# Effiziente Berechnung von generalisierten Voronoi-Diagrammen als Grundlage zur Skelettierung von Konturen

Felix Lange

Berlin, 3. Januar 2011

# Inhaltsverzeichnis

# 1 Einleitung

Das Konzept von Voronoi-Diagrammen existiert nun seit über drei Jahrhunderten. Seit den 1970ern werden schon Algorithmen zur Berechnung von Voronoi-Diagrammen entwickelt und es werden immer noch jährlich neue Ansätze, Erweiterungen und Anwendungsmöglichkeiten vorgestellt. In dieser Ausarbeitung soll es darum gehen, den Ansatz [HKL+99] vorzustellen, welcher zur approximierten Berechnung generalisierter Voronoi-Diagramme stark auf der Verwendung von Computergrafikhardware aufbaut.

In Abschnitt 2 wird mit einer Einführung und Übersicht zum Thema Voronoi-Diagramme begonnen und in Abschnitt 3 werden dann die Berechnungarten betrachtet, mit Schwerpunkt auf die approximierte Berechnung und die Verwendung von Grafikbeschleunigung. Im letzten Abschnitt 4 geht es um die Verbindung zwischen Voronoi-Diagrammen und Skelettierung von Körpern und Formen, einem Thema aus der Bildverarbeitung. Zum Schluss wird gezeigt, dass sich eine Skelettierung effizient und ohne weitere Berechnungen erstellen lässt, indem man nur leichte Modifikationen am Verfahren [HKL+99] vornimmt.

# 2 Voronoi-Diagramme

Dieser Abschnitt stellt eine Übersicht zum Thema Voronoi-Diagramme dar. Er beinhaltet Begriffserklärungen und mathematische Definitionen und zeigt zudem eine Auswahl an Anwendungsbeispielen.

## 2.1 Krankenhausproblem

Als erstes wird eine Variante des ”Post Office” Problems betrachtet [dBCvKO08]: Gegeben sei zu einem begrenzten Gebiet (z.B. zu einer Stadt oder zu einem Land) eine Menge an Krankenhäusern mit ihren geographischen Positionen. Außerdem gilt die Annahme, dass die Krankenhäuser unspezialisiert sind und so jeder Mensch in das Krankenhaus geht, welches für ihn am nächsten liegt. Dieses Szenario betreffend lassen sich aus unterschiedlichen Blickpunkten mehrere Fragen stellen:

- Welches Krankenhaus ist für mich das Nächste?
- Wie hoch ist die Auslastung eines Krankenhauses?
- In welchem Bereich werden zusätzliche Krankenhäuser benötigt?

- Wo sollten im optimalen Fall neue Krankenhäuser plaziert werden?

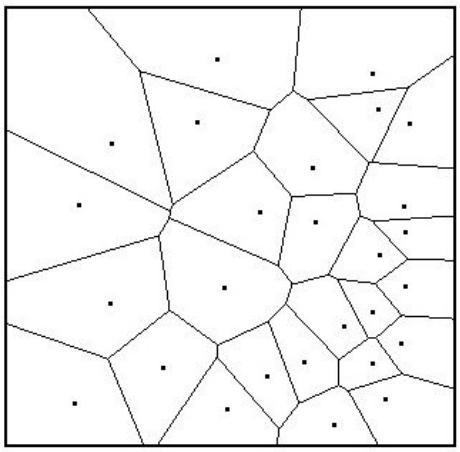

**Abbildung 1:** *Voronoi-Diagramm zu einer gegebenen Punktmenge.*

Die Antworten zu diesen geographischen Fragen lassen sich über Voronoi-Diagramme (auch Dirichlet-Zerlegungen genannt) finden. Zur Erstellung eines Voronoi-Diagramms wird jeder Punkt des Gebiets einem Krankenhaus zugeordnet, welches ihm am nächsten liegt. Durch diese Operation wird das gesamte Gebiet in Regionen/Zellen (*Voronoi cell/face*) eingeteilt. Jede Zelle hat genau ein Krankenhaus als Zentrum (*Voronoi site*). Die Punkte des Gebiets, die zu mehreren Krankenhäusern den gleichen Abstand haben, bilden die Grenzen (*Voronoi boundaries*) zwischen den einzelnenen Zellen.

Voronoi-Zentren können im Allgemeinen auch aus komplexeren Formen als aus Punkten bestehen, wie z.B. aus Linien, Kurven oder ganzen Polygonen. Eine genauere Betrachtung hierzu erfolgt in Abschnitt 3.1.

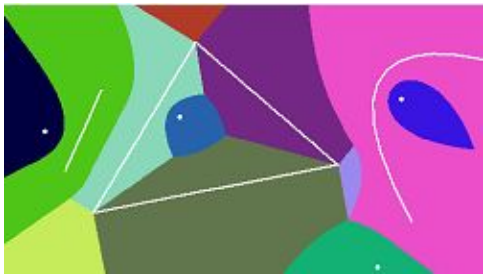

**Abbildung 2:** *Voronoi Diagramm mit komplexeren Zentren (in weiß) und eingefärbten Zellen [HKL+99].*

## 2.2 Definitionen und Eigenschaften

Als Abstandsfunktion wird hier der Euklidische Abstand verwendet. Die Abstandsfunktion für Voronoi-Diagramme ist aber beliebig und hängt von der Anwendung ab.

$$dist(p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{1}$$

Sei $M = \{z_1, ..., z_n\}$ die Menge der Zentren und $Vor(M)$ das dazugehörige Voronoi-Diagramm, so besteht jede Zelle $V(z_i)$ mit dem Zentrum $z_i$ genau aus der Menge der Punkte $p$, die näher zu $z_i$ liegen als zu allen anderen Zentren [HKL+99].

$$V(z_i) = \bigcap_{j \neq i} \{\ p \mid dist(p, z_i) \leq dist(p, z_j)\ \} \tag{2}$$

Voronoi-Diagramme dienen, genau wie Triangulationen, der Zerlegung von Flächen. Der wesentliche Unterschied ist aber, dass die Zerlegung bei Voronoi-Diagrammen durch die Zentren und eine eventuelle Gewichtung der Zentren eindeutig ist. Eine spezielle Form der Triangulation, die Delauney Triangulation, lässt sich direkt aus einem Voronoi-Diagramm erstellen. Hierzu werden die Zentren von benachbarten Voronoi-Zellen mit Strecken verbunden und bilden damit die Kanten der Delauney Triangulation. Dieses Thema wird in zwei weiteren folgenden Vorträgen dieser Vortragsreihe genauer behandelt.

## 2.3 Anwendungsbeispiele

Es folgt eine kurze Zusammenstellung an Anwendungsbeispielen, welche zeigen soll, wie unterschiedlich die Anwendungsbereiche und damit auch die Anforderungen an Genauigkeit oder Geschwindigkeit für die Berechnung der Voronoi-Diagramme sein können.

**Robot Motion Planning**

Ein autonomer Agent (Roboter) soll in einer definierten Umgebung, bestehend aus sowohl statischen als auch dynamischen Objekten, von einem Startpunkt zu einem Zielpunkt gelangen, ohne dabei mit der Umgebung zu kollidieren.

Wenn die Objekte der Umgebung als Voronoi-Zentren behandelt werden, stellen die Voronoi-Grenzen mögliche kollisionsfreie Wegstücke dar (unter der Bedingung, dass sie einen ausreichenden Abstand zu den Objekten haben - siehe Abbildung 3). In einem Szenario mit einer dynamischen Umgebung und hohen Bewegungsgeschwindigkeiten, ist es wichtig, dass Berechnungen schnell und effizient ausgeführt

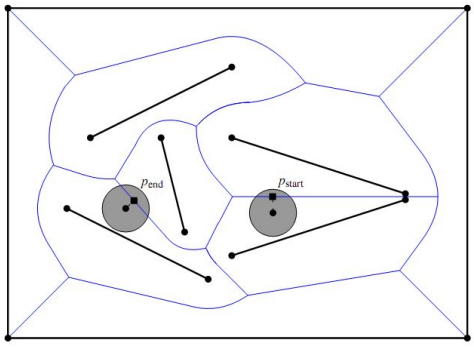


**Abbildung 3:** *Voronoi-Diagramm zur Problemstellung mit Start- und Zielpunkt des Agenten [dBCvKO08].*

werden können. Im Gegensatz dazu steht ein statisches Szenario, bei dem Präzision erforderlich ist und somit kein Spielraum für Fehler besteht. Statt effiziente Algorithmen werden nun robuste Ergenisse benötigt.

Das Thema Motion Planning wird in der Vortragsreihe noch genauer behandelt und beinhaltet auch Lösungsstrategien, die nicht auf Voronoi-Diagrammen aufbauen.

**Voronoi Deformation Density**

Voronoi-Diagramme finden sogar in der computergestützen Chemie Anwendung. So kann über die *Voronoi Deformation Density* (VDD) Methode in der Chemie die Partialladung von Molekülen berechnet werden. Als Voronoi-Zentren werden hier die Postion der Kerne verwendet [GHBB04]. In diesem Anwendungsbeispiel wird im atomaren Wertebereich gerechnet und somit werden für robuste Ergebnisse auch nummerisch korrekte Algorithmen benötigt.

**Mosaikeffekt**

Mosaik ist eine Gestaltungstechnik, bei der durch Zusammenfügen von einfarbigen Elementen Muster oder Bilder entstehen. Dieses Erscheinungsbild lässt sich als Effekt auf Bilder/Fotos anwenden. Das Ergebnis ist ein Bild, welches aus vielen kleinen einfarbigen Flächen besteht. Die Form der Flächen ist dabei in bestimmten Grenzen bewusst randomisiert und die Farbe wird aus dem Orginalbild ermittelt.

Unter der Anforderung, die Berechnungen in Echtzeit durchführen zu können, bietet es sich an, diesen Effekt über Voronoi-Diagramme zu berechnen (siehe Abbildung 4). Die Einteilung in kleine Flächen richtet sich nach den Voronoi-Zellen zu einer randomisierten Punktmenge. Mit effizienten Algorithmen zur Berechnung der

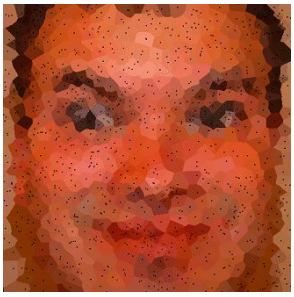

**Abbildung 4:** *Mosaikeffekt angewendet auf ein Porträt der Auflösung 512x512. Das verwendete Voronoi-Diagramm wurde aus 1000 Punktzentren berechnet [HKL+99].*

Voronoi-Diagramme ist es möglich, sowohl Parameter für die Einteilung als auch die Bildquelle selbst in Echtzeit zu ändern.

**Formanalyse**

Ein weiterer Anwendungsbereich,der Voronoi-Diagramme, liegt in der Objekt- und Schrifterkennung aus dem Bereich der Mustererkennung. Mittels Skelettierung können Formen von Objekten oder Schriftzeichen auf wesentlich geringere aber charakteristischere Datenmengen reduziert werden. Sie lassen sich damit zur Bestimmung/Erkennung leichter mit schon identifizierten Daten aus Datenbanken vergleichen. Eine Skelettierung lässt sich effizient aus einem Voronoi-Diagramm

**Abbildung 5:** *Eine mögliche Skelettierung zum Buchstaben B.*

berechnen. Dieser Zusammenhang wird in Abschnitt 4 genauer betrachtet.

# 3 Effiziente Berechnung von generalisierten Voronoi-Diagrammen

In diesem Abschnitt liegt der Schwerpunkt auf der Berechnung von Voronoi-Diagrammen im zweidimensionalen Raum. Zusätzlich dazu werden weitere gängige Varianten von Voronoi-Diagrammen betrachtet. Grundsätzlich unterscheidet man zwei Gruppen von Berechnungsarten: die implizten und expliziten Berechnungen, bzw. die effizienten und exakten Algorithmen. Bei effizienten Algorithmen werden Teile des Problems approximiert und damit die Berechnungen erheblich vereinfacht. Der Nachteil gegenüber den exakten - und aufwendigeren Algorithmen - liegt bei den Fehlergrenzen. Je nach Algorithmus sind die Fehlergrenzen aufgrund der Approximationen wesentlich höher, nicht im ganzen Umfang bekannt und/oder lassen sich nicht beliebig verkleinern. Die Wahl des Algorithmus ist, wie in Abschnitt 2.3 gezeigt wurde, stark von den Anwendungsanforderungen abhängig.

## 3.1 Berechnung von Voronoi-Diagrammen

In Abschnitt 2.2 wurde die mathematische Definition für eine Voronoi Zelle gezeigt. Die einfachste Möglichkeit, ein Voronoi-Diagramm zu erstellen wäre es, die Gleichung 2 für alle Zellen einzeln durchzuführen. Pro Zelle benötigt dieser Algorithmus eine Laufzeit von $\mathcal{O}(log\ n)$, was zu einer Gesamtlaufzeit von $\mathcal{O}(n^2\ log\ n)$ führt. Diese Laufzeit ist nicht optimal und kann z.B. durch Fortune's Sweepline Algorithmus auf $\mathcal{O}(n\ log\ n)$ reduziert werden [dBCvKO08].

Die bisher angeführten Betrachtungen waren auf Zentren mit der Form eines Punktes beschränkt. Zentren können aber auch, wie oben schon erwähnt, aus komplexeren Formen, wie Linen, Kurven und ganzen Polygonen, bestehen. In diesem Fall spricht man von generalisierten Voronoi-Diagrammen [Sug93]. Die Generalisierung betrifft aber nicht nur die Form des Zentrums, sondern auch die Art der Abstandsfunktion (z.B. Manhatten Metrik statt Euklidischer Abstand) und die Ordnung des Diagramms.

Die Ordnung des Diagramms darf jedoch nicht mit der Dimension des Raumes verwechselt werden. Bei Voronoi-Diagrammen höherer Ordnung wird eine Zelle nicht nur durch ein Zentrum $z_i$ bestimmt, sondern durch eine Teilmenge an Zentren $Z = \{z_1,...,z_k\}$ (auch Generatoren genannt), wobei $k$ der Ordnung des Diagramms entspricht. $Vor(M)$ besteht dann aus $n - k$ Zellen [ABMS98]. Die Ordnung und die Form der Zentren hängen oft eng zusammen, da ein komplexeres Zentrum auch aus mehreren Elementen besteht.

Die Berechnung von generalisierten Voronoi-Diagrammen wird zwangsläufig

komplexer. Es gibt eine Erweiterung vom Fortune's Sweepline Algorithmus, welche auch Linien als Zentren berücksichtigt. Dieser Algorithmus hat zwar immernoch eine Laufzeit von $\mathcal{O}(n\ log\ n)$[dBCvKO08], aber generell muss bei der Berechnung und bei der Darstellung dieser Diagramme mit hochgradigen algebraischen Kurven und Flächen und deren Schnittpunkten gearbeiten werden[HKL+99].


**Abbildung 6:** *Die Grenze zwischen zwei Punkten (a) und zwischen einem Punkt und einer Strecke (b).*

In Abbildung 6 wird veranschaulicht, dass die Voronoi-Grenzen zu Punktzentren nur aus Geraden bestehen. Diese lassen sich aus den Schnittpunkten von Kreisen leicht bestimmen. Im Vergleich dazu muss mit Parabeln als Grenzen gearbeitet werden, sollten noch Linien als Zentren hinzukommen.

Es ist eine Vielzahl an Algorithmen bekannt, die generalisierte Voronoi-Diagramme mit unterschiedlich hoher Komplexität nummerisch robust (mit minimalem Fehler $\varepsilon$) lösen. Es ist aber nach [HKL+99] keiner bekannt, der die Problemstellung für hochfrequente Anwendungen gleichzeitig effizient und numerisch robust lösen kann. Daher greift man für viele Anwendungsbereiche, wie zB. in der Computergrafik, auf die approximierte Berechnung von generalisierten Voronoi-Diagrammen zurück.

## 3.2 Approximierte Voronoi-Diagramme

Die Komplexität, der höhere Grad der algebraischen Kurven und Flächen, die das Voronoi-Diagramm beschreiben, werden bei der approximierten Berechnung reduziert. Zu diesem Zweck können die Form der Zentren, der Raum oder die Grenzen approximiert werden, wobei jedoch ein höhrer Fehler in Kauf genommen werden muss. Vor der Verwendung von approximierten Verfahren ist es notwendig, die Fehlergrenzen abschätzen zu können und zu analysieren, ob diese für die Anwendung aktzeptabel sind.

Eine verbreitete Variante aus [ABMS98] beinhaltet *point-sampling of sites*. Hierbei werden die Zentren beliebiger Form durch eine bestimmte Menge an Punk-

ten, mit einem geringen Abstand $\delta$ zueinander, ersetzt. Dadurch wird die gesamte Berechnung auf Abstandsberechnungen zwischen Punkten reduziert, wobei die Ergebnisse für mehrere zusammengehörige Punkte zu einer Zelle zusammengefasst werden müssen. Bei diesem Verfahren ist es nach [HKL+99] schwierig, die Fehlergrenzen abzuschätzen und die Gesamtkomplexität ist noch nicht eindeutig geklärt.

Neben dieser Möglichkeit gibt es noch weitere Ansätze der approximierten Berechnung, bei denen z.B. der Raum zur Vereinfachung adaptiv unterteilt wird. Nach [HKL+99] sind die Ansätze bis 1999 aber in der Praxis für große Modelle zu zeit- und speicheraufwendig, können nicht einfach für dynamische Systeme erweitert werden oder sind in ihrer Allgemeinheit beschränkt.

## 3.3 Effiziente Berechnung mittels OpenGL

In diesem Abschnitt wird "Fast Computation of Generalized Voronoi Diagrams using Graphics Hardware" aus [HKL+99] vorgestellt. Dieses stellt ein weiteres Verfahren mit approximierten Berechnungen dar, mit welchem die bisher aufgezeigten Probleme gelöst werden. Es können also, mit ausreichender Genauigkeit und bekannter Fehlergrenze, generalisierte Voronoi-Diagramme effizient berechnet werden. Der Algorithmus arbeitet sowohl im zwei- als auch im dreidimensionalen Raum, ist aber nicht für Voronoi-Diagramme höherer Ordnung geeignet. In dieser Ausarbeitung wird sich bei der Betrachtung auf den zweidimensionalen Raum beschränkt.

Ziel des Ansatzes [HKL+99] ist es, alle Pixel, die nach Definition einem Zentrum am nächsten liegen, mit der eindeutigen Farbe des Zentrums einzufärben (siehe Abbildung 7 rechts). Danach werden anhand der Farbunterschiede die Voronoi-Grenzen bestimmt. Hierdurch unterscheidet sich der Ansatz stark von anderen Ansätzen, bei denen die Voronoi-Grenzen direkt mathematisch berechnet werden.

### 3.3.1 Naive Implementierung

Für Voronoi-Diagramme deren Zentren nur aus Punkten bestehen, könnte man dies mittels einer einfachen brute-force Variante lösen. Um alle Pixel richtig einzufärben, betrachtet man alle Zentren $z_i \in M$ nacheinander und durchläuft dabei jeweils immer die Menge der Pixel. Bei jedem Pixel $p_j$ wird die Entfernung zum aktuellen Zentrum $dist(z_i, p_j)$ betrachtet und wird mit der bisher zu $p_j$ kleinsten bekannten Entfernung verglichen. Sollte die Entfernung $dist(z_i, p_j)$ kleiner sein, wird sowohl

die kleinere Entfernung gespeichert, als auch $p_j$ mit der entsprechende Farbe von $z_i$ eingefärbt. Diese Variante ist jedoch schon für Punkt Voronoi-Diagramme mit $n$ Zentren und $m$ Pixeln bei einer Laufzeit von $\mathcal{O}$(n m) ineffizient und müsste für komplexere Zentren noch erweitert werden. Das prinzipielle Verfahren kann aber über den Gebrauch von Grafiksoft- und hardware erheblich beschleunigt werden.

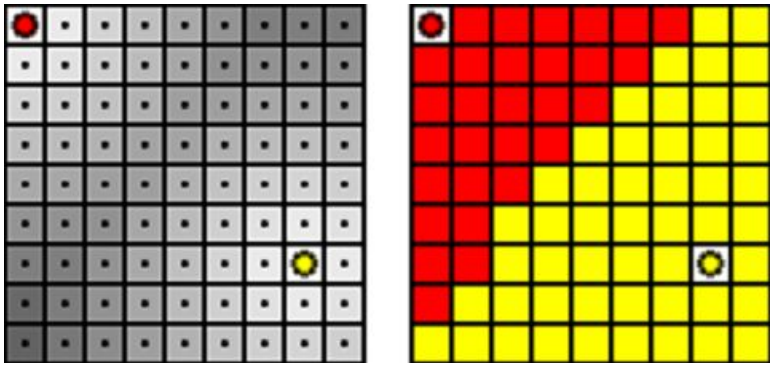

**Abbildung 7:** *Kleinste Abstände zu einem der beiden Zentren als Graufstufen (links). Dominanzregionen der Zentren, eingefärbt mit der entsprechenden Farbe (rechts) [HKL+99].*

### 3.3.2 Grafikbeschleunigung

In der Computergrafik wird bei der Projektion eines 3D-Modells auf die Bildebene (z.B. auf den Bildschirm) überprüft, welches Teilobjekt durch welches andere Teilobjekt überdeckt ist und damit für den Betrachter nicht mehr sichtbar ist. Hierfür wird der *Framebuffer*, der *Z-buffer* und der Tiefenvergleich (*depth-test*) benötigt. Im Z-buffer werden die entsprechenden Abstände von der Bildebene zum Teilobjekt für jeden Pixel gespeichert. Wird ein weiteres Objekt in den 3D-Raum gezeichnet, wird an Stellen, wo sich Teilobjekte auf der Projektionsebene schneiden, über den Tiefenvergleich bestimmt, welches Teilobjekt näher zum Betrachter liegt. Dieses Teilobjekt dominiert damit und die entsprechenden Farbwerte werden in den Framebuffer geschrieben. Die dazugehörigen Abstände werden dabei immer im Z-buffer aktualisiert [SWND07].

Statt die Tiefenwerte beim Schreiben in den Z-buffer pro Pixel einzeln zu berechnen, können sie linear interpoliert werden. Dies bedeutet im Detail, dass Objekte in Triangulationen zerlegt und zum Rendern nur die Eckpunkte der einzelnen Dreiecke übergeben werden. So können alle Tiefenwerte über lineare Interpolation zwischen den Eckpunkten bestimmt werden [HKL+99].

### 3.3.3 Abstandsfunktion

Beide beschriebenen Komponenten (lineare Interpolation und der Tiefenvergleich mit Hilfe des Z-buffers) aus dem letzten Abschnitt finden ihre Anwendung bei diesem Ansatz [HKL+99] in der Abstandsberechnung. Ziel ist es, die Abstandsfunktionen von Zentren durch dreidimensionale Körper zu repräsentieren. Zu diesem Zweck müssen die Abstände vom einem Zentrum zu jedem Punkt der Ebene an den Punkten senkrecht zur Ebene (auf der Z-Achse) eingetragen werden.

Es soll zunächst die Abstandsfunktion eines Punktzentrums betrachtet werden. Diese breitet sich auf der Ebene kreisförmig aus und ergibt somit den Körper eines Kegels (*cone*). Der Kegel kann nun als approximiertes Polygonnetz (*polygon mesh*) mit dem Voronoi-Zentrum als Spitze ins Bild gezeichnet werden (siehe Abbildung 8 a). Die Tiefenwerte zwischen Spitze und maximalem Abstand werden daher nicht einzeln berechnet, sondern beim Zeichnen des Mantels von der entsprechenden Grafikroutine interpoliert.

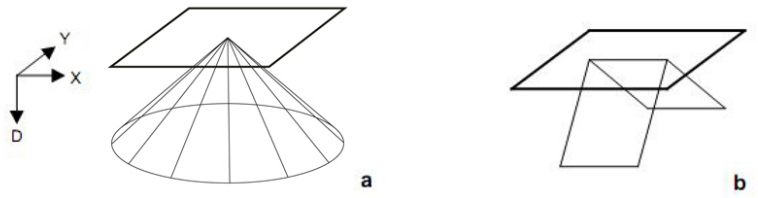


**Abbildung 8:** *Kegel als Abstandsfunktion für Punktzentren (a) und ein Zelt für linienförmige Zentren (b) [HKL+99].*

Ein linienförmiges Zentrum besteht aus drei Komponenten: der Linie und den zwei Endpunkten. Das Polygonnetz der Abstandsfunktion der Linie entspricht einem Zelt (*tent* - siehe Abbildung 8 b). Es besteht aus zwei rechteckigen Flächen, welche die Abstandsfunktion exakt repräsentieren. Die beiden Endpunkte werden wieder durch Kegel repräsentiert.

Ein komplexeres Zentrum, welches die Form eines Polygons hat, setzt sich nun aus mehreren Linien und Punkten zusammen. Ein approximiertes Polygonnetz kann daher auch hierfür, wie oben beschrieben, erstellt werden.

Nun werden zu allen Zentren beliebiger zweidimensionaler Form nacheinander die entsprechenden dreidimensionalen Polygonnetze erstellt, welche sich in der Tiefe schneiden. Bei jedem Schritt kommen der Z-buffer und der Tiefenvergleich zum Einsatz, um das bis dahin entstandene 3D-Modell über die Parallelprojektion auf

die Bildebene abzubilden. Darüber wird nun am Ende berechnet, welcher Kegel an welcher Stelle überdeckt wird oder im Bezug auf den geringsten Abstand dominant ist. Als Ergebnis ergibt sich im Framebuffer das gesuchte Voronoi-Diagramm.

### 3.3.4 Approximation

Bei der Repräsentation der Abstandsfunktion von Punktzentren wurde von einem approximierten Polygonnetz gesprochen. Ursache hierfür ist, dass die exakte Abstandsfunktion einem Kegel mit kreisförmiger Basis entspricht. Für das Zeichnen in Frame- und Z-buffer können jedoch nur die Eckpunkte von triangulierten Flächen übergeben werden. Zu diesem Zweck wird aus dem Kegel ein Polygonnetz erstellt. Sein Mantel wird über Dreiecke approximiert und die Basis ist nun ein Vieleck (siehe Abbildung 9). Mit der Anzahl der Dreiecke steigt die Genauigkeit und der Approximationsfehler $\varepsilon$ geht gegen 0.

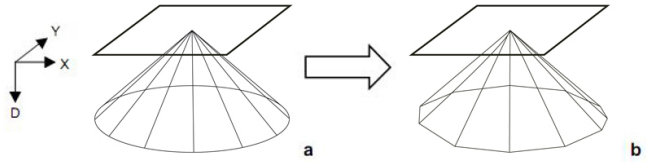


**Abbildung 9:** *Exakter Kegel (links) und sein approximiertes Polygonnetz (rechts).*

Die Idee zur Verwendung von Kegeln, um die Abstandsfunktion von Punktzentren darzustellen, stammt aus einer Randnotiz eines OpenGL Programmierhandbuch [SWND07]. Der Autor ging aber davon aus, dass man für einen Kegel bei geringem Fehler tausende Polygone benötigt. Dies wird nun genauer betrachtet.

Der durch die Approximation der Grundfläche entstehende maximale Fehler $\varepsilon$ des Kegels liegt genau zwischen zwei benachbarten Endpunkten. Da der Winkel zwischen Basis und Mantel des Kegels $45°$ beträgt, entspricht dieser Fehler auch dem maximalen Fehler in der Tiefe, also dem Fehler auf den Abstand bezogen.

Aus Abbildung 10 folgt folgende Berechnung für den Winkel $\alpha$ der Dreiecksspitzen, abhängig von $\varepsilon$ und $R$:

$$\alpha = 2\cos^{-1}\left(\frac{R-\varepsilon}{R}\right) \tag{3}$$

Aus $\alpha$ folgt trivial die Anzahl der Dreiecke. Durch die Wahl von $\alpha$ kann nun noch weiter zwischen Effizienz und Genauigkeit geregelt werden.

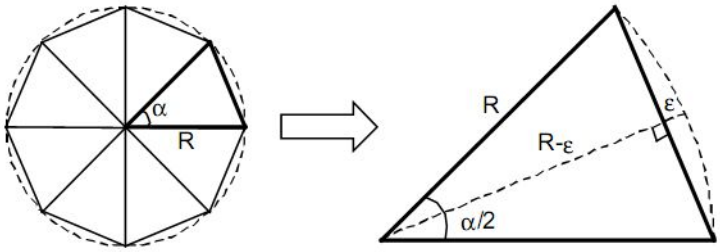


**Abbildung 10:** *Kegel von oben (links) und ein einfaches Dreieck aus dem Polygonnetz (rechts). Der Winkel $\alpha$ soll maximiert werden [HKL+99].*

Durch die Verwendung des Frame- und Z-buffers sind die Berechnungen an das Pixelmaß gebunden. Die Genauigkeit und Komplexität steigen somit monoton mit der Auflösung (bei konstantem Approximationsfehler $\varepsilon < 1$ Pixel). Der Radius $R$ der Basis entspricht der maximalen Distanz, welche bei einem quadratischen Bild der Auflösung $M \times M$ genau $M\sqrt{2}$ beträgt. Daraus ergibt sich, dass für eine Auflösung von $1024 \times 1024$ pro Kegel nur 85 Dreiecke benötigt werden.

### 3.3.5 Kurven

Sollten Zentren aus geschwungenen Linien/Kurven bestehen, sind weitere Approximationen nötig. Diese Kurven müssen über lineare Paketierung (Tesselation) durch verbundene Linienstücke ersetzt werden, da sonst wieder mit hochgradigen Voronoi-Grenzen gearbeitet werden müsste. Dazu können Algorithmen von [GF87] und [KML95] verwendet werden, bei denen auch die Fehlergrenzen klar definiert sind und über eine feinere Approximation (höhere Anzahl an Liniensegmenten für eine geschwungene Linie) der Fehler reduziert werden kann. Der Gesamtfehler der Approximation ergibt sich dann einmal aus der linearen Paketierung, sowie aus den Triangulationen der Kegel für die Eckpunkte.

### 3.3.6 Polygone and Per-feature Voronoi-Diagramme

Es gibt in Summe nur zwei verschiedene Polygonnetze, aus deren Kombination die Abstandsfunktion zu allen Formen der Zentren beschrieben werden können. Sobald man diese aber kombiniert, entstehen immer Überlappungen an den Endpunkten, also doppelte Berechnungen. Der Aufwand kann nun erneut reduziert werden, indem bei den Endpunkten von Linien nur halbe Kegel gerendert werden und bei den Polygonen und Kurven entsprechende Teilkegel (siehe Abbildung 11 b und 12 b).

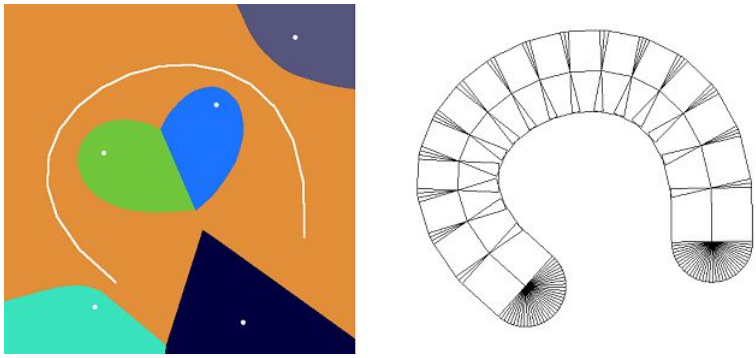

**Abbildung 11:** *Voronoi-Diagramm von einer Bezierkurve und 5 Punkten (links). Abstandspolygonnetz der Bezierkurve (rechts) [HKL+99].*

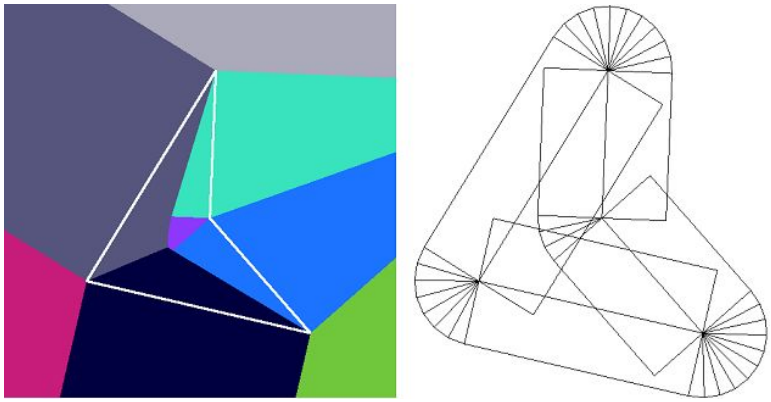

**Abbildung 12:** *Per-feature Voronoi-Diagramm zu einem Polygon (links) und sein approximiertes Polygonnetz (rechts) [HKL+99].*

Betrachtet man die Komponenten von Polygonen am Ende (trotz Reduzierung der Kegel) wieder einzeln mit eigenen eindeutigen Farben, so erhält man ein sogenanntes Per-feature Voronoi-Diagramm (Abbildung 12). Diese Art der Interpretation ist für verschiedene Anwendungen interessant und wird z.B. für Skelettierung und der darauf aufbauenden Formanalyse verwendet (siehe Abschnitt 4).

### 3.3.7 Bestimmung der Voronoi-Grenzen

Um die Grenzen zwischen den Zellen zu finden, sollen hier zwei Ansätze vorgestellt werden. Für beide Ansätze wird die Tatsache verwendet, dass zwei benachbarte Pixel unterschiedlicher Farbe ein Stück einer Grenze definieren. Dies ergibt sich aus der Definition von Voronoi-Grenzen und der Art wie Voronoi-Diagramme in diesem Verfahren erstellt werden.

In der ersten Variante wird jeder Pixel des Bildes einzeln durchlaufen und

dabei seine Nachbarn betrachtet. Dies lässt sich leicht implementieren und findet zuverlässig alle Kanten. In der Bildverarbeitung gibt es zusätzlich Algorithmen unter dem Begriff Kantenerkennung, um dieses Problem effizienter zu lösen.

In der zweiten Variante wird bei einem Startwert, welcher eine Grenze darstellt, begonnen und von diesem entlang der gefundenen Grenzstücke weitergesucht. Ein Startwert kann leicht gefunden werden, indem der Rand des Bildes absucht wird. Dieses Verfahren basiert auf der Eigenschaft, dass alle Grenzen verbunden sind. Diese gilt nicht für alle Arten von Voronoi-Diagrammen. Zum Beispiel können durch Kurven und Kreise Voronoi-Zellen eingeschlossen werden, wodurch ihre Grenzen nicht mit Anderen verbunden sind (siehe Abbildung 10). Der Algorithmus hat aber den Vorteil, dass man leichter benachbarte Voronoi-Zellen indentifizieren kann, was z.B. für die Delauney-Triangulation wichtig ist[HKL+99].

# 4 Skelettierung (Mediale Achsen)

Dieser Abschnitt widmet sich der genaueren Betrachtung einer der im Abschnitt 2.3 vorgestellten Anwendungsbeispiele. Unter dem Begriff Skelletierung (*skeleton*) stellt es eine Klasse an Verfahren aus der Bildverarbeitung dar und findet z.B. in der Formanalyse, Schrift- und Mustererkennung Anwendung. Da die Skelettierung effizient und robust aus gerneralisierten Voronoi-Diagrammen ermittelt werden kann, bildet das Verfahren aus Abschnitt 3.3 eine wichtige Grundlage. Besonders für Echtzeit-Anwendungen, z.B. Anwendungen aus der Computer Vision, bei denen Kamerastreams analysiert werden, ist die Effizienz von großer Bedeutung.

Ein weiterer wichtiger Vorteil der Skelettierung liegt in der Datenreduktion. Mit dem Ziel, eine Form wieder vollständig aus seinem Skelett rekonstruieren zu können, bietet es sich an, die Form mit einer wesentlich kleineren Datenmenge abzuspeichern. Daneben ist die Datengröße auf diese Art unabhängig von der Auflösung und die Form kann nach einer Vektoriserung beliebig skaliert werden.

## 4.1 Definitionen

Der Begriff Skelettierung wurde als Erstes von Blum [Blu67] vorgestellt. Ein Skelett zu einem Objekt besteht aus einer dünnen linienförmigen Figur, welche in der Form zentriert, das Objekt topologisch korrekt repräsentiert. Die einzelnen linienförmigen Bestandteile des Skeletts werden als Mediale Achsen (*medial axis*) bezeichnet.

Ein Skelett $S$ zu einer Form $G \in R^2$ setzt sich aus einer Menge an Kreiszentren $p$ zusammen, dessen Kreise $B(p, r)$ komplett in $G$ liegen und maximal sind.

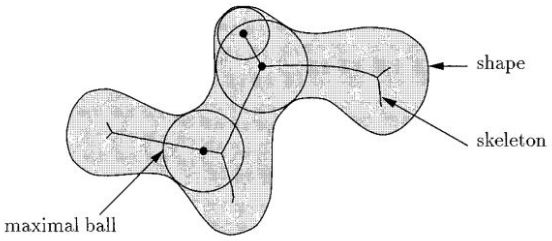


**Abbildung 13:** *Skelett als Menge der Zentren von maximalen Kreisen [AM97].*

Maximal bedeutet hierbei, dass der Radius $r$ maximal ist und es keinen anderen Kreis $B'(p', r')$ gibt, der $B$ vollständig umschließt [BA91].

$$B(p,\ r) \mathrel{\widehat{=}} \{\ q \mid dist(p,\ q) < r\ \} \tag{4}$$

$B'$ umschließt $B$, wenn gilt:

$$B(p,\ r) \subseteq B'(p', r') \Leftrightarrow r' \geq dist(p,\ p')\ +\ r \tag{5}$$

Ein Kreis zum Zentrum $p \in G$ kann nur maximal sein, wenn sein Radius $r$ gleich dem kleinsten Abstand zwischen $p$ und der Form $G$ ist: $r\ =\ dist(p, G)$. Aus Gleichung 4 und 5 ergibt sich nach [BA91] für $S$ folgende Berechnung:

$$S\ =\ \{p \in G \mid \forall p' \neq p,\ dist(p',\ G) < dist(p,\ p') + dist(p, G)\} \tag{6}$$

Dies entspricht der Menge der Zentren der maximalen Kreise. Die Betrachtung ist in diesem Abschnitt wieder auf den zweidimensionalen Raum beschränkt, lässt sich aber analog mit einer maximalen Kugel für den dreidimensionalen Raum definieren.

## 4.2 Repräsentationsarten

Um die Medialen Achsen berechnen zu können muss zunächst festgelegt werden, in welcher Form das Objekt dargestellt ist. Es gibt zwei gängige Varianten, die Kontur des Objekts zu beschreiben: point-sampling oder die polygonale Approximation [AM97]. Von der Wahl der Repräsentationsart hängt es schließlich ab, welche Algorithmen für die Berechnung in Frage kommen. Die Approximation über point-sampling ist schon aus Abschnitt 3.2 bekannt und wird dort zur Vereinfachung von Voronoi-Zentren verwendet. Die polygonale Approximation unterscheidet sich nur in der Form vom point-sampling, dass Punktmengen zu einer Kontur mit Kanten

verbunden werden. Dabei ist es das Ziel, durch die Kantenverbindungen bei gleicher Genauigkeit mit weniger Punkten auszukommen. Es gibt Verfahren, die nicht mit der Kontur eines Objekts arbeiten. In diesem Fall wird direkt auf einer Kopie des Bildes gearbeitet und dem entsprechend auch andere Algorithmen benötigt.

## 4.3 Berechnungsgruppen

Man unterscheidet zwei Gruppen von Algorithmen, den *border-based* und den *region-based* Algorithmen [BA91]. Letztere arbeiten direkt auf der Abbildung des Objekts und reduzieren unter Berücksichtigung der topologischen Struktur stückweise die Anzahl der Pixel bzw. die Regionen. Charakteristisch sind die Bild-zu-Bild Transformationen, die benötigt werden, um die Form bis auf ihr Skelett zu komprimieren.

Die *border-based* Algorithmen werden dadurch charakterisiert, dass sie als Eingabe, in einer der oben beschriebenen Form, die Kontur des Objekts erhalten. Zu dieser Gruppe gehören auch die Algorithmen, welche mit Hilfe von Voronoi-Diagrammen die Skelettierung durchführen.

## 4.4 Skelettierung über Voronoi-Diagramme

Für den Zusammenhang zwischen Voronoi-Diagrammen und einer Skelettierung betrachtet man die Eckpunkte und Kanten einer polygonialen Approximation $P$ eines Objekts als Voronoi-Zentren. Daraus lässt sich das Voronoi-Diagramm $Vor(P)$ berechnen. Nun lässt sich zeigen, dass nach der Definition von maximalen Kreisen, deren Kreiszentren nur auf den Voronoi-Grenzen von $Vor(P)$ liegen können.
**Begründung:** Sollte es einen maximalen Kreis $B(p, r)$ mit $p \in V(z)$ geben, der nicht auf einer Voronoi-Grenze liegt, so gibt es einen maximalen Kreis $B'(p', r')$ mit $p' \in V(z)$ dessen Zentrum $p'$ in der Verlängerung von $p$ und $z$ auf einer Grenze von $V(z)$ liegt, mit $dist(p', z) \geq dist(p, p') + dist(p, z)$. Nach Gleichung 5 wird damit $B$ vom maximalen Kreis $B'$ umschlossen (siehe Abbildung 14 rechts) und ist somit nach Definition kein maximaler Kreis [BA91].

Daraus folgt, dass die Medialen Achsen ein Teilgraph des Voronoi-Diagramms zum Polygon $P$ sind. Es müssen nur noch die Voronoi-Grenzen entfernt werden, welche nicht der Definition von maximalen Kreisen entsprechen oder nicht innerhalb des Polygons liegen (siehe Abbildung 15). Zweiteres ist trivial.

Es gibt nur eine Art von Grenzen, die innerhalb des Polygons liegen, und nicht der Definition von maximalen Kreisen entsprechen. Sie sind dadurch charakterisiert,

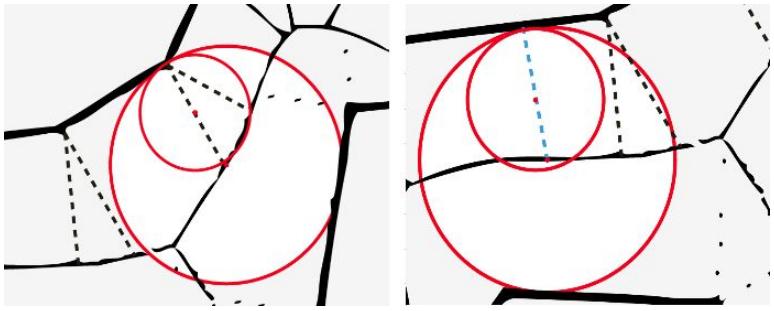

**Abbildung 14:** *Kreiszentren von nicht maximalen Kreisen: auf einer Grenze zu einem konkaven Eckpunkt (links), innerhalb einer Zelle (rechts).*

dass sie sich mit einem konkaven Eckpunkt $z$ schneiden.
**Begründung:** Alle Kreise, deren Zentren auf diesen Grenzen $g_i$ liegen, werden von einem maximalen Kreis mit dem Zentrum $q$ umschlossen. $q$ liegt auf dem Schnittpunkt der Grenze $g_i$ und der Grenze, die gegenüber des konkaven Eckpunkts $z$ liegt und hat damit maximalen Abstand zu $z$ (siehe Abbildung 14 links).

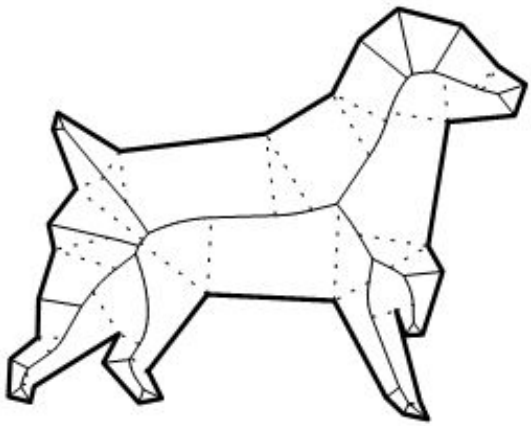

**Abbildung 15:** *Skelettierung zu einem Polygon. Die gestrichelten Kanten sind die zu entfernenden Voronoi-Grenzen [dBCvKO08].*

Mit dem in Abschnitt 3.3 vorgestellten Verfahren zur approximierten Berechnung von generalisierten Voronoi-Diagrammen ist also eine effiziente Grundlage geschaffen, auf den Diagrammen aufbauend, die Skelettierung zu bestimmen. Es lässt sich sogar zeigen, dass durch geringe Änderungen der Implementierung des Verfahrens die Skelettierung direkt berechnet wird. Über eine geschickte Farbwahl zu Per-feature Voronoi-Diagrammen können bei der Erstellung alle Voronoi-Grenzen

entfernt werden, auf denen keine maximalen Kreise liegen. Hierzu müssen alle Linienabschnitte, welche eine konkave Kurve beschreiben, jeweils einfarbig gehalten werden. Zusätzlich müssen alle Abschnitte von Zellen, welche außerhalb des Polygons liegen, in der Farbe des Hintergrunds gezeichnet werden.

# 5 Fazit

Die approximierte Berechnung von generalisierten Voronoi-Diagrammen mittels Computergrafikroutinen ist ein effizientes Verfahren, welches sich stark von anderen Algorithmen unterscheidet. Es arbeitet pixelbasiert und bestimmt die Grenzen über die entstehenden Zellen, anstatt die Grenzen direkt mathematisch zu berechnen. Durch diesen Unterschied lässt es sich sehr leicht implementieren, ist es gut verständlich und sowohl die Ergebnisse als auch das Verfahren selbst können ohne zusätzlichen Aufwand gut veranschaulicht und dargestellt werden. Die einzigen Einschränkungen liegen in der Höhe der Dimension und dem Grad der Ordnung des Voronoi-Diagramms. Es wurde nur für den zwei- und dreidimensionalen Raum entwickelt und berücksichtigt keine höhergradigen Ordnungen. Die Generalisierung in Bezug auf die Form der Zentren ist aber uneingeschränkt und der Gesamtfehler ist dabei bekannt und einstellbar.

Durch diese Eigenschaften ist es eine performante Berechnungsweise, welche die Anforderungen für viele Echtzeit-Anwendungen abdeckt. Während der Betrachtung des Zusammenhangs mit dem Thema Skelettierung wird zusätzlich deutlich, dass bei der zweidimensionalen Variante eine Modifikation ausreicht, um die Skelettierung mit dem Algorithmus direkt zu bestimmen.

Das Finden von Medialen Achsen, welche zusammen das Skelett eines Objekts bilden, ist kein Anwendungsbeispiel von Voronoi-Diagrammen, sondern ein ganz eigenes Thema aus der Bildverarbeitung. Es bietet viele Anwendungsmöglichkeiten und mit Themen wie der Schrifterkennung, ist es aktueller Bestandteil von Forschungen im Bereich Computer Vision und künstlicher Intelligenz.

Als Ausblick soll hier noch auf eine junge Entwicklung verwiesen werden. Unter dem Begriff *creative coding* vesteht man die Verbindung von Kunst und Technologie. Künstlerische Installationen, welche mit Hilfe von Technologie und programmierten Systemen aufgebaut sind, gewinnen für Austellungen und Events immer mehr an Bedeutung. In dieser Szene und dem Bereich der Spieleentwicklung kann die Skelettierung für die Analyse von *human body interaction* genutzt werden. Darunter versteht man die Steuerung durch Bewegungen des menschlichen Körpers,

welche von Kameras erfasst werden. Indem die menschliche Kontur zu einem Skelett vereinfacht wird, ist es möglich Bewegungen und Positionen von bestimmten Körperteilen leichter zu erfassen.

# Literatur


[ABMS98] Pankaj K. Agarwal, Mark de Berg, Jirí Matousek, and Otfried Schwarzkopf. Constructing levels in arrangements and higher order voronoi diagrams. *SIAM J. Comput.*, 27:654–667, June 1998.

[AM97] Dominique Attali and Annick Montanvert. Computing and simplifying 2d and 3d continuous skeletons. *Comput. Vis. Image Underst.*, 67:261–273, September 1997.

[BA91] Jonathan W. Brandt and V. Ralph Algazi. Continuous skeleton computation by voronoi diagram. *CVGIP: Image Underst.*, 55:329–338, May 1991.

[Blu67] Harry Blum. A Transformation for Extracting New Descriptors of Shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

[dBCvKO08] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 2008.

[GF87] Ronald N. Goldman and Daniel J. Filip. Conversion from bezier rectangles to bezier triangles. *Computer-Aided Design*, 19(1):25 – 27, 1987.

[GHBB04] Célia F. Guerra, Jan-Willem Handgraaf, Evert J. Baerends, and F. Matthias Bickelhaupt. Voronoi deformation density (VDD) charges: Assessment of the Mulliken, Bader, Hirshfeld, Weinhold, and VDD methods for charge analysis. *Journal of Computational Chemistry*, 25(2):189–210, 2004.

[HKL+99] Kenneth E. Hoff, III, John Keyser, Ming Lin, Dinesh Manocha, and Tim Culver. Fast computation of generalized voronoi diagrams using graphics hardware. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 277–286, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[KML95] Subodh Kumar, Dinesh Manocha, and Anselmo Lastra. Interactive display of large-scale nurbs models. In *Proceedings of the 1995*

*symposium on Interactive 3D graphics*, I3D '95, page 51ff., New York, NY, USA, 1995. ACM.

[Sug93] K. Sugihara. Approximation of generalized voronoi diagrams by ordinary voronoi diagrams. *CVGIP: Graphical Models and Image Processing*, 55(6):522 – 531, 1993.

[SWND07] Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 2.1 (6th Edition)*. Addison-Wesley Professional, 2007.