Universität Passau
Fakultüt für Informatik und Mathematik

Bachelorarbeit

im Studiengang Informatik

zur Erlangung des akademischen Grades
Bachelor of Science

Thema: Evaluation verschiedener Abstandsmaße zur Bewertung von Modellierungsalgorithmen

Autor: Christian Reisinger
Matrikelnr. 64244

Betreuer: Prof. Dr. Tomas Sauer

Abstract

Um Fahrbahnmarkierungen zu modellieren, werden am FORWISS an der Uni Passau verschiedene Algorithmen benutzt. Diese Arbeit beschäftigt sich mit der Frage, wie diese Algorithmen bewertet werden können. Dafür werden mithilfe einer hierfür entwickelten Software Modelle betrachtet, welche auf der gleichen Eingabemenge basieren und mit einem idealen Modell verglichen, welches vom Benutzer manuell definiert werden kann.

Die grundlegende Idee besteht darin, eine Ideallösung zu erstellen und diese ohne großen zusätzlichen Aufwand für beliebig viele Modelle verwenden zu können.

Die Bewertung einzelner Modelle findet mithilfe unterschiedlicher Maße statt, welche in dieser Arbeit evaluiert werden.

Dabei zeigt sich, dass einige Maße wie beispielsweise die minimale oder maximale Abweichung eines Modells zum Ideal zwar leicht zu berechnen sind, jedoch wenig Aussagekraft bezüglich der Modelle besitzen. Um konkrete Fehler zu entdecken werden speziellere Maße benötigt, welche direkt auf bekannte Fehlerquellen prüfen. So werden z.B. Lücken im Modell erkannt. Diese Maße arbeiten zuverlässig und können als eindeutiges Qualitätsmerkmal verwendet werden.

So ermöglichen es diese Software durch die beschriebenen Maße beliebig viele verschiedene Modellierungsalgorithmen schnell und zuverlässig zu bewerten und potentielle Fehler aufzuspüren.

Inhaltsverzeichnis

1 Einleitung

Das Institut für Softwaresysteme in technischen Anwendungen der Informatik (FORWISS) der Universität Passau entwickelt derzeit zusammen mit Ibeo Automotive Systems ein System um Straßenmarkierungen auf eine neue Art zu modellieren. Dafür fährt ein Auto einen Straßenabschnitt entlang und sammelt mithilfe eines Lasers Daten von diesem. Trifft der Laser auf einer Oberfläche auf, so wird er je nach Oberfläche unterschiedlich stark reflektiert. Die Stärke dieser Reflexion wird anschließend gemessen. Der Laser erfasst dabei die gesamte Breite der Fahrbahn in regelmäßigen Intervallen und speichert dazu Punkte mit ihrer Position und dem gemessenen Reflektanzwert. Dadurch kann in der Modellierung zwischen Fahrbahnmarkierungen mit hohem Reflektanzwert und der eigentlichen Fahrbahn mit niedrigem Wert unterschieden werden. Mithilfe verschiedener Algorithmen werden diese Daten anschließend aufbereitet und in ein Computermodell umgerechnet.
Dieses Modell basiert auf Kreisbogensplines (nachfolgend oft durch "Splines" abgekürzt), einer Kombination aus Geradenstücken und Kreisbögen, die mit tangentialer Glätte aufeinanderfolgen. Im Gegensatz zu bisherigen Modellen können so Kurven als tatsächliche Kreis-Abschnitte statt als Annäherung durch Polygonzüge mit vielen kurzen Abschnitten dargestellt werden.
Da sich diese Modellierung noch in der Entwicklung befindet, ist eine objektive Möglichkeit der Qualitätssicherung nötig, um verschiedene Modellierungsalgorithmen miteinander zu vergleichen und zu quantifizieren und mögliche Fehler zu identifizieren. Zusätzlich können beispielsweise entgegenkommende Fahrzeuge, Schmutz oder ähnliches die Messung beeinträchtigen, was sich potentiell negativ auf das berechnete Modell auswirken kann. Diese Arbeit beschäftigt sich mit der Aufgabe, eine schnelle und zuverlässige Methode der Qualitätssicherung bereitzustellen.
Um sie zu lösen wurde eine Software entwickelt, die es ermöglicht auf Basis der durch den Laser gemessenen Punkte eine ideale Modellierung zu approximieren. Dafür kann ein Benutzer Polygonzüge einzeichnen, mit welchen anschließend die errechneten Kreisbogensplines zu bewerten. Diese Bewertung erfolgt mithilfe verschiedener Maße, welche in dieser Arbeit evaluiert werden, um für diese Anwendung aussagefähige Maße zu finden.

2 Software

Um die verschiedenen Herausforderungen bei der Betrachtung und Verarbeitung der Kreisbogensplines zu erläutern wird zunächst ein Überblick über die speziell dafür erstellte Software geben.
Sie wurde auf mit der Java Standard Edition eigens für diese Arbeit entwickelt um ihre speziellen Anforderungen möglichst genau zu erfüllen. Das beinhaltet sowohl die Benutzeroberfläche, die auf höchste Bedienbarkeit und Übersichtlichkeit ausgelegt ist, als auch die Logik, welche es ermöglicht verschiedene Modelle zu verwalten, zu erstellen und miteinander zu vergleichen.
Das Programm ermöglicht es dem Benutzer anhand der gemessenen Punktewolke Polygonzüge zu zeichnen, welche eine Approximation an eine Idealkurve darstellen, anhand derer die berechneten Splines bewertet werden können. Ein Polygonzug ist eine Aneinanderreihung von Strecken, wobei der Endpunkt des Abschnitts i immer der Startpunkt des Abschnitts $i + 1$ darstellt.
Die Idee dahinter ist, dass einmal gezeichnete Polygonzüge mit verschiedenen Kreisbogensplines verglichen werden können um so schnell Aussagen über die zugrundeliegenden Daten treffen zu können.

2.1 Oberfläche

Neben verschiedenen Schaltflächen zum Laden, Speichern, Darstellen und Bearbeiten der Daten befindet sich das Panel mit den Punkten, Polygonzügen und Splines mittig im Fenster, siehe Abbildung 1.
Grüne Kurven zeigen die errechneten Kreisbogensplines. Rote Geraden sind eingezeichnete Polygonzüge, wobei ein ausgewählter Zug blau hinterlegt ist.
Die Darstellung der Punkte kann angepasst werden. So kann durch ein Farbspektrum von rot bis blau die Höhe oder der gemessene Reflektanzwert jedes Punktes dargestellt werden.

Zusätzlich zu der Oberfläche bietet das Programm auch eine Kommandozeilen-Schnittstelle, die es ermöglicht, alle Maße automatisiert berechnen zu lassen.

2.2 Funktionsweise

Vorhandene Punkte, Polygonzüge und Kreisbogensplines können über das Datei-Menü geladen werden. Um Polygonzüge zu zeichnen stehen in der Werkzeugleiste verschiedene Tools zur Verfügung.
Hat der Benutzer Polygonzüge und Splines geladen oder erstellt, kann er durch ein Dialogfenster auswählen, welche Maße er berechnen lassen möchte und mit welcher Genauigkeit dies geschehen soll. Was "Genauigkeit" an dieser Stelle bedeutet, wird

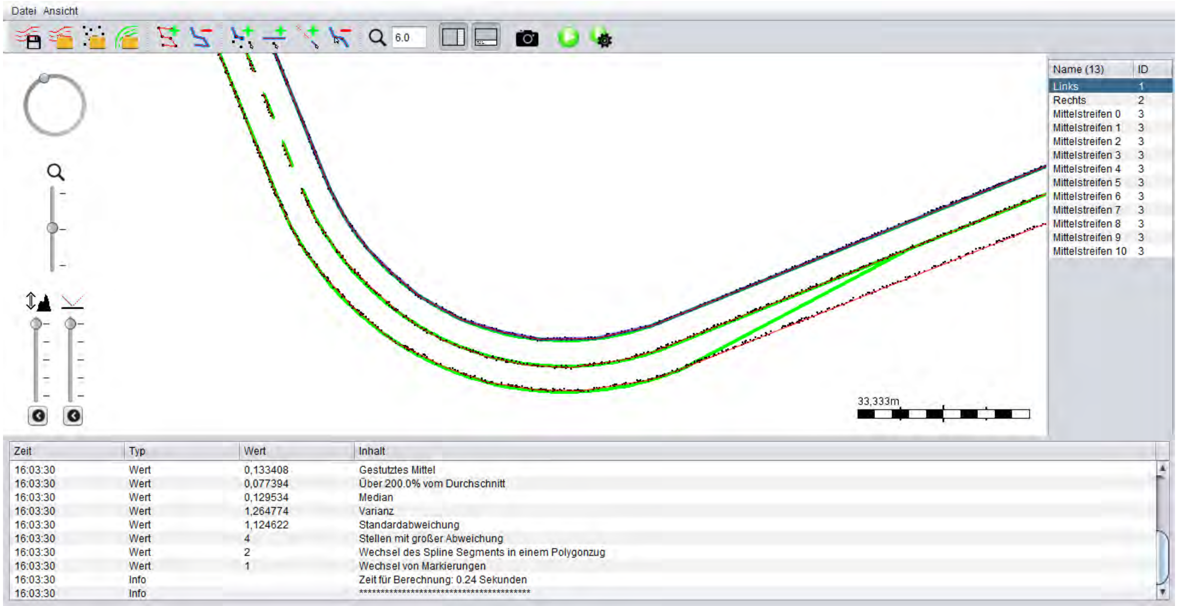


Abbildung 1: Benutzeroberfläche der Software

später im Kapitel "Sampling" (siehe Kapitel 4) näher erläutert.
Die Punktewolke dient einzig als optische Hilfe um die Ideallinie zu zeichnen und hat für die weitere Berechnung keine Bedeutung. Sie kann im Vorfeld z.B. nach Reflektanzwerten gefiltert werden, um das Einzeichnen zu erleichtern.

2.3 Allgemeines

Die Software wurde in der Sprache Java entwickelt und ist damit plattformunabhängig. Das bedeutet, sie kann auf jedem Gerät, auf dem die Java Virtual Machine installiert ist, ausgeführt werden und muss nicht extra selbst installiert werden. Außerdem kann sie so z.B. direkt von einem USB-Stick gestartet werden.
Das Interface wurde mit Priorität auf hohe Benutzerfreundlichkeit entwickelt und bietet zu den Schaltflächen Hilfestellungen und zu allen Buttons aussagekräftige Grafiken an. Die Oberfläche ist so aufgebaut, dass der Benutzer immer das sieht, was er gerade sehen möchte. Das Zeichenfeld nimmt im Zeichenbetrieb den Großteil der Fensterfläche ein. Werden Maße berechnet, so wird dei Log-Tabelle erst eingeblendet. Werte wie Zoom, Drehung oder Filter werden durch verschiedene Regler und Textfelder angezeigt. Die Größe der Oberfläche kann beliebig verändert werden und ist damit an keine bestimmte Hardware gebunden.
Außerdem wurde im Hinblick auf die zukünftige Verwendung der Software besonderer Wert darauf gelegt, dass neue Maße sehr schnell und leicht integriert werden können, ohne dass sich ein Benutzer lange mit dem Programmcode beschäftigen muss.
Alle Maße müssen dafür ein bestimmtes Interface implementieren, welches bestimmte Funktionen fordert. Dazu gehören unter anderem ein Name, ein Kürzel und der Wert des Maßes, welcher von den bereitgestellten Punkten abhängt. Dadurch wird sicher-

gestellt, dass alle Maße untereinander konsistent sind und von einer unveränderten Oberklasse gesteuert werden können.

2.4 Aufgabenspezifisches

Um die Punkte darstellen zu können muss zunächst ihre Position am Bildschirm berechnet werden. Diese wird für das Clipping und für diverse Benutzereingaben benötigt. Dafür wird sie in der Punkte-Klasse gespeichert. Um die ursprünglichen, tatsächlichen Werte der Punkte nicht versehentlich zu korrumpieren, werden diese in einer Unterklasse gekapselt und können nur abgefragt, jedoch nicht überschrieben werden. Dazu gehören die x, y und z Positionen sowie der Reflektanzwert der gemessenen Punkte. Diese Kapselung wird allgemein als Decorator-Pattern (siehe Erich [2]) bezeichnet. Der Höhen- und der Reflektanzwert sind für die Berechnungen dieses Programms irrelevant, sie können dennoch zum Filtern oder Darstellen der Punkte verwendet werden.

2.5 Ablauf

Als Erstes muss eine Punktewolke geladen werden. Dies kann über den entsprechenden Button mit einem daraufhin geöffneten Auswahlfenster geschehen. Anschließend werden vom Benutzer mithilfe verschiedener Tools Polygonzüge eingezeichnet und deren IDs festgelegt.
Zu den Tools gehört eine Werkzeug, welches einen beliebig ausgewählten Punkt an die Stelle des selektierten Polygonzugs setzt, welche am nächsten ist. So können auch nachträglich noch zusätzliche Punkte eingezeichnet werden. Um Mittelmarkierungen leichter zu definieren, gibt es auch hier ein Tool, welches abwechselnd zwei Punkte einzeichnet und dann einen neuen Polygonzug anlegt. Fehlerhaft eingezeichnete Punkte können anschließend wieder herausgelöscht werden. Durch das Drücken der Strg-Taste wird zwischen den Modi "Freies Zeichnen" und "Gelenktes Zeichnen" gewechselt, wobei bei ersterem jeder beliebige Punkt auf dem Zeichenfeld ausgewählt werden kann, bei letzterem können nur Punkte markiert werden, welche zur geladenen Punktewolke gehören.
Die Kreisbogensplines können zu einem beliebigen Zeitpunkt geladen werden, jedoch empfiehlt es sich, dies erst nach dem Zeichnen der Polygonzüge zu tun, da sonst eventuell die Objektivität des Benutzers beeinträchtigt werden könnte.
Sind Kreisbogensplines und Polygonzüge geladen, so können die Maße berechnet werden. Optional dazu kann ein Fenster geöffnet werden, in dem Einstellungen für die nachfolgende Berechnung getroffen werden können.
Nachdem die Maße berechnet wurden, werden ihre Ergebnisse in der Log-Tabelle ausgegeben. Möchte man diese Ergebnisse für einen späteren Zeitpunkt speichern, so können sie als Textdatei gesondert vom restlichen Log-Text exportiert werden. Zusätzlich gibt es eine weitere Schaltfläche um den gesamten Log-Text zu speichern.

3 Clipping

Da die Anzahl der gemessenen Punkte bei längeren Strecken schnell einen zweistelligen Millionenbereich erreicht, reicht es nicht für jeden Punkt seine Position am Bildschirm zu berechnen und ihn dort auszugeben. Die Berechnung von mehreren Millionen Werten würde zu lange dauern, was für den Benutzer in einer stockenden Darstellung enden würden.
Um die Geschwindigkeit der Berechnungen zu erhöhen, werden nur die Positionen derjenigen Punkte berechnet, welche auch dargestellt werden müssen.
Anders als bei gängigen Clipping-Methoden, welche, wie beispielsweise die durch Hudges et al [1] angesprochene Near Clipping-Plane-Methode, darauf abzielen dreidimensionale Modelle zweidimensional darzustellen oder die Grafikkarte zu entlasten, dient diese Art des Clippings primär der Entlastung des Prozessors bei der Berechnung zweidimensionaler Punkte. Da bereits vor der Transformation entschieden werden kann, welche Punkte benötigt werden, muss diese nur noch bei ausgewählten Punkten berechnet werden.
Um die Position eines Punktes zu bestimmen, muss zunächst seine Drehung berechnet werden. Dazu wird seine Position unter anderem mit der Rotationsmatrix (siehe Abbildung 2) multipliziert. θ bezeichnet dabei den Winkel, um den gedreht werden soll, in Radiant.

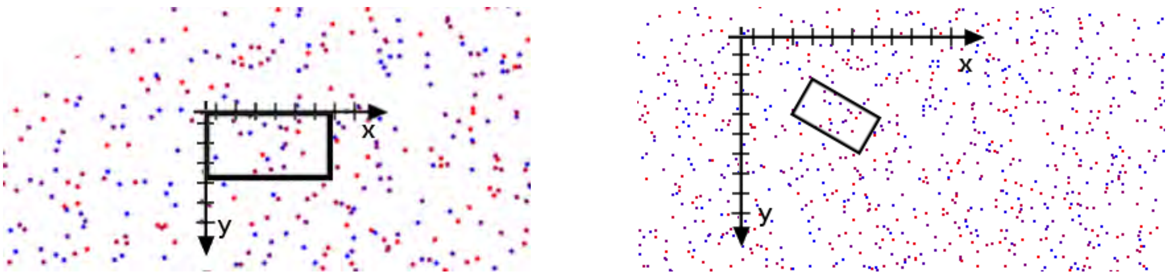
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Abbildung 2: Rotationsmatrix(siehe [3])

Diese Multiplikation enthält mehrere trigonometrische Funktionen, die viel Rechenleistung benötigen. Ein moderner Rechner (im Test: Intel Core i5 2,3 GHz) kann daher eine sehr große Anzahl an Punkten nicht in Echtzeit[1] berechnen. Um dennoch eine flüssige Darstellung ohne Informationsverlust zu gewährleisten, kommt hier das Clipping (vom engl. "to clip" = "abschneiden") zum Einsatz. Dabei wird zunächst berechnet, welche Punkte für die aktuelle Darstellung relevant sind und anschließend nur für diese Punkte ihre Lage im Zeichenfeld berechnet.
Um das zu erreichen, betrachtet man zwei Bezugssysteme. Einerseits im Koordinatensystem der gemessenen Punkte und andererseits im Koordinatensystem des Zeichenfelds.

Um die Position der Punkte im Zeichenfeld (Abbildung 3.a) zu berechnen, dreht man sie zunächst um den sichtbaren Mittelpunkt, skaliert sie anschließend um die aktuelle Zoom-Stufe und verschiebt sie danach an die Position, die der Benutzer betrachten

[1] In einer Geschwindigkeit, sodass die Darstellung für den Benutzer flüssig erscheint



(a) Koordinatensystem des Zeichenfelds (b) Koordinatensystem der Punkte

Abbildung 3: Verschiedene Bezugssysteme für Punkte und Bildschirm

möchte. Also gilt $x' = R(x) \times z + v$, wobei $R(x) = Rotation(\alpha, (bm - v)/z) =$ Die Drehung um den Winkel α um den Punkt, der im sichtbaren Bereich der Mittelpunkt sein wird, z ist der aktuelle Zoomwert, v die Verschiebung entlang der x-Achse und bm der Mittelpunkt des Bildschirms. Die Berechnung für die y-Koordinate verläuft äquivalent.
Um die Position des Bildschirm-Rechtecks (Abbildung 3.b) im Bezugssystem der Punkte zu berechnen, braucht man die Umkehrung dieser Funktion. Damit gilt: $x = R^{-1}(x' - v) \times z$.
Möchte man bei einer Drehung von $\neq 0$ Grad die Position des sichtbaren Bereichs ändern, so muss man den entsprechenden Vektor für die Verschiebung ebenfalls erst mit der Rotationsmatrix multiplizieren. Möchte der Benutzer die Darstellung aus seiner Sicht z.B. um eine Einheit in der horizontalen und null Einheiten in der vertikalen Ebene verschieben, während die Ansicht momentan um 45 Grad gedreht ist, so addiert man nicht (1, 0) zu allen Punkten, sondern (0.707, -0.707). Die Formel für die Verschiebung lautet daher $v_{neu} = (v_{alt} + (\Delta(x'), \Delta(y')) = (v_{alt} + (\Delta(x), \Delta(y)) \times R$, wobei R die Rotationsmatrix ist.
Zunächst berechnet man die Position des Bildschirm-Rechtecks im System der Punkte. Dadurch muss nur alles innerhalb dieses Rechtecks dargestellt werden. Allerdings kann es sehr lange dauern, wenn man für jeden Punkt erst prüft, ob er sich mit diesem Rechteck überschneidet. Daher nimmt man das minimale Rechteck mit achsenparallelen Seiten, welches den Bildschirm einschließt, und kann damit durch triviale größer/kleiner-Abfragen feststellen, welche Punkte zu berechnen sind. Ein mögliches Beispiel dafür zeigt Abbildung 4. Der daraus resultierende Overhead ist geringer als der Aufwand durch das Abfragen jedes Punktes.

Ist die Zoomstufe so klein, dass Punkte übereinander dargestellt werden, so könnte man redundante Punkte verwerfen um Rechenleistung zu sparen. Allerdings ist das Überprüfen, ob Punkte übereinander liegen, z.B. durch Verwendung einer duplikatfreien Struktur aufwändiger, als die Mehrbelastung durch das doppelte Berechnen von gleichen Punkten. Deshalb wurde auf eine solche Implementierung verzichtet.

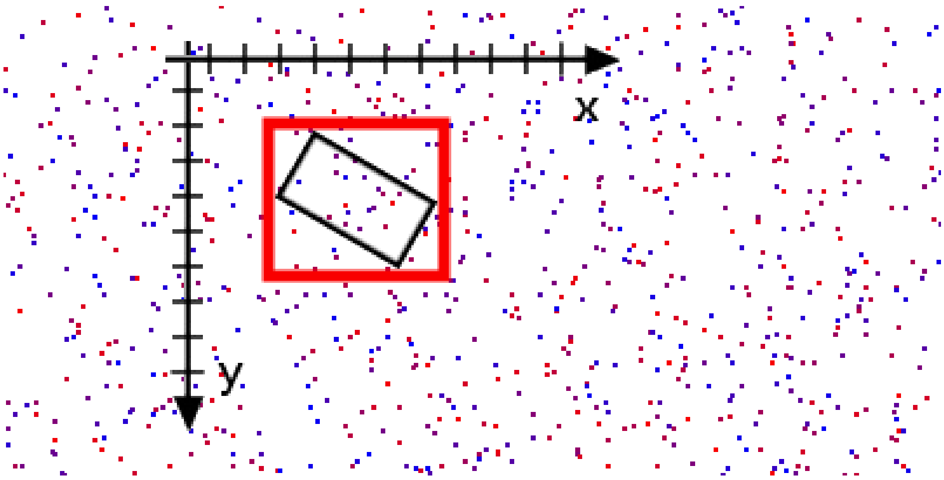


Abbildung 4: Minimal umschließendes Rechteck

4 Sampling

Da man Kreisbogensplines und Polygonzüge nur schwer direkt vergleichen kann, wählt man den Ausweg auf einen bekannten Verlgiech. Dafür benutzt man das sogenannte Sampling (dt. "Abtasten"). Dabei werden Punkte auf einem der beiden Strukturen erzeugt, welche mit der jeweils anderen Struktur verglichen werden können. Da das Berechnen neuer Punkte auf Strecken im Gegensatz zu Kreisabschnitten keine trigonometrischen Funktionen benötigen, können diese von einem Computer deutlich schneller generiert werden. Daher werden die Polygonzüge als Basis für das Sampling verwendet. Möchte man zusätzliche Informationen und Maße auswerten können, wäre es in Zukunft möglich auch die Umkehrung des Samplings zu implementieren. Dadurch könnten beispielsweise Kreisbogenspline-Abschnitte entdeckt werden, welche Stellen modellieren, die nicht modelliert werden sollen. Ein Problem besteht dabei darin, dass diese Maße nur nutzbar sind, wenn die gesamte Punktewolke durch Polygonzüge modelliert ist, was ein Benutzer nicht immer wissen kann. Ist dies nicht der Fall, so werden Fehler erkannt, die keine tatsächlichen Fehler sind.
Dieses Kapitel beschäftigt sich mit der Frage, welche Sampling-Methode auf Basis der Polygonzüge am geeignetsten ist, um korrekte Aussagen über das Verhältnis von idealem zu berechnetem Modell treffen zu können.

4.1 Halbierung von Strecken

Jeder Polygonzug[1] setzt sich aus den eingezeichneten Punkten und den Strecken zwischen diesen zusammen. Eine Option ist es, zunächst alle bekannten Punkte zu übernehmen und anschließend die Stellen zwischen diesen einzuzeichnen.
Dieser Vorgang kann beliebig oft wiederholt werden, um die Anzahl der erzeugten

[1] mit mehr als einem Punkt

Punkte zu erhöhen. Abbildung 5 zeig das Ergebnis am Beispiel-Polygonzug nach zwei Durchläufen.

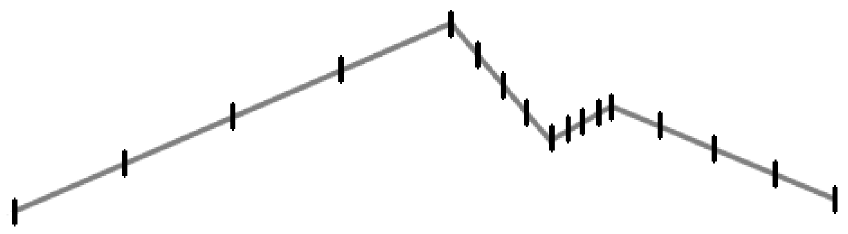


Abbildung 5: Strecken halbieren, 2. Iterationsstufe

Ein Vorteil dieser Methode ist, dass sie sehr schnell berechnet werden kann, da jeder neue Punkt (x, y) direkt aus seinen Nachbarn (x_1, y_1) und (x_2, y_2) berechnet werden kann. Es gilt $(x, y) = ((x_1, y_1) + (x_2, y_2))/2$. Ein weiterer Pluspunkt ist die Tatsache, dass diese Berechnung symmetrisch ist. Das bedeutet, egal ob man beim ersten oder letzten Punkt beginnt, das Ergebnis ist immer gleich. Der Grund dafür ist die kommutative Addition der bekannten Punkte.

Allerdings ist der Nachteil, den man in Abbildung 5 sehr schnell erkennen kann, dass die erzeugten Punkte nicht gleichmäßig auf dem Polygonzug verteilt sind. Das führt dazu, dass Berechnungen auf Basis dieser Punkte an Stellen mit kurzen Segmenten schwerer gewichtet sind, als an langen Abschnitten. Außerdem können ganze Fehler übersehen werden, wenn sich diese zwischen Punkten befinden, die zu weit voneinander entfernt liegen.

Gemäß dem schwachen Gesetz der großen Zahlen (siehe Skript [4]) nähern sich die berechneten Werte der Realität immer näher an, je mehr Werte man hat. Das bedeutet an dieser Stelle, dass sehr wenige Werte im Verhältnis zur Länge eines Abschnittes (z.B. im ersten Abschnitt in Abbildung 5) wenig Aussagekraft über das tatsächliche Verhalten der zugrundeliegenden Geraden haben. Diese Methode ist daher nicht zu gebrauchen.

4.2 Gewichtetes Halbieren

Hierbei werden die Punkte auf die selbe Weise wie in Kapitel 4.1 generiert, allerdings werden spätere Maße zunächst auf einzelnen Abschnitten berechnet und anschließend entsprechend der Länge des Abschnittes gewichtet. Die Gewichtung eines Abschnittes wäre damit $LängedesAbschnitts/Gesamtlänge$.

Diese Erweiterung behebt zwar die Verzerrung durch die ungleiche Verteilung der Punkte, die ungenaue Berechnung bei langen Abschnitten bleibt allerdings. Daher ist auch dieses Verfahren für die angestrebte Verwendung nicht ideal.

4.3 Halbierung der längsten Abschnitte

Ähnlich wie bei Kapitel 4.1 werden hier zunächst alle bekannten Punkte übernommen. Allerdings wird danach nicht jeder Abschnitt halbiert, sondern nur die längsten. Damit erhält man ein gleichmäßiger verteiltes Bild, da sich die Länge der Abschnitte immer mehr angleicht. Die Symmetrie bleibt auch hier erhalten.
Nachteil dieser Methode ist, dass sehr viele Durchläufe benötigt werden um viele kleine Abschnitte zu erzeugen. Außerdem ist für den Benutzer nicht absehbar, wie klein diese Strecken in einer bestimmten Iterationsstufe werden, da er nicht weiß, wie viele neue Punkte in einem Durchlauf erzeugt werden.
Wie Abbildung 6 zeigt, ist das Ergebnis nach einigen Durchläufen schon deutlich gleichmäßiger als bei Methode 4.1. Fügt man nun noch eine Gewichtung der einzelnen Geraden hinzu, so gleicht man gegebenenfalls leichte Abweichungen aus und erhält ein besseres Ergebnis, welches jedoch noch nicht ideal ist.

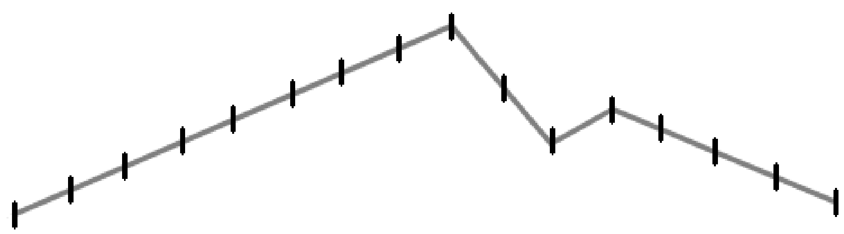


Abbildung 6: Längste Abschnitte halbieren, 6. Iterationsstufe

4.4 Größter gemeinsamer Teiler

Möchte man alle bekannten Abschnitte in exakt gleich große Stücke zerteilen, so liegt der Gedanke nahe, diesen neuen Stücken die Länge des größten gemeinsamen Teilers der ursprünglichen Strecken zu geben. Dieser lässt sich durch den "Euklidischen Algorithmus" berechnen. Der Pseudo-Code (siehe Abbildung 4.4) zeigt die Funktionsweise dieses Algorithmus.

```
groesster GemeinsamerTeiler(Zahl1, Zahl2) {
  solange (Zahl1 > 0 und Zahl2 > 0) {
    falls (Zahl1 > Zahl2) {
      Zahl1 = Zahl1 % Zahl2;
    } sonst {
      Zahl2 = Zahl2 % Zahl1;
    }
  }
  return Zahl1;
}
```

Um diese Funktion nutzen zu können, muss man zunächst alle bekannten Werte auf eine natürliche Zahl skalieren, um das Ergebnis danach zurück zu skalieren. Bei Versuchen mit Testwerten konnte dabei festgestellt werden, dass die Berechnung mehrere Minuten und damit ein vielfaches länger als andere Sampling-Methoden benötigt.
Das Ergebnis ist dabei immer gleich $1/Skalierungsfaktor$, was darauf zurückzuführen ist, dass 1 bei sehr vielen Werten fast immer der größte gemeinsame Teiler ist. Daher ist diese Berechnung überflüssig. Außerdem muss der Skalierungsfaktor sehr groß sein ($> 1 \times E^7$) um nicht zu viel Information zu verlieren. Das Ergebnis ist damit zu klein um verwendet werden zu können, es müssten zu viele Punkte generiert werden.

4.5 Festabstand

4.5.1 Interpretieren von Festabstand

Möchte man Punkte mit einem festen Abstand voneinander erzeugen, so kann dies auf zwei Arten geschehen. Betrachtet man den Polygonzug in seinem zweidimensionalen Raum, so könnte z.B. Abbildung 7 entstehen.

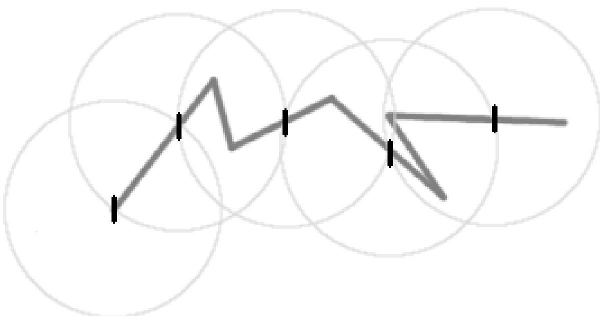


Abbildung 7: Erzeugen von Punkten mit festem Abstand

Es zeigt sich, dass dabei viele Streckenabschnitte ignoriert werden. Außerdem können mehrere Punkte auf dem Polygonzug den definierten Abstand haben. Nimmt man beide dieser Werte zum weiteren Berechnen erhält man inkonsistente Daten.

Die andere Option wäre es, das Polygon eindimensional zu betrachten, sodass jeder Punkt von seinem Nachbarn auf dieser Strecke gleich weit entfernt ist. Wendet man dieses Verfahren auf den Daten von Abbildung 7 an, so erhält man Abbildung 8. Dabei verliert man die x- oder y-Werte nicht, man betrachtet den Abstand nur auf dem Polygonzug. Die generierten Punkte liegen trotzdem tatsächlich auf den ursprünglichen Geradenstücken.

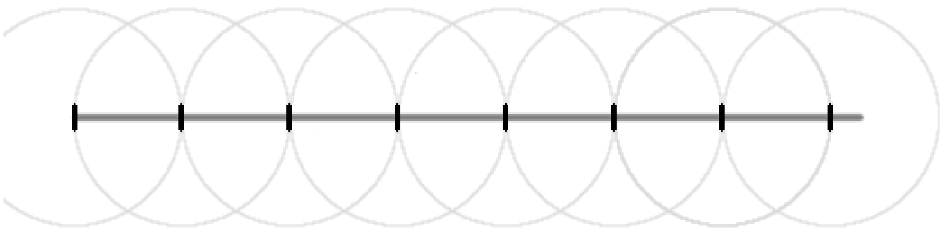


Abbildung 8: Erzeugen von Punkten mit festem Abstand auf dem Polygonzug

Die folgende Methode basiert auf dem Verfahren, welches in Abbildung 8 gezeigt wird, da dieses ein gleichmäßigeres Ergebnis liefert.

4.5.2 Sampling mit Festabstand

Die Idee ist es, einen festen Abstand d zwischen den erzeugten Punkten festzulegen. Dies kann vor jedem Start der Berechnung durch den Benutzer geschehen. Grundlegend gibt es dafür zwei Möglichkeiten. Option 1: Man ignoriert die vom Benutzer eingezeichneten Punkte und startet mit dem Generieren der neuen an einem Ende des Polygonzugs. Option 2: Man übernimmt alle vorhandenen Punkte und generiert immer für einen Abschnitt. Ist kein Platz mehr für einen neuen Punkt, so beginnt man mit dem nächsten.

Der Vorteil der ersten Methode wäre eine sehr gleichmäßige Verteilung der Punkte über die gesamte Strecke, wie in Abbildung 4.5.1 gezeigt wurde. Als Fehler tritt dabei nur das letzte Stück an dem Ende des Polygonzugs auf, an dem man nicht begonnen hat. An dieser Stelle bleibt ein Abschnitt k mit Länge $< d$ ohne Punkt. Um die Genauigkeit noch weiter zu erhöhen, kann man diese Methode mit allen Maßen zwei mal berechnen, einmal beginnt man beim ersten und das andere mal beim letzten Punkt. Damit stellt man sicher, dass alle Abschnitte des Polygonzugs abgesampelt werden.
Ein Nachteil dieser Option ist allerdings, dass bei zweimaligem Durchlaufen eine doppelte Laufzeit benötigt wird, obwohl der Mehrwert verschwindend gering ist. Die Punkte, welche beim zweiten Durchlauf erzeugt werden, weichen jeweils um genau die Länge

k vom ersten Durchgang ab. Ist dieses k sehr nah an 0 oder d, so erzeugt man zwei mal fast identische Punktemengen. Liegt k sehr nah an $d/2$, was eine ideale Verteilung der erzeugten Punkte bedeuten würde, so könnte man direkt d mit halber Größe wählen und alle möglichen schlechten Szenarien ausschließen.
Außerdem können ähnlich wie in Abbildung 4.5.1 ganze Abschnitte übersehen werden, wenn die Schrittweite zu groß gewählt wird. Der Benutzer weiß dabei nicht, ab wann ein d zu groß ist, da er oft den Unterschied zwischen Polygonzug und Kreisbogenspline nicht kennt.

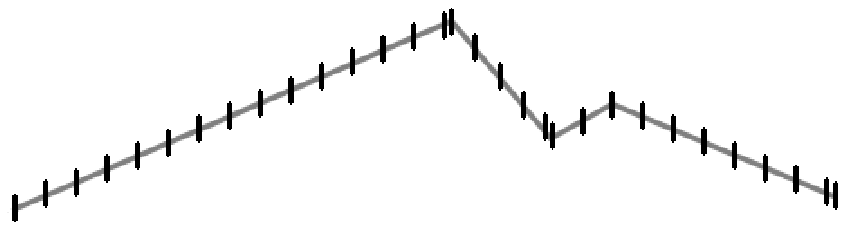


Abbildung 9: Punkte mit festem Abstand

Option 2 ist möglicherweise zunächst weniger intuitiv, da, wie in Abbildung 9 deutlich wird, an jedem Abschnitt-Ende eine ungleiche Verteilung entsteht. Dieser ist im Schnitt $d/2$ groß. Betrachtet man das anhand des Beispiels "Mittelmarkierung", so wird deutlich, dass diese Ungenauigkeit vernachlässigt werden kann. Geht man für den kurven Streifen von einer Länge von $3m$ bei einer hohen Schrittweite d von 10cm aus, so betrifft die Ungenauigkeit $3,33\%$ der Gesamtlänge. Bei der Bewertung der Kreisbögen spielt eine exakte Berechnung bis zur zweiten Nachkommastelle für einzelne Punkte keine Rolle, da beispielsweise durch das manuelle Eintragen der Polygonzüge schon größere Abweichungen entstehen können. Daher ist diese Ungenauigkeit zu vernachlässigen.
Aus demselben Grund ist es auch nicht nötig, die Punkte mit beiden Enden der Abschnitte als jeweiligen Startpunkt zu generieren und die Maße doppelt zu berechnen. Der dadurch erzielte Mehrwert kann vernachlässigt werden, die doppelte Laufzeit ist unnötig.
Ein weiterer möglicher Negativ-Aspekt ist die fehlende Symmetrie dieser Methode. Je nachdem von welcher Seite man die Berechnung startet wird man andere Punkte generieren. Allerdings ist für die verwendeten Maße und deren Zweck diese Symmetrie irrelevant.
Vorteil dieser Erzeugung ist einerseits, dass sie sehr schnell funktioniert. Zunächst wird

die Länge l des zu betrachtenden Abschnittes bestimmt. Anschließend das Verhältnis $r = d/l$. Damit berechnet sich der neue Punkt C auf der Strecke $\overline{AB}$ folgendermaßen: $C = r * B + (1 - r) * A$, wobei $A =$ Startpunkt oder zuletzt generierter Punkt und $B =$ Endpunkt des Abschnitts. Im Test konnten so 1,25 Mio. Punkte in ca. 2,5 Sek. erzeugt werden.
Andererseits werden alle Abschnitte berücksichtigt, da mindestens ihr Start- und Endpunkt verwendet werden. Dennoch bietet diese Methode bei ausreichender Genauigkeit ein gleichmäßiges Bild der erzeugten Punkte. Außerdem ist diese Genauigkeit für den Benutzer intuitiv und verständlich. Es sind keine rekursive Berechnungen oder generierte Abstände nötig, welche beispielsweise durch Halbieren der einzelnen Abschnitte entstehen würden. Der Benutzer definiert den Abstand der erzeugten Punkte und damit die Genauigkeit seiner Berechnung selbst.
Wegen dieser Vorteile und den Nachteilen oben genannter Methoden wurde diese Variante implementiert.

5 Maße

Der Kern dieser Arbeit ist das Bewerten von Kreisbogensplines auf Basis der gesampelten Punkte, die in Kapitel 4 vorgestellt wurden. Dieses Kapitel befasst sich zunächst mit grundlegenden Problemen zum Zusammenhang von Samplingpunkten und Splines und anschließend mit verschiedenen, einfachen und komplexeren Maßen und welche Bedeutung man diesen beimessen kann.

5.1 Berechnen des Abstandes

5.1.1 Grundlegende Überlegung

Um Aussagen über die Kreisbogensplines treffen zu können, benötigt man zunächst den Abstand zwischen den gesampelten Punkten und dem zugehörigen Abschnitt. Das kann man auf zwei Arten tun. Einerseits könnte man jedem Spline-Segment einen Abstand zum nächsten Polygonzug zuweisen, andererseits kann man die Distanz von jedem Punkt zum nächsten Segment verwenden. Da man mit der 2. Variante leichter Aussagen über den gesamten Verlauf der Kreisbogensplines treffen kann, wird sie für diese Arbeit verwendet.
Dabei wird davon ausgegangen, dass der Abschnitt mit dem geringsten Abstand auch der ist, auf den sich ein bestimmter Punkt beziehen soll. In der Praxis ist das zwar nicht immer garantiert, es ist in den meisten Fällen aber eine gute Annäherung.

5.1.2 Abstand zwischen Punkten und Kreissegmenten

Ein Kreisbogenspline setzt sich aus Strecken und Kreisbögen zusammen. Da es in der Java Standard Library für Geometrien bereits eine Funktion zum Berechnen des Abstands zwischen einer Strecke und einem Punkt gibt, kann diese verwendet werden.
Möchte man den Abstand eines Kreisbogens zu einem Punkt berechnen, so reicht es nicht aus, den Abstand zum gesamten Kreis zu berechnen ($Abstand(Mittelpunkt, Punkt) - Radius$). Zum einen erhält man dadurch möglicherweise falsche Werte beim Berechnen des Abstandes. Zum anderen können wegen der Annahme aus Kapitel 5.1.1 Sprünge bei den Segmenten auftreten, auf welche sich die Punkte beziehen. Das würde später bei verschiedenen Maßen zu Problemen führen. Daher braucht man eine andere Herangehensweise an dieses Problem.
Abbildung 10 veranschaulicht das verwendete Verfahren. Hier sind Punkte schwarz, das Kreissegment grün und die möglichen Abstände rot eingezeichnet. M ist der Mittelpunkt des Kreises, S der Start- und E der Endpunkt des Segments. Untersucht wird der Punkt P.

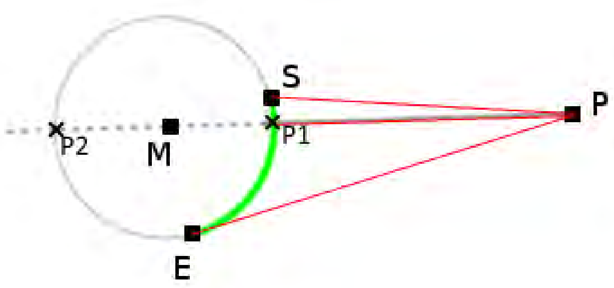


Abbildung 10: Distanz zwischen Punkt und Kreissegment

Die Idee ist es, zunächst die Schnittpunkte P_1 und P_2 der Geraden $\overline{MP}$ mit dem Kreis zu berechnen. Anschließend wird für jeden der Schnittpunkte geprüft, ob er sich auf dem Segment befindet. Da ein Kreisbogen im oder gegen den Uhrzeigersinn verlaufen kann, muss hier differenziert werden. Verläuft er im Uhrzeigersinn, so muss sich ein Schnittpunkt rechts der Geraden $\overline{MS}$ und links der Geraden $\overline{ME}$ befinden. Bei entgegengesetzter Orientierung verhält es sich genau umgekehrt. Beträgt der Winkel SME 180 Grad oder mehr, so reicht es, wenn eine der beiden Bedingungen erfüllt ist. Hat man nun die errechneten Schnittpunkte gefiltert, so ist der minimale Abstand zwischen Kreisbogen und P gleich dem minimalen Abstand von P zu X $\in \{P_1, P_2, S, E\}$.

5.2 Herangehensweise

Zunächst werden zusätzlich zu den eingezeichneten Stellen weitere nach dem Schema 4.5.2 erzeugt. Da viele Maße die Abstände zwischen den Punkten und den Segmenten als Grundlage haben, werden diese vorab berechnet und gespeichert. Außerdem wird für jeden Punkt sein nächstes Spline-Segment vermerkt. Anders herum wird zu jedem Kreisbogenspline eine Liste von Punkten geführt, welche sich auf ihn beziehen. Alle Maße erhalten diese gespeicherten Informationen, weshalb sie nicht manipuliert werden dürfen. Sollen sie aus irgendwelchen Gründen verändert werden, muss zunächst eine Kopie erstellt werden. So wird sichergestellt, dass alle Maße immer auf denselben Werten berechnet werden.

5.3 Simple Maße

Auch ohne großen Aufwand können erste Maße berechnet und damit erste Aussagen getroffen werden. In diesem Abschnitt wird eine Auswahl an solchen Maßen vorgestellt und mögliche Interpretationen aufgezeigt. Das tatsächliche Interpretieren dieser Werte obliegt trotzdem dem Endbenutzer.

5.3.1 Minimum und Maximum

Die offensichtlichsten Maße, wenn es um Abstände geht, sind Minimum und Maximum. Sie können in Laufzeit $O(n)$, einem einzelnen Durchlauf aller berechneten Abstände errechnet werden.
Ein großes Minimum deutet darauf hin, dass die eingezeichneten Polygonzüge entweder nichts mit den verglichenen Kreisbogensplines zu tun haben, oder an einer falschen Stelle im Koordinatensystem gezeichnet wurden.
Ein großes Maximum deutet auf mindestens eine fehlerhafte Stelle hin. Ein kleines Maximum dagegen spricht dafür, dass die Splines fehlerfrei sind.

5.3.2 Durchschnitt

Der Durchschnitt berechnet sich aus $\sum_{i=1}^{n} A(i) \times \frac{1}{n}$ wobei n die Anzahl der berechneten Punkte und A die Menge der Abstände ist. Er wird als Grundlage für weitere Maße verwendet. Er kann auch als grobe Bewertung dienen. Ein geringer Durchschnitt ist in der Regel "besser" als ein hoher. Das ist allerdings keine verlässliche Aussage, da viele genau eingezeichnete Punkte einzelne sehr schlechte Stellen ausgleichen können. Der Durchschnitt sollte daher nicht alleine betrachtet werden um eine verlässliche Aussage zu treffen.

5.3.3 Median

Der Median ist der Wert, der bei der sortierten Liste der Abstände mit Länge n an der $n/2$ten Stelle steht, bzw. an der $n/2 + 1$ten Stelle, falls n gerade ist. Betrachtet man den Median zusammen mit dem Durchschnitt, so kann man Aussagen über die Verteilung der Abstände treffen. Liegt der Median unter dem Durchschnitt, so deutet das darauf hin, dass es nur wenige Punkte mit großem Abstand gibt.

5.3.4 Winsorisiertes Mittel

Um extreme Werte beim Durchschnitt auszublenden, werden beim winsorisierten Mittel, auch gestutztes Mittel genannt, zunächst alle Abstände sortiert und anschließend die oberen und unteren Bereiche entfernt. Welcher Anteil entfernt werden soll kann vom Benutzer gewählt werden. Liegt das gestutzte Mittel unter dem Durchschnitt, kann vermutet werden, dass die hohen Werte extremer ausfallen, als die niedrigen. Im Allgemeinen ist es auch hier schwierig, eine verlässliche Aussage zu treffen.

5.3.5 Prozentuale Abweichung

Für dieses Maß wird der Anteil an Punkten berechnet, deren Abstände über einen bestimmten Prozentsatz vom Durchschnitt abweichen. Damit kann man bestimmen, welchen Anteil die unerwünschten Werte ausmachen. Die Wahl, groß dieser Prozentsatz ist, liegt beim Benutzer. Wählt man den Prozentsatz sehr hoch, so können einige wenige Stellen großer Abweichung gefunden werden, kleinere Abweichungen bleiben unberücksichtigt. Wählt man ihn sehr klein, so können bei großem Durchschnitt eventuell auch "gute" Werte in diesen Anteil fallen.

5.3.6 Varianz

Die Varianz berechnet sich durch die Formel $Var(X) = E(X^2) - (E(X))^2$. $E(X)$ ist der Erwartungswert, siehe Skript Prof. Müller-Gronbach([4]). Er wird definiert durch: $E(X) = \sum_{i=1}^{n} x_i * p_i$, wobei gilt, dass p_i die Wahrscheinlichkeit ist, mit der x_i auftritt. x_i bezeichnet den Abstand des Punktes mit Index i. Durch die Gleichverteilung der Punkte ist die Wahrscheinlichkeit p_i für jeden Punkt gleich $\frac{1}{n}$. Damit ist $E(X)$ gleich dem Durchschnitt (Siehe Kapitel 5.3.2). Gemäß Transformationssatz gilt: $E(X^2) = \sum_{i=1}^{n} x_i^2 * p_i$. Damit ist $E(X^2)$ gleich dem Durchschnitt der quadrierten Einzelwerte.
Die Varianz wird für die Berechnung von Maß 5.3.7 benötigt und kann für mögliche andere mathematische Funktionen verwendet werden.

5.3.7 Standardabweichung

Die Standardabweichung σ ist gleich der Wurzel der Varianz. Sie ist die zu erwartende Abweichung vom Durchschnittswert. Eine hohe Standardabweichung deutet darauf hin, dass es viele Extrema bei den Abständen gibt. Ist sie klein, so kann man eine geringe Streuung um den Durchschnittswert erwarten. Ein geringer Durchschnitt zusammen mit geringem σ deuten auf ein sehr gutes Modell hin.

5.4 Komplexere Maße

Da einzelne Punkte für konkrete Aussagen nicht bedeutend sind und sich aus den oberen Maßen keine direkten Aussagen über Fehler machen lassen, wird in diesem Kapitel ein anderer Ansatz gewählt. Es werden ganze Bereiche gesucht, welche großen Abstand haben um diese anschließend genauer zu betrachten.

5.4.1 Finden von Bereichen mit hohem Abstand

Die bisher vorgestellten Maße betrachten alle Sampling-Punkte und ihre dazugehörigen Abstände unabhängig von den übrigen Punkten. Um weitere Aussagen über das Verhältnis von Polygonzügen zu Kreisbogensplines treffen zu können werden nun mehrere Punkte als zusammenhängender Abschnitt betrachtet.

Um Abschnitte mit hohem Abstand zu finden, gibt es verschiedene Möglichkeiten. Man kann beispielsweise jeden Punkt im Kontext seiner Nachbarn betrachten. Das dauert bei einer großen Anzahl sehr lange. Besser ist es einen Algorithmus zu verwenden, welcher in kürzerer Zeit alle Stellen mit großer Abweichung findet.
Der Algorithmus, der gewählt wurde, benötigt einen Mindestabstand mA Abstand der Punkte sowie eine Mindestlänge mL des Bereichs mit dem Mindestabstand. Diese können vom Benutzer festgelegt werden. Gemäß den Richtlinien für die Standardisierung des Oberbaus von Verkehrsflächen (kurz RStO) müssen alle Fahrstreifen eine Mindestbreite von 2,5m haben. Daraus folgt, dass $mA < 1.25m$ (plus Messtoleranz) gewählt werden sollte, da sonst Wechsel zwischen den Fahrstreifen übersehen werden könnten. Der Wert für mL sollte danach gewählt werden, was der Benutzer als Fehler ansieht. Möchte er kleine Ausschläge verzeihen, so kann mL größer gewählt werden und umgekehrt.
Der Algorithmus sucht zunächst den Punkt mit dem größten Abstand heraus. Liegt dessen Wert über mA, so wird er aus der Liste gelöscht und seine Werte sowie sein Index i vermerkt. Dann wird der neue Punkt an Index i betrachtet. Da man das Element an der Stelle i gelöscht hat, rücken alle darauffolgenden Elemente um 1 nach vorne. Betrachtet man das Element an der Stelle $i+1$, so überspringt man eines. Liegt dieser Punkt über der Grenze, so wird er gelöscht und vermerkt. Das wiederholt man so lange, bis ein Punkt mit einem Abstand kleiner mA auftritt. Anschließend betrachtet man die Liste an der Stelle $i-1$. Liegt der Abstand des Punktes über mA, so wird auch er aus der Liste gelöscht und der Punkt vermerkt. Dann wird Punkt $i-2$ betrachtet usw. Diese Hälfte terminiert ebenfalls, wenn man einen Punkt antrifft, dessen Abstand kleiner als mA ist. Ist die Suche beendet, so betrachtet man alle Abstände zwischen den gemerkten Punkten und addiert sie. So erhält man die Gesamtlänge ihres Abschnittes auf den Polygonzügen. Aus den in Kapitel 4.5.1 angeführten Gründen wird nicht die Differenz zwischen dem letzten und ersten Element betrachtet. Liegt die errechnete Summe über mL, so hat man einen Abschnitt gefunden, der durchgehend einen erhöhten Abstand(größer als mA) besitzt.
Die Punkte werden nach dem Betrachten aus der Liste gelöscht um beim nächsten Durchlauf keine Redundanz zu erhalten.

Abbildung 11 zeigt die Vorgehensweise exemplarisch.Rote Punkte liegen über mA, an den grünen Punkten terminiert jeweils eine Hälfte der Suche. Die schwarze Verbindung zwischen den roten Punkten zeigt die Länge des Abschnittes, welche mit mL verglichen wird.
Sind mA und mL groß genug gewählt um Zeichenfehler und Toleranzen zu ignorieren, so zeigt jeder gefundene Abschnitt eine unerwünschte Stelle im Modell an. Dabei kann es sich um eine Lücke in den Messdaten handeln, welche durch entgegenkommende Fahrzeuge, Verkehrsschilder o.Ä. auftreten kann. Diese sollten vor der Berechnung des

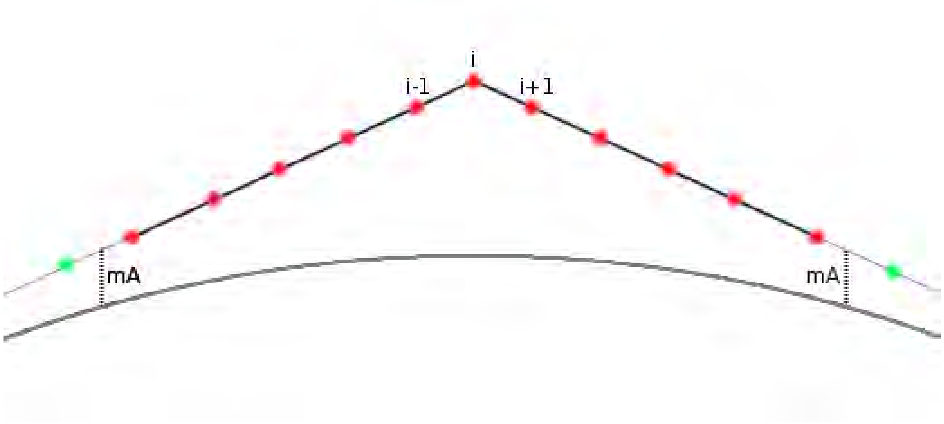


Abbildung 11: Finden von Abschnitten mit hohem Abstand

Modells herausgefiltert und ausgeglichen worden sein.
Andererseits kann es sich um einen Sprung von einer Markierung zu einer anderen handeln. Dabei folgt ein Polygonzug zunächst einer Fahrbahnmarkierung (z.B. einer äußeren) und wechselt in seinem Verlauf zu einer anderen(z.B. einer inneren). Dieses Verhalten ist ebenfalls nicht erwünscht.
Die dritte Möglichkeit für solche Stellen sind generelle Fehler im Spline. Ist das Koordinatensystem verschoben oder folgt er einem falschen Pfad, so können lange Abschnitte mit Fehlern auftreten.
Um diese verschiedenen Varianten unterscheiden zu können, werden die folgenden Maße betrachtet.

5.4.2 Lücken im Kreisbogenspline

Die Fehler, die besonders oft auftreten und auf die man nur schwer Einfluss nehmen kann, sind Lücken in den Messdaten. Besonders durch Gegenverkehr wird die Erfassung der Straßenmarkierungen für eine kurze Zeit unterbrochen. Diese Stellen müssen in der Modellierung kompensiert werden. Geschieht dies nicht richtig oder nicht zuverlässig, können, wie in Abbildung 12 beispielhaft gezeigt wird, Lücken in des Kreisbogensplines (grün eingezeichnet) zurückbleiben.
Wie durch die Punkte und die rote Idealspur angedeutet, sollte die Straßenmarkierung hier weiterhin verlaufen. Ist diese Lücke lange genug, handelt es sich hier um einen Abschnitt mit erhöhtem Abstand1. Er wird von dem Algorithmus von Kapitel 5.4.1 gefunden. Um ihn nun als Lücke zu identifizieren wird zusätzliche Information benötigt.

Jedem zusammengehörigen Kreisbogenspline wird beim SMAP-Algorithmus eine eindeutige ID zugewiesen. Diese ID halten auch alle seine Segmente. Damit kann jedes Segment einer Spur zugewiesen werden.
Dasselbe geschieht nun mit den Polygonzügen. Ihnen wird beim Erstellen eine neue, freie ID zugeteilt. Möchte der Benutzter nun mehrere Polygonzüge zu einem zusam-

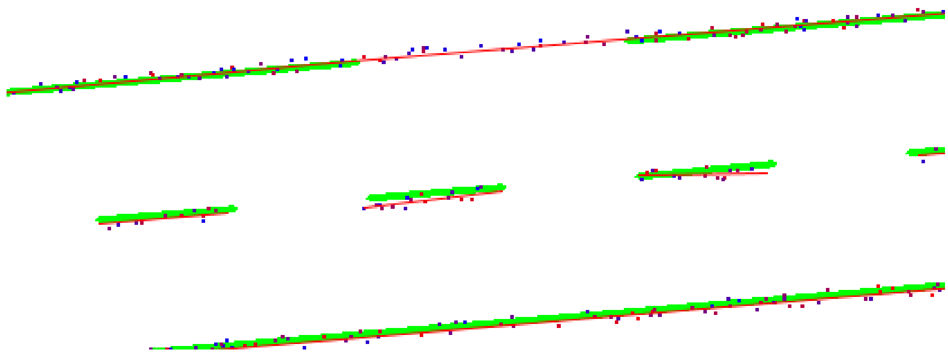


Abbildung 12: Lücke im Kreisbogenspline

menfassen, kann ihr deren ID auf einen gemeinsamen Wert setzen. Das ist z.B. bei Mittelstreifen sinnvoll, da ein einzelner Polygonzug keine Lücken haben kann.
Möchte man nun erkennen, ob es sich bei einem erkannten Abschnitt mit erhöhtem Abstand um eine Lücke in einem Spline handelt, so betrachtet man diese IDs. Dafür durchläuft man alle Punkte, die sich in diesem Bereich befinden. Haben mehrere davon die selbe Polygonzug-ID und beziehen sich auf verschiedene Kreisbogensplines, so hat man eine Lücke gefunden.

5.4.3 Markierungswechsel

Ähnlich wie in Kapitel 5.4.2 basiert auch die Erkennung von Markierungswechseln auf dem Wechsel von IDs. Bei einem solchen Wechsel folgt ein Spline zunächst einem Polygonzug und springt an einer Stelle zu einem anderen. Setzt man mA für den Algorithmus aus Kapitel 5.4.1 kleiner als die Hälfte der Fahrbahnbreite, so schlägt er auch bei diesen Wechseln an. Zu beachten ist hier, dass mL nicht zu lange sein darf, da ein Wechsel über eine kurze Strecke passieren kann. Abbildung 13 zeigt ein Beispiel eines solchen Sprungs.

Um einen Markierungswechsel zu erkennen durchläuft man alle durch das Verfahren aus Kapitel 5.4.1 gefundenen Punkte in dem Abschnitt. Beziehen sich Punkte mit verschiedenen Polygonzug-IDs auf den gleichen Spline, so ist ein Wechsel zwischen den Markierungen gefunden.
Die Abstände der darauffolgenden Punkte werden dennoch weiter mithilfe des nächsten Segments berechnet. Es wird nur der Wechsel der Spur, nicht der weitere Verlauf bestraft.

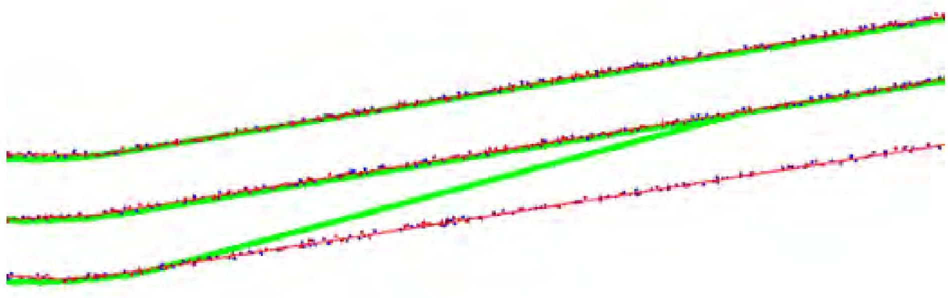


Abbildung 13: Wechsel der Straßenmarkierung, ein Polygonzug(grün) wechselt von der äußeren zur inneren Markierung

5.4.4 Zusätzliches

Zieht man die Zahl der durch gefundenen Lücken und Markierungswechsel erkannten von den insgesamt gefundenen Abschnitten ab, so erhält man eine Zahl an nicht genauer definierten Abschnitten. Diese können verschiedene Ursachen haben und nicht genau unterschieden werden. Werden z.B. Daten verglichen, die nicht auf denselben Punkten basieren, so ist das Verhalten nicht vorhersehbar. Sehr hohe Werte bei diesen Maßen deuten somit auf Benutzerfehler hin.

Möchte der Benutzer die exakten Stellen dieser Abschnitte wissen, so kann er sich diese über den Programm-Log ausgeben lassen. Dadurch können konkrete Fehler schneller gefunden werden.

6 Praktische Anwendung

Dieses Kapitel zeigt eine praktische Anwendung der zuvor genannten Software und zeigt die verwendeten Maße am konkreten Fall.

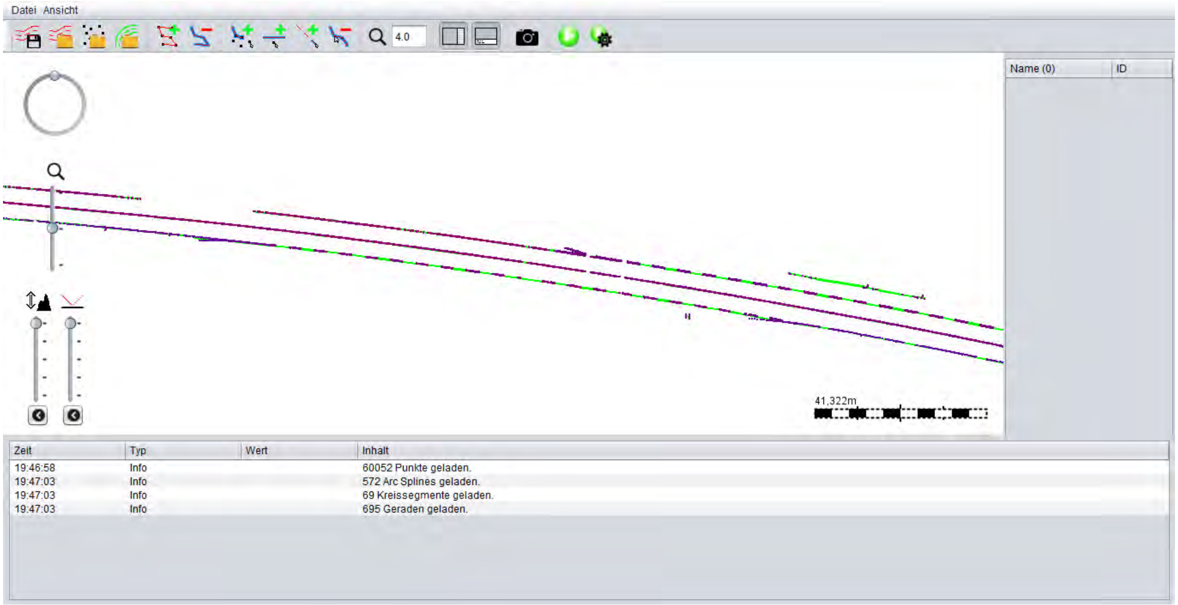


Abbildung 14: Software-Bedienoberfläche

Zu Beginn werden die Punkte und Kreisbogensplines geladen. Dass der Ladevorgang erfolgreich war, zeigt die detaillierte Ausgabe im Log (untere Tabelle) und die Tatsache, dass beides im Zeichenbereich sichtbar ist. Abbildung 14 zeigt dies anhand eines Beispiels von tatsächlich im Laufe des Ibeo-Projekts gemessenen und berechneten Daten.

Mit Hilfe der Werkzeuge in der Werkzeugleiste werden anschließend Polygonzüge eingezeichnet, welche den idealen Verlauf des Modells darstellen. Wie die Oberfläche nach diesem Schritt aussieht, zeigt Abbildung 15. Die rote, gepunktete Linien entsprechen den einzeichneten Polygonzügen.

In der Tabelle rechts im Bild ist zu sehen, dass jeder der eingezeichneten Polygonzüge eine andere ID besitzt, da sie sich auf unterschiedliche Markierungen beziehen.

An diesem Abschnitt ist auffällig, dass die Lücke in den Punkten links, von den grün einzeichneten Kreisbogensplines ebenfalls ausgespart werden. Das ist ein Fehler in der Modellierung. Die Lücke sollte erkannt werden und das Modell den Verlauf der Markierung an dieser Stelle abschätzen. Dieser Fehler muss daher von den verwendeten Maßen erkannt werden.

Da die gezeichneten Polygonzüge für das Sampling verwendet werden, werden auch nur die Abschnitte bewertet, die der Benutzer durch sein Einzeichnen bestimmt. Dadurch können bewusst Stellen ausgelassen werden, welche noch nicht implementierte Logik verlangen würden, oder an denen sehr wenige Messdaten zur Verfügung stehen.

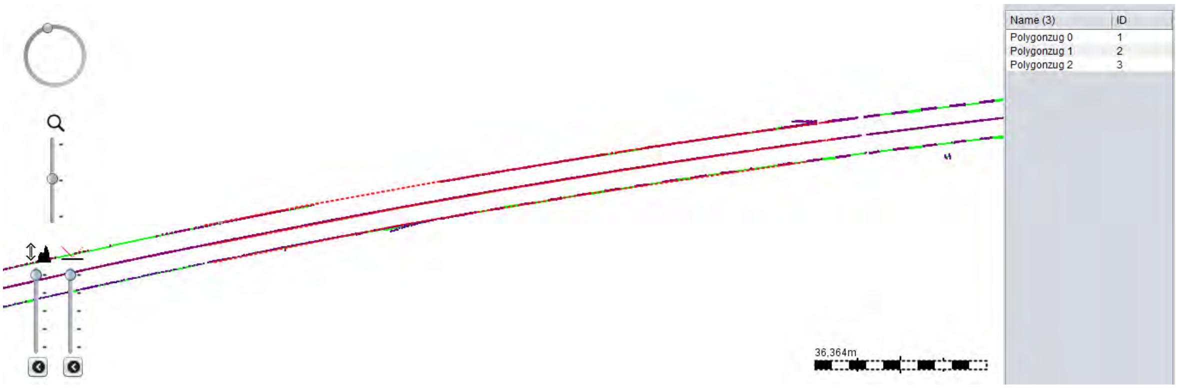


Abbildung 15: Zeichenfeld mit Polygonzügen

In Abbildung 15 ist zu sehen, dass der eingezeichnete Abschnitt mit Ausnahme der Lücke sehr nahe am Ideal liegt. Dies sollte durch die Ergebnisse der Maße widergespiegelt werden.
Abbildung 16 zeigt die berechneten Maße sowie deren Werte. Die Berechnung fand dabei mit einer Genauigkeit von 0.1 statt, was bedeutet, dass alle 10 Zentimeter entlang der eingezeichneten Polygonzüge ein Samplingpunkt generiert wurde. Wie die oberste Zeile in der Tabelle anzeigt wurden für diesen Abschnitt 6391 Punkte generiert. Die unterste Zeile zeigt an, dass die Berechnung der Werte sehr schnell vonstatten ging, was besonders im Hinblick auf häufiges Ausführen von Bewertungen von Vorteil ist.

Wert	Inhalt
6391	Generierte Punkte
4,181476	Maximum
0,000014	Minimum
0,240523	Durchschnitt
0,081121	Gestutztes Mittel
0,044125	Über 200.0% vom Durchschnitt
0,08059	Median
0,535281	Varianz
0,731629	Standardabweichung
1	Stellen mit großer Abweichung
1	Lücken in einer Markierung
0	Wechsel von Markierungen
	Zeit für Berechnung: 0.355 Sekunden

Abbildung 16: Ergebnisse der Maße

Die berechneten Werte bilden exakt das Erwartete ab. So wurde die Lücke im Kreisbogenspline erkannt, ein Spurwechsel jedoch nicht, da die ID des Polygonzugs an dieser Stelle nicht wechselt. Auch sind der Durchschnitt und das Minimum sehr gering, was auf eine sehr genaue Annäherung deutet. Der Median liegt bei etwa 33% des Durch-

schnittswertes, was anzeigt, dass der Großteil der berechneten Strecke einen geringen Abstand zum idealen Verlauf hat, jedoch einige Ausreißer mit erhöhtem Abstand vorhanden sind. Das zeigt auch das Verhältnis von Maximum zu Minimum und das gestutzte Mittel, welches ebenfalls deutlich unter dem Durchschnitt liegt.
Das Fazit dieser Berechnung lautet: Die Modellierung ist bis auf einen deutlichen Fehler sehr gut.
Spielt die benötigte Zeit eine untergeordnete Rolle, so kann man die verwendete Genauigkeit auch erhöhen. Abbildung 17 zeigt die selbe Berechnung wie zuvor, jedoch mit einer Genauigkeit von 0.0005. Das entspricht einem Samplingpunkt pro 0.5 Millimeter. Es wird deutlich, dass eine so hohe Genauigkeit nicht nötig ist um verlässliche Aussagen treffen zu können, da sich die Werte der beiden Berechnungen um höchsten 0.15% unterscheiden während sich die benötigte Zeit auf das 186-Fache erhöht hat. Das zeigt, dass die Software auch bei sehr schnellen Berechnungen genaue Ergebnisse liefert.

Wert	Inhalt
1276278	Generierte Punkte
4,181666	Maximum
0	Minimum
0,240644	Durchschnitt
0,081042	Gestutztes Mittel
0,044106	Über 200.0% vom Durchschnitt
0,080477	Median
0,536056	Varianz
0,732158	Standardabweichung
1	Stellen mit großer Abweichung
1	Lücken in einer Markierung
0	Wechsel von Markierungen
	Zeit für Berechnung: 62.376 Sekunden

Abbildung 17: Ergebnisse der Maße mit hoher Genauigkeit

7 Ausblick

Da die Entwicklung der Modellierung nicht abgeschlossen sind, kann auch diese Software in Zukunft erweitert werden. Ein interessantes Beispiel dafür sind Kreuzungen, welche aktuell nicht berücksichtigt werden. An diesen gelten ganz neue Gesetzmäßigkeiten, auf die man separat prüfen muss. Durch die Struktur des Programms können neue Funktionalitäten ohne besondere Vorkenntnisse und ohne großen Aufwand hinzugefügt werden. Dazu gehören neue Werkzeuge die für neue Herausforderungen optimiert sind oder auch neue Maße, die besonders auf diese neuen Aufgaben eingehen. Bereits vorhandene Maße wie Durchschnitt oder Varianz sind von der Art des Modells unabhängig und können auch für neue Szenarien entsprechen angewandt und interpretiert werden.

8 Danksagung

An dieser Stelle mochte ich mich bei Prof. Dr. Tomas Sauer sowohl für die Betreuung dieser Arbeit als auch für seine Anmerkungen im Laufe der Ausarbeitung und zu meiner Präsentation danken, welche unmittelbaren Einfluss auf diese Arbeit hatten.
Des Weiteren möchte ich mein Dank auch allen Mitarbeitern des FORWISS ausdrücken, welche mir bei verschiedensten Anliegen immer gerne geholfen haben. Ganz besonders möchte ich Christian Pieringer und Peter Barth danken, die den Anstoß zu dieser Arbeit gaben und mich von Anfang an sehr unterstützt haben. Durch ihre Anregungen und Tipps, für die sie sich immer sehr viel Zeit nahmen, standen sie mir während der Ausarbeitung dieser Arbeit stets zur Seite.

Literaturverzeichnis

[1] John Hughes, Andries Van Dam, Morgan McGuire, David Sklar, James Foley Steven Feiner and Kurz Akeley, 2013-07-20, Computer Graphics: Principles and Practice (3rd Edition) ,

[2] Gamma, Erich et al. (1995). Design Patterns. Reading, MA: Addison-Wesley Publishing Co, Inc. pp. 175ff. ISBN 0-201-63361-2.

[3] http://math.stackexchange.com/questions/792302/calculating-translation-value-and-rotation-angle-of-a-rotated-2d-image-using-mat, 2015

[4] Prof. Dr. Thomas Müller-Gronbach, Vorlesungsskript "Einführung in die Stochastik", 2015

Eidesstattliche Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Ort,Datum Unterschrift