Softwarepraktikum:
Erkennung von Keilabdrücken in SVG-Dateien

Judith Massa

Ruprecht-Karls-Universität Heidelberg
Interdisziplinäres Zentrum für Wissenschaftliches Rechnen
Forensic Computational Geometry Laboratory

30. Juni 2014

Zusammenfassung

In dieser Arbeit wird eine Möglichkeit vorgestellt, Umzeichnungen von Keilabdrücken, die als einzelne Elemente in einem SVG-Dokument vorliegen, zu einem Element pro Eindruck zusammenzufassen. Die Umzeichnungen bestehen aus drei Kurven für den Keilkopf und oft einer zusätzlichen Linie, die die Hauptrichtungskante eines Keils verlängert. Die Elemente werden zunächst zu kubischen Bézierkurven vereinfacht, wobei für die Berechnung der optimalen Kurve die Methode der kleinsten Quadrate verwendet wird. Die Entscheidung, ob es sich bei einer Kurve um eine Linie handelt, wird durch einen Schwellwert-Verfahren getroffen. Die Zusammengehörigkeit dreier Kurven wird durch räumliche Nähe von berechneten Referenzpunkten der Kurven und anschließender Überprüfung festgestellt. Zugehörige Linien werden gegebenenfalls durch geringe Abstände von Kurven- zu Linienenden ermittelt. Die Elemente der Umzeichnung werden durch Entfernung der mittelpunktnahen Endpunkte zu einer dreiteiligen kubischen Polybézierkurve und dadurch einem einzigen SVG-Objekt zusammengefasst.

Inhaltsverzeichnis

1 Einleitung

Keilschrift entwickelte sich 2700 v.Chr. in Sumer in Südmesopotamien aus stilistischen Piktogrammen. Einer der ältesten Keilschrifttexte ist dabei das berühmte Gilgamesch-Epos [Mau05], das sich über zwölf Tafeln erstreckt. Da Keilschrift einfacher zu schrieben war als die sonst in Vorderasien gebräuchlichen bildhaften Schriften, breitete sie sich bald auf andere Kulturen aus. So gelangte sie etwa 2500 nach Ebla im heutigen Syrien, wo Ausgrabungen ein Palastarchiv mit ca. 6500 Tontafeln offenlegten [Haa04]. Bis zum heutigen Tag wächst die Anzahl der entdeckten Keilschrifttexte im Vorderen Orient kontinuierlich.

Keilschrift-Handzeichnungen liegen digitalisiert in vielen verschiedenen Darstellungsarten vor, unter anderem als 3D-Modelle [MKJB10], Fotos und Umzeichnungen [Neu35] von Fotos in verschiedenen Formaten (Abb. 1). Die Umzeichnungen unterscheiden sich zum einen im Dateiformat der Abbildung, aber auch im Stil der Darstellung der Keile.

(a) (b) (c) (d)

Abbildung 1: Formate für Keilschrifttexte: (a) Screenshot eines 3-D Modells, (b) Fotografie, (c) Umzeichnung als Rastergrafik und (d) Umzeichungen als Vektorgrafik.

Die Aufgabe dieser Arbeit bestand darin, aus den Umzeichnungen von Keilabdrücken im SVG-Format [Eis02] die zu einem Keileindruck gehörenden Teile zu finden und zu einem SVG-Objekt, einer aus drei kubischen Bézierkurven bestehenden Polybézierkurve, zusammenzufassen.

Herausforderungen bildeten folgende Eigenheiten der Darstellung:

- Die einzelnen Kurven, die als kubischen Bézierkurven oder Polybézierkurven vorliegen, können in der SVG-Datei auf verschiedene Arten dargestellt werden. (vgl. Abb. 2)

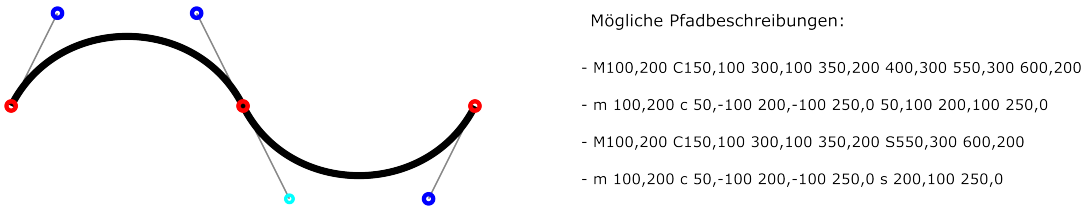
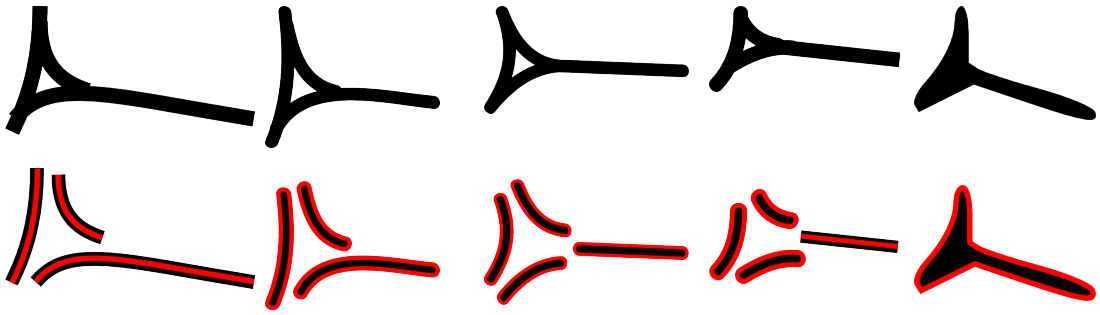


Abbildung 2: Verschiedene Pfadbeschreibungen einer kubischen Polybézierkurve. In rot sind die Endpunkte der einzelnen Bézierkurven eingezeichnet und in blau die Kontrollpunkte. Der hellblaue Kontrollpunkt wird bei Verwendung des *S*-Befehls im *path*-Element automatisch generiert (siehe 1.1).

- Damit die Kurven ansprechend dargestellt werden, sind diese meist ausschließlich statt durch die vier Punkte einer Bézierkurve durch eine Polybézierkurve gespeichert, die den Umriss des Striches beschreibt. Selten ist außer diesem Umriss noch die Original-Bézierkurve gespeichert, die im Grafikprogramm nicht angezeigt wird. (vgl. Tabelle 3, Abb. 3a und 3b)
- Die Keilabdrücke bestehen nicht immer nur aus drei Kurven; in vielen Dokumenten ist der Großteil der Hauptrichtungskante separat zum restlichen Abdruck gezeichnet. Diese meist gerade Linie ist nur zum Teil durch den SVG-Typ *line* realisiert; der Rest ist wie die Kurven durch den Typ *path* dargestellt (vgl. Abb. 3c und 3d). Dies erfordert die Klassifizierung von Pfaden zu Kurven und Linien. Der Pfad von Linien muss außerdem nicht immer eine reine Polybézierkurve beschreiben; manchmal enthält dieser zusätzlich Befehle für Linien.

- Die Pfadbeschreibung ist in manchen Dokumenten in Bezug auf Transformationen angegeben, wobei die Transformationen in übergeordneten Gruppen beschrieben sind.



(a) (b) (c) (d) (e)

Abbildung 3: Verschiedene Arten der Realisierung von Keilabdrücken: (a) mit 3 einfachen Bézierkurven, (b) mit 3 Polybézierkurven, (c) mit 3 Polybézierkurven und einer Linie, (d) mit 3 Polybézierkurven und einer Polybézierkurven als Linie, (e) mit einer Polybézierkurve, gewonnen durch Bitmap-Tracing [Sel03]; dieses Form kann durch den vorgestellen Algorithmus nicht als Keilabdruck erkannt werden.

1.1 Das SVG-Format

SVG-Dateien sind spezielle XML-Dateien [Ray03] mit dem Tag *svg* dessen Unterelemente mithilfe der Angabe von Vektoren unter anderem Linien und geometrische Formen definieren [Eis02]. Als relevant für die bevorstehende Aufgabe stellten sich folgende Tags heraus:

g ein Element, das die Unterelemente zu einer Gruppe zusammenfasst. Attribute in diesem Element haben Auswirkungen auf die Darstellung aller Elemente der Gruppe. Unter anderem kann das Attribut *transform* definiert werden, welches Skalierung, Rotation, Verzerrung und Position der Vektoren, die die Objekte der Kindselemente beschreiben, ändern kann. In den zu betrachtenden Dateien sind die Kurven der Umzeichnungen außerdem in einer Gruppe zusammengefasst. Kann diese zu Beginn identifiziert werden, so können alle anderen Elemente der Datei ignoriert werden ohne das Ergebnis zu verfälschen.

line ein Element, das mindestens vier Attribute enthält: *x1*, *y1*, *x2*, *y2* - diese beschreiben die absoluten Koordinaten der Endpunkte einer Linie.

path ein Element, mit dessen Attribut *d* jede beliebige Form definiert werden kann. Dies geschieht mithilfe der Angabe von absoluten oder relativen Punkten und vorausgehenden Befehls, die die Bedeutung der Punkte festlegen. Die Befehle bestehen im Attribut-String aus einem einzelnen Buchstaben, der groß oder klein vorkommen kann. Kleinbuchstaben sind Indikatoren für relative Punkte, Großbuchstaben weisen auf absolute Punkte hin.

- **M, m** Dieser Befehl ist zu verstehen als das Absetzen und Bewegen eines Stiftes an eine neue Stelle: folgende neue relative Punkte sind davon abhängig. Jede Pfaddefinition fängt mit diesem *moveto*-Befehl an; an dieser Stelle ist es gleichgültig, ob es sich um einen Groß- oder Kleinbuchstaben handelt; folgen jedoch weiter Punkte in einem dieser Kommandos, so wird dies relevant, weil diese als implizite *lineto*-Befehls verstanden werden, die je nachdem relativ oder absolut sind.

- **C, c, S, s** Diese Befehle weisen auf kubische Bézierkurven oder Polybézierkurven hin. Auf jedes *C* folgt die Angabe von drei Punkten: die zwei Kontrollpunkte der Bézierkurve und der Endpunkt. Zum Verständnis der Kurve muss also außer diesen Punkten noch der letzte Punkt des vorausgegangenen Befehles betrachtet werden. Polybézierkurven bestehen somit aus einem *moveto*-Befehl und der Aneinanderkettung von mehreren *C*- oder *S*-Befehlen. Folgen auf ein Buchstaben mehr als die erwartete Anzahl von Punkten, so weist dies auf mehrere Kurven der letzten angegebenen Art hin. *S*-Befehle tauchen nur in Polybézierkurven auf und erwarten zwei Punkte: den zweiten Kontrollpunkt und den Endpunkt - der erste Kontrollpunkt wird berechnet durch die Spiegelung des zweiten Kontrollpunktes der letzten Kurve am Endpunkt der letzten Kurve, welcher gleichzeitig der Anfangspunkt der betrachteten Kurve ist.

- **L, l** wie *M, m* zu verstehen nur ohne Absetzen des Stiftes; dadurch wird eine gerade Linie zum angegebenen Punkt beschrieben.

- **Q, q, T, t** Diese Befehle zeigen quadratische (Poly-)Bézierkurven an und sind äquivalent zu *C, c, S, s* der kubischen Bézierkurven.

- **A, a** zeichnet einen Bogen zum angegebenen Punkt

1.2 Bézierkurven

Da die Möglichkeiten der darstellbaren Kurven beschränkt sind, wenn diese in Form von Funktionen vorliegen, wird bei Vektorgraphiken für gewöhnlich auf die die Darstellung durch Bézierkurven zurückgegriffen.

Definition Bézierkurve [SK11, S. 131]:
Seien für $0 \leq t \leq 1$ die Funktionen $x_k(t)$, $k = 1, 2, \ldots, d$, Bézierpolynome.
Dann ist $(x_1, x_2, \ldots, x_d(t))^T$ eine Bézierkurve im Raum $\mathbb{R}^d$.

Ein Bézierpolynom ist dabei ein Polynom, dass zur Basis von Bernsteinpolynomen dargestellt ist:

$$p(t) = \sum_{i=0}^{n} \beta_i B_{i,n}(t) \qquad t \in [0,1],$$

wobei β_i Bézierkoeffizienten heißen und

$$B_{i,n}(t) := \binom{n}{i}(1-t)^{n-i}t^i \qquad i = 0, 1, \ldots, n$$

die Bernsteinpolynome n-ten Grades in t bezüglich des Intervalls $[0,1]$ sind. [SK11, S. 127]

Für kubische Bézierkurven im zweidimensionalen Raum gilt beispielsweise:

$$\binom{x(t)}{y(t)} = P_{Bezier}(t) = \sum_{i=0}^{3} B_{i,3}C_i = (1-t)^3 \cdot C_0 + (1-t)^2 \cdot t \cdot C_1 + (1-t) \cdot t^2 \cdot C_2 + t^3 \cdot C_3$$

Die Kontrollpunkte $C_i = \left(\beta_i^{(1)}\ \beta_i^{(2)}\right)^T$ haben für beliebigen Grad $n \geq 2$ dabei folgende Eigenschaften:

- $P_{Bezier}(0) = C_0$ und $P_{Bezier}(1) = C_n$
- $P'_{Bezier}(0) = n \cdot (C_1 - C_0)$ und $P'_{Bezier}(1) = n \cdot (C_n - C_{n-1})$
- Die Menge der Punkte einer Bézierkurve $M := \{P_{Bezier}(t) = \sum_{i=0}^{n} B_{i,n}C_i, t \in [0,1]\}$ liegt in der konvexen Hülle (= *Bézierpolygon*) der $n+1$ Kontrollpunkte $C_0, C_1, \ldots, C_n$.

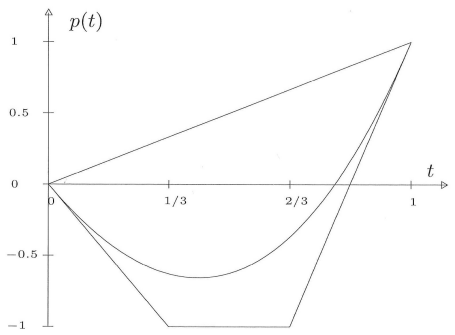


Abbildung 4: Bézierpolygon eines kubischen Bézierpolynoms mit $C_0 = (0\ 0)^T$, $C_1 = (\frac{1}{3}\ -1)^T$, $C_2 = (\frac{2}{3}\ 1)^T$ und $C_3 = (1\ 1)^T$ (Quelle: [SK11])

2 Konvertierung der SVG-Pfade

Da ein Großteil der Kurven durch einfache kubische Bézierkurven dargestellt wird - teilweise indirekt durch Polybézierkurven, die die Originalkurven umgeben - ist der Algorithmus so konzipiert, dass er für eine Kurve vier Punkte entgegennimmt, die die Kontrollpunkte einer kubischen Bézierkurve darstellen. Der Klassifikator, der die Pfade in Kurven und Linien einteilt, nimmt dann die berechneten Kontrollpunkte von Bézierkurven entgegen und gibt für Linien nur die Endpunkte zurück. Somit finden zur Aufbereitung der Daten folgende Umformungen statt:

1. SVG-Pfad $\overset{Pfad-}{\underset{interpreter}{\longrightarrow}}$ Liste mit $(k \cdot 3 + 1)$ absoluten Punkten
2. Punktliste $\overset{Reduktion}{\longrightarrow}$ Kontrollpunkte einer kubischen Bézierkurve
3. Kontrollpunkte der Bézierkurve $\overset{Klassifizierung}{\longrightarrow}$ 4 Punkte für Kurven, 2 Punkte für Linien

2.1 Interpretation des SVG-Pfades

Bei der Interpretation des Pfades werden folgende Umformungen durchgeführt:

- Auflösung der expliziten (L/l) und impliziten (M/m) *lineto*-Befehls durch Verdreifachung des letzten Punktes
- Auflösung des *smoothcurveto*-Befehls (S/s) durch Berechnung des ersten Kontrollpunktes
- Berechnung der absoluten Punkte
- Auflösung der Transformationen

Keine dieser Maßnahmen beeinflussen das Aussehen der Kurve; sie dienen alleine dem Zweck die resultierenden Punktlisten in ein einheitliches Format zu bringen mit dem weitergearbeitet werden kann ohne die Zeichenbefehle des Pfades oder die Transformationsbefehle von übergeordneten Gruppen mit übergeben zu müssen.

Nach Durchführung der Interpretation liegt die Kurve als Punktliste mit $(k \cdot 3 + 1)$ absoluten Punkten vor, die als reine kubische Polybézier-Punktliste verstanden werden muss.

Beispiele:

- <path d="M 100,200 L 250,200" >
  ⇒ [[100 200] [250 200] [250 200] [250 200]]
- <path d="M 100,200 C 100,100 250,100 250,200 S 400,300 400,200" >
  ⇒ [[100 200] [100 100] [250 100] [250 200] [250 300] [400 300] [400 200]]
- <path d="M 100,200 c 0,-100 150,-100 150,0 c 0,100 150,100 150,0" >
  ⇒ [[100 200] [100 100] [250 100] [250 200] [250 300] [400 300] [400 200]]
- <g transform="translate(100,200)" >
  <path d="M 0,0 C 0,-100 150,-100 150,0 C 150,100 300,100 300,0" >
  </g>
  ⇒ [[100 200] [100 100] [250 100] [250 200] [250 300] [400 300] [400 200]]

2.2 Reduktion von Polybézier-Punktlisten

Für Kurven müssen die vier Kontrollpunkte einer kubischen Bézierkurve gefunden werden, die die durch den Pfad beschriebene Kurve am besten approximiert. Da die Polybézierkurve aus zwei Hauptkurven besteht - die an der Innenseite entlang führenden und die äußere begrenzende Kurve, bietet es sich an, das Mittel dieser Kurven wählen (Abb. 5).

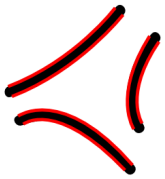


Abbildung 5: Formgebende Teile der Polybézierkurve

Bestehen beide Hauptkurven aus einfachen kubischen Bézierkurven, so lässt sich diese Aufgabe effektiv durch einen einfachen k-Means-Algorithmus lösen, der die Punktliste in vier Cluster gruppiert (Abb. 6). Bestehen die Hauptkurven aber nicht aus einfachen Bézierkurven oder die Punkte liegen zu nahe beieinander, spiegeln die berechneten Zentren nicht die Mittelpunkte der Kontrollpunkte der umschließenden Kurven wider und der Ansatz führt zu fehlerhaft rekonstruierten Kurven (Abb. 7a).

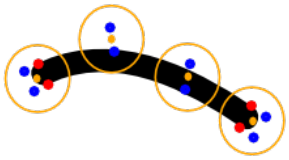
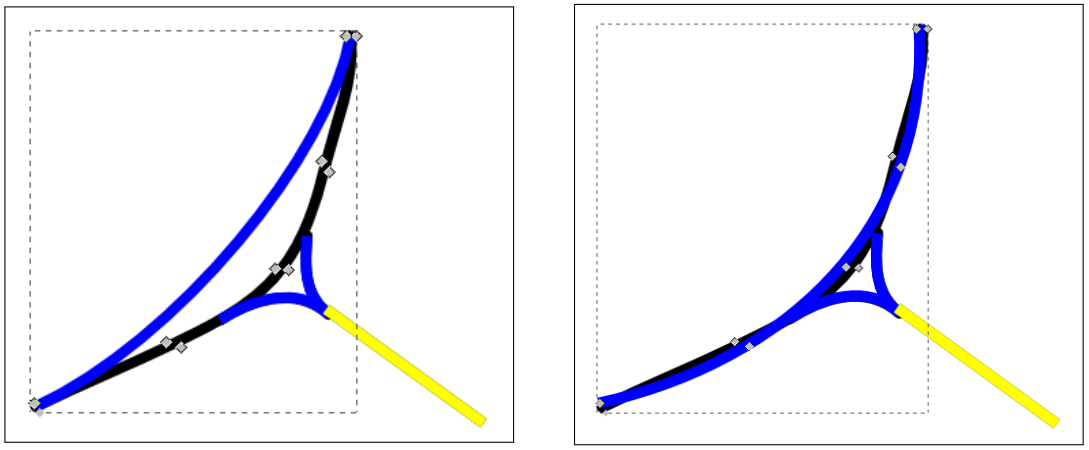


Abbildung 6: Idealfall von k-Means auf Punktlisten: Cluster und Clusterzentren

Eine bessere Lösung erhält man in dem Fall durch das Approximieren der Hauptkurven durch die Methode der kleinsten Quadrate, wobei der Durchschnitt der Kurven als Ergebnis genommen wird (Abb. 7b):

$$C_0 = \frac{C_0^{(1)} + C_0^{(2)}}{2} \qquad C_1 = \frac{C_1^{(1)} + C_1^{(2)}}{2} \qquad C_2 = \frac{C_2^{(1)} + C_2^{(2)}}{2} \qquad C_3 = \frac{C_3^{(1)} + C_3^{(2)}}{2}$$



(a) (b)

Abbildung 7: Rekonstruktion der Kurven (blau) und Linien (gelb) eines Keiles: (a) mit dem k-Means Algorithmus (k=4) und (b) mit der Methode der kleinsten Quadrate. In diesem Fall versagt k-Means da die Hauptkurven selbst aus Polybézierkurven mit vier Segmenten bestehen.

Dafür werden für jede Hauptkurve zunächst für alle zugehörigen Segmente der Polybézierkurve Punkte berechnet, sodass eine geordnete Reihe von Punkte auf der Kurve vorliegen. Es werden nun jeweils die Lageparameter und Kontrollpunkte einer kubischen Bézierkurve gesucht, die den Abstand zu diesen Punkten minimieren:

$$\arg\min_{t_i,C} \sum_i \|P_{Bezier}(t_i; C) - P_i\|, \tag{2.1}$$

wobei

- $C = (C_0, C_1, C_2, C_3)^T \ldots$ Vektor der Kontrollpunkte der Kurve
- $t_i \ldots$ Lageparameter des i-ten Punkes auf der gesuchten Bézierkurve
- $P_{Bezier}(t_i, C) = (1-t_i)^3 \cdot C_0 + (1-t_i)^2 t_i \cdot C_1 + (1-t_i) t_i^2 \cdot C_2 + t_i^3 \cdot C_3$
- $P_0, \ldots, P_n$ die berechneten geordneten Punkte

Mit den unbekannten Parametern t_i ist dies ein nicht-lineares Ausgleichsproblem, welches nur durch aufwändige Konvergenzverfahren lösbar ist [SK11, 296-304]. Das Problem lässt sich aber vereinfachen, indem eine Approximation des Parameters t_i für jeden Punkt berechnet wird. Dies geschieht durch die Miteinbeziehung der Punkt-zu-Punkt-Abstände auf der Kurve:

$$\begin{aligned} t_0 &= 0 \\ t_i &= \frac{\sum_{j=1}^{i}(P_j - P_{j-1})}{\sum_{j=1}^{n}(P_j - P_{j-1})} \qquad \forall\, i = 1, \ldots n \end{aligned} \tag{2.2}$$

Das verbleibende Problem ist nun linear,

$$\arg\min_C (\mathbf{B}C - \mathbf{P}) \tag{2.3}$$

mit

- $\mathbf{P} = [P_i]_{i=0\ldots n}$
- $\mathbf{B} = [B_{j,3}(t_i)]_{\substack{i=0\ldots n \\ j=0\ldots 3}}$,
- $B_{j,3}(t_i) := \binom{3}{j}(1-t_i)^{3-j} t_i^j \ldots j$-tes Bernsteinpolynom dritten Grades in t_i,

und lässt sich mit der Methode der kleinsten Quadrate lösen [SK11, 274-276]:

$$\arg\min_C (\mathbf{B}C - \mathbf{P}) = \arg\min_C (\mathbf{B}C - \mathbf{P})(\mathbf{B}C - \mathbf{P})^T \tag{2.4}$$

Da ein Extrempunkt gesucht ist, muss gelten:

$$\frac{\partial \left((\mathbf{B}C - \mathbf{P})(\mathbf{B}C - \mathbf{P})^T\right)}{\partial C} = 2\mathbf{B}^T\mathbf{B}C - 2\mathbf{B}^T\mathbf{P} = 0 \tag{2.5}$$

$$\Rightarrow \qquad \hat{C} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}$$

2.3 Klassifizierung von Kurven

Ist es nötig, bei den Punktlisten zu unterscheiden, ob sie den Pfad einer Kurve oder einer Linie beschreiben, so muss eine Klassifizierung durchgeführt werden. In diesem Fall reicht es, die grob approximierte Länge des Pfades mit der Länge der rekonstruierten Linie zu vergleichen:

$$|\underbrace{\|C_1 - C_0\|_2 + \|C_2 - C_1\|_2 + \|C_3 - C_2\|_2}_{approximierter\,Pfad} - \underbrace{\|C_3 - C_0\|_2}_{Linie}| \overset{?}{<} T, \qquad (T > 0) \tag{2.6}$$

Da sich die Rekonstruktion von Linen aus deren Umriss als verlässlich herausstellt, kann in den meisten Fällen T beliebig klein gewählt werden. In einzelnen Fällen kommt es jedoch vor, dass dem Umriss eines Pfades, der Teil einer Hauptrichtungskante ist, keine gerade Linie sondern eine Kurve zugrundeliegt. In diesen Fällen muss bei der Wahl des Parameters T darauf geachtet werden, dass dieser nicht zu klein gewählt wird, da sonst Hauptrichtungskanten übersehen werden, aber auch nicht zu groß, da sonst unter Umständen einzelne Kurven nicht als Teil des restlichen Keilabdruckes erkannt werden und so der gesamte Abdruck nicht rekonstruiert wird.

Wird ein Pfad als Linie erkannt, so können die zwei inneren Kontrollpunkte der berechneten Bézierkurve ignoriert werden und es werden nur die Endpunkte zurückgegeben.

3 Matching von Keilabdrücken

Die Herausforderung bei der Suche nach den drei zu einem Keil gehörenden Kurven ist die Größe des Suchraums. Ein Beispiel: In einer Keilschrifttafel sind etwa 500 Keilabdrücke. Für die SVG-Datei bedeutet das im Idealfall, dass in der entsprechenden Schicht 1500 Kurven beschrieben sind. Für den Suchraum bedeutet das $\binom{1500}{3} \approx 3{,}3$ Mrd. mögliche Keilabdrücke, die sich daraus zusammensetzen lassen. Es muss also eine Reduktion des Suchraums stattfinden, ohne dass dadurch ein Keilabdruck übersehen wird.

3.1 Einschränkung der Suche nach zusammengehörenden Kurven

Der intuitive Ansatz ist, nur Kurven zu betrachten, die möglichst nahe zusammenliegen. Dazu muss zunächst ein Punkt berechnet werden, der die Lage der jeweiligen Kurve beschreibt. Eine Möglichkeit dafür ist, das Mittel der Kontrollpunkte der Bézierkurve zu wählen:

$$R_M = \frac{1}{4}\sum_{j=0}^{3} C_j \tag{3.1}$$

Allerdings muss dieser nicht unbedingt nahe an den entsprechenden Punkten der mit zum selben Keil gehörenden Kurven liegen, wenn die Längen der Kanten des Abdruckes erheblich voneinander abweichen. Eine andere sinnvolle Wahl wäre, nur das Mittel der zwei innereren Kontrollpunkte zu nehmen, was unter Umständen den Referenzpunkt mehr ins Zentrum des Abdrucks rückt. Es gibt aber keine Garantie dafür, dass diese tatsächlich nahe an der Kurve liegen, wenn die Kurve entsprechent gekrümmt ist. Eine weitere Möglichkeit, die sich in den meisten Fällen als eine sehr gute Wahl herausstellt, ist es, den Schnittpunkt der Geraden zu wählen, die durch jeweils den ersten und zweiten und den dritten und vierten Punkt verläuft:

$$[C_1 - C_0,\, C_2 - C_3] \cdot R_S = C_2 - C_0 \tag{3.2}$$

Dieser Ansatz kann aber auch fehlschlagen, wenn diese Geraden etwa parallel verlaufen. In diesen Fällen wird auf den Mittelpunkt zurückgegriffen (Abb. 8).

Die erste Strategie ist nun, zunächst einmal für jede Kurve die zwei naheliegensten Kurven zu berechnen. Stimmt dies bei jeweils drei Kurven überein, so ist es sehr wahrscheinlich, dass diese zum gleichen Keil gehören, und die Zusammengehörigkeit wird überprüft. Diese Strategie wird mit den verbleibenden Kurven so oft wiederholt, bis keine neuen Keile entdeckt werden.

Für die zweite Strategie wird die maximale Distanz der Kurven von den schon entdeckten Keilabdrücken verwendet um den Suchraum einzuschränken. Da außerdem die Anzahl der zu betrachtenden Kurven mittlerweile stark gesunken ist, lassen sich weitere Keile in angemessener Zeit durch Überprüfung von demnach sinnvollen Dreiergruppen finden.

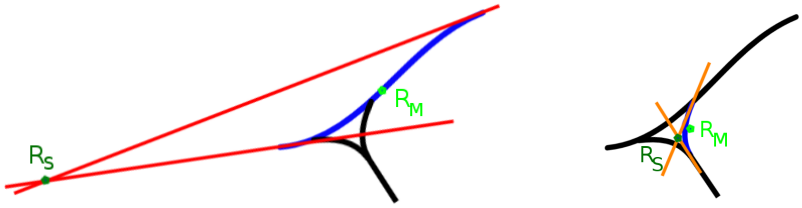


Abbildung 8: Hier wird ein Referenzpunkt für die blaue Kurve gesucht. Der Schnittpunkt der Tangenten ist in dunkelgrün, der Mittelpunkt in hellgrün eingezeichnet.

3.2 Überprüfung von Dreiergruppen

In den ersten Versionen des Skriptes basierte die Prüfung der angenommenen Abdrücke auf der Idee, dass die beieinanderliegenden Enden zweier Kurven des Abdrucks etwa die gleiche Richtung haben müssten. Wenn jedoch die Kurven nicht am selben Punkt enden, ist dies nicht unbedingt der Fall. Zudem ergibt sich hier das Problem ab wieviel Grad Abweichung die Dreiergruppe verworfen werden soll und das Problem von falschen, als Keilabdrücke erkannten Gruppierungen im Fall von ähnlichen Nachbarn (Abb. 9).

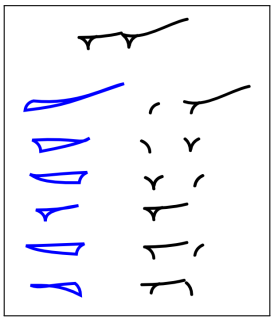


Abbildung 9: Ähnlich aussehende nebeneinander liegende Keile sind besonders schwer zu behandeln. Hier ein paar Beispiele für falsche Rekonstruktionen der Keile.

Der intuitive Ansatz ist es dagegen, den tatsächlichen Abstand der Kurven an den entsprechenden Enden zu vergleichen:

$$\min_{t_1,t_2} \sum_i \|B_{i,3}(t_1)C^{(1)} - B_{i,3}(t_2)C^{(2)}\|_2 \overset{?}{<} T \qquad (T > 0,\ t_1, t_2 \in [0,1]) \tag{3.3}$$

mit der Bedingung, dass eine Kurve etwa auf der anderen endet:

$$t_1 \approx 0 \quad \vee \quad t_1 \approx 1 \quad \vee \quad t_2 \approx 0 \quad \vee \quad t_2 \approx 1$$

Da aber auch hier das Problem nur durch unzuverlässige Konvergenzverfahren lösbar ist, muss erneut eine Approximation der Lösung gefunden werden.

Denkbar dafür ist die Berechnung des Schnittpunktes der jeweiligen Bézierkurven und die Überprüfung, ob die berechneten Parameter t der Schnittpunkte für beide Kurven zwischen 0 und 1 liegt, also tatsächlich auf der Bézierkurve oder nur geringfügig darüber oder darunter. Sind die Ableitungen an den Enden jedoch tatsächlich fast gleich, so kann dieser Schnittpunkt beliebig weit von den Kurven entfernt sein.

Gewählt wurde deshalb der Schnittpunkt jeweils einer Kurve und der Geraden, die durch die Endpunkte der anschließenden Kurve verläuft. Dieser wird darauf überprüft, ob er nicht zu weit von einem Endpunkt der Kurve entfernt ist, durch deren Endpunkte die Gerade gezogen wurde:

$$\|S - C_0\|_2 \overset{?}{<} T \qquad \vee \qquad \|S - C_3\|_2 \overset{?}{<} T \qquad (T > 0) \tag{3.4}$$

- $S \ldots$ Schnittpunkt einer Kurve und der Geraden durch die Endpunkte der anschließenden Kurve

Mit den Gleichungen für die Bézierkurve

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = P_{Bezier}(t) = (1-t)^3 \cdot C_0 + (1-t)^2 \cdot t \cdot C_1 + (1-t) \cdot t^2 \cdot C_2 + t^3 \cdot C_3 \tag{3.5}$$

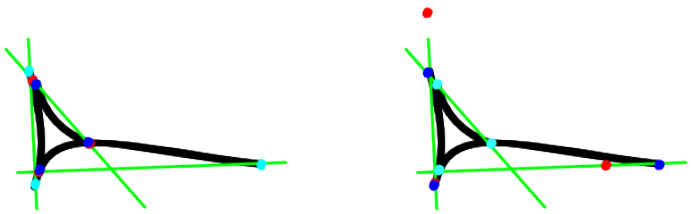
und der Gleichung der Geraden

$$\frac{(x - x_1)}{(x_2 - x_1)} = \frac{(y - y_1)}{(y_2 - y_1)}, \tag{3.6}$$

die durch die Punkte $P_{G1} = \binom{x_1}{y_1}$ und $P_{G2} = \binom{x_2}{y_2}$ verläuft, reduziert sich das Schnittpunktproblem auf die Suche einer Nullstelle einer kubischen Funktion

$$x(t) \cdot (y_2 - y_1) + y(t) \cdot (x_1 - x_2) + x_1 \cdot (y_1 - y_2) + \cdot(x_2 - x_1) = 0 \tag{3.7}$$

Diese Berechnung wird für jeden Endpunkt zweimal durchgeführt, wobei die Kurven beim zweiten Mal die Rollen wechseln. Allerdings ist pro Ecke meist nur einer dieser berechneten Punkte sinnvoll, da der jeweilige andere nicht auf der Strecke der entsprechenden Kurve liegt (Abb. 10).



(a) (b)

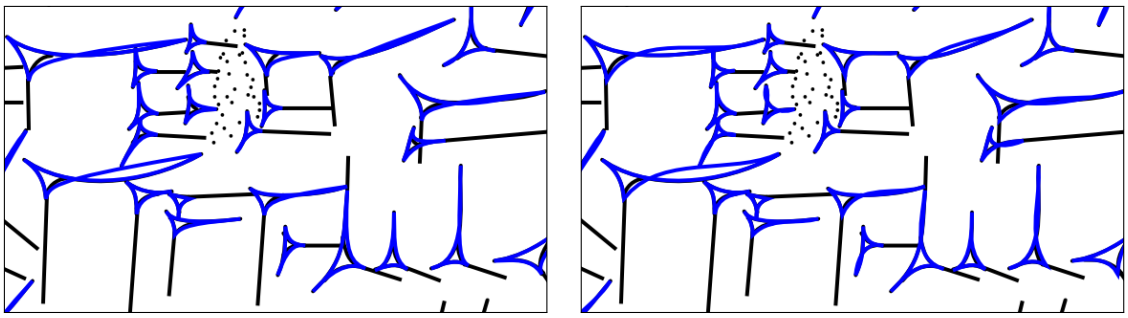
Abbildung 10: Die berechneten Punkte (rot) werden mit den enstprechenden Kurvenenden (dunkelblau) verglichen. Durch die eingezeichneten Enden der für den Schnittpunkt verwendeten Kurven (hellblau) lässt sich erkennen, dass bei (a) die Punkte tatsächlich Schnittpunkte der Kurve sind und bei (b) die Punkte außerhalb der Kurve liegen. Da pro Ecke nur einer der berechneten Punkte sinnvoll sein muss und die Abstände zwischen den zu vergleichenden Punkten gering ist, ist in diesem Fall die Überprüfung erfolgreich.

3.3 Zusammenfassung der drei Bézierkurven

Sind nun die drei zu einem Keilabdruck gehörenden Kurven gefunden, muss nun eine Polybézierkurve gefunden werden, die etwa dieselben Punkte durchläuft wie die ursprünglichen Kurven. Die einfachste Möglichkeit dafür ist es, für die Ecken des Abdrucks die Ecke von zwei zusammenlaufenden Kurven zu wählen, die weiter vom Mittelpunkt entfernt ist. Die entstehende Polybézierkurve

sieht dadurch etwas anders aus als die drei Originalkurven zusammen, trotzdem ist dieses Ergebnis nicht schlecht, weil es unter Umständen sogar mehr einem Abdruck gleicht als die Originalumzeichnung.

Eine andere Möglichkeit ist es, zwischen den beiden zu synchronisierenden Kurvenenden und auf der Kurve, deren Ecke weniger weit vom Mittelpunkt entfernt ist, Punkte zu berechnen und ein neues Fitting durchzuführen so wie es bei der Vorbereitung der einzelnen Kurven stattgefunden hat. Dies führt zu einer Darstellung, die oft näher an den ursprünglichen Umzeichnungen ist, manchmal jedoch auch zu weniger wünschenswerten Ergebnissen führt (Abb. 11). Was als die bessere Wahl der Rekonstruktionsmethode bezeichnet werden kann hängt demnach von den persönlichen Preferenzen ab, da jedoch die erste Methode um einiges performanter ist (ein Fitting hat eine Komplexität von $O(n^3)$ bezüglich der Anzahl der Punkte) wurde diese für den Algorithmus gewählt.



(a) (b)

Abbildung 11: Unterschiede bei der Synchronisierung von Kurvenenden (a) Auswahl einer der beiden Endpunkte (b) Neuberechnung einer Bézierkurve

3.4 Auffinden und Einbeziehung von Richtungskanten

Da die Keile in einigen Dokumenten nicht allein durch drei Kurven beschrieben sind, sondern oft der Großteil der Richtungskante separat durch eine oder mehrere Linien dargestellt ist, müssen auch diese gefunden und einbezogen werden. Das Finden von entsprechenden Kandidaten lässt sich leicht realisieren durch einen maximalen Abstand an den Ecken der Keilumzeichnung bzw. den entsprechenden Enden der Linien. Zur Überprüfung wird in diesem Fall der Winkel zwischen den Richtungsvektoren der Linie und der Kurve am entsprechenden Ende verglichen:

$$(i^*, j^*) = \min_{i \in \{0,3\}, j \in \{0,1\}} \|C_i - L_j\|_2 \overset{?}{<} T \qquad (T > 0) \tag{3.8}$$

mit den Linienendpunkten L_0 und L_1 und

$$\angle(\overline{C_{i^*} C_k}, \overline{L_0 L_1}) \overset{?}{<} \alpha \tag{3.9}$$

mit

$$k = \begin{cases} 1 & \text{falls } i^* = 0 \\ 2 & \text{falls } i^* = 3 \end{cases}$$

Mit dieser Methode können im Durchschnitt 97,6% der korrekten Keilzusätze gefunden werden. Nur 0,7% der gefundenen Zusätze werden dem falschen Abdruck zugeordnet, meist weil eine Linie doppelt zwei Keilen zugerechnet wird (Tabelle 1). Bei diesen Zahlen kommt es aber stark auf die Konfiguration des Skriptes an. Zu beachten ist, dass nur SVG-Linien und als Linien erkannte Splines als Verlängerung der Hauptrichtungskante erkannt werden können. Aufgrund der gekrümmten

Form der Umzeichnung mancher dieser Kanten können somit in manchen Dokumenten keine hundert Prozent erreicht werden.

Das Kombinieren der Kantenverlängerung mit den drei Hauptkurven des zugeordneten Keilabdrucks geschieht wie das Verschmelzen zweier Kurven durch Korrektur des entsprechenden Endpunktes.

Testnr.	Korrekte Zuweisung	Fehler 1. Art	Fehler 2. Art
2	361	20	8
3	784	24	6
4	3	1	0
5	668	7	6
6	1158	23	4
7	384	14	4
8	308	1	3
9	136	10	0
10	504	6	0
11	230	6	0
insgesamt:	4536	112	31

Tabelle 1: Statistiken über die Zuordnung der Verlängerungen der Hauptrichtungskanten

4 Fazit und Ausblick

Im Verlauf der Konstruktion des Algorithmus hat sich gezeigt, dass die Effektivität der Abdruckerkennung durch zwei Faktoren bestimmt wird:

1. die Lage des Referenzpunktes der Kurve; dieser sollte sich so nah wie möglich am Schwerpunkt des Abdrucks befinden, damit eine schnelle Zuordnung zu den zwei anderen Kurven der Umzeichnung möglich ist und die Wahrscheinlichkeit einer falschen Gruppierung reduziert wird,
2. die Methode zur Überprüfung, ob drei Kurven die Umzeichnung eines Keilabdrucks sein können; ist diese hunderprozentig zuverlässig, so reicht es aus, alle möglichen Gruppierungen zu testen um alle eingezeichneten Abdrücke zu finden.

Mit der Wahl der Methoden können im Durchschnitt etwa 95,2% der Abdrücke richtig erkannt werden. Nur 0,3% stellen Fehler der 1. Art dar, da die Gruppierung hier fehlschlägt. Etwa 4,5% stellen Fehler der 2. Art dar, da Dreiergruppierungen nicht erkannt werden; 39,3% davon aus dem Grund, da die dritte Kurve des Keils in der Umzeichnung nicht vorhanden ist. (Tabelle 2)

Die endgültige Darstellung wird bestimmt durch die Methode, die beim Synchronisieren der Kurvenenden und der Einbeziehung der Keilzusätze verwendet wird (Abschnitt 3.3), sowie die Rekonstruktion der Ausgangskurven durch eine kubische Bézierkurve bei der Vorbereitung der Daten.

Es ist ersichtlich, dass es an einigen Punkten noch Raum für Verbesserungen gibt. Darüber, ob es sinnvoll ist, auch fragmentare Abdrücke, die nur durch zwei Kurven vorliegen, zu erkennen, lässt sich streiten; bei den restlichen nicht erkannten Abdrücken aber müsste gründlich untersucht werden, ob dies auf den ersten oder zweiten der oben genannten Faktoren zurückzuführen ist, was den Rahmen dieser Arbeit jedoch sprengen würde. Es ist auch denkbar, auf die Einschränkung der möglichen Gruppierungen durch die Lage eines Referenzpunktes zu verzichten. Da jedoch damit die Wahrscheinlichkeit von Fehlern 1. Art deutlich gesteigert und der Rechenaufwand damit drastisch erhöht wird bleibt der Vorteil dieser Änderung fragwürdig.

Ein weiterer Punkt, bei dem eine alternative Methoden Sinn machen könnten, ist das Auffinden der Verlängerung der Hauptrichtungskanten. Hier wäre es interessant zu analysieren, ob ohne die Unterscheidung von Linien und Kurven bessere Ergebnisse erzielt werden können, d.h. dass als Hauptrichtungskante auch gekrümmte Linien akzeptiert werden.

Verbesserungen in der Rekonstruktion der Keilabdrücke können auch vorgenommen werden, allerdings sind diese von subjektiver Natur, solange die Resultate nur durch die Betrachtung durch Menschen evaluiert werden können. Eine Überarbeitung der Darstellung wird erst dann wieder sinnvoll, wenn die weitere Verarbeitung der Ergebnisse dieses Algorithmus feststeht.

Liegen die Keilabdrücke dieses Formates schließlich einheitlich als kubische Polybézierkurven vor, kann nun die nächste Herausforderung angegangen werden: die Erkennung von Keilabdrücken, deren SVG-Darstellung aus dem Tracing von Bitmap-Bildern [Sel03] enstanden ist (Abb. 12).

(a) (b) (c)

Abbildung 12: (b) zeigt das durch Bitmap-Tracing von (a) gewonnene SVG-Objekt in schwarz. In (c) ist mit rot ist die Polybézierkurve gezeichnet, die das Objekt mit Vektoren beschreibt.

Testnr.	Korrekte Zuweisung	Fehler 1. Art	Fehler 2. Art	
			3 Kurven	2 Kurven
1	395	0	3	11
2	383	2	16	5
3	845	6	33	10
4	6	0	2	2
5	715	0	16	16
6	1208	5	48	26
7	415	3	11	13
8	308	1	0	2
9	167	0	5	2
10	521	0	13	3
11	240	1	1	6
insgesamt:	5203	18	148	96

Tabelle 2: Statistiken über die Zuordnung von Kurven zu einem Keilabdruck

5 Referenzen

[Eis02] J. David Eisenberg. *SVG Essentials*. O'Reilly, 1 edition, 2002.

[Haa04] Harald Haarmann. *Geschichte der Schrift*. Beck'sche Reihe. Beck, 2 edition, 2004.

[Mau05] Stefan M. Maul. *Das Gilgamesch-Epos*. C.H.Beck, Munich, Germany, 3 edition, 2005.

[MKJB10] Hubert Mara, Susanne Krömker, Stefan Jakob, and Bernd Breuckmann. GigaMesh and Gilgamesh 3D Multiscale Integral Invariant Cuneiform Character Extraction. In Alessandro Artusi, Morwena Joly, Genevieve Lucet, Denis Pitzalis, and Alejandro Ribes, editors, *VAST: International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage*. The Eurographics Association, 2010.

[Neu35] Otto Neugebauer, editor. *Mathematische Keilschrift-Texte*. Springer, Berlin, 1935.

[Ray03] Erik T. Ray. *Learning XML*. O'Reilly, 2 edition, 2003.

[Sel03] Peter Selinger. Potrace: a polygon-based tracing algorithm. In *In http://potrace.sourceforge.net*, 2003.

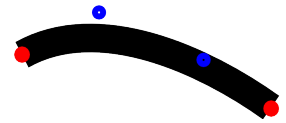
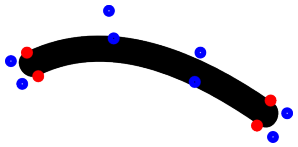
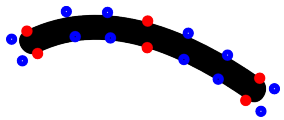
[SK11] Hans Rudolf Schwarz and Norbert Köckler. *Numerische Mathematik*. Vieweg + Teubner, Wiesbaden, 2011.

6 Anhang

6.1 Konfigurationen der Statistiken

1. python svgcfrec.py VAT_10908_Vs.svg -v -l Kopie
2. python svgcfrec.py VAT_10321_Vs_SJakob.svg -v -l Autographie -p -a 0.7 -e
3. python svgcfrec.py VAT_09671_Rs_SJakob.svg -v -l Autographie -p -e
4. python svgcfrec.py VAT_09671_Rs_SJakob.svg -v -l Autographie_Rand -p -e
5. python svgcfrec.py VAT_09898+10964_Vs_SJakob.svg -v -l Autographie -p -e
6. python svgcfrec.py VAT_09898+10964_Vs_SJakob.svg -v -l Autographie_09898 -p -e
7. python svgcfrec.py VAT_11022_SJakob.svg -v -l Autographie -p -e
8. python svgcfrec.py VAT_10622_HPSchaudig.svg -v -l g20 -p -e -t 0.001
9. python svgcfrec.py VAT_10686+Obv_HPSchaudig.svg -v -l g8902 -p -e
10. python svgcfrec.py VAT_10686+Obv_HPSchaudig.svg -v -l g74 -p -e -t 0.001
11. python svgcfrec.py VAT_10833-SeiteB_HPSchaudig.svg -v -l g20 -p -e -t 0.001

Tabelle 3: Darstellungsarten von kubischen Bezierkurven

	Knotenpunkte (rot)	Kontrollpunkte (rot und blau)	*path*-String in der SVG-Datei
	2	4	m 312.86484,353.24798 c -0.833,1.167 -1.667,3 -0.917,4.333
	4	12	m 309.704,352.924 c -0.911,1.318 -1.677,3.028 -0.917,4.585 0.142,0.29 0.572,0.036 0.432,-0.252 -0.674,-1.381 0.124,-2.934 0.917,-4.082 0.183,-0.264 -0.25,-0.514 -0.432,-0.251
	6	18	m 309.704,352.924 c -0.45328,0.65578 -0.87065,1.40861 -1.08203,2.19135 -0.21345,0.79042 -0.21683,1.61135 0.16503,2.39365 0.142,0.29 0.572,0.036 0.432,-0.252 -0.33897,-0.69455 -0.30562,-1.4326 -0.0878,-2.14075 0.21525,-0.69991 0.61066,-1.37061 1.00484,-1.94125 0.183,-0.264 -0.25,-0.514 -0.432,-0.251