Masterarbeit

Entwurf eines integrierten Algorithmus zur Flächenrückführung und Punktreduktion in Punktwolken

Bearbeitet von:

Alexander Bucksch

am Lehrstuhl Graphische Systeme

an der Fakultät 1

der Brandenburgischen Technischen Universität

Cottbus, Februar 2005

Betreuer: Dipl.-Phys. Thomas Mangoldt
Verantwortlicher Professor: Prof. Dr. Winfried Kurth

Eidesstattliche Erklärung

Ich versichere, diese Arbeit selbst verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfsmittel bedient zu haben.

Inhaltsverzeichnis

Kurzfassung *7*

1. Einleitung *9*

1.1 Ziel **9**

1.2 Motivation **9**

1.3 Voraussetzungen **9**

1.4 Arbeitsumgebung **9**

1.5 Abriss **10**

2. Verarbeiten von Punktwolken *11*

2.1 Allgemeine Vorgehensweise **11**

2.2 Vorverarbeitung der Punktwolke **12**

2.3 Überblick über Flächenrückführungs-Algorithmen **15**

2.3.1 Raumunterteilung 16
2.3.2 Algorithmen auf Basis von Distanzfunktionen 33
2.3.3 Warping-Algorithmen 40
2.3.4 Oberflächenorientierte Wachstumsalgorithmen 41
2.3.5 Clustering-Methoden 44

2.4 Punktreduktion **45**

2.4.1 Adaptive Reduktion 45
2.4.2 Edge-Collapse-Verfahren [Hop1994] [Hop1996] 46

2.5 Diskussion und Schlussfolgerung **48**

3. Überblick über den Algorithmus *50*

3.1 Die drei Schritte im Überblick **51**

3.2 Einführung in die benötigten Parameter **52**

3.3 Zusammenführen der Ansichten **55**

3.4 Dekomposition der Szene **57**

3.4.1 Vorverarbeitung und Bestimmung der Startpunkte 57
3.4.2 Ablauf der Quadrikenbestimmung 59

3.5 Reduzieren und Vermaschen der Punkte **62**

3.5.1 Ebenen 62
3.5.2 3D-Quadriken 63

3.6 Zusammenführen der gefundenen Einzelflächen **66**

3.7 Einordnung des Algorithmus **67**

4. Entstehung des Verfahrens 69

4.1. Zusammenführen der Punktwolken 69

4.2. Quadrikenfindung 71

4.2.1 Entstehung des Verfahrens zur Parameterschätzung 71

4.2.2. Verwendung der Transformationen 72

4.2.3. Punkte-Durchlauf-Strategie und Abbruchkriterien 73

4.3. Punktreduktion 75

4.3.1. Kenntnis der polygonalen Gesamtrepräsentation 75

4.3.2. Reduktionswinkel α 76

4.3.3. Krümmungen 77

5. Ergebnisse 78

5.1 Grundkörper 78

5.2 Pflanzenteile 81

5.3 Vollständige Gerstenpflanze 88

5.4. Verhalten bei Störungen 90

5.5. Vergleich mit anderen Algorithmen 92

6. Fazit und Ausblick 95

6.1. Einordnung 95

6.2. Mögliche Erweiterungen 96

6.3. Einsatzgebiete 96

Anhang A – Glossar 97

Anhang B – Benutzung von Quadriken 101

Anhang C – Beispielanwendung 103

Abbildungsverzeichnis 108

Kurzfassung

Die vorliegende Arbeit befasst sich mit der Verarbeitung von Punktwolken. Dabei wurde der klassische Prozess zur Verarbeitung von Punktwolken wesentlich verändert. Der klassische Prozess setzt sich aus 3 Phasen zusammen:

- einer Vorbehandlung der Punktwolke zur Korrektur der Eingangsdaten,
- einem Algorithmus zur Bestimmung eines polygonalen Gitters über alle Punkte der Punktwolke und
- einem Algorithmus zur Reduktion der Daten.

Phase eins ist angepasst an das Aufnahmegerät. Sie wird teilweise manuell vollzogen, aber auch automatisch mittels Einsatz von Filtern.

Für Phase zwei wurde eine Vielzahl von Verfahren entwickelt [Boi1984] [EM1994] [ACK01]. Meist werden in diesen Ansätzen Eigenschaften der Delaunay-Triangulierung ausgenutzt. Seltener werden Flächen bestimmt.

In Phase drei wird anhand des bestimmten Gitternetzes die Punktwolke reduziert.

Der hier vorgestellte Algorithmus bestimmt Quadriken als Näherung der Fläche, welche ohne Kenntnis der Triangulierung reduziert werden können. Somit reduziert sich die Anzahl der zu triangulierenden Punkte drastisch. Des weiteren wird das Problem der dreidimensionalen Triangulierung auf ein zweidimensionales Problem in Polarkoordinaten zurückgeführt.

Im Vergleich zu anderen Algorithmen ist der Algorithmus in der Lage, Löcher zu modellieren und mehrere Objekte in einer Szene zu bestimmen und abzuarbeiten. Auch weist er eine hohe Robustheit gegenüber gestörten Eingangsdaten auf, welche die Vorverarbeitung unter bestimmten Voraussetzungen hinfällig werden lässt.

Somit beschreibt der Algorithmus im Vergleich zu klassischen Verfahren und beschreibt eine neue Vorgehensweise, welche sich bislang kaum behandelten und betrachteten Problemen zuwendet.

1. Einleitung

1.1 Ziel

Das Ziel dieser Arbeit ist es, ein Verfahren zu finden, Punktwolken, welche mittels strukturiertem Licht erzeugt wurden, in ein Gitternetz zu überführen. Bei diesem Verfahren sollen Flächen zweiter Ordnung die Oberfläche approximieren und eine adaptive Reduktion der Punkte ermöglichen. Am Beispiel von Gerstenpflanzen soll ein komplexes Modell erstellt werden.

1.2 Motivation

Die meisten heute bekannten Methoden sind in der Praxis nur mit starken Einschränkungen anzuwenden. Oftmals weisen sie schon bei idealen Datensätzen eklatante Schwächen wie Abflachungen oder Löcher im polygonalen Netz auf. Bei nicht vorverarbeiteten Daten versagen sie oft gänzlich. Als Beispiele können hierfür Boissonats Volumenbasierter Ansatz [Boi1984] und Edelsbrunner's and Mücke's $\alpha - shapes$ [EM1994] gesehen werden.

1.3 Voraussetzungen

Zur Verfügung stehen Punktwolken des 3D-Scanners DigiScan2000 der Firma RSI. Dieser erzeugt je vier Ansichten einer Gersten-Pflanze. Die Aufnahmen liegen im .dxf Format vor. Der verwendete Scanner arbeitet mit strukturiertem Licht in Verbindung mit dem Phasenshift-Verfahren [RSI2004].

1.4 Arbeitsumgebung

Als Arbeitsumgebung wurde zur Programmierung Microsoft C++ mit dem Visual Studio 6 als Entwicklungsumgebung gewählt. Für die Delaunay-Triangulierung und Konstruktion des Voronoi-Diagramms [CG2004] wurde die Template-Bibliothek CGAL (Computational Geometry Libary) eingesetzt. Diese ist frei erhältlich. Zur Programmierung der Benutzungsoberfläche kam Qt von Trolltech in der frei verfügbaren Version 2.4 zum Einsatz. Die grafischen Elemente wurden auf der OpenGL-Schnittstelle implementiert.

1.5 Abriss

Kapitel 2 erläutert die Thematik der Verarbeitung von Punktwolken anhand eines Drei-Schritte-Konzepts. Es werden Probleme und mögliche neue Ansätze diskutiert. Anhand einer Konstruktionsvorschrift führt Kapitel 3 an den hier beschriebenen Algorithmus heran. Ebenso wird hier eine Zuordnung zu den in Kapitel 2 vorgestellten Konzepten getroffen. Die Einzelheiten zur Wahl bestimmter Teilverfahren, die Begründung der Vorteile dieses Verfahrens und seine besonderen Eigenschaften finden sich in Kapitel 4. Eine Veranschaulichung der Ergebnisse anhand einiger erläuterter Illustrationen ist Kapitel 5 zu entnehmen. Abschließend gibt Kapitel 6 in einem Fazit einen Ausblick auf mögliche Erweiterungen.

2. Verarbeiten von Punktwolken

2.1 Allgemeine Vorgehensweise

Es ist möglich, die Verarbeitung von Punktwolken in drei wesentliche Schritte zu unterteilen:

1. Vorverarbeitung der Punktwolke
2. Flächenrückführung
3. Punktreduktion

Die Vorverarbeitung enthält Schritte wie das Beseitigen von Ausreißern, Überführen mehrerer Aufnahmen in eine Punktwolke und Glätten der erwarteten Oberfläche durch Anwendung von Filtern (z.B. Median). Die Anwendung von Filtern hat in diesem Kontext vorrangig den Zweck, Schwellwerte des zweiten Schritts einzuengen. Abschnitt 2.2 geht auf diese Aspekte genauer ein.

Im zweiten Schritt wird die eigentliche Flächenrückführung durchgeführt. In diesem Schritt wird versucht, einen Zusammenhang zwischen den einzelnen Punkten der Punktwolke herzustellen. Dies geschieht mit dem Ziel, die Oberfläche zu rekonstruieren. Es werden hierbei fünf verschiedene Grundkonzepte unterschieden. Diese werden in Abschnitt 2.3. ausführlich behandelt. Dieser Abschnitt ist der detaillierteste in diesem Kapitel.

Für den letzten Schritt der Punktreduktion ist in diesem Kontext fast ausschließlich der „edge-collapse“ Algorithmus von Hoppe [Hop1994] relevant. Alle bekannten Flächenrückführungsalgorithmen liefern ein trianguliertes, polygonales Gitter als Ergebnis. Der „edge-collapse“ Algorithmus hat sich über Jahre als ein Optimum zwischen Geschwindigkeit und Qualität etabliert[1]. Abschnitt 2.4 erläutert dieses Thema und greift mit einer Definition eines adaptiven Ansatzes dem später verwendeten Verfahren vor.

[1] Der Multiphasen-Ansatz [GS2002] zeigt nach Meinung des Autors noch bessere Ergebnisse. Da dieser Algorithmus sehr neu ist, gibt es noch keine dem Autor bekannte Kombination mit Flächenrückführungsalgorithmen.

Abschnitt 2.5 stellt die vorgestellten Verfahren in einen neuen Zusammenhang und begründet das gewählte Vorgehen.

2.2 Vorverarbeitung der Punktwolke

Bei der Vorverarbeitung der Punktwolke werden, unter Einbeziehung der Nachbarschaft, die Punkte ihrer Umgebung angeglichen. Meist geschieht dies unter Benutzung des Median-Filters. Da die Art der auftretenden Ausreißer den Mittelwert verfälscht, verbietet sich die Anwendung des Mittelwertfilters. Diese Vorverarbeitung besitzt vorrangig die Aufgabe, die Wahl von Entscheidungswerten in den Algorithmen einzuengen.

Oftmals besteht die Aufgabe darin, mehrere Punktwolken in einander zu überführen[2]. Dabei lassen sich verschiedene Verfahren zur Überführung anhand ihrer Transformation unterscheiden. Unterschieden werden:

1. rigide Verfahren: Bei diesen Verfahren werden nur Translation und Rotationen der Punktwolken zueinander berücksichtigt.

2. affine Verfahren: Bei Verwendung affiner Verfahren wird zusätzlich die Skalierung betrachtet.

3. projektive Verfahren: Projektive Verfahren brücksichtigen die perspektivische Verzerrung bei der Zusammenführung der Punktwolken.

4. elastische Verfahren: Bei dieser Art von Verfahren wird mit lokalen Transformationsparametern gearbeitet.

Im Besonderen sei in diesem Zusammenhang der Iterative Closest Point Algorithmus (ICP) von Besl und McKay [BM1992] erwähnt. Der ICP benötigt keine übereinstimmenden Punktepaare, um die Überführung vorzunehmen. Unter Verwendung von Quaternionen zur Bestimmung der Translations- und Rotationsparameter wird nur ein Vektor für die Rotationsmatrix benötigt.

[2] „Überführung“ wurde gewählt, da das englische Wort „Matching“ vermieden werden sollte und das deutsche Wort „Registrierung“ von Seiten des Autors als unpassend empfunden wird.

Der ICP-Algorithmus aus [Haa2003]

Definition Abstand: $d(p_i, X)$ ist der kleinste Abstand von p_i zu einem Punkt in der zu vergleichenden Punktmenge X.

Initialisierung: $k = 0$ (Iterationszähler), Punktwolke P, Rotationsmatrix R_k, Translationsvektor T_k, Quadratsumme der Überführungsfehler E_k.

Schritt 1: Zu jedem Punkt $p_i \in P$ wird der nächste Punkt in X berechnet. Die erhaltene Punktmenge sei Y_k.

Schritt 2: Die Überführungsparameter R_k, T_k und E_k für die beiden Punktmengen P_0 und Y_k werden bestimmt. Zur Berechnung von E_k ist ein Maß für den Überführungsfehler notwendig. Hierfür sind mehrere Maße anwendbar, meistens wird jedoch die euklidische Distanz gewählt.
Zur Schätzung der Transformationsparameter sind nach [AN2002] vier Verfahren im Einsatz[3]:

1. Die Transformationsschätzung mittels der Singulärwertzerlegung einer Matrix,
2. Die Transformationsschätzung mit Hilfe von Orthogonal-Matrizen,
3. Die Transformationsschätzung unter Verwendung des Einheitsquaternions,
4. Die Transformationsschätzung mit Dualzahl-Quaternion.

Schritt 3: Die Punktmenge $P_{k+1} = P_0 * R_k + T_k$ wird berechnet.

Schritt 4: Ist $0 < E_{k+1} - E_k <$ Grenzwert τ, terminiert der Algorithmus. Anderenfalls erhöht sich k um eins und Schritt eins bis vier werden wiederholt.

Der Grenzwert τ wird vor der Initialisierung bestimmt. Mit Hilfe von τ kann die Qualität der Überführung beeinflusst werden.

[3] Unter http://www.ais.fraunhofer.de/ARC/3D/download/diplom/node37.html sind die Verfahren im Detail erläutert.

Der ICP wurde an dieser Stelle erwähnt, da die Vermutung nahe liegt, dass die vorliegenden Punktwolken mit diesem Algorithmus in einander überführt wurden. Diese Vermutung begründet sich in den Artefakten am Stengel, welche ein elastisches Verfahren benötigen würden. Erläuternd hierzu ist Abbildung 1, welche die Artefakte am Stengel zeigt.

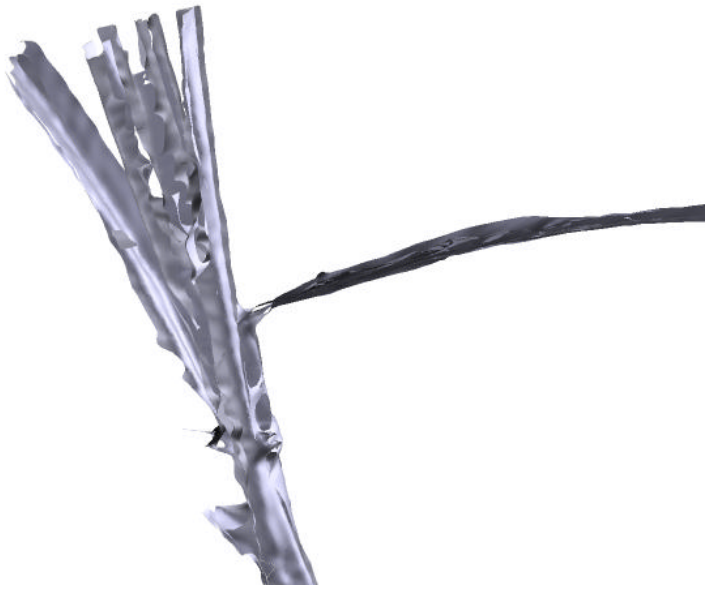


Abbildung 1: chaotische Verdrehungen der einzelnen Stengelaufnahmen in einem Bereich des Stengels

2.3 Überblick über Flächenrückführungs-Algorithmen

Das Problem der Flächenrückführung soll in diesem Abschnitt näher erläutert werden. Anhand einiger Algorithmen wird eine Einteilung in Grundkonzepte vorgenommen. Aus Abbildung 2 ist die gewählte Einteilung ersichtlich [Men2001]. Die ausgewählten Algorithmen sind immer als der älteste gefundene Vertreter eines Algorithmen-Typs zu verstehen.

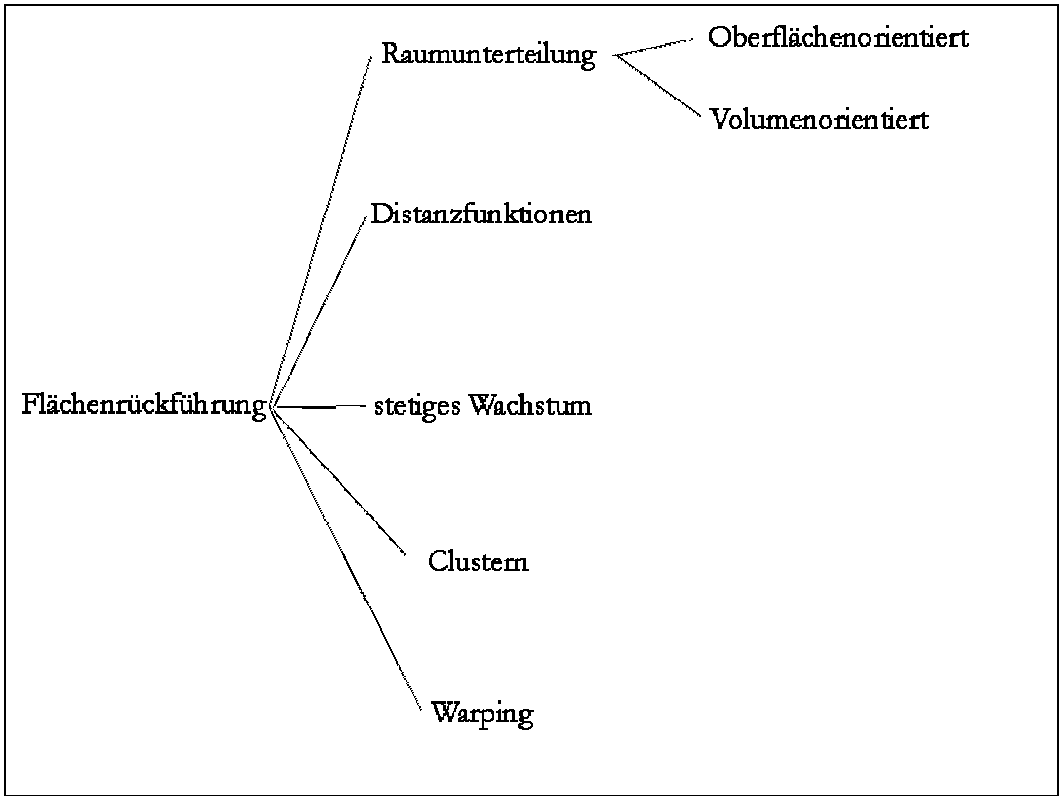


Abbildung 2: Einteilung von Flächenrückführungs-Algorithmen

2.3.1 Raumunterteilung

Algorithmen, welche auf Basis von Raumunterteilungen funktionieren, stellen die am häufigsten auftretende Klasse von Flächenrückführungs-Algorithmen dar. Sie lassen sich in zwei weitere Gruppen unterteilen: Zum einen in die Gruppe der oberflächenorientierten Algorithmen, und zum anderen in die Gruppe der volumenorientierten Algorithmen. Eine allgemeine Darstellung des Ablaufes solcher Verfahren ist aus Abbildung 3 zu entnehmen.

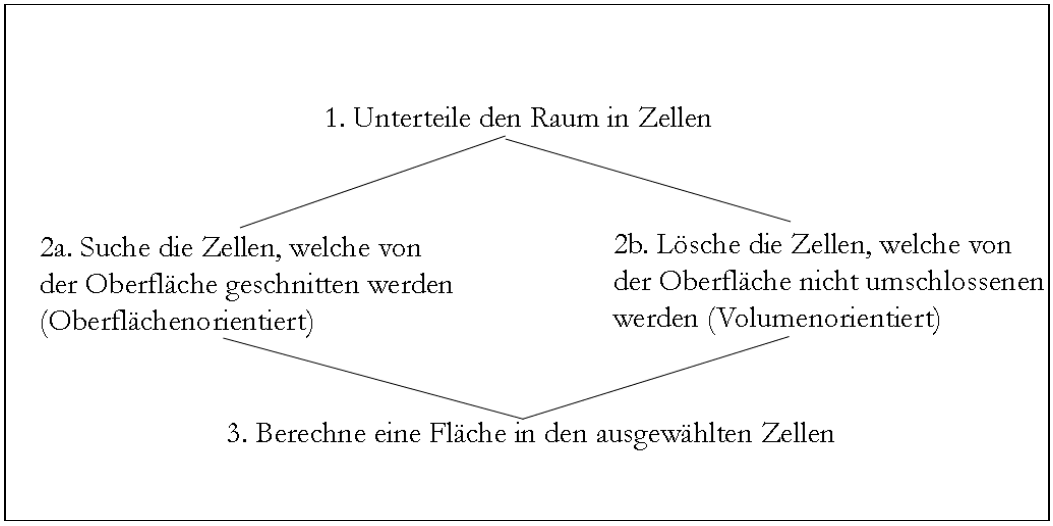


Abbildung 3: Schematischer Ablauf von Raumteilungs-Algorithmen

2.3.1.1 Oberflächenorientierte Algorithmen

Der Ansatz von Algorri und Schmitt [AS1996]

Algorri und Schmitt unterteilen den durch die Bounding-Box gegebenen Raum des Objektes in gleichmäßige, quaderförmige Voxel. Jedes Voxel entspricht hierbei einer Zelle im Raum. Im Folgenden werden all jene Voxel selektiert, welche mindestens einen Punkt der Punktwolke enthalten. Im dritten Schritt werden nun die nach außen gerichteten Seiten der Raumelemente als erste Näherung der Oberfläche genommen. Die ausgewählten quadratischen Seiten werden mittels diagonaler Unterteilung trianguliert. Das Ergebnis dieses „naiven“ Verfahrens kann durch Glättungsalgorithmen verbessert werden. Bessere Ergebniskorrekturen wurden mit Warping-Techniken erzielt, welche in Abschnitt 2.2.3 näher erläutert werden.

Ansatz von Hoppe et al. [Hop1994]

Der Ansatz von Hoppe et al. ist eine Kombination aus Raumaufteilungs- und Distanzfunktions-Algorithmus. An dieser Stelle wird auf die besonderen Elemente der Raumaufteilung eingegangen und in Abschnitt 2.2.4 auf die besondern Elemente der Distanzfunktionsberechnung. Das hier vorgeschlagene Vorgehen gliedert sich in die drei Phasen :

1. Generieren der Ausgangsvermaschung,
2. Optimieren der Vermaschung und
3. Glätten des Gitternetzes.

In der ersten Phase werden ebenfalls gleichmäßige, quaderförmige Voxel erzeugt. Es werden all jene Voxel selektiert, welche entgegengesetzte Vorzeichen besitzen. Dieses Vorgehen wird in Abbildung 4 veranschaulicht. Die Oberfläche wird mittels des Algorithmus von Wyvill et al. [WPW86] approximiert. Dieser ist dem Marching – Cubes-Algorithmus[4] sehr ähnlich.

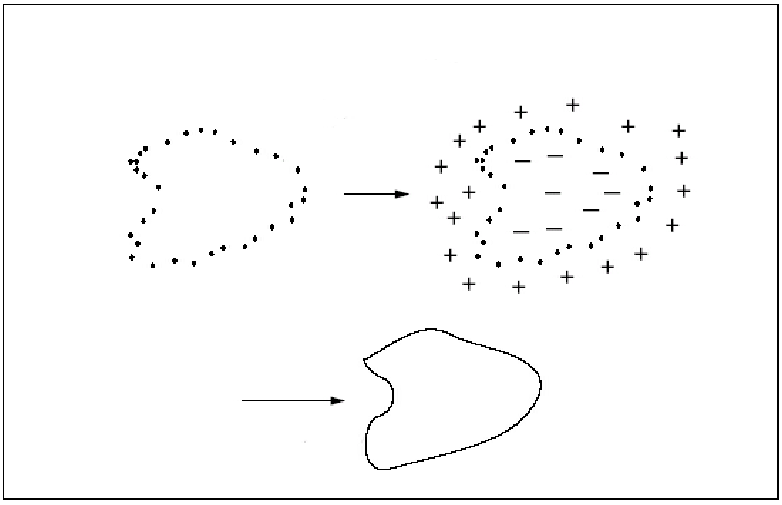


Abbildung 4: Generieren der Kontur nach Hoppe

Je nach Wahl der Größe der „Cubes“ können in diesem Algorithmus Löcher entstehen. Des weiteren werden Kanten stark abgeflacht.

[4] Erläuterungen im Anhang

Der Ansatz von Bajaj, Bernardini et al. [BBX1995]

Der Algorithmus von Bajaj, Bernardini et al. ist dem Algoihmus von Hoppe sehr ähnlich. Es werden auch hier die beiden Prinzipien der Raumunterteilung und der Distanzfunktion benutzt. Jedoch beschreibt bei diesem Prinzip die Distanzfunktion den Abstand zwischen Original und angenäherter Oberfläche. Die angenäherte Oberfläche wird mit $\alpha - solids$ berechnet, welche in Abschnitt 2.2.1.2 näher erklärt werden. Um die Punktwolke wird ein erster Tetraeder[5] gelegt. Dieser wird in vier weitere Tetraeder zerlegt, indem in jedes Zentrum der Tetraeder ein Vertex einfügt wird. Diese Vorgehensweise ist in Abbildung 4 dargestellt. Dabei sind rechts an den farbigen Flächen drei der vier Tetraeder erkennbar. Der vierte Tetraeder befindet sich auf der Rückseite. Dabei wurden in der rechten Darstellung die Außenseiten entfernt, um die Unterteilung im Inneren des Tetraeders sichtbar zu machen.

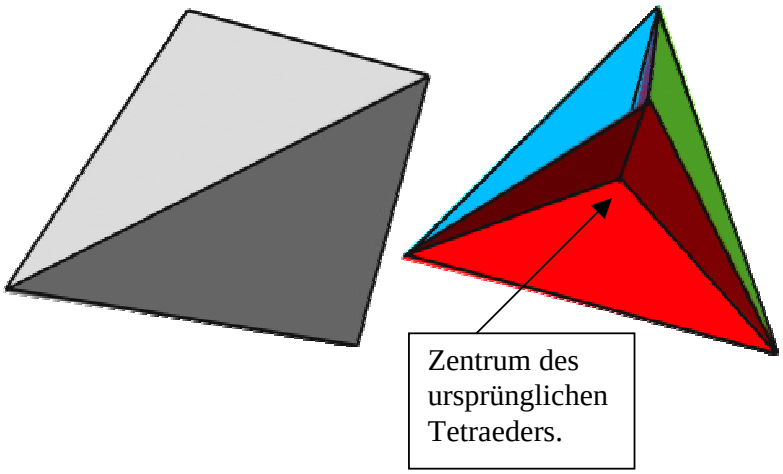


Abbildung 5: Tetraederunterteilung. Ein Tetraeder (links) und der unterteilte Tetraeder (rechts).

[5] Diese entstehen durch Delaunay-Zerlegung

Anhand des Vorzeichens der Distanzfunktion werden alle Tetraeder selektiert, welche von der anzunähernden Oberfläche durchzogen sind. Für jeden dieser Tetraeder wird eine Oberfläche mittels eines Bézier-Patches erzeugt. Für den Patch werden zwei Bernstein-Bezier-Formen benötigt:

$$f_\tau^D(p) = \sum_{|\lambda|=n} b_\lambda^D B_\lambda^n(\alpha)$$
$$f_\tau^F(p) = \sum_{|\lambda|=n} b_\lambda^F B_\lambda^n(\alpha)$$

In diesem Kontext beschreibt D die Originaloberfläche und F eine skalare Funktion[6], welche D näherungsweise beschreibt. Es wird für S auch die Bezeichnung „Oberfläche auf der Oberfläche“ (engl.: “surface on surface“) benutzt. τ ist der betrachtete Tetraeder. $B_\lambda^n(\alpha)$ ist das Bernsteinpolynom bezüglich der baryzentrischen Koordinaten α vom Grad n. f^D ist dabei der gesuchte Patch zur Näherung der Fläche. f^F wird zur Bewertung des Fehlers benutzt.

Um f^D anzunähern, werden für die Koeffizienten b_λ^D die durch die Distanzfunktion als positiv markierten Punkte verwendet. f^F wird mit der Methode der kleinsten Fehlerquadrate über alle im Tetraeder enthaltenen Datenpunkte bezüglich f^D approximiert. Im Folgenden wird der Näherungsfehler υ berechnet. Wenn die berechneten Koeffizienten einem drei- oder vierseitigen Patch zugeordnet werden können (siehe Abbildung 6), dann wird υ berechnet. Anderenfalls werden $\upsilon_\tau^D = \upsilon_\tau^F = \infty$ gesetzt.

$$\upsilon_\tau^D = \frac{\sqrt{\sum_{k=1}^{4} \sum_{p_i \in P \cap \tau} f_\tau^D(p_i)^2}}{Kard(P \cap \tau)}$$
$$\upsilon_\tau^F = \frac{\sqrt{\sum_{p_i \in P \cap \tau} f_\tau^F(p_i - w_i)^2}}{Kard(P \cap \tau)}$$

P: die gesamte Punktwolke

Kard(): Kardinalität bzw. Anzahl der Elemente

p_i: ein Punkt

[6] in Originalveröffentlichung wird keine nähere Angabe zu F gemacht. Zitat: „... while the values W are assumed sampled from some scalar function F...“ (aus [BBX1995] Seite 1 Spalte 2 Zeile 25)

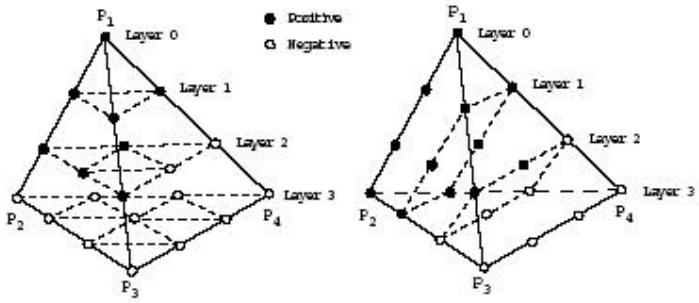


Abbildung 6: dreiseitiger Patch (links) und vierseitiger Patch (rechts)

Wenn $P \cap \tau = \varnothing$, dann wird $v_\tau^D = v_\tau^F = 0$. Der Algorithmus terminiert, wenn der größte auftretende Wert für v_τ^D und v_τ^F in allen Tetraedern kleiner als ein vorgegebener Wert ist. Anderenfalls beginnt der Prozess der Tetraederunterteilung von vorne.

Edelsbrunner's und Mücke's alpha-shapes, [Kas2000]

Edelsbrunner und Mücke benutzen ebenfalls Delaunay -Tetraeder zur Dekomposition des Raumes. Es gilt als bewiesen, dass der $\alpha - shape$ eine Teilmenge dieser Delaunay-Tetraederzerlegung ist [Kas2000]. Dies ist in Abbildung 7 nachzuvollziehen.

Darauffolgend werden Tetraeder, Dreiecke und Kanten entfernt, welche nicht der Hülle des Objektes entsprechen. Dies geschieht durch Benutzung sogenannter α-Kugeln. α beschreibt in diesem Zusammenhang den maximalen Radius einer Kugel. Für jeden Tetraeder, jede Kante und jedes Dreieck gilt nun, dass die kleinste sie umgebende Kugel kleiner sein muss als die α-Kugel. Diese Bedingung wird *„Alpha-Test"* genannt. Jeder Tetraeder, jede Kante und jedes Dreieck, dessen umgebende Kugel diesen Test nicht erfüllt, wird gelöscht. Ebenso gelöscht werden Inhalte von Kugeln, für die keine α-Kugel existiert, deren Volumen leer ist. Veranschaulicht wird dieses Vorgehen in Abbildung 7. Erkennbar sind, hier im zweidimensionalen Fall, die Alpha-Kreise im Alpha-Komplex ganz links. In der Mitte ist die Delaunay-Triangulierung zu sehen, und in der rechten Hälfte der Abbildung ist zu erkennen, dass der Alpha-Komplex die Hülle des Alpha-Shapes ist. Die Delaunay-Triangulierung beschreibt dagegen das Innere des Alpha-Shapes.

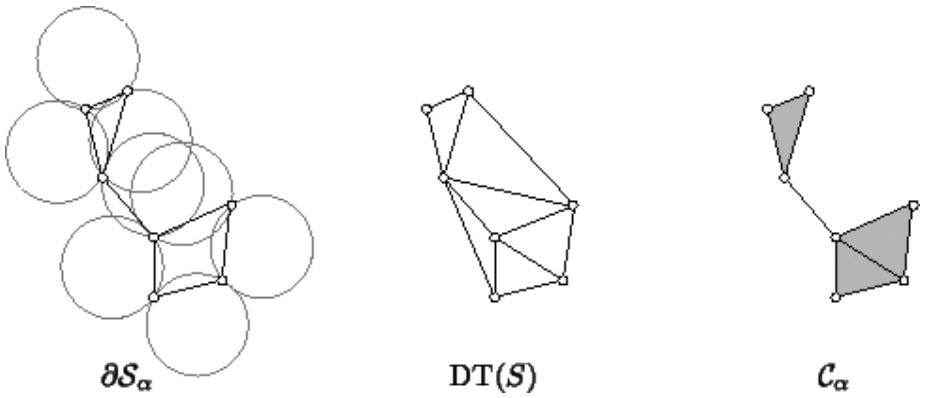


Abbildung 7: Zusammenhang Alpha-Complex (links), Delaunay-Triangulierung (mitte) und Alpha-Shape (rechts)

Das größte Problem dieser Verfahrensweise ist die Wahl eines geeigneten Alpha-Wertes. Zu kleine Alpha-Werte erzeugen Löcher im Modell, zu große Alpha-Werte lassen Details im Modell verschwinden. Dieses Problem ist in Abbildung 8 illustriert.

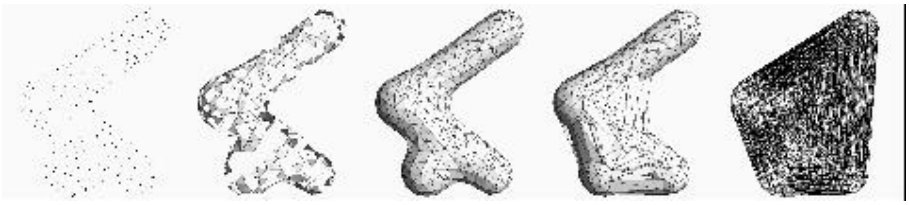


Abbildung 8: Variation der Alpha-Werte vom 0-shape (links) bis zum ∞-shape (rechts)

Attali's "Normalized Meshes" [Att1997] [Hra2003]

Auch Attali´s Ansatz der „Normalized Meshes" basiert auf einer Raumunterteilung mittels Delaunay –Tetraeder. Jedoch ist das resultierende Gitternetz eine Interpolation der Oberfläche.

„Normalized Meshes" sind polygonale Gitter einer Oberfläche *S*, der eine Anzahl *P* Näherungspunkte zugeordnet ist. Dabei besteht *S* aus den Delaunay *k*-simplices $T_j = [p_1,..., p_k]$ für die gilt $p_i \in P$. Im folgenden wird ein Punkt $m \in S$ gesucht, für den gilt $d(m, p_1) = ... = d(m, p_k) = d(m, P)$.

Laut seiner Definition ist das „Normalized Mesh" ein Teil des Delaunay – Diagramms.

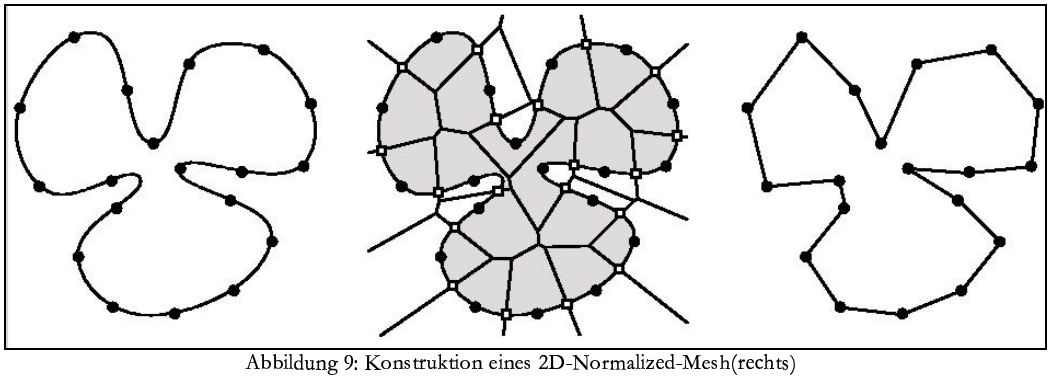


Abbildung 9: Konstruktion eines 2D-Normalized-Mesh(rechts) aus einer Kurve mit Näherungspunkten (links) und deren Voronoi-Zellen (Mitte).

Um die Suche nach dem „Normalized Mesh" zu vereinfachen, wird die Annahme getroffen, dass S eine r-reguläre Form ist. In diesem Kontext wird von einer r-regulären Form *X* gesprochen, wenn diese mit einer Kreisform mit Radius $r > 0$ morphologisch geöffnet und geschlossen wird.

$$X = (X \Theta rB_0) \oplus rB_0 = (X \oplus rB_0) \Theta rB_0$$

Θ : morphologische Erosion

$\oplus$: morphologische Dilatation

B_0 : Einheitskugel

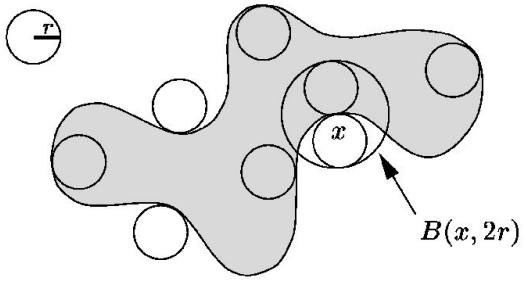


Abbildung 10: Morphologische Operation auf der Oberfläche

Das "Voronoi Filtering" von Amenta und Bern [AB1998]

Das Voronoi Filtering beruht wiederum auf der Delaunay-Tetraeder-Zerlegung. Amenta und Bern geben eine kurze 2D-Konstruktionsanleitung für den Algorithmus, bevor auf die Probleme im 3D-Raum eingegangen wird. Ähnlich wie bei den $\alpha - shapes$ existiert ein Parameter, der sukzessive verkleinert werden muss, bis das Ergebnis als gut erscheint.

1. Berechne das Voronoi-Diagramm der Punktwolke S.

2. Für jeden Punkt der Wolke muß gelten:

   - Wenn ein Punkt $s \in S$ nicht zur konvexen Hülle von S zählt, dann sei der Pol p^+ der Voronoi-Vertex der s umgebenden Voronoi-Zelle, welcher zu s am entferntesten ist. In diesem Kontext sei ein Polpaar p bestehend aus den beiden entferntesten Vertices einer Voronoi-Zelle, die einem Punkt s zugeordnet sind. Abbildung 11 zeigt ein vereinfachtes Beispiel der Pole.

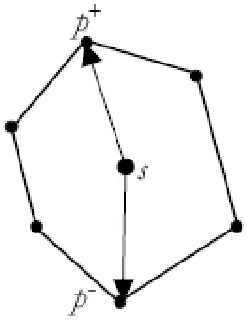


Abbildung 11: Die beiden Pole eines Punktes s im 2D-Beispiel

- Wenn s auf der konvexen Hülle von S liegt, so soll der Pol p^+ ein unendlich weit entfernter Punkt außerhalb der konvexen Hülle von S sein. Dabei entspricht die Richtung von $\overrightarrow{sp^+}$ der durchschnittlichen Normalenrichtung aller sich in s treffenden Hüllflächen.

- Unter allen Vertices des Voronoidiagramms von S sei als negativer Pol p^- der Punkt mit dem größten Abstand zu s gewählt, der folgender Bedingung genügt (vergleiche Abbildung 11):

$$(\sphericalangle p^+ sp) > (\pi / 2)$$

3. Im folgenden werden alle Pole p^+ und p^- der Punktwolke hinzugefügt, mit Ausnahme der Pole p^+, die dem infiniten Punkt zugeordnet sind. Über die entstandene neue Punktwolke wird wiederum eine Delaunay-Triangulierung berechnet.

4. Der vierte Schritt bewirkt das eigentliche Voronoi-Filtering. Es werden alle Dreiecke beibehalten, welche aus Vertices der Original-Punktwolke bestehen.

5. In einem zweiten Filtervorgang, dem Normalen-Filter, werden all jene Dreiecke als entfernbar markiert, deren Normale in einem Vertex und ihr zugehöriger Pol einen Winkel α einschliessen, welcher grösser als der Eingabe-Winkel θ ist. Des weiteren wird im Folgenden die Bedingung gestellt, dass jeder weitere Vertex des zu entfernenden Dreiecks einen Winkel $\alpha \geq \frac{3}{2}\theta$ zu seinem Pol aufweist. Sind beide Bedingungen erfüllt, wird das Dreieck entfernt.

6. In einem letzten Schritt wird die Konsistenz (Umlaufsinn der Polygone / Richtung der Normalen) des Polygongitters überprüft und korrigiert.

Der wohl größte Nachteil dieses Algorithmus ist die mehrfache Notwendigkeit der Berechnung der Delaunay-Triangulierung. Jedoch ist ihm historisch größere Bedeutung zuzuordnen, da er die Grundlage zu einer Vielzahl von Algorithmen bildet. Die Verwandschaft zu diesem Algorithmus wird bei den weiteren vorgestellten Algorithmen auffällig sein.

Der "Short Crust" Algorithmus von Amenta and Choi[7] *[AC1999]*

Der Short Crust Algorithmus ist als eine Weiterentwicklung des Voronoi-Filtering zu interpretieren. Er benötigt im Gegensatz zum Voronoi-Filtering nur eine einmalige Delaunay-Triangulierung.

Um dies zu erreichen, wird in jedem Punkt eine Normale bestimmt. Die Normale wird unter Hinzunahme der im Voronoi Filtering [AB1998] vorgestellten Pole bestimmt. Für jeden Punkt p der Punktwolke wird der Voronoi-Vertex v, welcher am weitesten von einem Punkt p entfernt ist, als Pol bestimmt. $\overline{vp}$ sei von nun an die Normale zur Oberfläche S des Objektes in p. Um jede dieser Normalen wird ein invertierter Doppelkegel C_p mit der Spitze in p berechnet. Abbildung 12 zeigt einen solchen Kegel. Hierbei beträgt der Winkel zwischen der Normalen und der Kegelhülle $\pi/2 - \theta$. θ ist der vom Benutzer bestimmte Eingabeparameter. Selektiert werden nun die Kanten e des Voronoi-Diagramms, welche die invertierten Doppelkegel C_{p_i} aller drei Vertices eines Delaunay - Dreiecks schneiden. Die so gefundenen Dreiecke bilden eine Ausgangsmenge T zur Rekonstuktion von S.

[7] Es bleibt anzumerken, dass an dieser Methodik noch immer aktuell geforscht wird. Fortgeführt wird die Arbeit von Tamal K. Dey an der Ohio State University, http://www.cse.ohio-state.edu/~tamaldey/cocone.html

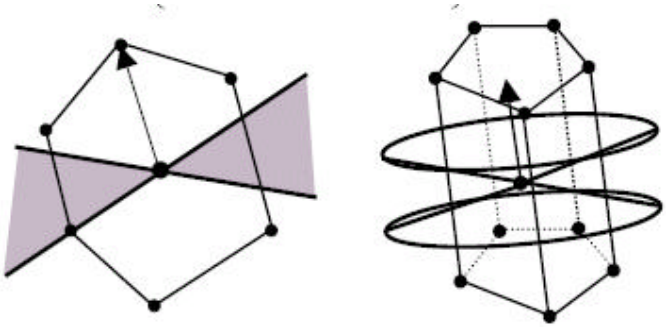


Abbildung 12: Beispiel des invertierten Doppelkegels (rechts) und ein vereinfachtes 2D-Beispiel (links). Der Pfeil entspricht hierbei der Normalen eines Punktes

Scharfe Kanten in T sind durch einen im Uhrzeigersinn gemessenen Winkel zwischen zwei Dreiecken definiert. Ist dieser größer $\frac{3}{2}\pi$, so gilt eine Kante als scharf. Dreiecke, welche dieser Definition entsprechen, werden Kantendreiecke genannt. Dreiecke, welche an Kantendreiecke grenzen und selber kein Kantendreieck sind, werden entfernt. Um zu verhindern, dass alle Dreiecke rekursiv entfernt werden, wird der *„Umbrella-Check“* durchgeführt. Dieser bewirkt, dass nur Dreiecke entfernt werden, deren Vertices mit ihren angrenzenden Vertices eine ebene Scheibe aufspannen.

“Umbrella Filter Algorithm” von Adamy, Giesen und John [AGJ00] [AGJ01]

Wie viele andere Algorithmen beginnt auch dieser mit einer Delaunay-Tetraederaufteilung der Punktwolke *P*. Im Folgenden wird für jeden Punkt ein „Schirm“ berechnet. So ein Schirm besteht aus einer Aneinanderreihung von Dreiecken in der Form einer Scheibe mit einem Punkt p in der Mitte der Scheibe. Darauffolgend werden alle Dreiecke, welche nicht zu einem Schirm gehören, gelöscht. Überflüssige Dreiecke werden in einer auf Topologien basierenden Stufe eliminiert. Entstehende Löcher werden mit einem Holefilling - Algorithmus geschlossen.

Der Schirm wird mit Hilfe von sogenannten Gabriel-Dreiecken[8] geformt. Die Dreiecke werden nach folgender Vorschrift ausgewählt:

[8] Gabriel-Dreiecke entstehen beim Zeichnen eines Gabriel-Graphen. In diesem speziellen Fall können sie der Delaunay - Triangulierung gleich sein. Der Gabriel-Graph ist dem relativen Nachbarschaftsgraphen ähnlich, jedoch benutzt er den Durchmesser anstatt des Radius als Distanzmaß. Erläuterungen hierzu sind dem Anhang zu entnehmen.

Die untere λ – Intervall-Grenze wird schrittweise vergrößert, bis die Dreiecke einen Schirm bilden. Die λ – Intervall-Grenze bestehend aus λ_1 und λ_2 eines Dreiecks t wird mit

$$\lambda_i = diam(t) / diam(t_i), i = 1,2$$

und t_i, den beiden Schnitt – Tetraedern der Delaunay-Triangulierung berechnet. λ_1 und λ_2 sind die obere und untere Grenze des Intervalls. $diam()$ soll hier eine Abkürzung für Durchmesser sein.

2.3.1.2 Volumenorientierte Ansätze

Boissonnat's volumenbasierter Ansatz [Boi1984] nach [Men2001]

Ausgangspunkt dieses einfachen Algorithmus ist wiederum die Delaunay-Tetraederunterteilung. Im Anschluss daran werden auf zwei Arten die Polygone unvollständiger Tetraeder entfernt:

1. Polygone aller unvollständigen Tetraeder bestehend aus zwei Flächen, fünf Kanten und vier Punkten und

2. Polygone aller unvollständigen Tetraeder bestehend aus einer Fläche, drei Kanten und drei Punkten.

Aufgrund dieser Regel können, wie in Abbildung 13 dargestellt, nur Objekte ohne Löcher rekonstruiert werden. Diese Tetraeder werden nun iterativ entfernt. Um einen Abbruch der Iteration zu gewährleisten, wird ein Abstandsmaß benötigt. Dieses entspricht dem Abstand d einer Tetraederseitenfläche zu der den Tetraeder umgebenden kleinsten Kugel. Der Abbruch erfolgt wiederum nach zwei Regeln:

1. Wenn alle Punkte auf der Oberfläche liegen und

2. wenn d die Summe $\frac{\sum d_i}{i}$ nicht verkleinert. d_i sind die Abstände aller an der Außenseite des Objektes liegenden Tetraederseiten.

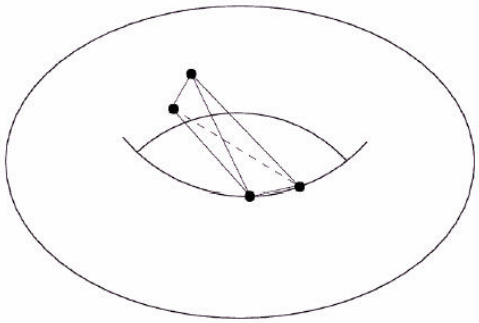


Abbildung 13: Beispiel für ein mit diesem Ansatz nicht entfernbares Polyeder

Der erweiterte Gabriel-Hypergraph von Attene und Spagnuolo [AS2001]

Der hier vorgestellte Ansatz kann als eine direkte Erweiterung des Ansatzes von Boissonnat gesehen werden. Zur Modellierung von Löchern in Objekten wird eine Unterteilung eingeführt. So wird bei diesem Ansatz bei der Entfernung von Dreiecken aus der Delaunay-Triangulierung der Punktwolke zwischen „*constraint*“ und „*non-constraint*“ unterschieden. „*Non-constraint*“ entspricht den Kriterien des Ansatzes von Boissonnat.

Die „*constraint*“ Eliminierung der Dreiecke verwendet als Ausgangspunkt den erweiterten Gabriel-Hypergraph (EGH) und den euklidisch minimal aufspannenden Baum (EMAB). Um Dreiecke aus der Delaunay-Tetraederzerlegung zu entfernen, müssen folgende Eigenschaften erfüllt sein:

1. unvollständige Tetraeder mit zwei Flächen, fünf Kanten und vier Punkten (vgl. [Boi1984])

2. unvollständige Tetraeder mit einer Fläche, drei Kanten und drei Punkten (vgl. [Boi1984])

3. unvollständige Tetraeder mit nur einer Fläche auf der Außenseite der Tetraederaufteilung dürfen nicht zum EGH gehören.

4. unvollständige Tetraeder mit zwei Flächen auf der Außenseite der Tetraederunterteilung dürfen weder zum EGH, noch zum EMAB gehören.

Um im weiteren Verlauf Löcher modellieren zu können, muss eine gesonderte Regel auf die entstandene Oberfläche angewandt werden. So gilt für Löcher die Vorschrift, dass alle Tetraeder, deren Entfernen aus der Tetraederzerlegung eine neue Kante im EMAB erzeugt, potentielle Löcher sind. Alle Tetraeder, welche diesem Kriterium Genüge leisten, werden entfernt. Kanten des EMAB, welche die Eigenschaft besitzen, nicht zur äußeren Fläche der Tetraederraumteilung zu gehören, und die Möglichkeit besitzen, ein Loch zu erzeugen, werden samt ihrer dazugehörigen Flächen gelöscht. Dies ist ein iterativer Prozess. Dazu erläuternd ist Abbildung 14.

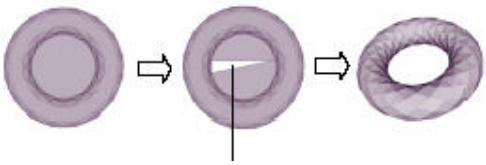


Abbildung 14: Modellieren der Löcher nach Attene und Spanguolo

Ansatz nach Isselhard, Brunnett und Schreiber [IBS1997]nach [Jem204]

Isselhard, Brunnet und Schreiber befassen sich ebenfalls damit, den Algorithmus von Boissonnat zu erweitern, um mit ihm Löcher modellieren zu können. Der Algorithmus unterscheidet sich von seinem Vorgänger darin, dass er neben unvollständigen Tetraedern der folgenden Eigenschaften auch vollständige Tetraeder entfernen kann:

- Eine Fläche und vier Vertices oder
- Drei Flächen, als auch
- Vier Flächen.

Über die Löschung eines Tetraeders wird mittels einer Eliminierungsfunktion entschieden. Diese ist definiert als der maximale Wert des Abbruchkriteriums [Boi1984] aller verbliebenen und entfernbaren Tetraeder. Diese Funktion weist charakteristische Sprünge auf. Einer dieser Sprünge zeigt an, ob die gewünschte Oberfläche erreicht ist. Dieser mögliche Abbruch ist die Grundlage dafür, auch Löcher zu modellieren. Im weiteren Verlauf entspricht der Algorithmus den Regeln von Boissonnat.

Der Ansatz mittels Gamma–Indikator von Veltkamp[Vel1992] [Vel1995]

Der γ–Indikator basiert auf dem γ–Graphen. Dieser Graph vereinigt viele bekannte Graphentypen[9] in sich, auch die in diesem Zusammenhang wichtige Delaunay-Tetraederunterteilung. Der γ–Indikator wird wie folgt berechnet:

v_i, v_j, v_k sind die Vertices eines Dreieckes der durch den γ–Graphen entstandenen Tetraederunterteilung[10]. v_l ist der dem Dreieck gegenüberliegende Vertex desselben Tetraeders. Wenn R der Radius der Kugel durch v_i, v_j, v_k und v_l, und r Radius des Kreises durch v_i, v_j, v_k ist, dann folgt daraus der γ–Indikator d:

$$d = 1 - \frac{r}{R}$$

[9] Namentlich sind dies die konvexe Hülle, die Delaunay-Triangulation und im 2D der Gabriel-Graph und das β-Skeleton

[10] In diesem Fall soll der γ–Graph der Unterteilung nach Delaunay entsprechen

d besitzt die Eigenschaft, positiv zu sein, wenn das Zentrum der Kugel innerhalb des Tetraeders liegt, und negativ, falls sich das Zentrum außerhalb des Tetraeders befindet. In einem zweidimensionalen Beispiel ist diese Eigenschaft in Abbildung 15 veranschaulicht.

Im Folgenden werden mit dem Regeln von Boissonnat entfernbare Tetraederflächen bestimmt. Als Auswahlregel für die zu entfernenden Dreiecke aus einem Tetraeder gilt nun:

„Entferne alle Dreiecke eines Tetraeders, so dass die Summe der entfernten γ–Indikatoren maximal wird“

Als Abbruchkriterien werden dieselben wie bei Boissonnat gewählt.

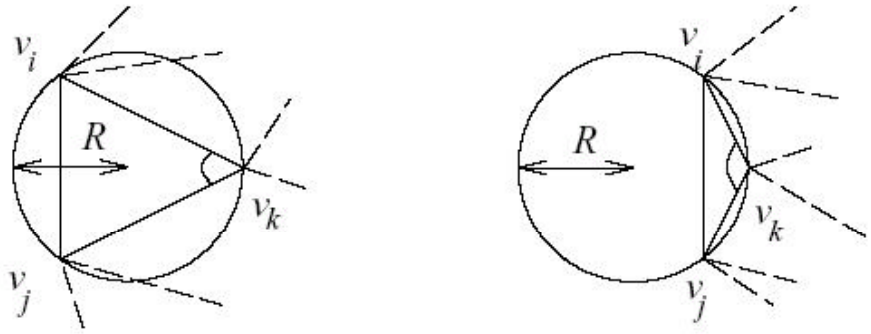


Abbildung 15: 2D Beispiel für den γ–Indikator. links: positiv rechts: negativ

Der Ansatz von Schreiber and Brunnett [11] *[SB1997] nach [Men2001]*

Dieser Ansatz nutzt die Dualität zwischen Voronoi-Diagramm und Delaunay-Tetraederunterteilung aus.

1. Jeder Vertex des Voronoi-Diagramms entspricht einem Tetraederzentrum
2. Jede Kante des Voronoi-Diagramms gehört zu zwei benachbarten Tetraederflächen

Der Ablauf des Algorithmus:

1. Schritt: Berechne die Delaunay-Tetraederunterteilung und das Voronoi-Diagramm.
2. Schritt: Die Selektion der Tetraeder wird mittels des minimal aufspannenden Baumes erreicht. Unter Anwendung einer einfachen Strategie wird der Baum in mehrere disjunkte Teilbäume (Regionen) zerlegt. Hierbei werden zwei Regeln angewendet:

- Entferne alle Kanten, deren Endpunkt nicht im umgebenden Kreis der beiden Dreiecke des anderen Punktes liegt

- Eine Kante wird entfernt, wenn sie kürzer ist als der Radius beider umgebender Kreise der angrenzenden Dreiecke.

Die Anzahl der zu entfernenden Kanten ist durch einen Abbruchparameter begrenzt. Dieser ist eine von einem Benutzer wählbare minimale Kantenlänge. Die entstandenen Regionen werden als „intern“ und „extern“ markiert. Um dies durchzuführen, werden die Teilbäume durchlaufen, beginnend an einem Punkt auf der konvexen Hülle. An diesen Startpunkt wird weiterhin die Bedingung gestellt, an exakt zwei Regionen zu grenzen. Zusätzlich wird das Komplement der konvexen Hülle als neue Region eingeführt. Regionen, die sich von dem Komplement der konvexen Hülle unterscheiden, werden als „intern“ markiert. Benachbarte Regionen, welche an zwei Regionen grenzen, werden ebenfalls als “intern“ markiert. Wenn alle Regionen korrekt markiert sind, wird die Oberfläche aus der äußeren Hülle aller internen Regionen rekonstruiert.

[11] Dieser Algorithmus liegt nur in einer sehr knappen allgemeinen Beschreibung vor. Das Original war nicht zu beschaffen, da der Herausgeber auf eine e-mail Nachfrage nicht reagierte.

„alpha–solids“ nach Bajaj, Bernardini et al. [BBX1995] [BBX1997]

Bajaj, Bernardini et al. beschreiben ihre α–solids als eine Art “reguläre α–shapes”. Eines der Hauptprobleme bei der Verwendung von α–shapes ist, daß sie aus verschiedenen Elementen bestehen können. Sie können Tetraeder, Dreiecke, Polylinien oder auch isolierte Punkte sein. Ein α–solid ist definiert als ein Teil der Dreiecke eines Tetraeders t aus der Menge aller Tetraeder T. Ein α–solid ist durch eine von zwei Eigenschaften definiert:

1. Dreiecke eines Tetraeders gehören zum α–shape oder
2. sie bilden zusammen mit den anderen Dreiecken des α–shapes eine kontinuierliche Oberfläche.

Die Ermittlung solcher Tetraederflächen ist einfach durchzuführen, ohne den ganzen α–shape durchlaufen zu müssen. Begonnen wird die Suche an einem Tetraeder t_1 mit infinitem Punkt. Folgend wird ein angrenzender Tetraeder t_2 betrachtet. Wenn Flächen von t_2 zum α–shape gehören, wird er als „intern“ markiert, anderenfalls als „extern“. Alle externen Tetraeder werden in einer Warteschlange zwischengespeichert. Diese Warteschlange wird dann nach dem Prinzip von Boissonnat abgearbeitet.

2.3.2 Algorithmen auf Basis von Distanzfunktionen

Algorithmen, welche Distanzfunktionen verwenden, stellen die zweite große Algorithmengruppe dar. Distanzfunktionen liefern als Ergebnis den Abstand eines Punktes zur Oberfläche. Für geschlossene Körper kann das Ergebnis positiv oder negativ sein. Negative Ergebnisse weisen einen Punkt als innerhalb des Objektes aus. Ein positives Ergebnis zeigt an, daß der Punkt außerhalb des Objektes liegt. Im vorangegangenen Kapitel wurde schon auf die Distanzfunktionen kurz eingegangen. Außer dem bereits kurz erwähnten Marching-Cubes-Algorithmus spielen die medialen Achsen eine wesentliche Rolle in der Rekonstruktion von Oberflächen. Sie können auch zur Verbesserung von Ergebnissen anderer Verfahren eingesetzt werden.

Der Ansatz von Hoppe et al.[Hop1994]

In Hoppes Ansatz wird als Distanzfunktion der Abstand $A(p)$ eines Punktes p der Punktwolke zu seiner Projektion z auf eine geschätzte Tangentialebene T_p der Oberfläche verstanden. Für die Tangentialebene gilt, dass ihr Zentrum o_i so nah wie möglich am Punkt p liegen sollte. Daraus leitet sich folgende Bedingung ab:

$$A(p) = (p - o_i) \bullet \hat{n}_i$$

Verdeutlicht wird das Prinzip in Abbildung 16.

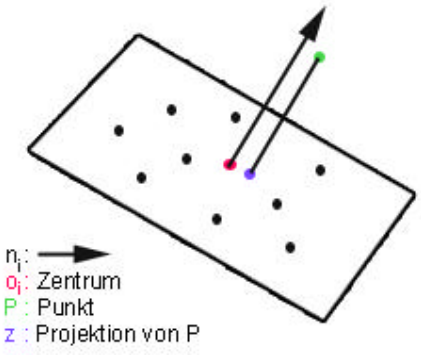


Abbildung 16: Prinzip der Distanzermittlung nach Hoppe

Die Tangentialebene wird dabei lediglich als Regression über Punkte in einer frei wählbaren Nachbarschaft berechnet. So eine Tangentialebene wird für jeden Punkt der Punktwolke berechnet. Um die Orientierung der Tangentialebenen und die Oberfläche herauszufinden, wird ein Riemanngraph durch alle Zentren der Tangentialebenen gelegt. Es wird festgelegt[12], dass die gesuchte Oberfläche nahe an den Kanten des Riemanngraphen verläuft. Zwei Zentren o_i, o_j werden miteinander durch eine Kante verbunden, wenn beide sich in der nächsten Nachbarschaft der jeweils anderen befinden. Diese Kante wird mit $1 - n_i \bullet n_j$ gewichtet. Der Punkt mit dem größten z-Wert wird als Startwert zur Bestimmung der Orientierungen der Ebenen gewählt. Seine Tangentialebene wird in Richtung der positiven z-Achse festgelegt. Nun wird entlang des MAB des

[12] In der Veröffentlichung [Hop1994] wird kein Beweis für die Richtigkeit dieser Aussage geführt.

Riemann-Graphen[13] traversiert. Über die Orientierung der Normalen in den Punkten entscheidet die Richtungsänderung im Graphen.

Der Ansatz von Roth and Wibowoo[RW1997]

Der Ansatz von Roth und Wibowoo benutzt ein Raster aus Voxeln, zu dem die Distanz bestimmt wird. Sie beschreiben ihren Algorithmus wie folgt:

1. Setze die frei wählbare Voxeldimension fest.

2. Weise jedem Punkt der Punktwolke ein ihn umgebendes Voxel zu. Wenn es noch nicht existiert, dann erzeuge es neu.

3. Entferne Störpunkte. Aufgrund der Messung entstandene Störpunkte sind definiert als Punkte, deren Voxel mehrmals von einem Strahl getroffen werden. Der Strahl wird vom Sensor-Standpunkt auf das Objekt gesendet. Die Strahlen werden zu jedem besetzten Voxel gesendet. Erläuternd hierzu ist Abbildung 17. Der weiße Punkt in dem durchkreuzten Rasterelement zeigt einen solchen Störpunkt.

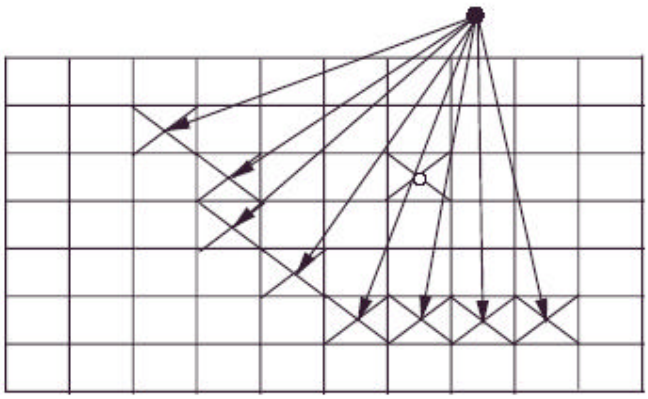


Abbildung 17: Entfernen der Störpunkte nach Roth und Winbowoo

[13] Siehe Anhang

4. Bestimme die Normale für jeden Punkt. In diesem Fall werden nur die beiden nächsten Nachbarn dazu verwendet. Die Schwierigkeit besteht darin, die Orientierung der Normalen zu bestimmen. Zu diesem Zweck wird aus allen sechs Richtungen auf die Punktwolke geblickt. Alle sichtbaren Punkte erhalten Normalen entgegen der Blickrichtung. Alle Normalen der nicht sichtbaren Punkte werden aus ihren Nachbarn gemittelt. Ein zweidimensionales Beispiel ist Abbildung 17 zu entnehmen. Die ausgefüllten Punkte entsprechen hier den nichtsichtbaren Punkten. Die Pfeile geben die Blickrichtung an.

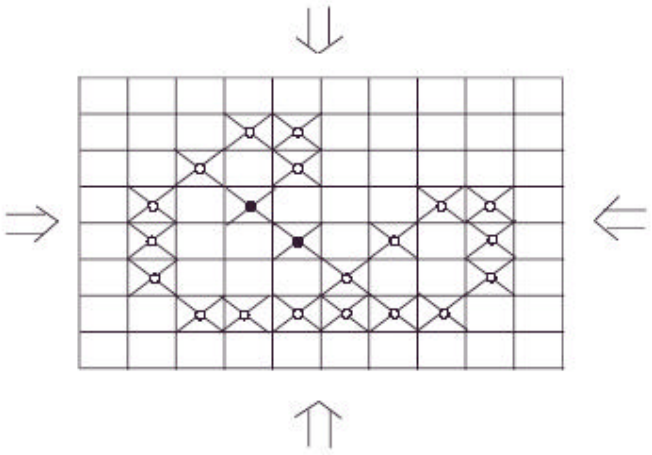


Abbildung 18: Unterscheidunge zwischen sichtbaren (weiss) und nicht sichtbaren Punkten (schwarz). Die jeweilige Blickrichtung ist als Pfeil dargestellt.

5. Zur Unterdrückung des Sensorrauschens, führe einen Normalen-Ausgleich durch. Entweder aufgrund der Mittelung vorhandener Normalen in den Punkten, oder aufgrund von gemittelten Flächennormalen in einer frei wählbaren Nachbarschaft. Roth und Wibowoo machen an dieser Stelle keinen genaueren Angaben zum Verfahren. Sie überlassen die Auswahl dem Anwender.

6. Zur Extraktion der Iso-Oberfläche wende den Marching-Cubes-Algorithmus[14] an. Die vorzeichenbehaftete Distanz $d(p)$ wird berechnet, indem der gewichtete Durchschnitt p_{8er} der Achter-Nachbarschaft des Punktes berechnet wird. Nun gilt: $d(p)$ ist die euklidische Distanz zwischen p und p_{8er}. Wenn der Winkel zwischen dieser Strecke und der Normalen größer als 90° beträgt, erhält die Distanz ein positives Vorzeichen.

7. Verbessere das Ergebnis mit Hole-Filling und Entfernung von Polygongitterfehlern.

Bittar's et al. Flächenrückführung mittels medialer Achsen [BTG95] nach [Men2001]

Bittar et al. schlagen in ihrem Vorgehen eine Zwei-Phasen-Strategie vor. Erst werden die medialen Achsen berechnet. In einer zweiten Phase wird mit diesen die Oberfläche extrahiert.

Bevor die medialen Achsen extrahiert werden können, wird der Raum partitioniert. Dies geschieht wiederum mit Voxeln. Dazu wird die Bounding-Box der Punktwolke unterteilt. Voxel, welche einen Datenpunkt enthalten, werden als „außenliegend" bezeichnet. Somit ist ebenfalls das Innere des Objektes bekannt. Die Chamfre-Distanz[15] wird zur Berechnung einer Enfernungskarte benutzt, mit deren Hilfe die medialen Achsen extrahiert werden.

Im Folgenden wird für jede Kugel der medialen Achse der Wert einer Feldfunktion berechnet. Die Summe aller Feldfunktionen ergibt die Feldfunktion der gesamten Oberflächen. Die Oberfläche besteht aus all jenen Punkten, für welche die Feldfunktion einen konstanten Wert annimmt.

Dieser Algorithmus sei nur aus Gründen der Vollständigkeit erwähnt. Er besitzt praktisch keine Bedeutung in der Flächenrückführung. Seine Ergebnisse weisen meist eine sehr unruhige Oberfläche auf. Diese bedarf häufig intensiver Korrekturen.

[14] Erläuterungen hierzu sind im Anhang zu finden

[15] Erläuterungen im Anhang

Der Power Crust Algorithm von Amenta und Choi [ACK01]

Der Power Crust Algorithmus gehört zu den aktuell am weitesten entwickelten Algorithmen. Er hat eine hier schon dokumentierte lange Entwicklung über viele Stufen hinter sich[16]. In der selben Weise wie die anderen bisher vorgestellten Algorithmen von Amenta wird zu Beginn das Voronoi-Diagramm der Punktwolke berechnet. Wie in 2.1.1 beschrieben, werden im zweiten Schritt die Pole bestimmt. Im dritten Schritt folgt nun die Konstruktion des Power-Diagramms. Dieses ist eine Art gewichtetes Voronoi-Diagram. Seine Wichtung erhält es durch die „Power"-Distanz der Punkte zu den Voronoi-Kugeln. Die Voronoi-Kugeln der Pole werden als Power-Balls bezeichnet. Hierbei gilt zur Berechnung der „Power"-Distanz d_{pow}:

$$d_{pow}(x, B_{c,p}) = d^2(x, c) - p^2$$

$B_{c,p}$: ein Power-Ball

x: ein Punkt

c: Mittelpunkt einer Kugel

p: Radius einer Kugel

Abbildung 19 zeigt ein 2D-Voronoi-Diagramm einer Punktmenge, in welches ein 2D-Power-Ball eingezeichnet ist. Zu erkennen ist, dass ein Power-Ball eine die Oberfläche berührende Kugel ist.

[16] Die gesamte Entwicklung ist unter http://www.cs.ucdavis.edu/~amenta/pubs/pubs.html zu finden

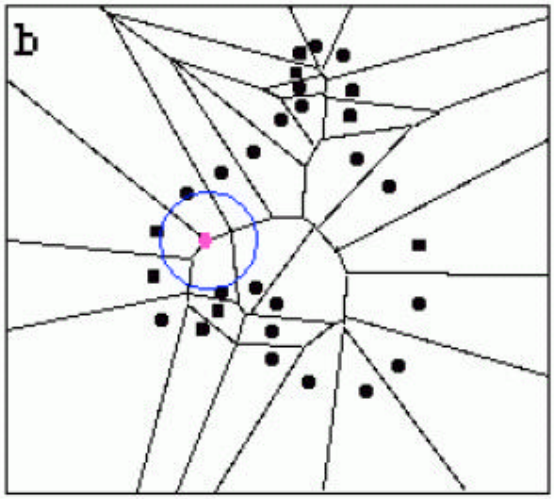


Abbildung 19: Beispiel für ein zweidimensionals Voronoi-Diagramm mit eingezeichnetem Powerkreis. [Kur2003]

Power-Balls geben Aufschluss über die Oberfläche. Die auftretende Grenze zwischen Voronoi-Zellen aus inneren und äußeren Power-Balls beschreibt die Oberfläche. Um die äußeren und inneren Pole zu bestimmen, werden zwei Eigenschaften der Powerballs ausgenutzt.

1. Power-Balls um einen inneren und äußeren Pol schneiden sich nur wenig bis gar nicht

2. Jeder Punkt besitzt genau zwei Pole.

Zu Beginn werden alle Pole außerhalb der Bounding-Box der Punktmenge als „außen“ markiert. Sukzessive werden den beiden Eigenschaften folgend alle Pole markiert.

Als zusätzliches Ergebnis ergeben sich aus der Verknüpfung aller inneren Pole die medialen Achsen des Objektes.

2.3.3 Warping-Algorithmen

Unter Warping wird in diesem Zusammenhang das gezielte Verformen eines Polygongitters hin zu der gesuchten Oberfläche verstanden. Dieser Verformung liegen häufig physikalische Modelle oder auch Transformationen zugrunde. Einen sehr knappen Überblick über solche Beispiele gibt das folgende Kapitel. Als alleinstehende Algorithmen sind sie sehr selten anzutreffen. Meist werden sie zur Verbesserung der Ergebnisse anderer Verfahrensweisen hinzugezogen.

Spatial Free Form Warping – Freie Raumdeformation

Hierunter ist eine Gruppe von Ansätzen zusammenzufassen. Sie haben alle gemeinsam, dass sie den gesamten Raum transformieren, um damit das Objekt gezielt zu verändern. Um dies zu erreichen, werden Skalierungen, Scherungen, Rotationen (entweder als Eulerwinkel, seltener als Quaternion) und Translationen geschätzt oder berechnet. Meist werden bei diesen Methoden grobe Approximationen der aufgenommenen Szene benutzt, und diese an die Punktwolke angepasst. Häufig wird die Punktwolke als Menge mehrerer lokaler Transformationen betrachtet (z.B. Layer, Slices oder Segmentierungsergebnisse). Für die Schätzung der Transformation einer groben Form in die Zielform existieren gute Schätzungen. Der Vorteil dieser Methoden liegt in wenigen benötigten Punkten.

Die Kohonen-Eigenschaftskarte

Die Kohonen-Eigenschaftskarte ist aus dem Bereich der neuronalen Netzen bekannt. Sie ist ein zweidimensionales Array aus sogenannten Neuronen, denen ein Wichtungsvektor zugeordnet wird. Dieser hat zur Zeit der Initialisierung die Länge eins. Die Eingabedaten beeinflussen diese Wichtungsvektoren. Das Neuron, welches am stärksten auf die Eingabedaten reagiert, wird als Zentrum des zu untersuchenden Gebiets gewählt. Im weiteren Prozess werden vom Zentrum ausgehend die Wichtungsvektoren aller Nachbarn beeinflusst. Nach Veränderung der Wichtungsvektoren werden diese wieder normalisiert. Dieser Prozess muss mehrfach wiederholt werden, um gute Ergebnisse zu erzeugen beziehungsweise „zu erlernen“. Auf dieser Basis existiert eine Vielzahl von Algorithmen.

2.3.4 Oberflächenorientierte Wachstumsalgorithmen

Die Grundidee besteht darin, nach und nach einen Punkt zur Oberfläche hinzuzufügen. Entscheidungskriterien geben dabei an, ob eine Verbindung zu anderen Punkten entsteht oder nicht. Beispielsweise wird mit einem Punkt begonnen und seine nächsten Nachbarn hinsichtlich der festgelegten Eigenschaften getestet. Diese Algorithmen besitzen eine Verwandtschaft zu den aus der Bildsegmentierung bekannten Wachstumsalgorithmen. Ein Beispiel für die Verwandtschaft ist in Abbildung 20 illustriert. Es ist ersichtlich, wie einzelne Zwischenstufen der Oberfläche eines regelmäßigen Ringes aussehen. Die Anordnung der Punkte ist dem Ergebnisring rechts unten zu entnehmen.

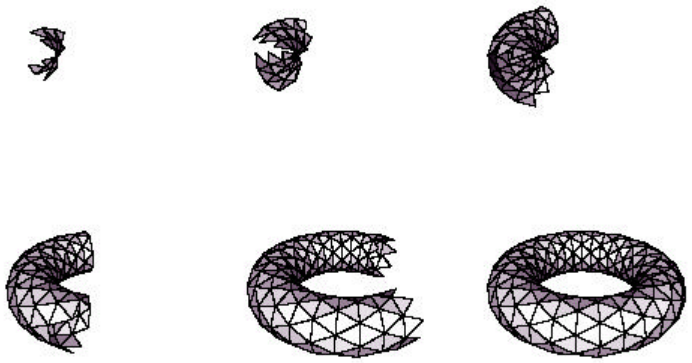


Abbildung 20: Die Verwandschaft zu Wachstumsalgorithmen
am Beispiel von Boissonat [Boi1984]

Boissonat's Oberflächenorientierter Ansatz [Boi1984]

Startpunkte sind die beiden Punkte einer Punktewolke mit dem kleinsten Abstand zueinander. Um durch Anfügen von Kanten Dreiecke zu konstruieren, wird eine Tangentialebene geschätzt. Dies wird mittels Regression über eine freiwählbare Anzahl von Punkten in der Nachbarschaft erreicht. Auf diese Ebene werden alle Nachbarschaftspunkte projiziert. Gewählt wird derjenige Punkt, welcher in der Projektion den größten Winkel zwischen seinen beiden Schenkeln aufweist. Dies ist in Abbildung 21 veranschaulicht. Der Abbruch erfolgt, wenn keine Kante mehr verfügbar ist.

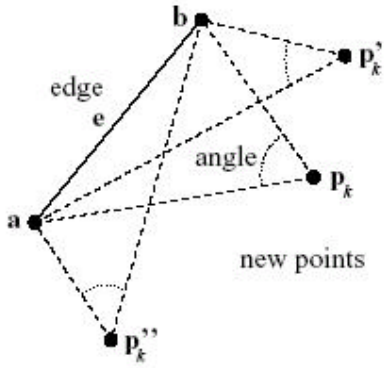


Abbildung 21: Auswahlkriterium für das entstehende Dreieck

Reconstruction with Lower Dimensional Localized Delaunay Triangulation [GKS2000]

Die hier beschriebene Methode besitzt 4 wesentliche Schritte:

1. Schätzung der Normalen aufgrund der Nachbarschaft. Es wird zu diesem Zweck eine *k*-Nachbarschaft gewählt. Die gesuchte Normale $\vec{n_p}$ ergibt sich aus

$$\min\left(\sum_{i=1}^{k}\left(\left(\vec{V_i}-\frac{\sum_{i=1}^{k}\vec{V_i}}{k}\right)\bullet\vec{n_p}\right)^2\right)$$

   $\vec{V_i}$ ist hier als Vektor aus den Koordinaten der Punkte aus der betrachteten Nachbarschaft zu verstehen. Somit ergibt sich ein klassisches Minimierungsproblem der Form $\min(||An_p||_2)$. Um den gesuchten Vektor zu finden, kann die Singulärwertzerlegung eingesetzt werden. Zur Bestimmung der Orientierung wird die in Abschnitt 2.2.2 vorgestellte Methode von Hoppe et al. benutzt.

2. Berechnung möglicher Nachbarn in der Endvermaschung der Delaunay-Triangulation. An dieser Stelle wird in [GKS2000] auf eine bewiesene Methode von Taubin verwiesen. Diese Methode zeigt, dass die Hauptkrümmung und Hauptrichtungen einer Umgebung gleich den Eigenwerten und Eigenvektoren einer 3x3-Matrix M_p sind.

$$M_p = \sum_{i=1}^{k} \left(\frac{\alpha_{i+1} - \alpha_{i-1}}{4\pi} \right) k_p^{v_i} \overrightarrow{vv}^T$$

Hierbei gilt $\vec{v} = \overrightarrow{v_1^p} \cos\theta + \overrightarrow{v_2^p} \sin\theta$ mit den beiden Hauptrichtungen $\overrightarrow{v_1^p}$ und $\overrightarrow{v_2^p}$ in einem Punkt p. θ ist dabei der eingeschlossene Winkel zwischen $\vec{v}$ und $\overrightarrow{v_1^p}$. α_i ist der Winkel, welchen $\vec{v_i}$ mit der Tangentialebene einschließt. $k_p^{v_i}$ ist die Standard-Krümmung.
Aus der berechneten Hauptkrümmung wird in der 2∂-Nachbarschaft für jeden Punkt die Konstante $m_p = \frac{2k_{\max}}{k_{\min}}$ berechnet. ∂ ist die Sampelrate. m_p ist das im nächsten Schritt benutzte Merkmal zur Auswahl der benötigten Nachbarn.

3. Auswahl des Nachbarn aus den in Schritt zwei berechneten Kandidaten. In einem ersten Teilschritt wird mittels der ∞-Norm eine erste Einschränkung der Kanidaten getroffen. Im zweiten Teilschritt werden mittels euklidischer Metrik Punkte außerhalb einer Kugel mit Zentrum in Punkt p ausgeschlossen. Dies ist nun in beiden Fällen sehr einfach möglich. Wenn s die Distanz (gemessen mit der jeweiligen Metrik) zwischen p und seinem nächsten Nachbarn ist, so sind die eingeschlossenen Punkte nicht weiter entfernt als $m_p s$.

4. Zur Triangulation eines Punktes und seiner Nachbarschaft werden diese auf die Tangentialebene des Punktes projiziert. Im lokalen Koordinatensystem der Tangentialebene wird die Umgebung mittels Delaunaytriangulierung vermascht.

2.3.5 Clustering-Methoden

Wenn eine Szene aus mehreren Objekten besteht, können die bisher beschriebenen Algorithmen keine zufriedenstellenden Ergebnisse liefern, da sie von einer geschlossenen Oberfläche ausgehen. Daher ist eine Segmentierung oder ein Clustering der Szene notwendig. Beispielsweise kann ein solches Problem in Räumen mit vielen Gegenständen auftreten. Trotz auftretender Verdeckungen der Gegenstände müssen diese voneinander getrennt werden.

Ein Ansatz von Fua und Sander[FS1992] nach [Men2001]

Fua und Sander geben für diese Aufgabe ein einfaches Vier-Schritte Schema an:

1. Für jeden Punkt der Punktwolke wird eine Quadrik durch eine wählbare Umgebung bestimmt.

2. Die Quadriken werden in einem regulären Gitter angeordnet.

3. Je ähnlicher die Quadriken zueinander sind, desto größer ist ihr Zusammenhang. Zwei sehr ähnliche Cluster werden verschmolzen. Dieser Vorgang wird bis zum Erreichen eines Schwellwertes wiederholt. Der Schwellwert ist dabei die gewünschte Anzahl von Clustern, oder eine aus einem Ähnlichkeitsmaß bestimmte Ähnlichkeit.

4. Auf jedes der gefunden Cluster kann ein beliebiger Rekonstruktionsalgorithmus angewendet werden.

2.4 Punktreduktion

3D-Aufnahmegeräte erzeugen immer höhere Punktdichten, und somit auch eine immer gewaltigere Anzahl von Punkten. Im vorliegenden Fall sind dies bis zu 400.000 Punkte in einem Datensatz. Aus diesem Grund muss der Punktreduktion eine immer bedeutsamere Rolle zugeschrieben werden.

Es soll in diesem Abschnitt nur auf das weitverbreitete „Edge-Collapse"-Verfahren und allgemein auf adaptive Reduktion eingegangen werden, da diese die einzigen in Kombination mit Flächenrückführungsalgorithmen auftretenden Reduktionsalgorithmen sind. Starre Methoden, wie das Vertex Clustering[17], besitzen nur noch in der Echtzeitverarbeitung Bedeutung. Die folgenden Beschreibungen beziehen sich auf Algorithmen, welche bezüglich der Entwicklung des in dieser Arbeit vorgestellten Verfahrens von Bedeutung sind.

2.4.1 Adaptive Reduktion

Unter adaptiver Reduktion wird in diesem Kontext die Reduktion zusammengehöriger Punkte in Abhängigkeit des benötigten Detailgrades der Darstellung verstanden. Beispielsweise kann darunter verstanden werden, dass zum Modellieren starker Krümmungen mehr Punkte benötigt werden, um eine festgelegte Genauigkeit zu erreichen, als zum Modellieren einer Ebene.

[17] In diesem Verfahren werden einzelne frei bestimmbare Zellen zu einem Vertex zusammengefasst

2.4.2 Edge-Collapse-Verfahren [Hop1994] [Hop1996]

Bei diesem Ansatz wird anhand einer Pseudo-Energiefunktion[18] entschieden, ob zwei Vertices einer Kante zusammengefaltet werden. Dieser Vorgang ist in Abbildung 22 beschrieben. Die hier vorgestellte Variante zur Anwendung des Edge-Collapse-Verfahrens entspricht den optimierten Erweiterungen für „progressive meshes“ [Hop1996].

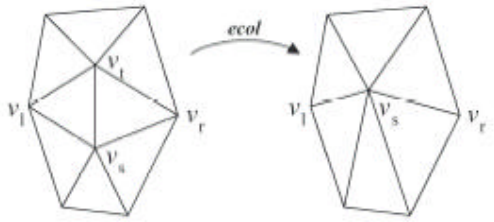


Abbildung 22: Edge Collapse – Prinzip [Hop1996]

Die Pseudoenergiefunktion ist dabei definiert als:

$$E(M) = E_{dist}(M) + E_{spring}(M) + E_{scalar}(M) + E_{disc}(M)$$

M sei das entstandene Polygongitter und E die Energie. Die Energie $E_{dist}(M)$ entspricht hierbei der Summe der quadratischen euklidischen Abstände aller in M verwendeten Punkte zum ursprünglichen Polygongitter. Sei $d(v_i, v_j)$ der quadratische euklidische Abstand zweier verschmolzener Punkte v_i und v_j, so ist $E_{spring}(M)$ die Summe aller $d(v_i, v_j)$. $E_{scalar}(M)$ soll kontinuierliche Bereiche erhalten. Dazu wird eine Parametrisierung benötigt. Diese kann der Abstand zwischen den Wertetripeln der Normalen in den Punkten sein, und bei texturierten Objekten zusätzlich der Abstand zwischen den Tripeln des RGB-Modells. Die Summe der Abstände jedes Punktes wird nun mit einer wählbaren Konstanten c_{scalar} multipliziert, so dass $E_{scalar} = c_{scalar}^2 \sum_{i,j=0}^{max} d(b_i, b_j)$ ist. b_x seien die Parametrisierungen der Normalen und max die bisherigen collapse-Paare. Zuletzt sei E_{disc} die Energiefunktion zur Erhaltung von Kanten.

[18] In der Veröffentlichung von Hoppe wird ausschließlich von einer Energiefunktion gesprochen. Es soll jedoch ein Unterschied zu Energie im physikalischen oder elektrotechnischen Sinn gemacht werden.

Hierzu wird ein Regelwerk definiert. Zur Anwendung dieses Regelwerkes müssen alle Kanten im Punkt als „scharf“ oder „nicht scharf“ markiert werden. Dies geschieht anhand des Winkels [WW2003] oder des Abstandes zwischen den Normalen[19].

Der Vorteil dieses Verfahren liegt in seiner Echtzeitfähigkeit. Nachteilig ist die Notwendigkeit des Vorhandenseins eines polygonalen Gitters. Zudem wird an das Gitter die Forderung gestellt, exakt zu sein, da die Pseudo-Energiefunktion den Winkel zwischen den Normalen in den Punkten berücksichtigt.

[19] Im Original wird nicht weiter auf die Bestimmung eingegangen. Jedoch werden die gleichen Beispiele zur Parametrisierung erwähnt wie zu Definition von E_{scalar}.

2.5 Diskussion und Schlussfolgerung

Die Vielzahl existenter Algorithmen lässt den Schluss zu, dass bisher keine allgemein zufriedenstellende Lösung für das Problem der Flächenrückführung gefunden wurde. Alle bekannten Verfahrensweisen unterliegen Einschränkungen. Beispiele für solche Einschränkungen können sein:

1. Löcher sind nicht modellierbar.

2. Mehrere Objekte in einer Szene sind nicht handhabbar.

3. Nicht rekonstruierbare Unterbrechungen im Modell.

4. Die Schätzung von Normalen ist häufig schlecht, oder nicht möglich. Meist geht sie von der Korrektheit der Daten aus. Gestörte Aufnahmen müssen einer Vorverarbeitung unterzogen werden, damit die Normalenschätzung funktioniert.

Zudem tritt bei der Klasse der Raumunterteilungsalgorithmen die Rechenzeit für die Delaunay-Tetraeder-Unterteilung als begrenzender Faktor in den Vordergrund. Bei großen Datensätzen wird diese zum Problem.

Des weiteren ist im Zuge der Literaturarbeit kein Algorithmus gefunden worden, welcher auf Störungen und Fehlmessungen unempfindlich, oder korrigierend reagiert. In vielen Fällen sind keine Versuche an realen Datensätzen gemacht worden. Meist handelt es sich in den Beispielen um ideale Formen oder Datensätze, welche mit sehr viel Aufwand korrigiert worden sind. Diese Modelle stammen häufig aus der Sammlung der Stanford University[20] und besitzen die Eigenschaft, dass Normalen in den Punkten geschätzt werden können. Jedoch ist dieses bei vielen realen Punktwolken nicht möglich. Diese Normalen werden aber bei gängigen Teil-Algorithmen, wie dem Marching Cubes Algorithmus, benötigt.

Bei der Betrachtung der Aufnahme-Systeme ist festzustellen, dass die meisten Hersteller Charakteristiken ihrer Systeme ausnutzen, um eine Triangulierung zu erzeugen. So auch bei dem Zusammenführen mehrerer Aufnahmen. Dreidimensionale elastische Verfahren zum

[20] http://www-graphics.stanford.edu/data/3Dscanrep/

Zusammenführen von Punktwolken besitzen bislang noch keine Relevanz. Vermutlich aus dem Grund, dass nicht entschieden werden kann, an welcher Stelle eine Aufnahme genauer ist als eine andere.

Aus diesen Grund ist ein Algorithmus notwendig, der sich möglichst vielen dieser Herausforderungen stellt.

3. Überblick über den Algorithmus

Das dritte Kapitel soll einen Überblick über die Funktionsweise des in dieser Arbeit entwickelten Algorithmus geben. Das Kapitel gibt sowohl die mathematischen Grundlagen, als auch die anschauliche Vorgehensweise wieder. Die Diskussion der einzelnen Methoden und ihrer Probleme findet in den darauf folgenden Kapiteln statt. 3.2 führt die zur Ausführung des Algorithmus benötigten Parameter und ihre Abhängigkeiten ein. Abschnitt 3.3 beschreibt in knapper Weise eine mögliche Korrektur der Daten, um Ansichten ineinander überzuführen. In 3.4 wird die Findung der Quadriken beschrieben und der dadurch gewonnene Verzicht auf die schlechte bis unmögliche Schätzung der Normalen. Abschnitt 3.5 widmet sich der Punktreduktion.

Im Folgenden soll untenstehende Notation Gültigkeit besitzen:

Punktwolke	P
Ein Punkt der Punktwolke	p_i
Quadrik	Q
v^t	Transponierter Vektor
${}^0n_{p_i}$	Nächster Nachbar von p_i
${}^jn_{p_i}$	Ein j-ter nächster Nachbar des nächsten Nachbarn von p_i, welcher noch keiner Quadrik zugeordnet ist.
$R_{Achse}(\varphi)$	Drehung um eine Koordinatenachse x, y oder z um Winkel φ im mathematischen Umlaufsinn
$S_{Achse}(\lambda)$	Skalierung um Faktor λ in x-,y-, oder z- Richtung
$T(v)$	Translation um einen Vektor $v(a,b,c)^t$

3.1 Die drei Schritte im Überblick

Phase 1:

- Überführung zweier Aufnahmen mittels Bestimmung der nächsten Nachbarn in der Vergleichsaufnahme
- Sortieren nach Abständen der gefundenen Paare
- Zusammenführen aller Punkte, deren Abstand kleiner $2 \bullet (AA + MMF)$ ist.

  AA: minimaler messbarer Abstand

  MMF: maximaler Messfehler

Phase 2:

- Der größte Abstand einer betrachteten Teilmenge der Punktwolke wird als erste Achse definiert
- Es wird eine Wachstumsstrategie über einen jeweils nächsten Nachbar, welcher noch nicht in die Quadrik einbezogen wurde, angewendet
- Die Schätzung der beiden anderen Achsen wird mit der Methode der Pseudo-Inversen durchgeführt
- Das Ergebnis kann Ebene, Zylinder, Kugel oder Ellipsoid sein.
- In der Praxis sind meist Ellipsoide anzutreffen

Phase 3:

- Für Ellipsoide und Zylinder wird eine adaptive Reduktion anhand der Gaußschen Krümmung angewendet.
- Ebenen werden durch Entfernen aller Punkte ohne infiniten Nachbar im Voronoi-Diagramm reduziert.
- Triangulierung der Ellipsoide in Polarkoordinaten als 2D Delauny-Triangulierung

Abbildung 23: schematischer Ablauf des Algorithmus

3.2 Einführung in die benötigten Parameter

Der gesamte Algorithmus benötigt 4 Parameter zum Start:

1. Den minimal messbaren Punktabstand ΔmPa ,
2. den Entscheidungsparameter ξ zwischen Ebene und Ellipsoid,
3. das maximale Ergebnis der Quadrikengleichung ζ und
4. den Adaptionsparameter α zur Punktreduktion.

Für die Parameter ξ und ζ bestehen hinreichend gute Näherungen, welche mit Parameter eins berechnet werden können. Daher kann die Anzahl der benötigten Parameter auf zwei reduziert werden.

Definition 1: ΔmPa *ist der kleinste messbare Abstand inklusive des doppelten maximalen Messfehlers des Systems.*

Die folgenden Überlegungen basieren auf der Voraussetzung:

$$P = \left\{ p_i \,\middle|\, p_i \in P; -1 \le element(p_i) \le 1 \right\}$$

$element(p_i)$: Die Einzelelemente des xyz-Wertetripels eines Punktes p_i

Es sind folgende Näherungen möglich:

$$\xi = \Delta mPa \cdot 2\sqrt{\Delta mPa}$$

Das Zweifache des minimalen Abstandes ΔmPa resultiert aus dem Nyquisttheorem, welches besagt, dass der max. Fehler einer Quantisierung doppelt so gross ist wie eine Quantisierungsstufe. Somit muss zur Entscheidung das Doppelte des im Systemauftretenden Messfehlers betrachtet werden. $\sqrt{\Delta mPa}$ spiegelt den Fehler, welcher in

der Achsenlänge gemacht wird wieder, da die Achsenlänge quadratisch in allen Berechnungen vorliegt.

$$\zeta = a \cdot \sqrt{\Delta mPa}$$

a ist hierbei der Skalierungsfaktor der gesamten Daten in einen Bereich zwischen –1 und 1, da alle Wertetripel der Punkte in diesen Bereich transformiert werden. Dies kann auch jede andere Transformation sein, welche die Daten einfacher behandelbar macht. $\sqrt{\Delta mPa}$ steht wiederum für den Fehler der quadratisch vorliegenden Achse.

Der Parameter α besitzt ausschließlich Einfluss auf die Punktreduktion. α sei wie folgt definiert (vergleiche Abbildung 24):

Definition 2: *Werden zur Näherung einer Kugelfläche regelmäßig verteilte Punkte verwendet, so ist α der Winkel zwischen zwei dieser Punkte bezogen auf das Zentrum der Kugel.*

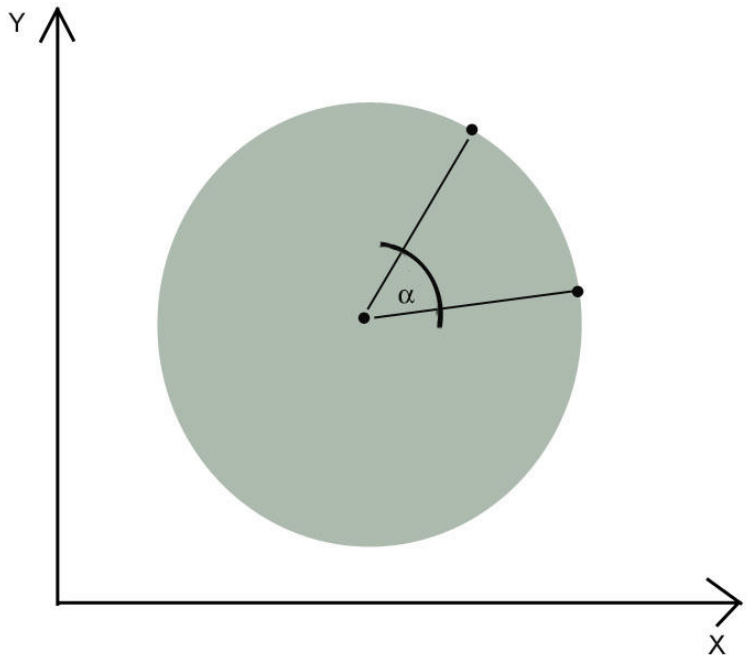


Abbildung 24: schematische 2D-Darstellung des Winkels α

Somit ist für die Flächenrückführung nur ein Parameter notwendig. Dieser ist von dem Punktwolkenerzeugungsverfahren[21] abhängig. Darin besteht ein deutlicher Unterschied zu den meisten in Kapitel zwei beschriebenen Rückführungsalgorithmen. Die bislang vorgestellten Algorithmen besitzen allesamt mindestens einen benutzerdefinierten Parameter. Die vollautomatischen Ansätze besitzen Schwächen in der Modellierung von Löchern und flachen Bereichen, welche aus Körpern herausragen.

[21] Anmerkung: Punktwolken können nicht nur mit Aufnahmesystemen wie Laser- oder Streifenprojektionsscanner erzeugt werden, sondern auch durch Stereoverfahren aus zwei Ansichten einer Szene in Fotos. Auch die Erzeugung aus Filmmaterial ist möglich (Structure from Motion).

3.3 Zusammenführen der Ansichten

Wie schon in Kapitel zwei erwähnt, wurden die vier Ansichten des Versuchmusters vermutlich mit dem ICP-Algorithmus ineinander übergeführt. Diese Annahme begründet sich in den typischen Artefakten der ICP-Methode. Das charakteristischste Merkmal dieser Methode ist die Reaktion auf Fehlmessungen. Diese werden wieder ausgeglichen. Dies hat zur Folge, dass zwei Punktwolken oftmals leicht gekreuzt werden oder ungewollte Abstände zwischen den Punktwolken entstehen. Ein Beispiel dafür ist in Abbildung 25 zu erkennen.

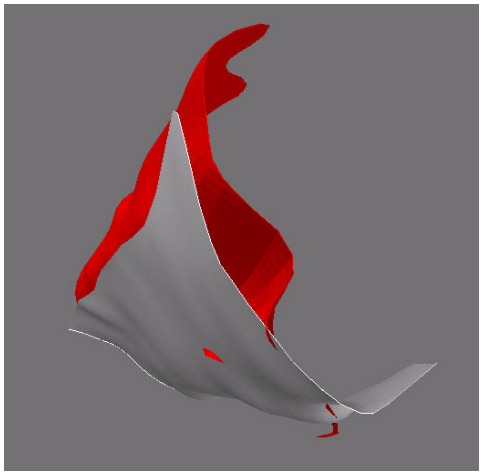


Abbildung 25: Ausschnitt aus einfach vermaschten Einzelaufnahmen, die ICP- Charakteristiken aufweisen. Hierbei sind große Abweichungen zwischen den Messungen erkennbar.

Im Prozess der Überführung in eine Punktwolke wird zu Beginn der nächste Nachbar in der korrespondierenden Aufnahme bestimmt. Alle nächsten Nachbarn, welche einen geringeren Abstand besitzen als ΔmPa , werden zusammengeführt. Die Nachbarsuche wurde durch vorhergehende Clusterbildung (Raumunterteilung) beschleunigt. Hierzu wurde der Raum in regelmässige Voxel-Cluster unterteilt, so dass nicht alle Punkte durchsucht werden müssen. Um die Toleranz gegenüber Messfehlern zu verbessern, kann auch nach einem extremen Sprung in den sortierten[22] Abständen gesucht werden[23]. Dies besitzt den Vorteil, auf Kalibrationsfehler weniger stark zu reagieren. Das bedeutet, dass

[22] Hier verwendet: Quicksort.

[23] Eine effiziente Ermittlung ist in der Implementierung mittels des Quotienten zwischen Vorgänger und Nachfolger möglich

statisch auftretende Fehler, keinen Einfluss auf die Zuordnung besitzen. Als Beispiel hierfür ist ein konstanter Abstand zwischen zwei Aufnahmen des gleichen Objektes aus unterschiedlichen Richtungen zu nennen. Dies kann aufgrund von Kalibrationsfehlern des Systems entstehen oder auch durch den ICP-Algorithmus selbst. Aus dieser Überlegung entsteht die vollständige Beschreibung des Kriteriums zur Zusammenführung. In Abbildung 26 ist ein reales Beispiel zu sehen. Erkennbar werden die Messfehler im oberen rechten Bereich des Ausschnitts. Zudem wird an der Dicke des Blattes und des Stammes die Ausdünnung der Punktwolke ersichtlich.

Abbildung 26: links der Zustand vor der Zusammenführung,
rechts der Zustand nach der Zusammenführung.

3.4 Dekomposition der Szene

Zur Näherung der Oberflächen werden Quadriken in der homgenen Form verwendet. Quadriken sind Flächen zweiter Ordnung:

$$f(x, y, z) = ax^2 + by^2 + cz^2 + 2dxy + 2exz + 2fyz + 2gx + 2hy + 2jz + k$$

$$F = \{(xyz)^t \in R^3 \mid f(x, y, z) \leq 0\}$$

Die homogene Form einer Quadrik Q ist somit beschrieben als[24]:

$$Q = \begin{pmatrix} a & d & e & g \\ d & b & f & h \\ e & f & c & j \\ g & h & j & k \end{pmatrix}, \quad F(Q) = \{v \in P^3 \mid v^t Q v \leq 0\}$$

3.4.1 Vorverarbeitung und Bestimmung der Startpunkte

Als Ausgangspunkt zur Bestimmung der Quadriken wird Kenntnis über die Nachbarschaftsbeziehungen der Punkte benötigt. Diese muss im hier vorliegenden Fall berechnet werden. Diese Nachbarschaft besteht aus den nächsten Nachbarn. Bei den bekannten Aufnahmesystemen sind die Nachbarschaftsbeziehungen gegeben. Diese müsste jedoch im Datenformat mit abgespeichert werden und lag in diesem Fall nicht vor. Aus diesem Grund wird eine 26er – Nachbarschaft benutzt. Der maximal zulässige Abstand zwischen zwei benachbarten Punkten ist ΔmPa . Die 26er – Nachbarschaft ist dabei wie folgt bestimmt.

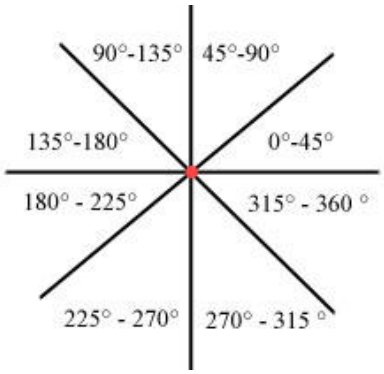


Abbildung 27. : Darstellung der Zelleinteilung zur Nachbarschaftsbestimmung. Die obere Grenze ist als Grenzwert zu interpretieren.

[24] Beispiele sind im Anhang zu finden

Zu jedem Punkt wird der nächste Nachbar in einem ausgewählten Gebiet bestimmt. Wird eine Gerade durch diesen nächsten Nachbarn und den Ursprung gelegt, so sind die Winkel zwischen der Gerade und den Koordinatenachsen berechenbar. Die Winkel sind zur x-, y- und z –Achse sind wie in Abbildung 27 eingeteilt. Der rote Punkt in der Mitte der Abbildung symbolisiert den aktuellen Punkt p_i. Somit ergibt sich eine 26er-Nachbarschaft pro Datenpunkt. In Abbildung 28 ist ein Beispiel für eine 26er-Nachbarschaft gegeben. Diese spiegelt nicht die tatsächliche Raumaufteilung wieder, sondern die dient nur der Verdeutlichung der 26er-Nachbarschaft. Die berechneten Winkel werden einem 3x3x3 Arraywürfel in der Datenstruktur zugeordnet. Das Arrayelement im Zentrum des Arraywürfels ist der betrachtete Punkt.

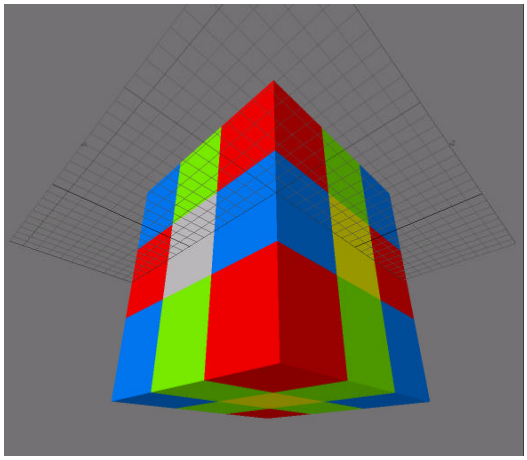


Abbildung 28: Darstellung von Raumelementen in einer 26er-Nachbarschaft.

Auch hier ist die 26er-Nachbarschaftsbestimmung erkennbar. Diese Vorgehensweise ist im vorliegenden Fall zweckmäßig, da auf die Winkel zu den Achsen des Koordinatensystems in der Beispielanwendung direkt über die gewählte Datenstruktur zugegriffen werden kann. Andere Vorgehensweisen zur Bestimmung der Nachbarschaften sind möglich.

Nach dem Aufbau eines solchen Nachbarschaftsnetzes wird der Punkt mit dem größten Abstand zum Ursprung des karthesischen Koordinatensystems als Startpunkt p_S zur Suche nach Quadriken gewählt.

3.4.2 Ablauf der Quadrikenbestimmung

Als zweiter Startwert zur initialen Berechnung der Quadrik wird der nächste Nachbar ${}^{0}n_{p_i}$ aus der Nachbarschaft von p_S gewählt. Zusammen mit ${}^{1}n_{p_i}$ wird eine Ebene aufgespannt. Schrittweise wird nun ein nächster Nachbar ${}^{j}n_{p_i}$ nach dem anderen in die Berechnung der Quadrik mit einbezogen. Die Besonderheit besteht darin, daß für die Schätzung des Parameters a der Quadrik folgender Wert angenommen wird:

$$d_{\max} = \max\left\{\overline{p_i\,{}^{j}n_{p_i}}\,; Anz \geq j, i > 0\right\}$$

$$a = \frac{1}{d_{\max}^2}$$

Anz: die in einem Durchgang betrachtete Teilmenge einer Punktwolke

$\overline{p_i\,{}^{j}n_{p_i}}$: der maximale Abstand zwischen einem Punkt und all seinen zu diesem Zeitpunkt betrachten Nachbarn.

Dabei kann $d_{\max}$ als Skalierung einer Ellipsoidachse interpretiert werden. Anhand dieser sind die Rotationen der Quadrik um die x, y und z-Achse bestimmbar. Das Wertetripel des Mittelpunktes $p_M(x, y, z)$ entspricht der Translation der Quadrik in x-y- und z-Richtung. Somit sind bis auf b und c alle Parameter einer die ausgewählten Punkte beschreibenden Quadrik bekannt. Diese werden nun mit Hilfe der Methode der Pseudoinversen geschätzt. Die Methode der Pseudoinversen ist ein Verfahren zur Lösung überbestimmter Gleichungssysteme der Form $Ax - m = 0$.

Mit bekannter Transformation der Quadrik kann Q durch Rückwärtsanwendung der Transformationen auf die folgende Form überführt werden:

$$Q = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Bei der Anwendung der Transformationen ist darauf zu achten, dass die durch d_{max} bestimmte Strecke entlang der x-Achse liegt. Eine solche Lage einer Quadrik wird auch als Nulllage bezeichnet. Da a bekannt ist, besteht der Lösungsvektor x der Pseudoinversen-Methode aus den Parametern b und c.

$$\vec{x} = \begin{pmatrix} b \\ c \end{pmatrix}$$

$$A^+ = (A^t A)^{-1} A^t$$

A^+ : die pseudoinverse Matrix (Moore-Penrose-Inverse)

A besteht aus den y- und z-Werten der betrachteten Punkte $Anz(p_i)$ aus P.

$$A = \begin{pmatrix} p_{S_y} & p_{S_z} \\ {}^0n_{p_{S_y}} & {}^0n_{p_{S_z}} \\ : & : \\ : & : \\ {}^in_{p_{S_y}} & {}^in_{p_{S_z}} \end{pmatrix}, 0 \le i \le Anz(p_i)$$

$A^t A$ ist in dieser Form mit nur einem Durchlauf der betrachteten Punkte zu erzeugen. Zudem kann auf eine aufwändige Singulärwertzerlegung verzichtet werden, da die inverse Matrix $(A^t A)^{-1}$ direkt berechnet werden kann[25].

Es ergibt sich hieraus durch Umstellen des Ausgangsterms:

$$Ax - m = 0$$

$$x = A^+ m$$

[25] $A^t A$ ist hier eine 2x2 Matrix. Die Summen $\sum_{i=1}^{n} y^4$, $\sum_{i=1}^{n} z^4$ und $\sum_{i=1}^{n} y^2 z^2$ werden beim Durchlaufen der Punkte als Zwischenergebnis berechnet. Deshalb ist sowohl die Determinante, als auch die inverse Matrix direkt bestimmbar.

$$\begin{pmatrix} b \\ c \end{pmatrix} = (A^t A)^{-1} A^t m$$

Zur Bestimmung des Ergebnisses ist weiterhin die Berechnung von $A^t m$ notwendig. Da A^t bekannt ist, muss m bestimmt werden. Der Parameter m setzt sich aus dem Quadriken-Parameter $k = -1$, der Skalierung $S_x = d_{\max}$ und dem x-Wert der betrachteten Punkte aus P zusammen. Durch das Umformen der Quadrikengleichung ergibt sich:

$$m = \begin{pmatrix} 1 \\ 1 \\ : \\ : \\ 1 \end{pmatrix} - S_x^4 \begin{pmatrix} {p_{S_x}}^2 \\ {}^0n_{p_{S_x}}{}^2 \\ : \\ : \\ {}^in_{p_{S_x}}{}^2 \end{pmatrix}, 0 \le i \le Anz(p_i)$$

Aus diesen Überlegungen folgt für die Berechnung des gesamten Lösungsvektors $\vec{x}$:

$$\begin{pmatrix} b \\ c \end{pmatrix} = (A^t A)^{-1} A^t \left(\begin{pmatrix} 1 \\ 1 \\ : \\ : \\ 1 \end{pmatrix} - S_x^4 \begin{pmatrix} {p_{S_x}}^2 \\ {}^0n_{p_{S_x}}{}^2 \\ : \\ : \\ {}^in_{p_{S_x}}{}^2 \end{pmatrix} \right), 0 \le i \le Anz(p_i),$$

Der Abbruch der Quadrikenberechnung geschieht auf zwei verschiedene Weisen. Zum einen wird bei der Auswahl der Punkte ζ berücksichtigt. Ist das Ergebnis der Quadrikengleichung eines Punktes größer als ζ, so wird dieser nicht in der Quadrik berücksichtigt.

Des weitern wird ein einfaches Fehlermaß verwendet, um festzustellen, ob die gefundene Quadrik eine gute Näherung darstellt.

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}|A_{ij}x_j - b_i - 1|}{Anz(Punkte)}$$

Die euklidische Norm zur Begutachtung des Fehlers musste ausgeschlossen werden, da sie auch auf gute Näherungen sehr stark reagierte. Die Ursache liegt in den quadratischen Anteilen der Quadrik. Beispielsweise sind annähernd ideale Ebenen nicht immer eindeutig als gültige Quadrik identifizierbar gewesen.

Dies lässt sich durch folgende Überlegung verdeutlichen. Die euklidische Norm als Fehlerbewertung bezieht in ihr Ergebnis nicht die Anzahl der Punkte ein. Somit ist ihr Ergebnis ein aufsummierter Fehler, der sehr groß werden kann. Eine große Anzahl gefundener Punkte mit kleinen Abweichungen würde somit einen großen Fehler erzeugen [Sta2004]. Die Ähnlichkeit der obenstehenden Fehlerfunktion zur euklidischen Norm ist jedoch erkennbar.

3.5 Reduzieren und Vermaschen der Punkte

Zur Reduktion der Punktwolke und der Vermaschung der reduzierten Punktwolke wird zwischen Ebenen und 3D-Quadriken unterschieden. Zur Vermaschung der Punkte wird in beiden Fällen die Delaunay-Triangulierung eingesetzt.

3.5.1 Ebenen

Zur Reduzierung wird die gefundene Fläche in die x-z-Ebene transformiert[26]. Auf diese wird die Delaunay-Trinagulierung angewendet. Alle Punkte, die im Voronoi-Diagramm keinen infiniten Nachbarn besitzen, werden gelöscht. Als Ergebnis bleibt der Rand der Ebene. Der infinite Nachbar wird in Algorithmen zur Delaunay-Triangulierung verwendet, um Punke, welche auf der konvexen Hülle liegen, zu triangulieren.

[26] In der Praxis ist dies in vielen Fällen ein Ändern der Triangulierungsebene in einem Templateübergabeparameter.

3.5.2 3D-Quadriken

Zu den approximierten 3D-Quadriken sollen hier Ellipsoide, Kugeln und Zylinder zählen. An dieser Stelle werden die Parameter ξ und α benötigt. Sind alle Achsen größer als ξ, so wird der folgende Algorithmus angewendet.

1. Bringe die Quadrik in Nulllage.
2. Bestimme die Pole und wähle den nächsten Nachbarn aus der Punktwolke als Repräsentant für die reduzierte Darstellung.
3. Bilde die Verhältnisse der Gaussschen Krümmung an den Polen der b- und c-Achse des Ellipsoids, zu dem Pol der a-Achse des Ellipsoids.
4. Multipliziere diese Verhältnisse mit α, um die nächste Stützstelle zu bestimmen.
5. Bestimme den nächsten Nachbarn aus der Punktwolke als Repräsentanten.
6. Berechne das Verhältnis der Krümmung an der neuen Stützstelle zur Krümmung an dem Pol der a-Achse des Ellipsoids.
7. Wiederhole Schritt 4 bis 6 solange, bis der Pol der a-Achse des Ellipsoids erreicht ist.

In Schritt 1 wird die Quadrik in Nulllage gebracht. Hiefür wird gefordert, dass die durch den Parameter a[27] charakterisierte Ellipsoidachse entlang der x-Achse liegt. Ihr Mittelpunkt liegt im Ursprung. Die durch die Parameter b und c beschriebenen Ellipsoidachsen besitzen eine Lage entlang der y- und z-Achse. Somit entsprechen die Achsen des Ellipsoids denen eines kartesischen Koordinatensystems. Unter Anwendung der bereits bestimmten Translationsstrecke und Rotationswinkel wird die Quadrik in Nulllage transformiert.

[27] a, b, c sind wiederum als Achsen eines Ellipsoids zu interpretieren

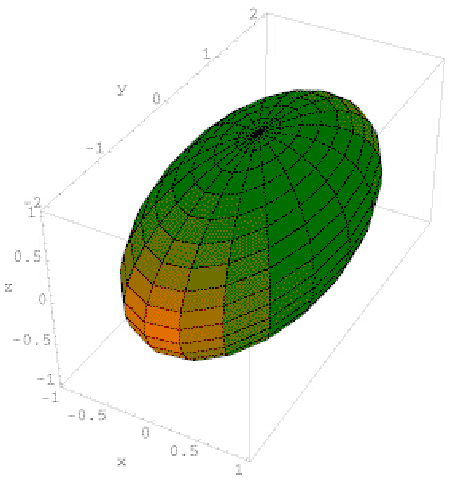
Nach der in Schritt 2 vollzogenen Nachbarbestimmung wird in Schritt 3 die Gausssche Krümmung an den Polen der zu reduzierenden Quadrik bestimmt. Die Gausssche Krümmung für ellipsoide Körper ist wie folgt beschrieben.

$$K(u,v) = \frac{a^2 b^2 c^2}{\left[c^2 \cos(v)^2 \left(b^2 \cos(u)^2 + a^2 \sin(u)^2\right) + a^2 b^2 \sin(v)^2\right]^2}$$

u: überstrichener Winkel von der y-Achse ausgehend zur x-Achse hin

v: überstrichener Winkel von der x-Achse ausgehend zur z-Achse hin

An dieser Stelle muss der Fall beachtet werden, dass die Krümmung an den beiden infiniten Polen von Zylindern nicht definiert ist. Jedoch wird im Folgenden die Krümmung 1 angenommen, da sie keinen Einfluss auf das entstehende Ergebnis bei einem Zylinder besitzt. Die Gausssche Krümmung hat in diesem Fall den Vorteil, an sehr stark gekrümmten Oberflächenteilen größere Werte zu liefern. Dies wird in den folgenden Abbildungen 29 und 30 verdeutlicht.



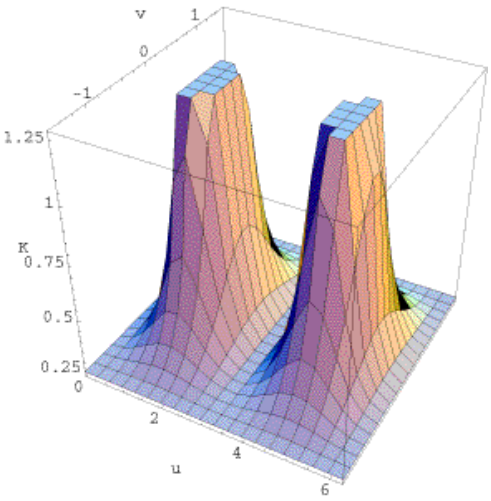


Abbildung 29: (links) Verhalten der Gaussschen Krümmung. Rot entspricht hierbei starker Krümmung, grün geringerer Krümmung
http://www.math.hmc.edu/faculty/gu/curves_and_surfaces/surfaces/ellipsoid.html

Abbildung 30: (rechts) Die k-Achse ist die Krümmung. u- und v-Achse entsprechen den beschriebenen Winkeln in der Gleichung für die Gausssche Krümmung
http://www.math.hmc.edu/faculty/gu/curves_and_surfaces/surfaces/ellipsoid.html

Aus den berechneten Krümmungen der Pole werden Verhältnisse gebildet:

$$kr_{ab} = \frac{K(0,0)}{K(\pi/2,0)} \;, kr_{ac} = \frac{K(0,0)}{K(0,\pi/2)}$$

kr: Verhältnis der Krümmungen an den Polen

kr_{ab} und kr_{ac} sind Multiplikatoren für α,um die in Schritt 4 beschriebene Stützstelle zu bestimmen.. Dies bewirkt, dass an Stellen starker Krümmung sich die Dichte der Stützstellen erhöht und an flachen Oberflächenteilen weniger Stützstellen verwendet werden. In Schritt 5 und 6 werden kr_{ab} und kr_{ac} für die Stützstelle neu berechnet. Ausgehend von den Polen der *b*- und *c*- Achse des Ellipsoids werden die Stützstellen in Richtung der Pole der *a*-Achse gesucht. Alle Pole bilden jeweils eine eigene Stützstelle. Unter visuellen Aspekten ist $\alpha = 5°$ als ein guter Wert zu betrachten. Bei Verwendung eines einfachen Shadings erscheint dem Auge eine mit $\alpha = 5°$ reduzierte Kugel als glatt [WS1999]. Bei größeren Werten für α besteht die Gefahr des Untersamplings oder es müssen nach der Vermaschung Objektrepräsentationen wie NURBS oder Subdivision Surfaces genutzt werden. Bei diesem Vorgang muss beachtet werden, dass der Punkt der Punktwolke als Stützpunkt gewählt wird, der den kleinsten Abstand zum ideal berechneten Punkt der approximierten Oberfläche besitzt. Dadurch werden große Aussreißer durch einen in der Nähe liegenden Punkt ersetzt.

Zum Vermaschen der Punkte werden die Punkte in Kugelkoordinaten[28] umgewandelt. Somit ist die Triangulierung wiederum auf den 2D-Fall mit den Koordinatenachsen φ und ϕ zurückgeführt. r wird als konstant betrachtet. Nach der Rücktransformation entsteht die fertige Form. Zwei einfache Beispiele zeigt Abbildung 30.

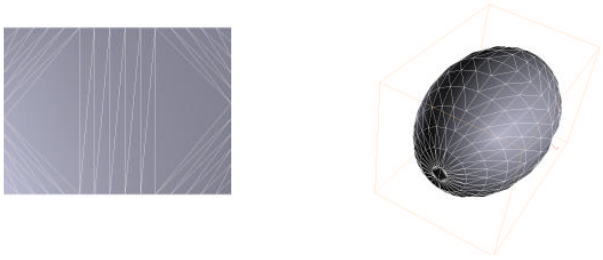


Abbildung 31: Einfache Rückgeführte Grundkörper. Ebene (links) und Ellipsoid (rechts)

3.6 Zusammenführen der gefundenen Einzelflächen

Um gefundene Flächen zu verbinden, wird ein im Softwarepaket Lightwave 8.0 von Newtek vorhandener Holefilling-Algorithmus verwendet. Dieser schließt auftretende Löcher, welche bei der Vermaschung im polaren Koordinatensystem entstehen. Im gleichen Arbeitsgang werden die auftretenden Zwischenräume der Einzelflächen geschlossen.

Wenn gering gekrümmte Flächen als Ellipsoid interpretiert werden, können sehr lange Kanten entstehen. Diese Kanten liegen in der Größenordnung des betrachteten Flächenstückes. Diese werden entfernt.

[28] siehe Anhang

3.7 Einordnung des Algorithmus

Der Algorithmus ist nach der in Kapitel zwei vorgestellten Einteilung als Kombination verschiedener Verfahren zu sehen. Er enthält Elemente eines oberflächenorientierten Wachstumsalgorithmus, aufgrund des schrittweisen Hinzufügens von Datenpunkten. Ebenso enthält er die Vorgehensweise von Clusteralgorithmen, da er Teile der Punktwolke zu einem Objekt/Flächentyp zuordnet, beziehungsweise Datenpunkte als zusammengehörig zusammenfasst. Als nicht sofort erkennbares Element enthält er einen wesentlichen Charakterzug der Warping-Algorithmen. Durch die Verwendung der Originalpunkte zur Vermaschung nähert er sich ausgehend von einer groben Oberflächenschätzung, der Quadrik, an die Originaloberfläche an. Diese Wesenszüge des Algorithmus unterscheiden ihn von allen zu diesem Zeitpunkt bekannten Algorithmen, welche meist nur einen konkreten Charakter aufweisen. Bekannte Verfahren weisen maximal eine Kombination von zwei Verfahrensweisen auf. Des weiteren ist die Anwendung einer Punktreduktion im Algorithmus neu. Dies sind bislang immer getrennte Verfahren gewesen, mit mindestens zwei Parametern. Hier wird nur α benötigt. Wird die durch α auf der Quadrik überstrichene Weite w kleiner als $\triangle mpa$ gewählt, so entspricht das Ergebnis einer Rückführung ohne Reduktion. Somit ist mittels α der Reduktionsgrad beeinflussbar.

Ein weiterer Vorteil besteht in der Modellierbarkeit von Löchern und mehreren Objekten in einer Szene. Dies ist über den Parameter $\triangle mpa$ steuerbar. Über die Abbruchkriterien ist die Detailgenauigkeit des Algorithmus beeinflussbar. Dies ist vor allem notwendig, wenn stark gestörte Aufnahmen verwendet werden. Hilfreich kann auch der Entscheidungsparameter ξ sein, der es erlaubt, den Algorithmus zwischen tendenzieller Volumenbestimmung und ebenen Flächen auszutarieren. In diesem Zusammenhang kann auch die Fehlermessung von Aufnahmen eine Rolle spielen.

Auch wenn die augenscheinliche Einfachheit des Verfahrens die Komplexität versteckt, so werden im folgenden Kapitel einige Effekte beschrieben, deren Auswirkungen ungewöhnlich sind. So zum Beispiel die Auswirkung der Durchlaufstrategie der Punkte auf die Genauigkeit der Torsion.

Da die Parameter auf dem Aufnahmesystem basieren, ist der Algorithmus trotzdem für die automatische Rückführung von Flächen geeignet. Es wird in den folgenden Kapiteln auch diskutiert werden, ob im hier vorliegenden Fall nicht auf die Zusammenführung der Punktwolken verzichtet werden kann, um somit große Teile der Vorverarbeitung einzusparen. Diese Überlegung ergibt sich aus der vorliegenden Robustheit gegenüber starken Störungen.

4. Entstehung des Verfahrens

In diesem Kapitel soll die Entstehung des Verfahrens erläutert werden. Zudem werden einige Aussagen über die Genauigkeit gemacht. Des weiteren werden mögliche nicht verwendete Teilverfahren erörtert. Ebenso Erwähnung finden einige Irrwege, welche während der Entwicklung gegangen wurden.

4.1. Zusammenführen der Punktwolken

Zwei Aspekte sind beim Zusammenführen zu beachten:

1. Der benötigte Minimalabstand der Punkte für die nächste Nachbarbestimmung

2. Ist das Zusammenführen der Punktwolken überflüssig ?

Der erste Aspekt zielt darauf ab, dass nicht der typische kd-tree [TF2004] zur Bestimmung der nächsten Nachbarn eingesetzt wurde. Die hohe Punktdichte und Punktanzahl in den Daten macht den Aufbau des Baumes sehr langsam. Der getestete Algorithmus zum Aufbau des Baumes verwendet das „divide and conquer“ –Prinzip zur Unterteilung des Raumes. Dieses Prinzip wächst in seiner Komplexität mit der Punktanzahl. Da die Punkteanzahl sehr hoch ist, war es ebenso effektiv, eine Abstandsliste zu berechnen. Der Komplexitätsunterschied besteht vorwiegend in den Operationen auf der Datenstruktur.

Der zweite Aspekt zeigt auf, dass auf die Vorverarbeitung verzichtet werden kann und Fächerfehler (Abbildung 32) in der Aufnahme als Volumen interpretiert werden können. So kann ein solches Volumen den Messfehler darstellen. Dies ist möglich, indem Punkte im Inneren einer Quadrik als gültig betrachtet werden. Dieser Fehler tritt nur an den Blättern der Pflanze auf Abbildung 33 zeigt dieses Verhalten an aufgefächerten Blättern der Gerstenpflanze.

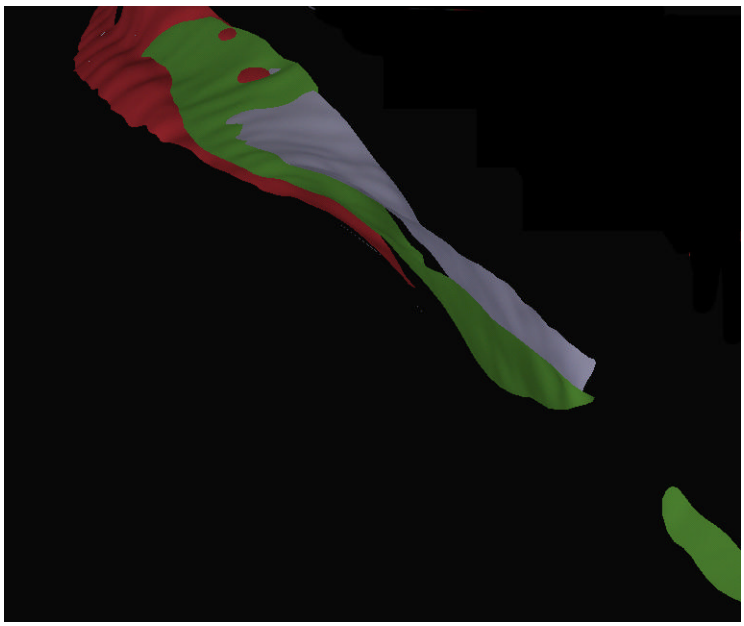


Abbildung 32: Auffächern der Scans, welches durch die Dämpfungswirkung des Algorithmus unterdrückt werden kann

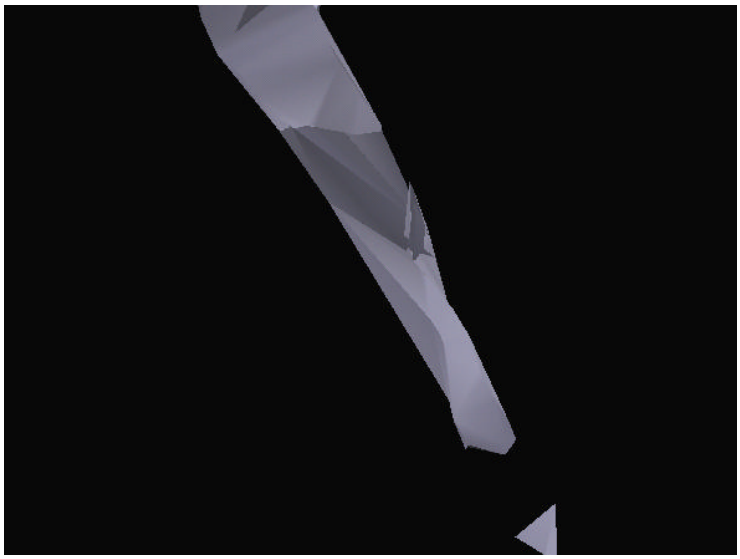


Abbildung 33: gedämpfte Rückführung der Auffächerung aus der obigen Abbildung

4.2. Quadrikenfindung

4.2.1 Entstehung des Verfahrens zur Parameterschätzung

Nach der Entstehung der Idee wurde sie zuerst mittels einer Newtoniteration auf ihre Machbarkeit überprüft. Nachdem die Iteration der Achsen gute Ergebnisse lieferte, wurde versucht, die Gleichung der Quadrik zu linearisieren. Dies ist möglich unter der Bedingung, dass ein Parameter bekannt ist. Durch Umstellen der Gesamtgleichung wird die Form einer Geradengleichung $y = mx + c$ erreicht. Die Anwendung der Regression auf dieses Problem gelang nicht. Es zeigte sich, dass der Parameter b noch zufriedenstellend angenähert werden konnte, jedoch der Parameter c in vielen Fällen kein interpretierbares Ergebnis ergab.

Ein Normalenausgleichsverfahren wurde ausgeschlossen. Den Erfahrungen am DLR und auch den Erfahrungen von Herrn Wernecke zufolge war es nicht möglich, eine Normale zu bestimmen, welche eine brauchbare Grundlage zu einer weiterführenden Berechnung liefern könnte. Dies war auch der Hauptgrund, ein Verfahren zu entwickeln, welches gänzlich auf die Verwendung der Normalen in den Messpunkten verzichtet. Tabelle 1 zeigt den Unterschied zwischen Regression und der Pseudoinversen-Methode anhand der Ergebnisse mit den Grundkörpern.

	Regression	**Pseudoinverse Methode**	**orginal**
Kugel	a = 0.100 b = 0.103 c = 0.128	a = 0.100 b = 0.107 c = 0.110	a = b = c = 0.1
Ellipsoid	a = 0.149 b = 0.008 c = 0.125	a = 0.149 b = 0.101 c = 0.098	a = 0.15 b = c = 0.1
Ebene (x-y-Ebene)	a = 0.100 b = 0.000 c = 0.000	a = 0.100 b = 0.000 c = 0.000	Ebene in der x-y-Ebene
Ebene schief im Raum	a = 0.100 $b = -9.616 \cdot 10^{-16}$ $c = -9.616 \cdot 10^{-16}$	a = 0.100 b = 0.000 $c = -9.616 \cdot 10^{-16}$	

Tabelle 1: Messwertevergleich der Methoden, bezogen auf einen maximalen Wertebereich $W\{-1;1\}$

In der Festlegung der ersten Achse ist ein weiterer Vorteil enthalten. Viel Zeit nahm die Behandlung der Singularität der Quadriken in Anspruch. Aus der Tatsache, dass die erste Achse als der längste Abstand in der betrachteten Punktmenge definiert ist, ergibt sich folgende Schlussfolgerung: Wenn eine angenäherte Achse unendlich lang ist, so ist auch die festgelegte Achse unendlich. Ist die festgelegte Achse sehr viel größer als die angenäherte Achse, so ist die Quadrik durch einen Zylinder beschreibbar. Im Falle eines sehr langen und dünnen Ellipsoids würden die „Kappen des Ellipsoids“ getrennt berücksichtigt werden (siehe Abbruchkriterien in Kapitel 3). Es entsteht ein Zylinder mit zwei Halbkugeln. Der Fall der Ebene ist durch die Fehlerfunktion eindeutig definiert und somit trennbar von den anderen auftretenden Fällen. Die Fehlerfunktion liefert für Ebenen den Wert 1. In der Praxis kann man feststellen, dass an der vorliegenden Gerstenpflanze sehr häufig Ellipsoide Flächen entstehen, die innerhalb der Fehlertoleranz eine rotationssymmetrische Interpretation erlauben. Diese Beobachtung ist nicht auf alle Objekte zu übertragen.

4.2.2. Verwendung der Transformationen

Der Algorithmus verwendet die direkt zugänglichen Euler-Transformationen. In Betracht kommt auf den ersten Blick auch die Benutzung von Quaternionen [CT2004]. Diese beschreiben Rotationen mittels Zahlen mit drei komplexen Anteilen. Somit sind Rotationen mit einer einzigen Operation berechenbar. Mit Quaternionen können folglich Rotationen um freie Achsen unter Berücksichtigung der Eigenrotation beschrieben werden. Der ausschlaggebende Nachteil besteht in der hohen Anzahl der benötigten Sinus- und Kosinus- Berechnungen zur Berechnung der komplexen Anteile. Diese sind meist sehr ungenau und eignen sich zur weiteren Verwendung meist nur bedingt. Viele Grafik-Echteit-Systeme benutzen daher intern von vorneherein Quaternionen. Diese geschickte Möglichkeit ist hier nicht anwendbar.

Bei der Verwendung der Transformationen wurde die Ungenauigkeit der Sinus- und Kosinus-Routinen berücksichtigt. Die Routinen besitzen die Eigenschaft, bei Rotationen um einen Gesamtwinkel von 360° sehr gut zu funktionieren. Da jede Transformation wieder rückgängig gemacht wird, geschieht dies durch Anwendung der Rotation um $360° - \alpha$, wenn α der Winkel der ersten Transformation ist.

4.2.3. Punkte-Durchlauf-Strategie und Abbruchkriterien

4.2.3.1 Abbruch aufgrund der Achsenlänge

Ein wesentlicher Denkfehler wurde bei der Bestimmung der Abbruchkriterien gemacht. So wurde angenommen, dass kein geschätzter Parameter größer als der festgelegte Parameter sein kann. Dies ist falsch. „Unendlich“ kann in der hier vorliegenden Art der Berechnung jeden „sehr großen“ Wert annehmen. Zudem können auch Abweichungen bei stark abgeflachten Ellipsoiden entstehen.

4.2.3.2. Einfluss der Punkte-Durchlauf-Strategie auf die Torsion

Die auffälligste Eigenschaft des Algorithmus ist die Eigenschaft, dass die Genauigkeit der erzeugten Torsion im Objekt durch die Punkte-Durchlauf-Strategie beeinflusst wird. Zu Beginn wurde jeder Nachbar eines Punktes anhand der Abbruchkriterien hinzugefügt. Es erwies sich als eine deutliche Qualitätsverbesserung, den Durchlauf darauf zu beschränken, nur den nächsten ungenutzten Nachbar eines jeden Punktes in die Betrachtung einzubeziehen. Abbildung 34 zeigt dies an einem Beispiel. Zu erkennen ist, dass der erste Ansatz in Abbildung 34 deutlich mehr kleine Flächen liefert, als der verbesserte Ansatz in Abbildung 35.

Abbildung 34: Unsprüngliches Verfahren unter Benutzung eines jeden Nachbarn.

Abbildung 35: Durch Verwendung des jeweils nächsten Nachbarn verändert sich die Torsion zu größeren Flächen.

4.3. Punktreduktion

4.3.1. Kenntnis der polygonalen Gesamtrepräsentation

Da zur Reduktion keine Kenntnis über die polygonale Repräsentation der Punktwolke notwendig ist, kann auf die rechenintensive Delaunay-Tiangulierung/Tetraederunterteilung der gesamten Punktwolke verzichtet werden. Aufgrund der Umrechnung in Kugelkoordinaten kann die eigentliche Berechnung der zu selektierenden Tetraederseiten auf den 2D-Fall reduziert werden. Die dringende Notwendigkeit einer solchen Art der Reduzierung, respektive Triangulierung, ergibt sich aus den oftmals nicht nachvollziehbaren Ergebnissen aus anderen Veröffentlichungen[29]. Trotz Nutzung derselben Geometrie-Bibliothek[30] zur Berechnung der Tetraeder/Dreiecke konnten die angegebenen Zeiten aus anderen Veröffentlichungen nicht nachvollzogen werden. Es liegt die Vermutung nahe, dass die Angaben sich nur auf die Berechnungszeit der jeweiligen Verfahren beziehen, und die Tetraederunterteilung zur Vorverarbeitung gezählt werden. So beträgt nach Angaben im Handbuch der CGAL-Bibliothek die Komplexität verschiedener Operationen [CG2004]:

Finden des nächsten Nachbars im Voronoidiagramm	$O(n)$
Entfernen eines Punktes des Grades d	$O(d^2)$
Anzahl der Flips pro eingefügten Punkt vom Grad d	$O(d)$

Tabelle 2: Komplexitäten des Delaunay-Algorithmus

[29] Bei Amenta und Mücke sind Empfehlungen zur CGAL Biliothek zu finden

[30] CGAL Version 2.4 (http://www.cgal.org/)

4.3.2. Reduktionswinkel α

Der Reduktionswinkel α ist ein sehr flexibel einsetzbares Element. Während die klassischen Edge-collapse-Verfahren eine umständliche, kaum vorstellbare Pseudo-Energiefunktion zur Bestimmung des Reduktionsgrades benutzen, ist der Reduktionswinkel α sehr einfach definiert. Er führt zu einem direkt vorstellbaren Ergebnis. Die Energiefunktion des Edge-collapse-Verfahrens dagegen reduziert in Abhängigkeit des Detailgrades des Objektes. Dies bedeutet, dass ein detailreiches Objekt evtl. unkenntlich wird. Durch Bestimmung der Flächen bleiben die Details in dem durch α definierten Maß erhalten. Die Annahme von 5° als einen guten Wert für α resultiert aus den Angaben in der OpenGL Standard Literatur [WS1999]. 5°-Abstände sind ein durch ein Gouraud-Shading kaschierbares Minimum. Bei größeren Gradzahlen müsste ein NURBS oder Subdivision Modell gewählt werden. Abbildung 36 und 37 zeigen einen mit $\alpha = 5°$ rekonstruierten Ellipsoiden. Zum einen als Gitternetz in Abbildung 36. Zum anderen als ein mit Flatshading gerendertes Modell in Abbildung 37.

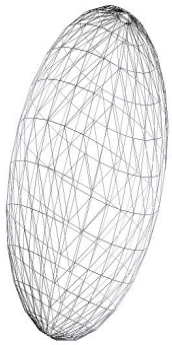


Abbildung 36: Wireframe eines mit $\alpha = 5°$ reduzierten Ellipsoids

Abbildung 37: Ellipsoid mit Flatshading und $\alpha = 5°$

4.3.3. Krümmungen

Grundlegend für die Punktreduktion ist die Wahl der Krümmung. Als Maß für die Krümmung kann der Winkel zwischen Tangentialebene und einer Koordinatenachsenebene ausgeschlossen werden. Für dieses Verfahren wäre Kenntnis über die Normale notwendig. Dies wurde jedoch in den vorangegangenen Kapiteln ausgeklammert. An das zu wählende Krümmungsmaß wird folgender Anspruch gestellt:

1. Es soll die quadratische Eigenschaft der Parameter a, b und c wiedergeben

2. Es soll als Formel direkt anwendbar sein

Zur Wahl standen die Gausssche Krümmung und mittlere Krümmung. Die mittlere Krümmung gibt die quadratische Eigenschaft nicht wieder. Aus diesem Grund wurde die Gausssche Krümmung als Maß gewählt.

5. Ergebnisse

5.1 Grundkörper

Die im Folgenden gezeigten Körper sind mit einem $\alpha = 5°$ und $\Delta mpa = 0.15$ berechnet worden.

Ebene

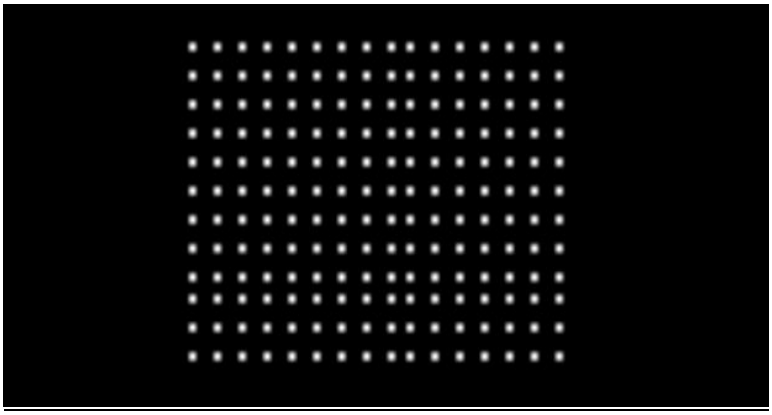


Abbildung 38: Original Punktwolke einer Ebene als Screenshot aus der Beispielanwendung (oben). Reduzierte Ebene (unten). Die Fläche wurde mit Flatshading gerendert und die Polygone eingezeichnet.

In Abbildung 38 ist die Wirkung der Ebenen-Reduktion ersichtlich. Deutlich zu erkennen ist, dass nur die Randpunkte der Ebene in regelmäßigen Abständen verbleiben. Charakteristisch für die Randpunkte ist, daß sie in der Triangulierung einen infiniten Nachbar besaßen und somit identifizierbar sind.

Kugel

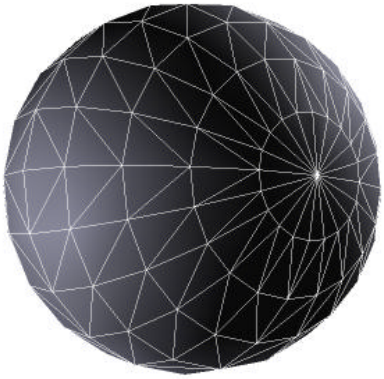


Abbildung 39: Reduzierte Kugel. Sie wurde mit Flatshading gerendert und die Polygone eingezeichnet.

Die in Abbildung 39 gezeigte Kugel veranschaulicht die regelmäßige Triangulierung, die der Algorithmus erzeugt. Der Grund der Regelmässigkeit ist die konstante Krümmung auf der gesamten Kugel. Somit besitzen die Krümmungsmaßzahlen kr_{ab} und kr_{ac} immer den Wert 1.

Ellipsoid

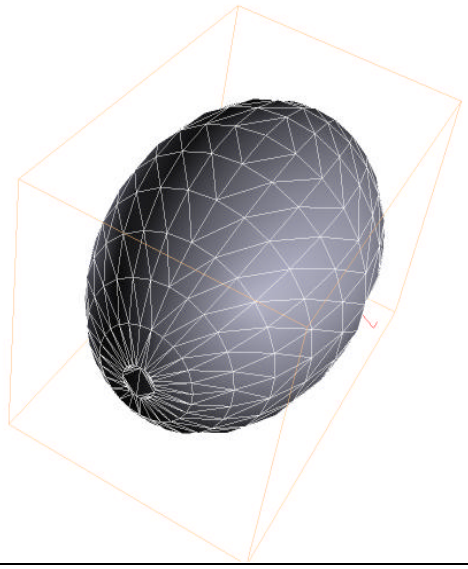


Abbildung 40: : Reduziertes Ellipsoid. Es wurde mit Flatshading gerendert und die Polygone eingezeichnet.

Das Ellipsoid in Abbildung 40 zeigt die Wirkungsweise der unterschiedlichen Krümmungen. Zu erkennen ist dies an der längeren Form der einzelnen Polygone. Die hier ebenfalls eingezeichnete Bounding Box soll einen Eindruck über die Lage des Ellipsoids im Raum geben.

d5.2 Pflanzenteile

Blattauschnitt (realer Zylinder)

Der hier gewählte Blattausschnitt ist von einem Gerstehalm. Abbildung 41 zeigt die Originalaufnahme. Abbildung 42 (links) zeigt ein Gittermodell. Das dazugehörige Flächenmodell ist in Abbildung 42 (rechts) und 43 zu entnehmen. Im Vergleich ist das Verhalten des Parameters α ersichtlich.

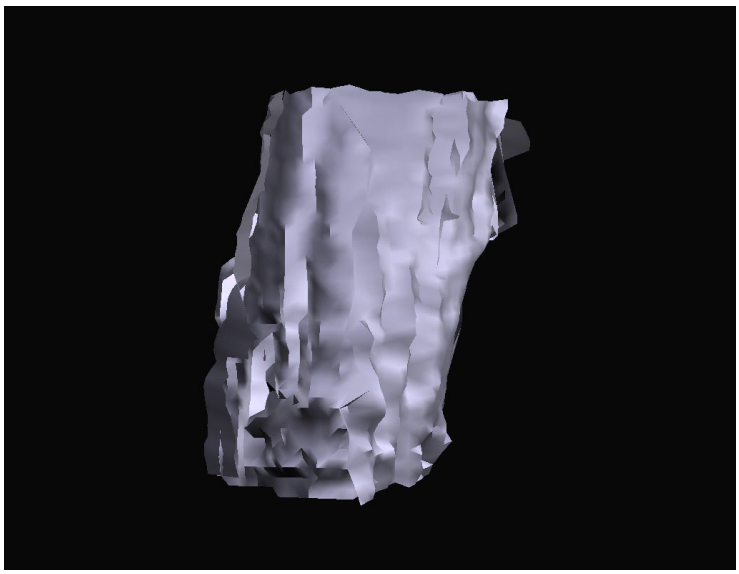


Abbildung 41: Orginalausschnitt aus dem Stengel einer Gerste
(mit einfacher Triangulation überalle Punkte dargestellt)

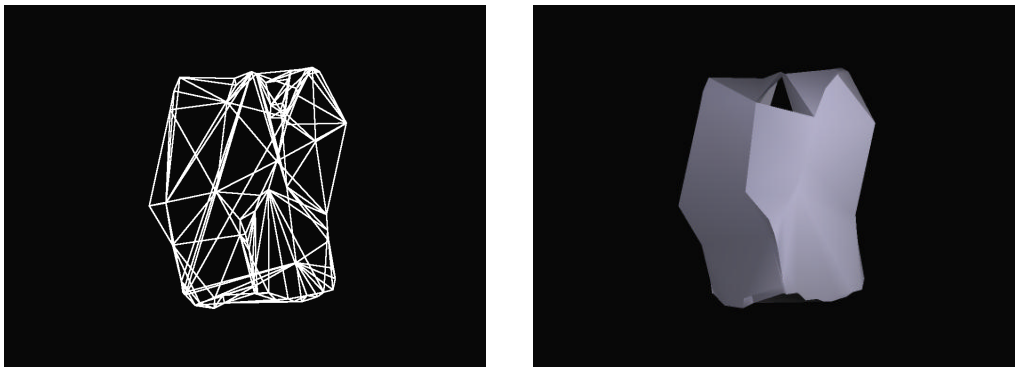


Abbildung 42: Reduziert mit verschiedenen α - Winkeln in den zx- und yx-Richtungen (5° und 45°)

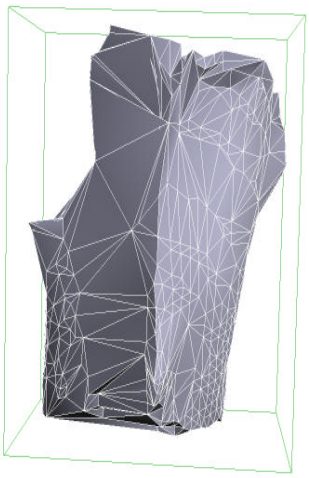


Abbildung 43: reduzierter Stengelausschnitt mit $\alpha = 10°$

Blattausschnitt (reale Ebene)

Anhand der Abbildungen 44 und 45 ist die Wirkung des Parameters ξ ersichtlich. Ersichtlich ist der Unterschied zwischen der Interpretation eines Blattausschnittes als ellipsoide Fläche (Abbildung 44) und der Interpretation als Ebene (Abbildung 45).

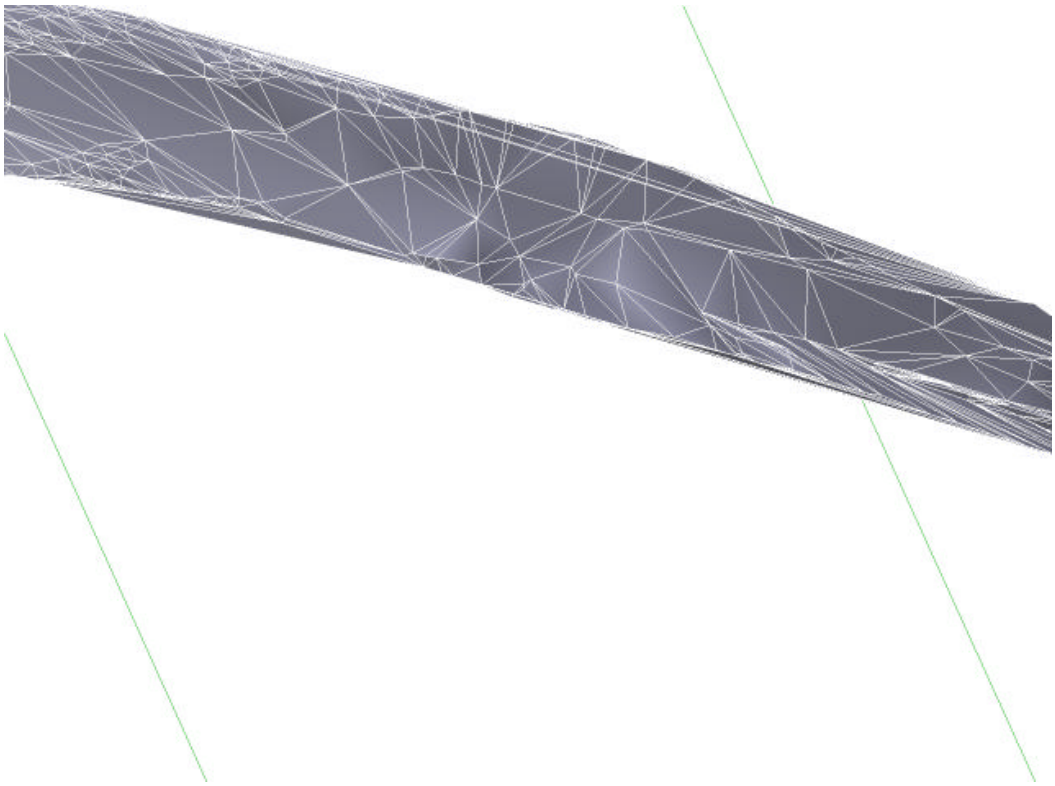


Abbildung 44: Blatt mit $\alpha = 10°$ und als Oberfläche eines Ellipsoids interpretiert

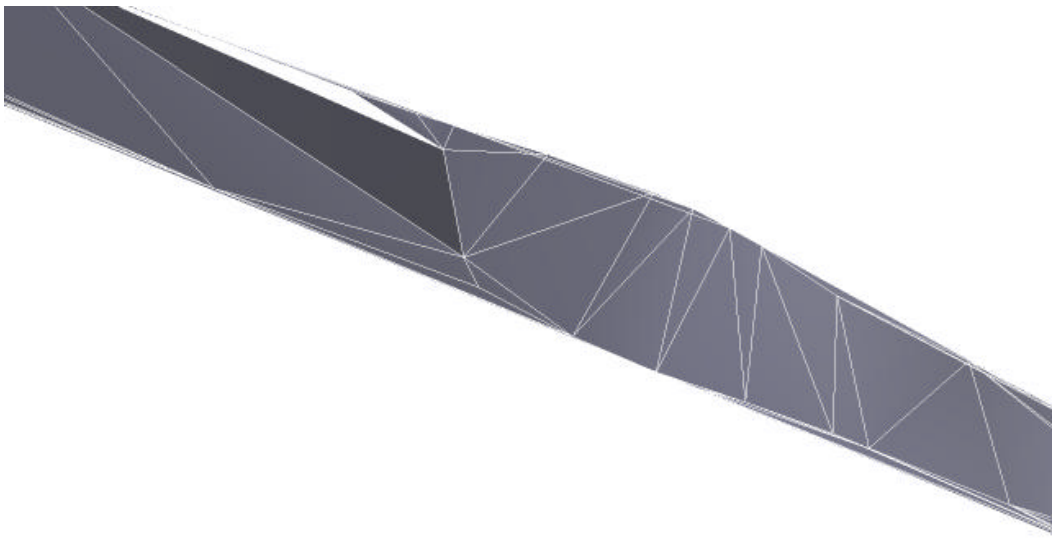


Abbildung 45: zwei verschiedene α-Werte (5° und 45°) und Interpretation als Ebene

Blattauschnitt (ganzes Blatt mit Stengel)

In der Abbildung 46 ist ein gesamtes Blatt mit Torsion zu erkennen. Wiederum wurde $\alpha = 10°$ gewählt, um einen Eindruck über das Verhalten und den möglichen Reduktionsgrad zu geben. Die Eingangspunktwolke in Abbildung 47 besitzt 20193 Punkte. Das in Abbildung 46 gezeigte Ergebnis besteht aus 889 Vertices. Im hinteren Bereich ist das Chaos erkennbar aus Abbildung 1. Dieses wird im folgenden Abschnitt noch genauer erläutert.

Abbildung 46: gesamter Stengel mit $\alpha = 10°$

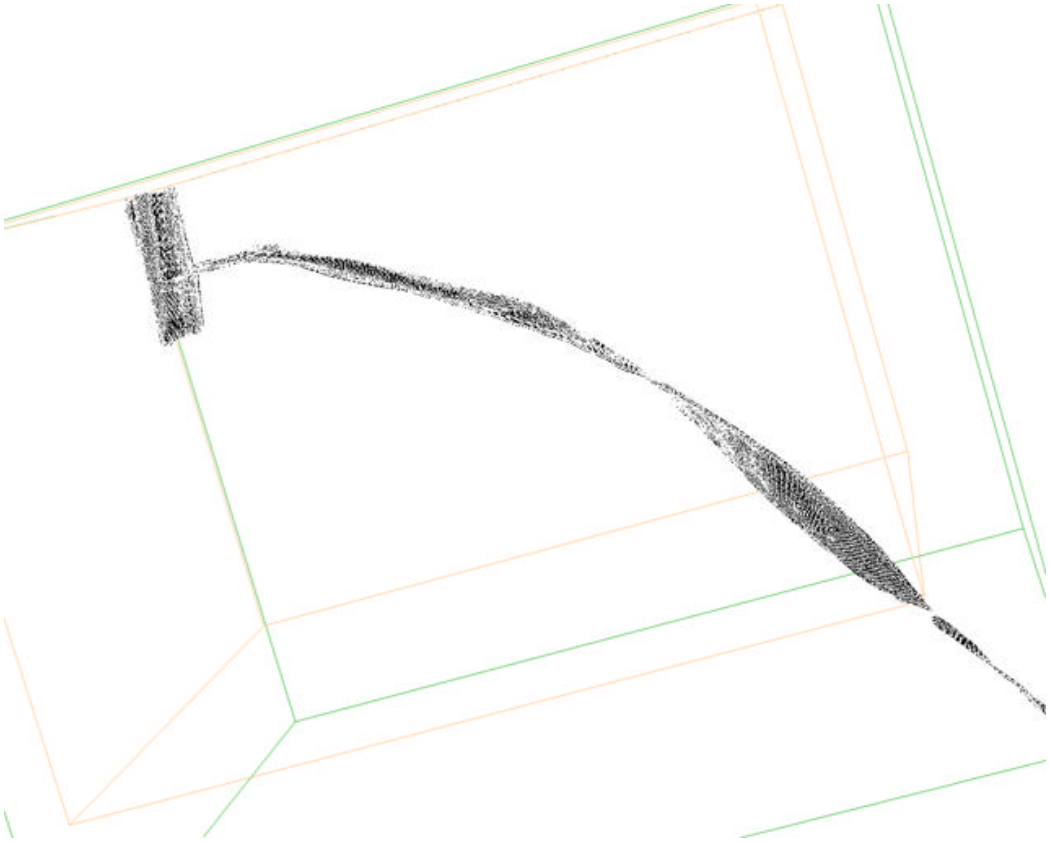


Abbildung 47: Original des in Abbildung 46 gezeigten Ergebnisses.

5.3 Vollständige Gerstenpflanze

Das Gesamtergebnis ist in Abbildung 48 und 49 visualisiert. Weiß symbolisiert einzelne Messwerte beziehungsweise Vertices im Ergebnismodell in Abbildung 49. Schwarz stellt die triangulierten Flächen dar. Das Original besteht aus 88354 Messpunkten. Das reduzierte Modell besteht aus 3098 Vertices.

Erkennbar sind:

- die nicht rekonstruierbaren Teile,
- die Reduktion der Punktdichte,
- Reaktion des Algorithmus auf verschiedene Störungen.

Abbildung 48: Originalpunktwolke der Gerstenpflanze

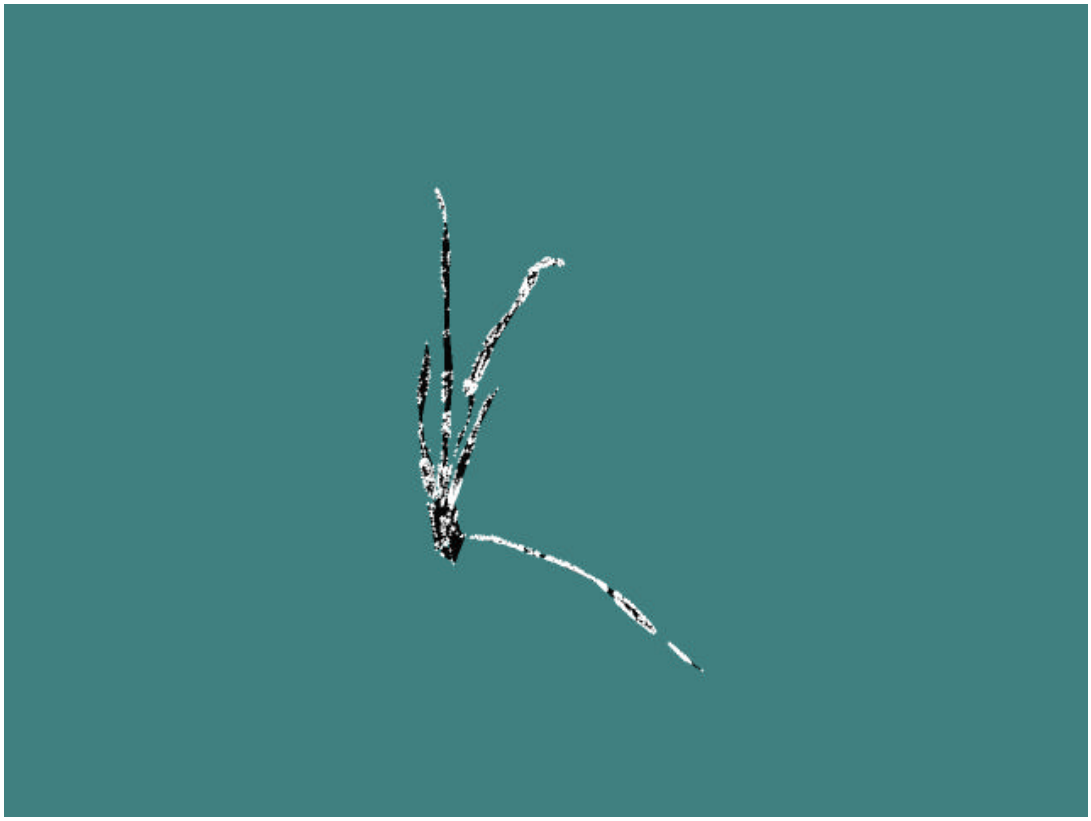


Abbildung 49: Rekonstruierte Gerstenpflanze mit $\alpha = 10°$

5.4. Verhalten bei Störungen

In der folgenden Abbildung 50 wird das Verhalten bei Störungen gezeigt. Auffällig ist vor allem die Behandlung von stark gestörten und „Chaosaufnahmen“. Dieser Ausschnitt zeigt eine sehr stark gestörte Aufnahme aus dem Bereich eines Überganges von Blatt zu Stengel. Es ist erkennbar, dass der Algorithmus eine Hülle um den chaotischen Bereich bildet. Des weiteren ist in zusammenhangslosen Bereichen eine Lochbildung erkennbar. Zum Vergleich kann Abbildung 1 herangezogen werden, sie zeigt das Original.

Abbildung 50: Chaoswerte im Bereich des Stengels

Verbesserungen des Ergebnisses könnten durch Verkleinerung von α und Vergrößerung des Wertes Δmpa erzielt werden, da in diesem Bereich der Aufnahme nicht die angegebene Messgenauigkeit erreicht wird. Dies führt jedoch zu einer Verschlechterung des Gesamtergebnisses.

Es sei auf die Behandlung der Doppelflächen verwiesen, welche in Kapitel 4 bereits behandelt wurde, jedoch hier nicht mehr aufgeführt wird.

5.5. Vergleich mit anderen Algorithmen

In diesem Abschnitt soll anhand realer Daten an Ausschnitten gezeigt werden, wie Vorteile gegenüber anderen Algorithmen zu erreichen sind. Desweitern wird gezeigt, welche Freiräume der Algorithmus lässt.

Die Modellierbarkeit mehrerer Objekte in einer Szene wird in Abbildung 51 dargelegt. Anhand der als Doppelflächen auftretenden Messungenauigkeiten wird gezeigt, wie die vorgelagerte Ebene (rot) von dem restlichen Stengel (grau) getrennt wird. Dies verdeutlicht, dass durch ξ auch Abstände kleiner als die Messgenauigkeit des Aufnahmegerätes aufgelöst werden können.

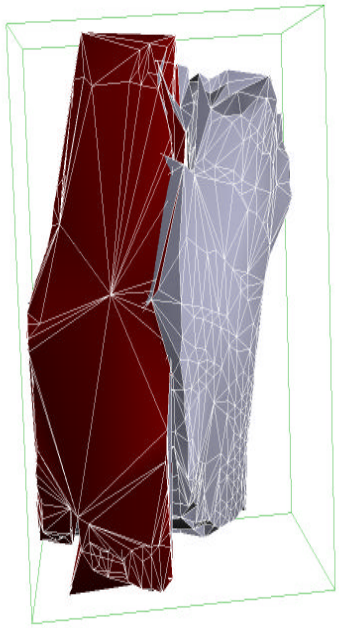


Abbildung 51: vorgelagerte Ebene (rot) von dem restlichen Stengel (grau) getrennt

Anhand der in Abbildung 51 gezeigten Rückseite des Stengels soll gezeigt werden, an welcher Stelle ein Holefilling ansetzt. Das entstandene Loch hat seinen Ursprung in der Polarvermaschung. Abbildung 51 und 52 sind des weiteren völlig unbehandelt. Es wurde

keinerlei Verbesserung des Ergebnisses angewendet. Erkennbar ist daraus auch, wie einzelne Streupunkte unterdrückt werden. Der darauf folgenden Abbildung 53 kann entnommen werden, dass es nicht immer sinnvoll ist, das Ellipsoid zu schließen, da es sich um eine Ebene handeln kann. Aus diesem Grund ist die Unterscheidung im Nachhinein notwendig. Das Loch in Abbildung 52 ist als Loch identifizierbar und füllbar.

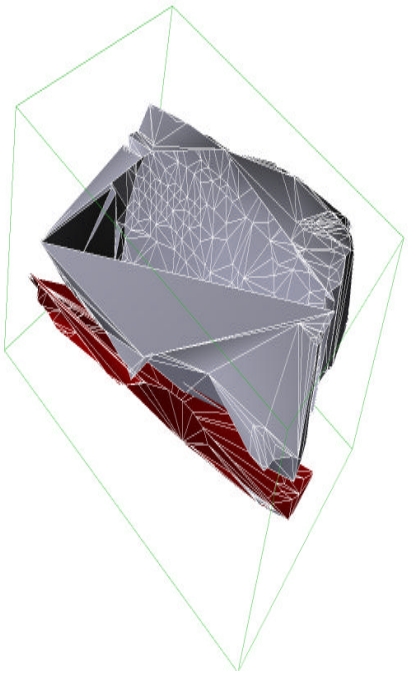


Abbildung 52: nicht geschlossenes Ellipsoid

In Abbildung 53 wird ein Ausschnitt eines Blattes gezeigt. Dargestellt wird eine Möglichkeit, Löcher zu modellieren. In diesem Fall wurde ξ dahingehend variiert, dass der gezeigte Ausschnitt als Ellipsoid interpretiert wird. Eine weitere denkbare Möglichkeit ist die Interpretation als mindestens vier Teilebenen, welche sich um das Loch herum gruppieren.

Abbildung 53: reale Ebene mit Loch. Die Ebene wurde über Variation von ς als Ellipsoid interpretiert

6. Fazit und Ausblick

6.1. Einordnung

Der in dieser Arbeit vorgestellte Algorithmus unterscheidet sich in einigen wesentlichen Punkten von bisher bekannten Verfahrensweisen. Während heutige Verfahren immer die gesamte Punktwolke triangulieren, und diese in einem späteren Schritt reduzieren, beruht das hier vorgestellte Prinzip auf der Triangulierung der benötigten Punkte.

Ein zweiter wesentlicher Unterschied besteht darin, dass die Raumunterteilung zur Triangulierung zweidimensional erfolgen kann. Dies beruht auf den zuvor gefundenen Quadriken.

Der dritte auffällige Unterschied ist in der gesamten Arbeitskette zu sehen. Während beim Ablauf einer klassischen Rekonstruktion erst ein Detailmodell mit allen Punkten erzeugt wird, reduziert dieses Verfahren die Punktanzahl vor der eigentlichen Triangulierung. Hierfür wird der bestmögliche Punkt anhand des Abstands zu der Quadrik gewählt. Aufgrund dieser Vorgehensweise werden Ausreißer nicht in das Modell miteinbezogen.

Trotz der Unterschiede erlaubt das Verfahren die Modellierung von Löchern, welche in vielen der in Kapitel zwei vorgestellten Verfahren ein Problem ist. Ebenso erlaubt es theoretisch die Rückführung mehrerer Objekte innerhalb einer Szene.

Ein ebenso interessanter Aspekt ist die Möglichkeit, in beschränktem Maße Aussagen über die vorliegenden Teilflächen zu treffen.

6.2. Mögliche Erweiterungen

Schwächen besitzt das Verfahren bei der Behandlung von negativ gekrümmten Flächen. In der momentanen Form werden diese durch eine hohe Anzahl von Flächen dargestellt. Es bleibt zu untersuchen, ob eine höhere Reduktion durch die Bestimmung der negativ gekrümmten Fläche als Quadrik möglich ist.

Anzuraten ist zur Verbesserung der Ergebnisse die genaue Bestimmung des Messfehlers an realen Grundkörpern, welche mit dem Scanner aufgenommen wurden.

Das Verfahren könnte als Verfahren zur Entfernung von Fehlmessungen benutzt werden. Die Entfernung von Ausreißern durch die Wahl eines sehr kleinen α – Wertes ist denkbar. Dies müsste jedoch verifiziert und der Algorithmus in seiner Implementierung verändert werden.

6.3. Einsatzgebiete

Mögliche Einsatzgebiete sind in der Rekonstruktion von Objekten zu sehen, welche als Einzelpunktmessung in Form einer Punktwolke vorliegen. Diese kann durch 3D-Scanner, aber auch durch Stereographie entstehen. Das Einsatzgebiet ist dabei nicht ausschließlich in der Visualisierung zu sehen, sondern auch in weiterer analytischer Behandlung der Objekte. Diese Schlussfolgerung ist möglich, weil:

1. Die Originalmessdaten nicht verändert werden und

2. Aussagen über die Teilflächen getroffen werden können.

Anhang A – Glossar

Chamfer Distanz [Fou2004]

Eine Chamfer-Maske ist eine Anzahl gewichteter Vektoren $M_c = \left\{\left(\vec{v_i}, \omega_i\right), 1 \le i \le m\right\}$.

Diese Vektoren besitzen ihr Zentrum in *O*, und sind zu diesem symmetrisch. Sie bilden mindestens eine Basis von E.

Ist eine Chamfer-Maske $\mathrm{M_c}$ und zwei Punkte A,B gegeben, dann ist der Pfad von A nach B eine Aneinanderreihung von Vektoren $\vec{v_i} \in M_c^g$ die von A nach B führen :

$$\overrightarrow{AB} = \sum_{\vec{v_i} \in M_c} n_i \vec{v_i}$$

Die Kosten W zum durchlaufen des Pfades P_{AB} sind beschrieben als:

$$W(P_{AB}) = \sum_{i=1}^{m} n_i \omega_i$$

Daraus ergibt sich für die Chamfer-Distanz d_c:

$$d_c = \min_{P_{AB}} W(P_{AB})$$

Gabriel-Graph

Der Gabriel-Graph (GG) besteht aus allen Kanten $\overline{pq}$ zwischen zwei Punkten p,q einer Punktwolke P, deren verbindende Kugel mit dem kleinstmöglichem Durchmesser keinen anderen Punkt einschließt.

Iso-Oberfläche

Eine Iso-Oberfläche I ist einer kontinuierlichen Fläche A angenähert. „Dabei wird für die Annäherung der Fläche A entschieden, ob die Ecken (in seltenen Fällen die Kanten) eines Raumelementes oberhalb oder unterhalb der Fläche A liegen. Die auf diese Weise extrahierte Oberfläche wird Iso-Oberfläche I genannt.“ [Hes2003] Der bekannteste Vertreter zur Extraktion einer solche Oberfläche ist der Marching Cubes Algorithmus.

Marching Cubes Prinzip [LC2005]

Das Prinzip teilt sich in 7 Schritte auf:

1: Unterteilung des Raumes in gleichmäßige Voxel und Wahl eines Start-Voxels.

2: Für jede Ecke eines Voxels wird entschieden, ob sie innerhalb der Oberfläche (Wert <= Isowert) oder außerhalb der Oberfläche (Wert > Isowert) liegt.

3: Erstellen eines Index

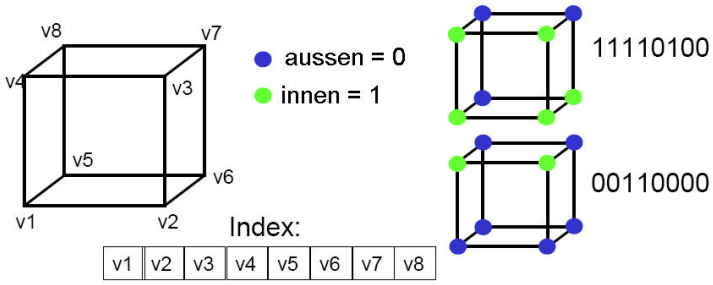


Abbildung 54: Indexerstellung des Marching-Cubes-Algorithmus

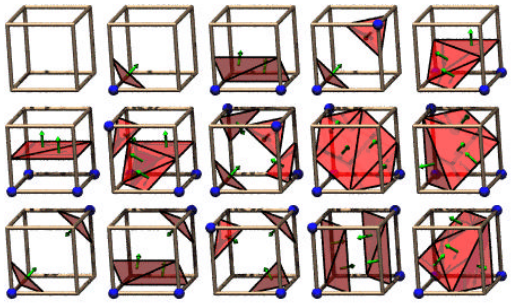


Abbildung 55: Mögliche Kantenkombinationen des Marching-Cubes-Algorithmus

4: Benutzen einer Look-up Table, in der alle Kantenkombinationen gespeichert sind.

5: Mittels des Indexes werden die zur Interpolation der Fläche notwendigen Dreiecke bestimmt und Mehrdeutigkeiten aufgelöst.

6: Die genaue Position der Ecken der Dreiecke auf den Kanten des Voxels wird mittels linearer Interpolation berechnet.

7: Die Normalen werden aus den Normalen der Ecken des Voxels interpoliert

Mediale Achse

Die mediale Achse wurde 1967 von Harry Blum zur Darstellung und Analyse ebener abgeschlossener Gebiete eingeführt. Sie besteht aus dem Abschluss der Menge aller Mittelpunkte maximaler Kreisscheiben. Eine Kreisscheibe heißt dabei maximal, sofern sie ganz in dem Gebiet enthalten und nicht echte Teilmenge einer ebenfalls in dem zu betrachtenden Gebiet liegenden Kreisscheibe ist. Auf der medialen Achse operiert die sogenannten Radiusfunktion. Sie ordnet jedem Punkt der medialen Achse den Radius der zugehörigen maximalen Kreisscheibe zu. Abbildung 53 zeigt ein 2D-Beispiel der medialen Achse.

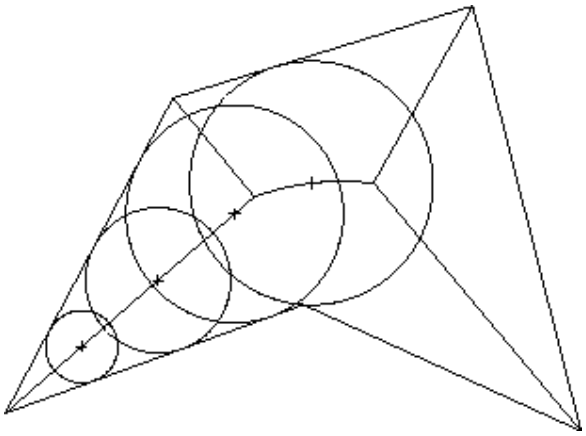


Abbildung 56: Konstruktion der medialen Achse

Minimal aufspannender Baum

Der Minimal aufspannender Baum (MAB) verbindet alle Vertices unter der Voraussetzung, dass die Summe aller Kantenlängen minimiert wird. Für die Kantenlänge ist die euklidische Distanz zu wählen.

Riemann-Graph

Der Riemann-Graph entsteht durch Hinzufügen von Kanten zum euklidischen minimal aufspannenden Baum. Eine Kante wird zwischen zwei Vertices des Baumes eingefügt, wenn der eine der beiden Vertices in einem festgelegten Nachbarschaftsverhältnis (k-Nachbarschaft) des anderen liegt. Das Ergebnis wird Riemann-Graph genannt.

Voronoi-Graph

Der Voronoi-Graph besteht aus den Kanten und Vertices des Voronoi-Diagramms.

Voronoi-Kugel

Eine Voronoi-Kugel ist eine Kugel, welche durch die Punkte eines Delaunay-Tetraeders läuft. Das Zentrum dieser Kugel ist ein Voronoi-Vertex. Dies ergibt sich aus der Dualität zwischen Delaunay-Triangulierung und Voronoi-Diagramm.

Anhang B – Benutzung von Quadriken

Grundkörper als homogene Quadrik (Beispiele):

Einheitskugel:

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Ellipsoid mit den Achsenlängen a,b,c:

$$Q = \begin{pmatrix} 1/a^2 & 0 & 0 & 0 \\ 0 & 1/b^2 & 0 & 0 \\ 0 & 0 & 1/c^2 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Unendliche Einheitsröhre:

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Transformationsmatrizen

Rotation um die x-Achse:

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation um die y-Achse:

$$R_y(\alpha) = \begin{pmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation um die z-Achse:

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Translation um Vektor $\nu(a,b,c)^t$:

$$T(\nu) = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Skalierung um λ:

$$S(\lambda) = \begin{pmatrix} \lambda_x & 0 & 0 & 0 \\ 0 & \lambda_y & 0 & 0 \\ 0 & 0 & \lambda_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Die inversen Matrizen der Transformationen:

$T(\nu)^{-1} = T(-\nu)$, $S(\lambda)^{-1} = S(1/\lambda)$, $R\ (\alpha) = R\ (-\alpha)$

Somit entfällt die aufwendige Berechnung der inversen Matrizen.

Anhang C – Beispielanwendung

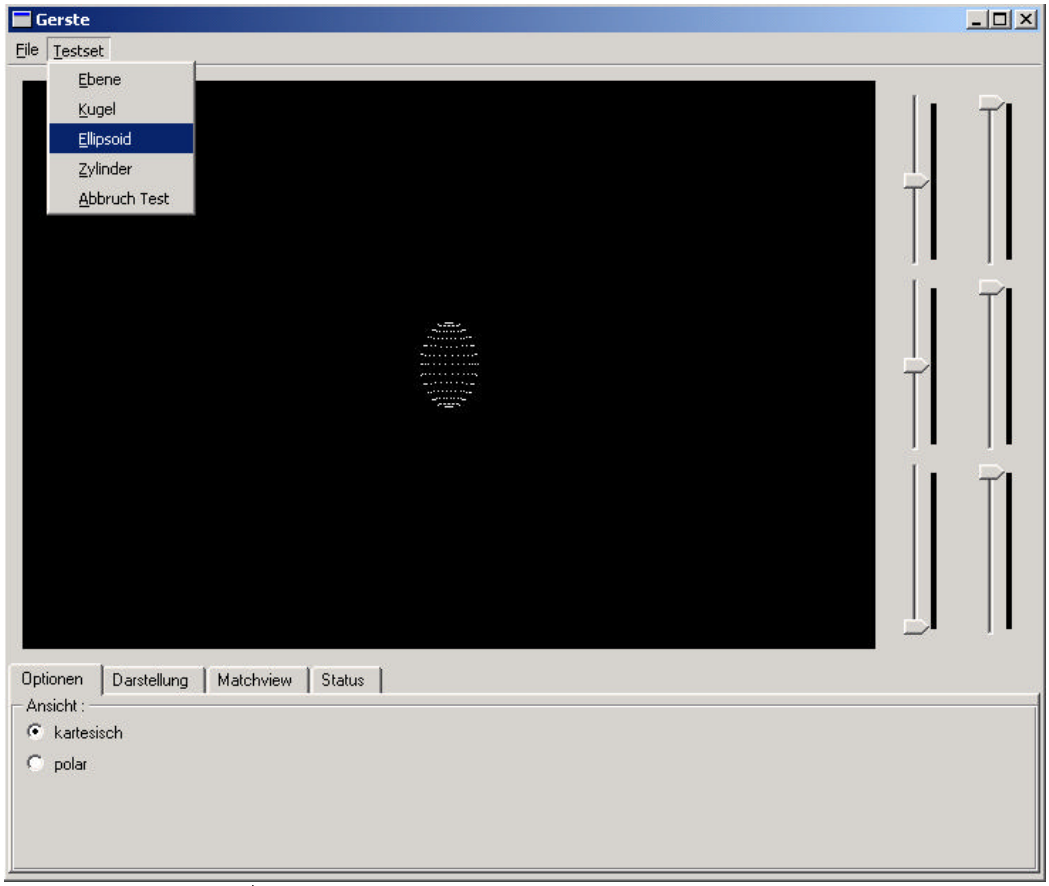


Abbildung 57: Oberfläche des Beispielprogramms

Menüpunkt „Testset“

Er enthält ideale Testkörper, welche zum Testen des Algorithmus verwendet werden.

Menüpunkt „File“

Enthält die Menüpunkte zu Laden der vrml-Files und alle Einzelschritte des Algorithmus zum Ausführen. Ebenso ist die gesamte Ausführung aller Schritte möglich. Ebenso wird die Möglichkeit zum Speichern des Ergebnisses angezeigt.

Vorschaufenster

Das Vorschaufenster zeigt die zu untersuchende Punktwolke. Mittels der Regler auf der rechten Seite kann vergrössert, verschoben und rotiert werden.

Karteireiter „Optionen“

Hier kann zur näheren Betrachtung der Punktwolke zwischen polarer und kartesischer Darstellung umgeschaltet werden.

Karteireiter „Darstellung“

In dem Karteireiter Darstellung kann zwischen 3D-Ansicht und auf die x-y-Ebene projizierter 2D-Ansicht gewählt werden.

Karteireiter „Matchview“

Dieser Karteireiter bietet die Möglichkeit, zwischen zusammengeführter und Originalpunktwolke interaktiv zu vergleichen

Karteireiter „Status“

Ein einfaches Ausgabefenster, in dem die Quadrikenergebnisse der betrachteten Punkte, das Ergebnis der Fehlerfunktion und die Ergebnisquadriken ausgegeben werden.

Literaturverzeichnis

[AB1998] N. Amenta and M. Bern. *Surface reconstruction by Voronoi filtering.* 14th ACM Symposium on Computational Geometry, Seite 39–48, 1998.

[AC1999] N. Amenta and S. Choi. *One-pass Delaunay filtering for homeomorphic 3D surface reconstruction.* Technical Report TR99-08, Universität Texas, Austin, 1999.

[ACK01] N. Amenta, S. Choi, and R. K. Kolluri. *The power crust, unions of balls, and the medial axis transform.* Computational Geometry: Theory and Applications, 19(2-3). Seite 127–153, 2001.

[AGJ2000] U. Adamy, J. Giesen, and M. John. *New techniques for topologically correct surface reconstruction.* In Proceedings of IEEE Visualization 2000. IEEE Computer Society Press, 2000. (aus [Men2001])

[AGJ2001] U. Adamy, J. Giesen, and M. John. *Surface reconstruction using umbrella filters.* Computational Geometry: Theory and Applications, 2001. (aus [Men2001])

[AM1998] U. Alt, K. Meisner. *Das Modell,* Praktikum der Computergrafik, Technische Universität Berlin 1998

[AS1996] M.-E. Algorri and F. Schmitt. *Surface reconstruction from unstructured 3D data. Computer Graphics Forum*, 15(1), Seite47–60, 1996. (aus [Men2001])

[AS2001] M. Attene and M. Spagnuolo. *Automatic surface reconstruction from point sets in space,* Genua 2001

[Att1997] D. Attali. *r-regular shape reconstruction from unorganized points.* In ACM Symposium on Computational Geometry, Seite 248–253. Nizza, Frankreich, 1997.

[BBX1995] C.L. Bajaj, F. Bernardini, and G. Xu. *Automatic reconstruction of surfaces and scalar fields from 3D scans.* Computer Graphics Proceedings, SIGGRAPH '95, Annual Conference Series, Seite 109–118, 1995.

[BBX1997] C. Bajaj, F. Bernardini, and G. Xu. *Reconstructing surfaces and functions on surfaces from unorganized 3D data.* Algorithmica, 19, Seite 243–261, 1997.

[BM1992] P. J. Besl, N. D. McKay, *A Method for Registration of 3-D Shapes*, IEEE Transactions on Pattern Analysis and Machiene Intelligence 14(2), Seite 239-256

[Boi1984] J.-D. Boissonnat. *Geometric structures for three-dimensional shape representation.* ACM Transactions on Graphics, 3(4), Seite 266–286, Oktober 1984. (aus [Men2001])

[BS1979] I.N. Bronstein; K.N. Semendjajew. *Taschenbuch der Mathematik.* 24. Auflage Thun, 1989.

[BTG1995] E. Bittar, N. Tsingos, and M.-P. Gascuel. *Automatic reconstruction of unstructured data: Combining a medial axis and implicit surfaces.* Computer Graphics Forum, 14(3), Seite 457–468, 1995. Proceedings of EUROGRAPHICS '95.

[CG2004] unbekannter Autor. *Delaunay Triangulation im Handbuch der Template Bibliothek* CGAL http://www.cgal.org/Manual/doc_html/frameset/fsBasic.html (zuletzt besucht:12.12.2004)

[EM1994] H. Edelsbrunner and E.P. Mücke. *Three-dimensional alpha shapes.* ACM Trans. Graph., 13(1), Seite 43-72, Januar 1994.

[Fis2000] K. Fischer. *Introduction to Alpha Shapes.* ETH Zürich 2000. http://n.ethz.ch/student/fischerk/alphashapes/as/as.html (zuletzt besucht am: 25.10.2004)

[Fou2004] C. Fouard und G. Malandain. *3-D chamfer distances and norms in anisotropic grids.* Image and Vision Computing, 2004 http://www-sop.inria.fr/epidaure/personnel/Celine.Fouard/ChamferMasks/ (zuletzt besucht: 21.11.2004)

[FS1992] P. Fua and P.T. Sander. *Segmenting unstructured 3D points into surfaces.* In G. Sandini, editor, Computer Vision : ECCV '92, Proc. Second European Conference on Computer Vision, Santa Margherita Ligure, Springer Verlag.Seite 676–680, Italien, Mai 1992.

[GKS2000] M. Gopi, S. Krishnan, C.T. Silva. *Surface reconstruction based on lower dimensional localized Delaunay triangulations.* Computer Graphics Forum, 19(3). Proceedings of EUROGRAPHICS 2000 , 2000.

[GS2002] M. Garland, E. Shaffer: *A Multiphase Approach to Efficient Surface Simplification* Proceedings of IEEE VIS 2002, Seite 117-124. 2002

[Haa2003] M. Haas, *Registrierung CT / MRT*, Seminararbeit Universität Karlsruhe, 2003

[Hes2003] J. Hesser, *Bildgebende Systeme 1 +2 (Vorlesungsskript), Kapitel 9* , Universität Mannheim, 2003.

[Hop1994] H. Hoppe. *Surface Reconstruction from Unorganized Points.* PhD thesis, Universität Washington, Seattle WA, 1994.

[Hop1996] H. Hoppe: *Progressive meshes.* ACM SIGGRAPH 96 Conference Proceedings, Seite 99–108, 1996.

[Hra2003] J. Hrádek. *Methods of surface reconstruction from scattered data.* Technical Report No. DCSE/TR-2003-02. University of West Bohemia in Pilsen 2003

[IBS97] F. Isselhard, G. Brunnett, and T. Schreiber. *Polyhedral reconstruction of 3D objects by tetrahedra removal.* Technical report, Fachbereich Informatik, Universität Kaiserslautern, Februar1997. Interner Report Nr. 288/97. aus [Men2001] und [Jem2003]

[Jem2004] A. Jeměljanov, *Surface Reconstruction from Points,* Doktorarbeit, Universität West Bohemia in Pilsen, 19. Mai 2004

[Kur2003] W. Kurth. *Vorlesungsskript Algorithmische Geometrie*, TU Cottbus 2003

[LC2005] W.Lorensen, H.Cline. *Marching Cubes.* Vortrag an der Universität Britsh Columbia, 2005

[Men2001] M. Mencl. *Reconstruction of Surfaces from Unorganized Three–Dimensional Point Clouds.* Doktorarbeit, Universität Dortmund, 2001

[Nag2004] W. Nagel. *Matchen und Mergen von 3D Punktwolken.* Seminararbeit Universität Karlsruhe, 2004

[Näc2002] A. Nächter. *Autonome Exploration und Modellierung von 3D-Umgebungen.* Diplomarbeit an der Universität Bonn 2002.

[RSI2004] Berschreibung des Scanners DigiScan2000 http://www.rsi.gmbh.de/zdfscan.htm (zuletzt besucht am: 25.10.2004)

[SB1997] T. Schreiber and G. Brunnett. *Approximating 3D objects from measured points.* Proceedings of 30th ISATA, Florenz, 1997. aus [Men2001]

[Sta2004] T. Stamm. *Kleinste Fehlerquadrate, least squares.* Zürich 2004. http://n.ethz.ch/student/stammt/Statistik.html (zuletzt besucht am: 25.10.2004)

[TF2004] H. Tangelder and A. Fabri, *CGAL-Manual, Kapitel 61: Spatial Searching, 6.4.1 Software Design / kd-tree* http://www.cgal.org/Manual/doc_html/cgal_manual/Spatial_searching/Chapter_main.html (zuletzt besucht:12.12.2004)

[Tor2004] C. Tornau, *grundstudium.info* http://www.grundstudium.info/animation/node17.php (zuletzt besucht am: 27.3.2005)

[Vel1992] R. Veltkamp.*The gamma-neighborhood graph.* Computational Geometry, Theory and Applications 1(4), April 1992, Seite 227-246.

[Vel1995] R.C. Veltkamp. *Boundaries through scattered points of unknown density.* Graphics Models and Image Processing, 57(6), Seite 441–452, November 1995.

[WPW1986]G.Wyvill, C. McPheeters, and B. Wyvill. *Data structures for soft objects.* The Visual Computer, 2(4). Seite 227–234, August 1986. (aus [Hop94])

[WS1999] R.S. Wright Jr., M. Sweet. *OpenGL Super Bible Second Edition.* Waite Group Press, Indianapolis, 1999. *ISBN 1-57169-164-2.*

[WW2003] P. Watzl, P. Wild. *Progessiv Meshes.* Vortrag 15. Juni 2003, Universität Salzburg.

Abbildungsverzeichnis

Abbildung 1: chaotische Verdrehungen der einzelnen Stengelaufnahmen in einem Bereich des Stengels 14
Abbildung 2: Einteilung von Flächenrückführungs-Algorithmen 15
Abbildung 3: Schematischer Ablauf von Raumteilungs-Algorithmen 16
Abbildung 4: Generieren der Kontur nach Hoppe 17
Abbildung 5: Tetraederunterteilung. Ein Tetraeder (links) und der unterteilte Tetraeder (rechts). 18
Abbildung 6: dreiseitiger Patch (links) und vierseitiger Patch (rechts) 20
Abbildung 7: Zusammenhang Alpha-Complex (links), Delaunay-Triangulierung (mitte) und Alpha-Shape (rechts) 21
Abbildung 8: Variation der Alpha-Werte vom 0-shape (links) bis zum ∞ -shape (rechts) 21
Abbildung 9: Konstruktion eines 2D-Normalized-Mesh(rechts) aus einer Kurve mit Näherungspunkten (links) und deren Voronoi-Zellen (Mitte). 22
Abbildung 10: Morphologische Operation auf der Oberfläche 23
Abbildung 11: Die beiden Pole eines Punktes s im 2D-Beispiel 23
Abbildung 12: Beispiel des invertierten Doppelkegels (rechts) und ein vereinfachtes 2D-Beispiel (links). Der Pfeil entspricht hierbei der Normalen eines Punktes 26
Abbildung 13: Beispiel für ein mit diesem Ansatz nicht entfernbares Polyeder 28
Abbildung 14: Modellieren der Löcher nach Attene und Spanguolo 29
Abbildung 15: 2D Beispiel für den γ -Indikator. links: positiv rechts: negativ 31
Abbildung 16: Prinzip der Distanzermittlung nach Hoppe 34
Abbildung 17: Entfernen der Störpunkte nach Roth und Winbowoo 35
Abbildung 18: Unterscheidunge zwischen sichtbaren (weiss) und nicht sichtbaren Punkten (schwarz). Die jeweilige Blickrichtung ist als Pfeil dargestellt. 36
Abbildung 19: Beispiel für ein zweidimensionals Voronoi-Diagramm mit eingezeichnetem Powerkreis. [Kur2003] 39
Abbildung 20: Die Verwandschaft zu Wachstumsalgorithmen am Beispiel von Boissonat [Boi1984] 41
Abbildung 21: Auswahlkriterium für das entstehende Dreieck 42
Abbildung 22: Edge Collapse - Prinzip [Hop1996] 46
Abbildung 23: schematischer Ablauf des Algorithmus 51
Abbildung 24: schematische 2D-Darstellung des Winkels α 53
Abbildung 25: einfach vermaschte Einzelaufnahmen, die ICP- Charakteristiken aufweisen. Hierbei sind große Abweichungen zwischen den Messungen erkennbar. 55
Abbildung 26: links der Zustand vor der Zusammenführung, rechts der Zustand nach der Zusammenführung. 56
Abbildung 27: Darstellung der Zelleinteilung zur Nachbarschaftsbestimmung. Die obere Grenze ist als Grenzwert zu interpretieren 57
Abbildung 28: Darstellung von Raumelementen in einer 26er-Nachbarschaft 58
Abbildung 29: (links) Verhalten der Gausssschen Krümmung. Rot entspricht hierbei starker Krümmung, grün geringerer Krümmung 64
Abbildung 30: (rechts) Die k-Achse ist die Krümmung. u- und v-Achse entsprechen den beschriebenen Winkeln in der Gleichung für die Gausssche Krümmung 64
Abbildung 31: Einfache Rückgeführte Grundkörper. Ebene (links) und Ellipsoid (rechts) 66
Abbildung 32: Auffächern der Scans, welches durch die Dämpfungswirkung des Algorithmus unterdrückt werden kann 70
Abbildung 33: gedämpfte Rückführung der Auffächerung aus der obigen Abbildung 70
Abbildung 34: Unsprüngliches Verfahren unter Benutzung eines jeden Nachbarn. 73

Abbildung 35: Durch Verwendung des jeweils nächsten Nachbarn verändert sich die Torsion zu größeren Flächen. 74
Abbildung 36: Wireframe eines mit $\alpha = 5°$ *reduzierten Ellipsoids* 76
Abbildung 37: Ellipsoid mit Flatshading und $\alpha = 5°$ 77
Abbildung 38: Original Punktwolke einer Ebene als Screenshot aus der Beispielanwendung (oben). Reduzierte Ebene (unten). Die Fläche wurde mit Flatshading gerendert und die Polygone eingezeichnet. 78
Abbildung 39: Reduzierte Kugel. Sie wurde mit Flatshading gerendert und die Polygone eingezeichnet. 80
Abbildung 40: : Reduziertes Ellipsoid. Es wurde mit Flatshading gerendert und die Polygone eingezeichnet. 81
Abbildung 41: Orginalausschnitt aus dem Stengel einer Gerste (mit einfacher Triangulation überalle Punkte dargestellt) 82
Abbildung 42: Reduziert mit verschiedenen α *- Winkeln in den zx- und yx-Richtungen (5° und 45°)* 82
Abbildung 43: reduzierter Stengelausschnitt mit $\alpha = 10°$ 83
Abbildung 44: Blatt mit $\alpha = 10°$ *und als Oberfläche eines Ellipsoids interpretiert* 84
Abbildung 45: zwei verschiedene α *-Werte (5° und 45°) und Interpretation als Ebene* 85
Abbildung 46: gesamter Stengel mit $\alpha = 10°$ 86
Abbildung 47: Original des in Abbildung 46 gezeigten Ergebnisses. 87
Abbildung 48: Originalpunktwolke der Gerstenpflanze 88
Abbildung 49: Rekonstruierte Gerenpflanze. 89
Abbildung 50: Chaoswerte im Bereich des Stengels 90
Abbildung 51: vorgelagerte Ebene (rot) von dem restlichen Stengel (grau) getrennt 92
Abbildung 52: nicht geschlossenes Ellipsoid 93
Abbildung 53: reale Ebene mit Loch. Die Ebene wurde über Variation von ς *als Ellipsoid interpretiert* 94
Abbildung 54: Indexerstellung des Marching-Cubes-Algorithmus 98
Abbildung 55: Mögliche Kantenkombinationen des Marching-Cubes Algorithmus 98
Abbildung 56: Konstruktion der medialen Achse 99
Abbildung 57: Oberfläche des Beispielprogramms 103