Umrissmodellierung - oder der Traum von der SE2

Die ursprüngliche Version 1 Steno Engine (SE1), die inzwischen mit Version 0.1 von VSTENO (Hephaistos) veröffentlicht wurde, verwendet für die Schattierung der Zeichen die so genannte Mittellinienmodellierung. D.h. dickere und dünnere Stellen der Zeichen werden durch Änderung der Strichdicke einzelner Abschnitte umgesetzt. Dies hat den Nachteil, dass die Verdickungen „treppenförmig“ - also nicht „fliessend“ und mit klar erkennbaren Kanten (je nach Vergrösserung) - dargestellt werden. Für die SE2 wurde deshalb die so genannte Umrissmodellierung angedacht:

Mittellinienmodellierung

Umrissmodellierung

Im Folgenden soll nun beschrieben werden, wie die Umrissmodellierung funktioniert, welche Probleme sich dabei stellen und mit welchen Parametern die Modellierung angepasst und verbessert werden kann.

Überlagerung

Die Mittellinienmodellierung funktioniert als Überlagerung, d.h. es wird zuerst, wie bei der SE1, eine - nicht schattierte - Mittellinie generiert und dann die entsprechende Schattierung „darübergelegt“:

Umrissmodellierung = Mittellinie + Schattierung

Man sieht hier das urpsprüngliche Zeichen SP (schwarz) mit der darübergelegten, halbdurch-sichtigen Schattierung (rot).

Die Generierung der ursprünglichen Mittellinie ist notwendig, damit die Verbindungslinien zum und vom Zeichen weg mit der richtigen Spannung (Bezier-Kurve) gezeichnet werden.

Spitz vs rund

Stenografiezeichen können spitzig oder rund beginnen und enden. Bei runden Zeichen - wie dem gezeigten SP - macht es Sinn, dass die Schattierung (Fläche) auf der Mittellinie beginnt und dann fliessend (durch eine Bezier-Kurven-Modellierung) an Breite gewinnt. Spitze Zeichen hingegen müssen bereits im Anfangs- und Endpunkt die volle breite erreichen und spitz modelliert werden. In der SE1 werden solche Zeichen prinzipiell orthogonal modelliert, d.h. der Schatten verläuft als langgezogenes Rechteck parallel zum Schattierten Abschnitt.

Beispiel: „Bad“ in der SE1

Das Zeichen D wird orthogonal als langgezogenes Rechteck modelliert.

Diese Modellierung weist zwei Unschönheiten auf:

(1) die geneigte, rechtwinklige Fläche wirkt im Schriftbild sperrig; ausserdem werden Unter- und Oberlinie in einer nicht näher definierten Weise von je einer Ecke überschritten.

(2) Die Verbindungslinie, die zur Mittellinie verläuft, tritt ebenfalls relativ undefiniert irgendwo in den Schatten ein; und auch hier übertritt ein kleiner, dreieckiger „Spickel“ den Rand des Schattens.

Trapeze

Statt der orthogonalen (Umrissvektor steht senkrecht auf Mittellinie) empfielt sich bei spitzen Zeichen deshalb eine horizontale Modellierung, d.h. der Schatten soll parallel zur Ober- und Unterlinie verlaufen, was also zu einer trapezförmigen Fläche führt. Gleichzeitig soll auch der Ein- und Austritt der Verbindungslinien so korrigiert werden, dass sie sich mit den Eckpunkten verbinden und keine herausschauende Spickel mehr entstehen:

Das schattierte D wird horizontal (trapezförmig) modelliert

Eintritts- und Austrittslinie wurden den Eckpunkten angenähert

Linienübertritte

Im Zusammenhang mit spitzen Zeichen gibt es noch eine dritte Unschönheit. Während runde Zeichen die Ober- und Unterlinie jeweils um die halbe Liniendicke überschreiten (= Mittelpunkt der Linie liegt auf Ober- oder Unterlinie), liegt die Schattierung von spitzen Zeichen nun genau der Linie:

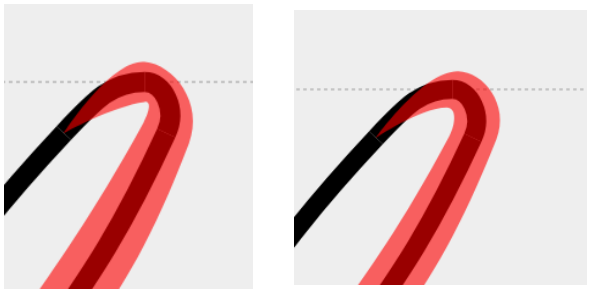
Beispiel „Adam“:
- das Zeichen D übertritt die Ober-/Unterlinie nicht
- das Zeichen M übertritt die Ober-/Unterlinie

Als Folge dieses fehlenden Übertretens wirken spitze Zeichen nun kleiner als runde. Dies kann korrigiert werden, indem spitze Zeichen explizite über die Ober- und/oder Unterlinie hinaus verlängert werden. Hier gleich mehrere Beispiel mit Zeichen die entweder spitz-spitz, spitz-rund oder rund-spitz verlaufen:

„Bade“ (spitz-spitz)
„Wabe“ (spitz-rund)
„Haken“ (rund-spitz)

Normalvektor vs Halbwinkel

Um die Schattierung als Fläche realisieren zu können, muss ausgehend von der Mittellinie zuerst ein Umriss berechnet werden. Hierfür werden - ausgehend von den Punkten, die einzelne Abschnitte der Mittellinie definieren - linke und rechte Aussenpunkte berechnet. Die Berechnung derselben erfolgt prinzipiell orthogonal, d.h. sie wir mit einem Normalvektor vorgenommen, der Senkrecht zur Steigung im Verbindungspunkt der Splines (= Verbindungspunkt zwischen zwei Bezier-Kurven) steht. Diese Methode hat allerdings Nachteile: Bei sehr „engen Kurven" können Knicks entstehen. Dies kann gelöst werden, indem mit der so genannte Halbwinkel-Methode gearbeitet wird, d.h. der Vektor verläuft dann halbwinklig (= in der Mitte) zwischen den beiden Splines:



links: orthogonale Modellierung
=> Knick in oberer Rundung

rechts: Halbwinkel-Modellierung
=> kontinuierliche Rundung

Zwischenpunkte

In der SE1 besteht die Möglichkeit, Zwischenpunkte (intermediate shadow points) zu definieren, um schönere (= weniger treppenförmige) zu generieren. Ein typisches Beispiel ist hier das Zeichen R.

links: schattiertes Anlaut-R ohne Zwischenpunkte (SE1)

rechts: schattiertes Auslaut-R mit Zwischenpunkten (SE1)

Bei der Umrissmodellierung stellt sich die Frage, ob die Punkte übernommen werden sollen. Beides hat Nachteile:

(1) Mit Zwischenpunkten: Die zusätzlichen, für die SE1 gedachten Zwischenpunkte stören u.U. den kontinuierlichen Verlauf der Bezierkurven bei der Umrissmodellierung.

(2) Ohne Zwischenpunkte: Es stehen nur noch sehr wenige (z.B. 3 beim Anlaut-R) Punkte zur Verfügung, um eine schön verlaufende Bezier-Kurve zu generieren.

Das Problem wird zusätzlich dadurch erschwert, dass die Schattierung z.B. beim Anlaut-R im letzten Punkt endet, sodass für den 3. Punkt prinzipiell keine Daten zur Generierung einer sanft verlaufenden Bezier-Kurve mehr zur Verfügung stehen. VSTENO wendet hier zwei Kniffs an, um trotzdem schöne Kurven generieren zu können:

(1) Zirkularität: Wenn das Zeichen „zirkulär" ist (= Anfangs- und Endpunkt fallen zusammen), so wird z.B. beim Anlaut-R der auf den Anfang folgende zweite Punkt als weiterer Hilfspunkt zur Generierung der Bezier-Kurve verwendet.

(2) Startdicke: Wenn nur sehr wenige (hier z.B. 3) Punkte zur Generierung einer Schattenfläche zur Verfügung stehen, so wird die Anfangsdicke (= Dicke des ersten und letzten Umrisspunktes) auf 1.0 (Normalliniendicke) gesetzt.

Durch diese Kniffs ist es möglich, auch bei diesen Zeichen eine einigermassen brauchbare Schattierungsfläche zu berechnen:

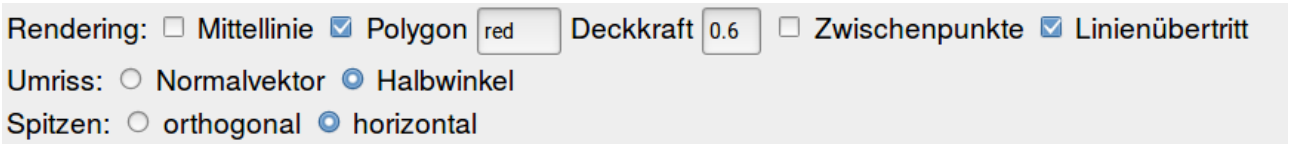
Anlaut-, Auslaut- und Doppel-R Umrissmodellierung unter Verwendung von Zwischenpunkten

Anlaut-, Auslaut- und Doppel-R Umrissmodellierung ohne Zwischenpunkten

Die Resultate ohne Zwischenpunkte sind hier eindeutig besser.

Optionen

Die erörterten Optionen und deren Werte können vom/von der Anwender/in im Eingabeformular angewählt oder angepasst werden:



- Mittellinie: Mittellinienmodellierung nach SE1
- Polygon: Umrissmodellierung (in Anlehnung an die SE2)
- Farbe (Textfeld): Farbe der Schattierungsfläche
- Deckkraft: 1 = 100% deckend, 0<1 = semitransparent, 0 = transparent
- Zwischenpunkt: Zwischenpunkte für Modellierung verwenden oder nicht
- Linienübertritt: Höhe von spitzen Zeichen korrigieren oder nicht
- Normalvektor: Umriss immer mit orthogonalem Vektor berechnen
- Halbwinkel: Umriss so berechnen, dass der Aussenpunkt immer in der Mitte liegt
- orthogonal: spitze Zeichen wie in SE1 berechnen
- horizontal: spitze Zeichen trapezförmig berechnen

Die empfohlenen Einstellungen sind: Polygon, schwarz, 1, ohne Zwischenpunkte, mit Linienübertritt, Halbwinkel, horizontal. Diese Optionen und Werte werden beim Start von VSTENO automatisch gesetzt.

Programmierbar

Zusätzlich besteht die Möglichkeit die Optionen auch via Inline-Options bzw. im Header-Teil des Modells zu setzen:

```
„rendering_middleline_yesno“ := „yes“;
```

Bedeutet z.B. das die Mittellinienmodellierung eingeschaltet wird. Die übrigen Variablen lassen sich mit folgenden Variablen programmieren:

rendering_polygon_yesno, rendering_polygon_color, rendering_vector_type
rendering_sharp_modelling, rendering_polygon_opacity,
rendering_intermediateshadowpoints_yesno, rendering_lineoverpass_yesno

Browser und PDF-Viewer

Die Umriss-Modellierung ist wesentlich komplexer als die Mittellinien-Modellierung der SE1. Es fallen für den/die Betrachter/in am Endbildschirm – Browser, PDF-Viewer, E-Reader - also wesentlich mehr Daten an. Je nach Viewer kann die Qualität der gerenderten Fläche unterschiedlich ausfallen:

- Ältere Viewer oder Geräte mit weniger Rechenleistung und der Standardeinstellung „Geschwindigkeit vor Qualität“ stellen die Flächen unter Umständen treppenförmig dar.

- Neuer Viewer und Geräte mit der Einstellung „Qualität vor Geschwindigkeit“ können damit sehr präzise Ergebnisse erzielen.

Standardmässig setzt VSTENO die Option „geometricPrecision“ (also Qualität vor Geschwindigkeit). Je nach Gerät oder Programm kann diese jedoch – wie erwähnt – besser oder weniger gut umgesetzt werden. In diesem Fall besteht immer die Möglichkeit zur Mittellinien-Modellierung zurückzukehren.

Abschliessende Bemerkungen

Die Umriss-Modellierung war ursprünglich für die SE2 geplant. Diese sollte sogar die Möglichkeit bieten, den genauen Verlauf des Umrisses mit dem grafischen Editor VPAINT präzise zu definieren. Im Laufe der letzten Monate bin ich jedoch zur Überzeugung gelangt, dass die SE2 ein regelrechter Overkill ist: Sie missachtet aus meiner Sicht die 20:80er-Regel (= 20% Aufwand/Komplexität garantieren 80% Erfolg/Präzision) sträflich. Sprich: Sie führt eine Komplexität ein, die der Sache in den meisten Fällen nicht dienlich ist.

Somit erfolgte bei der oben geschilderten Umriss-Modellierung eine Rückbesinnung auf die SE1: eine SE2-ähnliche Qualität sollte erreicht werden, ohne dass dafür auch nur eine Variable der SE1 verändert werden muss. Sprich: Der Algorithmus sollte so intelligent sein, dass er – auch dort, wo die SE1 eine Umriss-Modellierung praktisch unmöglich macht – akzeptable Resulate erreicht. Wie bereits geschildert waren dazu erhebliche Tricks und Kniffs nötig (und viele weitere werden im Hinblick auf die Version 0.2 vermutlich noch nötig sein).

Zum jetztigen Zeitpunkt zeichnen sich somit zwei Dinge ab:

(1) Die SE1 wird dadurch noch einmal bis zur Grenze auf ein Maximum an Funktionalität ausgedehnt. Sicher sind bereits erwähnt Rückportierung der SE2 auf die SE1 (wie parallele Rotationsachsen und proportionale Knots) noch möglich – aber es ist offensichtlich, dass die SE1 irgendwann definitiv an ihr Ende gelangt nach einer Neuentwicklung ruft.

(2) Die SE2 kann kaum die sinnvolle Stossrichtung für eine solche Weiterentwicklung sein. Sie muss zwingend um vereinfachende und effiziente Elemente, wie sie in der SE1 vorkommen (z.B. kombinierte Zeichen) erweitert werden.

Anders gesagt: Die Zukunft liegt vermutlich in einer neuen SE3, welche dann wirklich das Beste aus der SE1 und SE2 vereint. Zusätzlich sollte diese SE3 auch neue Ansätze für Problemstellungen (notabene Zeichenverbindungen bzw. Zeichenabstände) erhalten, welche weder in der SE1 noch in der SE2 gut gelöst wurden.