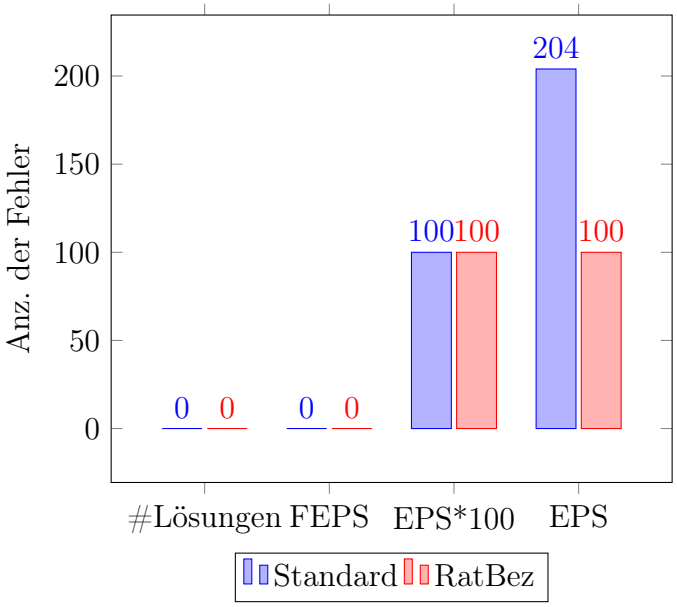

Abb. 20: *Testszenario 1 - Testreihe 2*

Im Unterschied zur obigen Übersicht werden hier zusätzlich die Fehler bezüglich EPS*100-Genauigkeit dargestellt. Sowohl der Standard- als auch der RatBez-Ansatz, fanden für alle Testfälle die gewünschten Lösungen auf FEPS-Genauigkeit. EPS*100-Genauigkeit konnten beide Verfahren im Falle von $d_1 = \frac{3\pi}{4} - \text{FEPS} * 10$ aus obigem Grund nicht halten. Bezüglich EPS-Genauigkeit schneidet RatBez besser ab, wobei dies eine relativ hohe Genauigkeit ist und diese nicht unbedingt erreicht werden muss.

- Testreihe 3
  In der dritten Testreihe wird der zweite Kreisbogen so gewählt, dass er beinahe einen Vollkreis mit großem Radius beschreibt. Demzufolge stammt d_2 aus der

Menge $\{0.5 - i * 10^7 | 1 \leq i < 20, i \in \mathbb{N}\}$.

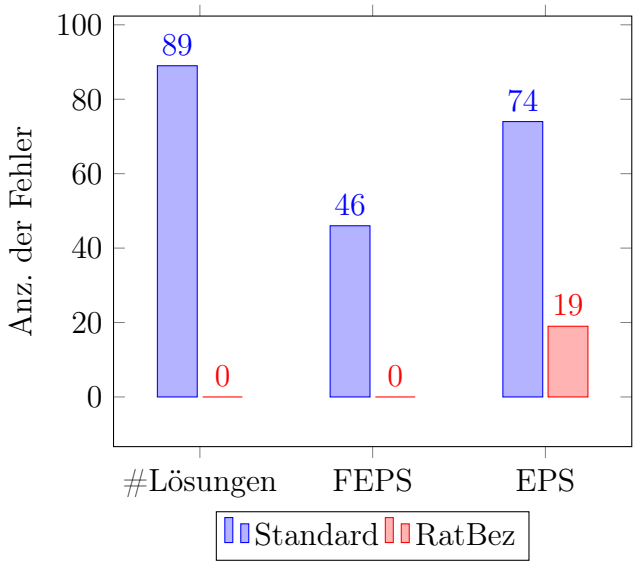

Abb. 21: *Testszenario 1 - Testreihe 3*

Abbildung 21 zeigt die Resultate. Der RatBez-Ansatz schafft diese Herausforderung sehr gut zu bewältigen und weist erst Fehler im EPS-Bereich auf, die erneut auf $d_1 = \frac{3\pi}{4} - \text{FEPS} * 10$ zurückzuführen sind. In 89 der insgesamt 171 Testfälle kommt es beim Standardansatz bei der Berechnung von μ in (1) zu numerischen Problemen. Der Term unter der Wurzel ist entweder 0 oder sogar negativ, was in beiden Fällen zu einer unzureichenden Anzahl an Lösungen führt. Darüber hinaus gibt es 46 Lösungen, die um mehr als FEPS von einem der beiden Schnittpunkte abweicht.

Insgesamt zeigt die RatBez-Methode in diesem Testsetting bessere Ergebnisse. Man muss jedoch etwas vorsichtig sein, da die beiden gesuchten Lösungen die Endpunkte des zweiten Kreisbogens sind. Für die Standardmethode ist dies unerheblich, da sie mit Vollkreisen arbeitet. Der RatBez-Ansatz könnte dadurch jedoch bevorzugt sein. Deshalb folgt ein weiteres Testszenario, in dem dieser Vorteil nicht auftritt.

5.2.2. Testszenario 2

Dieses Testszenario ist dem obigen relativ ähnlich. Der erste Kreisbogen wird durch die Punkte S_1, E_1 und C aufgespannt. $C = (0, 1)$ ist dabei fest und $E_1 = (d_1, -3)$ ist die Spiegelung von $S_1 = (d_1, 3)$ an der x-Achse. Der Faktor d_1 wird im Laufe des Tests im negativen Zahlenbereich bewegt. Analog ist der zweite Kreisbogen durch die Punkte S_2, E_2 und C definiert. $S_2 = (d_2, 2)$ und $E_2 = (d_2, -2)$ bewegen sich im Testverlauf auf Parallelen zur x-Achse im positiven Bereich der x-Achse.

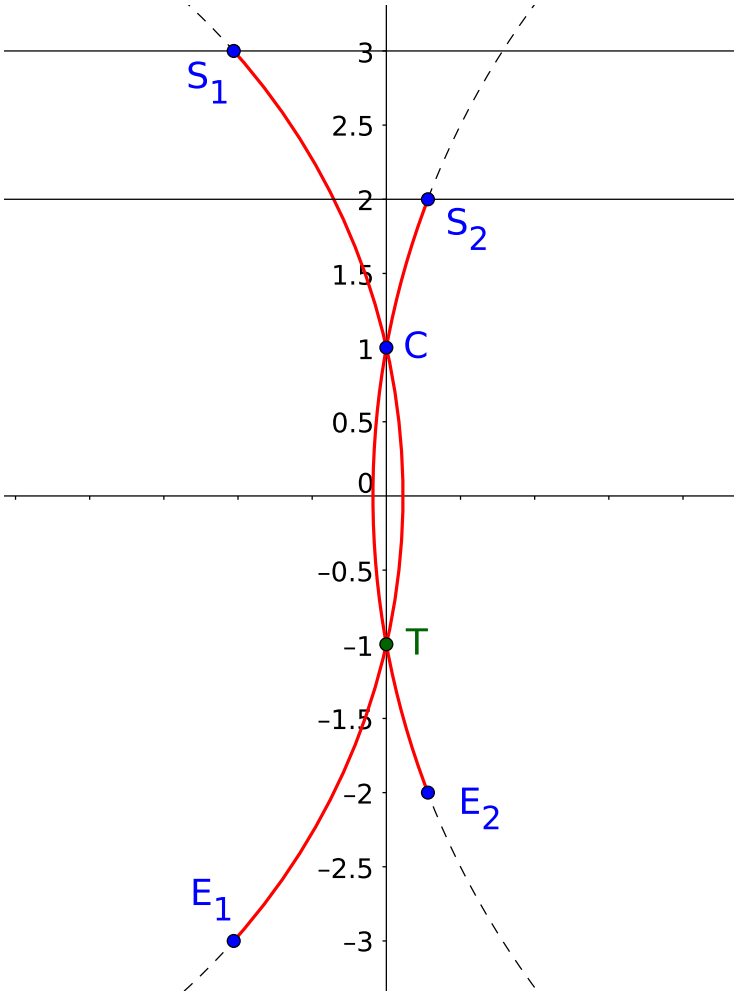

Abb. 22: *Graphische Darstellung des zweiten Testszenarios zur Schnittberechnung*

Abbildung 22 dient zur Illustrierung des Szenarios. Trivialer Weise ist C einer der beiden Schnittpunkte. Auf Grund von Symmetrie ist $T = (0, -1)$ ebenfalls ein Schnittpunkt. Dieser wird im Gegenzug zu obigem Testszenario nicht bei der Modellierung der Kreisbögen verwendet. Dadurch sollte weder RatBez noch Standard im Vorteil sein. Der Abstand der Lösungen der beiden zu testenden Verfahren zu T wird als Maßstab für die Exaktheit der Verfahren funktionieren. Dieses Testszenario ist ebenfalls in drei Testreihen unterteilt:

- Testreihe 1
  Der Parameter d_1 ist aus der Menge

$$\left\{ -\frac{1}{100 * i + 1} \middle| 0 \leq i < 10, i \in \mathbb{N} \right\}$$

  und d_2 aus

$$\left\{ \frac{1}{25 * i + 1} \middle| 0 \leq i < 100, i \in \mathbb{N} \right\}.$$

Damit sind beide Kreisbögen kleine Ausschnitte von Kreisen mit großem Radius. Die Testergebnisse sind in Abbildung 23 zu sehen.

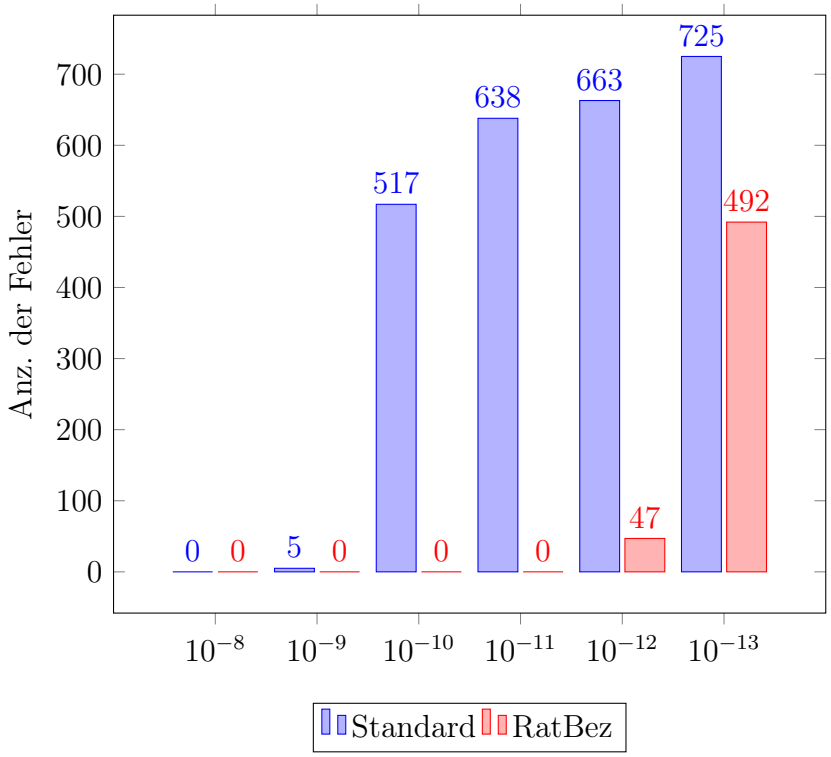

Abb. 23: *Testszenario 2 - Testreihe 1*

In den Spalten ist jeweils die Anzahl der Testfälle angegeben, bei denen keine Lösung erzielt werden konnte, deren Abstand zu T geringer war als das angegebenen Toleranzniveau. Sowohl die Standard- als auch die RatBez-Methode fanden stets zwei Lösungen und eine Lösung davon war maximal um 10^{-8} von T entfernt. Bei dem Genauigkeitstest auf 10^{-10} gab es beim Standard-Ansatz 517 Fehlerfälle bei 1000 Testdurchläufen. RatBez konnte immer noch 0 Fehler aufweisen. Erst bei 10^{-12} musste RatBez die ersten Fehlerfälle verzeichnen. Bei 10^{-13} stehen 725 Fehler der Standard-Methode 492 Fehlern der RatBez-Methode gegenüber. RatBez konnte folglich merklich bessere Ergebnisse liefern. Nimmt man 10^{-7} als kritisches Genauigkeitsintervall, so konnte die Standard-Methode ebenfalls ausreichende Ergebnisse erzielen.

- Testreihe 2
  Um die Standard-Methode zu einem größeren Fehler zu treiben, wird in dieser Testreihe nach derselben Art und Weise getestet, die Start- und Endpunkte der Kreisbögen rücken nur näher an die y-Achse heran. Dabei ist d_1 aus der Menge

$$\left\{ -\frac{1}{100 * i + 1} \middle| 0 \leq i < 10, i \in \mathbb{N} \right\}$$

und d_2 aus

$$\left\{ \frac{1}{500 * i + 1} \middle| 0 \leq i < 100, i \in \mathbb{N} \right\}$$

gewählt. Die Testergebnisse liefert Graphik 24.

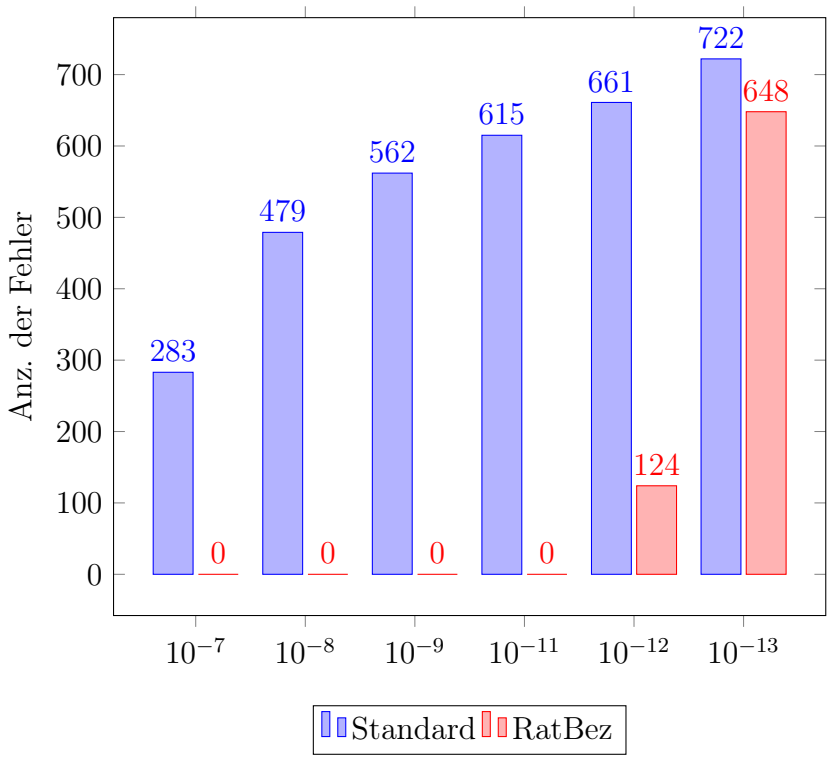

Abb. 24: *Testszenario 2 - Testreihe 2*

Im Gegensatz zu der vorherigen Testreihe gibt es nun einige Fehler bei 10^{-7}-Genauigkeit bei der Standard-Methode. Die RatBez-Methode verzeichnet erst bei 10^{-12}-Niveau 124 Fehler. Dies zeigt erneut die Stärke des Ansatzes, der auf der Darstellung durch rationale Bézierkurven beruht. Die Unabhängigkeit vom Radius macht ein solches Ergebnis möglich, wohingegen der Standard-Ansatz mit wachsendem Radius beliebig große Fehler hinnehmen muss.

- Testreihe 3
  In der dritten Testreihe ist es nun das Ziel zu überprüfen, ob das RatBez-Verfahren auch in Testfällen, für die es nicht primär entwickelt wurde, konkurrenzfähig ist. Dazu wurde d_1 aus der Menge

  $$\{-200 * i - 1 | 0 \leq i < 100, i \in \mathbb{N}\}$$

  und d_2 aus

  $$\{500 * i + 1 | 0 \leq i < 100, i \in \mathbb{N}\}$$

  gewählt. Die dadurch definierten Kreisbögen entsprechen beinahe einem Vollkreis.

Da in einem solchen Fall zur Erstellung eines Kreisbogens nach dem RatBez-Verfahren auf die Mittelpunkt-Radius Darstellung zurückgegriffen wird, ist kein besseres Ergebnis als nach der Standard-Methode zu erwarten.

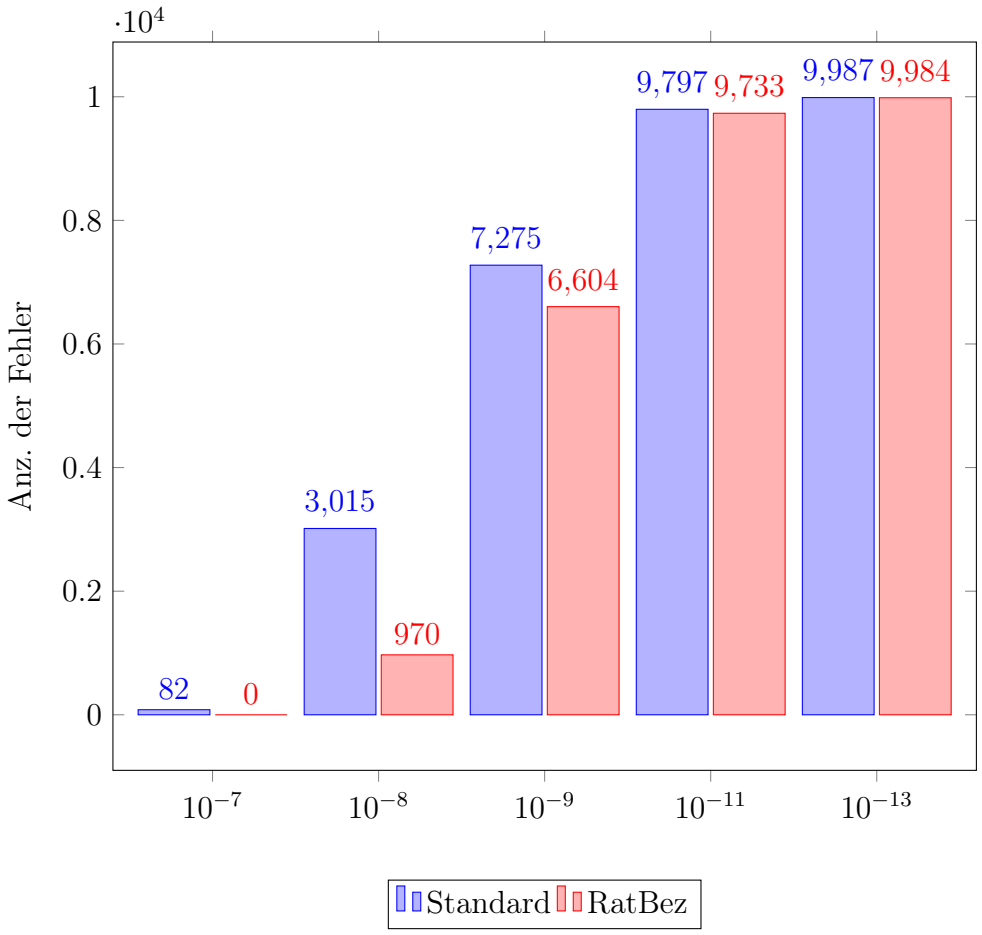

Abb. 25: *Testszenario 2 - Testreihe 3*

Die Resultate in 25 überraschen etwas, da RatBez leicht besser abschneidet als die Standard-Methode. RatBez hat stets weniger Fehler und erzeugt sogar keine Abweichungen auf 10^{-7}-Genauigkeit, wie es in 82 Fällen der Standard-Methode passiert. Dies ist erstaunlich. Es muss an der Berechnung der Schnittpunkte liegen, die nach diesem Test tendenziell bei der RatBez-Methode etwas numerisch stabiler erfolgt.

5.2.3. Robustheit der Schnittberechnung ohne Nachoptimierung

Bei den theoretischen Überlegungen zur numerischen Stabilität der Schnittpunktberechnung durch Implizitierung im Abschnitt 4.3.2.6 wurden Argumente geliefert, dass die Implizitierung bei kleinen Öffnungswinkeln unabhängig vom Radius numerisch stabil sein sollte. Dies soll an dieser Stelle durch Testresultate unterstrichen werden. Dazu

wurde das Testszenario 2 herangezogen und die einzelnen Testreihen wurden ohne die Nachoptimierung bei der Schnittberechnung durchlaufen.

- Testreihe 1
  In der ersten Testreihe wurden exakt dieselben Ergebnisse erzielt, wie sie in Abbildung 23 zu sehen sind. Die Schnittberechnung konnten auch ohne Nachoptimierung diese guten Resultate erzielen. Damit ist aus Sicht der Praxis die theoretische Überlegung untermauert, dass die Implizitierung zu keiner Abhängigkeit vom Radius führt.

- Testreihe 2
  In der zweiten Testreihe wurden ebenfalls dieselben Ergebnisse erzielt, wie sie in Abbildung 24 zu sehen sind, was die eben getroffene Aussage unterstreicht.

- Testreihe 3
  In der dritten Testreihe, die große Kreise mit großen Öffnungswinkeln bedient, waren deutliche Unterschiede zu Diagramm 25 zu verzeichnen. Im Gegensatz

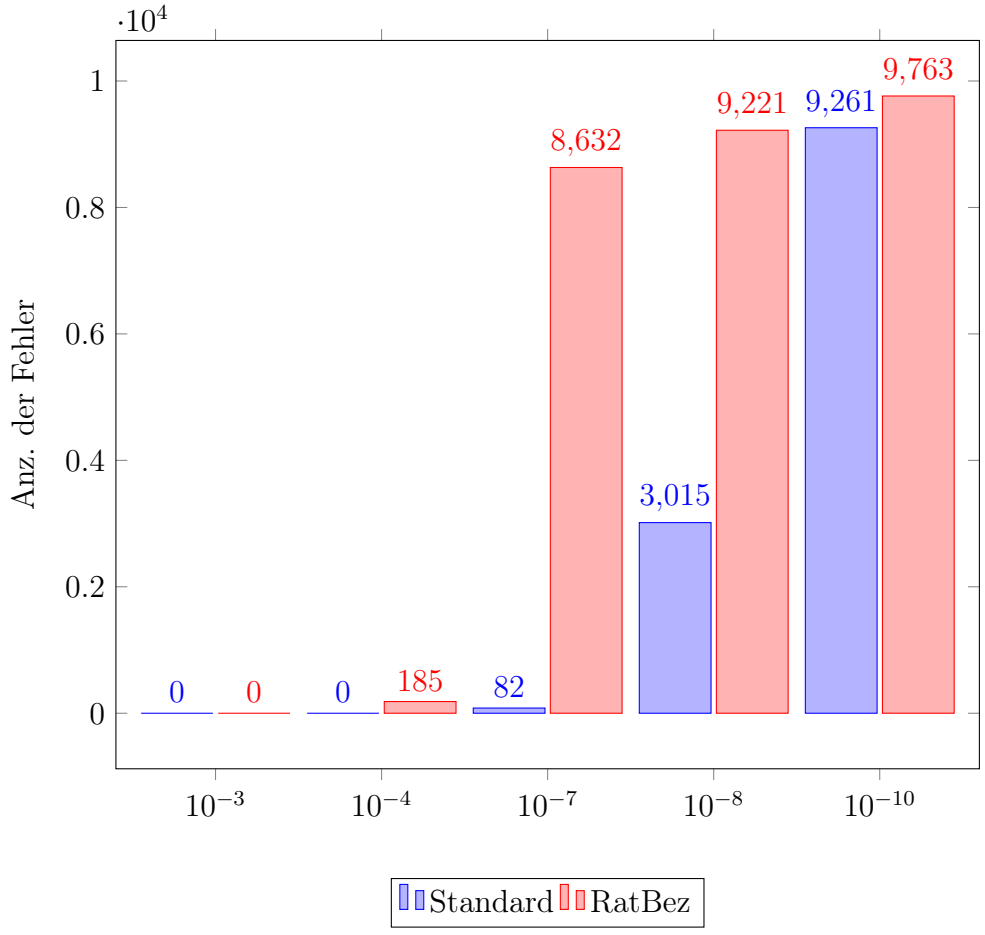

Abb. 26: *Testszenario 2 - Ohne Optimierung*

zu den Ergebnissen mit Nachoptimierung ist hier die RatBez-Methode deutlich

schlechter als der Standard-Ansatz und nicht mehr leicht besser. Erst bei einer Toleranz von 10^{-3} weist RatBez keine Fehler mehr auf. Bei 10^{-7}, wo sich die Standardmethode zum ersten Mal Fehler zugesteht, verzeichnet RatBez bereits 8632. Die Nachoptimierung ist folglich in diesem Fall sehr erfolgreich und schafft es, ein durch den großen Radius unausweichlich schlechtes Resultat deutlich zu verbessern. Dadurch ist die RatBez-Methode sogar leicht besser ist als die Standard-Lösung, obwohl diese Testreihe nicht zu der Anwendung gehört, für die die RatBez-Modellierung primär gedacht ist.

5.3. Vergleich der Ausführungszeiten

Das Hauptaugenmerk dieser Arbeit liegt darauf, eine numerisch bessere und stabilere Methode zu finden, unter anderem Punkte zu projizieren oder Schnittpunkte zu berechnen. Dabei spielte die Ausführungszeit bisher keine Rolle. Da wir uns jetzt jedoch dem Ende dieser Arbeit nähern und im nächsten Kapitel die Implementierung im Zuge des SMAP-Projekts angesprochen wird, so sollte hier am Ende der Testdurchläufe zumindest eine tendenzielle Aussage stehen, mit wie viel zusätzlicher Ausführungszeit man die höhere Genauigkeit erkauft.
Da der Kreis in der Mittelpunkt-Radius Darstellung die einfachste Methode ist, alle Punkte zu beschreiben, die von einem gegebenen Punkt einen bestimmten Abstand besitzen und sowohl die Punktprojektion als auch die Schnittpunktberechnung sehr einfach nach der Standard-Vorgehensweise durchgeführt werden, muss einem klar sein, dass jeder numerisch stabilere Ansatz mit deutlich mehr Ausführungszeit verbunden ist.
Die Krümmungsmethode ist wie die Standard-Methode ein direkter Ansatz. Die Ausführungszeit ist weitgehend unabhängig von der Eingabe, da stets die gleiche Berechnung durchgeführt wird. Die Berechnungen sind bei der Krümmungsmethode etwas aufwändiger, weshalb man knapp die vierfache Zeit benötigt, wie bei der Standard-Methode. Dies kommt dem Verhältnis der Berechnungen relativ nahe.
Aufwändiger ist es dagegen eine zuverlässige Ausführungszeit für die rationale Bézier-methode zu ermitteln. Die Punktprojektion ist ein iterativer Algorithmus. Die Anzahl der Teilkreisbögen orientiert sich am Öffnungswinkel und bestimmt zugleich wie oft eine parametrisierte Kurve in die Implizitierung bei der Schnittberechnung eingesetzt werden muss. Darüber hinaus besitzen die Punktprojektion, die Schnittberechnung und sogar die Konstruktoren eine Nachoptimierung, deren Ausführungszeit durch eine maximale Iterationszahl begrenzt werden, jedoch nie exakt bestimmt werden kann. Deshalb dienen folgende Ausführungszeiten mehr einer Tendenz als einer wahren Aussage. Darüber hinaus muss angemerkt werden, dass der Fokus nicht auf die Optimierung der Ausführungszeit gelegt wurde und man diese mit etwas Refactoring durchaus

noch reduzieren könnte. Des Weiteren wäre eine Überlegung, auf Kosten der Genauigkeit die Nachoptimierung einzuschränken oder gar wegzulassen, um Zeit zu sparen, jedoch trotzdem eine vernünftige Genauigkeit zu erhalten. Dies sind nur Gedanken und ist nicht umgesetzt worden. Abgesehen davon kann so eine Entscheidung sinnvoll nur unter Einbezug einer Anwendung gefällt werden.

5.3.1. Ausführungszeit für Punktprojektion

Für die Bestimmung der Berechnungszeit der Punktprojektion wurde der in 1 dargestellte Algorithmus angewandt.

```
1  S ← (10,0);
2  r ← 10;
3  for i ← 1 to 100 do
4  |  E ← r ∗ (cos(i/100), sin(i/100));
5  |  for j ← 1 to 10 do
6  |  |  B ← r ∗ (cos(i/100 ∗ j/10), sin(i/100 ∗ j/10));
7  |  |  arc ← Erzeuge einen Kreisbogen aus S, E und B
8  |  |  for k ← 0 to 1000 do
9  |  |  |  P ← (k − 100, 5);
10 |  |  |  ProjectPoint(arc, P);
11 |  |  end
12 |  end
13 end
```

Algorithm 1: Algorithmus zur Bestimmung der Ausführungszeit der Punktprojektion

Der Kreisbogen auf den projiziert werden sollte, wird durch drei Punkte aufgebaut. Der Startpunkt S wird in der ersten Zeile auf $(10, 0)$ fest gesetzt. Der zu dem Kreisbogen gehörige Kreis hat seinen Mittelpunkt im Ursprung und Radius $r = 10$. Auf diesem Kreis wandert nun der Endpunkt E in Hundertstelschritten und der Punkt auf dem Bogen B wird an 10 verschiedenen Punkten zwischen S und E gewählt. Der zu projizierende Punkt P wandert an 1000 verschiedenen Punkten auf einer Parallele zur x-Achse durch 5. In Zeile 7 wird je nachdem, welche Methode getestet werden soll, ein Kreisbogen nach der Standard- oder RatBez-Methode erzeugt. Da die Konstruktoren verschieden sind, wird auch die Zeit gemessen, wenn Zeile 10 auskommentiert ist, also die Projektion nicht berechnet wird. Dabei wurden folgende Messungen gemacht:

	Konstruktion	mit Projektion
Standard	0.00048	0.07646
RatBez	0.05614	4.47945

Die Zeiten in Sekunden wurden auf dem Rechner des Autors gemessen. Jedoch sollten die absoluten Zeiten keine Rolle spielen. Entscheidend ist die folgende relative Auswertung. Die Zeit für die Konstruktion beträgt für den Standard-Fall etwa ein Hundertstel derer nach dem RatBez-Ansatz. Bezieht man die Projektion mit ein, so ist der Algorithmus nach RatBez etwa um das 58-fache langsamer. Dies gilt auch, wenn man die reine Zeit für die Punktprojektion betrachtet.
Die Konstruktion der rationalen Bézierkurven dauert sehr lange im Vergleich zum Standard-Ansatz, jedoch ist dies nicht so gewichtig, wenn dies nicht so oft aufgerufen wird. Die Projektion ist um mehr das 50-fache langsamer und führt zu einem großen Unterschied in der Ausführungszeit in diesem Test.

5.3.2. Ausführungszeit für Schnittpunktberechnung

Analog zu obigem Algorithmus zur Berechnung der Ausführungszeit, ist auch dieser aufgebaut. In Algorithmus 2 ist zu sehen, dass der Abschnitt zur Erstellung des ersten Kreisbogens aus dem obigen Pseudocode übernommen ist. Der zweite Kreisbogen wird analog zum ersten erzeugt und ist Teil des Kreises mit Mittelpunkt $(12.5, 0)$ und Radius $r2 = 4.5$.

```
1  S1 ← (10,0);
2  r1 ← 10;
3  S2 ← (8,0);
4  r2 ← 4.5;
5  for i ← 1 to 100 do
6  |  E1 ← r1 ∗ (cos(i/100), sin(i/100));
7  |  for j ← 1 to 10 do
8  |  |  B1 ← r ∗ (cos(i/100 ∗ j/10), sin(i/100 ∗ j/10));
9  |  |  arc1 ← Erzeuge den ersten Kreisbogen mit S1, E1 und B1
10 |  |  for k ← 1 to 40 do
11 |  |  |  E2 ← (12.5, 0) + r2 ∗ (cos(k/40), sin(k/40));
12 |  |  |  for l ← 1 to 10 do
13 |  |  |  |  B1 ← (12.5, 0) + r2 ∗ (cos(k/40 ∗ l/10), sin(k/40 ∗ l/10));
14 |  |  |  |  arc2 ← Erzeuge den zweiten Kreisbogen mit S2, E2 und B2
15 |  |  |  |  IntersectArcs(arc1, arc2);
16 |  |  |  end
17 |  |  end
18 |  end
19 end
```

Algorithm 2: Algorithmus zur Bestimmung der Ausführungszeit der Schnittpunktberechnung

Die Ausführungszeit wurde für $arc1$ und $arc2$ als Standard, bzw. RatBez-Kreisbögen

berechnet. Um die Konstruktionszeit zu ermitteln, wurde Zeile 15 auskommentiert.

	Konstruktion	mit Projektion
Standard	0.207344	0.44357
RatBez	21.5366	33.0893

Die Konstruktionszeit macht in diesem Test einen größeren Anteil aus, als im Vorigen. RatBez braucht für die Konstruktion erneut etwa 100 mal so lange wie die Standard-Methode. Dies ist konsistent zu obigem Test und sollte es auch sein. Für die reine Schnittpunktberechnung ergibt sich hier ziemlich genau ein Faktor 50, um den RatBez langsamer ist.

5.4. Fazit der Testreihen

In den Testdurchläufen wird die Motivation, die zu dieser Arbeit geführt hat, noch einmal deutlich unterstrichen. Die Standard-Methode basierend auf der Kreisdarstellung durch den Mittelpunkt und den Radius ist mit wachsendem Radius einem ebenso zunehmenden Fehler ausgesetzt.
Die Krümmungsmethode ist ein interessanter Ansatz, kann sich jedoch weder theoretisch, noch praktisch ganz von den Nachteilen der Mittelpunkt-Radius Darstellung befreien. Vor allem stellt die Abhängigkeit des zu projizierenden Punktes zum Kreisbogen ein Problem dar.
Für Kreisbögen mit kleiner Krümmung und kleinem Öffnungswinkel ist die Modellierung durch rationale Bézierkurven mit Nachoptimierung die überlegene Methode. Auch im Falle größer Öffnungswinkel ist der Ansatz nicht schlechter als die Standard-Methode. Ein Nachteil besteht in der Ausführungszeit, die mindestens 50 mal so groß ist, wie bei der bisherigen Implementierung.

6. Implementierungsdetails und Integration in den SMAP-Code

Die Modellierung von Kreisbögen durch rationale Bézierkurven hat herausragende numerische Stärken gezeigt, weshalb dieser Ansatz gewählt wurde, um in den bestehenden SMAP-Algorithmus eingebaut zu werden. Für die Kreisbögen basierend auf der Mittelpunkt-Radius Darstellung waren jedoch bereits einige weitere Funktionalitäten implementiert, die auch vom SMAP-Algorithmus benutzt werden. Deshalb war es nötig, diese Methoden ebenfalls in Form der rationalen Bézierdarstellung zu implementieren. Insgesamt umfasst die Implementierung ohne Testfälle etwa 3500 Codezeilen in der Sprache C++. Die wichtigsten Methoden werden im Folgenden aufgezählt und deren Idee beschrieben. Die Syntax orientiert sich an der Funktionenschreibweise in C++. In Klammern angegeben werden jeweils die Objekte mit Typangabe, die für die Funktionalität benötigt werden. Die Berechnungsart und das Resultat werden in textueller Form beschrieben.

6.1. Im Verlauf der Arbeit vorgestellte Funktionalität

- $create\_with\_third\_point(Point\ start, Point\ end, Point\ between)$
  Diese Methode bekommt drei Punkte und erzeugt daraus den dazugehörigen Kreisbogen.

- $create\_with\_tangent\_at\_start(Point\ start, Point\ end, Point\ tangent)$
  Diese Methode bekommt anstatt einem dritten Punkt die Tangente am Startpunkt und berechnet daraus den Kreisbogen.

- $derivative\_at(double\ u)$
  Diese Funktion bekommt einen Parameterwert zwischen 0 und 1 und berechnet an dieser Stelle die erste Ableitung.

- $second\_derivative\_at(double\ u)$
  Analog zur vorherigen Funktion wird hier die zweite Ableitung berechnet.

- $get\_curvature()$
  Diese Funktion berechnet die Krümmung der Kurve, die in diesem Fall als der inverse Radius des zum Kreisbogen gehörigen Vollkreises interpretiert wird.

- $point\_at(double\ u)$
  Diese Funktion berechnet den Punkt auf der Kurve an dem Parameterwert u.

- $project\_point(Point\ p)$
  Der gegebene Punkt p wird auf den Kreisbogen projiziert. Es gibt zwei Varianten

dieser Funktion. Eine gibt den Parameterwert zurück, an dem sich der projizierte Punkt befindet, die andere gibt den projizierten Punkt selbst zurück.

- $intersect(Arc\ arc1, Arc\ arc2)$
  Es sind zwei Kreisbögen gegeben und zurückgegeben werden die berechneten Schnittpunkte.

6.2. Neue Funktionalität

- $revert()$
  Diese Methode soll die Orientierung des gegebenen Kreisbogens umdrehen. Dies geschieht durch Vertauschen des Startpunkts P_0 mit dem Endpunkt P_2.

- $tangent\_vector(Point\ p)$
  Diese Funktion bekommt einen Punkt p, projiziert diesen auf den Kreisbogen und berechnet die Ableitung an dem Parameterwert der Projektion. Die Ableitung entspricht der Tangente und wird zurückgegeben.

- $get\_extremal\_points(Point\ p)$
  Diese Funktion liefert die maximal zwei Punkte auf einem Kreisbogen, die extremale Eigenschaft gegenüber einem gegebenen Punkt haben, d.h. entweder am weitesten entfernt oder am nächsten sind. Das Kriterium ist wie bei der Punktprojektion das Skalarprodukt aus dem Richtungsvektor zwischen einem Punkt auf der Kurve und p und der Ableitung an dem Punkt. Das Mittel der Wahl, um diese extremalen Punkte zu finden, ist die Newton Iteration.

- $length()$
  Diese Funktion berechnet die Länge des Kreisbogens. Nach [KA11] bzw. durch geometrische Überlegung lässt sich die Bogenlänge durch $\frac{\alpha\|P_0-P_2\|_2}{\sin(\alpha)}$ berechnen, wobei $\alpha := \arccos(w_1)$ gilt.

- $sample(int\ num)$
  Diese Funktion soll eine Liste von num äquidistant auf dem Kreisbogen verteilten Punkten erzeugen. Dafür wäre eine Bogenlängenparametrisierung vorteilhaft. Wäre die standardisierte rationale Bézierdarstellung eine Bogenlängenparametrisierung des Kreisbogens, so könnte man das Parameterintervall $[0, 1]$ in $num - 1$ gleich lange Intervalle unterteilen und die zu den num Unterteilungspunkten gehörigen Punkte auf dem Kreisbogen würden diesen ebenfalls in $num - 1$ gleich lange Abschnitte unterteilen. In [SFV09] wird jedoch beschrieben, dass nur Strecken durch rationale Funktionen bezüglich der Bogenlänge im $\mathbb{R}^2$ parametrisiert

werden können. Es kann jedoch gezeigt werden, dass die standardisierte rationale Bézierdarstellung von Kreisbögen sehnenlängenparametrisiert ist [Far06]. Für einen beliebigen Parameterwert $u_0 \in [0, 1]$ bedeutet dies:

$$u_0 = \frac{\|\mathfrak{C}(u_0) - P_0\|_2}{\|\mathfrak{C}(u_0) - P_0\|_2 + \|\mathfrak{C}(u_0) - P_2\|_2}$$

Unter anderem bedeutet dies, dass an dem Parameterwert 0.5 auch der Mittelpunkt des Kreisbogens angenommen wird. Besonders für flache Kreisbögen nähert sich die Sehnenlängenparametrisierung der Bogenlängenparametrisierung an. Allgemein ist sie eine gute Näherung. Da die *sample*-Funktion nur für Zeichenzwecke verwendet wird, ist die Sehnenlängenparametrisierung ausreichend. Möchte man eine exakte äquidistante Verteilung haben, so kann dies berechnet werden, ist jedoch mit Aufwand verbunden. In der aktuellen Implementierung geschieht dies durch einen Optimierungsprozess, mit dem Startwert aus der Sehnenlängenparametrisierung und der *length*() Funktion als Hilfsfunktion für die Fehlerbestimmung.

- *set_start*(*Point p*)
  Diese Funktion setzt den Startpunkt P_0 auf den neuen Punkt p. Dabei ändert sich das Kontrolldreieck, d.h. der Punkt P_1 und das Gewicht w_1. Intern wird deshalb der Konstruktor *create_with_third_point*(...) mit den Punkten p, dem alten Endpunkt und dem Punkt $\mathfrak{C}(0.5)$ aufgerufen, wenn p nicht auf dem aktuellen Kreisbogen liegt. Falls sich p auf dem aktuellen Kreisbogen an dem Parameterwert u_p befindet, so wird als dritter Punkt $\mathfrak{C}(\frac{1+u_p}{2})$ verwendet. Die Wahl des dritten Punktes ist darauf bedacht, dass der neue Kreisbogen stets durch drei verschiedene Punkte erzeugt wird und damit definiert ist.

- *set_end*(*Point p*)
  Analog zur vorigen Funktion gibt es diese Methode, die das Ende des Kreisbogens neu setzt.

- *create_full_circle*(*Point start*, *Point end*, *Point between*)
  Dies ist ein Konstruktor, der aus drei gegebenen Punkten mit dem Umweg über die Mittelpunkt-Radius-Darstellung einen Vollkreis erzeugt, der aus drei kongruenten Teilkreisbögen in der rationalen Bézierdarstellung besteht. Das Vorgehen ist dasselbe, wie im *create_with_third_point*-Algorithmus für große Öffnungswinkel.

- *project_point_on_infinite_segment*(*Point p*)
  Diese Funktion berechnet die Projektion des gegeben Punktes p auf den zum Kreisbogen gehörigen Vollkreis. Die prinzipielle Vorgehensweise erzeugt aus drei

Punkten des Kreisbogens einen Vollkreis mit der *create_full_circle*-Methode und projiziert p auf den Kreis mit Hilfe der *project_point*-Methode. Damit ist der große Vorteil der lokalen Kreisbogendarstellung schnell verloren. Wenn ein großer Kreis mit kleinem Öffnungswinkel gegeben ist und die Projektion nahe an dem gegebenen Kreisbogen liegt, so ist es numerisch nicht sinnvoll sofort einen Vollkreis aufzubauen. Um die Lokalität zu konservieren, wird ein benachbarter

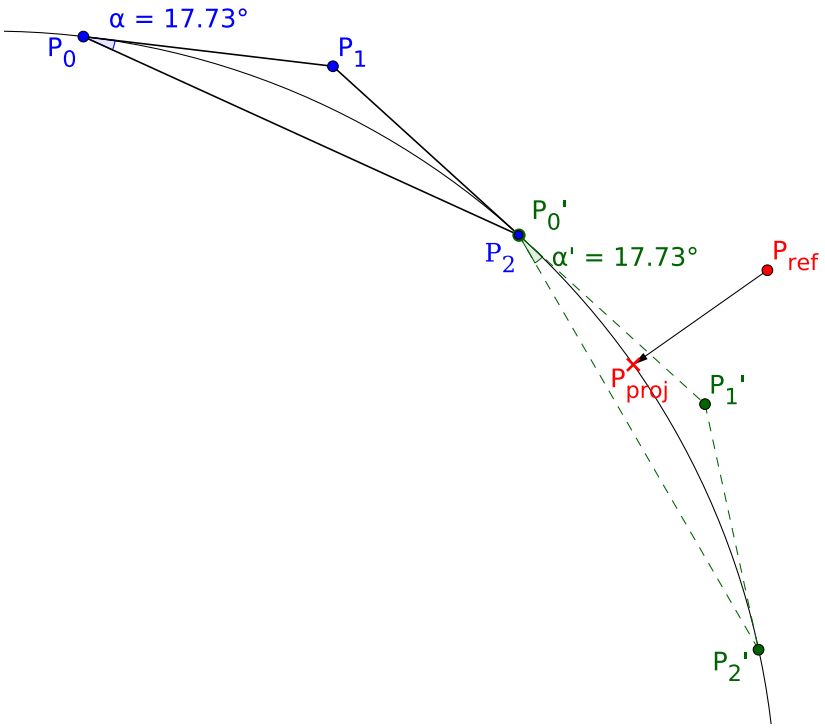

Abb. 27: *Konstruktion eines benachbarten Kreisbogens*

Kreisbogen zu dem Aktuellen erzeugt. Ein benachbarter Kreisbogen, ist ein Kreisbogen, der zu demselben Kreisbogen gehört und der Startpunkt des einen mit dem Endpunkt des anderen zusammenfällt. Soll der benachbarte Kreisbogen erzeugt werden, der an dem Endpunkt des bestehenden Kreisbogens anknüpft, so gilt für die neuen Parameter P_0', P_1', P_2' und w_1' sofort $P_0' = P_2$ und $w_1' = w_1$. P_1' ergibt sich durch $P_0' + (P_0' - P_1) = 2 * P_0' - P_1$. P_2' erhält man, indem man den Vektor $P_1' - P_0'$ um den Winkel $\arccos(w_1')$ dreht, normiert, mit $\|P_2 - P_0\|_2$ multipliziert und zu P_0' addiert. Geometrisch kann diese Konstruktion in Abbildung 27 nachvollzogen werden.

Der Algorithmus zur Punktprojektion auf den Vollkreis läuft wie folgt ab. Zunächst wird *project_point* aufgerufen. Diese gibt einen Parameter u zurück. Ist dieser echt zwischen 0 und 1, so ist $\mathfrak{C}(u)$ die Projektion. Gilt $u = 1$ oder $u = 0$, so kann der Projektionspunkt an einem Endpunkt liegen, aber auch auf dem komplementären Kreisbogen. Ist der Öffnungswinkel 90° oder mehr, so wird ein Vollkreis erzeugt und auf diesen projiziert. Ist der Öffnungswinkel kleiner, so

wird der Nachbarkreisbogen am Start bei $u = 0$ bzw. am Ende für $u = 1$ erzeugt. Anschließend wird der Punkt auf den Nachbarkreisbogen projiziert. Führt dies erneut zu keiner Lösung auf dem Bogen, so werden die beiden Kreisbögen zusammengefasst und mit diesem Kreisbogen der Algorithmus neu gestartet.

6.3. Probleme und Herausforderungen bei der Migration des neuen Codes in die bestehende Implementierung

Mit den eben vorgestellten Methoden und ein paar weiteren Kleineren ist die gesamte Funktionalität abgedeckt, die durch den Kreisbogen mit Mittelpunkt und Radius in der bisherigen Implementierung ebenfalls bedient wurde. Deshalb sollte es nun möglich sein, die RatBez- anstatt der Standard-Implementierung zu verwenden. Dies führte jedoch zunächst zu einigen Problemen.
Ein erstes Problem offenbarte sich in der numerischen Abstimmung. Die Standard-Methode wurde mit dem SMAP-Projekt zusammen aufgebaut, weshalb die Systeme gut aufeinander abgestimmt waren. Die RatBez-Methode wurde isoliert entwickelt und im Nachhinein eingebaut. Darüber hinaus ist diese Darstellung mittels rationaler Bézierkurven eine ganz andere, mit der Folge, dass etwa in der einen Implementierung ein Punkt als auf dem Bogen liegend betrachtet wurde, in der anderen jedoch nicht mehr. Von dieser Art gab es mehrere Beispiele. Diese Herausforderungen sind typisch in einem Softwareprozess und konnten relativ leicht überwunden werden.
Etwas schwieriger wurde es, ein weiteres Problem des Darstellungsunterschieds zu beheben. In der bisherigen Implementierung wurde ein Kreisbogen intern prinzipiell stets durch einen Vollkreis mit zwei Markern für den Start, bzw. das Ende dargestellt. So stellte es auch kein Problem dar, wenn der Start und das Ende auf denselben Punkt fielen. Für die rationale Bézierdarstellung war dies hingegen ein solches, da, wie in *set_start* beschrieben, ein neues Kontrollpolygon aufgespannt werden muss und dieses nur für drei verschiedene Punkte eindeutig definiert ist. Die Lösung bestand darin, den SMAP-Algorithmus dahingehend anzupassen, dass er besser auf die Darstellung des RatBez-Ansatzes ausgerichtet war. Dies leitet zu dem letzten Punkt über, der angesprochen werden sollte.
Bisher bestand kein wesentlicher Unterschied darin, einen Kreisbogen oder den dazugehörigen Vollkreis bei Schnitt- oder Projektionsaufgaben etwa zu verwenden. Deshalb wurde oft anstatt des Kreisbogens der Vollkreis betrachtet. In der rationalen Bézierdarstellung besteht sowohl bezüglich der Ausführungszeit, als auch bezüglich der numerischen Genauigkeit und Stabilität ein Unterschied zwischen dem Kreisbogen und dem dazugehörigen Vollkreis. Darauf wurde in der *project_point_on_infinite_segment*-Methode reagiert, indem zunächst versucht wird, eine lokale Lösung zu finden, wenn der Öffnungswinkel klein ist. Diese Idee muss durch die gesamte Implementierung bis

hin zum Aufrufeverhalten im SMAP-Algorithmus durchgezogen werden, da nur dann der Vorteil des RatBez-Ansatzes voll und ganz ausgeschöpft wird.
Im Rahmen dieser Arbeit wurde dies noch nicht vollends umgesetzt. Es müsste als erster Schritte eine lokale Lösung des Schnittproblems überlegt werden, sodass ein Schnittpunkt nahe dem betrachteten Kreisbogen gefunden wird, ohne den Vollkreis aufzubauen. Aber auch noch weitere Funktionen müssten angepasst werden. Mit der Idee, die in der $project\_point\_on\_infinite\_segment$-Methode verwendet wurde, dürfte jedoch der Grundstein gelegt sein.

7. Resümee

Diese Arbeit begann damit darzustellen, dass Berechnungen auf Kreisen bzw. Kreisbögen eine gewisse absolute Genauigkeit nicht mehr einhalten können, wenn der Radius zu groß wird. Zumindest gilt dies für Kreise, die durch den Mittelpunkt und den Radius dargestellt werden und nach der intuitiven Berechnungsart implementiert sind. Dieses Problem stellte die Herausforderung an diese Arbeit dar. Es sollte ein Weg gefunden werden, für Kreisbögen mit kleinem Öffnungswinkel unabhängig von der Größe des Radius numerisch stabil zu rechnen.
Stellvertretend für die Bemühungen basierend auf der bestehenden Kreisdarstellung und Berechnungsmethode Umformungen zu finden, die diesen numerischen Effekt umgehen, wurde eine auf dem Strahlensatz basierende Methode zur Bestimmung der Projektion eines gegebenen Punktes auf einen Kreisbogen vorgestellt. Dieser Ansatz (Krümmungsmethode) versteht es geschickt unendlich große Radien zu vermeiden. Jedoch konnten trotzdem aus theoretischer Sicht Abhängigkeiten von einem größer werdenden Radius gefunden werden. Dazu kam eine neue Abhängigkeit, nämlich die von dem zu projizierenden Punkt. Dies war Anlass genug, sich damit abzufinden, dass es wohl basierend auf der Mittelpunkt-Radius Darstellung keinen Algorithmus gibt, der bei Kreisberechnungen nicht von dem Radius abhängig ist.
Deshalb wandte man sich einem völlig anderen Ansatz zu. Die Kreisbögen wurden durch rationale Bézierkurven dargestellt. Dazu wurde zunächst in die Theorie von rationalen Bézierkurven eingeführt und hergeleitet, warum und in welcher Form diese Kreisbögen repräsentieren können. Es wurde auf wichtige Eigenschaften, wie die erste und zweite Ableitung, die Krümmung und einige mehr eingegangen. Explizit vorgestellt wurden die Berechnungsmethoden zur Ermittlung der für die Darstellung notwendigen Parameter. Mit diesem Hintergrund wandte man sich der Implementierung der Punktprojektion und der Schnittpunkte zweier Bögen zu. Für die Punktprojektion wurden mit den Ableitungen nur bereits bekannte Eigenschaften ausgenutzt und mittels des Newton-Verfahrens angewandt. Für die Implizitierung, die aus den verschiedenen Möglichkeiten zur Schnittberechnung ausgewählt wurde, musste mehr neues Gedankengut eingeführt und zudem auf weiterführende Literatur verwiesen werden. Am Ende stand aus theoretischer Sicht ein schnelles und numerisch robustes Verfahren.
Numerik ist per Definition mit der Praxis und der Ausführung verbunden. Theoretisch auftretende Schwierigkeiten müssen nicht zwangsläufig in der Praxis zu Problemen führen. Deshalb folgte nach der Theorie um Problemdarstellung, einem krümmungsbasierten und einem auf rationalen Bézierkurven basierenden Ansatz eine ausführlichere Testreihe. Es wurden mehrere Testszenarios vorgestellt. Zunächst wurde die Punktprojektion betrachtet. Dabei fiel auf, dass die Krümmungsmethode zum einen bessere Ergebnisse lieferte als die Standard-Methode, jedoch die theoretisch beobachteten Pro-

blemstellen auch in der Praxis auftraten. Die rationale Béziermethode hingegen konnte bei großen Radien mit kleinem Öffnungswinkel die besten Ergebnisse liefern, teils sogar mit Abstand. Dies gilt auch für die Schnittpunktberechnung. Aber auch für große Öffnungswinkel war die rationale Béziermethode stets mindestens so exakt in seinen Ergebnissen, wie die Standard-Methode. Das heißt, dass es aus funktionaler Sicht nur Vorteile bringt, Kreise und Kreisbögen durch rationale Bézierkurven darzustellen. Der große Nachteil, der damit einhergeht, ist die Ausführungszeit. Die Implementierungen der Punktprojektion und der Schnittpunktbestimmung waren etwa um das 50-fache langsamer als die bisherige Implementierung nach der Kreisdarstellung mit Mittelpunkt und Radius.
Dennoch wurde die rationale Béziermethode in den SMAP-Algorithmus eingebaut, da eine höhere Genauigkeit das entscheidende Kriterium war. Dazu wurden weitere Funktionen vorgestellt, die basierend auf der Darstellung implementiert werden mussten. Dabei wurde auf einige Unterschiede der beiden Kreisdarstellungen hingewiesen und versucht die dadurch entstandenen Probleme darzulegen. Soweit es möglich war, wurden Lösungen vorgestellt und implementiert. Es bleibt jedoch eine Herausforderung, den SMAP-Algorithmus und die rationale Bézierimplementierung so anzupassen, dass die Vorteile der lokalen Genauigkeit am besten ausgenutzt werden können. Damit ist theoretische Arbeit verbunden, die bei der Punktprojektion auf Vollkreise angedeutet wurde, jedoch konsistent auf die gesamte Implementierung ausgeweitet werden muss. Dies führt über diese Arbeit hinaus und ist zudem von der Anwendung abhängig.
Abschließend ist festzuhalten, dass man für Berechnungsmethoden auf dem Kreis unabhängig von der Größe des Radius eine Darstellung benötigt, die auf den Radius verzichtet. Die Modellierung durch rationale Bézierkurven schafft es einen fließenden Übergang von einem Kreisbogen zu einem Geradenstück darzustellen, der stets numerisch stabil ist. Dies ist genau die Anforderung, die es in der Problemstellung am Anfang zu erfüllen galt.

A. Appendix

A.1. Berechnung eines Kreises nach Mittelpunkt-Radius Darstellung aus drei gegebenen Punkten

Die Berechnung eines Kreises aus drei gegebenen Punkten ist grundlegend für den Ansatz mit der Mittelpunkt-Radius Darstellung eines Kreises. Dazu werden die Punkte zunächst so transformiert, dass ein Punkt auf dem Ursprung liegt und ein weiterer auf der x-Achse. Die Transformation geschieht durch eine Drehung und eine Translation und ist vollkommen analog zu der, die zur Konstruktion aus drei Punkten in 4.2.1 angewandt wird.
Ohne Beschränkung der Allgemeinheit kann nun also angenommen werden, dass die Punkte $P_0 = (0, 0)$, $P_1 = (d, 0)$ und $P_2 = (x, y)$ gegeben sind. Wenn y betragsmäßig kleiner einem vorgegebenen ϵ ist, so sind die drei Punkte kollinear und der Kreis entartet zu einer Gerade. Definiere

$$n_1 = \frac{\begin{pmatrix} 0 \\ d \end{pmatrix}}{\left\| \begin{pmatrix} 0 \\ d \end{pmatrix} \right\|_2} \qquad n_2 = \frac{\begin{pmatrix} -y \\ x - d \end{pmatrix}}{\left\| \begin{pmatrix} -y \\ x - d \end{pmatrix} \right\|_2}$$

Der Mittelpunkt M des Kreises ergibt sich durch

$$M = \frac{1}{2}(P_1 + P_2) + \frac{det(0.5 * P_2, n_1)}{-n_2.x * n_1.y} n_2,$$

wobei $det(.,.)$ die Determinantenfunktion ist. Um den Mittelpunkt in der Ausgangslage zu erhalten, muss dieser durch die inverse Transformation zurücktransformiert werden. Der Radius lässt sich durch den Abstand von M zu einem der Punkte errechnen.

A.2. Berechnung eines Kreises nach Mittelpunkt-Radius Darstellung aus zwei gegebenen Punkten und einer Tangente

Um einen Kreis zu erstellen, der durch zwei Punkte und die Tangente an einem der beiden Punkte gegeben ist, wird in der bisherigen Implementierung eine etwas allgemeinere Methode verwendet, die den Kreis bestimmt, der zwei Punkte passiert und eine gegebene Gerade berührt. Die Situation wird zu Beginn in die spezielle Lage transformiert, dass die Gerade die x-Achse bildet und einer der beiden Punkte auf der y-Achse liegt.

Die Lösung des allgemeineren Falles wird in [Rei14] genauer beschrieben. Hier wollen wir uns auf den benötigten Spezialfall beschränken, der in normierter Lage dazu führt, dass der eine Punkt auf der y-Achse zusätzlich am Ursprung und damit auf der Geraden liegt. Die beiden Punkte haben somit die Koordinaten $P_0 = (0, 0)$ und $P_1 = (x, y)$. Liegt P_1 ebenfalls auf der x-Achse, gilt also $|y| < \epsilon$, so wird die Gerade zum Lösungskreis. Ansonsten berechnet sich der Lösungskreis durch den Schnitt der Mittelsenkrechten der beiden Punkte P_0 und P_1 mit der y-Achse. Der Mittelpunkt stellt sich durch

$$M = \begin{pmatrix} 0 \\ \frac{x^2+y^2}{2y} \end{pmatrix}$$

dar. Wie oben gilt, dass dieser Punkt zunächst zurücktransformiert werden muss, um als allgemeine Lösung zu fungieren. Der Radius wird durch die Entfernung von M zu einem der gegebenen Punkte berechnet.

A.3. Schnittpunkte zweier Kreisbögen nach der Mittelpunkt-Radius Darstellung

Im Folgenden wird dargestellt, wie die Berechnung der Schnittpunkte zweier Kreisbögen nach der Mittelpunkt-Radius Darstellung durchgeführt werden kann, bzw. sie in der Referenzimplementierung durchgeführt ist. Gegeben sind zwei Kreise K_1 und K_2 mit den Kenngrößen M_1, r_1, bzw. M_2, r_2. Die Idee besteht darin, die beiden Kreisgleichungen

$$\left\| M_1 - \begin{pmatrix} x \\ y \end{pmatrix} \right\|_2 - r_1 = 0 \quad \text{und} \quad \left\| M_2 - \begin{pmatrix} x \\ y \end{pmatrix} \right\|_2 - r_2 = 0$$

gleich zu setzten und nach x und y aufzulösen. Ohne Beschränkung der Allgemeinheit sei r_1 größer gleich r_2. Definierte:

$$\begin{aligned} \delta &:= M_1 - M_2 \\ n &:= \begin{pmatrix} -\delta.y \\ \delta.x \end{pmatrix} \\ d &= \|M_1 - M_2\|_2 \end{aligned}$$

Dabei stellt n einen Normalenvektor des Differenzvektors der Mittelpunkte dar. Zudem sei ϵ ein Schwellwert für die numerische Null. Wenn $d < \epsilon$ und $|r_1 - r_2| < \epsilon$ gilt, dann sind die Kreise identisch und es würde unendlich viele Schnittpunkte geben. Wenn eine der beiden Bedingungen $|d - r_1 - r_2| < \epsilon$ oder $|d - r_1 + r_2| < \epsilon$ erfüllt ist, so berühren

sich die beiden Kreise und es gibt nur einen Schnittpunkt. Dieser kann durch

$$S = M_1 + r_1 * \frac{\delta}{d}$$

berechnet werden. Gilt entweder $d > r_1 + r_2$ oder $d < r_1 - r_2$ so gibt es keinen Schnittpunkt. In allen anderen Fällen muss es folglich zwei Schnittpunkte geben. Für deren Berechnung werden zwei Hilfsvariablen angelegt:

$$\lambda := \frac{(r_1^2 - r_2^2 + d^2)}{2d}$$
$$\rho = r_1^2 - \lambda^2$$

Das eben berechnete ρ ist die Diskriminante des quadratischen Gleichungssystems und entscheidet über die Anzahl der Lösungen. Nach voriger Fallunterscheidung sollte also $\rho > 0$ gelten. Aus numerischen Gründen muss dies jedoch nicht immer so sein. Deshalb wird noch einmal unterschieden:

$\rho < -\epsilon$:
Kein Schnittpunkt

$|\rho| < \epsilon$:

$$S = M_1 + \frac{\lambda}{d} * \delta$$

$\rho > \epsilon$:

$$S_1 = M_1 + \frac{\lambda}{d} * \delta + \frac{\sqrt{\rho}}{d} * n$$
$$S_2 = M_1 + \frac{\lambda}{d} * \delta - \frac{\sqrt{\rho}}{d} * n$$

Literatur

[BB10] Laurent Busé und Thang Luu Ba. "Matrix-based Implicit Representations of Rational Algebraic Curves and Applications". In: *Computer Aided Geometric Design* 27 (2010), S. 681–699.

[Bus14] Laurent Busé. "Implicit matrix representations of rational Bézier curves and surfaces". In: *Computer-Aided Design* 46 (2014), S. 14–24.

[CD06] Alan Kaylor Cline und Inderjit S. Dhillon. *Computation of the Singular Value Decomposition*. CRC Press, Jan. 2006.

[Far06] Gerald Farin. "Rational quadratic circles are parametrized by chord length". In: *Computer Aided Geometric Design* 23.9 (Dez. 2006), S. 722–724.

[Far83] Gerald Farin. "Algorithms for rational Bézier curves". In: *Computer-Aided Design* 15 (1983), S. 73–77.

[Flo92] M.S. Floater. "Derivatives of rational Bézier curves". In: *Computer Aided Geometric Design* 9 (1992), S. 161–174.

[GLR82] I. Gohberg, P. Lancaster und L. Rodman. *Matrix Polynomials*. Classics in Applied Mathematics. Society for Industrial und Applied Mathematics, 1982.

[IGW09] Philip Irle, Lutz Gröll und Moritz Werling. "Zwei Zugänge zur Projektion auf 2d-Kurven für die Bahnregelung autonomer Fahrzeuge". In: *at - Automatisierungstechnik Methoden und Anwendungen der Steuerungs-, Regelungs- und Informationstechnik* 57 (2009), S. 403–410.

[KA11] Seon-Hong Kim und Young Joon Ahn. "Arc-length estimations for quadratic rational Bézier curves coinciding with arc-length of special shapes". In: *KSIAM* 15 (2011), S. 123–135.

[KM83] P.A. Koparkar und S.P. Mudur. "A new class of algorithms for the processing of parametric curves". In: *Computer-Aided Design* 15 (1983), S. 41–45.

[Lee87] E.T.Y. Lee. "Rational quadratic Bézier representation for conics". In: *Geometric Modeling: Algorithms and New Trends*. Hrsg. von G.E. Farin. Philadelphia: SIAM, 1987, S. 3–19.

[LR80] Jeffrey M. Lane und Richard F. Riesenfeld. "A theoretical development for the computer generation and display of piecewise polynomial surfaces". In: *IEEE Transactions on pattern analysis and machine intelligence* 2 (1980), S. 35–46.

[Mai10] Georg Maier. *Smooth Minimum Arc Paths. Contour Approximation with Smooth Arc Splines*. Deutschland: SHAKER Verlag GmbH, 2010. ISBN: 978-3-8322-9554-7.

[MD92] Dinesh Manocha und James W. Demmel. *Algorithms for Intersecting Parametric and Algebraic Curves*. Techn. Ber. UCB/CSD-92-698. EECS Department, University of California, Berkeley, Aug. 1992. URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/1992/6266.html.

[PT89] Les Piegl und Wayne Tiller. "A menagerie of rational B-spline circles". In: *IEEE Computer Graphics and Applications* 9.5 (1989), S. 48–56.

[PT97] Les Piegl und Wayne Tiller. *The NURBS Book*. 2nd Edition. Springer, 1997.

[Rei14] Günther Reitberger. *Implementierung des Apollonischen Berührproblems*. Techn. Ber. Bachelorarbeit. Universität Passau, 2014.

[Sauer13] Tomas Sauer. *Einführung in die Numerische Mathematik*. Techn. Ber. Vorlesungsskript. Universität Passau, 2013.

[Sed14] Thomas W. Sederberg. *Computer Aided Geometric Design*. Techn. Ber. Vorlesungsskript. Brigham Young University, 2014.

[SFV09] Takis Sakkalis, Rida T. Farouki und Leonid Vaserstein. "Non-existence of rational arc length parameterizations for curves in". In: *Journal of Computational and Applied Mathematics* 228.1 (2009), S. 494–497.

[SN90] Thomas W. Sederberg und Tomoyuki Nishita. "Curve intersection using Bézier clipping". In: *Computer-Aided Design* 22 (1990), S. 538–549.

[SP86] Thomas W. Sederberg und Scott R. Parry. "Comparison of three curve intersection algorithms". In: *Computer-Aided Design* 18 (1986), S. 58–63.

Abbildungsverzeichnis

Tabellenverzeichnis

Eidesstattliche Erklärung

Hiermit versichere ich, Günther Reitberger, an Eides statt, dass ich diese Masterarbeit selbstständig und ohne Benutzung anderer als der ausgewiesenen Quellen und Hilfsmittel angefertigt habe und alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, als solche gekennzeichnet sind.

Ich erkläre weiterhin, dass ich diese Masterarbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt habe.

Passau, den 17. Juni 2016

_____

Günther Reitberger