Diplomarbeit im Studienfach Informatik

Animationsorientierte Optimierung von Polygonnetzen

Oliver Bunsen
Matr.-Nr. 189943

Betreuer:

Prof. Dr. Georg Fleischmann
Fächergruppe Kunst- und Medienwissenschaften
Kunsthochschule für Medien Köln

Prof. Dr. Walter Oberschelp
Lehrstuhl für angewandte Mathematik, insbes. Informatik
RWTH Aachen

29. September 1996

Ich versichere hiermit, daß ich die vorliegende Arbeit selbstständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Köln, 29. September 1996

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Kurzvorstellung

Die vorliegende Arbeit befaßt sich mit der rechnergestützten Erstellung von Oberflächenpolygonnetzen für die Animation virtueller Darsteller. Aus der Problemanalyse und einem strukturierten Überblick klassischer Methoden der Polygonnetzerzeugung wird der Entwurf einer animationsorientierten Triangulierungssoftware abgeleitet. Die resultierende Software basiert auf der Advancing Front Technique. In Versuchsreihen wird geklärt, ob das zur Implementation gebrachte Konzept die zuvor formulierten Anforderungen der Animation erfüllen kann.

Motiviert wurde diese Arbeit durch die derzeitig verfügbare, unbefriedigende Methode der Netzerstellung. Der Arbeitsschritt der Netzgenerierung ist sehr zeitaufwendig und stellt ein großes Hindernis für die Serienherstellung neuer virtueller Darsteller dar.

1.2 Projektgruppe für virtuelle Darsteller

Diese Diplomarbeit entstand im Rahmen der Projektgruppe für virtuelle Darsteller an der Kunsthochschule für Medien in Köln. Zu den Inhalten der Projektarbeit zählt die Entwicklung neuer Methoden, die virtuellen Darstellern zu größerer Autonomie in Mimik, Bewegung und Sprache verhelfen. Darüber hinaus wird nach effizienten Verfahren zur Erstellung und Produktion von virtuellen Darstellern gesucht. Das Konzept beinhaltet die Verifikation unter Produktionsbedingungen.

1.3 Virtuelle Darsteller

Mit der heute verfügbaren Leistungsfähigkeit von Grafik-Workstations ist es möglich, mit wirtschaftlich vertretbarem Aufwand komplexe, virtuelle Welten in Echtzeit zu generieren und darzustellen. Ein Bereich der 3D-Computergrafik ist die Erstellung und Animation von virtuellen Darstellern.

Virtuelle Darsteller liegen als mathematisches Modell vor und werden mit Hilfe mathematischer Operationen animiert. Die rechnergesteuerte Animation erlaubt, komplexe Bewegungsabläufe zu realisieren. Die Steuerungsmodule für das Abrufen der Bewegungen

und die Sprachausgabe können in Abhängigkeit von Umweltsignalen (Sprache, Bilder) angelegt werden. Auf diese Weise läßt sich die Autonomie der Figur bis zur Interaktivität mit ihrer Umwelt steigern.

Die Hauptunterschiede zu traditionellen Verfahren der Kreation und Animation künstlicher Figuren (Zeichentrickanimation, Puppentrickanimation, Puppenspiel, usw.) sind:

- Virtuelle Darsteller können autonom agieren. Bei klassischen Verfahren der Figurenanimation wird jede Bewegung in jeder Phase vom Animator, bzw. Puppenspieler unmittelbar veranlaßt.
- Überwindung physikalischer Grenzen der Puppentrickanimation.
- Beliebige Reproduzierbarkeit, auch unter veränderter Kameraposition und bei variierenden Beleuchtungsverhältnissen.
- Wirtschaftliche Produktion.

Die Forschungsarbeit im Bereich virtueller Darsteller beschäftigt sich mit der computergrafischen Darstellung der Figur, mit der Bewegungsmodellierung und mit Fragen "intelligenter" Autonomie. Für den Produktionsbetrieb werden darüber hinaus effiziente Verfahren zur Erstellung und Produktion von Darstellern gesucht.

Abbildung 1.1: Minenspiel eines virtuellen Darstellers

In den Film- und Fernsehmedien sind virtuelle Darsteller (Abb. 1.1) in rein virtuellen Szenen einsetzbar oder können in reale Szenen eingefügt werden. In Verbindung mit Spracherkennung und Sprachausgabe eignen sie sich zur Konstruktion ergonomischer Mensch-Maschine-Schnittstellen. Hier eröffnet sich für Systeme des computerunterstützten Lernens ein breites Anwendungsgebiet. Darüber hinaus bietet sich ihr Einsatz in der Wissenschaft an. Beispiele sind ergonomische Analysen oder psychologische Studien.

1.4 Ziele der Arbeit

Ziel der Arbeit ist die Formulierung eines Konzepts für die rechnerunterstützte Animationsnetzerstellung. Eingabeparameter sind die 3D-Laserscandaten und die Randbedingungen der Animation in einer möglichst benutzerfreundlichen Beschreibungsform. Die Ausgabe besteht aus dem Animationsnetz, dessen Struktur für die in der Projektgruppe verwendeten Animationsverfahren optimiert sein soll.
Die Arbeit konzentriert sich auf die Animationsnetzerstellung für das Gesicht des Darstellers. Im Gesicht stellt die Animation die differenziertesten Anforderungen an das Netz.

Die bekannten Arbeiten zur Erstellung von Polygonnetzen aus 3D-Laserscandaten konzentrieren sich auf den Aspekt der Datenreduktion. Die Verfahren stellen sicher, daß die Form der Objekte bei deutlich reduzierter Datenpunktzahl in definierten Grenzen erhalten bleibt [DZ91], [HDD+93], [HDD+94], [HDDM92], [SZL92], [Tur92]. Einige Autoren beschreiben Verfahren zum variablen Zugriff auf Polygonnetze verschiedener Auflösung (*multiresolution*) zur effizienten Darstellung eines Objekts, abhängig vom Betrachtungsabstand [EDD+95], [GGS95]. Alle Verfahren eignen sich nur für statische Objekte.

Ein animationsorienter Ansatz, bei dem generische Netze auf die gescannte Oberfläche abgebildet werden, findet sich in [LTW95]. Das generische Netz ist abgestimmt auf menschliche Gesichtsproportionen. Der generische Ansatz ist nicht flexibel genug, um spezielle Gesichtsformen und Bewegungen von Cartoon-Figuren zu berücksichtigen.

In der Arbeit werden daher die Voraussetzungen für eine eigenständige Konzeption zur rechnerunterstützten Erstellung von Animationsnetzen geschaffen. Ergebnis der Problemanalyse ist der Katalog der Anforderungen an ein Animationsnetz. Die Konzeption der Software basiert auf dem Anforderungskatalog. Die Implementation der Software wird vor diesem Hintergrund getestet und bewertet.

Der theoretische Teil der Arbeit führt in die Theorie der Triangulierung ein und enthält eine strukturierte Übersicht verschiedener Netzgenerierungsverfahren aus der Literatur der Computergrafik und Finite Elemente Methoden *(FEM)* (siehe S. 24). Die Übersicht stellt einige typische Algorithmen vor und ermöglicht die fundierte Auswahl einer Triangulierungsmethode als Bestandteil der Software zur rechnergestützten Animationsnetzerstellung.

1.5 Überblick

Der Aufbau der Arbeit entspricht der Reihenfolge der in der Zielsetzung genannten Arbeitspunkte. Kapitel 2 “Polygonnetze in der Figurenanimation” führt zuerst in den Kontext der virtuellen Darsteller ein, indem es die Arbeitsschritte zur Erstellung eines Charakters beschreibt und die Möglichkeiten zur Modellierung von Bewegungen vorstellt. Anschließend werden die Skulpturierungs- und Modellierungsmethoden vertieft, die in der Projektgruppe zum Einsatz kommen. Damit ist die Grundlage für den entscheidenden Abschnitt in Kapitel 2 geschaffen, in dem die Anforderungen an ein animationsoptimiertes Dreiecksnetz diskutiert und formuliert werden.

Kapitel 3 “Grundlagen der Polygonnetzgenerierung” führt in die fundamentalen Begriffe Delaunay-Triangulierung und Constrained-Delaunay-Triangulierung in zwei Dimensionen ein.

Das inhaltlich darauf aufbauende Kapitel 4 “Algorithmen zur Netzgenerierung” strebt eine strukturierte Übersicht der Ansätze und Methoden der Netzgenerierung in der Ebene und auf gekrümmten Oberflächen an. Im Rahmen der Übersich werden einige typische Algorithmen ausführlicher vorgestellt. Insbesondere werden die Algorithmen vertieft, die in Kapitel 5 Bestandteil der Konzeption zur rechnerunterstützten animationsorientierten Triangulierung sind.

Kapitel 5 entwirft eine Triangulierungsoftware für Animationsnetze. Das Konzept organisiert die Partitionierung der Objektoberfläche in Regionen, die einzeln trianguliert werden und anschließend zum Gesamtnetz zusammengesetzt werden.

In Kapitel 6 “Versuche und Ergebnisse” werden unterschiedliche Konfigurationen der Komponenten zur Segmentinitialisierung und Triangulierung mit variierenden Parametern gewählt und auf authentisches Datenmaterial angewendet. Die Ergebnisse werden im Bild vorgestellt und die Eignung der Netze für die Animation bewertet.

Das Kapitel 7 bewertet die Eignung des Konzepts und macht Ergänzungsvorschläge für die dauerhafte Implementierung.

Kapitel 2

Polygonnetze in der Figurenanimation

In diesem Kapitel wird die Problemstellung der Arbeit motiviert. Im Überblick werden zuerst die Arbeitsschritte zur Erstellung einer virtuellen Figur geschildert. Der Abschnitt "Bewegung durch Oberflächendeformation" stellt anschließend die Bewegungsmodellierung mit Hilfe der Transformation der Oberflächengeometrie vor. Der Abschnitt "Erstellung eines virtuellen Darstellers" beschreibt die Arbeitsschritte zur Realisierung einer neuen Figur, wie sie in der Projektgruppe erfolgen. Dabei wird ersichtlich, wie zeitaufwendig das derzeitige Verfahren der Netzerstellung ist. Aus "Anforderungen der Animation an das Dreiecksnetz" ergibt sich das Entwicklungsziel der Software zur animationsorientierten Dreiecksnetzerstellung.

2.1 Arbeitsschritte bei der Produktion eines virtuellen Darstellers

Die komplette Produktion eines virtuellen Darstellers bis zur fertigen Video- oder Filmszene verlangt im allgemeinen sechs Arbeitsphasen [Hag96]. Die ersten drei Phasen behandeln den Entwurf und die Realisierung der Figur. In den Phasen vier bis sechs wird das Rechnermodell des Darstellers im Produktionsbetrieb eingesetzt.

1. Während der *Entwurfsphase* werden das Aussehen, die Mimik, die Gestik und die Sprache der Figur entwickelt. Im Vergleich zur herkömmlichen Figurenanimation ergeben sich bei der Computeranimation sowohl besondere Möglichkeiten (siehe S. 2), als auch bestimmte Einschränkungen. Zu den Einschränkungen zählt beispielsweise die mangelhafte Möglichkeit, Haare computergeneriert darzustellen und zu animieren.
   Ergebnis der Entwurfsphase sind, neben Sprachproben und der schriftlichen Beschreibung des Charakters, eine Sammlung von Zeichnungen, die die Figur aus möglichst vielen Richtungen, in vielen verschiedenen Variationen zeigen. Zusätzlich kann anhand der Zeichnungen ein plastisches Modell der Figur erstellt werden. Das Gipsmodell legt die Proportionen der Figur aus allen Richtungen verbindlich fest.

2. In der *Skulpturierung* wird das dreidimensionale Rechnermodell der Figur erarbeitet. Damit ist die Form der Figur im Rechner erfaßt.

3. In der Phase der *Modellierung* werden die Bewegungen der Figur im Rechnermodell realisiert (z.B. die Ausformung verschiedener Gesichtsausdrücke).
   Die Modellierung (S. 6) basiert auf einem spezifischen Rechnermodell. Im Rechnermodell der Figur müssen bereits in der Skulpturierungsphase die erforderlichen Strukturmerkmale für die angestrebten Bewegungen angelegt werden.

4. In der *Animation* wird der zeitliche Ablauf und die Intensität festgelegt, in der die Bewegungen abgerufen werden. Man spricht von *Performance Animation*, falls die Figur in Echtzeit animiert wird. Hierzu existiert eine Vielzahl von Steuerungstechniken. Eine Methode, die vielen Bewegungsparameter zur natürlich wirkenden Animation einer vielgliederigen Figur in Echtzeit bereitzustellen, sind *Motion Tracking Systeme*. Mit ihrer Hilfe kann die Körperstellung des Animators in Echtzeit auf die virtuelle Figur übertragen werden. Zur Steuerung der Gesichtsanimation bieten sich *Face Tracking Systeme* an, die den Gesichtsausdruck eines Schauspielers auf die Mimik der Figur abbilden. Die zur Zeit geeignetste Mensch-Maschine-Schnittstelle zur Erzeugung differenzierter multidimensionaler Steuerdaten basiert auf der menschlichen Hand: *Datenhandschuhe* erlauben es, feinste Veränderungen in der Winkelstellung der Fingerglieder zu registrieren. Durch die hohe Lernfähigkeit des Menschen in der Bewegung seiner Hand ist es praktikabel, mit der Bewegung eines Fingers elementare Bewegungen in der Mimik der Figur abzurufen. Wie ein Puppenspieler lernt der Animator, der Figur ein ausgereiftes Minenspiel zu verleihen.

   Das Rechnermodell der Figur und die modellierten Bewegungsmuster bilden zusammen mit den Kontrollstrukturen der Animation ein hierarchisches System. Der Abstraktionsgrad des Systems kann bis zu regelbasierten Kontrollstrukturen für die Interaktion der Figur in einer virtuellen Gesellschaft mit anderen Figuren oder mit der realen Umwelt reichen.

5. Das *Rendering* ist der rechenintensivste Schritt, der aus den 3D-Daten ein zweidimensionales Bild der Szene errechnet. Aus den Randbedingungen Kameraposition, Kamerawinkel, Beleuchtung und den Oberflächeneigenschaften der Objekte werden die Bilder der Filmsequenz errechnet. Die Rechenzeit beim Rendering ist abhängig von der Komplexität der Objekte und der Größe der Bildmatrix. Für die Echtzeitanimation müssen diese Parameter sorgfältig aufeinander abgestimmt sein.

6. In der *Nachbearbeitung* werden die Animationssequenzen geschnitten und bei Bedarf über ein Maskenverfahren mit dem Hintergrund gemischt (Abbildung 2.1). Auf diese Weise kann man reale Filmszenen mit computeranimierten Figuren versehen.

2.2 Bewegung durch Oberflächendeformation

Im Modellierungsschritt werden alle grundlegenden Bewegungsmuster und damit die Mimik und Gestik der Figur angelegt. Jede Bewegung führt zu einer bestimmten Deformation der Oberfläche im Raum. In der Animation werden die zeitliche Abfolge, die Überlagerung

Abbildung 2.1: Virtueller Darsteller in realer Filmszene
Mit Hilfe eines Maskenverfahrens wird die computergenerierte Filmsequenz in ein reales Videobild eingesetzt.

und die Aktivierungsstärke der Bewegungen festgelegt.

Die Oberflächenrepräsentation in der Computeranimation erfolgt polygonal oder parametrisch. Beide Repräsentationsformen interpolieren zwischen einer Menge von Stützpunkten. Bei der polygonalen Repräsentation besteht die Oberfläche aus einem Polygonnetz. Die parametrische Repräsentation stellt komplexe Oberflächen mit Hilfe rechteckiger, bikubischer Patches dar (siehe S. 43). Parametrische Oberflächen können an den Übergängen zwischen zwei Patches stetig definiert werden. Polygonmodelle besitzen flache Oberflächenelemente und sind an den Kanten im allgemeinen nicht stetig. Trotzdem werden Polygonmodelle in der Praxis bevorzugt eingesetzt. Mit hardwareunterstützten Bibliotheken werden die Transformation, das Shading (siehe S. 2.4), die Texturabbildung und das Rendering sehr effizient realisiert [GB89], [JNW94]. Nach dem Gouraud-Shading erscheint auch eine polygonale Oberfläche glatt.

Die Oberflächendeformation wird bei beiden Repräsentationen durch Veränderung der Raumkoordinaten der Stützpunkte erreicht. Die Dichte der Stützpunkte begrenzt die maximal mögliche Deformation. Außerdem müssen bei der Veränderung der Stützpunktkoordinaten die Nachbarschaftsverhältnisse der Stützpunkte respektiert werden, damit die Oberflächeninterpolation gültig bleibt. Mit dem folgenden Verfahren werden elementare Bewegungen zu komplexen Bewegungsmustern kombiniert:

Transformation der Oberflächengeometrie. Bei dieser Methode stehen Werkzeuge zur Formulierung von Koordinatentransformationen einzelner Stützpunkte oder Gruppen von Stützpunkten zur Verfügung (zur Koordinatentransformation im homogenen Koordinatensystem siehe [GB89]). Jede Punkttransformation basiert auf der Ausgangsposition und ist über einen Parameter bis zu seiner Maximalposition aktivierbar. Bereits mit der kombinierten Rotation und Translation sind in der Praxis ausreichend differenzierte Bewegungsbahnen definierbar. Kompliziertere Bewegungsbahnen können mit Splines definiert werden [WW92].
Dieses Modell bietet besonders viel Flexibilität für die Gesichtsanimation. Bewegungen

des Gesichts bestehen größtenteils aus Muskelbewegungen, die die Oberfläche verformen, indem sie bestimmte Hautbereiche relativ zueinander stauchen oder strecken. Mit dem Transformationsansatz kann direkt das Ergebnis des Muskelspiels, die Hautverformung, subtil nachgebildet werden.

Zur effizienten Definition von Bewegungen werden benachbarte Knotenpunkte eines Hautbereiches zu Clustern zusammengefaßt. Die Punkte eines Clusters werden mit einer Bewertungsfunktion belegt. Die Bewertungsfunktion nimmt zu den Randpunkten des Hautbereichs hin linear oder exponentiell ab (siehe Abbildung 2.2). Sie reguliert die Intensität, mit der ein Punkt des Clusters von der Transformation erfaßt wird. Auf diese Weise kann die Elastizität der Haut modelliert werden.

Die parametrisierten elementaren Bewegungen werden zu Ausdrücken zusammengefaßt. In der Animation reguliert ein gemeinsamer Parameter den Aktivierungsgrad der kombinierten Bewegung.

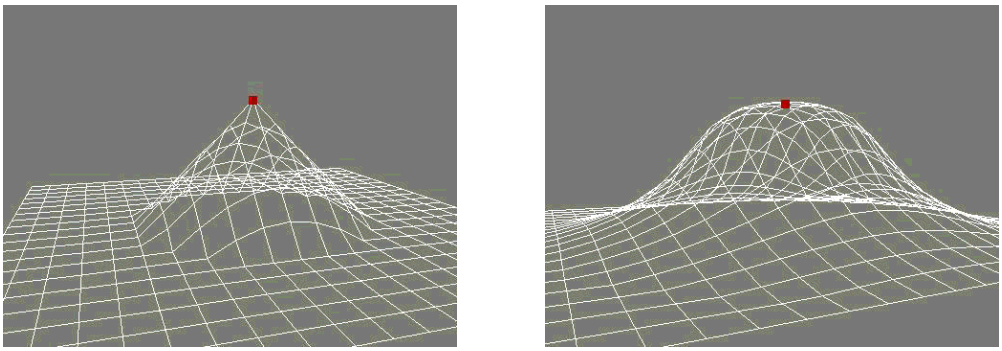


Abbildung 2.2: Bewegungsmodellierung durch Transformation von Oberflächenpunkten
Transformation einer Oberfläche, die von einem Viereckснetz approximiert wird. Die Gitterpunkte im zu transformierenden Areal sind links mit einer linearen und rechts mit einer exponentiellen Bewertungsfunktion belegt. Die Bewertungsfunktion reguliert den Grad der Translation eines Punktes (aus [Hag96]).

2.3 Erstellung eines virtuellen Darstellers

In diesem Abschnitt wird die Umsetzung der beiden Arbeitsschritte beschrieben, wie sie derzeit in der Projektgruppe vollzogen wird. Die Modellierung erfolgt als Transformation der Oberfläche, folglich wird bei der Skulpturierung ein Oberflächenpolygonmodell erstellt.

2.3.1 Skulpturierung

Der Weg von der Zeichnung zum Rechnermodell (Abb. 2.3) führt über das plastische Gipsmodell der Figur, das im letzten Schritt der Entwurfsphase hergestellt wurde.

Abbildung 2.3: Entwicklung: Zeichnung, plastisches Modell, Rechnermodell

Die Datengrundlage für die Skulpturierung ist das Ergebnis eines dreidimensionalen Scanvorgangs. Der Rotationsscanner umfährt das Gipsmodell der Figur in 512 Winkelinkrementen (diskrete Koordinate ϕ) und tastet mit einem Laserstrahl für jedes Winkelinkrement vertikal 512 Datenpunkte (diskrete Koordinate h) ab. An jedem Datenpunkt werden die Tiefeninformation, d.h. die Entfernung vom Scandetektor zum Objekt (Fließkomma-Koordinate r) und ein RGB-Farbwert ermittelt (Abb. 2.4).
Die Farbinformation wird in einer gesonderten Bilddatei gespeichert und kann später mit Hilfe von Funktionsaufrufen der 3D-Grafikbibliothek auf das Polygonmodell abgebildet werden *(texture mapping)*. Im Fall des Gipsmodells wird die gescannte Textur allerdings nicht weiterverwendet. Stattdessen wird eine eigens erstellte Texturdatei mit natürlich wirkender Haut- und Augenfarbe und Bartschatten auf das Polygonmodell abgebildet.

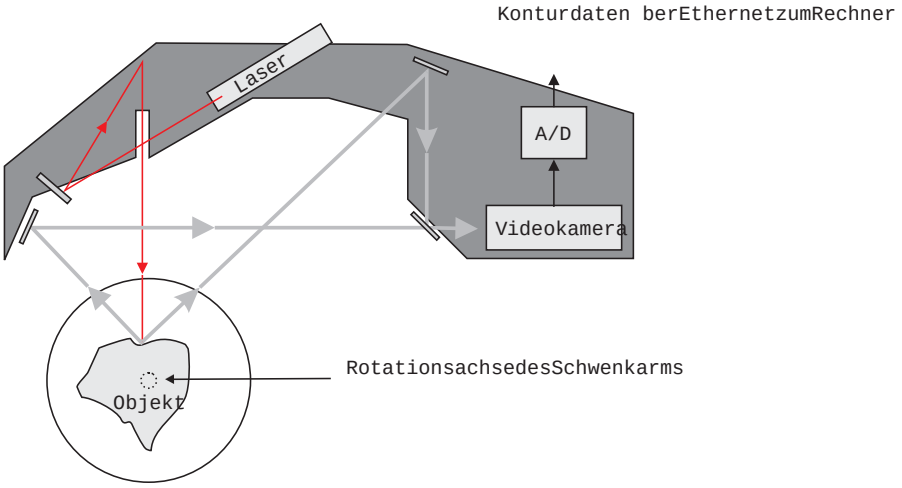


Abbildung 2.4: Schematische Darstellung des 3D-Laserscanners (aus [Hag96])

Abbildung 2.5 visualisiert das im Scanvorgang gewonnene Datenmaterial in Gouraud-Shading-Darstellung. Die gewonnene Rauminformation liegt entlang vertikaler Linien vor. Haare führen beim Scan zu Meßfehlern. Desweiteren ist es nicht möglich, konkave Geometrien zu scannen, bei denen Überschneidungen in Laserstrahlrichtung vorliegen (z.B. Ohr). Areale, die sich tangential zur Strahlrichtung befinden, führen zu Meßfehlern oder Meßwertausfällen (Unterseite Kinn, Schädelkalotte). Daher ist in der Regel die Nachbear-

beitung der Rohdaten mit Routinen aus der Scanner-Software notwendig.

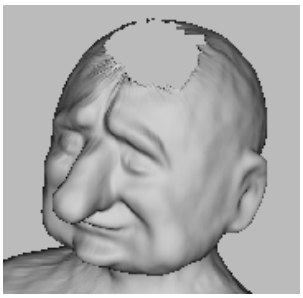


Abbildung 2.5: Gouraud-Shading der Scandaten
Auf der Schädelkalotte besitzt der Laserstrahl eine nahezu tangentiale Richtung zur Oberfläche. Meßausfälle sind die Folge.

Ziel der Skulpturierungsphase ist die Erstellung eines Oberflächenpolygonmodells für die Bewegungsmodellierung mittels Transformation der Netzknotenpunkte. Abbildung 2.6 zeigt den Katalog der in der eingesetzten 3D-Grafikbibliothek "OpenGL" [JNW94] verfügbaren graphischen Datenstrukturen. Zur Modellierung von Flächenstücken existieren außer den elementaren Polygonobjekten (`GL_TRIANGLES, GL_QUADS, GL_POLYGON`) im Speicherbedarf effizientere Flächenelemente in 3D. Die Beispielabbildung eines `GL_TRIANGLE_FAN` in Abb. 2.6 modelliert ein Flächenstück bestehend aus drei Dreiecken auf der Basis von fünf Punkten (`V0,...,V4`). Mit der elementaren Datenstruktur `GL_TRIANGLES` sind neun Punkte für das gleiche Flächenmodell notwendig. Bei den komplexeren Datenstrukturen ergeben sich bei der Speicherung und Berechnung der transformierten Flächen Effizienzvorteile.

In der Figurenanimation kommen die `STRIP-` und `FAN-`Datenstrukturen in OpenGL, trotz der geringen Effizienzvorteile kaum zum Einsatz. Die Polygonnetze für Freiformoberflächen sind in der Regel nicht homogen genug, um mit geringem Aufwand `STRIPS` oder `FANS` aufgeteilt zu werden.

Derzeitige Software zur Netzerstellung. Die Software ermöglicht dem Benutzer die manuelle Erstellung des Polygonnetzes. Eingabe der Software ist die Datei mit der 3D-Scaninformation der Gipsfigur. Die Benutzerschnittstelle umfaßt zwei Fenster (Abb. 2.8) zur Darstellung der Daten. Im Editierfenster blickt der Benutzer auf die 2D-Zylindermantelprojektion der Texturdaten. Sie ermöglichen die Orientierung auf der Objektoberfläche. In diesem Editierfenster kann der Benutzer durch Mausklick Netzknotenpunkte setzen. Intern werden die 2D-Koordinaten der gesetzten Punkte auf 3D-Koordinaten der gescannten Oberfläche abgebildet. Der Benutzer gibt im Editierfenster vor, wie die Netzknotenpunkte durch Kanten zu Dreiecken verbunden werden. Während der Benutzer schrittweise im Editierfenster das Netz erstellt, erlaubt ihm ein zweites Fenster sein bisher erstelltes Netz in der 3D-Ansicht zu überprüfen. In der 3D-Ansicht läßt sich ein Punktmodell der räumlichen Scandaten mit aufprojiziertem Polygonnetz aus jeder beliebigen Raumposition (daher

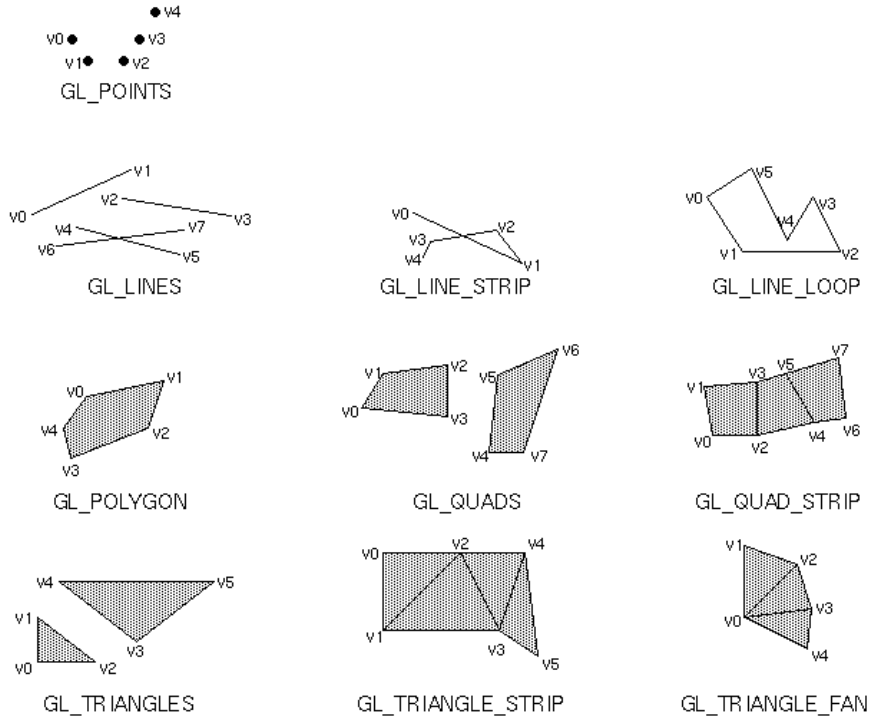


Abbildung 2.6: OpenGL - geometrische Primitive
Auflistung der geometrischen Datenstrukturen in OpenGL (aus [JNW94]).

"3D-Ansicht") auf dem Monitor betrachten. Wird in der 3D-Ansicht ein Fehler im Netz offensichtlich, dann kann der Benutzer im Editierfenster die Knotenpunkte verschieben oder Elemente löschen. Zu Abschluß wird das erstellte Polygonnetz *(mesh)* im sogenannten Mesh-Dateiformat gesichert.

Alle Elemente des Dreiecksnetzes werden manuell vom Benutzer erstellt. Die Qualität des Netzes hängt allein von seiner Geschicklichkeit und Erfahrung ab. Die Netzerstellung ist ein langwieriger Prozeß. Ein erfahrener Animator benötigt viele Stunden, um zu einem ersten Netz zu kommen, das zyklisch verfeinert wird. Die Kriterien für ein optimales Netz im Sinne der Animation werden im nächsten Abschnitt aufgeführt.

2.3.2 Modellierung

Die Basisausdrücke der Figur werden mit Hilfe einer gesonderten Software erstellt. Bewegungen werden beim Dreiecksnetz-Oberflächenmodell durch Translation und Rotation ausgesuchter Punkte realisiert. In der Benutzerschnittstelle der Software werden für jeden an der Gesamtbewegung beteiligten Punkt Translation und Rotation eingegeben. Die Translation wird definiert durch Angabe der Endposition des angewählten Punktes. Für die Rotation wird zusätzlich eine Rotationsachse angegeben. Durch Überlagerung von Translation und Rotation können sehr differenzierte, elliptische Bewegunglinien formuliert werden. Ein Basisausdruck ist komplett beschrieben, wenn für alle an der Bewegung beteiligten Punkte die Bewegungsparamter der Transformation vorliegen. Die Bewegung

erfolgt durch stufenlose Skalierung der Punkttransformation.

In der Animation können die Basisausdrücke gewichtet gebündelt werden. In der Animation werden diese komplexen Bewegungsmakros über Kurven gesteuert, die die Skalierung zu einem Zeitpunkt angeben. Die Skalierungskurven werden, in einem entsprechenden Zeit-Skalierungsfaktor-Diagramm dargestellt, in dem auch die Amplitude des Sprachsignals als Bezugspunkt der Animation angezeigt wird. Manipuliert werden die Kurven entweder durch mausgesteuertes Editieren im Zeitdiagramm oder durch Signale von Eingabegeräten wie z.B. Motion Tracking, Datenhandschuh oder durch automatische Analyse des Sprachsignals.

2.4 Anforderungen der Animation an das Dreiecksnetz

Dieser Abschnitt stellt den Anforderungskatalog der Animation durch Oberflächentransformation an das zugehörige Dreiecksnetzoberflächenmodell der Figur auf. Die ausgeführten Punkte umfassen die Kriterien, nach denen die Animationsnetze durch manuelles Editieren erstellt werden. Die gleichen Kriterien gelten für eine Software zur automatischen Generierung von Animationsnetzen.

Form der Netzelemente. Bei den Netzelementen handelt es sich um Dreiecke nichtuniformer Größe. Dreiecke passen sich flexibler als andere Polygone an vorgegebene Geometrien an und erlauben bei gleicher Netzknotenanzahl die feinste Oberflächenbeschreibung.

Eine günstige Netzstruktur besteht aus möglichst gleichschenkligen Dreiecken und geringen Sprüngen in der Größe benachbarter Dreiecke (Homogenität). Zwischen Adaptivität und Homogenität muß optimiert werden. Gleichschenklige Dreiecke sind der allgemeinste Ausgangspunkt für Transformationen. Ein ohnehin spitzes Dreieck gestattet kaum noch Transformationen, die es noch spitzer machen. Stark inhomogene Netzstrukturen können zu fehlerhaftem Shading führen. Beim Gouraud-Shading wird über den Dreiecken ein linearer Helligkeitsverlauf interpoliert. Die Interpolation basiert auf Normalenvektoren zu den drei Eckpunkten des Dreiecks. Jede Normale errechnet sich durch Mittelung aller angrenzenden Flächennormalen. Die Gewichtung der einzelnen Flächennormalen basiert in der Implementation nicht auf der Größe des anliegenden Winkels. Die Winkelbestimmung für alle Dreiecke ist zu rechenintensiv. Stattdessen werden die Flächennormalen abhängig von der Dreiecksfläche gewichtet. Starke Sprünge in der Dreiecksgröße verfälschen die Berechnung der Flächennormalen und führen zu fehlerhaftem Shading (Abb. 2.7).
Spitze Dreiecke sind die Folge von starken Sprüngen in der Dreiecksgröße. Eine günstige Netzstruktur vermeidet spitze Dreiecke.

Für die Approximation einer kontinuierlichen Oberfläche durch ein Dreiecksnetz existiert ein mathematischer Zusammenhang zwischen der Dreiecksform und dem Approximationsfehler. EPPSTEIN [BE92] weist auf das Theorem von RIPPA hin, nach dem für jede Funktion $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ und jede Menge Stützstellen die *Elevated-Delaunay-Triangulierung* der Stützstellen die optimale Approximation von f im energetischen Sinne darstellt. Ungünstige Netzstrukturen beinhalten spitze Dreiecke. Die Delaunay-Triangulierung vermeidet spitze Dreiecke. Das Theorem von RIPPA bestätigt also, daß günstige

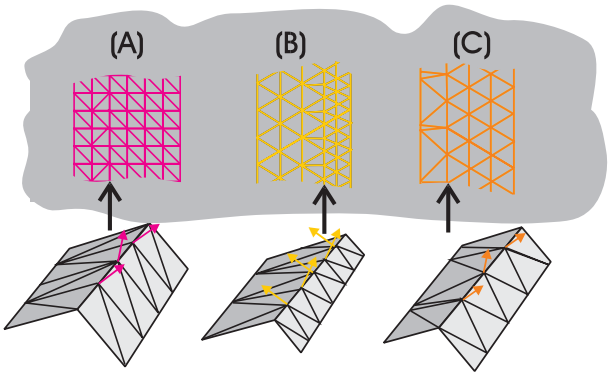


Abbildung 2.7: Fehlerhafte Normalenapproximation bei ungünstigen Netzstrukturen (aus [Hag96])

Netzstrukturen eine gute Approximation der Oberfläche bewirken.

Oberflächenapproximation mit lokaler Fehlertoleranz. Das Polygonnetz approximiert die fein aufgelöste Oberflächeninformation der Scandaten mit einer maximalen Zahl von Netzknotenpunkten (z.B. um 1000). Trotz der erheblichen Datenreduktion soll das Polygonmodell das Objekt möglichst gut approximieren. Gleichmäßiges Verteilen von Netzknotenpunkten auf der Oberfläche mit anschließender Triangulierung würde bei der geringen Knotenpunktzahl zum Verlust wichtiger Details führen. Die Knotenpunktdichte wird daher detailabhängig gewählt und bestimmt die Kantenlänge und Anzahl der Dreiecke. Bei der manuellen Netzerstellung plaziert der Animator die Punkte ahängig von seiner Kenntnis der Oberflächenkrümmung.

Die Knotenpunktdichte erfährt zusätzlich eine Korrektur um einen konstanten Wert, der von der betreffenden Region abhängt. Die Objektoberfläche umfaßt mehrere *regions-of-interest*, mit unterschiedlicher Bedeutung für die Skulpturierung der Figur. Beispielsweise weist der Hinterkopf eine mittlere Dreiecksgröße auf, die geringer ist, als es seine mittlere Oberflächenkrümmung vorgibt. Der Hinterkopf wird während der Animation kaum sichtbar sein und kann deshalb sparsamer approximiert werden. Die charakteristischen Partien im Gesicht werden hingegen über ihren krümmungsabhängigen Punktdichtewert hinaus ausgebildet. Die Bedeutung einer Region hängt von der Wahrnehmungpsychologie des Betrachters (Augen, Mund und Nase werden besonders wahrgenommen) und vom Abstand der virtuellen Kamera ab.

Höhere Dreiecksdichte in stark animierten Regionen. Die krümmungsabhängige Dichtefunktion wird um einen zusätzlichen, animationsabhängigen Korrekturwert positiv verschoben. Mit den bisherigen Anforderungen wird lediglich eine gute Approximation der Oberfläche des statischen Gipsmodells sichergestellt. Während der Animation sollen besonders am Mund und den Augen sehr feine Oberflächendetails ausgebildet werden. Im Dreiecksnetz muß die Fähigkeit dazu zuvor angelegt sein.

Für die bedeutungsbasierte und animationsorientierte Dichteerhöhung kann die gleiche Partitionierung der Oberfläche in Regionen verwendet werden. Die beiden Korrekturwerte

können daher zu einem durch den Animator definierten Wert zusammengefaßt werden.

Das Konzept einer regionenabhängigen Approximationstoleranz formalisiert das intuitive Vorgehen des Animators bei der Wahl der Knotenpunktdichte in der manuellen Netzerstellung.

Öffnungen im Netz und Verbindungen zu weiteren Netzen. Das Netz muß Öffnungen für Augen und Mund aufweisen. Außerdem ist zu beachten, daß eine Ganzkörperfigur aus mehreren Körperteilen besteht, und ein komplettes Gipsmodell nicht in einem Scanvorgang digitalisiert werden kann. Das Modell ist entweder zu detailliert für einen Scan oder nicht ohne Verdeckungen herstellbar (z.B. Arme verdecken Oberkörper). In diesem Fall werden die Gliedmaßen einzeln modelliert und gescannt. Auch die Animationsnetze werden zuerst getrennt erstellt, um dann zur Gesamtfigur zusammengesetzt zu werden. Dazu ist es notwendig, die Schnittkanten sorgfältig aufeinander abzustimmen.

Kontrollierte Faltung während der Animation. An dieser Stelle wird das Konzept der *Animationslinien* eingeführt. Diese gedachten Linien ergeben sich in der Bewegungsmodellierung als Faltungslinien und Flußrichtungslinien auf der Oberfläche. Sie müssen bereits bei der Skulpturierung im Dreiecksnetz angelegt werden. Jede Animationslinie wird im Polygonnetz durch eine genügend große Anzahl Netzknotenpunkte und ihre Verbindungskanten als Kantenzug approximiert.
Animationslinien verlaufen orthogonal zur typischen Bewegungsrichtung des sie umgebenden Bereichs. Zur kontrollierten Faltung der Hautoberfläche während der Animation finden sich Animationslinien beispielsweise in der Umgebung der Mundpartie im Übergang zur Wange. Abb. 2.9 illustriert die vier Bewegungsphasen beim Grinsen. Anhand der Abbildungen ist sichtbar, wie sich die Oberfläche entlang der blau eingetragenen Animationslinien faltet.

Die Animationslinien lassen sich nicht auf Grundlage der statischen Information des 3D-Scans gewinnnen. Sie stellen Meta-Wissen des Animators dar und ergeben sich erst in der Phase der Bewegungsmodellierung. Daher müßte die Bewegungsmodellierung einen rückgekoppelten Einfluß auf die Skulpturierung besitzen. Der Animator nimmt mit seinem Wissen über die angestrebten Bewegungen der Figur diese Rückkoppelung bei der Skulpturierung vorweg.

Flußrichtung der Dreiecke. Die zweite Eigenschaft, die durch Animationslinien beschrieben wird, ist eine bestimmte Konfiguration der Dreiecke im Netz, die sich als Flußrichtung betiteln läßt. Um eine Animationslinie herum ordnen sich die Dreicke schichtweise *(successive layers)* an. Diese Anordnung fördert die Ausbildung homogenerer Netzstrukturen in intensiv animierten Bereichen, in denen stärkere Veränderungen der Oberflächenkrümmung stattfinden. Homogene Netzstrukturen vermeiden Schattierungsfehler. Weiterhin ergibt die Layer-Anordnung, die stark an Höhenlinien erinnert, eine feine Approximation der Oberfläche, wenn sich während der Animation ein stärkere Krümmung in Bewegungsrichtung ausbildet.

Die Einhaltung der Flußrichtung stellt sicher, daß das Dreiecksnetz bei Bewegungen die Elastizität der Hautoberfläche ausreichend modelliert. Ungleichmäßiges Shading,

verursacht durch ungünstige Netzstrukturen, entspricht nicht dem Oberflächenverhalten natürlicher Haut unter Stauchung und Streckung. Natürlicher Haut wird bei Verformung gedämpft und über die Umgebung geglättet ausgelenkt.

Die schichtweise Konfiguration der Dreiecke auf dem Augenlid in den Abbildungen 2.10 ergibt sich aus den umgebenden Animationslinien. Die Flußrichtung verläuft in der Nachbarschaft der Animationslinien orthogonal. In der Bewegungssequenz in der Abbildung staucht sich das Augenlid gleichmäßig, indem die Layer gleichmäßig gestaucht werden.

Abbildung 2.11 zeigt das manuell erstellte Animationsnetz in einem größeren Ausschnitt. Besonders auf der Wange kann man anhand der Flußrichtung der Dreiecke die typischen Bewegungsrichtungen der Animation ablesen.

Vergegenwärtigt man sich den Vorgang des Textur-Mappings auf das in der Animation deformierte Dreiecksnetz als komplexes *Image Warping* [Wol90], dann kommt den Animationslinien eine weitere Aufgabe zu. Der durch sie induzierte Dreieckskantenzug im Netz wirkt als Kontrollinie für die Bildtransformation. Beispielsweise befindet sich die zentrale Linie der Augenbrauen in Abb. 2.10 auf einer Animationslinie. Würde die Augenbraue durch die Fläche eines großen Dreiecks laufen, so könnten die Bildverzerrungen des Textur-Mapping innerhalb dieses Dreiecks die Augenbraue aus ihrer Linie auslenken.

Umsetzung des kaum formalisierbaren Expertenwissens. Die Erstellung eines Animationsnetzes unter gleichzeitiger Beachtung der bisher aufgeführten Gesichtspunkte stellt ein kompliziertes Optimierungsproblem dar. Zusätzlich zur formalen Unschärfe bestimmter Kriterien ("Flußrichtung"), kann das Gewicht eines Kriteriums lokal variieren. Bei der manuellen Erstellung des Netzes kann der Animator auf Grund seiner Erfahrung beispielsweise entscheiden, ob für ein bestimmtes Dreieck das Kriterium der Gleichseitigkeit besonders vernachlässigt werden kann, wenn sich dadurch die übrige Netzstruktur besser ausbilden läßt.
Diese Vorgänge lassen sich kaum formalisieren und basieren auf der handwerklichen Intuition des Animators.

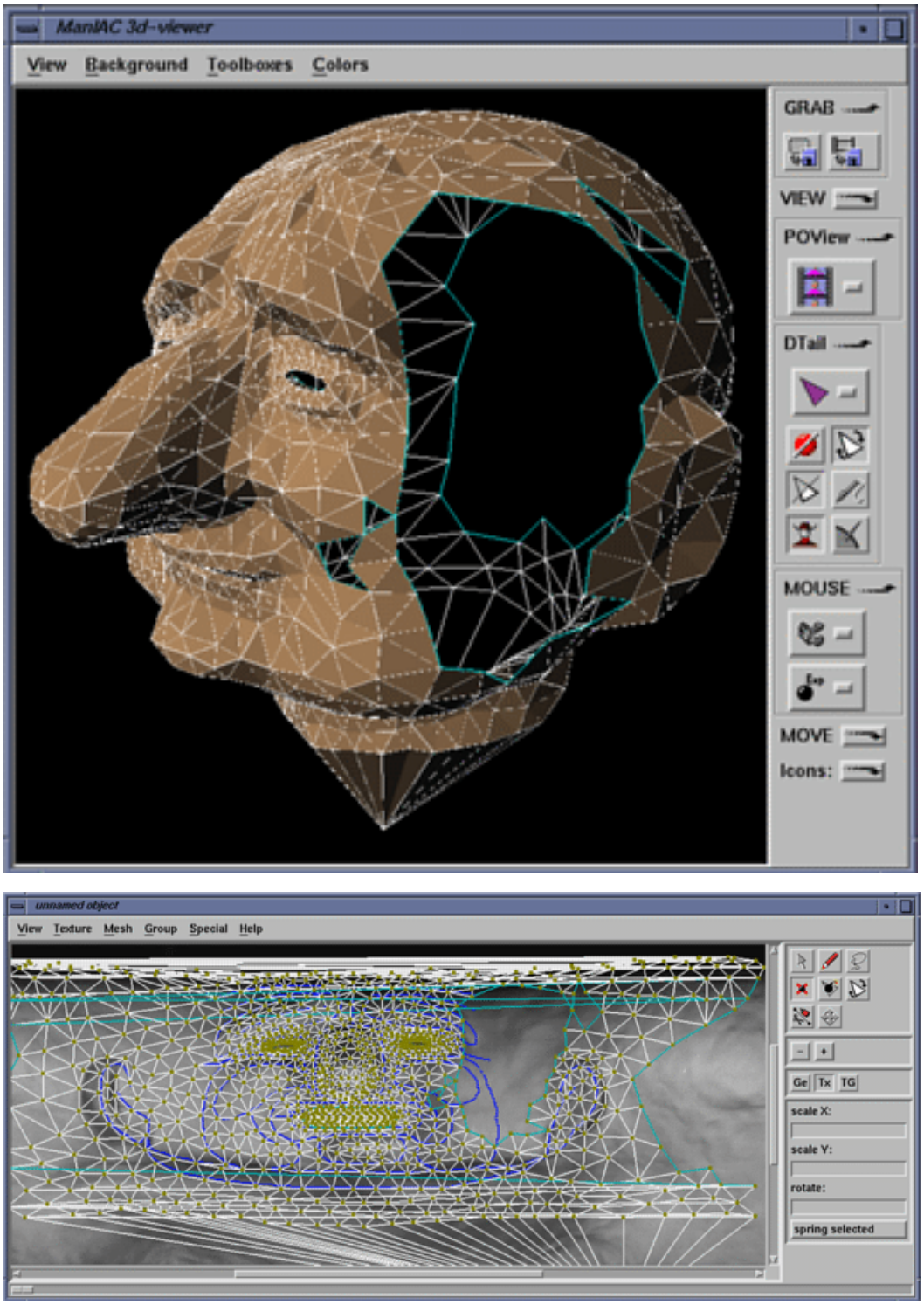
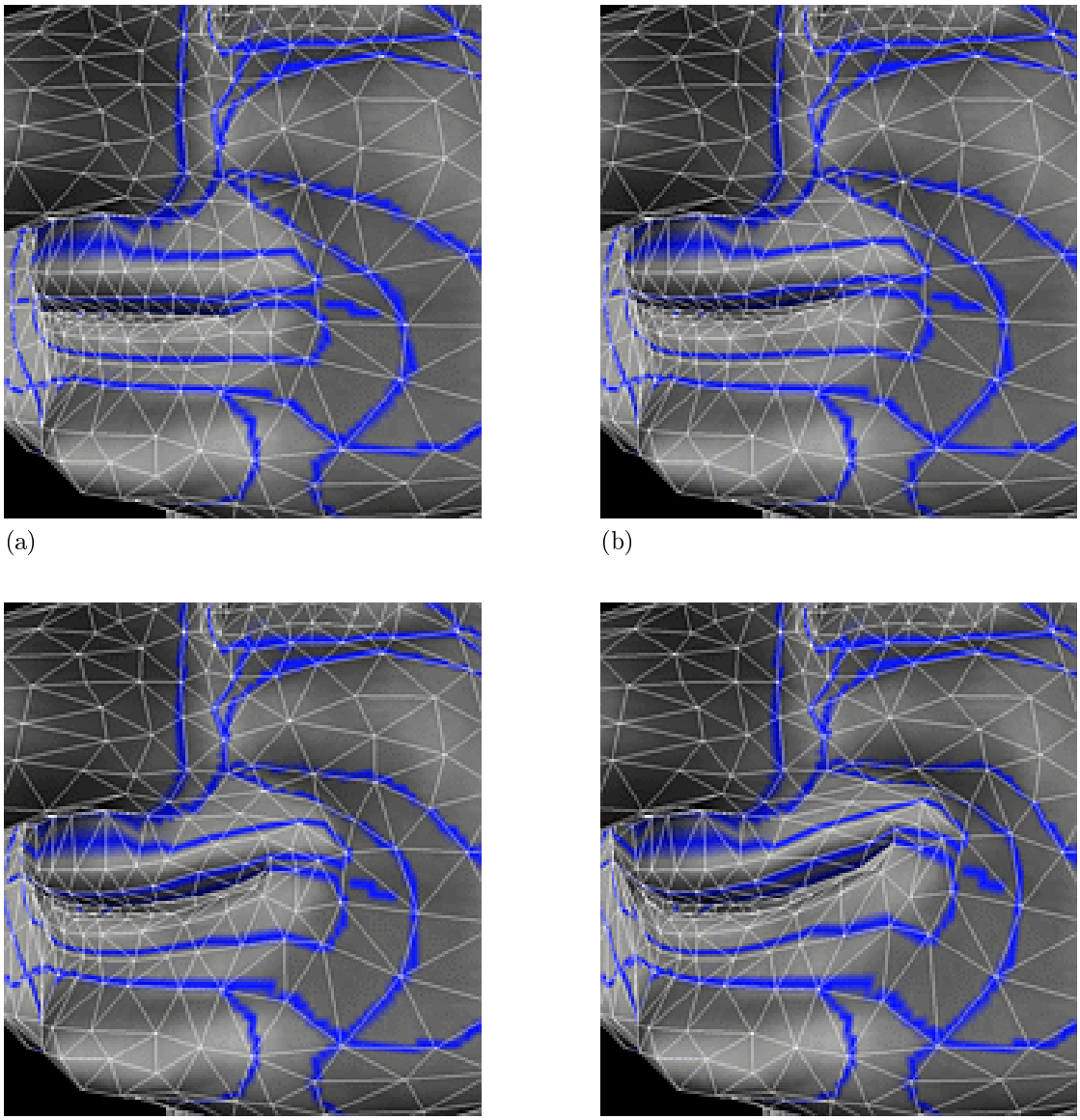


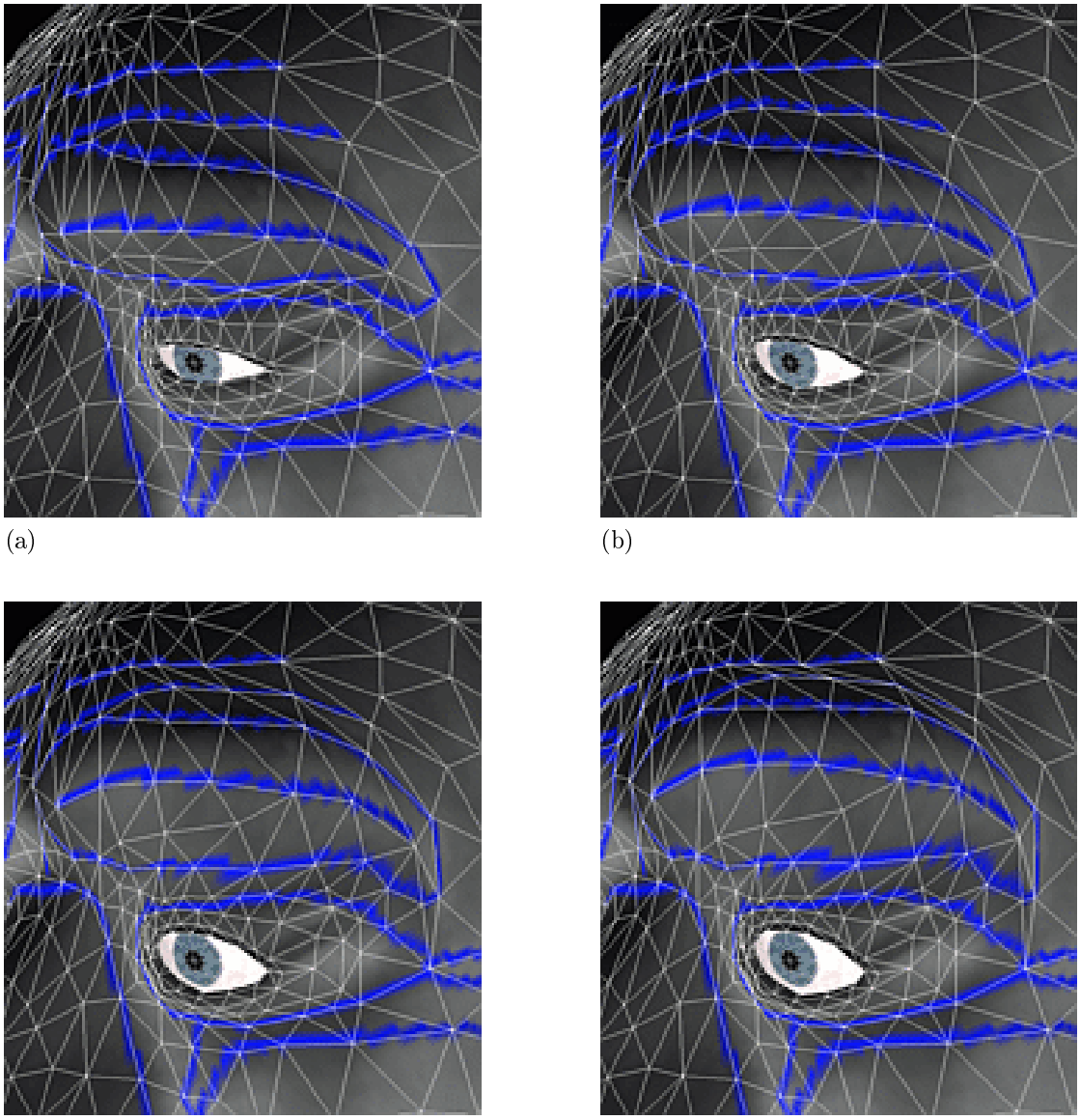
Abbildung 2.8: Existierende Editiersoftware zur Netzerstellung
Zwei Fenster aus der Software zur interaktiven Erstellung des "mesh". Oben die 3D-Ansicht des aktuellen Polygonnetzoberflächenmodells in Polygon-Shading-Darstellung. Unten das 2D-Editierwerkzeug, mit dem Punkte auf dem per 3D-Scanner erfaßten Objekt gesetzt werden und zu Dreiecken verbunden werden können. Für ein vollständiges Netz muß das Loch auf der linken Seite geschlossen werden.



(a) (b)

(c) (d)

Abbildung 2.9: Animation des Grinsens
Beim Grinsen (Sequenz (a) bis (d)) faltet sich das gestauchte Oberflächennetz entlang bestimmter Animationslinien.



(a) (b)

(c) (d)

Abbildung 2.10: Animation der Augenlider
Zur Animation der Augenlider und Augenbrauen wird das Netz gestreckt und gestaucht (Sequenz (a) bis (d)).

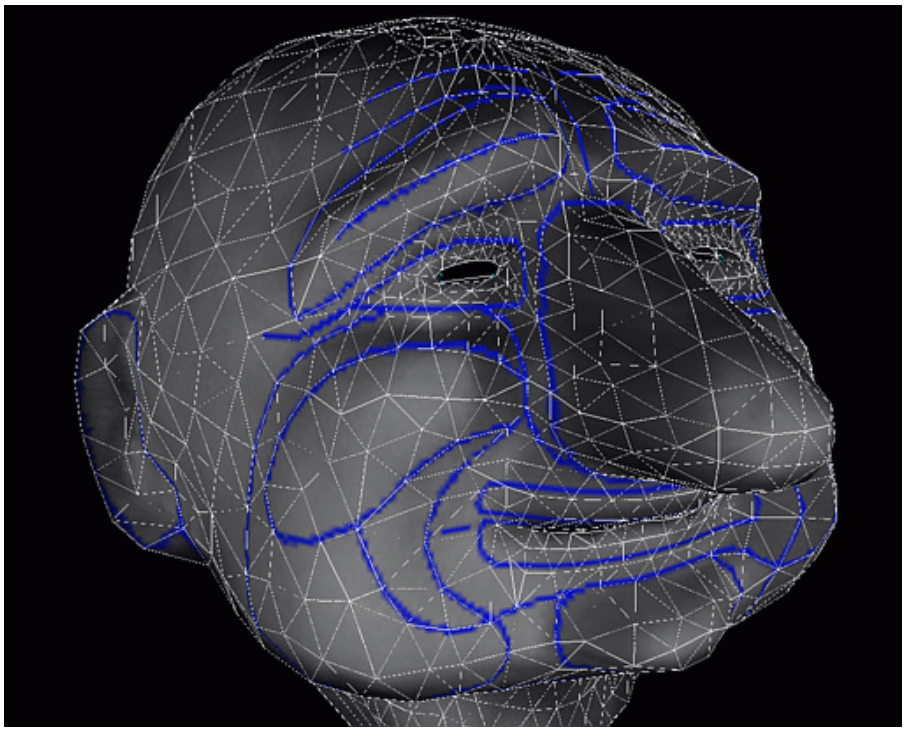


Abbildung 2.11: Animationsnetz mit Animationslinien
Auf die 3D-Dreiecksnetzoberfläche wurde eine Textur mit den Animationslinien abgebildet.

Kapitel 3

Grundlagen der Polygonnetzgenerierung

Wie in Kapitel 2 geschildert, kommen in der Computergrafik bevorzugt Dreiecksnetze zum Einsatz. Die Punkte der Dreiecke bilden Samples der Oberflächenfunktion dieser Körper, die durch die Dreiecksflächen stückweise linear approximiert wird. Alternativ kann die Oberflächenfunktion statt auf Dreiecken basierend auch mit Hilfe von Polygonen mit mehr als drei Kanten approximiert werden. Das Hauptaugenmerk dieses Kapitels wird auf Dreiecksnetzen liegen, da diese nicht nur in der Charakteranimation am häufigsten anzutreffen sind, sondern sich auch durch Verschmelzung von Dreiecken einfach in Netze aus Vielecken überführen lassen.

Die fundamentalen Gesichtspunkte für die Betrachtung von Algorithmen zur Polygonnetzerzeugung lauten:

1. Wie werden die Punkte auf der Domain gewählt?
2. Wie werden zwischen den Punkten Kanten gesetzt, um ein Polygonnetz zu erzeugen?

Die beiden Fragestellungen gelten gleichermaßen für Netze auf Oberflächen und Netze in der Ebene und bilden die Grundlage der im nächsten Kapitel vorgenommenen Klassifikation von Algorithmen zur Netzgenerierung.

In diesem Kapitel werden die Grundlagen für die Triangulierung gegebener Punktmengen in der Ebene dargelegt. Dreiecksbasierte Verfahren eignen sich als Basis für Polygone mit mehr als drei Seiten, und zweidimensionale Verfahren bilden die Grundlage für Oberflächenapproximationen in 3D. Zur Überführung von 2D Verfahren in die mit 2.5 Dimensionen bezeichneten Körperoberflächen, existieren die Projektion und andere Transformationen zwischen beiden Domains.

Ergebnis dieses Kapitels werden die Begriffe der Delaunay-Triangulierung (3.6) und Constrained-Delaunay-Triangulierung (3.7) sein. Als einziges lokales Kriterium (3.2) zur Triangulierung von vier Punkten in der Ebene impliziert das lokale Max-Min-Winkelkriterium das globale Optimalitätskriterium für die Triangulierung von Punktmengen (3.3). Die Triangulierung nach dem Max-Min-Winkelkriterium wird auch als Delaunay-Triangulierung bezeichnet. Äquivalente Definitionen der Delaunay-Triangulierung sind das

Umkreiskriterium (3.6) und der Dualismus zur Voronoï-Parkettierung (3.5) [HL92].
Durch eine Modifikation des Umkreiskriteriums erhält man die Constrained-Delaunay-Triangulierung, die einen begrenzenden Kantenzug um die Punktmenge respektiert, anstatt die Punktmenge konvex zu triangulieren[Ren96],[Che89a]. Außerdem erlaubt sie die Definition zwingender innerer Kanten und mehrfach verbundener Domains.

3.1 Allgemeines über Triangulierungen

Die Triangulierung einer Punktmenge in $\mathbb{R}^2$ ist ein formaler Begriff, dessen Definition lautet:

Definition 3.1 (Triangulierung) *Sei $P = \{P_i = (x_i, y_i) : i \in \{1, ..., n\}\}, n \in \mathbb{N}$ eine Menge von n Punkten $P_i \in \mathbb{R}^2$ und Ω die konvexe Hülle von P, dann bildet eine Menge $\mathcal{T} = \{(a_j, b_j, c_j) : a_j, b_j, c_j \in \{1, ..., n\}\}$ bestehend aus m Punktindextripeln (a_j, b_j, c_j) eine Triangulierung $\mathcal{T}$ genau dann, wenn folgende Bedingungen gelten:*

1. *Die Punkte $P_{a_j}, P_{b_j}, P_{c_j}$ bilden für jedes $j \in \{1, ..., m\}$ die Ecken eines nichtentarteten Dreiecks T_j.*
2. *Jedes Dreieck wird exakt durch drei Punkte aus P definiert, die Dreiecke enthalten sonst keine Punkte aus P.*
3. *Der Durchschnitt des Inneren von zwei Dreiecken T_k, T_l $k, l \in \{1, ..., m\}$ ist leer.*
4. *Die Vereinigung aller Dreiecke ergibt die konvexe Hülle Ω.*

Graphentheoretisch stellt die Triangulierung einen planaren, zusammenhängenden Graph dar, der die konvexe Hülle der Punktmenge vollständig parkettiert. Das gilt auch für Oberflächentriangulierungen in 3D, wenn die Oberfläche nicht gefaltet ist und die Punktmenge in eine Ebene projiziert wird.

Bereits für vier Punkte kann ihre Triangulierung nicht eindeutig sein. Zur Beurteilung der Güte einer Triangulierung wurden daher zahlreiche Kriterien aufgestellt.

3.2 Kriterien zur Triangulierung von vier Punkten

Man betrachte nun als lokalen Fall die Triangulierung einer Menge von vier Punkten, die ein streng konvexes Quadrilateral bilden, das heißt das Quadrilateral ist identisch mit der konvexen Hülle seiner vier Eckpunkte. Außerdem seien keine drei Punkte kollinear, dann ist die Triangulierung nicht eindeutig. Es gibt zwei Möglichkeiten (Abb. 3.1).

Die Bedingung des streng konvexen Quadrilaterals bedeutet, daß sich alle Punkte ausschließlich auf dem Rand der konvexen Hülle befinden. Befindet sich einer der Punkte innerhalb der konvexen Hülle, dann gibt es nur eine mögliche Triangulierung, bei der drei Dreiecke entstehen (Abb. 3.2).

Sind mindestens drei Punkte kollinear, dann ist ebenfalls nur eine Triangulierung möglich, bei der zwei Dreiecke entstehen 3.3.

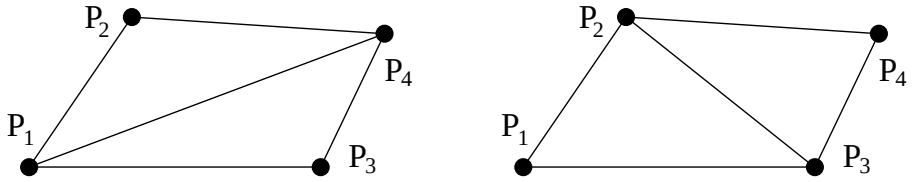


Abbildung 3.1: Triangulierung von vier Punkten
Liegen vier Punkte auf dem Rand ihrer konvexen Hülle, dann gibt es zwei Möglichkeiten, sie zu triangulieren.

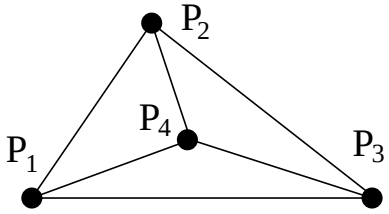


Abbildung 3.2: Triangulierung von vier Punkten mit einem Punkt innerhalb der konvexen Hülle

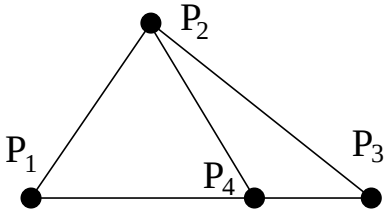


Abbildung 3.3: Triangulierung von vier Punkten, wobei drei Punkte kollinear sind

In den nichteindeutigen Fällen bieten sich eine Reihe von Entscheidungskriterien für die lokale Triangulierung von vier Punkten an. Die Kriterien sind aus der Anwendung von Dreiecksnetzen motiviert [BE92]. Beispiele sind:

1. **Definition 3.2 (Kriterium der kürzeren Diagonalen)** *Eine Triangulierung T ist eine bessere Triangulierung als T' genau dann, wenn gilt $d < d'$, wobei d die Länge der Diagonalen P_iP_k der Triangulierung T und d' die Länge der Diagonalen P_jP_l der Triangulierung T' bezeichnet [HL92].*

   Dieses Kriterium ist nicht in der Lage, die Erzeugung langer dünner Dreiecke zu vermeiden. In der Charakteranimation (siehe Abschnitt 2.4) wie auch in den FEM ist man bestrebt, kleine Winkel zu vermeiden, um "wohlgeformte" Dreiecke zu erhalten.

2. **Definition 3.3 (Max-Min-Winkelkriterium)** *Eine Triangulierung T von vier Punkten ist eine bessere Triangulierung als T' genau dann, wenn gilt $a(T) > a(T')$, mit $a(T) = \min\{a(T_j) \mid T_j \in T\}$, wobei $a(T_j)$ den kleinsten Winkel im Dreieck T_j bezeichnet [HL92].*

   Das Kriterium ist das Kriterium des größten kleinsten Winkels und vermeidet lange, spitze Dreiecke. Es liefert nur für vier Punkte in *allgemeiner Lage* eine eindeutige Entscheidung. Nicht allgemein ist die Lage von vier Punkten auf einem Umkreis. Das Kriterium wird in diesem Fall von beiden möglichen Triangulierungen erfüllt. Die Entscheidung für eine Triangulierung ist beliebig. Weiterhin gilt als nicht allgemein die Situation von vier Punkten, bei denen drei Punkte kollinear sind. Wie bereits skizziert, existiert dann ohnehin nur eine gültige Triangulierung.

3. **Definition 3.4 (Min-Max-Winkelkriterium)** *Eine Triangulierung $\mathcal{T}$ ist eine bessere Triangulierung als $\mathcal{T}'$ genau dann, wenn gilt $a(\mathcal{T}) < a(\mathcal{T}')$, mit $a(\mathcal{T}) = \max\{a(T_j) \mid T_j \in \mathcal{T}\}$, wobei $a(T_j)$ den größten Winkel im Dreieck T_j bezeichnet [HL92].*

   Finite Element Methoden approximieren, basierend auf einer endlichen Zahl von Dreiecksflächensegmenten, eine kontinuierliche Funktion und erlauben, das dynamische Verhalten von Objekten zu simulieren. Simuliert werden beispielsweise das Verformungsverhalten von Automobilkarosserien oder thermodynamische Vorgänge in Turbinen und Verbrennungsmotoren. Die Konvergenz dieser Verfahren und damit die Fehlerabschätzung hängt nicht allein von der Größe des kleinsten Winkels ab. Zu große Winkel sind ebenfalls ungünstig. BERN und EPPSTEIN [BE92] weisen auf Arbeiten hin, in denen theoretisch dargelegt wird, wie sich die Konvergenz in FEM bei Auftreten großer Dreieckswinkel verschlechtert. Statt den kleinsten Winkel zu maximieren, minimiert das Min-Max-Winkelkriterium daher den größten Winkel.

4. **Definition 3.5 (Max-Min- und Min-Max-Radiuskriterium)** *Die bei diesem Kriterium zu maximierende, bzw. minimierende Eigenschaft $a(T_j)$ bezeichnet den kleinsten, bzw. größten Radius der in die beiden angrenzenden Dreiecke einbeschriebenen Kreise.*

   Das Max-Min-Kriterium schließt diese beiden Kriterien ein.

5. **Definition 3.6 (Max-Min-Flächenkriterium)** *Maximiere bei der Triangulierung der vier Punkte den kleinsten Flächeninhalt der beiden Dreiecke.*

6. **Definition 3.7 (Max-Min-Höhenkriterium)** *Maximiere für alle Triangulierungen von vier Punkten die kleinste Höhe in beiden Dreiecken.*

7. **Definition 3.8 (Minimale Summe der Kantenlängen)** *Minimiere die Summe der Längen der fünf Kanten* (minimum weight triangulation).

   Dies ist ein einfaches Maß für die Größe der Dreiecke.

8. **Definition 3.9 (Minimales Verhältnis von Umkreis- zu Dreiecksflächen)** *Wähle diejenige Triangulierung der vier Punkte, bei der über beide Dreiecke das Verhältnis der Flächeninhalte der Umkreise um die Dreieckspunkte zu den Dreiecksflächen minimiert wird.*

Je nach geometrischer Situation der vier Punkte führen die Kriterien zu identischen Triangulierungen.

3.3 Global optimale Triangulierungen

Dieser Abschnitt führt den Begriff der globalen Optimalität einer Triangulierung ein und stellt das Max-Min-Kriterium als lokales Kriterium heraus, bei dem lokale Optimalität globale Optimalität garantiert.

Größere Punktmengen lassen sich durch sukzessive Hinzunahme jeweils eines (z.B. zu einer der Seiten nächstgelegenen) Punktes triangulieren. Mit jedem Schritt entsteht ein neues lokal optimales Dreieck.

Definition 3.10 (Lokal optimale Triangulierung) *Eine Triangulierung T heiß lokal optimal bzgl. eines Kriteriums K genau dann, wenn jedes Viereck in T, das von zwei Dreiecken gebildet wird, die sich eine Kante teilen, lokal optimal trianguliert ist bzgl. K [HL92].*

Abhängig von der Reihenfolge der gewählten Punkte können dabei verschiedene lokal optimale Triangulierungen entstehen. Beispielsweise erfüllen beide Triangulierungen in Abb. 3.4 lokal das Min-Max-Winkelkriterium. Es ist also erforderlich, den Begriff der *globalen Optimalität* einzuführen, mit dem sich auch zwei lokal optimale Triangulierungen vergleichen lassen. Hierzu werden die lokalen Maße des betrachteten Kriteriums für alle Dreiecke der Triangulierung in einem lexikographisch sortierten Vektor zusammengefaßt und die Vektoren lexikographisch verglichen.

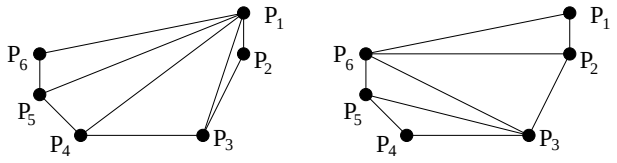


Abbildung 3.4: Zwei lokal optimale Triangulierungen
Beide Triangulierungen sind bezüglich des Min-Max-Winkelkriteriums lokal optimal (nach [HL92]).

Für jede Triangulierung T der Punktmenge P mit M Dreiecken wird ein Vektor $a(\mathsf{T}) = (a_1, ..., a_M)$ aus den lokalen Entscheidungskriterien aufgestellt, bei dem die Komponenten $a_i = a_i(\mathsf{T}_j)$ der Größe nach sortiert werden. Eine Triangulierung gilt als global optimal, falls keine von T verschiedene Triangulierung T' existiert, deren Vektor kleiner (minimierendes Kriterium) oder größer (maximierendes Kriterium) ist. $a(\mathsf{T}) \leq a(\mathsf{T}')$ im Sinne des lexikographischen Vektormasses gilt genau dann, wenn es ein $k \in \mathbb{N}$ gibt, so daß gilt $a_i = a'_i$ für $i = 1, .., k \Leftrightarrow 1$ und $a_k < a'_k$.

Definition 3.11 (Global optimale Triangulierung) *Eine Triangulierung T einer Punktmenge P heißt global optimal bzgl. eines Kriteriums K genau dann, wenn mit dem obigen Vektormaß $a(\mathsf{T})$ gilt: $a(\mathsf{T}) \geq a(\mathsf{T}')$ (bzw. $\leq$) für jede Triangulierung T' von P [HL92].*

Die global optimale Triangulierung ist, bis auf Änderungen die das Optimalitätsmaß $a(\mathsf{T})$ konstant halten, eindeutig.

Ist eine Triangulierung global optimal, dann ist sie auch lokal optimal. Umgekehrt ergibt nur das Max-Min-Winkelkriterium aus einer lokal optimalen Triangulierung zwingend eine global optimale Triangulierung. Lokal optimale Triangulierungen bezüglich anderer lokaler Entscheidungskriterien (Min-Max-Winkelkriterium, minimales Verhältnis von Umkreis- und Dreiecksflächen) sind oft weit entfernt vom globalen Optimum.
Triangulierungsalgorithmen basieren häufig auf einfachen lokalen Operationen, die das betreffende lokale Entscheidungskriterium umsetzen. Beispiele sind der *Diagonalentausch* (*edge flip*) oder das Einfügen zusätzlicher Kanten (*edge insertion*). Bei anderen Kriterien als dem Max-Min-Kriterium kann ein sukzessiver Algorithmus vor dem Erreichen des

globalen Optimums in ein lokales Optimum laufen. Zur effizienten Annäherung an das globale Optimum können diese Algorithmen um Kontrollstrukturen aus der statistischen Optimierung (z.B. *simulated annealing*) ergänzt werden [BE92].

3.4 Globale Optimierung mit dem Max-Min-Winkelkriterium

Das Max-Min-Winkelkriterium vereint also zwei Eigenschaften:

1. Es ist Kriterium für wohlgeformte Dreiecke im Sinne der Computergrafik und FEM.

2. Die Eigenschaft, auf der Basis lokaler Operationen global optimal zu triangulieren, begünstigt den Entwurf von entsprechenden Triangulierungsalgorithmen.

Beispielsweise nutzt LAWSONs einfacher nachoptimierender Algorithmus [Law72] die zweite Eigenschaft zur max-min-winkel-optimierten globalen Triangulierung aller konvexen Quadrilaterale in allgemeiner Lage folgendermaßen aus:
Ausgangsbasis ist eine beliebige Initialtriangulierung der Punktmenge. In allen nicht lokal max-min-winkel-optimalen Quadrilateralen (konvex und in allgemeiner Lage) wird die Diagonale getauscht bis alle Quadrilaterale lokal optimal trianguliert sind. Wenn die gesamte Triangulierung lokal optimal ist, dann ist die Triangulierung nach Abschnitt 3.3 auch global optimal. Durch seine Komplexität $O(n^2)$ (n Anzahl der Eingabepunkte) eignet sich der "Lawson-Flip"- bzw. "Edge-Swap"-Algorithmus allerdings nur für kleine Punktmengen.

3.5 Voronoï-Diagramm und Delaunay-Triangulierung

Die mit dem Max-Min-Winkelkriterium konstruierte Triangulierung ist äquivalent zur *Delaunay-Triangulierung*. Diese leitet sich wiederum aus dem globalen Ansatz des *Voronoï-Diagramms* ab. Das Voronoï-Diagramm zu einer Punktmenge P (*Generatorpunkte*) (Abb. 3.5) unterteilt die Ebene in *Voronoï-Regionen*. Jede Region enthält alle Punkte der Ebene, die näher am zugehörigen Generatorpunkt, als jedem anderen Generatorpunkt liegen.

Definition 3.12 (Voronoï-Region) *Die Voronoï-Region R_i zu einem Generatorpunkt P_i ist die Menge der Punkte: $R_i = \{x \in \mathbb{R}^2 \mid j, i : d(x, P_i) \leq d(x, P_j)\}$. $d(x, y)$ ist der euklidische Abstand zwischen x und y.*

Der gemeinsame Rand von zwei Voronoï-Regionen wird als *Voronoï-Kante* bezeichnet. Die Voronoï-Kanten treffen sich in den *Voronoï-Punkten*. Das Voronoï-Diagramm hat folgende Eigenschaften:

- je nach Lage der Generatorpunkte gibt es drei Arten von Voronoï-Kanten:

    1. Die gewöhnliche Voronoï-Kante ist ein Liniensegment zwischen zwei Voronoï-Punkten.
    2. Die unendliche Voronoï-Kante beginnt an einem Voronoï-Punkt und breitet sich unendlich in der Ebene aus.

3. Der dritte Voronoï-Kantentyp ist eine Gerade und tritt auf, wenn sich sämtliche Punkte der Generatorpunktmengen auf einer Linie befinden. Die Voronoï-Kanten sind parallel.

- Ein Voronoï-Punkt ist der Mittelpunkt des Umkreises der Generator-Punkte, deren Voronoï-Regionen diesen Voronoï-Punkt berühren.
- Die Voronoï-Region ist ein konvexes Polygon, das seinen Generatorpunkt enthält.
- Die die Region umrandende Polylinie ist unterbrochen, und die Voronoï-Region ist unbegrenzt, falls sich der zugehörige Generatorpunkt auf der konvexen Hülle von P befindet.
- Die Anzahl der Voronoï-Regionen und Generatorpunkte ist identisch.
- Zwei Voronoï-Regionen haben höchstens eine Kante aus ihrem Rand gemeinsam.

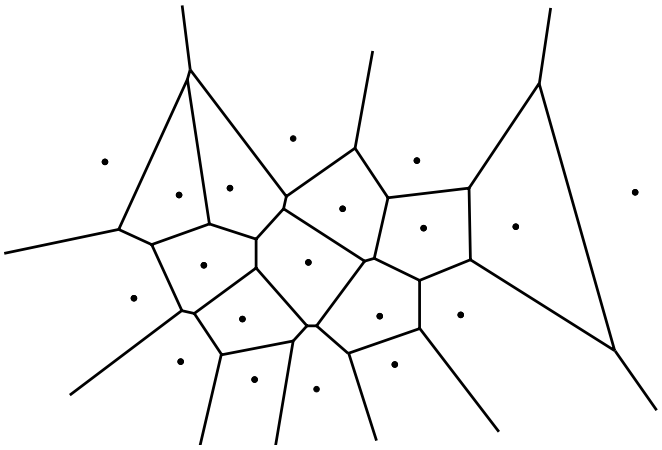


Abbildung 3.5: Voronoï-Diagramm
(aus [Far93])

Die Delaunay-Triangulierung $\mathsf{DT}(P)$ von P ergibt sich als duale Struktur zum Voronoï-Diagramm (Abb. 3.6).

Definition 3.13 (Voronoï-basierte Delaunay-Triangulierung) *Sei $\mathsf{Vor}(P)$ das zur Punktmenge P gehörige Voronoï-Diagramm und $\mathsf{Vor}(p_i)$ die Voronoï-Region zum Punkt p_i. Jedem Voronoï-Punkt, der den Mittelpunkt eines Umkreises mit drei zugehörigen Generatorpunkten bildet, wird eindeutig ein Dreieck der Delaunay-Triangulierung zugeordnet, wenn folgende Regeln beachtet werden:*

- *$\forall i \neq j : p_i$ und p_j werden zu einer Dreieckskante verbunden, wenn $\mathsf{Vor}(p_i)$ und $\mathsf{Vor}(p_j)$ eine gemeinsame Voronoï-Kante besitzen.*
- *Berühren sich mehr als drei Voronoï-Regionen in einem Voronoï-Punkt, dann gibt es mindestens vier Generatorpunkte der Punktmenge P, die auf dem gemeinsamen Umkreis liegen. Für diese Untermenge von P ist jede gültige Triangulierung gleichermaßen wählbar (vgl. Anmerkungen zum Max-Min-Winkelkriterium 3.2).*

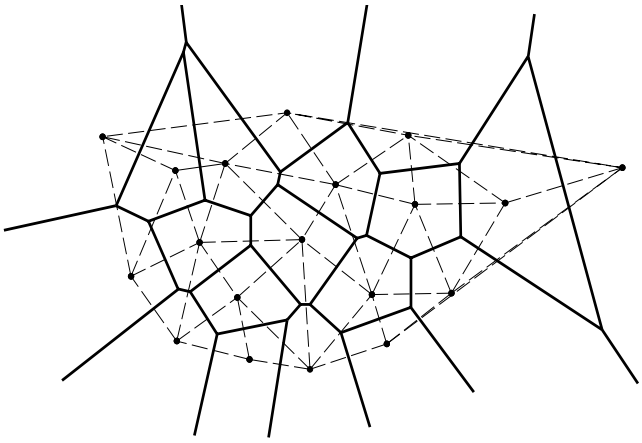


Abbildung 3.6: Delaunay-Triangulierung über Voronoï-Diagramm
Die Delaunay-Triangulierung wurde über das Voronoï-Diagramm aus Abb. 3.5 gelegt. Jeweils drei Punkte, die einen gemeinsamen Voronoï-Punkt besitzen, bilden ein Dreieck (aus [Far93]).

Besteht die Punktmenge nur aus drei kollinearen Punkten, dann sind die drei Voronoï-Kanten unendlich und parallel, und es gibt keine Voronoï-Punkte, die sich zu einem Dreieck verbinden lassen. Mit dem Vorhandensein eines vierten nicht-kollinearen Punktes entstehen zwei Voronoï-Punkte, und es läßt sich in Übereinstimmung mit den Ausführungen in Abschnitt 3.2 eine gültige Delaunay-Triangulierung, bestehend aus zwei Dreiecken konstruieren.

Umgekehrt erhält man das Voronoï-Diagramm aus der Delaunay-Triangulierung der Punktmenge P durch Errichten der Mittelsenkrechten in jedem Dreieck. Der Schnittpunkt bildet jeweils einen Voronoï-Punkt.

Der Voronoï-Ansatz zur Delaunay-Triangulierung macht verständlich, warum die DT eine ideale Datenstruktur für Probleme des nächsten Nachbarn eines Punktes in der Ebene darstellt *(closest node problems)*. Unter den Generatorpunkten, mit denen ein Generatorpunkt eine Verbindung in Form einer Delaunay-Kante eingeht, befindet sich in jedem Fall derjenige Punkt, der dem Bezugspunkt am nächsten liegt.

FARESTAM [Far93] verweist auf Algorithmen zur Errichtung des Voronoï-Diagramms zu einer Punktmenge.

3.6 Delaunay-Umkreiskriterium

Die klassische Charakterisierung der Delaunay-Triangulierung erfolgt über das lokale Umkreiskriterium [HL92], [Far93], [Ren96]. CLINE und RENKA definieren damit die Delaunay-Triangulierung wie folgt:

Definition 3.14 (Delaunay-Triangulierung) *Sei P eine Menge von Punkten in der*

Ebene, eine Menge von Dreiecken T ist eine Triangulierung von P, falls folgende Bedingungen gelten:

1. *Die Dreieckseckpunkte sind Elemente aus P.*
2. *Kein Dreieck enthält andere Punkte aus P als seine Eckpunkte.*
3. *Das Innere der Dreiecke ist paarweise disjunkt.*
4. *Die Vereinigung der Dreiecke ist die konvexe Hülle von P.*

Mit der zusätzlichen Bedingung 5. wird die Triangulierung zur Delaunay-Triangulierung:

5. *Das Innere des Umkreises jedes Dreiecks enthält keinen Punkt aus P.*

Die lokale Eigenschaft 5. (Abb. 3.7) ist äquivalent zum Max-Min-Winkelkriterium, und ein Verstoß dagegen läßt sich durch Diagonalentausch (*edge flip*) aufheben. Das lokale Umkreiskriterium zur Triangulierung von jeweils vier Punkten impliziert das globale Umkreiskriterium für größere Punktmengen, bei dem für sämtliche Dreiecke der Triangulierung gelten muß, daß ihr jeweiliger Umkreis keine weiteren Punkte aus P beinhalten darf.

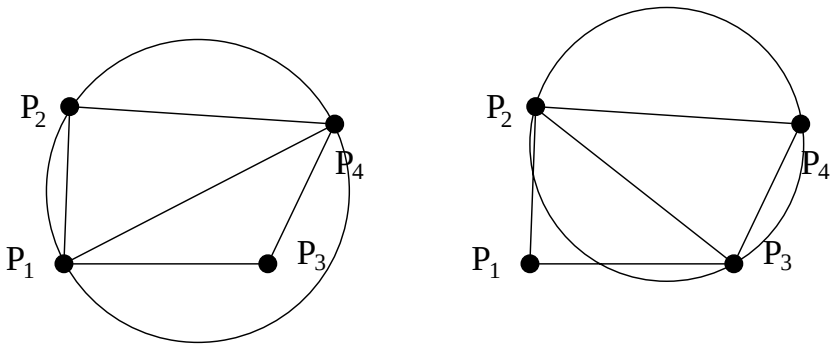


Abbildung 3.7: Umkreiskriterium der Delaunay-Triangulierung
Links wird das Umkreiskriterium verletzt. Durch Diagonalentausch (edge flip) wird rechts eine gültige Triangulierung erreicht. Es genügt, eines der beiden Dreiecke zu überprüfen. Das Umkreiskriterium verhält sich in beiden Fällen äquivalent.

Bei den im Zusammenhang mit dem Max-Min-Kriterium und der Überführung des Voronoï-Diagramms in die Delaunay-Triangulierung genannten, uneindeutigen lokalen Triangulierungen von vier oder mehr beteiligten Punkten, handelt es sich um einen Grenzfall des Umkreiskriteriums. Befinden sich mehr als drei Punkte auf einem Umkreis, dann ist jede gültige Triangulierung dieser Punkte zulässig. Bis auf diesen Fall ist die Delaunay-Triangulierung eindeutig.

3.7 Constrained-Delaunay-Triangulierung

Die Delaunay-Triangulierung definiert eine konvexe Triangulierung der Eingabepunktmenge. Bei praktischen Problemstellungen ist die zu triangulierende Domain jedoch oft durch

einen Kantenzug begrenzt oder enthält "Löcher" und ist damit topologisch mehrfach verbunden. CLINE und RENKA modifizieren daher die Punkte 4. (Vereinigung der Dreiecke ergibt konvexe Hülle der Eingabepunkte) und 5. (Umkreiskriterium) ihrer Definition der Delaunay-Triangulierung so, daß die Triangulierung sogenannte *constraints* berücksichtigt.

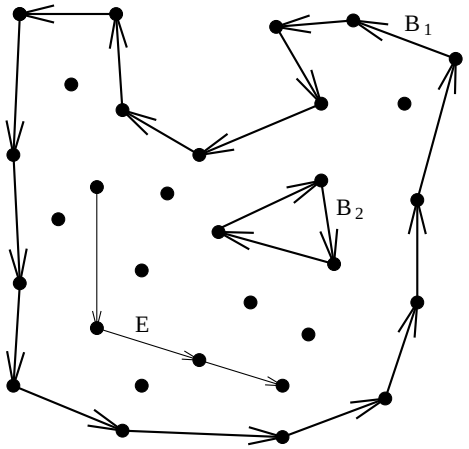


Abbildung 3.8: Punktmenge mit drei Arten von Constraints

Abb. 3.8 illustriert drei mögliche Typen von Constraints, die bei der Constrained-Delaunay-Triangulierung Berücksichtigung finden. B_1 und B_2 sind begrenzende Kantenzüge *(boundaries)*. Jede Kante in diesen Kantenzügen *(boundary edge)* verbindet zwei Punkte der Eingabepunktmenge (*boundary nodes*), aber ist sonst disjunkt zur Menge der Eingabepunkte. Die Kantenzüge sind so gerichtet, daß sich alle Eingabepunkte in der Schnittmenge aller Halbebenen links zu jeder gerichteten Kante *(interior)* befinden. Die Schnittmenge aller Halbebenen rechts zu jeder Kante *(exterior)* eines begrenzenden Kantenzuges enthält keine Eingabepunkte. Die zu triangulierende Domain Ω wird definiert:
$\Omega = closure(interior(B_1) \cap ... \cap interior(B_l))$.
Zusätzlich wird die Formulierung einer Menge zwingender Kantenzüge E innerhalb der Domain Ω gestattet. Deren Kanten verbinden paarweise Punkte aus der Eingabepunktmenge (keine boundary nodes). Außer der definierten Endpunkte enthalten diese Kanten keine weiteren Punkte der Eingabemenge und haben mit anderen Kantenzügen höchstens ihre Endpunkte gemeinsam.

Die Vereinigung von E mit der Menge der begrenzenden Kanten bildet die Menge aller zwingenden Kantenzüge R (constraints).
Die vollständige Definition der Constrained-Delaunay-Triangulierung (Abb. 3.9) lautet:

Definition 3.15 (Constrained-Delaunay-Triangulierung) *Sei P eine Menge von Punkten in der Ebene, eine Menge von Dreiecken T ist eine Triangulierung von P, falls die folgenden Bedingungen gelten:*

1. *Die Dreieckseckpunkte sind Elemente aus P.*
2. *Kein Dreieck enthält andere Punkte aus P als seine Eckpunkte.*
3. *Das Innere der Dreiecke ist paarweise disjunkt.*

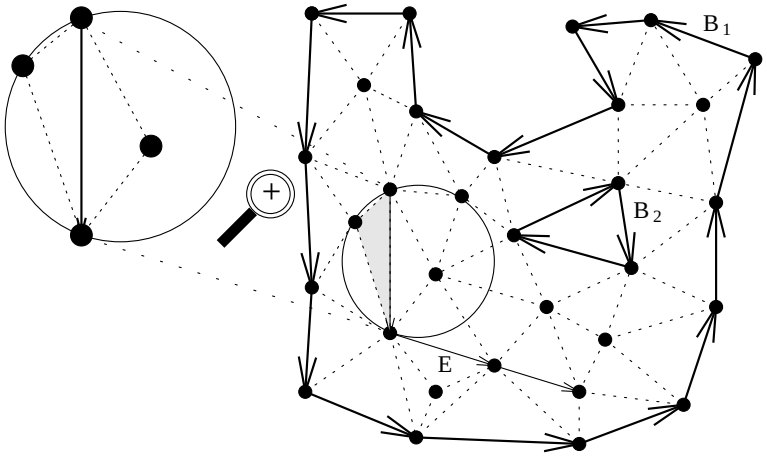


Abbildung 3.9: Constrained-Delaunay-Triangulierung der Abb. 3.8
Links oben in der Vergrößerung das modifizierte Umkreiskriterium, das Punkte im Umkreis erlaubt, sofern sie durch eine zwingende Kante vom betreffenden Dreieck separiert sind.

4. *Die Vereinigung der Dreiecke ist* Ω.

5. *Jeder Punkt, der sich im Inneren des Umkreises jedes Dreiecks befindet, ist von den Punkten des Dreiecks durch eine Kante aus* R *getrennt.*

6. *Jede Kante in* R *bildet die Kante eines Dreiecks.*

Kapitel 4

Algorithmen der Netzgenerierung

In der Literatur zur Computergrafik und FEM werden eine Vielzahl von Algorithmen zur Netzgenerierung beschrieben, die in unterschiedlichen Szenarien angewendet werden können. Eine erschöpfende Übersicht über diese Methoden würde den Rahmen dieses Kapitels sprengen. Stattdessen soll eine universelle Klassifikation vermittelt werden, die Algorithmen und Ansätze aus der Computergrafik und FEM gleichzeitig erschließt. Die skizzierten Algorithmen stellen typische Repräsentanten ihrer Klasse dar. Insbesondere finden solche Algorithmen Erwähnung, die in die Konzeption der Software zur animationsorientierten Triangulierung aufgenommen werden.

Ausführliche Klassifikationen von Algorithmen aus dem Bereich der FEM und umfangreiche Literaturverweise befinden sich in den Arbeiten von HO-LE [HL88] und EPPSTEIN [BE92].

Die Übersicht beinhaltet ausschließlich Algorithmen zur Erzeugung von Netzen in der Ebene und dreidimensionalen Oberflächennetzen. Volumenorientierte Netzgenerierungsverfahren werden ausgeklammert, obwohl die Volumenrepräsentation eines Objekts mit Polyedern ein polygonales Oberflächennetz impliziert. In der Konzeption der rechnerunterstützten Erstellung von Animationsnetzen wurden diese Algorithmen jedoch nicht berücksichtigt.

Polygonnetze werden in *strukturierte* und *unstrukturierte Netze* unterteilt. Strukturierte Netze [KS93] besitzen eine eindeutige Viereckstopologie. Jedes viereckige Netzelement besitzt, außer wenn es sich am Rand der Domain befindet, genau acht Nachbarelemente. Mit vier Nachbarn besitzt es je eine gemeinsame Kante, und mit den verbleibenden vier Nachbarn teilt es einen Knotenpunkt. Die Nachbarschaftsbeziehung ist also im Gegensatz zu Dreiecksnetzen eindeutig strukturiert.
Eingabe für die in [KS93] beschriebenen Algorithmen zur Erzeugung strukturierter Netze sind geschlossene Beschreibungen der Domain. Auf der Basis dieser Beschreibung wird ein regelmäßiges Viereckgitter aus dem Parameterraum in die Domain abgebildet und dabei deformiert. Die geschlossene Beschreibung, der in diskreten Scanpunkten gegebenen Oberfläche, läßt sich nur über den Umweg der Interpolation gewinnen.
Viereckgnetze sind weniger flexibel, sich an zwingende Kantenzüge anzupassen, die bei Animationsnetzen gefordert werden. Aus diesen Gründen finden Viereckgnetze in dieser Arbeit nur eine knappe Erwähnung.

Viele 2.5D-Netzgenerierungsalgorithmen basieren auf 2D-Verfahren. Entweder wird die gekrümmte Domain in die Ebene abgebildet, das Netz generiert und die Knotenpunktkoordinaten zurück auf die Oberfläche abgebildet, oder der Netzgenerierungsalgorithmus arbeitet direkt auf der Oberfläche. Im letzten Fall werden die gleichen Kontrollstrukturen wie in der Ebene benutzt.

Die Formulierung einer Abbildung gefalteter und gekrümmter Oberflächen in die Ebene ist nicht möglich. Daher wird die Oberfläche vor der Abbildung partitioniert, und die Partitionen werden getrennt abgebildet. Die Winkel- und Längenverzerrung der Abbildung steht im Zusammenhang mit der Größe und den Krümmungsverhältnissen einer Partition. Der Themenbereich der optimalen Partitionierung einer Oberfläche und anschließender Abbildung in die Ebene stellt ein eigenes mathematisches Forschungsgebiet dar und wird in dieser Arbeit nicht theoretisch vertieft.

4.1 Drei Problemklassen der Netzgenerierung

Bezüglich einer 2D- oder 2.5D-Domain (gekrümmte Oberfläche) existieren drei mögliche Szenarien, in denen die Generierung eines neuen Polygonnetzes *(meshing)* gewünscht wird:

1. In der Domain sind die zu verwendenden Netzknotenpunkte vorgegeben. Ein Polygonnetz aus Dreiecken wird ausschließlich unter Verwendung der vorhandenen Knoten erstellt (Abschnitt 4.2).

2. Nur die Grenzen der kontinuierlichen Domain sind gegeben. Sämtliche Netzknotenpunkte innerhalb der Domain werden vom Algorithmus erzeugt. Bei Bedarf muß die Domaingrenze durch initiale Knoten ebenfalls diskretisiert werden (Abschnitt 4.3).

3. Die Eingabe des Algorithmus ist bereits ein Polygonnetz, aus dem unter Verwendung existierender Knoten oder unter Einfügen von interpolierten Knoten ein neues Netz erstellt werden soll. Das neue Netz stellt eine optimierte Version des bestehenden Netzes dar. Häufig soll eine durch das Eingabenetz approximierte Oberflächenfunktion (z.B. durch Scanvorgang ermittelt) mit Hilfe weniger Datenpunkte adäquat dargestellt werden. Zu dieser Klasse sollen auch Verfahren des *mesh refinement* zählen, die nicht die Datenmenge, sondern die geometrische Struktur des 2D oder 2.5D-Netzes optimieren (Abschnitt 4.4.1).

Thema dieser Arbeit ist die Erstellung von Dreiecksnetzen auf der Datengrundlage eines hochaufgelösten Viereckснetzes, den 3D-Oberflächen-Scandaten. Dieses Problem markiert den Grenzbereich zwischen der zweiten und dritten Klasse. Zur zweiten Klasse zählt es, wenn man das gleichmäßige, hochaufgelöste Viereckсgitter als Approximation der kontinuierlichen Domain betrachtet, in der ein neues Netz erzeugt wird. Andererseits sind die neuen Knotenpunkte identisch zu Scanpunkten aus dem Viereckснetz, weshalb das Problem zur dritten Klasse gehört.

4.2 Algorithmen zur Triangulierung von Punktmengen

Algorithmen dieser Klasse akzeptieren als Eingabe eine endliche Punktmenge in 2D oder 3D. Die 3D-Punkte sind Samples einer 3D-Oberfläche, die durch die Triangulierung approximiert wird. Auf der Oberfläche (2.5D-Domain) stellt die Triangulierung ebenfalls einen planaren Graphen dar.
Die Grenze der Domain (boundary) wird von einem zwingenden äußeren Kantenzug gebildet, der alle Knotenpunkte einschließt oder enthält. Einige Algorithmen erlauben die Definition weiterer Constraints. Topologisch mehrfach verbundene Domains enthalten zusätzlich geschlossene innere Kantenzüge. Sie definieren die "Löcher" der Domain, die keine Knotenpunkte enthalten und frei von Kanten bleiben. Zwingende innere Kantenzüge sollen bei der Triangulierung nicht verschwinden.

Ein wichtiges Merkmal des Triangulierungsalgorithmus ist das Optimierungskriterium. Der Algorithmus approximiert das globale Optimum bezüglich eines lokalen Optimalitätskriteriums (siehe S. 23). Algorithmen, die nur triangulieren, ohne bezüglich irgendeines Kriteriums zu optimieren, seien an dieser Stelle ausgeklammert, da sie für Anwendungen nur eine unbedeutende Rolle spielen.
Das ideale Laufzeitverhalten für einen Algorithmus ist $O(n)$, wobei n die Anzahl der Eingabepunkte bezeichnet.
Eine Einführung in eine Reihe von Algorithmen zur optimalen Triangulierung bezüglich verschiedener globaler Optimalitätskriterien findet sich in [BE92]. Stellvertretend wird nun ein Algorithmus zur Berechnung der CDT einer Punktmenge vorgestellt, der erlaubt, sukzessive weitere zwingende Kanten einzusetzen.

4.2.1 RENKAs Algorithmus zur Constrained-Delaunay-Triangulierung von Punktmengen

Die Constrained-Delaunay-Triangulierung verbindet zahlreiche günstige Eigenschaften, die hier noch einmal zusammengefaßt werden:

1. Das Umkreiskriterium impliziert die lokalen Optimalitätskriterien Minimierung des größten Umkreises und das Max-Min-Winkelkriterium. Die lokale Erfüllung des Max-Min-Winkelkriteriums impliziert die global optimale Triangulierung.

2. Der Algorithmus erlaubt die Definition zwingender Kantenzüge (constraints).

3. Die DT ist ein planarer Graph. Nach Eulers Formel besitzt sie maximal $3n \Leftrightarrow 6$ Kanten und $2n \Leftrightarrow 5$ Dreiecke für eine Punktmenge P mit $|P| = n$. Zweidimensionale Probleme in der Ebene können mit der DT als Datenstruktur mit linearem Speicherbedarf dargestellt werden.

4. Sie stellt eine ideale Datenstruktur für das Problem des nächsten Nachbarpunktes in einer Punktmenge dar *(closest node problem)*. Insbesondere läßt sich leicht aus $\mathsf{DT}(P)$ der minimal spannende Baum (EMST - *euclidean minimum spanning tree*) [Mül87] und aus $\mathsf{CDT}(P)$ der constrained EMST gewinnen [Che89a].

Durch den Zusammenhang zwischen lokaler und globaler Optimalität beim Max-Min-Winkelkriterium ist der Entwurf effizienter DT- und CDT-Algorithmen möglich. Während global optimale Triangulierungen auf der Basis anderer lokaler Optimalitätskriterien Probleme mit Komplexität $O(n^2 logn)$ (Min-Max-Winkelkriterium, Max-Min-Höhenkriterium, siehe S. 23) darstellen oder eventuell sogar NP-vollständig sind (die Frage der Komplexität der minimum weight triangulation stellt ein offenes Problem dar [BE92]), erreichen DT-Algorithmen unter iterativer Anwendung lokaler Max-Min-Winkeloperationen oder mit Hilfe von Divide-And-Conquour-Strategien das globale Optimum in $O(n \log n)$. Seit [Che89a] existiert auch zur Berechnung der CDT ein Algorithmus mit Komplexität $O(n \log n)$.

CHEWs Algorithmus verlangt als Eingabe die gesamte Menge der Punkte und zwingenden Kantenzüge. CLINE und RENKA präsentieren einen flexibleren Algorithmus mit Komplexität $O(n \log n)$ für die initiale DT. Im Anschluß daran können sukzessive weitere zwingende Kantenzüge der Triangulierung hinzugefügt werden. Der Schritt, nachträgliche Kanten einzusetzen, besitzt die Komplexität $O(k^3)$, wobei k die Zahl der von der Kante geschnittenen Delaunay-Dreiecke darstellt, welche in praktischen Situationen hinreichend beschränkt ist. Die Möglichkeit, sukzessive neue Kanten hinzuzufügen, zusammen mit der frei verfügbaren FORTRAN-Implementierung, macht den Algorithmus von CLINE und RENKA für den praktischen Einsatz besonders interessant [Ren96]. Im folgenden sei er daher als Beispiel skizziert. Die Notation der zwingenden begrenzenden und inneren Kantenzüge entspricht Abschnitt 3.2.7. Eine Prozedur besitzt die Notation:
Prozedurname($< Eingabeparameter >$; $< Ausgabeparameter >$).

Der Algorithmus basiert auf gegebener DT(P), berechnet in $O(n \log n)$. Er umfaßt drei zusätzliche Teilalgorithmen **constrainedDelaunayTriangulation**($P, B_1, ..., B_l, E,$: T), **addedge**(e, T, R ; T, R) und **retriangulate**($nodeset, a, b$, DT(P), R ; T). Der hier eingefügte DT-Algorithmus **DelaunayTriangulation**(P ; T) stammt aus [Rei91].

constrainedDelaunayTriangulation($P, B_1, ..., B_l, E$; T)

Eingabe: Knotenmenge P, zwingende Kantenzüge (boundaries) $B_1, ..., B_l$ und Menge der zwingenden inneren Kanten E.
Ausgabe: CDT der Eingabeparameter.
(1) T := **DelaunayTriangulation**(P), sei H die Menge sämtlicher Delaunay-Dreieckskanten aus der konvexen Hülle von P und $R := H$.
(2) Für jede Kante $e \in (B_1 \cup ... \cup B_l \cup E) \setminus H$, **addedge**($e$, T($P$), R).
(3) Für $i = 1, ..., l$, lösche alle Dreiecke, die sich außerhalb B_i befinden.

addedge(e, T, R ; T, R)

Eingabe: Kante e, Triangulierung T(P), R Menge der zwingenden Kanten.
Ausgabe: R um e erweitert, T ist konvexe CDT von P unter Berücksichtigung von R.
(1) Bestimme alle Dreiecke in der momentanen Triangulierung T(P), die e schneiden und entferne diese aus T. Der Abbruch erfolgt, falls keine solchen

Dreicke existieren.
(2) Sei Ω_e die Vereinigung aller in (1) entfernten Dreiecke, seien B_e die Kanten auf dem Rand der konvexen Hülle von Ω_e, R_e die Menge aller zwingenden Kanten in Ω_e und *nodeset* die Menge der Knoten in Ω_e ohne a, b.
(3) Retrianguliere die Knoten aus Ω_e in T, die sich links von der gerichteten Kante $[a, b]$ befinden, **retriangulate**$(nodeset, a, b, \mathsf{DT}(P), R_e \cup B_e)$.
(4) Retrianguliere die Knoten aus Ω_e in T, die sich links von der gerichteten Kante $[b, a]$ befinden, **retriangulate**$(nodeset, a, b, \mathsf{DT}(P), R_e \cup B_e)$.
(5) $R := R \cup e$.

retriangulate$(nodeset, a, b, \mathsf{T}(P), R\,;\, \mathsf{T})$

Eingabe: *nodeset* die Menge der Knoten in Ω_e ohne a, b, Knoten a, b, Triangulierung T(P), R Menge der zwingenden Kanten.
Ausgabe: Triangulierung T ergänzt um die DT-Triangulierung der Knoten in *nodeset*, die sich links von der Kante $[a, b]$ befinden.
(1) Bestimme $X \subset nodeset$, so daß X alle Knoten links von Kante $[a, b]$ umfaßt, die nicht vom Mittelpunkt von $[a, b]$ durch eine Kante aus R separiert sind.
(2) Bestimme $x \in X$ so, daß $\sphericalangle bxa$ maximal wird.
(3) Füge $\triangle abx$ zu T hinzu.
(4) Entferne x aus *nodeset*.
(5) Falls $[a, x] \notin R$, **retriangulate**$(nodeset, a, x, \mathsf{T}(P), R)$.
(6) Falls $[b, x] \notin R$, **retriangulate**$(nodeset, b, x, \mathsf{T}(P), R)$.

DelaunayTriangulation$(P\,;\, \mathsf{T})$

Eingabe: Menge der zu triangulierenden Knoten P. Die Knoten dürfen nicht sämtlich kollinear sein.
Ausgabe: $\mathsf{DT}(P)$
(1) Wähle aus P drei Knoten aus, deren Umkreis keine weiteren Punkte aus P enthält, und bilde daraus die Initialtriangulierung DT_3.
(2) Bestimme einen bisher noch nicht berücksichtigten Knoten $p \in P$ so, daß die konvexe Hülle aus DT_i und p keine weiteren, noch nicht berücksichtigten Punkte aus P enthält.
(3) Suche alle von p aus sichtbaren Knoten der konvexen Hülle von DT_i (ein Knoten aus DT_i ist sichtbar von p, falls ihre verbindende Strecke die konvexe Hülle nicht schneidet).
(4) Erzeuge durch Verbinden der sichtbaren Knoten mit p eine vorläufige Triangulierung.
(5) Für jedes Dreieck $\triangle d$ der vorläufigen Triangulierung, vertausche die gemeinsame Kante von $\triangle d$ und dem anliegenden Dreieck aus DT_i. Teste rekursiv die beiden anderen Nachbarn des alten Dreiecks [Men94].
(6) $i := i + 1$,
(7) Abbruch und Ausgabe von $\mathsf{DT}(P) = \mathsf{DT}_i$, falls $i = |P|$, sonst weiter mit (2).

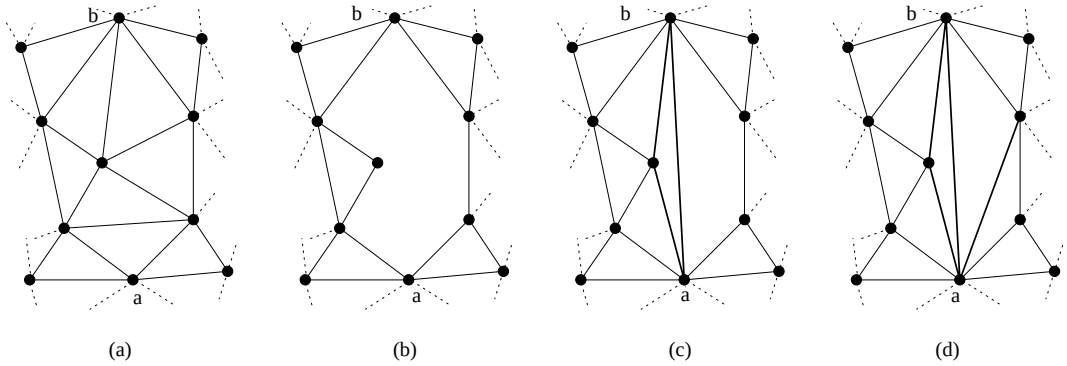


Abbildung 4.1: Einfügen einer Kante und Retriangulierung
(a) die ursprüngliche DT in diesem Bereich, (b) alle Dreiecke, die die Kante $[a, b]$ schneiden, wurden entfernt, (c) Retriangulierung des Bereichs links von der Kante $[a, b]$, (d) Retriangulierung des Bereichs rechts von der Kante $[a, b]$ (nach [Ren96])

Der zweite Schritt in **constrainedDelaunayTriangulation** stellt die Existenz der zwingenden Kanten sicher. Abbildung 4.1 illustriert, wie eine Kante in die bestehende Triangulierung eingesetzt wird. Mit dem dritten Schritt werden der konvexen Triangulierung alle Dreiecke entnommen, die sich außerhalb der begrenzenden Kantenzüge (boundaries) befinden. Im Ergebnis liegt eine Triangulierung wie in Abbildung 3.9 vor. Beweise zur Korrektheit des Algorithmus werden in [Ren96] gegeben.

4.3 Meshing mit Steiner-Punkten

Eingabe dieser Algorithmen ist eine begrenzte Domain in Form einer kontinuierlichen Funktion in 2D oder einer gekrümmten Oberfläche in 3D. Sämtliche Knotenpunkte des Netzes innerhalb der Domain werden vom Algorithmus gesetzt. Falls die Domaingrenze nicht bereits diskret als Kantenzug durch eine Punktmenge approximiert ist, muß der Algorithmus zusätzlich die Knotenpunkte auf der Domaingrenze wählen. BERN und EPPSTEIN [BE92] bezeichnen die vom Algorithmus bestimmten Netzknotenpunkte als *Steiner points*.

Steiner-Punkte ändern den Charakter des Triangulierungsalgorithmus aus Abschnitt 4.2. Bei der Triangulierung von Punktmengen besteht das Ziel, einen eindeutigen, die Netzstruktur optimierenden Algorithmus zu besitzen. Steiner-Triangulierer sind dagegen im idealen Fall approximierende Algorithmen. Unter der Bedingung, von einem Qualitätsmaß in definierten Grenzen abzuweichen, approximieren sie die optimale Anzahl und Position der dazu notwendigen Steiner-Punkte. Die Triangulierung der Steiner-Punktmenge soll das globale Kriterium an sich, wie z.B. minimale Summe der Kantenlängen oder minimale größte Winkel, unter gegebenen Randbedingungen optimieren. Beispiel für ein typisches Problem: Trianguliere eine Domain, deren Rand durch einen Kantenzug zwischen n Knotenpunkten beschrieben ist, kein Dreieckswinkel soll größer als $90°$ sein. und die Anzahl der erzeugten Steiner-Punkte muß polynomial in n sein.

Der optimierende Steiner-Triangulierungsalgorithmus approximiert das Optimum zwischen zwei Randbedingungen: Entweder wird die gewünschte Anzahl Steiner-Punkte festgelegt und danach die optimale Position der Punkte für eine optimale Triangulierung nach dem gewünschten Kriterium berechnet, oder das globale Qualitätsmaß wird zuvor spezifiziert, und der Algorithmus erfüllt das Qualitätsmaß mit minimal notwendiger Anzahl Steiner-Punkte. Im ersten Fall gibt es keine Garantie für die Qualität der berechneten Triangulierung. Im zweiten Fall bricht der Algorithmus unter Umständen nicht ab. Beispielsweise kann der minimale Winkel einer Triangulierung unter Zunahme weiterer Punkte beliebig nah an 60° angenähert werden.

Approximierende Steiner-Triangulierungsalgorithmen basieren auf Methoden, die die Plazierung der Steiner-Punkte mit der Erzeugung der Netzelemente (Dreiecke) verbinden. Durch diese Koppelung kann zwischen der Gestalt des Netzes (globales Optimalitätskriterium) und den Randbedingungen für die Steiner-Punkte (z.B. hohe Punktdichte in der Nähe der detaillierten Domain-Begrenzung, Obergrenze für Anzahl Steiner-Punkte) optimiert werden. Diese Algorithmen finden ab S. 45 Erwähnung.

Einfache Algorithmen der Steiner-Triangulierung konzentrieren sich zuerst auf die Verteilung der Steiner-Punkte in der Domain. Die Triangulierung der Punktmenge erfolgt in einem separaten Schritt oder entsteht beim Setzen der Steiner-Punkte "nebenbei", ohne daß die Gestalt der Dreiecke Einfluß auf die Lage und Anzahl der Punkte hat (S. 40).

Der große Bereich der strukturierten Netzgenerierung findet ab S. 42 nur kurze Erwähnung (siehe auch Ausführungen auf S. 33). Bei diesem Ansatz wird die Domain in vierseitige Regionen unterteilt, in die aus dem Parameterraum gleichmäßige Viereckgitter abgebildet werden.

4.3.1 Eigenschaften von Steiner-Algorithmen

Bevor in den folgenden Abschnitten die methodischen Ansätze und einige Algorithmen zur Steiner-Triangulierung vorgestellt werden, stellt die folgende Merkmalsliste Kriterien für den Vergleich der Algorithmen zusammen. Mit Hilfe dieser Merkmale lassen sich die einzelnen Algorithmen vergleichen und bewerten. Motiviert wurde diese Struktur von [HL88], [BE92] und [SNTM91].

1. Welcher Art sind die erzeugten Elemente: Dreiecke oder Vierecke?

2. Adaptiert sich die Steiner-Punktdichte an den Detaillierungsgrad der Domain-Begrenzung *(graded mesh)*, oder versucht der Algorithmus ein *isotropes* Netz zu approximieren, in dem alle Kanten Einheitenlänge besitzen? Zusätzlich für die Netzgenerierung auf Oberflächen in 3D: Ist die Steiner-Punktdichte eine Funktion der Oberflächenkrümmung?

3. Welches Laufzeitverhalten besitzt der Algorithmus?

4. Welche Eingabe-Domains werden akzeptiert: 2D, 2.5D oder gefaltete Oberflächen? Einfach oder mehrfach verbunden (Domain enthält Löcher)?

5. Für Triangulierungen: Bezüglich welchen globalen Kriteriums wird die Anzahl und Position der Steiner-Punkte optimiert, d.h. welche Form haben die erzeugten Dreiecke?

6. Arbeitet der Algorithmus heuristisch, und ist die Approximation der optimalen Anzahl und Position der Steiner-Punkte nur empirisch abzusichern, oder kann die Optimalität der Steiner-Polygonnetze garantiert werden?

7. Ist der Algorithmus in der Lage, zwingende Kantenzüge (constraints) im Netz zu respektieren?

8. Hat das erzeugte Netz über die Form der einzelnen Elemente hinaus noch andere Struktureigenschaften, z.B. Orientierung der Dreiecke gemäß einer Feldstruktur?

In den nächsten Abschnitten werden diese Kriterien bei der Vorstellung verschiedener Steiner-Triangulierungsalgorithmen erwähnt, sofern es sich um typische Eigenschaften der Algorithmen handelt.

4.3.2 Meshing ohne Optimierung der Netzelemente

Die Algorithmen dieser Klasse haben die gemeinsame Eigenschaft, daß die Plazierung der Steiner-Punkte ohne gesicherte Kontrolle der globalen Eigenschaften des Netzes geschieht. Basierend auf der Topologie der Domain werden die Steiner-Punkte z.B. mit einem Gitter-, Quadtree- (S. 57) oder anderen, rekursiven Unterteilungsverfahren in der Domain verteilt. Entweder muß die erzeugte Punktmenge in einem separaten Schritt trianguliert werden, oder sie entstand beim Setzen der Steiner-Punkte "nebenbei", ohne daß die Gestalt der Dreiecke Einfluß auf die Lage und Anzahl der Punkte hatte. In den folgenden Abschnitten werden zwei Algorithmen dieser Klasse vorgestellt.

Cavendish Methode

Bei diesem Verfahren wird die Domain zuerst manuell in unterschiedliche Zonen i aufgeteilt. Für jede Zone wird ein Maßstab d_i definiert, abhängig von der gewünschten Dreiecksgröße in dieser Zone. In jeder Zone wird die Domain-Begrenzung äquidistant mit Initialknotenpunkten besetzt. Die Distanz ist jeweils d_i. Die Knotenpunkte, die den begrenzenden Kantenzug der Domain bilden, werden zur Menge der Initialpunkte hinzugenommen. Anschließend wird jeder Zone i ein Gitter mit Linienabstand d_i überlagert (Abbildung 4.2). In jede Zelle wird nun zufällig ein Steiner-Punkt plaziert. Der Steiner-Punkt bleibt erhalten, wenn er sich innerhalb der Domain befindet und wenn sein Abstand von der Domain-Begrenzung und allen zuvor generierten Knotenpunkten ein Maß r_i nicht unterschreitet. Sonst wird der Punkt in dieser Zelle durch einen neuen, auch zufällig plazierten Steiner-Punkt ersetzt und die Prozedur wiederholt. Konnte nach einer festen Anzahl von Versuchen, z.B. 5, kein akzeptabler Kandidat gefunden werden, dann bleibt die Zelle leer. Abschließend wird die Punktmenge unter Respektierung der Domain-Begrenzung trianguliert (z.B. DT) [HL88].

Die abschließende Triangulierung der Punktmenge kann zwar global optimal (z.B. Max-Min-Winkelkriterium) sein, aber es ist dem Zufall überlassen, daß die Steiner-Punkte auch

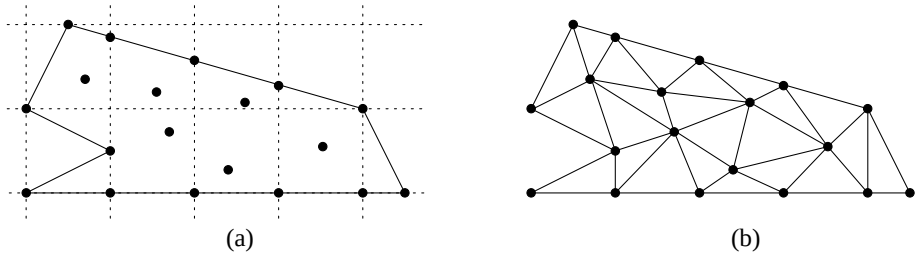


Abbildung 4.2: Cavendish Methode
Beispiel mit nur einer Zone. In (a) sind die Initialknotenpunkte auf der Domain-Begrenzung, sowie die in den Gitterzellen eingezeichneten, zufällig gesetzten Steiner-Knotenpunkte sichtbar. (b) zeigt das Netz nach Triangulierung der Punktmenge.

eine gute global optimale Triangulierung erlauben. Der Ansatz, Zonen mit unterschiedlichem geometrischen Maßstab zu definieren, erlaubt zwar prinzipiell, ein Netz mit variabler Knotenpunktdichte zu erstellen, aber ist auf manuelles Eingreifen angewiesen. Der Algorithmus unterstützt keine constraints in der Domain. Ein notwendiger Bestandteil des Algorithmus ist eine Funktion, die testet, ob ein erzeugter Punkt innerhalb oder außerhalb der Domain-Begrenzung liegt. Eine solche Funktion dürfte leicht erweiterbar sein, so daß sie mit mehrfach verbundenen Domains arbeiten kann. Dann arbeitet auch der Triangulierer auf mehrfach verbundenen Domains.

Rekursive topologische Dekomposition

Die Algorithmen dieser Familie spalten die Domain immer weiter in Subdomains auf, bis diese akzeptable Netzelemente darstellen. Auf der Basis von Heuristiken werden die Schnittkanten ausgewählt. Die Unterteilungsstrategie soll die Entstehung wohlgeformter Dreiecke begünstigen. Eine einfache Strategie besteht beispielsweise darin, zuerst Schnittkanten zwischen Kanten der Domainbegrenzung zu finden, die einen Winkel größer 180° bilden. Die entstandenen konvexen Polygone werden danach so unterteilt, daß die Kantenlängen nicht zu stark variieren (siehe Abbildung 4.3). Dadurch können spitze Dreiecke vermieden werden.

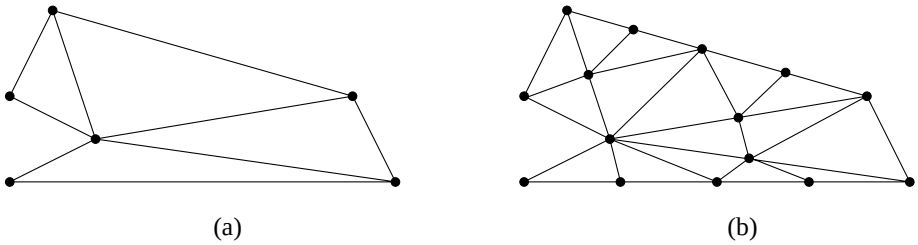


Abbildung 4.3: Topologische Dekomposition
Zuerst werden die Knoten des begrenzenden Kantenzuges verbunden und einfache Subdomains erzeugt (a), danach erfolgt die weitere Unterteilung der Subdomains durch Schnittkanten zwischen Steiner-Punkten.

Bei diesem Ansatz ist die Topologie der Domain maßgeblich für die Form der Dreiecke.

Die Wohlgeformtheit der Dreiecke ist bei der rekursiven Unterteilung durch ausgefeilte Heuristiken nur bedingt kontrollierbar.

4.3.3 Erzeugung strukturierter Netze

Bei der Generierung strukturierter Netze (siehe S. 33) werden Viereckgitter in die Domain abgebildet. Die Knotenpunkte im viereckigen Gitternetz stellen in diesem Fall die Steiner-Punkte dar. Steiner-Punkt ist eigentlich ein Begriff aus der Triangulierungstheorie für unstrukturierte Netze, aber soll in dieser Klassifikation für jeden Netzknotenpunkt benutzt werden, der von einem Algorithmus in der Domain gesetzt wird. Damit werden die beiden Bereiche der unstrukturierten und strukturierten Netzgenerierung gleichermaßen von der Klassifikation erfaßt. Jedes Viereckgitter ist ohnehin durch triviales Einsetzen einer Diagonalen in jedem Viereck in ein Dreieckgitter überführbar.

Abbildung eines Gitters in die Domain

Die trivialste und effizienteste Methode zur Erzeugung eines strukturierten Netzes besteht darin, ein Viereckgitter mit der gewünschten Kantenlänge 1:1 komplett in die Domain abzubilden. Alle Anteile, die über die Begrenzung der Domain ragen oder Kanten der Domain beinhalten, werden abgeschnitten. Je feiner das abgebildete Netz ist, umso weniger Details gehen an der Begrenzung der Domain verloren.
Bessere Ergebnisse können erreicht werden, wenn die Elemente an der Domain-Begrenzung nachbearbeitet werden. Abbildung 4.4 zeigt das Ergebnis nach der Justage der Netzkanten, die die Domain-Begrenzung berühren. Auch hier ist die Auflösung von Details an der Begrenzung limitiert. Hybridverfahren aus dieser Methode und einem Nachbearbeitungsschritt, der die unzulänglichen Netzelemente an der Begrenzung dekomponiert und dort kleinere dreieckige Netzelemente einfügt, können die Effizienz der Gittermethode mit Adaptivität an der Domain-Begrenzung paaren.

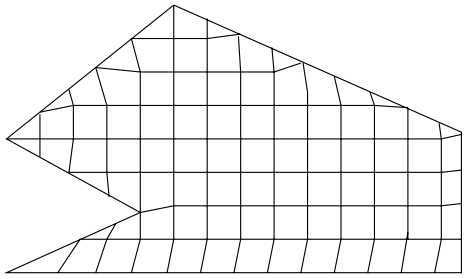


Abbildung 4.4: Abbildung eines regelmäßigen Gitters in die Domain
Im Innern der Domain sind die Netzelemente gleichmässig und besitzen die gewünschte Form. In Bereichen, in denen der Rand nicht zufällig mit der Viereckstruktur übereinstimmt, erzeugt auch die Nachbearbeitung keine wohlgeformten Elemente.

Der Algorithmus wird immer zwei strukturell verschiedene Bereiche im Netz generie-

ren. Im Innern sind die Netzelemente sehr gleichmäßig und nahe an der Domainbegrenzung ungleichmäßig. Ein weicher Übergang zwischen beiden Bereichen ist nicht möglich. Die Gesamtqualität ist umso besser, je größer die Region in Relation zur Größe eines Netzelements ist und je einfacher die Domain-Begrenzung ist.

Mit einer einfachen Abbildungsvorschrift, die die Kantenlänge des Netzes als Bogenlänge erhält, ist es möglich Oberflächen in 3D, die nur eine Krümmungsrichtung besitzen, mit einem Viereckgitter zu approximieren.

Statt eines Viereckgitters kann hier auch ein anderes Polygongitter verwendet werden. Ein gleichschenkliges Dreiecksnetz approximiert vor der Nachbearbeitung bereits genauer die Domain-Begrenzung.

Abbildung von Templates in die partitionierte Domain

Dieser Absatz weist auf Verfeinerungen der trivialen Methode hin. Erstens besteht die Möglichkeit, die Domain in einem Vorverarbeitungsschritt in mehrere viereckige Subdomains zu zerlegen und die Abbildung von allgemeinen Gittern aus dem Parameterraum für jede Subdomain getrennt, aber bezüglich der Verbindungen aufeinander abgestimmt vorzunehmen. Bereits bei zweidimensionalen Domains ergibt sich eine bessere Anpassung an kompliziertere Geometrien (siehe Abb. 4.5). Insbesondere bei gefalteten und kompliziert gekrümmten Oberflächen ist die Unterteilung der Domain in *Patches* die einzige Möglichkeit, die Komplexität der Abbildungsfunktionen zu beschränken (s. u.). Die Seiten der Patches können gekrümmt sein. Die Unterteilung der Domain in geeignete vierseitige Patches *(region decomposition)* stellt selbst immer noch ein Netzgenerierungsproblem dar und wird teilweise manuell gelöst.
Die zweite Verfeinerungsmöglichkeit bezieht sich auf die Abbildungsfunktion, die die Ko-

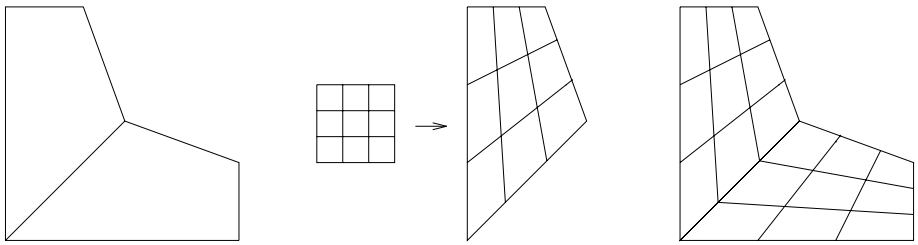


Abbildung 4.5: Abbildung von transformierten Templates in die partitionierte Domain
(a) die Domain wird in Patches partitioniert, (b) Netztemplates aus dem Parameterraum werden passend transformiert und (c) in die Patches abgebildet, um das Gesamtnetz zu erzeugen.

ordinaten des einfachen Gitters im Parameterraum so deformieren, daß es sich in das Patch einfügt. Die Koordinatentransformation hängt bei 2D-Patches von der Größe und der Krümmung der Seiten des Patch ab. Bei 3D-Patches werden die Gitterpunkte zusätzlich abhängig von der Oberflächenkrümmung in $\mathbb{R}^3$ transformiert (Abb. 4.6). Grundlage für die Adaptivität dieser Verfahren ist eine funktionale Beschreibung des Patch, die für

diskret gegebene Objekte (z.B. Scans vom Gipsmodell eines virtuellen Charakters) erst durch Interpolation gewonnen werden muß. Mathematisch wird die Abbildungsfunktion mit Hilfe algebraischer Methoden oder numerischer Methoden zur Lösung von Differentialgleichungen gewonnen.

EPPSTEIN [BE92] weist darauf hin, daß es für geometrisch zu komplizierte Domains unmöglich sein kann, eine Abbildung zu finden, mit der sich eine gute Approximation der Domain finden läßt. Die Aufteilung der Domain in Patches ermöglicht die Bestimmung einfacher Abbildungen für jedes Patch.

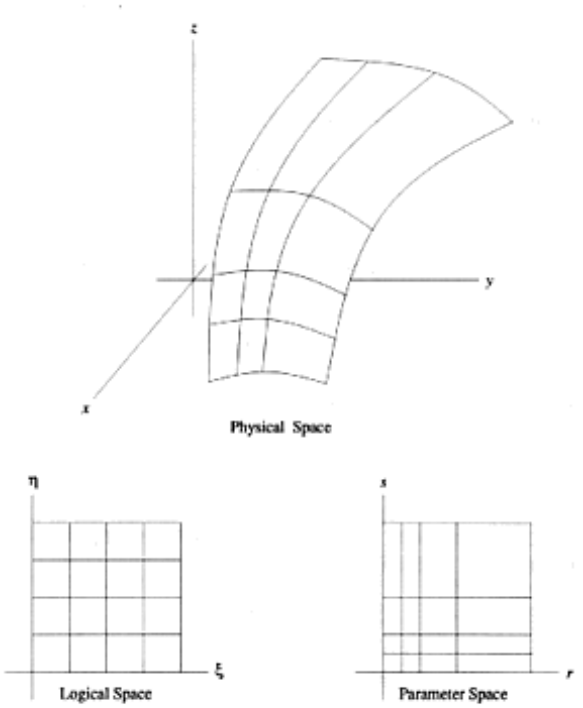


Abbildung 4.6: Vierseitiger Oberflächen-Patch
Im oberen Bild ist das strukturierte Netz auf einem vierseitige Oberflächen-Patch im dreidimensionalen physikalischen Koordinatensystem (physical space) sichtbar. Nach der Transformation liegt es im parametrischen zweidimensionalen Koordinatensystem (parameter space) vor (rechts unten). Der Parameterraum nähert in der kartesische Ebene die geodätischen Entfernungen auf der Oberfläche an. Der "logical space" links unten gibt die einheitlich strukturierte Nachbarschaftsbeziehung der Elemente wieder (aus [KS93]).

4.3.4 Optimierendes Meshing: Advancing Front Technique, Quadtree Meshing, CHEWs Algorithmus für Oberflächen

Die Algorithmen dieser Klasse berücksichtigen die gewünschte Form der Netzelemente bereits bei der Dekomposition der Domain. Die Dekomposition kann iterativ oder rekursiv erfolgen. Bei der iterativen Dekomposition wird die Größe der Arbeits-Domain (ungemeshte Region) um jedes neu erzeugte Dreieckselement reduziert. Die Triangulierung ist abgeschlossen, sobald die Arbeits-Domain verschwunden ist und die Domain vollständig parkettiert ist. In jedem Schritt hat der Algorithmus die Kontrolle über Form und Größe des neuen Dreieckselements. Die Kontrollstrukturen, die über den Ort des erzeugten Dreiecks entscheiden, arbeiten heuristisch und minimieren die Anzahl der Steiner-Punkte abhängig von der Größe und Komplexität der Domain. Typische Repräsentanten dieser Klasse sind Algorithmen der *Advancing Front Technique*.
Ausgereiftere Verfahren der rekursiven Dekomposition (verglichen mit S. 41) dekomponieren die Domain nicht mit Hilfe heuristisch bestimmter Schnittlinien, sondern zerlegen die Domain in definierte Elemente. Aus der Beschreibung der Domain mit Hilfe von Kreisen oder Quadraten (Quadtree S. 57) werden unmittelbar Dreieckselemente definierter Qualität gewonnen. Andere Verfahren repräsentieren die Domain als CDT ihrer Initialknotenpunkte. Nach Einfügen eines Steiner-Punktes wird die CDT sofort aktualisiert. Zu jedem Zeitpunkt liegt so eine Triangulierung der Domain mit CDT-optimalen Dreiecken vor. Der Triangulierungsalgorithmus bricht ab, sobald alle Dreiecke eine gewünschte Größe nicht überschreiten. CHEW [Che93] beschreibt einen derartigen Algorithmus zur Triangulierung von Oberflächen.

Advancing Front Triangulation

Advancing Front Techniques (AFT) bezeichnen einen Ansatz der Netzgenerierung, bei dem die Domain beliebiger Topologie (mehrfach zusammenhängend) ausgehend von ihren Rändern sequentiell mit Dreiecken besetzt wird. Der noch nicht triangulierte Bereich, die noch ungemeshte Region, schrumpft während der Bearbeitung wie ein schmilzender Eisblock und zerfällt dabei unter Umständen in mehrere Teile. Die Summe der Ränder aller noch nicht triangulierten Bereiche wird als Front bezeichnet. Die Domain ist vollständig trianguliert, sobald die Front verschwunden ist.

In Übereinstimmung mit der Arbeit von FARESTAM [Far93] wird die Arbeitsweise eines AFT-Algorithmus stufenweise erklärt. Nach der Vorstellung der grundsätzlichen Datenstrukturen der Methode, wird zuerst die Kontrollstrukur eines AFT-Algorithmus vorgestellt. Danach werden die verwendeten Prozeduren in einer Minimalausführung beschrieben. Diese Beschreibung des Algorithmus stellt lediglich sicher, daß das Verfahren nach einer endlichen Zahl von Schritten eine Triangulierung erzeugt. In der Ausbaustufe folgen Vorschläge für eine effiziente Implementierung der Teilschritte. Diese Implementierung benutzt die CDT als effiziente Datenstruktur zur Repräsentation der Domain und produziert CDT-optimale Triangulierungen mit einer moderaten Zahl von Steiner-Punkten.

Grundlegene Datenstrukturen der AFT-Methode. Die zentrale Datenstruktur der AFT-Methode ist die dynamische Front, die bei ihrer Initialisierung die Domain-

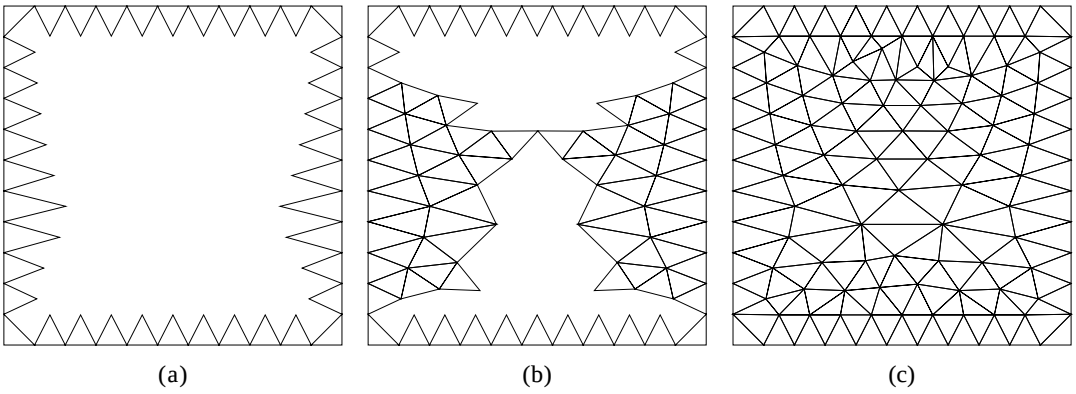


Abbildung 4.7: Triangulierung einer quadratischen Domain mit der AFT
(a) die Domaingrenze wurde durch Knotenpunkte initialisiert und eine erste Schicht Dreiecke wurde erzeugt, (b) Die Front teilt sich, (c) Die Front ist verschwunden, die Domain ist trianguliert (aus [Far93]).

begrenzungen approximiert und deren Verschwinden die Terminierung des Algorithmus beinhaltet. Die Datenstrukur zur Repräsentierung der Front soll Domains mit Löchern (mehrfach verbunden) als Eingabe zulassen und zur Laufzeit die automatische Aufsplittung der ungemeshten Domain in mehrere Subdomains gestatten.

Aus der Datenstruktur der Front soll jederzeit die ungemeshte Region eindeutig ableitbar sein. Dazu wird die Front als Sammlung gerichteter, endlicher, einfacher (keine Mehrfachkanten), planarer und geschlossener Kantenzüge definiert, die als Besonderheit gemeinsame Knotenpunkte besitzen können, aber sich sonst nicht überschneiden (Abb. 4.8 (b)). Bei der Initialisierung approximiert die Front die Domainbegrenzung, indem die kontinuierliche Begrenzung durch Setzen von Initialpunkten im gewünschten Abstand (z.B. äquidistant oder krümmungsabhängig) diskretisiert wird und je zwei aufeinanderfolgende Punkte $E.orig$ und $E.dest$ zu einer Kante E verbunden werden. Links von der gerichteten Kante befindet sich jeweils die ungemeshte Region oder eine ungemeshte Unterregion, die durch Aufsplittung zustande kam. Rechts befindet sich entweder der bereits gemeshte Anteil der Domain oder die Kante bildet die Domaingrenze (Abb. 4.8 (a)).
Nachdem über einer Frontkante eine neues Dreieck konstruiert wurde, wird die Front aktualisiert und beschreibt so zu jedem Zeitpunkt korrekt den ungemeshten Bereich der Domain.
Zwei beliebige Kanten schneiden sich höchstens an ihren Anfangs- und Endpunkten, wenn eine Kante die Nachfolgekante der anderen Kante im Kantenzug darstellt oder die spezielle Situation vorliegt, daß vier Kanten an einem Knotenpunkt anliegen (Abb. 4.8 (b)). Auch in diesem Fall sind Nachfolger und Vorgänger einer Kante eindeutig bestimmt. Nachfolger einer Kante E ist diejenige Kante F mit $E.dest = F.orig$, mit der sie den größten Winkel $\sphericalangle(E.orig, F.orig, F.dest)$ in der Ebene bildet. Mit dieser Vereinbarung sind die einzelnen Kantenzüge als Komponenten der Front eindeutig bestimmt.

Formal liegt nun die *Menge aller Frontkanten* $\mathcal{C}$ vor. Die Partitionierung in die Komponenten der Front (die jeweiligen Kantenzüge) ist wegen der Eindeutigkeit der Nachfol-

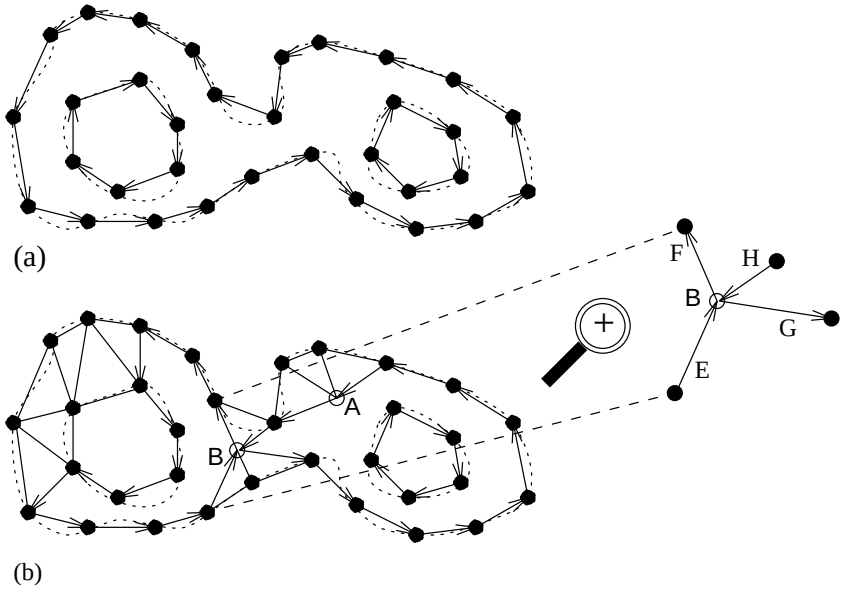


Abbildung 4.8: Advancing Front in einer mehrfach verbundenen Domain (a) Das System aus gerichteten Frontkanten approximiert die Domaingrenze. (b) Die Front, nachdem einige Dreiecke unter Zuhilfenahme von zwei Steiner-Punkten (ungefüllte Kreise) erzeugt wurden. Am Steiner-Punkt B liegen vier Kanten an. Jede Kante besitzt trotzdem einen eindeutigen Nachfolger: z.B. $succ(E) = F$, da

$$\sphericalangle(E.orig, F.orig, F.dest) = \max_{K \in \{F,G,H\}} (\sphericalangle(E.dest, K.orig, K.dest)).$$

gerelation einfach formulierbar. Dazu definieren wir die Relation **R**:

Definition 4.1 (Äquivalenzrelation R) *Seien $E, F \in \mathcal{C}$, es gilt $E\mathbf{R}F$, sofern es eine Folge von Kanten $K_i \in \mathcal{C}$, $i \in \{0, ..., k\}$ gibt, für die gilt: $\forall i < k : K_{i+1} = succ(K_i)$ und $E = K_0 \wedge F = K_k$.*

Aus den Voraussetzungen folgt unmittelbar, daß **R** reflexiv, transitiv und symmetrisch ist. Eine Komponente der Front ist damit definiert als die Äquivalenzklasse, die alle Bilder einer Kante $E \in \mathcal{C}$ aus der Komponente bezüglich der Relation **R** umfaßt. Die Komponente wird bezeichnet mit $\mathcal{C}(E)$, wobei E eine Kante aus der Komponente darstellt. $interior\,\mathcal{C}(E)$ bezeichnet die Schnittmenge aller Bereiche des Definitionsbereichs, die sich links von einer Kante aus $\mathcal{C}(E)$ befinden.

Jede einfach verbundenen Domain $\mathcal{D}$ wird durch eine Front $\mathcal{C}$ initialisiert, die nur aus einer äußeren Komponente $\mathcal{C}(E)$ besteht. Ist $\mathcal{D}$ mehrfach verbunden, dann besitzt es eine äußere Komponente $\mathcal{C}(E)$ und für jedes Loch eine weitere Komponente. Dabei gilt für jede Komponente zu einem Loch $\mathcal{C}(F)$, daß $\mathcal{C}(F) \subsetneq interior\,\mathcal{C}(E)$. $interior\,\mathcal{C}(E)$ der äußeren Komponente ist beschränkt (Kantenrichtung entgegen Uhrzeigersinn) und $interior\,\mathcal{C}(F)$ einer inneren Komponente ist unbeschränkt (Kantenrichtung im Uhrzeigersinn). Sowohl für die Domain zum Zeitpunkt der Initialisierung, als auch für die Subdomains, die durch Schrumpfen oder Aufspaltung der ursprünglichen Domain zur Laufzeit zustandekommen

und die ungemeshten Bereiche beschreiben, gelten die gleichen Formalismen.

Das Komponentensystem der Front formalisiert eine mehrfach verbundene Subdomain $\mathcal{S}(G)$ als Schnittmenge aller *interior* der Komponenten der Front.

Definition 4.2 (Subdomain $\mathcal{S}(G)$) *interior $\mathcal{C}(E)$ enthalte m Komponenten interior $\mathcal{C}(F_k)$, für $k = 1, 2, ..., m$ mit unbeschränkten interior, dann ist die Subdomain $\mathcal{S}(G)$, mit G aus der Menge aller Frontkanten $\mathcal{C}$, gegeben durch:*

$$\mathcal{S}(G) = interior\,\mathcal{C}(E) \bigcap_{k=1}^{m} interior\,\mathcal{C}(F_k)$$

Zur Laufzeit partitioniert der Algorithmus die Domain $\mathcal{D}$ in den ungemeshten Bereich $\bigcup_{E\in\mathcal{C}} \mathcal{S}(E)$ und den gemeshten Bereich $\mathcal{D} \Leftrightarrow \bigcup_{E\in\mathcal{C}} \mathcal{S}(E)$ mit der Menge der Frontkanten als Grenze zwischen beiden Bereichen.

Die Kontrollstruktur im AFT-Algorithmus. Nachdem die zentrale Datenstruktur der Front im vorherigen Abschnitt beschrieben wurde, wird nun die Kontrollstruktur eines AFT-Algorithmus vorgestellt. Der darauffolgende Abschnitt stellt die eingesetzten Funktionen in ihrer Minimalrealisierung vor. Sie stellt sicher, daß der Algorithmus terminiert und eine Triangulierung der Domain erzeugt.

Die Menge der Frontkanten $\mathcal{C}$ wird in Abb. 4.9 durch die Prozedur **CONVERT·TO·FRONTAL·EDGE·SET**($\mathcal{D}$; $\mathcal{C}$) aus der Eingabe-Domain $\mathcal{D}$ erzeugt. Der Kontrollparameter *fin* garantiert die Terminierung bei Erreichen einer Grenzbedingung. $\mathcal{T}$ bezeichnet die Triangulierung, die bei jedem Schleifendurchlauf um ein Dreieck erweitert wird.

Minimalentwurf der Prozeduren im AFT-Algorithmus. Nun folgen die Minimalanforderungen an jeden Einzelschritt zusammen mit einer kurzen Bemerkung.

1. **CONVERT·TO·FRONTAL·EDGE·SET**($\mathcal{D}$; $\mathcal{C}$)
   Minimalanforderung: Die Menge der Frontkanten wird als Datenstruktur $\mathcal{C}$ aus $\mathcal{D}$ generiert, so daß die Schnittmenge des *interior* der Komponenten die zu meshende Domain beschreibt.

2. **GET·NEXT·EDGE**($\mathcal{C}$; E)
   Minimalanforderung: Irgendeine Kante E aus $\mathcal{C}$ wird zurückgegeben. Diese stellt in diesem Schleifendurchlauf die aktuelle Kante dar.
   Bemerkung: Über der aktuellen Kante wird das nächste Dreieck generiert. Die Prozedur realisiert eine Auswahlstrategie für die Reihenfolge der Kanten, z.B. kürzeste oder längste Kanten zuerst oder zufällige Auswahl.

3. **COMPUTE·NEW·CANDIDATE**(E; R)
   Minimalanforderung: Die Routine gibt irgendeinen Punkt R aus dem Definitionsbereich zurück, der sich links von E befindet.
   Bemerkung: An dieser Stelle wird unabhängig von $\mathcal{C}$ und seiner Geometrie der Punkt R berechnet, der über der aktuellen Kante E ein ideales Dreieck errichtet. Es ist nicht sichergestellt, daß dieser Punkt von E aus sichtbar ist und

```
Procedure BASIC·AFT (D ; fin)
type   node                    R, S
       edge                    E
       logical                 New, New_is_preferred, Mesh_size_Constraint, Visible
       control parameter       fin
       boundary edge set       D
       frontal edge set        C
CONVERT·TO·FRONTAL·EDGE·SET (D ; C)
while C ≠ ∅
   GET·NEXT·EDGE (C ; E)
   COMPUTE·NEW·CANDIDATE (E ; R)
   COMPUTE·EXISTING·CANDIDATE (E, C ; S)
   New ← NEW·IS·PREFERRED (R ; S) and not MESH·SIZE·CONSTRAINTS (fin ; )
   if New
      then
         New ← VISIBLE (E, C, R ; )
   endif
   if New
      then
         UPDATE·INTERNAL (C, E, R, fin ; C, fin)
         ADD·TRIANGLE (T, E, R ; T)
      else
         UPDATE·BOUNDARY (C, E, S, fin ; C, fin)
         ADD·TRIANGLE (T, E, S ; T)
   endif
endwhile
```

Abbildung 4.9: Kontrollstruktur der Advancing Front Technique
aus [Far93]

das Dreieck aus E und R zu einer gültigen Triangulierung beiträgt, d.h. das Dreieck liegt vollständig in $\mathcal{S}(E)$. Wenn das berechnete ideale Dreieck zulässig ist, kann es zu einer Konstellation von $\mathcal{C}$ führen, die es in den nachfolgenden Schritten unmöglich macht, wohlgeformte Dreiecke in die ungemeshte Region zu setzen.
Die Kandidaten R bilden in der Anfangsphase des Algorithmus, in der die ungemeshte Region noch groß ist und die Geometrie der Domain kaum Beschränkungen auferlegt, große Bereiche mit wohlgeformter Netzstruktur aus.

4. **COMPUTE·EXISTING·CANDIDATE**$(E, \mathcal{C}\,;\, S)$
   Minimalanforderung: Ein Knotenpunkt S aus den Komponenten zur aktuellen Subdomain $\mathcal{S}(E)$ von $\mathcal{C}$, der zusätzlich von der aktuellen Kante E aus sichtbar ist, wird zurückgegeben.
   Bemerkung: Im Gegensatz zum neuen Kandidaten R, der zwar ein ideales Dreieck über E bildet, das aber nicht gültig sein muß und zu ungünstigen Kanten in $\mathcal{C}$ führen kann, stellt S einen sicheren Standardkandidaten für die Errich-

tung eines Dreiecks dar. Die Auswahl von S basiert auf der Geometrie von $\mathcal{C}$. Die Prozedur implementiert Verfahren, um aus der Menge der von E sichtbaren (Prozedur **VISIBLE**) Knotenpunkte einen Kandidaten auszusuchen, der außerdem ein zufriedenstellendes Dreieck erzeugt.
Typischerweise bietet sich ein Kandidat S an, wenn sich zwei Bereiche der Front nahe gegenüberstehen und verbunden werden sollten. In dieser Situation besteht kein Freiheitsgrad mehr für die Auswahl idealer Kandidaten zur Konstruktion des nächsten Dreiecks.

5. **NEW·IS·PREFERRED**(R, S; *logical*)
Minimalanforderung: Bei $R = S$ muß false zurückgegeben werden.
Bemerkung: Die Prozedur realisiert eine Entscheidungsstrategie zwischen R und S. Die Gesamtqualität des erzeugten Netzes hängt davon ab, bei welcher geometrischer Grenzbedingung von $\mathcal{C}$ der ideale Kandidat R verworfen wird und das Dreieck mit dem existierenden Kandidat S errichtet wird. Die Entscheidung für R ist unabhängig davon, ob R von E sichtbar ist. Die Prozedur **VISIBLE** prüft die Sichtbarkeit nur, wenn die Entscheidung für R fiel (S ist in jedem Fall sichtbar). Die Berechnung von **NEW·IS·PREFERRED** ist weniger aufwendig als **VISIBLE**, daher ist dieses Vorgehen effizienter, auch wenn eine Entscheidung für R eventuell zurückgenommen werden muß.
Falls die Prozedur konstant false zurückgibt, degeneriert die AFT zu einem einfachen Triangulierer der Domain $\mathcal{S}(E)$, der keine Steiner-Punkte setzt.

6. **MESH·SIZE·CONSTRAINTS**(*fin* ; *logical*)
Minimalanforderung: Die Prozedur zwingt den AFT-Algorithmus zur Terminierung bei Erreichen einer Grenzbedingung.
Bemerkung: In der Regel terminiert ein AFT-Algorithmus nach endlich vielen Schritten, sobald die Front verschwunden ist und die Domain damit vollständig trianguliert ist. Die Zahl der Schritte ist abhängig von der Fläche und Geometrie von $\mathcal{S}(E)$ und der Größe der Dreieckselemente.
MESH·SIZE·CONSTRAINTS implementiert zusätzlich eine direkte Kontrolle der Anzahl oder Größe der Dreiecke im Netz und führt den Algorithmus mit diesem Ziel zur Terminierung. Es wird entweder sichergestellt, daß die Triangulierung nicht mehr als N_{max} Dreiecke enthält oder, daß alle Dreiecke einen Mindestflächeninhalt besitzen.
Bei der ersten möglichen Formulierung von **MESH·SIZE·CONSTRAINTS** enthält das Feld *fin* die Information N_{max}, $N(\mathcal{T})$ und $N(\mathcal{C})$. Solange für die Summe aus der Zahl der Dreiecke in $\mathcal{T}$ und der Zahl der Frontkanten in $\mathcal{C}$ gilt: $N(\mathcal{T}) + N(\mathcal{C}) \leq N_{max}$ gibt die Prozedur true zurück, andernfalls false. Bleibt infolgedessen durch logische Verknüpfung New=false, dann wird die ungemeshte Region mit $N(\mathcal{C})$ Dreiecken trianguliert, die aus den Frontkanten und existierenden Knotenpunkten S konstruiert werden. Damit entsteht eine Gesamttriangulierung der Domain mit N_{max} Dreiecken.
Mit dieser Kontrollmethode ist eine exakte Vorausbestimmung der Anzahl der Dreiecke möglich. Die "Not-Triangulierung" der ungemeshten Region bei Eintreten der Bedingung $N(\mathcal{T}) + N(\mathcal{C}) = N_{max}$ bewirkt aber unter Umständen eine sehr inhomogene Gesamttriangulierung.
Im zweiten Fall enthält das Feld *fin* die Fläche des aktuellen Dreiecks und einen Wert für die Mindestfläche von Dreiecken im Netz. Bei Unterschreitung

der Mindestfläche gibt die Prozedur false zurück, und infolgedessen wird der ideale Kandidat R zugunsten des Kandidaten S aus $\mathcal{C}$ verworfen. In dieser Ausführung verhindert die Prozedur, daß eine adaptive Version der Prozedur **COMPUTE·NEW·CANDIDATE** unbeschränkt kleine Dreiecke erzeugt. Mit Einhaltung einer Mindestgröße bleibt die Anzahl der Dreiecke im Netz beschränkt, die exakte Anzahl der Dreiecke kann jedoch nicht vorhergesagt werden. An ein Netz mit gleichmäßiger Elementgröße *und* definierter Elementanzahl kann man sich nur empirisch annähern.

7. **VISIBLE**$(E, U, \mathcal{C}\,;\, logical)$
   Minimalanforderung: Die Prozedur gibt genau dann true zurück, wenn ein Knotenpunkt U von der Kante E sichtbar ist. Sichtbarkeit liegt vor, wenn sich das durch E und U gebildete Dreieck vollständig in $\mathcal{S}(E)$ befindet und außer mit E und U keine weiteren gemeinsamen Punkte mit $\mathcal{C}$ besitzt.
   Bemerkung: Mit Hilfe dieser Prozedur wird getestet, ob ein Idealkandidat $U = R$ mit E ein gültiges Dreieck erzeugt. Befinden sich Punkte einer Constraint-Kante im Dreieck, dann führt das Dreieck zu keiner gültigen Triangulierung.

8. **UPDATE·INTERNAL**$(\mathcal{C}, E, R, fin\,;\, \mathcal{C}, fin)$
   Minimalanforderung: Die Prozedur aktualisiert die Front $\mathcal{C}$ durch Einsetzen von zwei neuen Kanten F von $E.orig$ nach R und G von $E.dest$ nach R. E wird aus $\mathcal{C}$ und die Parameter in *fin* werden aktualisiert.
   Bemerkung: Die Subdomain $\mathcal{S}(E)$ schrumpft um die Fläche des neuen Dreiecks und die Zahl der Kanten in $\mathcal{C}$ hat sich um eins erhöht.

9. **UPDATE·BOUNDARY**$(\mathcal{C}, E, S, fin\,;\, \mathcal{C}, fin)$
   Minimalanforderung: Die Anforderungen stimmen mit **UPDATE·INTERNAL** überein. Lediglich wird R durch S ersetzt.
   Bemerkung: Die Aktualisierung der Front mit den Kanten F von $E.orig$ nach $S \in \mathcal{C}$ und G von $E.dest$ nach $S \in \mathcal{C}$ ist eine etwas kompliziertere Aufgabe als bei **UPDATE·INTERNAL** und umfaßt vier Fälle.
   Die vier Fälle sind die Kombinationen der beiden Aussagen $\pm F \in \mathcal{C}$ und $G \in \mathcal{C}$. Abbildungen 4.10 (a), (b) illustrieren den Fall, daß entweder nur $\Leftrightarrow F$ oder nur G Kanten aus $\mathcal{C}$ sind. Falls $S = succ(E).dest$ (äquivalente Formulierung zu $G \in \mathcal{C}$) werden G und E in $\mathcal{C}$ gelöscht und F wird eingefügt (a). Im Fall von $S = pred(E).orig$ (äquivalent zu $\Leftrightarrow F \in \mathcal{C}$) werden F und E in $\mathcal{C}$ gelöscht und G wird eingefügt (b). Sind F *und* G nicht Bestandteil der Front, dann wird die Subdomain in zwei neue Subdomains gespalten, von denen je eine F und eine G enthält. E wird gelöscht. Im letzten Fall ($F \in \mathcal{C}$ *und* $G \in \mathcal{C}$) bilden die Kanten E, F, G das letzte Dreieck der aktuellen Subdomain. Die Front verschwindet für diese Subdomain.

10. **ADD·TRIANGLE**$(\mathcal{T}, E, U\,;\, \mathcal{T})$
    Minimalanforderung: Das Dreieck, das vom Punkt U über der Kante E errichtet wird, wird der Triangulierung $\mathcal{T}$ hinzugefügt.

Qualitäts-Meshing mit dem AFT-Algorithmus Ein guter Triangulierungsalgorithmus sollte effizient sein und mit einer kontrollierbaren Anzahl von Steiner-Punkten eine

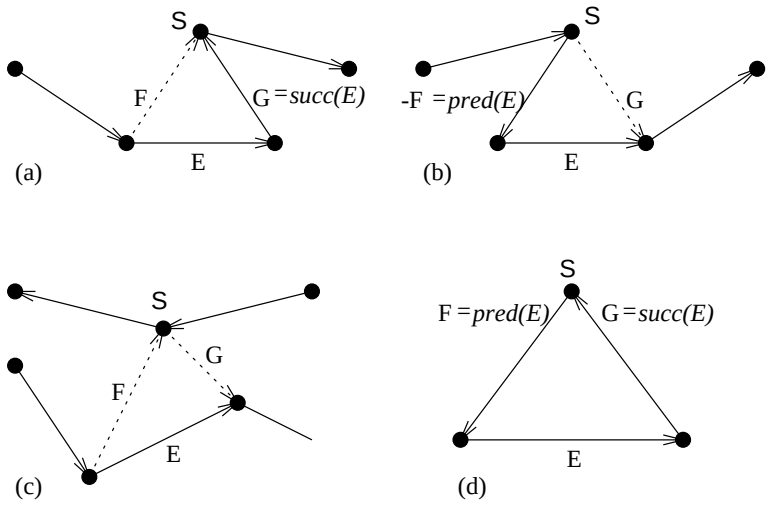


Abbildung 4.10: Vier Fälle für **UPDATE·BOUNDARY**
Eine der beiden Kanten F, G ist Kante der Front (a), (b). F, G gehören beide nicht zur Front. Die aktuelle Subdomain wird in zwei Subdomains aufgespalten (c). (d) zeigt den finalen Schritt, der zum Verschwinden der Subdomain führt. F und G sind mit E die einzigen Kanten der Front.

global optimale Triangulierung erzeugen. Dieser Abschnitt stellt eine Erweiterung des Minimalentwurfs vor, mit deren Hilfe der AFT-Algorithmus diese Ziele erfüllen kann. Die Modifikationen basieren auf der Verwendung der CDT als Datenstruktur zur Repräsentation der Domain.

Eine effiziente Datenstruktur für die Menge der Frontkanten $\mathcal{C}$ muß sowohl die Verwaltungsoperationen Einfügen und Löschen von Kanten, als auch Suchoperationen über die Knotenpunkte der Kanten (in **COMPUTE·EXISTING·CANDIDATE**) und Sichtbarkeitstests (in **VISIBLE**) unterstützen. Die letzten beiden Operationen machen eine Analyse der Geometrie der Subdomain $\mathcal{S}(E)$ notwendig und sind daher besonders zeitkritisch.
Da **COMPUTE·NEW·CANDIDATE** einen Kandidaten für ein ideales Dreieck berechnet, liegt die Ursache für unzulänglich geformte Dreiecke in den Prozeduren **COMPUTE·EXISTING·CANDIDATE** und der Entscheidungsfunktion **NEW·IS·PREFERRED**. Eine Datenstruktur, die die Erzeugung global optimaler Netze ermöglicht, muß also die Prozedur **COMPUTE·EXISTING·CANDIDATE** bei der Berechnung eines Kandidaten aus $\mathcal{C}$ unterstützen, damit ein wohlgeformtes Dreieck mit der aktuellen Kante erzeugt wird. Die Datenstruktur soll weiterhin die theoretische Grundlage für die Entscheidungsfunktion **NEW·IS·PREFERRED** bereitstellen.
Die Repräsentation von $\mathcal{S}(E)$ mit Hilfe der Constrained-Delaunay-Triangulierung (S. 29) kann diese Anforderungen erfüllen. In die CDT-Triangulierung gehen die Kanten aus $\mathcal{C}$ als Constraints ein.

Wie die Bemerkungen zu den modifizierten AFT-Prozeduren im Folgenden erläutern werden, verhindert die Datenstruktur $\mathsf{CDT}(\mathcal{S}(E))$ (Abb. 4.11) die Entstehung von Dreiecken mit spitzen Winkeln. Damit wird allerdings noch keine globale max-min-winkelop-

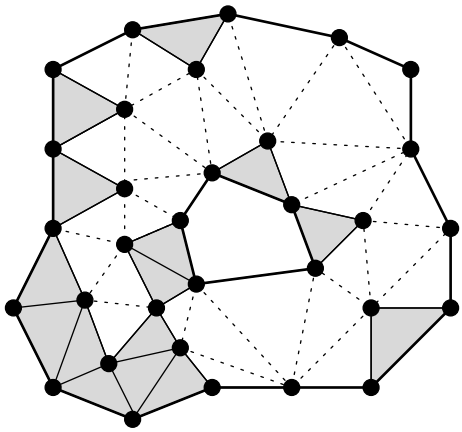


Abbildung 4.11: Repräsentation von $\mathcal{S}(E)$ mit Constrained-Delaunay-Triangulierung
Die äußere und die innere Domainbegrenzung sind als breitere Linie gezeichnet. Die bereits konstruierten Dreiecke sind schattiert dargestellt, und die CDT-Triangulierung der ungemeshten Subdomain ist gestrichelt sichtbar.

timierte Triangulierung unter Berücksichtigung der Domainbegrenzungen erzeugt. Jedes neue Dreieck wird über seiner Kante aus $\mathcal{C}$ konstruiert und zur Triangulierung $\mathcal{T}$ hinzugefügt. Dadurch bleibt auch nach der Aktualisierung von $\mathcal{C}$ jede ehemalige Frontkante in der Triangulierung des gemeshten Bereichs $\mathcal{D} \Leftrightarrow \mathcal{S}(E)$ erhalten. Die entstandene Triangulierung berücksichtigt faktisch immer noch jede ehemalige Frontkante als Constraint. Für die global optimale CDT der Punktmenge und Domainbegrenzung muß neu trianguliert werden. Alternativ kann man analog zum Vorgehen für den ungemeshten Bereich $\mathsf{CDT}(\mathcal{S}(E))$ ebenfalls zur Laufzeit den gemeshten Bereich mit Hilfe seiner CDT repräsentieren. $\mathsf{CDT}(\mathcal{S}(E))$ und $\mathsf{CDT}(\mathcal{D} \Leftrightarrow \mathcal{S}(E))$ bilden somit zu jedem Zeitpunkt gemeinsam die CDT aller Knotenpunkte, Begrenzungen und Frontkanten der Domain. Die modifizierte Version von **ADD·TRIANGLE** realisiert diesen Ansatz.

1. **CONVERT·TO·FRONTAL·EDGE·SET**$(\mathcal{D}\,;\,\mathcal{C})$
   Bemerkung: Als Datenstruktur für das Frontkantensystem $\mathcal{C}$ kommt $\mathsf{CDT}(\mathcal{S}(E))$ zum Einsatz. Die Prozedur erzeugt die CDT und weist sie $\mathcal{C}$ zu.

2. **GET·NEXT·EDGE**$(\mathcal{C}\,;\,E)$
   Bemerkung: Die Selektionsstrategie für die nächste Kante hat Einfluß auf die Netzqualität. In der Literatur wird eine Strategie favorisiert, bei der die kürzeste Frontkante als nächste aktuelle Kante gewählt wird [Far93]. Optimierte Datenstrukturen für die Sortierung und den schnellen Zugriff auf die Frontkanten steigern die Effizienz von **GET·NEXT·EDGE**.

3. **COMPUTE·NEW·CANDIDATE**$(E\,;\,R)$
   Bemerkung: R sollte mit E ein gleichseitiges Dreieck im ungemeshten Bereich der Domain, d.h. links von der gerichteten Kante E ergeben. Die Seitenlänge beträgt *unit_size*. AFT-Algorithmen, die die Elementgröße an den geometrischen Detaillierungsgrad der Umgebung anpassen, wählen die *unit_size*

abhängig vom *geometrischen Maßstab* der Umgebung. Der geometrische Maßstab ist ein lokales Maß, das abhängig ist von der Nähe und Krümmung der Domainbegrenzung oder von der Oberflächenkrümmung (Netzgenerierung auf Oberflächen).
E kann die doppelte *unit_size* und mehr besitzen. In einem solchen Fall wird kein neues Dreieck generiert, sondern E in zwei Kanten gespalten. Der neue Knotenpunkt wird in die Mitte von E gesetzt.

4. **COMPUTE·EXISTING·CANDIDATE**$(E, \mathcal{C}\,;\, S)$
Bemerkung: Wähle S als denjenigen Knotenpunkt aus $\mathcal{C}$, der in der $\mathsf{CDT}(\mathcal{S}(E))$ mit der Kante E ein leeres Dreieck bildet. Die Datenstruktur $\mathsf{CDT}(\mathcal{S}(E))$ bietet konstanten Zugriff auf den existierenden Kandidaten, mit dem E ein CDT-optimales Dreieck bildet. In Kombination mit dem Diagonalentausch in **ADD·TRIANGLE** wird garantiert, daß der AFT-Algorithmus die CDT der Eingabe-Domain erzeugt.

5. **NEW·IS·PREFERRED**$(R, S\,;\, \mathit{logical})$
Bemerkung: Für die Entscheidung zwischen R und S wird überprüft, ob sich das neue Dreieck aus E und R in der Delaunay-Triangulierung von $\mathcal{S}(E) \cap R$ befindet. Dieses Kriterium fördert die Netzqualität und ermöglicht in **UPDATE-INTERNAL** ein effizientes Verfahren zur Neubestimmung von $\mathsf{CDT}(\mathcal{S}(E))$ bei aktualisierter Front $\mathcal{C}$, nachdem das Dreieck aus E und R der Triangulierung hinzugefügt wurde.
Der Test, ob $\triangle(E, R) \in \mathsf{CDT}(\mathcal{S}(E) \cup R)$ wird durch Ausnutzung des folgenden Lemmas vereinfacht.

Lemma 4.1 *$\circ(E, R)$ bezeichne den Umkreis von $\triangle(E, R)$. Wenn R von E aus sichtbar ist, $\triangle(E, S) \in \mathsf{CDT}(\mathcal{S}(E))$ und E, R, S nicht auf einem Umkreis liegen, dann gilt: $\triangle(E, R) \in \mathsf{CDT}(\mathcal{S}(E) \cup R) \Leftrightarrow S \notin \circ(E, R)$.*

Der Beweis befindet sich in [Far93].
Die Abbildungen 4.12 (a),(b) und (c) zeigen die Kante E und den existierenden Kandidaten S. Das Dreieck aus E und S ist Bestandteil von $\mathsf{CDT}(\mathcal{S}(E))$. Der eingezeichnete Umkreis enthält nur dann weitere Netzknotenpunkte, wenn diese durch eine Constraint-Kante von den Dreiecksknotenpunkten separiert sind (CDT-Umkreiskriterium).
In Abb. 4.12 (a) fällt die Entscheidung zugunsten des neuen Kandidaten R, da das Dreieck aus E und R das CDT-Umkreiskriterium erfüllt. Im abgebildeten Beispiel hat **COMPUTE·NEW·CANDIDATE** für jede Dreieckseite $(E.dest, R)$ und $(E.orig, R)$ eine andere *unit_size* berechnet. Das Dreieck wird hier aus Rücksichtnahme auf geometrische Bedingungen in der Nachbarschaft von E und R nicht gleichseitig sein.
In Abb. 4.12 (c) enthält der Umkreis um R, $E.orig$ und $E.dest$ den Knotenpunkt S. Das CDT-Umkreiskriterium ist verletzt, da es keine Constraint-Kante geben kann, die R, $E.orig$ und $E.dest$ von S separiert. Eine solche Constraint-Kante hätte die Existenz des Dreiecks aus $E.orig$, $E.dest$ und S in $\mathsf{CDT}(\mathcal{S}(E))$ verhindert.
Die Umkreisbedingung wird überprüft, indem die relative Position der Umkreismittelpunkte auf der Mittelsenkrechten über E verglichen wird. Dazu sei

als positive Richtung dieser Zahlenachse die Richtung angenommen, die rechts von E weist. $\overline{R}$ bezeichnet die senkrechte Projektion von R auf die Mittelsenkrechte. Die Entscheidungen aus Abb. 4.12 (a),(c) erfüllen eine der folgenden Bedingungen:

- Wähle R, falls $S_c < \overline{R}$.
- Wähle S, falls $S_c > R_c$.

Falls sich S_c im Intervall $[\overline{R}, R_c]$ befindet, kann sich eine bessere Netzqualität mit feiner abgestuften Dreiecken ausbilden, wenn nicht einfach R gewählt wird, wie es das Umkreiskriterium nahelegt, sondern eine weitere Fallunterscheidung unternommen wird. Dabei stellt S_c selbst einen möglichen Alternativkandidaten zu R und S dar.

- Wähle R, falls $\overline{R} < S_c < L_1$.
- Wähle S_c, falls $L_1 < S_c < L_2$.
- Wähle S, falls $L_2 < S_c$.

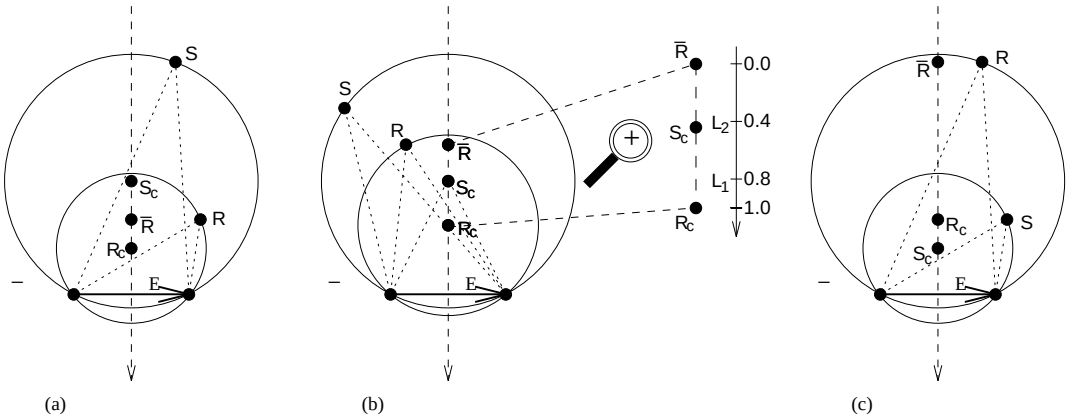


Abbildung 4.12: Fallunterscheidung in **NEW·IS·PREFERRED**
(a) $S_c < \overline{R}$ bedeuted die klare Entscheidung für R, (b) Entscheidung für R, S_c oder S ist abhängig von der Position von S_c im Intervall $[\overline{R}, R_c]$, (c) $S_c > R_c$ führt zur Wahl von S.

6. **MESH·SIZE·CONSTRAINTS**(*fin* ; *logical*)
   Bemerkung: Der bisherige Entwurf für **MESH·SIZE·CONSTRAINTS** hat weiterhin Gültigkeit. Für Versionen von **COMPUTE·NEW·CANDIDATE**, die mit adaptiver *unit_size* arbeiten, stellt die Begrenzung der minimalen Dreiecksfläche die Terminierung des Algorithmus sicher.

7. **CHECK·VISIBLE**($E, U, \mathcal{C}$; *logical*, $\mathcal{T}_R$)
   Bemerkung: Die Prozedur **VISIBLE** berechnet die zusätzliche Information $\mathcal{T}_R$, mit deren Hilfe **UPDATE·INTERNAL** effizient $\mathsf{CDT}(\mathcal{S}(E))$ aktualisieren kann. $\mathcal{T}_R$ stellt die Menge aller Dreiecke in $\mathsf{CDT}(\mathcal{S}(E))$ dar, die eine Schnittkante mit einer der potentiellen Kanten $(E.orig, R)$ und $(E.dest, R)$ besitzen. Nach CLINE und RENKA (siehe S. 35) genügt es zur Neubestimmung von $\mathsf{CDT}(\mathcal{S}(E))$ nach dem Einsetzen von R, die CDT der Untermenge $\mathcal{T}_R$ einschließlich R zu bestimmen und diese in $\mathcal{T}$ einzusetzen (Abb. 4.13).

8. **UPDATE·INTERNAL**$(\mathcal{C}, E, R, \mathcal{T}_R, \mathit{fin}\,;\, \mathcal{C}, \mathit{fin})$
   Bemerkung: Es bezeichne $\mathcal{B}_R$ den begrenzenden Kantenzug zu $\bigcup \mathcal{T}_R$, dann genügt: $\mathsf{CDT}(\mathcal{S}(E)) = \mathsf{CDT}(\mathcal{S}(E) \Leftrightarrow \mathcal{T}_R) \cup \mathsf{CDT}(\mathcal{B}_R \cup F \cup G)$. F, G sind die neuen Kanten $(E.\mathit{orig}, R)$ und $(E.\mathit{dest}, R)$.

9. **UPDATE·BOUNDARY**$(\mathcal{C}, E, S, \mathit{fin}\,;\, \mathcal{C}, \mathit{fin})$
   Bemerkung: Zur Neubestimmung von $\mathsf{CDT}(\mathcal{S}(E))$ ist keine Retriangulierung notwendig, da das Dreieck aus E und S bereits in der alten Triangulierung enthalten war. Die alten Triangulierungsdaten können in den drei Fällen, in denen die Subdomain nicht verschwindet, weiterverwendet werden. Das neue Dreieck wird der Triangulierung entnommen.

10. **ADD·TRIANGLE**$(\mathcal{T}, E, U\,;\, \mathcal{T})$
    Bemerkung: Wie bereits auf S. 53 erwähnt, genügt es nicht, das neue Dreieck $\triangle(E, R)$ oder $\triangle(E, S)$ der Triangulierung $\mathcal{T}$ hinzuzufügen, um mit dem AFT-Algorithmus die CDT der Domain zu erhalten.
    Nach der Aktualisierung von $\mathcal{C}$ ist E keine Constraint-Kante mehr, und alle nun sichtbaren Knotenpunkte im gemeshten Bereich müssen beim Umkreiskriterium zum neuen Dreieck $\triangle(E, U)$ $(U = R \vee U = S)$ Berücksichtigung finden.
    Zusätzlich muß in der gemeshten Region der Fall berücksichtigt werden, daß kein neues Dreieck hinzukommt, sondern die Kante E nur gespalten wurde.
    Die Retriangulierung von $\mathsf{CDT}(\mathcal{S}(E))$ nach Aktualisierung der Front kann in **UPDATE·INTERNAL** effizient realisiert werden, weil **CHECK·VISIBLE** die kleine Untermenge von $\mathsf{CDT}(\mathcal{S}(E))$ bestimmt, die retrianguliert und in $\mathsf{CDT}(\mathcal{S}(E))$ eingesetzt werden muß. Für die Aktualisierung von $\mathcal{T}$ nach jedem Schleifendurchlauf ist das Verfahren aus **CHECK·VISIBLE** zur Bestimmung der betroffenen Untermenge nicht anwendbar. Da die ständige Retriangulierung der gesamten Menge $\mathcal{T} \cup \triangle(E, U)$ rechenaufwendig ($O(n \log n)$, siehe Abschnitt 4.2.1) und für den Verlauf des Algorithmus nicht relevant ist, empfiehlt sich die Retriangulierung der Knotenpunkte nach dem Verschwinden der Front. Die schrumpfende Front hinterläßt in der Domain zur Laufzeit die Steiner-Punkte in kontrollierter Dichte. Zum Schluß werden diese Punkte unter Berücksichtigung der Constraints optimal trianguliert. Das Ergebnis ist identisch.

Die vorgestellte AFT-Methode produziert global optimale Triangulierungen nach dem Max-Min-Winkelkriterium. Gekoppelt mit einer Routine zur Berechnung der Initialknoten auf der Domain-Begrenzung stellt sie ein vollautomatisches Verfahren zur Netzgenerierung dar. Sie erlaubt beliebig komplizierte Geometrien der Domain. Nicht-isotropische Netze, d.h. Dreiecksnetze mit variierender Kantenlänge können erzeugt werden, wenn sich die *unit_size* an den lokalen geometrischen Maßstab der Domain anpaßt. Damit können sehr fein abgestufte Netze erzeugt werden, die sich an beliebig kleine geometrische Details an der Domain-Begrenzung oder auf einer 3D-Oberfläche anpassen. Mit der AFT-Methode hergestellt Netze bilden durch die schichtweise Generierung ihrer Dreieckselemente entlang der Domain-Begrenzungen eine Struktur mit Flußrichtung aus. AFT-Algorithmen erreichen ein Laufzeitverhalten von $O(n)$ (n ist die Anzahl der Dreiecke).

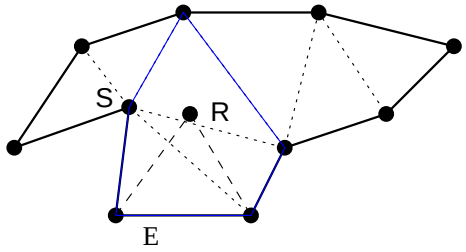


Abbildung 4.13: Die Untermenge $\mathcal{T}_R$ aus $\mathcal{S}(E)$
Die blau eingezeichnete Linie beschreibt die Begrenzung $\mathcal{B}_R$ der Menge $\mathcal{T}_R$. Nach Einsetzen von R muß nur die Untermenge $\mathcal{T}_R$ zur Aktualisierung von $\mathcal{S}(E)$ retrianguliert werden.

Quadtree-Triangulierung

Quadtrees stellen eine effiziente Methode zur topologischen Dekomposition einer Domain dar. Mit ihrer Hilfe wird die Domain in feiner aufgelöste Bereiche in der Nähe der komplizierten Domain-Begrenzung und in großflächige Bereiche aufgeteilt. Die Größe der Netzelemente wird bei der Triangulierung in Abhängigkeit von der ermittelten Auflösung gesteuert.

In [BE92] wird ein Quadtree-Triangulierungsalgorithmus vorgestellt, der die Quadtree-Dekomposition gleichzeitig zur kontrollierten Erzeugung der Netzelemente nutzt. Für die Form der Dreiecke kann die Garantie gegeben werden, daß kein Dreieckswinkel kleiner als 20° sein wird.

Der Algorithmus dekomponiert die Domain mit Hilfe balancierter Quadtrees.

Definition 4.3 (Quadtree) *Ein Quadtree ist eine rekursive Partition einer quadratischen Region in Quadrate, die entlang der Koordinatenachsen ausgerichtet sind. Die* **Wurzel** *ist das Ursprungsquadrat und umfaßt die gesamte Region. Ein horizontales und ein vertikales Liniensegment durch den Mittelpunkt teilt das Wurzelquadrat in vier* **Nachkommen**. *Die Gesamtheit der Quadrate stellt einen Baum dar, bei dem die Größe der Quadrate mit der Tiefe des Baumes abnimmt [BE92].*

Definition 4.4 (Balancierter Quadtree) *Jede Seite eines Quadrates, das im Baum ein Blatt darstellt, wird durch benachbarte Quadrate höchstens halbiert (siehe Abbildung 4.14).*

Der Domain wird ein Quadtree überlagert, das solange verfeinert wird, bis jeder Knotenpunkt aus dem begrenzenden Kantenzug der Domain von allen anderen Punkten *wohlsepariert* ist. Ein Punkt gilt als wohlsepariert, wenn er sich im Zentrum eines fünf mal fünf Elemente großen Gitters aus gleichgroßen Quadraten befindet und kein anderer Punkt in diesem Teilgitter liegt (Abb. 4.15). Im nächsten Schritt werden die Kanten in jedem Teilgitter so verschoben, daß jeder Punkt zum Knotenpunkt im Viereckснetz wird *(warping)*. Die Knotenpunkte des Viereckgitters bilden die Menge der Steiner-Punkte.

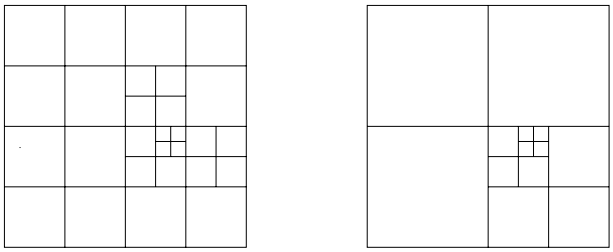


Abbildung 4.14: Balancierter und nicht balancierter Quadtree
Beim balancierten Quadtree hat jedes Quadrat nur mit solchen Quadraten Kanten gemeinsam, die mindestens seine halbe Größe besitzen (aus [BE92]).

Das Viereckснetz wird nun trianguliert. Aufgrund des balancierten Quadtree besitzt jedes nicht verschobene - also quadratische - Viereck höchstens eine Unterteilung an jeder Kante. Diese geometrische Struktur ist leicht durch neu eingefügte Kanten triangulierbar. Kein Winkel wird dabei kleiner als $arctan(1/2) \approx 26,5°$. Die verschobenen Dreiecke werden durch das Einfügen einer Diagonalen trianguliert. Die untere Schranke für die entstehenden Dreieckswinkel liegt bei 20°.

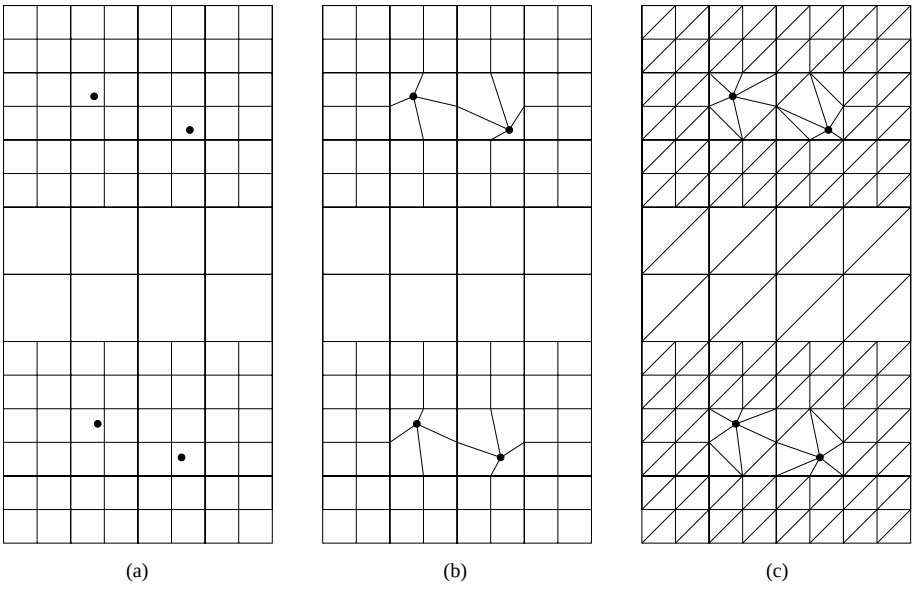


Abbildung 4.15: Quadtree-Triangulierung von vier Punkten
(a) Dekomposition bis jeder Knotenpunkt wohlsepariert ist, (b) warping, (c) Triangulierung

Zusätzlich zur quantitiven Aussage über die Winkelgrößen läßt sich die Anzahl der entstehenden Dreiecke abschätzen. In [BE92] finden sich Lemma und Beweis, daß der Algorithmus $O(n \log A)$ Dreiecke produziert, wobei A das maximale Verhältnis zwischen Grundfläche und Höhe in einem Dreieck der DT der Eingabepunkte bezeichnet.

Der Quadtree-Algorithmus vergrößert die Eingabe-Domain zu einer rechteckigen Domain. Um die ursprüngliche Domain wieder herzustellen, ist es notwendig, den begrenzenden Kantenzug der Eingabe-Domain nachzuziehen. Alle Dreiecke außerhalb des Kantenzugs werden gelöscht. Bei Dreiecken, die von der Domain-Grenze geschnitten werden, werden alle Knotenpunkte außerhalb der Domain gelöscht. Die Schnittpunkte der ursprünglichen Dreieckskanten mit der Domain-Grenze werden als neue Steiner-Punkte aufgenommen. Bilden die verbliebenen Knotenpunkte mit diesen Schnittpunkten ein Viereck, so wird dieses Viereck mit einer Diagonalen trianguliert. Entsteht dagegen wieder ein Dreieck, so wird es direkt übernommen.

Die Verallgemeinerung der Quadtree-Triangulierung für gekrümmte Oberflächen kann nur für nichtgefaltete Domains gelingen. Die quadratischen Elemente des Quadtree werden zu vierseitigen Patches verallgemeinert. Jedes Patch wird weiter unterteilt, falls es die gekrümmte Oberfläche nicht mit einer bestimmten Toleranz approximiert. In der Übersichtsliteratur [BE92], [HL88] finden sich jedoch keine Hinweise auf derartige Methoden, was in der mangelhaften Eignung von Viereckstrukturen, komplizierte Domains darzustellen, begründet sein mag.

Wie bei allen Methoden, die auf Viereckgittern basieren, entstehen an der Domain-Begrenzung häufig entartete Dreiecke mit spitzen Winkeln. Die Aussagen zur Qualität der Dreiecke lassen sich an der Domain-Begrenzung durch den Nachbearbeitungsschritt nicht aufrecht erhalten.
Insgesamt diktiert die Viereckstruktur die Gestalt des Netzes. Im Netz herrschen horizontale und vertikale Linien vor, und die meisten Dreiecke sind rechtwinkelig. Im ungünstigen Fall, daß sich ein Knotenpunkt der Domain-Begrenzung sehr nah, aber nicht auf einer Kante eines frühen Quadrats der Quadtree-Struktur befindet, wird eine hohe Schachtelungstiefe notwendig sein, um dem Knotenpunkt sein fünf mal fünf Quadrate umfassendes Teilgitter zur Verfügung zu stellen. Unnötig feine und viele Dreiecke werden die Folge sein.

CHEWs Algorithmus

CHEWs Algorithmus verwendet eine Generalisierung des Delaunay-Umkreiskriteriums für Oberflächennetze und erlaubt die Generierung von Oberflächendreiecksnetzen, die gleichzeitig folgende Eigenschaften besitzen [Che93]:

- Die Dreieckselemente erfüllen das Delaunay-Umkreiskriterium für gekrümmte Oberflächen.
- Die Triangulierung berücksichtigt initiale Knotenpunkte, sowie zwingende begrenzende und innere Kantenzüge. Damit stellt sie eine Constrained-Delaunay-Triangulierung bezüglich des modifizierten Umkreiskriteriums dar.
- Die Winkel der Dreiecke besitzen garantierte Größen zwischen 30° und 120°, sofern die zwingenden Kantenzüge keine anderen Winkelgrößen vorschreiben.
- Die maximale Größe der Dreiecke ist über eine Kontrollfunktion steuerbar. Die Kontrollfunktion ist über der Domain beliebig definierbar und kann beispielsweise die

gute Approximation der Oberfläche sicherstellen. Dazu wird der Wert für die obere Schranke der Dreiecksgröße vom Approximationsfehler abhängig gemacht.

- Der Algorithmus benötigt zur Erstellung einer Triangulierung, die die genannten Eigenschaften erfüllt, eine geringe Anzahl Steiner-Punkte, die sich maximal um einen kleinen konstanten Faktor von der optimalen Anzahl unterscheidet.

Der Algorithmus beginnt mit der initialen CDT der Eingabemenge (Knotenpunkte, äußere Begrenzung, innere Begrenzungen für Löcher, innere zwingende Kantenzüge). Anschließend werden alle Dreiecke ermittelt, die Winkelgrößen unter 30° aufweisen (Dreiecke an der äußeren Begrenzung ausgeschlossen) oder deren Größe die Vorgabe der Kontrollfunktion überschreiten. Die nicht-idealen Dreiecke werden ihrer Größe (Umkreisradius) entsprechend sortiert und in dieser Reihenfolge bearbeitet. Sofern die Knotenpunkte der betrachteten Dreiecke nicht durch eine zwingende Kante vom Umkreismittelpunkt getrennt sind, wird der Umkreismittelpunkt als neuer Knotenpunkt aufgenommen und die CDT aktualisiert. Andernfalls wird die zwingende Kante halbiert oder gedrittelt. Bevor die CDT aktualisiert wird, wird in der Umgebung mit Radius l um die neuen Knotenpunkte sichergestellt, daß keine sehr kurzen Kanten entstehen können. l ist die Länge der entstandenen zwei oder drei Kantensegmente. In der Umgebung werden alle in früheren Bearbeitungsschritten gesetzten Umkreisknotenpunkte gelöscht. Erst jetzt wird die CDT aktualisiert. Die ursprüngliche zwingende Kante wird in Form der zwei oder drei neu entstandenen zwingenden Kanten weiterhin respektiert. Der Algorithmus fährt mit der Bearbeitung des nächsten nicht-idealen Dreiecks fort oder terminiert, falls alle Dreiecke bearbeitet sind.

Die Formulierung des Umkreiskriteriums für gekrümmte Oberflächen wird in zweifacher Hinsicht eingesetzt. Erstens liegt zum Zeitpunkt der Initialisierung und nach jedem Schleifendurchlauf die CDT der aktuellen Menge aus Eingabepunkten und -kanten und Steiner-Punkten vor. Zweitens entstehen Steiner-Punkte aus den Umkreismittelpunkten nicht-idealer Delaunay-Dreiecke.

Eine mögliche Definition für einen Kreis ist mit Hilfe der geodätischen Entfernung (bzw. Entfernung in Bogenmaß) aufstellbar. Ein Kreis um einen Oberflächenpunkt ist die Menge aller Oberflächenpunkte mit gleicher, minimaler geodätischer Weglänge zum Mittelpunkt.
Diese Definition hat zwei Nachteile. Zuerst ist es für die meisten Oberflächen sehr aufwendig, den geodätischen Abstand zu berechnen. Dann können degenerierte Sonderfälle auftreten. Befindet sich beispielsweise ein Kreismittelpunkt in der Nähe einer steilen Erhebung, so kann der Kreis um die Erhebung herum reichen und sich selbst berühren, ohne die Spitze der Erhebung einzuschließen.

Eine alternative Kreisdefinition schützt zwar nicht vollständig vor degenerierten Sonderfällen, aber ist einfacher zu berechnen.

Definition 4.5 (Umkreis von drei Oberflächenpunkten) *Drei gegebene Punkte einer Oberfläche definieren eine unendliche Menge Kugelflächen, von denen jede die drei Punkte enthält. Die Kugelmittelpunkte befinden sich auf einer Geraden. Wähle diejenige Kugel, deren Mittelpunkt in der Oberfläche enthalten ist. Der Umkreis um diesen Mittelpunkt ist nun definiert als die Menge aller Schnittpunkte der Kugelfläche mit der Oberfläche.*

Diese Definition besitzt vier günstige Eigenschaften:

- Die Berechnung des kartesischen Abstands eines beliebigen Punktes der Oberfläche zum Kugelmittelpunkt genügt, um zu entscheiden, ob der Punkt innerhalb des Umkreises liegt. Diese Operation ist fundamental für die Konstruktion von Delaunay-Triangulierungen.
- Der Umkreismittelpunkt ist immer ein Punkt der Oberfläche und eignet sich damit als Knotenpunkt des oberflächenapproximierenden Dreiecksnetzes.
- Der Umkreismittelpunkt auf der Oberfläche ist durch iterative Verfahren relativ einfach bestimmbar. Gesucht wird der Schnittpunkt der Geraden aus den Kugelmittelpunkten und der Oberfläche. Falls die Oberfläche nicht zu stark gekrümmt ist, existiert genau ein Schnittpunkt.
- Diese Formulierung des Umkreises führt zu einem *konsistenten Umkreiskriterium*, falls die Flächennormalen benachbarter Dreiecke höchstens einen Winkel der Größe $\frac{\pi}{2}$ bilden. Für zwei Dreiecke mit gemeinsamer Kante gilt, daß das Umkreiskriterium für das erste Dreieck genau dann erfüllt ist, wenn es für das zweite Dreieck erfüllt ist.

Sofern für jedes Dreieck in der CDT der Eingabemenge der gemeinsame Winkel der Flächennormalen mit jedem benachbarten Dreieck mindestens $\frac{\pi}{2}$ beträgt, garantiert der Algorithmus die zu Beginn genannten Eigenschaften.

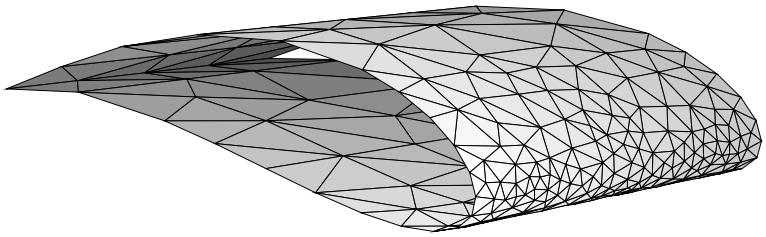


Abbildung 4.16: Mit CHEWs Algorithmus erzeugte Oberflächentriangulierung
Das Oberflächennetz für ein Flügelprofil wurde mit einer krümmungsabhängigen Kontrollfunktion erzeugt (aus [BE92]).

4.4 Netzverbessernde und datenreduzierende Methoden

Die netzverbessernden und datenreduzierenden Methoden arbeiten mit gegebenen Polygonnetzen. Mit der ersten Klasse von Methoden werden Netze verfeinert, die zweite Klasse macht Netze gröber. Die Netzverbesserung *(mesh refinement, mesh improvement)* wird als Nachbearbeitungsschritt heuristischer Netzgenerierungsalgorithmen eingesetzt und manipuliert die Position oder Verbindungskanten der Knotenpunkte mit dem Ziel, die Wohlgeformtheit der polygonalen Netzelemente zu erhöhen. Datenreduzierende Methoden *(mesh optimization)* werden zur Bearbeitung von Oberflächennetzen verwendet, die aus 3D-Scans

von Objekten (siehe S. 8) gewonnen wurden. In überabgetasteten Bereichen des Objekts nehmen sie gezieltes Löschen und Repositionieren von Knotenpunkten und Kanten vor, damit das Polygonnetz bei reduzierter Knotenpunktzahl das Ausgangsnetz mit definierter Fehlertoleranz approximiert.

4.4.1 Netzverbesserung durch Laplace-Glättung

Die Laplace-Glättung ist die bekannteste Methode zur Verbesserung der Netzqualität. Ihre Formel zu Repositionierung der Knotenpunkte läßt sich aus einer Approximation der Laplace-Gleichung mit Hilfe finiter Differenzen ableiten [BE92]. Die Methode wird wiederholt auf alle inneren Knotenpunkte des Netzes angewendet, bis das Netz die gewünschte Form erhalten hat. Die Position jedes internen Knotenpunktes wird dabei zum Zentrum des Polygons aus den benachbarten Knotenpunkten verlegt (Abb. 4.17 (b)). Der Polygonmittelpunkt wird durch das Mittel der Koordinaten der umgebenden Punkte berechnet. In einer Variation werden die Punktkoordinaten abhängig von der Fläche der umgebenden Netzelemente gewichtet.

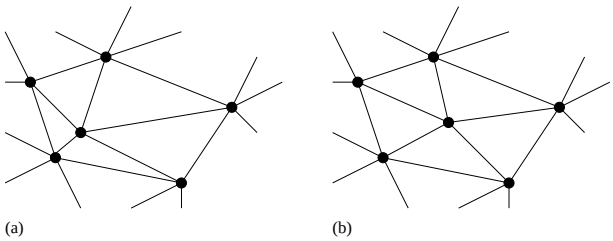


Abbildung 4.17: Netzverbesserung durch Laplace-Glättung
(a) ursprüngliche Position, (b) in der Laplace-Glättung erhält der Punkt die mittleren Koordinaten seiner Umgebungspunkte (nach [BE92]).

Ähnlich wie CHEWs Formulierung des Delaunay-Umkreiskriteriums für Oberflächen ist eine Formulierung der Laplace-Glättung für 2.5D-Netze denkbar. Der berechnete Mittelpunkt des polygonalen Kantenzuges (die benachbarten Knotenpunkte liegen nur im Sonderfall in einer Ebene) liegt in einer konvexen Umgebung unterhalb und in einer konkaven Umgebung oberhalb der Oberfläche und kann mit einem geeigneten Projektionsverfahren auf die Oberfläche abgebildet werden. Als mögliche Projektionsrichtung bietet sich die mittlere Oberflächennormale an den benachbarten Knotenpunkten an.

Die heuristische Methode reduziert die spitzen Winkel und verbessert damit in den meisten Fällen die Netzqualität. Dies kann allerdings nicht garantiert werden. Es existieren andere Methoden des mesh refinement, wie z.B. die *Rivara-Glättung* [BE92], für deren Verhalten theoretisch fundierte Aussagen gemacht werden können.

Anzumerken bleibt noch, daß der Lawson-Flip-Algorithmus zur Umwandlung einer Triangulierung in eine Delaunay-Triangulierung ebenfalls ein netzverbessernder Algorithmus ist.

4.4.2 Optimierung der Datenmenge

Datenreduzierende Algorithmen entsprechen einem von zwei Entwurfszielen.

Die erste Gruppe datenreduzierender Algorithmen befaßt sich mit zu hoch aufgelösten polygonalen Approximationen von Objektoberflächen. Die Datengrundlage der polygonalen Approximationen wird in einem Scanvorgang gewonnen (z.B. 3D-Laserscan, Vermessung der Erdoberfläche durch Satelliten), der die Tiefeninformation der Oberfläche an den Kreuzungspunkten eines feinen, regelmäßigen Gitters mißt. Jeweils vier quadratisch angeordnete Meßpunkte bilden die Eckpunkte eines vierseitigen Flächenelements im oberflächenapproximierenden Netz. Während des Meßvorgangs überabgetastete Bereiche erhalten damit eine unnötig feine polygonale Approximation. Der datenreduzierende Algorithmus verändert die Anzahl und Lage der Netzknotenpunkte und Verbindungskanten mit dem Ziel, eine möglichst gute polygonale Interpolation des Oberflächenmodells mit einer maximalen Anzahl von Stützpunkten (Knotenpunkte) zu erreichen oder mit minimaler Anzahl an Knotenpunkten eine festgelegte Fehlertoleranz der Approximation nicht zu unterschreiten.

Das zweite Anwendungsgebiet *multiresolution meshing* baut auf dem Ergebnis der statischen datenreduzierenden Algorithmen auf. Die Algorithmen legen ein generisches Oberflächenmodell des 3D-Objekts an, aus dem sich effizient Netze beliebiger Auflösung ableiten lassen. Zur effizienten computergraphischen Generierung von 3D-Szenen steht ein Objekt damit in der gerade notwendigen Auflösung abhängig vom Abstand zur virtuellen Kamera zur Verfügung.

Optimierung überabgetasteter polygonaler Approximationen einer Objektoberfläche

In [DZ91] wird die *adaptive Unterteilung (adaptive subdivision)* beschrieben, die eine Rechteck-Oberfläche auf den in gleichmäßiger, zylindrischer Gitterstruktur (Koordinaten (ϕ, h), siehe S. 8) vorliegenden Cyberware-Laserscandaten konstruiert. Dazu wird zunächst ein grobes Oberflächennetz aus Rechtecken erzeugt. Die Eckpunkte der Rechtecke sind Scanwerte an den Kreuzungspunkten eines groben quadratischen Gitters über der diskreten (ϕ, h)-Ebene. Auf Grund des ermittelten Approximationsfehlers der Oberfläche durch ein Rechteck, sowie weiterer lokaler geometrischer Analysen, wird für jedes Rechteck entschieden, in welcher Art es unter Hinzunahme weiterer Scandatenpunkte unterteilt werden soll. Die Unterteilung wird rekursiv auf den neu entstehenden Rechtecken fortgesetzt, bis sich kein Rechteck mehr findet, das die vorgegebene Fehlertoleranz überschreitet.

Der Knotenpunkt-Dezimierungs-Algorithmus in [SZL92] betrachtet jeden Knotenpunkt eines Dreiecksnetzes und seine lokale Umgebung aus den anliegenden Dreiecken. Zu Beginn wird jeder Knotenpunkt danach klassifiziert, ob es sich um einen normalen Knotenpunkt handelt oder der Knotenpunkt an der Ausbildung einer Kante oder einer Ecke des Objekts beteiligt ist. Die Analyse von Ecken oder Kanten erfolgt durch Winkelvergleich benachbarter Dreiecke. Im nächsten Durchlauf werden die Knotenpunkte der Klasse “Normal” gelöscht, falls die kartesische Distanz zur mittleren Ebene aus den benachbarten Punkten unterhalb eines festgelegten Wertes liegt. Ein Knotenpunkt der Klasse “Kante” wird gelöscht, falls sein Abstand zur angenommenen Kante unterhalb eines Grenzwertes liegt.

Das entstandene Loch in der Triangulierung wird mit den verbliebenen ehemaligen Nachbarpunkten neu trianguliert. Knotenpunkte der Klasse "Ecke" werden nicht gelöscht. Der Ablauf aus Klassifikation, Löschen und Retriangulieren wird mehrfach wiederholt, bis das Netz die gewünschten Eigenschaften aufweist.

Weitere Verfahren der Polygonreduktion werden in [HDD+93], [Tur92] beschrieben. Keine der Methoden erlaubt die Anlage zwingender Kantenzüge, wie sie für die animationsorientierte Optimierung von Polygonnetzen wünschenswert ist.

Bei der Optimierung von Polygonnetzen wird die polygonale Interpolation einer kontinuierlichen, glatten Oberfläche mit vielen Stützpunkten (Scanpunkte der Oberfläche) überführt in eine neue polygonale Interpolation mit wenigen, ausgesuchten Stützpunkten. Bemerkenswert ist der Umstand, daß die gefundenen Verfahren aus der Literatur der Computergrafik die Bestimmung der optimalen Position der ausgesuchten Stützpunkte nicht direkt durch die Differentialgeometrie für gekrümmte Flächen und die Interpolationstheorie motiviert sind. Das Problem wird in der Regel direkt in der Geometrie der polygonalen Oberflächennetze behandelt.

Multiresolution Meshing

Die Verfahren zur Erstellung von Netzen mit variabler Auflösung sind mathematisch abstrakter als die einfachen geometrischen Verfahren zur Polygonreduktion. Daher kann an dieser Stelle nur die Grundidee wiedergegeben werden.
In [EDD+95] wird die Oberfläche eines 3D-Objekts so partitioniert, daß sich für die Scanpunkte in jeder Partition eine günstige Darstellung als zweidimensionale Funktion finden läßt. Der Artikel [EDD+95] beschäftigt sich direkt mit zweidimensionalen Landschaftsdaten. Die 2D-Funktionen sind als Höhenbilder deutbar und werden anschließend mit Verfahren der diskreten *Multiresolution Analysis* betrachtet. In der Bildverarbeitung dient die Multiresolution Analysis der Bereitstellung aufeinanderfolgender Auflösungen eines Bildes. In jeder weiteren Auflösung geht Detailinformation verloren und mündet in einer konstanten 2D-Funktion ohne Informationsgehalt.

Das Spektrum der Auflösungen der Höhenbilder reicht hierarchisch von der Ebene bis zur maximal detaillierten Oberfläche aus allen Scanpunkten. In einem einfachen Denkmodell wird mit jedem Reduktionsschritt der Auflösung der Funktionswert eines detailbildenden, von seiner Umgebung abweichenden Scanpunktes an das mittlere Niveau seiner Umgebung angepaßt. In der polygonalen Oberfläche über den Punkten des Höhenbildes kann auf den nivellierten Punkt als Knotenpunkt verzichtet werden.
Mit jeder Reduktion der Auflösung werden weitere Scanpunkte als Knotenpunkte der Polygonoberfläche entnommen. Das Polygonnetz der Oberfläche wird an jeder Auflösungsstufe aus den verbleibenden Knotenpunkten neu konstruiert.

Die Verfahren zeichnen sich durch theoretisch fundierte und geschwindigkeitsoptimierte Elementreduktion in Polygonnetzen aus. Die Multiresolution Analysis wird mit der mathematischen Theorie der *Wavelet Transformation* [Chu92] erreicht.

Kapitel 5

Rechnerunterstützte Animationsnetzerstellung

Die strukturierte Zusammenstellung der Anforderung an das Animationsnetz in Abschnitt 2.4 und die Kapitel 3 und 4 zur Triangulierung von Punktmengen helfen beim Entwurf der Software zur rechnerunterstützten Erstellung von Animationsnetzen. Im ersten Abschnitt dieses Kapitels wird zu jedem Gesichtspunkt aus dem Anforderungskatalog ein algorithmisch formalisierbarer Lösungsvorschlag dargelegt. Die Einzelvorschläge fügen sich zum Gesamtkonzept der Software zusammen. Der darauffolgende Abschnitt geht auf die Umsetzung der Konzepte in der Testimplementierung ein.

5.1 Konzeption der Software

Der Anforderungskatalog zur animationsorientierten Netzerstellung aus Kapitel 2 wird in diesem Abschnitt die Grundlage für den strukturierten Entwurf der Software zur animationsorientierten Dreiecksnetzerstellung darstellen. Für jedes Merkmal eines manuell erstellten Animationsnetzes wird ein Vorschlag zu dessen algorithmischer Realisierung erläutert. Im Gesamtkonzept wird versucht, mit Hilfe der Advancing-Front-Triangulierung (Abschnitt 4.3.4) in Verbindung mit benutzerdefinierten Randbedingungen alle Merkmale eines Animationsnetzes nachzubilden. Von zentraler Bedeutung für die Benutzerinteraktion ist die Eingabe von Animationslinien. Aus den Animationslinien werden die getrennt triangulierten Regionen abgeleitet. Darüber hinaus definieren die Animationslinien den Rand der Domain, erlauben die Modellierung kontrollierter Bewegungsfalten und geben die Flußrichtung der Dreiecke im Netz vor.

5.1.1 Form der Netzelemente

Bei der manuellen Netzerstellung auf der gescannten Objektoberfläche wird darauf geachtet, daß die Dreiecke bei Berücksichtigung aller anderen Anforderungen keine zu spitzen Winkel aufweisen. In der Triangulierungstheorie der Ebene wird die Vermeidung spitzer Winkel mit dem Begriff Delaunay-Triangulierung (globale Optimierung des Max-Min-Winkel-Kriteriums siehe S. 28) bezeichnet.

Unter der Voraussetzung, daß es in der Software eine geeignete Methode gibt, die die Netzknotenpunkte auf der Oberfläche der Figur (die Oberfläche liegt vom 3D-Laserscan in

diskreter Form vor) bereits automatisch in die korrekte Position und Dichte gesetzt hat, sprechen zwei Einwände gegen die direkte Verwendung eines Delaunay-Triangulierungsalgorithmus. Erstens liegen die Oberflächenpunkte nicht in der Ebene. Daher muß entweder mit einem Algorithmus trianguliert werden, der auf einer 3D-Formulierung des Umkreiskriteriums basiert oder (siehe S. 59) die Oberflächenpunkte werden vor der Delaunay-Triangulierung mit einer geeigneten Transformation in die Ebene abgebildet. Die Hin- und Rücktransformation muß sorgfältig entworfen werden, damit die optimale Dreiecksform der Delaunay-Triangulierung auf der Oberfläche erhalten bleibt. Zweitens existieren weitere Anforderungen an das Animationsnetz, die eine Auswirkung auf die Dreiecksform haben, aber damit noch nicht berücksichtigt werden.

In der Testimplementierung soll auf frei zugängliche Software zur Delaunay-Triangulierung zurückgegriffen werden [Ren96]. Diese Software arbeitet nur mit Punktmengen in der Ebene. Daher werden die 3D-Koordinaten der Punkte in die Ebene transformiert. Das Auffinden einer korrekten Transformation basiert auf einer Formulierung des Umkreiskriteriums auf der Oberfläche. Beispielsweise könnte der Umkreis auf der Oberfläche bezüglich des Bogenmaßes definiert sein. Die Transformation $T : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ muß dann die Punktkoordinaten in einer Weise abbilden, die die Punktdistanzen in Oberflächen-Bogenmaß in der Ebene erhält. Dabei geht im Allgemeinen die Winkeltreue verloren. Die Längentreue ist unmöglich, sobald die Oberfläche mehr als eine Krümmungsrichtung besitzt. Abbildung 5.1 gibt ein Beispiel für dieses Problem, das sich auch bei der Projektion der Erdkugel auf Landkarten stellt.

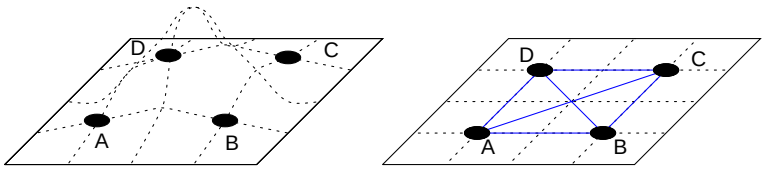


Abbildung 5.1: Das Bogenmaß ist im Allgemeinen nicht in zwei Dimensionen abbildbar
Die Geometrie auf der gekrümmten Oberfläche ist nicht in die Ebene übertragbar. In (b) wurden die vier Punkte aus (a) so in die Ebene abgebildet, daß für die Strecken $\overline{AB}, \overline{BC}, \overline{CD}$ und $\overline{DA}$ die 3D-Bogendistanz als kartesische Distanz in der Ebene erhalten bleibt. Die Bogendistanz der Diagonalen wird infolgedessen verzerrt.

Bei Landkarten wird der Projektionsfehler minimiert, indem nur kleine Ausschnitte des Globus in die Ebene abgebildet werden. Die gleiche Motivation besteht bei den Patches auf S. 43. Je kleiner ein Patch gewählt wird, um so kleiner wird die absolute Abweichung des Flächenstücks von einer Tangentialebene an einem typischen Oberflächenpunkt. Eine einfache Projektion der Oberflächenpunkte in Richtung der Normalen zum Patch kann in der Praxis brauchbare Approximationen liefern. Für jedes Patch gilt eine eigene lokale Metrik. Die frei erhältliche Triangulierungssoftware akzeptiert aber nur Eingabepunkte in der globalen Ebene, weshalb ein Ansatz mit lokaler Metrik ausscheidet.

Mit der Testimplementation soll überprüft werden, ob bei der speziellen Geometrie der Figuren eine globale Transformation aller Netzknotenpunkte ausreicht, um akzeptable Dreiecksformen auf der Oberfläche zu erhalten. Eine einfache Transformationsvorschrift ergibt sich direkt aus der Scangeometrie. Die Scandaten sind eine diskrete zweidimensionale Funktion. Die Entfernung vom Scandetector zum Objektpunkt für eine Winkelposition und eine vertikale Position stellt den Funktionswert dar. Ein Oberflächenpunkt ist über diese drei Zylinderkoordinaten definiert. Bei Verzicht auf die Tiefeninformation gibt die Scangeometrie eine einfache Abbildungsvorschrift für die 3D-Oberflächenpunktkoordinaten in die Ebene vor. In der Ebene besitzen die Netzknotenpunkte nur die Winkelkoordinate und die Vertikalkoordinate. Im unteren Fenster in Abbildung 2.8 sind die Knotenpunkte des manuell erstellten Netzes in die Ebene abgebildet. Die Transformation entspricht der Projektion der 3D-Koordinaten der Oberflächenpunkte auf den Zylindermantel der Scangeometrie (Abb. 2.4) in Richtung des Laserstrahls.

Im Fall dieser einfachen Transformationsvorschrift erweist sich die Unzulänglichkeit des Laserscanners, gefaltete Oberflächen mit Löchern oder Überdeckungen abzutasten, als Voraussetzung für die einfache Transformation in die Ebene. Trotz ihrer Trivialität besitzt die Transformation die Eigenschaft, nur geringe Abweichungen der Winkelgröße zwischen einander entsprechenden Dreiecken in 2D und in 3D zu erzeugen. Die Versuchsreihe wird zeigen, ob die relativ winkeltreue Transformation der 2D-Delaunay-Triangulierung ausreichend wohlgeformte Dreiecke auf der Oberfläche erzeugt. Bei unbefriedigenden Ergebnissen empfiehlt sich als alternatives Verfahren CHEWs Algorithmus zur Constrained-Delaunay-Triangulierung von Oberflächen. CHEW arbeitet mit einer direkten Formulierung des Delaunay-Kriteriums auf Oberflächen. Die Methode stellt einen schlankeren Ansatz dar, als die aufwendige Entwicklung einer differentialgeometrisch fundierten, globalen und längentreuen Transformation zwischen der Oberfläche und der Ebene (zur Differentialgeometrie von gekrümmten Flächen siehe [dC83],[AM91],[BS87]).

Der eingangs genannte zweite Einwand zur einfachen Delaunay-Triangulierung der Punktmenge erinnert an die anderen Anforderungen an das Animationsnetz, die bestimmte Kantenzüge in der Punktmenge erzwingen. Beispielsweise verlangt die kontrollierte Faltung während der Animation bestimmte Kantenzüge in der Triangulierung. Hierzu bietet die Triangulierungstheorie eine passende Modifikation der Delaunay-Triangulierung an. Die 2D-Constrained-Delaunay-Triangulierung optimiert die Dreiecksform unter Berücksichtigung zwingender Randkanten und interner Kanten. Sie ist Bestandteil der AFT-Methode. CHEWs Algorithmus zur Oberflächentriangulierung berücksichtigt ebenfalls benutzerdefinierte Kanten und realisiert dies mit einer echten 3D-Constrained-Delaunay-Triangulierung.

Die Wohlgeformtheit der Dreiecke, erzeugt mit der AFT-Methode aus Abschnitt 4.3.4, entsteht durch Konstruktion gleichseitiger Dreiecke über einer Kante der Front und den Rückgriff auf Constrained-Delaunay-Algorithmen. Beides sind Konzepte der Ebene und müssen für 3D-Oberflächen angepaßt werden. In der Implementierung arbeitet der CDT-Algorithmus auf den zylindrischen Projektionskoordinaten der Knotenpunkte. Die Konstruktion der gleichseitigen Dreiecke findet ebenfalls in dieser Ebene statt. In allen Bereichen, in denen die Flächennormale stark von der Scanrichtung abweicht, entstehen dadurch verformte Dreiecke in 3D (Abb. 5.2). Die ungünstige Plazierung des neuen Kandidaten R verhindert die Entstehung eines wohlgeformten Dreiecks, obwohl die Randbedingungen

dies zugelassen hätten.

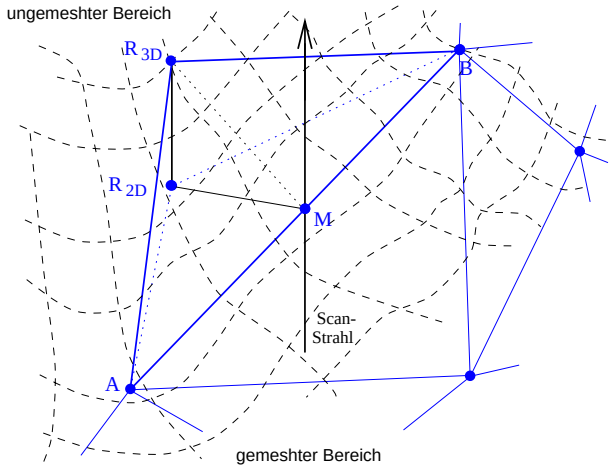


Abbildung 5.2: Verzerrung bei 2D-Konstruktion eines neuen Dreiecks
Die Flächennormale weicht in dem betrachteten Bereich stark von der Projektionsrichtung (Scanrichtung) ab. Durch die Konstruktion des gleichseitigen Dreiecks in der Projektionsebene entsteht auf der Oberfläche ein Dreieck mit vergrößerter Seitenlänge und spitzerem Winkel am Eckpunkt R.

Dieser Fehler, hervorgerufen durch die fehlende Längentreue der Transformation, wird vermieden, indem die Dreiecke direkt auf der 3D-Oberfläche konstruiert werden. Da die Oberfläche nur diskret vorliegt und sich daher der differentialgeometrische Ansatz nicht trivial anbietet, wird die optimale Position von R numerisch angenähert. Über der aktuellen Kante $\overleftrightarrow{AB}$ wird das gleichseitige (Seitenlänge ist definiert durch die Konstante oder Funktion *unit size*) Dreieck in eine senkrechte Ausgangslage gebracht. Danach wird das Dreieck so lange um die Achse $\overleftrightarrow{AB}$ in linker Richtung gedreht, bis R minimalen Abstand zum nächsten Oberflächenpunkt besitzt.

Die Koordinate von R in Abhängigkeit vom Drehwinkel δ wird durch Addition von drei Vektoren bestimmt: $R(\delta) = M + cos(\delta)\overleftrightarrow{Mp} + sin(\delta)\overleftrightarrow{Mq}$. Für $\overleftrightarrow{Mq}$ gilt: $\overleftrightarrow{Mq} \perp \overleftrightarrow{0M}$ und $\overleftrightarrow{Mq} \perp \overleftrightarrow{AB}$. $\overleftrightarrow{0M}$ bezeichnet die Richtung des Scanstrahls für M, den Mittelpunkt der aktuellen Kante. $\overleftrightarrow{0M}$ ist gleichzeitig Normalenvektor der Ebene E_2. $\overleftrightarrow{AB}$ ist Normalenvektor der Ebene E_1, in der R radial um M bewegt wird (Abb. 5.3). Die Berechnung von $\overleftrightarrow{Mq}$ erfolgt mit Hilfe des Vektorprodukts: $\overleftrightarrow{Mq} = \overleftrightarrow{0M} \times \overleftrightarrow{AB}$. Für $\overleftrightarrow{Mp}$ gilt: $\overleftrightarrow{Mp} \perp \overleftrightarrow{Mq}$ und $\overleftrightarrow{Mq} \perp \overleftrightarrow{AB}$, daher gilt: $\overleftrightarrow{Mp} = \overleftrightarrow{Mq} \times \overleftrightarrow{AB}$.

Der Abstand von R zur Oberfläche läßt sich im Zylinderkoordinatensystem in einer Näherung einfach bestimmen. Dazu wird während der Drehung jeweils der Punkt der Oberfläche, mit dem zu R identischen Winkelinkrement und Vertikalposition ausgewählt, und die Differenz der dritten Koordinate, der Tiefeninformation wird gebildet. Im Drehbereich links von der aktuellen Kante besitzt die Funktion genau eine Nullstelle. Diese Nullstelle ist mit dem Newton-Verfahren ausreichend effizient bestimmbar.

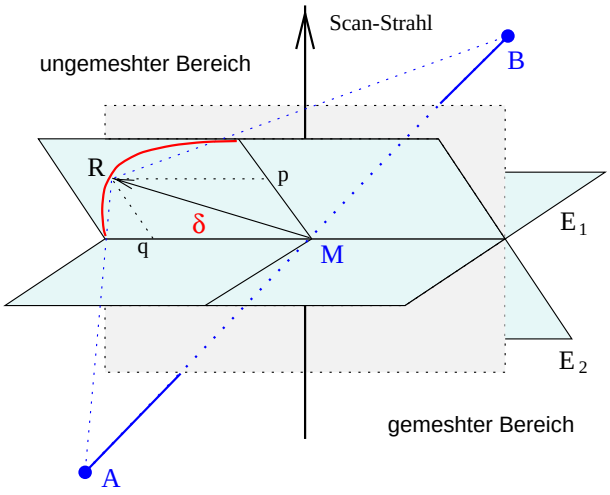


Abbildung 5.3: Plazierung jedes neuen Dreiecks auf der 3D-Oberfläche

5.1.2 Oberflächenapproximation mit lokaler Fehlertoleranz

Der Fehler in der Approximation der Oberfläche durch ein Polygonnetz wird minimiert, wenn die Dichte der Knotenpunkte abhängig ist von der lokalen Krümmung der Oberfläche. Die Krümmungsinformation berechnet sich aus den Scandaten und steht damit für die automatische Plazierung der Knotenpunkte zur Verfügung. Die zusätzliche Information über die Partitionierung der Oberfläche in Regionen mit zugehöriger Standardkantenlänge der Dreiecke (korrolar zur Knotenpunktdichte) ist nicht aus den Scandaten ableitbar. Die Kontrollinformation ist eine Vorgabe des Animators. Wie sich in den folgenden Abschnitten herausstellt, induzieren die benutzerdefinierten Animationslinien eine geeignete Partitionierung der Region. Die Standardkantenlänge in einer Region gibt der Animator über die Benutzerschnittstelle der Software ein.

Bei der Triangulierung einer Region gibt es zwei Situationen für die krümmungsabhängige Plazierung von Netzknotenpunkten. Die Initialisierung des Regionenrands stellt die erste Situation dar. Die zweite Situation liegt im Innern der Region vor, wenn die Kantenlänge jedes neu konstruierten Dreiecks, abhängig von der mittleren Oberflächenkrümmung in diesem Bereich, skaliert wird.

Zur Bestimmung der Krümmung auf einer Raumkurve und auf einer Fläche im Raum

Es sei $f : \mathbf{I} = (a, b) \to \mathbb{R}^3$ eine nach der Bogenlänge $s \in \mathbf{I}$ parametrisierte zweifach differenzierbare Kurve und $s_1 \in \mathbf{I}$.

Definition 5.1 (Tangente) *Der Einheitsvektor $t(s_1) = f'(s_1)$ hat die Richtung der Tangente im Punkt $f(s_1)$ und weist in die positive Kurvenrichtung.*

Definition 5.2 (Krümmung) *Die Zahl $k(s_1) = |f''(s_1)|$ heißt die Krümmung* (engl. curvature) *von f an der Stelle s_1.*

$|f''(s)|$ mißt die Änderungsrate des Tangentenwinkels $\frac{d\theta}{ds}$ und gibt an, wie schnell sich die Kurve in einer Umgebung von s von der Tangente bei s wegdreht (Abb. 5.4(a)).

Definition 5.3 (Krümmungsradius) $\varrho = \frac{1}{k}$ *heißt Krümmungsradius.*

Der Krümmungsradius ist der Radius des größten Kreises, der an der betrachteten Stelle einen gemeinsamen Punkt mit $f(s)$ besitzt und auf der Seite der Kurve liegt, in die $f''(s)$ zeigt (Abb. 5.4(b)).

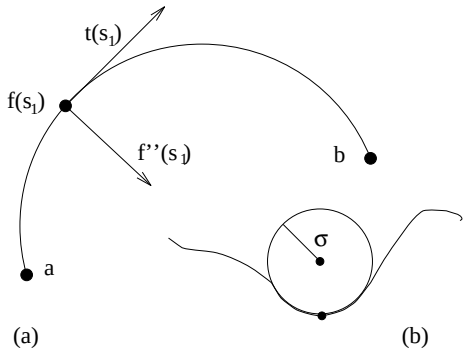


Abbildung 5.4: Krümmung einer Raumkurve
(a) Tangente $t(s)$ und $f''(s) = nk$. n heißt Hauptnormalenvektor. k ist die Krümmung. (b) Krümmungskreis mit Krümmungsradius ϱ.

Die Krümmungseigenschaften an einem Punkt einer Fläche werden mit Hilfe der zwei Hauptkrümmungsrichtungen beschrieben.

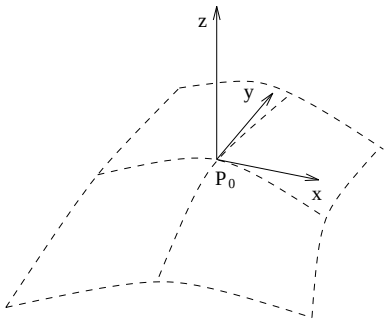


Abbildung 5.5: Kartesisches Koordinatensystem an einem Flächenpunkt P_0.
(nach [BS87])

Zu einem festen Flächenpunkt P_0 kann man stets ein kartesisches x, y, z-System wählen, dessen Ursprung in P_0 liegt und dessen x, y-Ebene mit der Tangentialebene durch P_0 übereinstimmt (Abb. 5.5). In diesem x, y, z-Koordinatensystem besitzt die Fläche in einer Umgebung von P_0 die Darstellung $z = z(x, y)$. Durch Drehung des Koordinatensystems um die z-Achse kann man ein ausgezeichnetes Koordinatensystem erzeugen, in dem sich

die beiden Fundamentalformen zur Beschreibung der Fläche besonders einfach mit Hilfe der Hauptkrümmungen ausdrucken lassen [BS87]. Zusätzlich vereinfacht sich die Taylorentwicklung am Punkt P_0 zu $z = \frac{1}{2}(k_1 x^2 + k_2 y^2) + \ldots$.

Definition 5.4 (Hauptkrümmungen) *k_1, k_2 sind die Hauptkrümmungen, $R_1 = \frac{1}{k_1}$, $R_2 = \frac{1}{k_2}$ sind Hauptkrümmungsradien, $H = \frac{1}{2}(k_1 + k_2)$ ist die mittlere Krümmung und $K = k_1 k_2$ ist die Gaußsche Krümmung im Punkt P_0.*

Für eine Kugel vom Radius R gilt beispielsweise $R_1 = R_2 = R, K = \frac{1}{R^2}, H = \frac{1}{R}$. Die Vereinigung aller Krümmungskreise an einem Oberflächenpunkt ergibt die Kugel.
Die Hauptkrümmungen geben die Krümmung der Flächenkurve mit der höchsten und der niedrigsten Krümmung bei P_0 an. Aus der Kenntnis der Hauptkrümmungen an einem Punkt kann man die Krümmung aller durch P_0 gehenden Flächenkurven berechnen. Die Hauptkrümmungen dienen darüber hinaus der Klassifikation der Flächenpunkte.

Damit ist nun der Krümmungsbegriff für Flächen im dreidimensionalen Raum ausreichend motiviert. Für die Theorie der Berechnung der Hauptkrümmungen sei auf die Literatur verwiesen [BS87], [AM91]. Voraussetzung für die aufwendige Berechnung der Hauptkrümmungen ist eine kontinuierliche mathematische Beschreibung der Fläche. Im Fall der 3D-Laserscandaten müßte diese Beschreibung erst noch gewonnen werden.

In der Literatur werden bei Algorithmen, deren Eingabe diskrete, explizite Beschreibungen von Raumkurven oder Flächen sind, zur Berechnung der Krümmungsinformation oft triviale Näherungen eingesetzt. Für die meisten Anwendungen liefern diese schnellen Methoden ausreichende Näherungen. Zur Berechnung der Kurvatur diskreter Kurven werden in [WS92] fünf Näherungsmethoden verglichen. Zur Bestimmung der Kurvatur an einem Punkt B der Kurve analysieren diese Methoden die geometrische Lage von B zu seinem Vorgängerpunkt A und Nachfolgerpunkt C auf der diskreten Kurve. Je feiner das Diskretisierungsraster der Kurve ist, umso bessere Näherungen der Krümmung lassen sich berechnen, falls die Werte in einer Umgebung um den betrachteten Punkt gemittelt werden.

Für einen Flächenpunkt P_0 ist es möglich, die Hauptkrümmungen anzunähern, indem die Krümmungen über eine endliche Zahl von Flächenkurven durch P_0 maximiert und minimiert werden.

In der Literatur zur datenreduzierenden Oberflächenrekonstruktion aus Scandaten ([DZ91], [EDD+95], [GGS95], [HDD+93], [HDD+94], [HDDM92], [SZL92], [Tur92]) und in CHEWs Algorithmus zur Triangulierung von Oberflächen (S. 59) basiert die Approximation der Oberfläche auf einfacheren Fehlerkriterien, für die keine differentialgeometrische Analyse der Krümmung notwendig ist. Die Algorithmen konstruieren dreieckige oder viereckige Netzelemente, die die Oberfläche in betreffenden Bereich approximieren. Ist der Approximationsfehler zu hoch, dann wird die Elementgröße entsprechend reduziert. Der Approximationsfehler berechnet sich aus dem Abstand der Oberfläche vom Mittelpunkt des Netzelements. Weist die Oberfläche in dem Bereich eine hohe Krümmung auf, dann erhöht sich der Approximationsfehler und die Elementgröße wird nach unten korrigiert.

Mit diesen Fehlerfunktionen läßt sich bei deutlich geringerem Rechenaufwand ebenfalls eine gute Approximation der Oberfläche erreichen. Der Ansatz berücksichtigt, daß

das Ziel der Algorithmen nicht darin besteht die kontinuierliche Oberflächenfunktion zu rekonstruieren, sondern eine ausreichende polygonale Approximation der Oberfläche zu bestimmen.

Realisierung der krümmungsabhängigen Dreieckskantenlänge

Zuerst werden die Ansätze zur krümmungsgesteuerten Initialisierung der Regionengrenzen vorgestellt. Danach wird auf die Konzepte innerhalb einer Region eingegangen.

Die Begrenzung einer Region setzt sich aus Oberflächenkurvensegmenten der Animationslinien zusammen. Ein Segment ist Teilmenge einer Animationslinie und umfaßt die Oberflächenkurve zwischen zwei Schnittpunkten mit anderen Animationslinien. Innerhalb eines Segments besitzt die Animationslinie keine weiteren Schnittpunkte. Die Beachtung der Schnittpunkte ist Randbedingung für die Initialisierung der Kurvensegmente mit Netzknotenpunkten, denn die Anfangs- und Endpunkte der Segmente fungieren als definierte Verbindungspunkte mit anderen Segmenten, wenn die Begrenzung einer Region zusammengesetzt wird. Zwischen den Endpunkten werden die Initialknotenpunkte so verteilt, daß der entstehende Kantenzug das Kurvensegment gut approximiert und die Kantenlänge der mittleren Kantenlänge in den beiden angrenzenden Regionen entspricht oder nur in einem begrenzten Rahmen davon abweicht.

Zwei Ansätze der Initialisierung sollen getestet werden. Die äquidistante Initialisierung weist in jedem Intervall zwischen zwei Knotenpunkten die gleiche Bogenlänge auf. Dazu werden die Knotenpunkte in gleichmäßigem Abstand entlang der Raumkurve verteilt (Abb. 6.3). Die krümmungsabhängige Initialisierung realisiert eine höhere Punktdichte bei höherer Krümmung der Kurve. Dazu liegt in jedem Intervall die gleiche Krümmungssumme vor. Die krümmungsbasierte Initialisierung hat nicht das primäre Ziel, die Raumkurve mit der kleinsten Stützpunktzahl zu approximieren. Eine mögliche Strategie zur linearen Approximation einer Kurve besteht darin, die Stützpunkte (entsprechen den Initialknotenpunkten) an den Extremstellen und Wendepunkten der Kurve zu setzen. Dabei können jedoch sehr ungleichmäßige Kantenlängen entstehen, die auch durch weitere Unterteilung nicht an die mittleren Kantenlängen der benachbarten Regionen anzupassen sind. Die Randbedingung, von den mittleren Kantenlängen in den jeweils zwei benachbarten Regionen nur begrenzt abzuweichen, ist aber maßgeblich. Daher wird die krümmungsbasierte Initialisierung ebenfalls mit einem Intervallsummenverfahren vollzogen.

Im ersten Schritt der Segmentinitialisierung wird eine Näherung der Bogenlänge oder der Gesamtkrümmung des Segments bestimmt. Dabei werden die kartesischen Abstände, beziehungsweise die Beträge der Einzelkrümmungen aller n direkt aufeinanderfolgenden diskreten Punkte P_i der Raumkurve zum Wert $s = \sum_{i=2}^{n} |P_i \Leftrightarrow P_{i-1}|$, beziehungsweise $s = \sum_{i=2}^{n} Krümmung(P_{i+1}, P_i, P_{i-1})$ summiert. Die Gesamtzahl der Knotenpunkte auf dem Segment berechnet sich $N_{nodes} = abs(s/d_I) + 2$, wobei d_I die angestrebte Bogenlänge oder Krümmungssumme für ein Intervall bezeichnet.
Beide Ansätze beziehen die Knotenpunkte an den Segmentenden in die Verteilung ein, indem die Intervallbogenlänge, bzw. die Intervallkrümmung entsprechend reduziert wird. Das Intervallmaß wird mit folgender Vorschrift reduziert: $d_{I'} = \frac{s}{abs(s/d_I)+1}$.

Die diskrete Näherung der Krümmung an einem Punkt auf dem Segment wird mit Hilfe

trigonometrischer Funktionen berechnet. Sollte der Einsatz trigonometrischer Funktionen in der Praxis zu kritischen Laufzeiten führen, so wird auf rein vektorielle Methoden zur Krümmungsberechnung zurückgegriffen werden müssen.
Zur Berechnung der Krümmung an einem Punkt wird eine einfache diskrete Umsetzung der Definition der Krümmung ($d\theta/ds$, s.o.) herangezogen (Abb. 5.6).

$$Krümmung(P_{i+1}, P_i, P_{i-1}) = \frac{\pi \Leftrightarrow \sphericalangle P_{i+1}P_iP_{i-1}}{|\overrightarrow{P_{i+1}P_i}| + |\overrightarrow{P_iP_{i-1}}|}$$

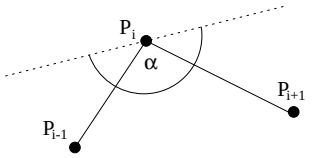


Abbildung 5.6: Berechnung der Krümmung
$\pi - \alpha$ ist ein Maß für die Abweichung von der Tangente (gestrichelte Linie). $|\overrightarrow{P_{i+1}P_i}| + |\overrightarrow{P_iP_{i-1}}|$ nähert das Bogendifferential.

Der Parameter d_I wird vom Benutzer so gewählt, daß die Kantenlängen auf dem Segment einen guten Übergang zwischen den Kantenlängen der anliegenden Dreiecke aus den benachbarten Regionen darstellen. Aus Kenntnis der Standardkantenlängen der Nachbarregionen und der Oberflächenkrümmung in den anliegenden Regionen läßt sich eine Berechnungsvorschrift zur Bestimmung von d_I formulieren.

Der Kantenzug als Approximation für ein Kurvensegment stellt die Schnittstelle zur Verbindung der Triangulierungen der beiden anliegenden Regionen dar. Bei Anwendung des AFT-Algorithmus sollte keine der begrenzenden Kanten in zwei Kanten gespalten werden, da der AFT-Algorithmus in der benachbarten Region mit anderem *unit_size* arbeiten kann und die Spaltung der betreffenden Kante nicht ebenfalls vollzieht. Nach dem Zusammenfügen der beiden Triangulierungen würde in diesem Fall eine ungültige Triangulierung entstehen.

Falls *unit_size* kleiner oder gleich der halben Länge einer inneren Kante ist, wird die Kante gespalten. Bei begrenzenden Segmentkanten wird die Spaltung abgewendet, indem *unit_size* solange um den Faktor $1,5$ erhöht wird, bis die Segmentkante und der neue Knotenpunkt R ein Dreieck bilden.

Zur Realisierung der krümmungsabhängigen Dreieckskantenlänge innerhalb einer Region gibt es einen indirekten, transformationsbasierten Ansatz und einen direkten Ansatz, bei dem *unit_size* eine Funktion der Oberflächendaten darstellt.

Der Transformationsansatz vollzieht eine krümmungsbasierte Abbildung
$T : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ der Initialknotenpunkte. Der AFT-Algorithmus arbeitet anschließend mit konstanter *unit_size* in der 2D-Domain, in der die Distanzen entsprechend der Krümmung zwischen zwei Punkten der Oberfläche reduziert sind. Nach dem Terminieren des AFT-Algorithmus werden die Knotenpunkte zurücktransformiert und CDT-trianguliert. Die Transformation hat die *unit_size* in Zonen hoher Krümmung gestaucht, so daß die Dreiecke

i.A. ihre Delaunay-Eigenschaft nicht bewahren, und die Knotenpunktmenge nachtrianguliert werden muß.

Zur Koppelung des Transformationsatzes mit der 3D-Konstruktion der Dreiecke aus Abschnitt 5.1.1 fehlt *unit_size* als expliziter Wert. Die Bestimmung von *unit_size* aus der Transformation ist nicht praktikabel. Der Test des Transformationsansatzes erfolgt daher ohne die 3D-Konstruktion der Dreieckskandidaten.

Für die Transformation dient eine Näherung der maximalen Hauptkrümmung für jeden Punkt der Oberfläche als Krümmungsinformation. Zur Berechnung der Näherung an einem Punkt B wird im zylindrischen Koordinatensystem der Scanpunkte eine 7×7-Umgebung der Oberfläche herangezogen. Die Nachbarschaft der Oberflächenpunkte in der Umgebung basiert auf den Zylinderkoordinaten Winkelinkrement ϕ und der vertikalen Position h. Über 12 Winkelpositionen (Abb. 5.7) wird der Näherungswert für die Krümmung maximiert.

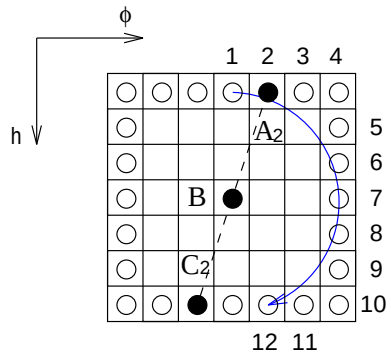


Abbildung 5.7: Näherung der maximalen Krümmung an einem Punkt der Oberfläche

$$Krümmung_{max} = \max_{i\in\{1,\ldots,12\}} \frac{\sphericalangle A_i B C_i}{|\overrightarrow{A_iB}| + |\overrightarrow{BC_i}|}$$

Die Abbildung eines Oberflächenpunktes $P = (\phi_P, h_P, r_P)$ mit der Krümmungstransformation wird folgendermaßen berechnet: Zuerst erfolgt die Bestimmung der Summe der Krümmungswerte über alle Flächenpunkte mit Winkelinkrement ϕ_P (entspricht einer Spalte des 512^2-Feldes) und über alle Flächenpunkte mit vertikaler Position h_P (entspricht einer Zeile). Die Berechnung der Komponenten der diskreten 2D-Bild-Koordinaten $x \in \{1,\ldots,512\}$, $y \in \{1,\ldots,512\}$ erfolgt unabhängig voneinander. Beim zweiten Durchlauf wird das Verhältnis der Teilkrümmungssumme zur Gesamtkrümmungssumme für die betrachtete Position in der aktuellen Zeile und Spalte bestimmt. Anhand der errechneten Verhältnisse werden die x- und y-Koordinate des Punktes bestimmt (Abb. 5.8). Die Abbildung der Zylinderkoordinaten ϕ, h nach x, y ist monoton, da die Summierung der Krümmungswerte monoton steigend ist.

Beim direkten Ansatz zur krümmungsabhängigen Steuerung der Dreieckskantenlängen ist *unit_size* eine Funktion des Fehlers der Approximation der Oberfläche durch das Dreieck. Der Approximationsfehler entsteht durch die Oberflächenkrümmung. Das neue Dreieck aus der aktuellen Kante E und dem Kandidaten R wird zuerst mit der Standard-

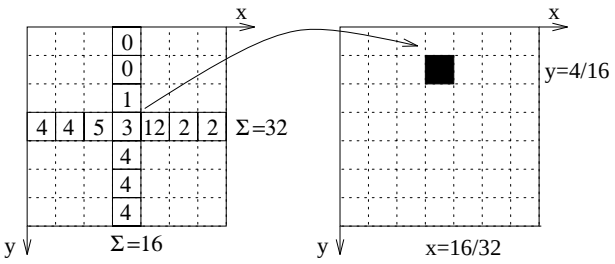


Abbildung 5.8: Berechnung der Krümmungstransformation

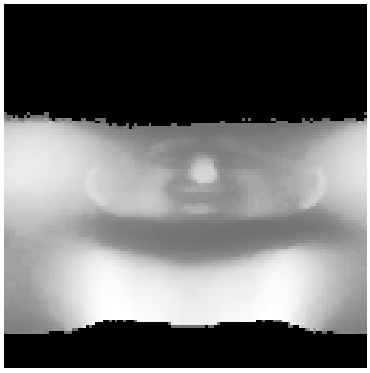


Abbildung 5.9: Krümmungsbild
Die lokale maximale Krümmung wurde einem Grauwert zwischen 0 und 255 zugeordnet.

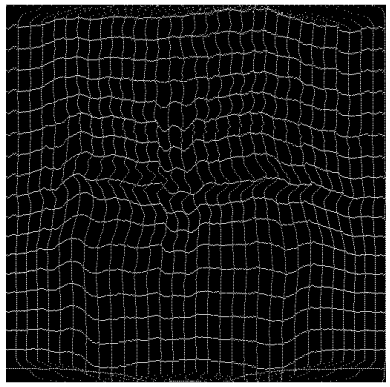


Abbildung 5.10: Abbildung eines Gitters
Verzerrung eines Gitters auf dem Zylindermantel nach der Transformation.

seitenlänge erzeugt. Danach wird der Abstand zwischen der halben Dreieckshöhe über E und der Projektion dieses Punktes auf die Oberfläche bestimmt. Die Projektion erfolgt in Scanrichtung. Liegt der Abstand über einem empirisch ermittelten Grenzwert, wird der Wert von *unit_size* kurzzeitig um ein Drittel verkleinert. Die Bestimmung von R wird wiederholt und der AFT-Algorithmus fährt fort.

Im Anforderungskatalog wird für jede Region eine eigene Standardkantenlänge gefordert, die der Benutzer abhängig von der Bedeutung und dem typischen Betrachtungsabstand der Region vorgibt. Dieser Wert dient als Ausgangswert für *unit_size* und wird entsprechend der lokalen Krümmung skaliert. Die Ausgangskantenlänge auf der Regionenbegrenzung stellt das Mittel der jeweils angrenzenden zwei Regionen dar.

5.1.3 Höhere Dreiecksdichte in stark animierten Regionen

Während der Animation werden insbesondere an Mund und Augen feinere Oberflächenstrukturen ausgebildet, als das Ausgangsgipsmodell aufweist. Dies ist nur möglich, wenn das Animationsnetz die gescannte Oberfläche in diesen Bereichen bereits mit der notwendigen höheren Knotenpunktdichte approximiert. Die Überabtastung wird vom Benutzer veranlaßt, indem er die Standardkantenlänge in den betreffenden Regionen entprechend klein wählt.

Für eine algorithmische Analyse der animationsbedingten Feinheit fehlt die Datengrundlage über die Animation der Figur. Die Modellierung der Bewegungen findet erst auf dem fertiggestellten Animationsnetz statt. Zum Zeitpunkt der Netzerstellung liegt das Wissen über die Animation beim Benutzer.

5.1.4 Öffnungen im Netz und Verbindungen zu weiteren Netzen

Öffnungen im Netz und Schnittstellen zu weiteren Netzen werden vom Benutzer ebenfalls in Form zwingender Kantenzüge angelegt. Die Eingabe ist identisch zu den Animationslinien. Die Randkantenzüge bilden ebenfalls Regionenbegrenzungen. Bei ihrer Initialisierung gelten leicht modifizierte Randbedingungen. Bei Öffnungen im Netz liegt nur eine Region am Kantenzug an. Das Verbot der Kantenspaltung könnte daher fallengelassen werden, was in der Praxis nicht relevant sein wird, da die Knotenpunktdichte auf dem Randkantenzug ohne Rücksicht auf eine zweite anliegende Region gewählt werden kann. Bei Schnittstellen zu anderen Netzen muß sichergestellt werden, daß die Schnittstelle mit identischen Knotenpunkten initialisiert wird. Bei automatischer Wahl der Knotenpunktdichte über einer Animationslinie in Abhängigkeit der beiden benachbarten Regionen, sind die Parameter für das zweite Netz nicht bekannt. Es empfiehlt sich, die Schnittlinie mit einem benutzerdefinierten Wert für die Knotenpunktdichte zu initialisieren und die gleiche Initialisierung für beide Netze zu verwenden.

5.1.5 Kontrollierte Faltung während der Animation

Dieser und der folgende Abschnitt bilden den Ursprung für das Animationslinienkonzept. Eine ursprüngliche Intention des Animationslinienkonzepts liegt darin, zwingende Kantenzüge im Netz zu definieren, die bei der Bewegungsmodellierung die kontrollierte Ausbildung von Faltungslinien auf der Haut gestatten.
Im Gesamtkonzept setzt der Benutzer die Animationslinien zur Kontrolle weiterer Netzmerkmale ein (Partitionierung der Domain in Regionen, Definition der Domainränder, sowie Steuerung der Flußrichtung).

5.1.6 Flußrichtung der Dreiecke

Ein Grund für die Wahl der Advancing-Front-Triangulierung ist ihre Eigenschaft, die Dreiecke sukzessive ausgehend von der Regionenbegrenzung zu plazieren. Die Regionenbegrenzungen werden zusammengesetzt aus Segmenten der Animationslinien. Damit ist es möglich, über die Lage von Animationslinien Einfluß auf die Flußrichtung der Dreiecke zu nehmen.

5.1.7 Umsetzung des kaum formalisierbaren Expertenwissens

Ohne Formalisierung ist es nicht möglich, eine Anforderung algorithmisch umzusetzen. Für die Ausbildung von Netzmerkmalen, die nicht vom Konzept der Software berücksichtigt werden können, soll der Benutzer daher eine nachträgliche Editiermöglichkeit erhalten. Sie erlaubt ihm, die Knotenpunkte und Kanten des Netzes zu manipulieren.

5.1.8 Benutzerschnittstelle zur Eingabe der Animationslinien und Partitionierung der Domain in Regionen

Der Benutzer zeichnet die Animationslinien in das Texturbild der Figur ein. Die Textur der Figur wurde im Skulpturierungsschritt zusammen mit dem Gipsmodell erstellt und dient hier dem Benutzer beim zweidimensionalen Zeichnen zur Orientierung auf der Oberfläche. Die 2D-Bildkoordinaten besitzen für jedes Pixel eine eindeutige Zuordnung zu einem Oberflächenpunkt, da die Bildkoordinaten den Zylinderkoordinaten ϕ, h entsprechen.
Die Endpunkte und die Schnittpunkte der Animationslinien partitionieren die Animationslinien in Segmente. Die Segmente werden abgespeichert, damit sie zur Bildung der Regionen zur Verfügung stehen. Die Begrenzung einer Region kann sich aus mehreren Segmenten zusammensetzen. Die Initialisierung einer Regionengrenze wird in Abschnitt 5.1.2 beschrieben.

5.2 Testimplementierung

Die vorgenommene Implementation umfaßt einen Regionentriangulierer, sowie mehrere Hilfsprogramme zur Eingabe der Animationslinien, Initialisierung der Segmente und zur Verbindung der triangulierten Regionen. Sie dient als Verifikationsumgebung für das entwickelte Konzept. Die Erfahrungen mit der Testimplementierung bezüglich der Qualität der Netze, der Kontrollierbarkeit der Netzeigenschaften mit den verwendeten Parametern und dem Laufzeitverhalten der Software sollen im Anschluß an diese Arbeit die Grundlage für eine dauerhafte Implementierung schaffen.

Abbildung 5.11 zeigt den groben Ablaufplan des Regionentriangulierers. Eingabe der Software sind zum Startzeitpunkt die 3D-Laserscandaten. Zur Laufzeit wird der Benutzer aufgefordert eine Region mit eventuellen, zwingenden Kantenzügen und ihren Triangulierungsparameter anzugeben. Die entsprechende Region wird trianguliert und das Ergebnis in einer Datei gesichert. In jedem Schleifendurchlauf kann der Benutzer die Triangulierung in einer 2D-Ansicht am Bildschirm betrachten und sich anschließend dafür entscheiden, die gleiche Region mit veränderten Parametern neu zu triangulieren, eine andere Region zu wählen oder das Programm abzubrechen, falls alle Regionen der Domain trianguliert sind.

Das Hauptprogramm zur Regionentriangulierung ist auf Hilfsprogramme zur Vor- und Nachbearbeitung angewiesen. In der Software zur Vorbereitung der Segmente definiert der Benutzer die Animationslinien. Aus der Analyse der Schnittpunkte werden die Segmente gewonnen, äquidistant oder krümmungsabhängig initialisiert und die Initialkantenzüge für den Zugriff aus dem Hauptprogramm vorbereitet. Die Nachbearbeitung liest die Regionentriangulierungen ein und fügt sie zum Animationsnetz zusammen. Das erzeugte Dateiformat ist kompatibel zu einem Standardformat. Die Netze können von Softwarepaketen der Computeranimation eingelesen, räumlich betrachtet und weiterverarbeitet werden.

Die nun folgenden Abschnitte stellen die Kontroll- und Datenstrukturen informell vor, ohne die programmiersprachlichen Details zu vertiefen. Die Implementierungen wurden auf einer UNIX-Grafik-Workstation in der Programmiersprache C vorgenommen. Die Grafikausgabe geschieht in einem Fenster des X-Windows-Systems. Zur Programmierung der Grafikausgabe wurden OSF/Motif-Bibliotheken verwendet.

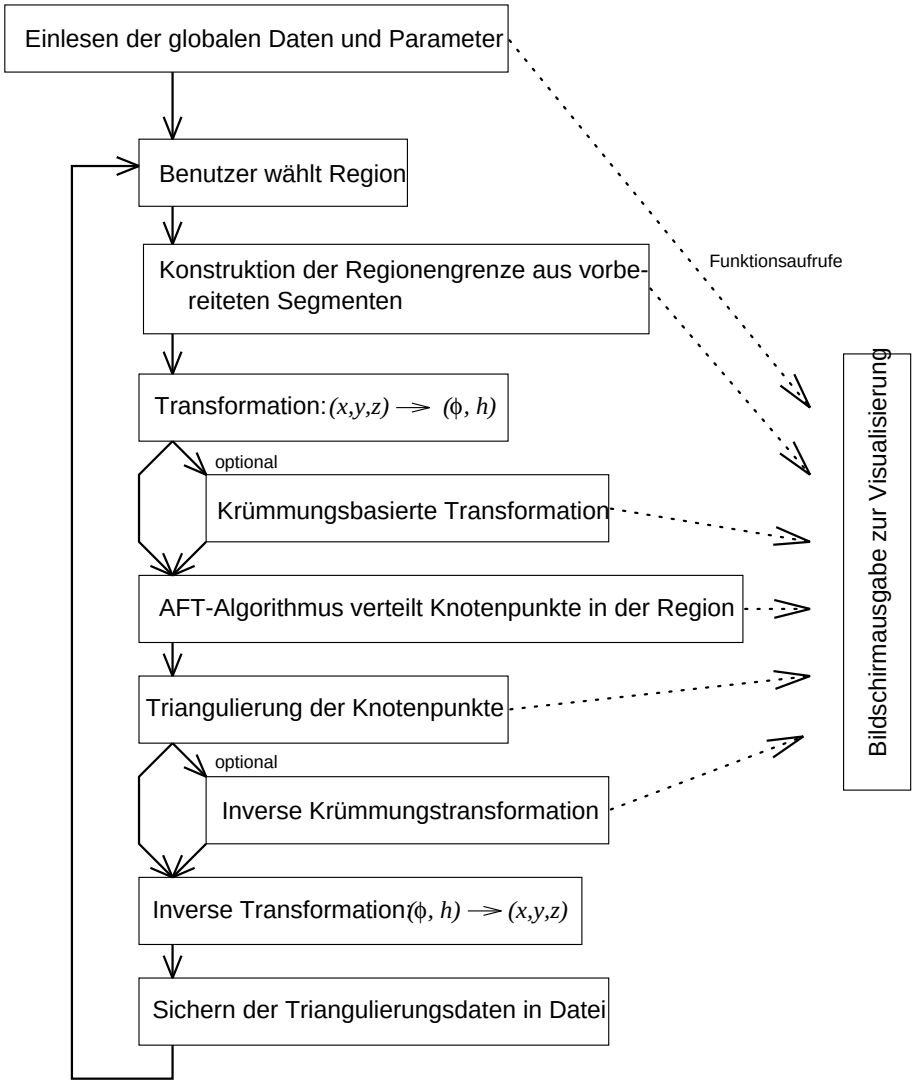


Abbildung 5.11: Ablaufplan der Software zur Triangulierung einer Region

5.2.1 Die Software zur Vorbereitung der Segmente

Die Testimplementation erlaubt noch keine integrierte Eingabe der Animationslinien mit automatischer Initialisierung der Segmente und Formulierung der Regionen. Der Benutzer wird aber bei diesen Arbeitsschritten (Abb. 5.12) von mehreren Hilfsprogrammen unterstützt.

Als Zeicheneditor zur Erstellung der Animationslinien wurde ein externes Bildbearbeitungsprogramm ausgewählt, das die Definition mehrerer Bildebenen gestattet (die Software "Photoshop" der Firma Adobe besitzt beispielsweise diese Möglichkeit). Zur Orientierung und zur Vorgabe des 512^2-Rasters der $(\phi, h) \Leftrightarrow$Zylinderkoordinaten wird das Bild der Textur in die erste Bildebene (*layer*) geladen. Jede Animationslinie wird in einer eigenen Bildebene über der Textur gezeichnet. Der Vorteil der Ebenentechnik besteht darin, daß der Benutzer die vollständige Kontrolle über die Lage der Animationslinien zueinander

(Anlage der Schnittpunkte) und auf der Textur hat. Trotzdem kann er durch Ausblendung aller fremden Ebenen auf einfache Weise Bilddateien erzeugen, die nur eine Linie enthalten.

Jede Animationslinie liegt nun als definierte Pixelmenge in einer 512^2-Bilddatei vor. Die Pixelkoordinaten entsprechen den $(\phi, h) \Leftrightarrow$Zylinderkoordinaten der Animationslinie auf der diskreten Scanoberfläche. Zur Überführung der Pixel in eine Liste sukzessiver Pixelkoordinaten wurde der Tracer-Algorithmus (in [DO91]) in einem Hilfsprogramm implementiert. Die Liste der sukzessiven Pixelkoordinaten wird vom Hilfsprogramm in einer Datei abgelegt. Der Benutzer spaltet in Kenntnis der Schnittpunktkoordinaten die Liste in einzelne Listen für jedes Segment. Die pixelweisen Segmentbeschreibungen sind Eingabe des Programms zur Bestimmung der Initialknotenpunkte auf dem Segment.

Die Initialisierung erfolgt krümmungsbasiert oder äquidistant mit dem in der Konzeption beschriebenen Verfahren. Bogenmaß und Krümmung werden über den diskreten Kurven in einer Näherung bestimmt. Die berechneten Positionen für die Initialknotenpunkte werden auf die nächste diskrete Position gerundet.
Die Initialknoten werden für jedes Segment in einer Datei abgelegt. Für jede Initialisierungsart entsteht ein eigenes Verzeichnis mit der entsprechenden Sammlung von Segmentdateien. Die Software zur Regionentriangulierung kann zur Konstruktion einer Regionenbegrenzung und interner zwingender Kantenzüge die Segmentdateien über den Dateinamen und das Verzeichnis eindeutig identifizieren.

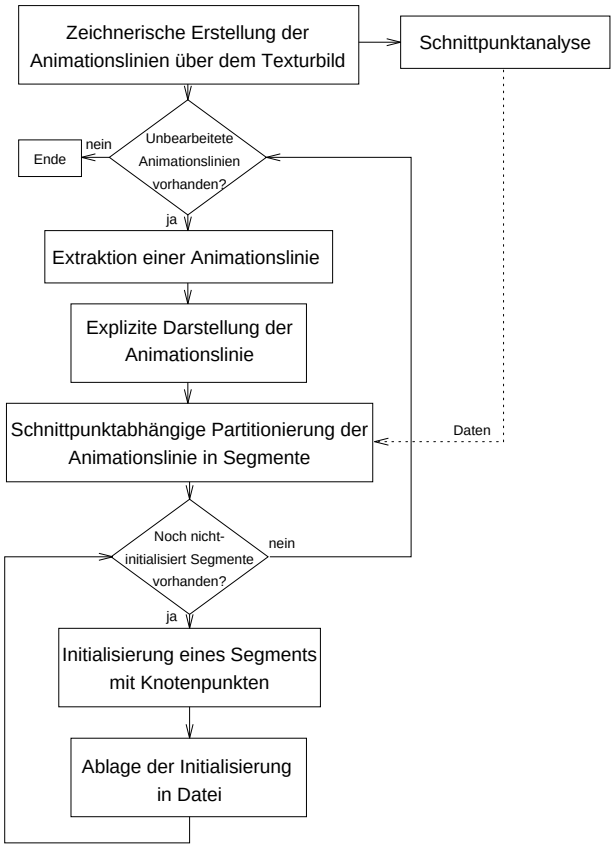


Abbildung 5.12: Arbeitsschritte zur Vorbereitung der Segmente

5.2.2 Implementation des Regionentriangulierers

Dieser Abschnitt gibt einen Überblick über die eingesetzten Datenstrukturen, die Dateitypen, die Implementierung der AFT-Methode und die Funktionen zur Bildschirmausgabe, die zur Realisierung des Ablaufplans aus Abb. 5.11 eingesetzt werden.

Datenstrukturen

Globale Datenstrukturen. Zur Laufzeit kann das Programm bis zu vier große globale, quadratische Felder unterhalten. Die Felder haben jeweils ein Größe von 512×512-Elementen und speichern Daten der Domain.

1. Die Scandaten der Figur bilden die Domain der Triangulierung und werden im Feld `data[lon][lat]` gespeichert. In Anlehnung an das geographische Koordinatensystem des Globus wird für ϕ die Bezeichnung `lon` (engl. *longitude*=Länge) und für h die Bezeichnung `lat` (engl. *latitude*=Breite) gewählt. Eingetragen werden die gemessenen Radien r als Fließkommatyp.

2. Sofern die Software in einem Modus gestartet wird, in dem die Krümmung an allen Oberflächenpunkten berechnet wird, nimmt das Feld `curvature[lon][lat]` die Krümmungswerte in Fließkommaformat auf. Bei der aufwendigen Methode zur Berechnung der maximalen Krümmung ist es wesentlich effizienter, bei Neustart des Programms die Daten aus der Datei einzulesen, als sie jedes Mal neu zu berechnen.

3. Falls die krümmungsbasierte Transformation eingesetzt wird, nehmen die Felder `map[lon][lat]` (Fließkommatyp) und `remap[lon][lat]` (ganzzahlig) die explizite Abbildungsvorschrift auf. Auch diese Werte können wahlweise aus einer zuvor erstellten Datei eingelesen werden.

Dynamische Datenstrukturen Bei der AFT-Methode stellen die Regionen ein hierarchisches System aus Front, Frontkomponenten, Kanten und Knotenpunkten dar, dessen Struktur ständig verändert wird. Zur Verwaltung dieser und weiterer Daten steht die dynamische Datenstruktur der linearen verketteten Listen zur Verfügung.

Der implementierte Datentyp einer verketteten Liste eignet sich zur Verwaltung von verketteten Listen (Typ: `llist`), Kanten (Typ: `edge`) oder Knotenpunkten (Typ: `node`). Die genannten Datentypen besitzen Speicherplätze für Zeiger auf Vorgänger und Nachfolger ihres Typs. Die verkettete Liste verwaltet ihre Elemente, indem sie diese Zeiger manipuliert. Ein Element kann infolgedessen Mitglied höchstens einer verketteten Liste sein.

Die Datentypen unterliegen einer klaren Hierarchie. Der elementare Datentyp, ein Punkt (Typ: `point`) mit drei nutzbaren Fließkommaspeicherstellen zur Aufnahme der kartesischen Koordinaten (x, y, z), wird vom Typ `node` referenziert. Verkettete Listen aus Elementen des Typs `node` treten nur behelfsweise auf (z.B. Übergabe der Initialknotenpunkte der Regionenbegrenzung). In den AFT-Datenstrukturen werden je zwei `node`-Elemente von Elementen des Typs `edge` referenziert. Eine verkettete Liste aus `edge`-Elementen stellt eine

Frontkomponente dar, und eine Front ist wiederum eine verkettete Liste aus Frontkomponenten.

Für die dynamische Datenstruktur sind folgende Basisoperationen definiert:

1. **new_list**(tp; L) initialisiert die verkettete Liste L (Typ: `llist`) als Liste für Elemente mit dem Typschlüssel $tp \in \{`n`, `e`, `l`\}$ (für `node`, `edge` oder `llist`). Ein Datenobjekt vom Typ `llist` enthält Speicherplatz für die Typangabe der verwalteten Elemente, die Elementanzahl und einen Zeiger auf das erste Element der Liste.

2. **new_point**($\Leftrightarrow$; P) kreiert im Speicher ein Datenobjekt vom Typ `point`. Analoge Initialisierungsoperationen existieren zur Initialisierung von Objekten des Typs `node`, `edge` und `llist`. Zum Löschen wird der Speicherbereich mit dem C-Funktionsaufruf `free` gelöscht.

3. **delete**($\Leftrightarrow$; L) löscht eine verkettete Liste, indem zuerst alle Elemente und dann die Listendatenstruktur gelöscht werden.

4. **elements**(L; n) ist die Anzahl der Elemente in der Liste.

5. **delete_no**(L, n; $\Leftrightarrow$) löscht die Listenzugehörigkeit des n-te Elements.

6. **add_edge**(L, E; L), **insert_edge_at**(L, E, n; L) fügt ein Datenobjekt vom Typ `edge` an eine Liste passenden Typs an, bzw. fügt es an der Position n ein. Analoge Operationen existieren für die Typen `node` und `llist`.

7. **get_edge_no**(L, n; E) gibt einen Zeiger auf die n-te Kante E in L zurück (analog für `node` und `llist`).

Die beiden Routinen des AFT-Algorithmus, die die Front aktualisieren, stellen gesonderte Operationen auf der verketteten Liste einer Frontkomponente dar. Bei der Aktualisierung der linearen Liste einer Frontkomponente berücksichtigen sie die zyklische Struktur der Frontkomponente.

8. **UPDATE·INTERNAL**(L, E, R; L) manipuliert die verkettete Liste L der Frontkanten in der aktuellen Frontkomponente. Über der Kante E wird ein neues Dreieck mit dem Punkt R errichtet. Die Funktion initialisiert die neuen Kanten F und G und versieht sie mit den korrekten Verweisen auf R, den Endknotenpunkt der Vorgängerkante (für F) und den Anfangsknotenpunkt der Nachfolgerkante (für G). Die Kante E wird aus der Liste und im Speicher gelöscht.

9. **UPDATE·BOUNDARY**(L, E, S; L_1, L_2) organisiert für den Fall, daß die neuen Kanten F und G beide nicht Bestandteil der Frontkomponenten sind, die Aufteilung der aktuellen Frontkomponente in zwei Frontkomponenten L_1 und L_2. Falls es nicht zur Spaltung der aktuellen Frontkomponente kommt, wird nur L_1 zurückgegeben.
   Die Operation berücksichtigt zusätzlich zu den in Abb. 4.10 gezeigten Fällen die Möglichkeit, daß der existierende Kandidat S nicht aus der Frontkomponente L, sondern aus einem regioneninternen zwingenden Kantenzug stammt.

Initialisierung der Regionenbegrenzung

Eine Region ist definiert über die Folge der Segmente, die ihre Begrenzung und inneren zwingenden Kantenzüge bilden (Abb. 5.13). Diese Angaben befinden sich in einer Regionendatei. Sobald der Benutzer zur Laufzeit des Programms die zu triangulierende Region spezifiziert hat, greift das Programm auf die entsprechenden Regionen- und Segmentdateien zu und initialisiert die Region.

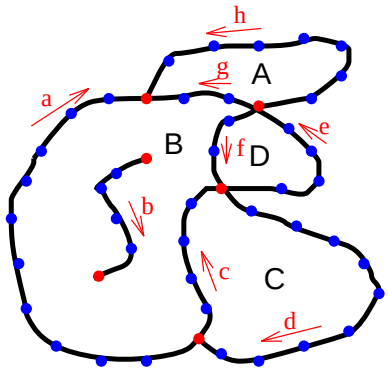


Abbildung 5.13: Initialisierung der Regionen
Die roten Punkte bezeichnen die Verbindungsknotenpunkte der Segmente. Die übrigen Initialpunkte auf der Regionenbegrenzung sind blau eingezeichnet. Die Region B setzt sich zum Beispiel aus der Segmentfolge $\{a, -g, f, -c, b\}$ und dem internen zwingenden Kantenzug b zusammen.

Implementierung der Advancing-Front-Triangulierung

Zur Triangulierung wurde eine einfache Version des AFT-Algorithmus aus Abschnitt 4.3.4 implementiert. Zur Berechnung der CDT wurde die frei verfügbare Software von CLINE und RENKA [Ren96] eingesetzt. Der FORTRAN-Code konnte mit einem Konverter [FGMS93] nach C übersetzt und als Modul eingebunden werden.

Die AFT-Implementierung besitzt folgende Merkmale:

- **GET·NEXT·EDGE**$(L\,;\,E)$ wählt die nächste Kante aus einer Frontkomponente von $L = \mathcal{C}$ zufällig aus. Damit soll ein Kompromiß zwischen den beiden Selektionsstrategien, die kürzeste und die längste Kante zuerst auszuwählen, gefunden werden. “Kürzeste-Kante-zuerst” sorgt für die schnelle Änderung der Elementgröße zur Anpassung an Details der Regionenbegrenzung. “Längste-Kante-zuerst” bildet besser geformte Dreiecke in Bereichen mit einheitlicher Kantenlänge aus [Far93].

- **UPDATE·INTERNAL**$(L, E, R\,;\,L)$ vollzieht zur Aktualisierung der CDT des ungemeshten Bereiches nach Konstruktion des neuen Dreiecks keine effiziente Retriangulierung der betroffenen Untermenge (siehe Abbildung 4.13), sondern des gesamten ungemeshten Bereiches. Es ist zu erwarten, daß die Untermenge nur wenig kleiner als der gesamte ungemeshte Bereich ist, weshalb der Effizienzverlust bei der Retriangulierung gering sein wird. Dafür entfällt der Verwaltungsaufwand für die Behandlung der Untermenge.

- Statt einer Funktion **ADD·TRIANGLE**$(\mathcal{T}, E, U\,;\,\mathcal{T})$, die zu jedem Zeitpunkt die Triangulierung des gemeshten Bereiches aufrechterhält, werden die neuen Knotenpunkte in der Domain in einer verketteten Liste gesammelt. Nach der Terminierung des AFT-Algorithmus wird aus allen Punkten und zwingenden Kantenzügen die CDT bestimmt.

- *unit_size* ist entweder konstant oder nimmt als Funktion keine unbeschränkt kleinen Werte an. Damit ist auch die maximale Zahl der Dreiecke in einer endlichen Region beschränkt. Daher entfallen die Funktion **MESH·SIZE·CONSTRAINTS** und die Datenstruktur *fin*.

Koordinatensystem

Die Oberflächenpunkte werden in zwei Koordinatensystemen repräsentiert. Zusätzlich zum zylindrischen Koordinatensystem werden für alle Distanz- und Winkelberechnungen normierte kartesische Koordinaten der Punkte berechnet.

Routinen zu Grafikausgabe

Die im Grafikmodul implementierten Routinen zur Bildschirmausgabe dienen der Visualisierung des Triangulierungsvorgangs. Durch kleine Veränderungen der Aufrufreihenfolge und anschließende Neukompilierung des Programms, stehen viele Variationen der Visualisierung zur Verfügung.

- **draw_point**$(color, lon, lat\,;\,\Leftrightarrow)$ zeichnet einen einzelnen Punkt in der Farbe *color* an die Koordinate (lon, lat).

- **draw_cross**$(color, lon, lat\,;\,\Leftrightarrow)$ zeichnet ein Kreuz in der Farbe *color* an die Koordinate (lon, lat).

- **draw_line**$(color, lon_1, lat_1, lon_2, lat_2\,;\,\Leftrightarrow)$ zeichnet eine Bresenham-Gerade [GB89] zwischen den Koordinaten (lon_1, lat_1) und (lon_2, lat_2).

- **draw_front**$(color, L\,;\,\Leftrightarrow)$ zeichnet alle Komponenten des aktuellen Frontkantenzugs.

- **draw_tri**$(color, Adj.List\,;\,\Leftrightarrow)$ zeichnet alle Dreiecke in der Adjazenzliste.

Ausgabedatei

Die Ausgabedatei enthält die (ϕ, h)-Koordinaten der Knotenpunkte der triangulierten Region, für initiale Knotenpunkte einen Verweis auf das Ursprungssegment und die Triangulierung in Form einer Adjazenzliste. Die Adjazenzliste enthält zu jedem Knotenpunkt eine Liste aller Knotenpunkte, mit denen der Knotenpunkt eine Kante bildet.

5.2.3 Verbindung der Regionentriangulierungen

Im Nachbearbeitungsschritt werden die triangulierten Regionen zu einem Animationsnetz zusammengefügt und in einem zu verfügbarer Animationssoftware kompatiblen Dateiformat abgelegt. Ein Hilfsprogramm erledigt diesen Arbeitsschritt der Datenkonvertierung von der Adjazenzlistenrepäsentation der Regionentriangulierungen zum kompatiblen Dateiformat. Das Dateiformat enthält eine Liste aller Knotenpunkte, gefolgt von einer

expliziten Liste aller Dreiecke. Ein Dreieck ist definiert als das Listenindextripel seiner Knotenpunkte. Der Benutzer startet das Programm "collect" mit folgenden Kommandozeilenoptionen:

- `BASENAME_SEGMENT_FILES` ist der Pfad und Basisname zu den Segmentdateien. Die Initialknotenpunkte der Segmente werden von "collect" zuerst eingelesen. Sie werden von mehreren Regionen gemeinsam genutzt. Bei der Reorganisation der Triangulierungsdaten wird über diese Knotenpunkte genau Buch geführt, damit sichergestellt ist, daß sie in der Knotenpunktliste nur einmalig auftreten.
- `BASENAME_TRIANG_FILES` ist der Pfad und Basisname zu den Regionendateien. Die Regionendateien enthalten Informationen zur Zuordnung der Randknotenpunkte zu den Ursprungssegmenten.
- `CYBERFILE` bezeichnet die Datei mit den Scandaten. Die Scandaten sind notwendig, um die (ϕ, h)-Koordinaten in kartesische Koordinaten zu konvertieren.
- `MESH_FILE` der gewünschte Dateiname.

Kapitel 6

Versuche und Ergebnisse

Dieses Kapitel dokumentiert anhand ausgewählter Ergebnisse das Verhalten der unterschiedlichen Konfigurationen der Testimplementierung. Die folgenden Abschnitte behandeln die äquidistante und krümmungsbasierte Initialisierung der Regionenbegrenzungen und die Triangulierung der Regionen nach den unterschiedlichen Strategien.

6.1 Initialisierung der Segmente

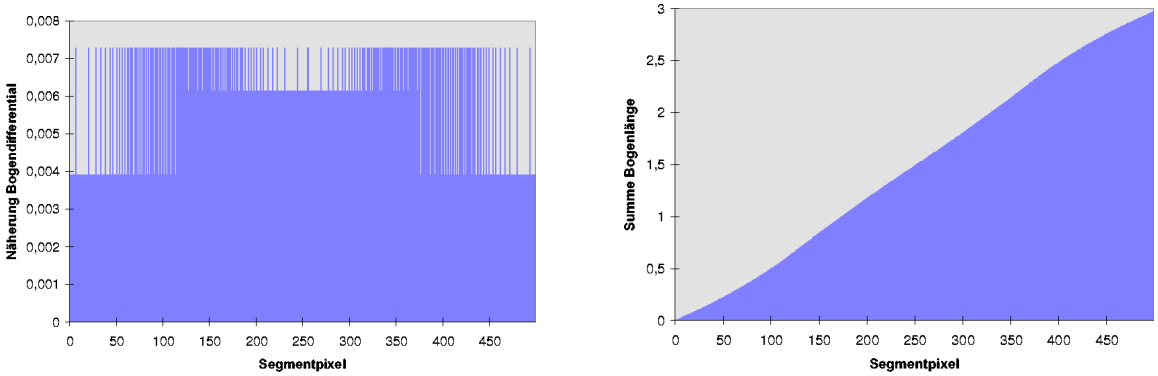
Mit Hilfe der Testläufe soll es möglich sein, die Umsetzung des äquidistanten oder krümmungsbasierten Initialisierungskonzepts in der diskreten Geometrie der Software zu bewerten.

6.1.1 Äquidistante Initialisierung

Das als diskrete Pixelfolge vorliegende Segment wird in äquidistante Intervalle unterteilt und die Initialknotenpunkte an die Intervallgrenzen gesetzt. Die Distanz ist eine diskrete Näherung an das Bogenmaß. In der Näherung berechnet sich die Bogendistanz zwischen zwei Pixeln der Kurve als Summe der kartesischen Einzeldistanzen aller direkt aufeinanderfolgenden Pixel, die zwischen den betrachteten Pixeln liegen.

Zur Überprüfung der Berechnungsmethode für das Bogenmaß wurde in Abb. 6.1 zu jedem Pixel eines Halbkreises (Abb. 6.2) der kartesische Abstand zum Vorgängerpixel aufgetragen. Der Halbkreis wurde mit dem Bresenham-Algorithmus für Kreise [GB89] in die diskrete 512^2-Pixelebene gezeichnet und umfaßt 500 Pixel. Das Programm zur Bestimmung der Initialknotenpunkte eines Segments lädt für den Testlauf keine reale Laserscandatei, sondern initialisiert sein Scandatenfeld mit dem halben maximalen Scanwert. Der Halbkreis wird als Pixelliste in $(\phi, h)\Leftrightarrow$Zylinderkoordinaten eingelesen, und jede Koordinate wird um den Radius $r = 0,5$ ergänzt. Dadurch erscheint der Kreis als Projektion auf einen Zylindermantel und wird elliptisch verzerrt. Die spitzere Seite der Ellipse befindet sich an der Stelle des Kreises mit waagerechter Tangente. Mit der Projektion auf die Zylinderoberfläche geht beim Übergang in diesen Bereich außerdem eine geringfügige Vergrößerung des Pixelabstands einher.

In Abbildung 6.1 (a) sind für jedes Pixel die Abstände zum Vorgängerpixel aufgetragen. Im Pixelraster liegt ein Nachfolgepixel entweder an einer Seite oder einer Ecke des



(a) Kartesische Distanz zum Vorgängerpixel (b) Summe der Pixeldistanzen

Abbildung 6.1: Diskrete Bogenmaßapproximation für einen Halbkreis

betrachteten Pixels an. Durch die Projektion auf die Zylinderoberfläche haben sich die Abstände für Pixel, die links und rechts anliegen (bzgl. h-Achse), leicht erhöht. Damit existieren insgesamt nur drei mögliche Abstände zwischen zwei aufeinanderfolgenden Pixeln. Durch den kleinen Wertebereich stellt der kartesische Abstand zum Vorgängerpixel nur eine extrem grobe Approximation des Bogenlängendifferentials an einer Stelle des Liniensegments dar.
Abbildung 6.1 (b) zeigt, daß sich durch Summation trotzdem eine brauchbare Approximation der Bogenlänge zwischen zwei ausreichend weit auseinanderliegenden Pixeln bestimmen läßt. Die Summation gleicht die Sprünge zwischen den lokalen Meßwerten aus und führt zu einer realistischen Wiedergabe der Bogenlänge über den Halbkreis. Beispielsweise ist im mittleren Bereich eine etwas höhere Steigung zu beobachten, was der erwarteten Auswirkung der Projektion des Halbkreises auf die Zylinderoberfläche entspricht.

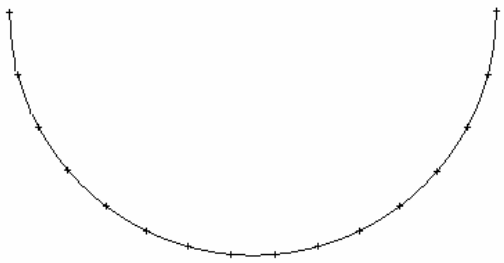


Abbildung 6.2: Äquidistante Initialknoten auf einem Halbkreis

Die Abbildung 6.2 zeigt die Initialisierung des Halbkreises mit 16 Knotenpunkten und in Abbildung 6.3 ist die äquidistante Initialisierung aller Segmente sichtbar. Die gewählte Länge der Intervalle hängt von den benachbarten Regionen ab. Der Einfluß der Initialisierung auf das Ergebnis der Triangulierung der Region wird im nächsten Abschnitt betrachtet.

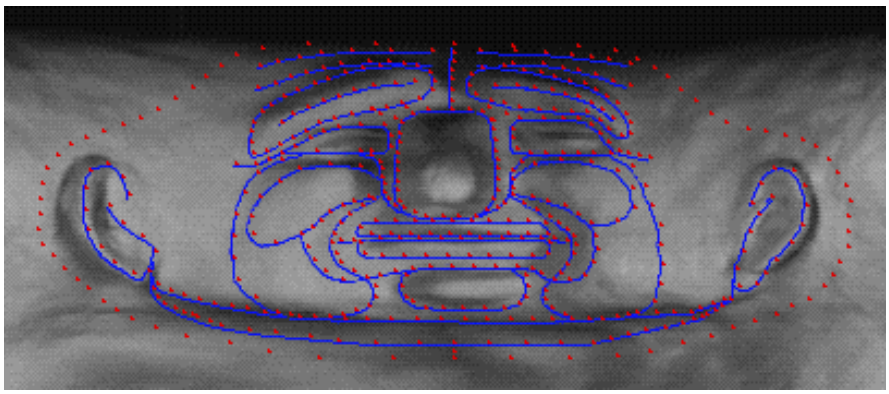


Abbildung 6.3: Initialknoten auf Animationslinien
Äquidistante Initialisierung der Segmente aller Animationslinien. Jeder Schnittpunkt von zwei Animationslinien wird mit einem Knotenpunkt besetzt.

6.1.2 Krümmungsbasierte Initialisierung

Die krümmungsbasierte Initialisierung teilt ein Segment in Intervalle mit konstanter Summe der Krümmungswerte auf. Statt der diskreten Approximation des Bogendifferentials an einem Pixel des Segments, wird die diskrete Approximation der Krümmung (Abb. 5.6) verwendet. Zur Erhöhung der Auflösung greift der Krümmungsoperator auf eine 5er-Pixelumgebung zu. Statt (P_{i-1}, P_i, P_{i+1}) (3er-Pixelumgebung) werden die Pixel (P_{i-2}, P_i, P_{i+2}) betrachtet.
Zur Untersuchung des Verhaltens des Krümmungsoperators für diskrete Kurven im (ϕ, h, r)-Koordinatensystem werden verschiedene Testdatensätze herangezogen.

Zu Beginn wird in der diskreten (ϕ, h)-Ebene eine waagerechte Linie, bestehend aus 414 Pixeln, betrachtet. Die Linie erscheint im (ϕ, h, r)-Raum auf einer Zylinderoberfläche mit $r_1 = 0,5$ und $r_2 = 1,0$ ($1,0$ ist der maximale Radius) und wird dadurch zu einem Kreissegment. Ein Kreis besitzt im Kontinuierlichen konstante Krümmung. Da sich die Kreisradien um den Faktor 2 unterscheiden, besitzt der äußere Kreis die halbe Krümmung. In Abbildung 6.4 ist für beide Kreissegmente die laufende Summe der gemessenen Krümmung entlang des Pixelindex aufgetragen. Der Krümmungsoperator gibt für jeden Kreis konstante Werte zurück (die Steigung im Summendiagramm ist linear) und unterscheidet sich korrekt um den Faktor zwei.

Sobald das Liniensegment auch in der (ϕ, h)-Ebene gekrümmt ist, liefert der Krümmungsoperator stark schwankende Werte. In Abbildung 6.5 (a) ist der ermittelte Krümmungswert für den bekannten Halbkreis aufgetragen. Für alle betrachteten 5er-Pixelfolgen, die in der (ϕ, h)-Ebene keine Linie bilden, entstehen hohe Werte. Unter Vernachlässigung der Nachkommastellen entstammen diese Werte einem kleinen Wertebereich. Der Wertebereich spiegelt die möglichen Konfigurationen der 5 Pixel im (ϕ, h)-Raster wieder. Die Krümmungswerte für 5er-Pixelfolgen, die in der (ϕ, h)-Ebene eine Linie bilden und die Krümmung der Zylinderoberfläche verfolgen, sind um ein Vielfaches geringer und haben im Diagramm kein sichtbares Gewicht.

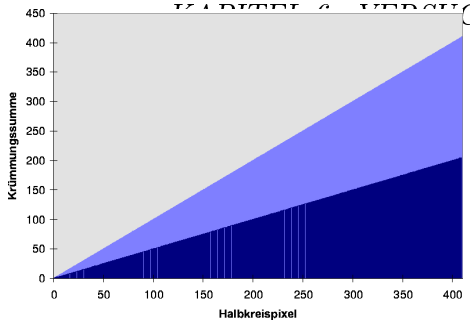
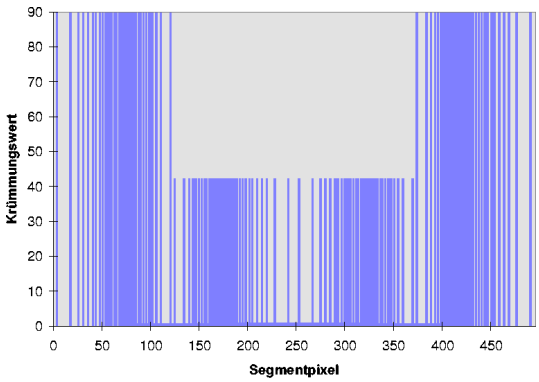


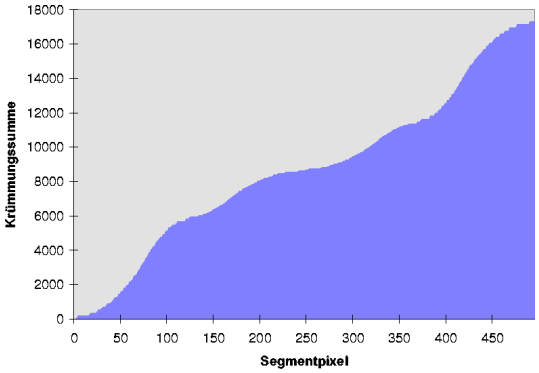
Abbildung 6.4: Krümmungssummen für zwei Kreissegmente
Der Krümmungsoperator liefert in diesem Beispiel korrekte Werte. Die Krümmungen sind konstant und unterscheiden sich um den Faktor 2.

Aus den diskreten Krümmungswerten sind keine direkten Rückschlüsse auf die Krümmung des kontinuierlichen Liniensegments an dieser Stelle möglich. Nur über die Dichte der Spitzenwerte, bzw. eine Summation der Werte über ein ausreichend großes Intervall, lassen sich Aussagen über die Krümmungsverhältnisse der Linie machen.

Abbildung 6.5 (b) zeigt die laufende Summe der Krümmungswerte. Die Spitzen sind verschwunden. Trotzdem werden die Krümmungsverhältnisse des Halbkreises verfälscht. Alle Zonen, die die Pixelebene diagonal durchlaufen, werden überbetont. Die Basis des Halbkreises erhält durch die elliptische Verzerrung bei der Projektion auf die Zylinderoberfläche eine höhere Krümmung, was im Diagramm nicht widergespiegelt wird.



(a) Berechneter Krümmungswert.

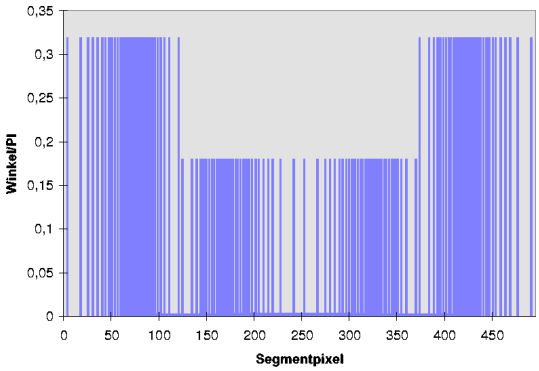


(b) Krümmungssumme über die Pixel des Halbkreises.

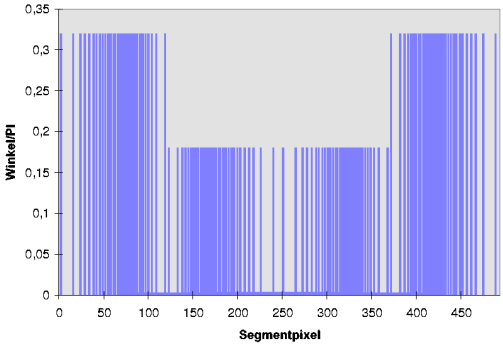
Abbildung 6.5: Ergebnisse des Krümmungsoperators für elliptische Linie

Wie aus Abbildung 6.6 hervorgeht, ändert die Vergrößerung der Pixelumgebung auf 7 Elemente (P_{i-3}, P_i, P_{i+3}) kaum den qualitativen Verlauf des Winkeldiagramms (Zähler des Operators). Die Auswirkung einer weiteren Vergrößerung der Pixelumgebung wird nicht untersucht, da die Segmente der Animationslinien einer typischen Figur zu kurz sind, um größere Operatoren zu erlauben. Das Histogramm in Abbildung 6.7 zeigt, daß die typischen Elemente weniger als 50 Pixel umfassen.

Der nächste Testdatensatz hat die Form einer Schlange und ist in der (ϕ, h)-Ebene aus sechs identischen Halbkreisen zusammengesetzt (Abb. 6.10 (a)). Abgesehen von den Nahtstellen zwischen zwei Halbkreisen, die die Unstetigkeitsstellen der Kurve darstellen und an



(a) Krümmungswinkel, gemessen mit (P_{i-2}, P_i, P_{i+2})-Operator.



(b) Krümmungswinkel, gemessen mit (P_{i-3}, P_i, P_{i+3})-Operator.

Abbildung 6.6: Krümmungswinkel am Halbkreis

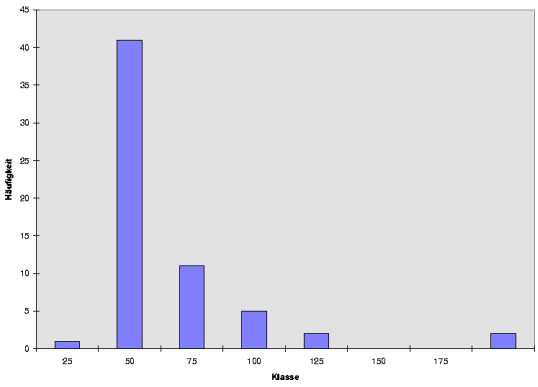
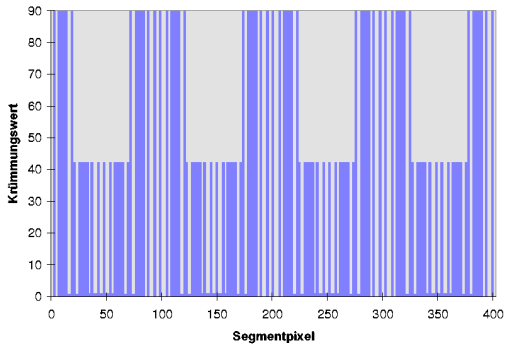


Abbildung 6.7: Histogramm der Segmentlängen
Histogramm der Längen der Animationslinienseg-mente einer Figur, gemessen in Pixeln der (ϕ, h)-Ebene.

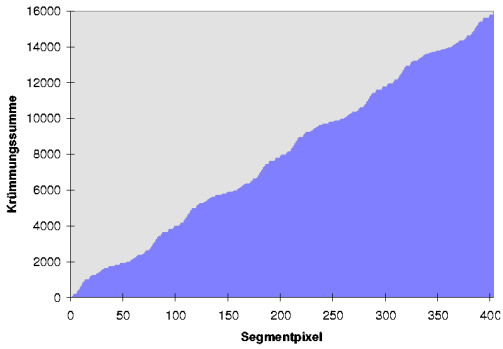
denen die Krümmung nicht definiert ist, besitzt das Liniensegment relativ gleichbleibende Krümmung. Nur durch die Projektion auf die Zylinderoberfläche ($r = 0,5$) vergrößert sich die Krümmung leicht, je mehr sich die Tangente an die Schlangenlinie aus der parallelen Lage zur h-Achse wegdreht.

Im Krümmungs- und Krümmungssummendiagramm (Abb. 6.8) werden die Krümmungsverhältnisse der kontinuierlichen Ausgangslinie ebenfalls unzureichend dargestellt. Alle Bereiche, die im (ϕ, h)-Pixelraster waagerechte Orientierung besitzen, werden durch zu niedrige Werte wiedergegeben.

Ein weitere Testlinie ist in der (ϕ, h)-Ebene aus zwei Geraden und einer fortschreitend gekrümmten Linie zusammengesetzt (Abb. 6.10 oben). Zwischen den Zonen befinden sich Knickstellen mit hoher Krümmung. In den Diagrammen (Abb. 6.9) sind die Knickstellen als Wertespitzen im Krümmungsdiagramm und Sprünge im Summendiagramm erkennbar. Die Geraden besitzen gleichmäßige Krümmungswerte auf niedrigem Niveau. Ihre Krümmung stammt von der Projektion auf die gekrümmte Oberfläche. Die diagonale Li-



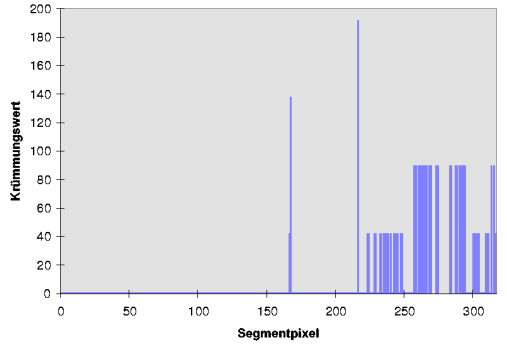
(a) Berechnete Krümmungswerte.



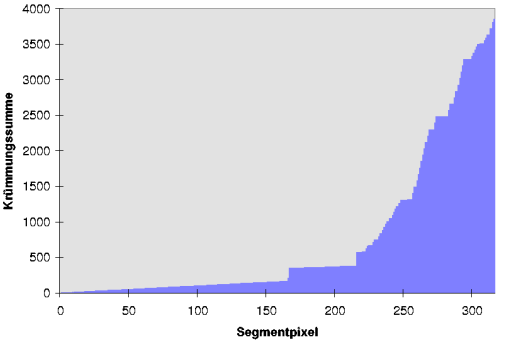
(b) Fortlaufende Krümmungssumme.

Abbildung 6.8: Ergebnisse des Krümmungsoperators für die schlangenförmige Linie

nie verfolgt die Oberflächenkrümmung weniger schnell und besitzt daher eine niedrigere Krümmung, was in den berechneten Krümmungswerten korrekt wiedergegeben wird. Die Krümmungen, die durch Lageänderungen im (ϕ, h)-Pixelraster zustande kommen, führen zu ungleichmäßig verteilten Spitzenwerten.



(a) Berechneter Krümmungswert an jedem Pixel der inhomogenen Linie.



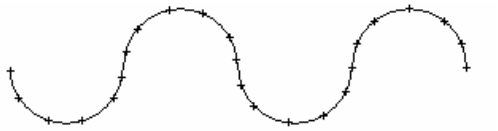
(b) Fortlaufende Summation der Krümmungswerte.

Abbildung 6.9: Ergebnisse des Krümmungsoperators für die inhomogene Linie

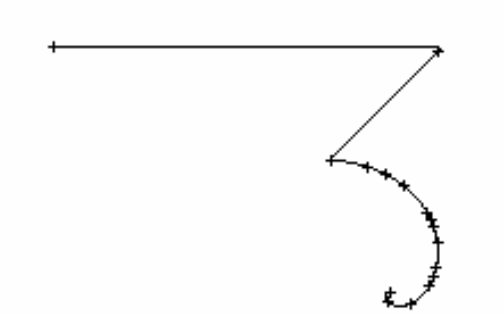
Das Ziel der Initialisierung ist, die Knotenpunkte in gleichmäßigem Basisabstand zu verteilen und die Verteilung in Zonen hoher Krümmung zu verdichten. Wie die Abbildungen in der linken Spalte von Abbildung 6.10 zeigen, führt das Initialisierungsverfahren mit den ermittelten Krümmungswerten zu unbrauchbaren Knotenpunktverteilungen. Die Datengrundlage, gewonnen mit Hilfe des Krümmungsoperators, liefert zu niedrige Werte für Zonen mit reiner Oberflächenkrümmung. Bei reiner Oberflächenkrümmung entstehen die Krümmungswinkel durch Veränderung von r. Krümmungen in der (ϕ, h)-Ebene werden

durch diskrete Richtungsänderungen im (ϕ, h)-Raster verursacht und besitzen typische Werte, die um den Faktor 40 bis 80 höher liegen können, als bei der reinen Oberflächenkrümmung.
In der rechten Spalte von Abbildung 6.10 sind die Ergebnisse der Initialisierung nach Anheben der niedrigen Krümmungswerte zu sehen. Die Verteilung der Knotenpunkte fällt deutlich gleichmäßiger aus, aber ähnelt bereits sehr einer äquidistanten Verteilung.

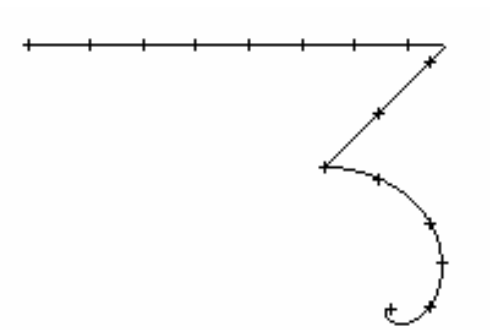
(a) Initialisierung der Schlangenlinie mit 24 Initialknotenpunkten auf der Basis unkorrigierter Krümmungswerte.



(b) Initialisierung der Schlangenlinie mit 24 Initialknotenpunkten nach Anheben der niedrigen Krümmungswerte.



(c) Initialisierung der inhomogenen Linie mit 16 Initialknotenpunkten auf der Basis unkorrigierter Krümmungswerte.

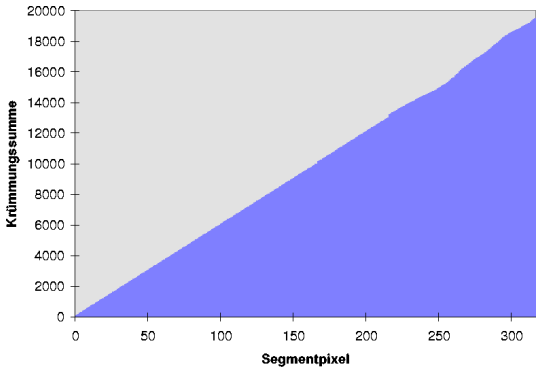


(d) Initialisierung der inhomogenen Linie mit 16 Initialknotenpunkten nach Anheben der niedrigen Krümmungswerte.

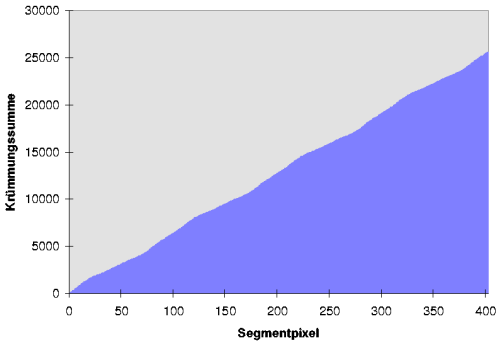
Abbildung 6.10: Initialisierung der Testsegmente

Die Korrektur der Krümmungswerte (Werte unter 2.0 werden auf 60 angehoben) verhilft den Krümmungssummendiagrammen der Testlinien (Abb. 6.11) zu linearer Charakteristik. Im Fall der Schlangenlinie spiegelt das lineare Summendiagramm wesentlich besser die tatsächlichen Krümmungsverhältnisse der Kontur wieder, während das lineare Summendiagramm der inhomogenen Linie kaum noch Rückschlüsse auf die Krümmungsverhältnisse zuläßt, und die resultierende Dichte der Initialknotenpunkte wenig mit der tatsächlichen Krümmungsverteilung korrespondiert.

Im Testlauf mit einem Animationsliniensegment aus der Praxis stellt sich ebenfalls eine beliebig erscheinende Verteilung ein (Abb. 6.12). Die Schwankungen der Bogenlänge der Intervalle um einen mittleren Wert korrespondieren nicht immer mit der Krümmung im Intervall. Die Zahl der Pixel in jedem Intervall ist so gering, daß auch durch Summation der diskreten Werte keine adäquate Näherung der tatsächlichen Krümmung ableitbar sein dürfte.



(a) Krümmungssumme über der Schlangenlinie nach der Korrektur.



(b) Krümmungssumme über der inhomogene Linie nach der Korrektur.

Abbildung 6.11: Ergebnisse des Krümmungsoperators für die inhomogene Linie

Abbildung 6.12: Krümmungsbasierte Initialisierung eines Animationsliniensegments

6.2 Triangulierung

Die Güte der Triangulierung für ein Animationsnetz läßt sich nur unzureichend mit wenigen quantitativen Aussagen zusammenfassen. Der optische, dreidimensionale Eindruck der Gesamtkonfiguration aus Knotenpunkten und Kanten erlaubt, verglichen mit Statistiken über geometrische Eigenschaften der Dreieckselemente, eine schnellere und ausreichende Bewertung der Netzqualität im Hinblick auf den komplexen Anforderungskatalog. In diesem Abschnitt werden daher die Triangulierungsergebnisse, die mit den unterschiedlichen Konfigurationen der Software erzielt werden, in Form von Bildschirmkopien der 2D- und 3D-Ansichten typischer Regionen präsentiert. Zu jeder Triangulierungsmethode werden charakteristische Ergebnisse ausgewählt.

Die Beispielfigur wird von den Animationslinien in 18 Regionen partitioniert (Abb. 6.13). Einige Regionen enthalten zwingende innere Kantenzüge (constraints). Diese Regionen werden nacheinander mit folgenden Ansätzen trianguliert.

- Die Konstruktion der Dreiecke findet im zweidimensionalen (ϕ, h)-Raum statt. *unit_size* ist in der (ϕ, h)-Ebene konstant.
- Der AFT-Algorithmus arbeitet in der krümmungskorrigierten (ϕ, h)-Ebene. Nach der Terminierung des AFT-Algorithmus werden die gesetzten Knotenpunkte in der

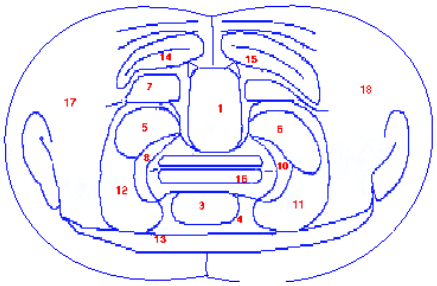


Abbildung 6.13: Regionen im Gesicht der Beispielfigur
Die Animationslinien partitionieren das Gesicht der Beispielfigur in 18 Regionen.

ursprünglichen (ϕ, h)-Ebene neu CDT-trianguliert.

- Der AFT-Algorithmus konstruiert den neuen Kandidatenpunkt für das nächste Dreieck auf der 3D-Oberfläche. Die Konstante *unit_size* wird also als echte 3D-Distanz realisiert. Das Ziel ist, auf der 3D-Oberfläche wohlgeformte Dreiecke zu erhalten.
- Die 3D-Konstruktion der Dreiecke wird mit einer krümmungsabhängigen *unit_size* kombiniert.

6.2.1 Einheitliche Eigenschaften und Grenzen des Triangulierungskonzepts

An Beispielen der einfachen Triangulierung in der (ϕ, h)-Projektionsebene mit konstanter *unit_size* werden in diesem Abschnitt die grundsätzlichen Eigenschaften der Triangulierung dokumentiert. Dazu zählen die Bedeutung der zufallsgesteuerten Kantenwahl für das interaktive Optimieren einer Regionentriangulierung, die Berücksichtigung regioneninterner zwingender Kantenzüge, die Verbindung triangulierter Regionen zu einem Gesamtnetz und die Ausbildung von Flußrichtungen. Anschließend wird auf die unberücksichtigt gebliebenen Aspekte des weichen Übergangs zwischen unterschiedlich dimensionierter Kantenlänge auf dem begrenzenden Segment und innerhalb der Region (*unit_size*), sowie die mangelnde Interaktion benachbarter Segmente bei der Initialisierung eingegangen.

Einfluß der zufallsgesteuerten Kantenwahl

Die Abbildungen 6.14 zeigen variierende Ergebnisse der Triangulierung der Nasenregion. Die zufällige Auswahlstrategie von **GET·NEXT·EDGE** führt in jedem Durchlauf zu anderen Netzknotenpunkten. Die geringe Laufzeit für eine Triangulierung erlaubt dem Benutzer im interaktiven Betrieb, wiederholt neue Prototypen einer Regionentriangulierung erstellen zu lassen, um sich schließlich für eine Triangulierung zu entscheiden.

Berücksichtigung regioneninterner zwingender Kantenzüge

In die Definition einer Region können Liniensegmente als interne zwingende Kantenzüge aufgenommen werden. Abbildung 6.15 zeigt ein Beispiel für die korrekte Berücksichtigung eines internen Kantenzuges in einer Region.

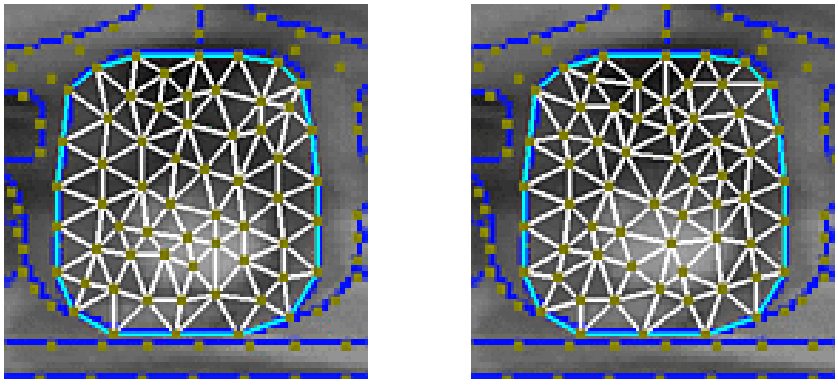


Abbildung 6.14: 2D-Ansicht variierender Triangulierungen durch zufallsgesteuerte Kantenwahl
Die zufallsgesteuerte Wahl der nächsten Kante führt in jedem Durchlauf zu einem veränderten Dreiecksnetz.

Verbindung von Regionentriangulierungen zu einem Gesamtnetz

Da sich zwei benachbarte Regionen ein Segment teilen, und keine der beiden während der Triangulierung die Knotenpunkte auf dem Segment ändert, existiert ohne gesonderte Nachbearbeitung eine passende Schnittstelle zwischen allen Regionentriangulierungen. Das Zusammenfügen der Regionen ist lediglich eine Aufgabe der Datenreorganisation der vorhandenen Knotenpunkte und Dreiecke. Abbildung 6.16 zeigt das Ergebnis nach Verbindung der 18 Testregionen.
Das dargestellte Netz besteht aus 701 Knotenpunkten, deren Verbindungskanten 1317 Dreiecke bilden. Der AFT-Algorithmus hat nur 182 Steiner-Punkte bestimmt. Die restlichen Knotenpunkte stammen von der Initialiserung der Testsegmente.

Ausbildung von Flußrichtungen

Sukzessive, in Schichten aneinandergelegte, gleichförmige Elemente vererben die Form der Regionenbegrenzung an die nächste Schicht und bilden damit die als Flußrichtung bezeichneten Strukturen aus. In den großflächigen Regionen in Abbildung 6.16 kann die AFT-Methode dieses Verhalten entfalten. Jedes neue Dreieck weicht wenig von der Idealform ab und fügt sich harmonisch in die Parkettierung ein.

Bei engen und komplizierten geometrischen Randbedingungen können sich keine gleichmäßigen Netzstrukturen mehr ausbilden. Flußrichtungen sind darin nicht erkennbar.

Abgestufter Übergang zwischen der Kantenlänge auf einem Liniensegment und in der Region

Die Kantenlänge auf einem begrenzenden Segment und innerhalb einer Region kann unter Umständen stark voneinander abweichen. Der Triangulierung in der Region kommt im Sinne der Wohlgeformtheit der Dreieckselemente die Aufgabe zu, einen weichen Übergang zwischen den Kantengrößen auszubilden. Abbildung 6.17 zeigt in 2D-Ansicht ein Beispiel für die Entwicklung von Dreiecken mit großer *unit_size* über kurzen Segmentkanten. Falls

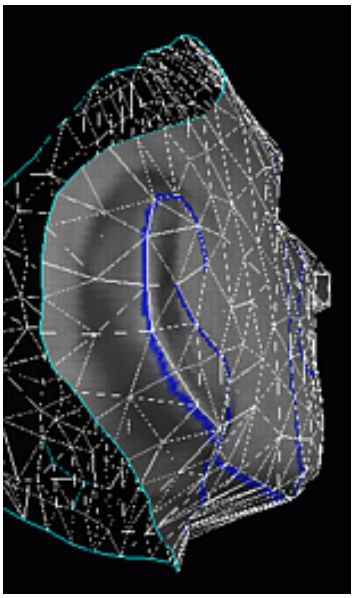


Abbildung 6.15: Zwingender Kantenzug in einer Region
Die Region Nr. 17 der Testfigur enthält eine Animationslinie für die Ausbildung des rechten Ohres. Nur durch Berücksichtigung der Linie bei der Triangulierung wird die Form des Ohres mit wenigen Dreiecken ausreichend nachgebildet, und das Ohr wird animierbar.

sich die Front auf das begrenzende Segment zubewegt, kann die Auswahlstrategie zwischen R, S und S_C (Abb. 4.12) einen weichen Übergang ausbilden.

Interaktion benachbarter Segmente bei der Initialisierung

Das Konzept der äquidistanten, bzw. krümmungsbasierten Initialisierung betrachtet die Segmente unabhängig voneinander. Für eine gute Triangulierung erweist es sich aber als notwendig, bei benachbarten Segmenten die Lage der Knotenpunkte aufeinander abzustimmen. Das trifft insbesondere auf näherungsweise parallel verlaufende benachbarte Segmente zu.

In Abbildung 6.18 befinden sich zwei parallele Segmente in engem Abstand. Die Verknüpfung von zwei Bedingungen verhindert die Ausbildung eines wohlgeformten Netzes in der schmalen Zone zwischen den parallelen Segmenten. Erstens befinden sich die Knotenpunkte auf den benachbarten Segmenten nicht in ausreichend versetzter Lage zueinander, so daß bereits die CDT der ungemeshten Region aus ungünstigen Dreiecken mit einem Winkel nahe 90° besteht. Zweitens befinden sich die Kantenlängen auf den Segmenten und der Abstand der Segmente in einem ungünstigen Verhältnis zueinander, so daß die AFT-Methode bei der Konstruktion neuer Dreiecke auf die ungünstige Anfangs-CDT der ungemeshten Region zurückgreifen muß. Neue wohlgeformte Dreiecke werden nur entstehen, wenn die *unit_size* deutlich kleiner als der Segmentabstand ist. Eine zu große *unit_size*

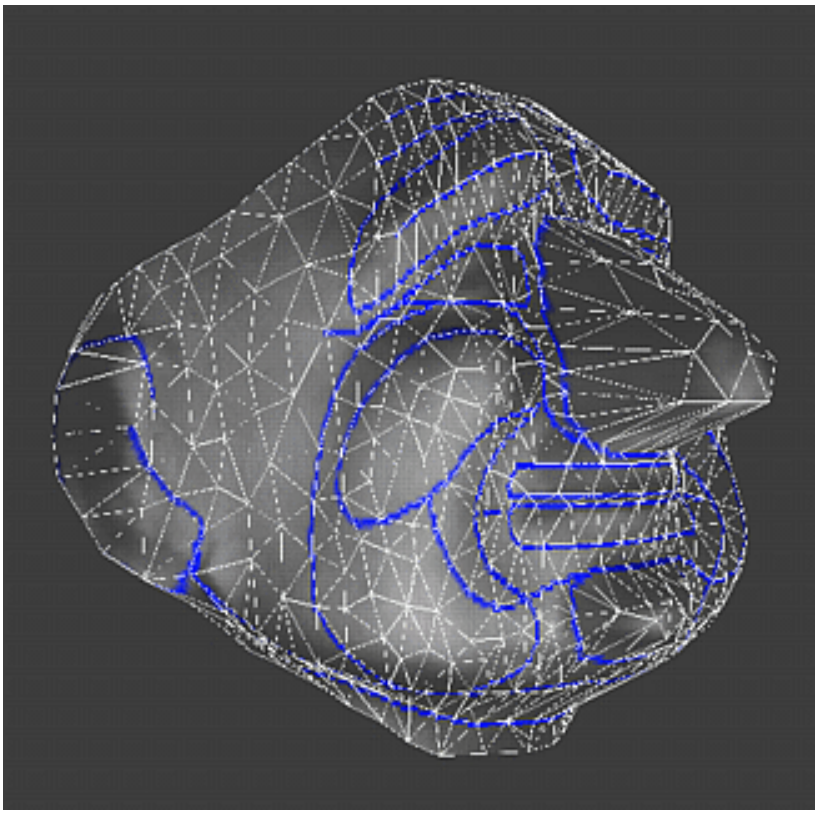


Abbildung 6.16: Zusammenfügen von Regionentriangulierungen
Ein vollständiges Animationsnetz entsteht durch Zusammenfügen der Regionentriangulierungen. Die in der Vorverarbeitungsstufe initialisierten Liniensegmente sind die Schnittstellen der Regionen und dürfen während der Triangulierung nicht verändert werden. In der texturierten Abbildung wurden die Testregionen zu einem Gesichtsnetz verbunden.

passiert nicht den Test **CHECK·VISIBLE**, da der neue Punkt außerhalb der Region liegt (Abb. 6.18).

Die Ausführungen machen deutlich, wie entscheidend die Wahl einer guten Segmentinitialisierung für die Qualität der Triangulierung ist.

6.2.2 Eigenschaften der 2D-Triangulierung mit konstanter *unit_size*

Die 2D-Triangulierung mit konstanter *unit_size* stellt den einfachsten Triangulierungsmodus der Software dar. Wie das Bildmaterial im vorhergehenden Abschnitt zeigt, können in diesem Modus in einigen Arealen brauchbare Triangulierungen erzeugt werden. Sobald jedoch die Oberflächennormale von der Scanrichtung abweicht (d.h. die Koordinate r ändert sich signifikant), werden in der (ϕ, h)-Ebene konstruierte, wohlgeformte Dreiecke in der (ϕ, h, r)-Ebene spitzwinklig verzerrt.

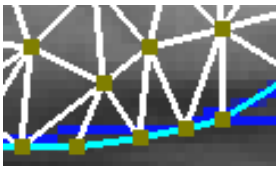


Abbildung 6.17: Harter Übergang der Kantenlänge
unit_size ist bei der Konstruktion eines Dreiecks über einer Segmentkante nicht abhängig von der Segmentkantenlänge.

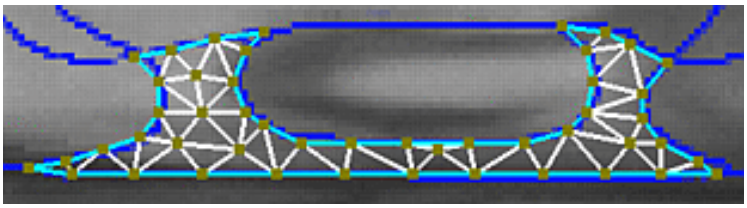


Abbildung 6.18: Ungünstige Abstimmung der Initialisierung benachbarter Segmente
Unter diesen geometrischen Bedingungen kann sich keine wohlgeformte Netzstruktur ausbilden.

Die Abbildung 6.20 (b) zeigt deutlich verzerrte Dreiecke an den Flanken der Nase. In der 2D-Ansicht (Abb. 6.14) sind die Dreiecke CDT-wohlgeformt.

6.2.3 Ergebnisse mit der Krümmungstransformation

Die Krümmungstransformation bewirkt eine krümmungsabhängige Erhöhung der Knotenpunktdichte und ist eine Maßnahme zur Verbesserung der Oberflächenapproximation. Die 2D-Koordinaten der (ϕ, h)-Ebene werden abhängig von der Krümmungssumme zwischen zwei Punkten manipuliert. Der AFT-Algorithmus arbeitet in der transformierten (ϕ, h)-

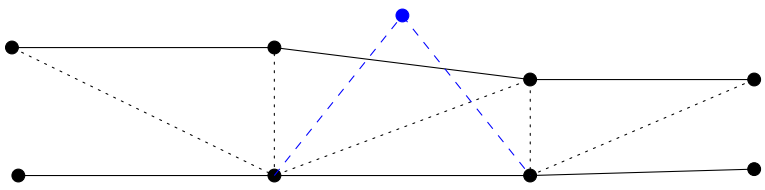
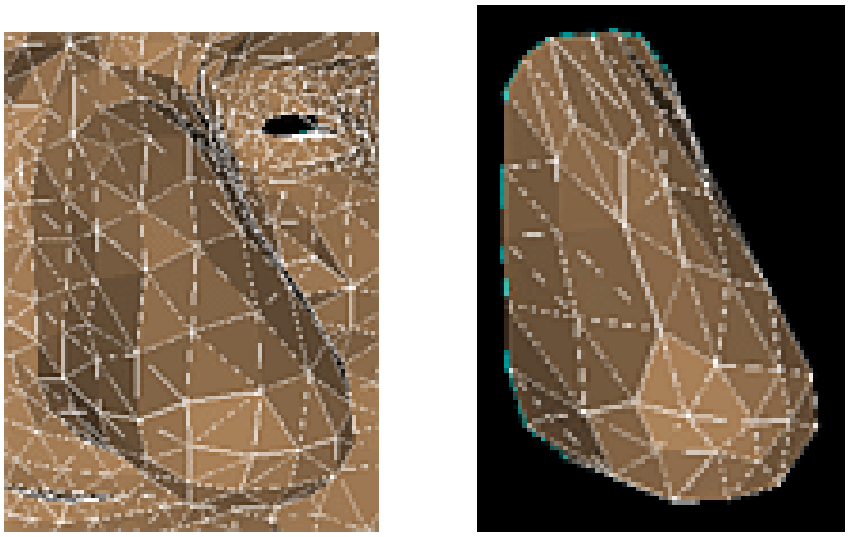


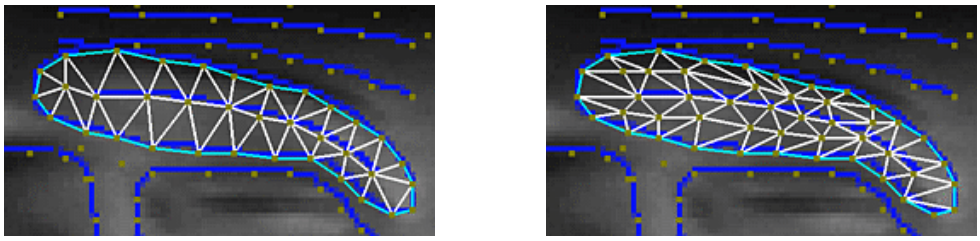
Abbildung 6.19: Ungünstiges Verhältnis der Größen bei parallelen Segmenten
unit_size ist im Verhältnis zum Segmentabstand zu groß. Es wird auf das nahezu rechtwinkelige CDT-Dreieck aus der gepunktet gezeichneten Grundtriangulierung zurückgegriffen.



(a) (b)

Abbildung 6.20: Manuell und automatisch in 2D erstelltes Netz der Nasenregion
Das Netz in (a) wurde manuell konstruiert. In (b) ist ein Ergebnis der automatischen Triangulierung in 2D mit konstanter *unit_size* abgebildet.

Ebene. Die erzeugten Knotenpunkte werden anschließend in die (ϕ, h)-Ebene zurücktransformiert und neu trianguliert.



(a) (b)

Abbildung 6.21: Triangulierung ohne und mit Krümmungstransformation in 2D-Ansicht

Abbildung 6.21 stellt das Triangulierungsergebnis ohne und mit Krümmungstransformation für eine Beispielregion gegenüber. Durch die Streckung der Region in vertikaler Richtung hat der AFT-Algorithmus zusätzliche Knotenpunkte zwischen den Segmentgrenzen und dem inneren zwingenden Kantenzug untergebracht. In der 3D-Ansicht (Abb. 6.22) bildet das Dreiecksnetz eine differenzierte Oberfläche.

Trotz möglicher guter Ergebnisse im Einzelfall, besitzt das Transformationsverfahren konzeptionelle Mängel.
Zunächst wird durch die zusätzliche Transformation die optimierende Eigenschaft des AFT-Algorithmus für das Ergebnis der Triangulierung der Oberfläche vollends wertlos. Die

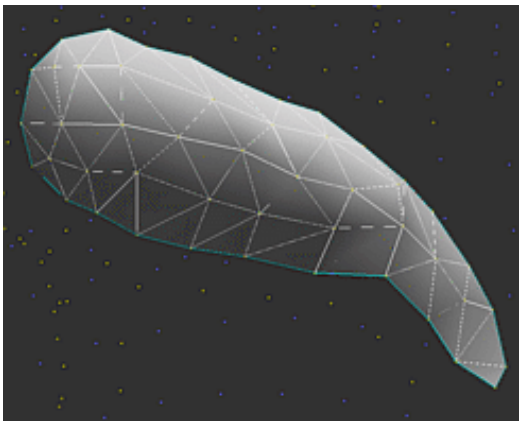


Abbildung 6.22: Triangulierung einer Region mit Krümmungstransformation in 3D-Ansicht

Krümmungstransformation stellt, neben dem Übergang von der 3D-Oberfläche (ϕ, h, r) zur (ϕ, h)-Ebene, einen zusätzlichen geometrieverzerrenden Schritt dar. Der AFT-Algorithmus ist bei der Erzeugung neuer Knotenpunkte durch eine Kette aus zwei winkel- und längenverzerrenden Transformationen von der eigentlichen 3D-Oberflächengeometrie getrennt. Mit diesem Konzept kann zwar die Dichte der Knotenpunkte an die lokale Krümmung angepaßt werden, aber es besteht keine Kontrolle, daß neue Knotenpunkte die Entstehung wohlgeformter Dreiecke begünstigen. Die Retriangulierung der erzeugten Knotenpunkte in der untransformierten (ϕ, h)-Ebene mit der CDT zielt darauf ab, eine Mindestmaß an Wohlgeformtheit auf der Oberfläche herzustellen.
Darüber hinaus wird bei typischen Laserscandaten die vertikale Richtung in stärkerem Verhältnis als die horizontale Richtung gestreckt. Die Transformation verteilt die in der zweidimensionalen Scandatei vorhandene Krümmung gleichmäßig über das 512^2-Pixelraster. Als quadratische Bilddatei betrachtet, erscheint durch die endliche Ausdehnung des Scanobjekts am oberen (z.B. oberhalb Schädelkalotte) und unteren (z.B. unter Halsansatz) Bildrand jeweils ein Streifen ohne Daten. Bei der Transformation werden die im mittleren Datenstreifen ermittelbaren Krümmungswerte über die gesamte Bildausdehnung gestreckt.
Zuletzt besteht bei der Transformation keine Absicherung gegen das Entstehen ungültiger Triangulierungen. Wie Abbildung 6.23 illustriert, können Knotenpunkte, die sich im Transformationsraum innerhalb einer Region befinden nach der Rücktransformation außerhalb der Region erscheinen. Die Regionengrenze wird mißachtet und das Zusammenfügen mit der Triangulierung einer Nachbarregion ist nicht mehr möglich.

6.2.4 Triangulierung durch Anlegen der Dreiecke an die Oberfläche

In diesem Modus konstruiert **COMPUTE·NEW·CANDIDATE** den Kandidatenpunkt R in der dreidimensionalen Geometrie der Oberfläche. Da keine geschlossene Beschreibung der Oberfläche vorliegt, mit deren Hilfe eine Bogenmetrik auf der Oberfläche definiert werden kann, wird die Oberflächenkoordinate des Kandidatenpunktes iterativ bestimmt. Dazu wird das Kandidatendreieck mit der gewünschten Seitenlänge mit dem neuen Knotenpunkt R so lange um die Kante gedreht, bis der Abstand von R zur Oberfläche minimal ist.

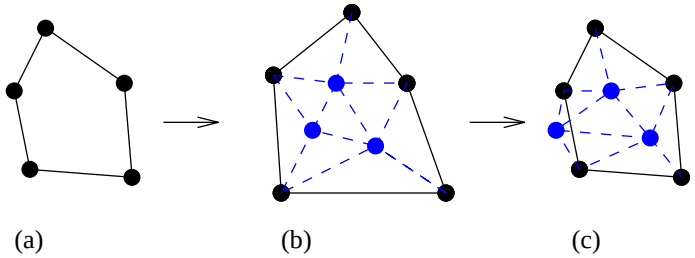
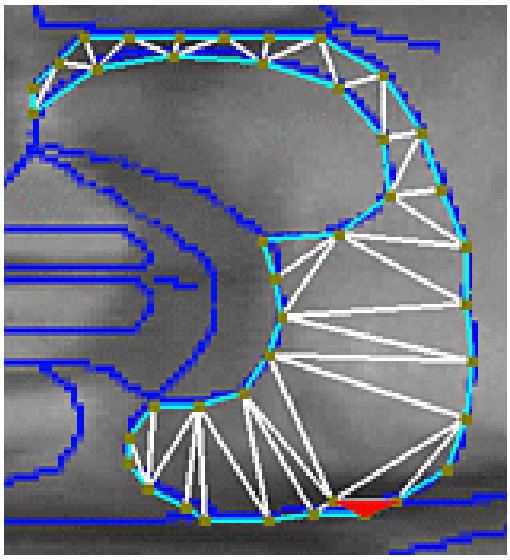
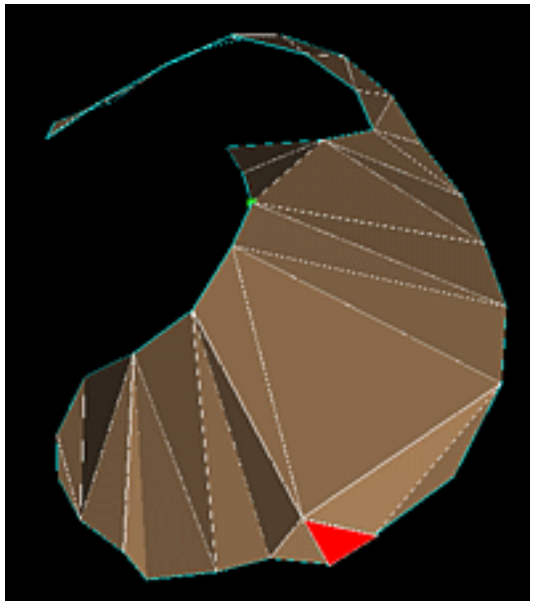


Abbildung 6.23: Ungültige Triangulierung durch Transformation
(a) Regionenbegrenzung, (b) AFT setzt neue Knotenpunkte im Transformationsraum, (c) nach der Rücktransformation befindet sich ein Knotenpunkt in einer Nachbarregion. In der Nachbarregion kommt es nach dem Zusammenfügen der Triangulierungen zur Überlagerung von zwei Dreiecken.

In Abbildung 6.24 (a) ist die 2D-Ansicht einer Region zu sehen, bei der der AFT-Algorithmus nach der Konstruktion eines Dreiecks (in der Abbildung rot markiert) abgebrochen wurde. Die übrige Triangulierung der Region ist die 2D-Grundtriangulierung. Das Dreieck wurde in einer Zone konstruiert, deren Flächennormale stark von der Scanrichtung (Projektionsrichtung der 2D-Darstellung) abweicht. Durch das Anlegen an die Oberfläche erscheint das Dreieck in der 2D-Ansicht deutlich verzerrt. In der 3D-Ansicht (Abb. 6.24 (b)) ist die Gleichschenkligkeit des roten Dreiecks erkennbar.



(a) Das rote Dreieck wurde durch Anlegen an die Oberfläche erzeugt und erscheint in der 2D-Ansicht durch die perspektivische Verzerrung degeneriert.



(b) In der 3D-Ansicht ist das Dreieck gleichschenklig.

Abbildung 6.24: Durch Anlegen an die Oberfläche konstruiertes Dreieck

In Abbildung 6.25 wurde die Kreisbahn des Kandidatenpunktes rot markiert und ver-

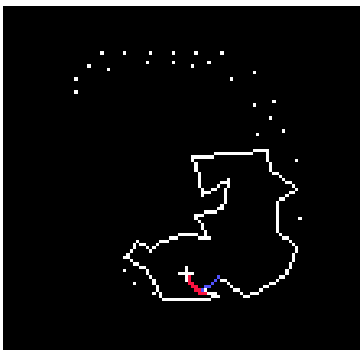


Abbildung 6.25: Bildschirmausgabe bei AFT-Algorithmus mit 3D-Drehkonstruktion
Die aktuelle Kante ist blau, die Projektion der Kreisbahn rot und die Endposition des Kandidatenpunktes bei Erreichen der Oberfläche mit einem Kreuz markiert. Der eingezeichnete Kantenzug ist die aktuelle Front. Die Initialknoten auf der Regionengrenze und die erzeugten Steiner-Punkte sind als weiße Pixel dargestellt.

deutlicht den Drehvorgang zur Konstruktion eines neuen Dreiecks. Die 2D-Projektion der Kreisbahn stellt eine elliptische Kurve dar, deren Form abhängig von der Raumlage der aktuellen Kante ist.

In den Probeläufen ergab sich, daß die Drehkonstruktion der Dreiecke zu unerwünschten Kandidatenpunkten führen kann. Die zu minimierende Fehlerfunktion kann mehr als eine Nullstelle besitzen (Abb. 6.26). Die implementierte Minimierung dreht das Dreieck aus der Ausgangslage in die linke Richtung zur aktuellen Frontkante und bricht bei der ersten genäherten Nullstelle der Fehlerfunktion ab. Falls *unit_size* größer als die geometrische Feinheit der Oberfläche in der Umgebung ist, kann über Kanten in Steigungsrichtung (Abb. 6.26) ein Kandidatenpunkt weit außerhalb der Region gefunden werden. Diese Lage wird von **VISIBLE** in den darauffolgenden Bearbeitungsschritten richtig erkannt, und der existierende Kandidat wird zur Bildung eines neuen Dreiecks herangezogen. Damit wird die Möglichkeit nicht wahrgenommen, ein ideales Dreieck zu konstruieren. Stattdessen wird ein existierender Kandidat S bevorzugt, der von **COMPUTE·EXISTING·CANDIDATE** mit Hilfe der Grundtriangulierung (2D-CDT) bestimmt wurde. Unter den kritischen geometrischen Bedingungen des Sonderfalls ist es wahrscheinlich, daß die 2D-CDT durch Projektionsfehler beim Übergang auf die Oberfläche zu einem spitzwinkligen Dreieck führt.

Der Sonderfall tritt nicht ein, wenn *unit_size* vom Benutzer für die betreffende Region ausreichend beschränkt wird. Andernfalls wäre eine aufwendige geometrische Analyse zur Ermittlung der richtigen Nullstelle notwendig.

Unter den beschriebenen geometrischen Verhältnissen kann sich ein weiterer Sonderfall einstellen, der in der ursprünglichen Formulierung des Algorithmus nicht nur zu schlechten, sondern auch ungültigen Triangulierungen der Oberfläche führt. Während bei der 2D-Version von **COMPUTE·NEW·CANDIDATE**(E; R) garantiert Kandidatenpunkte

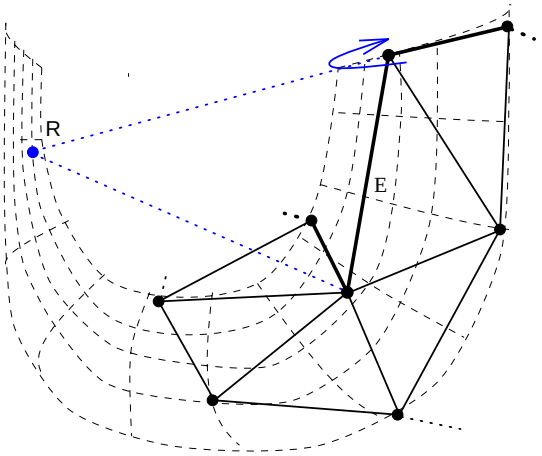


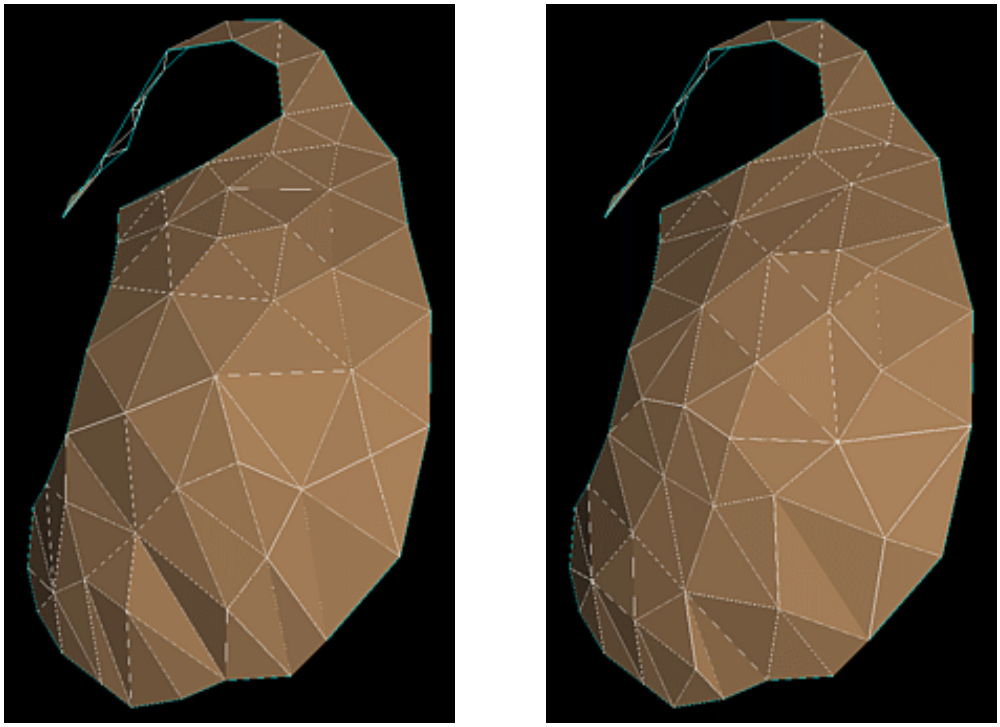
Abbildung 6.26: Die Fehlerfunktion der Drehkonstruktion kann mehrere Nullstellen besitzen
E ist die aktuelle Frontkante. Der Ausschnitt der Front ist als breiterer Linienzug eingezeichnet. Die Kantenlänge ist größer als die geometrischen Verhältnisse in der Umgebung. In diesem Fall ermittelt die Minimierung einen Kandidatenpunkt R außerhalb der Region, obwohl die Konstruktion eines idealen Dreiecks möglich wäre.

R konstruiert werden, die sich links von der aktuellen Kante befinden, kann mit dem 3D-Drehverfahren ein Kandidat R ermittelt werden, der sich rechts von der aktuellen Kante befindet. Der Kandidat passiert unter Umständen den Sichtbarkeitstest, da **VISIBLE** nur auf Schnittpunkte der Verbindungslinien zwischen allen Frontkanten und dem Kandidaten testet. Wenn R zusätzlich von **NEW·IS·PREFERRED** ausgewählt wird (Entscheidungskriterien in Abb. 4.12), entsteht das neue Dreieck über der falschen Seite der Front.

Der Fehler wird beseitigt, indem die Abbruchbedingung zur Drehung des Dreiecks um eine Abfrage ergänzt wird, in der geprüft wird, ob sich der Kandidat links von der aktuellen Kante befindet.

Abbildung 6.27 stellt die Ergebnisse der 3D-Drehkonstruktion der einfachen 2D-Triangulierung für die gekrümmte Region 12 der Testfigur gegenüber. Die 3D-Konstruktion bringt in den Zonen, deren Flächennormale von der Scanrichtung erkennbar abweicht, eine sichtbare Verbesserung der durchschnittlichen Elementqualität mit sich. Die gleichschenkligen Dreiecke sind direkt aus der Drehkonstruktion eines neuen Kandidatenpunktes entstanden. Die ungleichmäßigeren Dreiecke wurden mit einem existierenden Knotenpunkt S erzeugt oder kommen in der 2D-CDT der erzeugten Knnotenpunkte nach Terminierung des AFT-Algorithmus zustande.

In Bereichen mit besonders großem Winkel zwischen der Oberflächennormalen und der Scanrichtung werden ebenfalls mehr Knotenpunkte gesetzt als bei der einfachen 2D-Triangulierung. Abbildung 6.28 enthält die Ansicht der rechten Nasenflanke der Testfigur. Mehrere Dreiecke sind ideal geformt. Insgesamt ist das Ergebnis der Triangulierung jedoch noch unbefriedigend. Die Grundtriangulierung zur Bestimmung von S erfolgt in 2D, die Entscheidung zwischen R und S wird in 2D getroffen und die Retriangulierung erfolgt



(a) (b)

Abbildung 6.27: Vergleich 2D-Triangulierung und Triangulierung mit Drehkonstruktion
(a) zeigt das Ergebnis der 2D-Triangulierung in der (ϕ, h)-Ebene. (b) zeigt die gleiche Region nach der Triangulierung mit dem Drehverfahren, das die Dreiecke auf der gekrümmten Oberfläche konstruiert. Beide Netze enthalten 26 Steiner-Punkte.

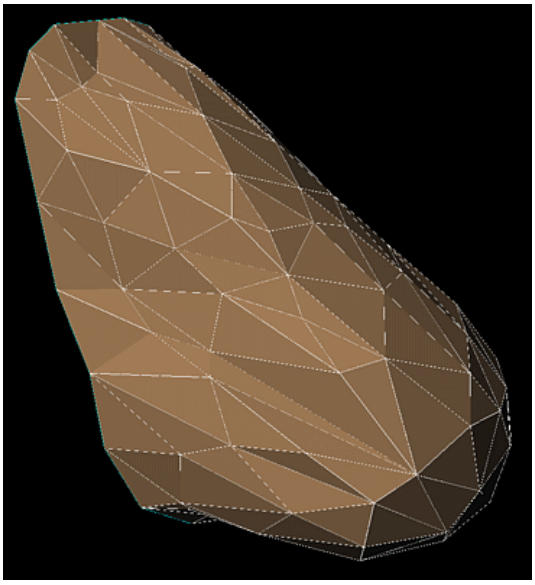


Abbildung 6.28: Drehkonstruktion an der Nasenflanke
An der Nasenflanke ist die Scanrichtung fast tangential. Mit der Drehkonstruktion können trotzdem mehrere ideale Dreiecke erzeugt werden.

in 2D. Diese Maßnahmen fördern gute Triangulierungen in der (ϕ, h)-Ebene. Allein die Bestimmung von R unter Einbeziehung der Rauminformation reicht nicht aus, um gute Oberflächentriangulierungen zu erzeugen.

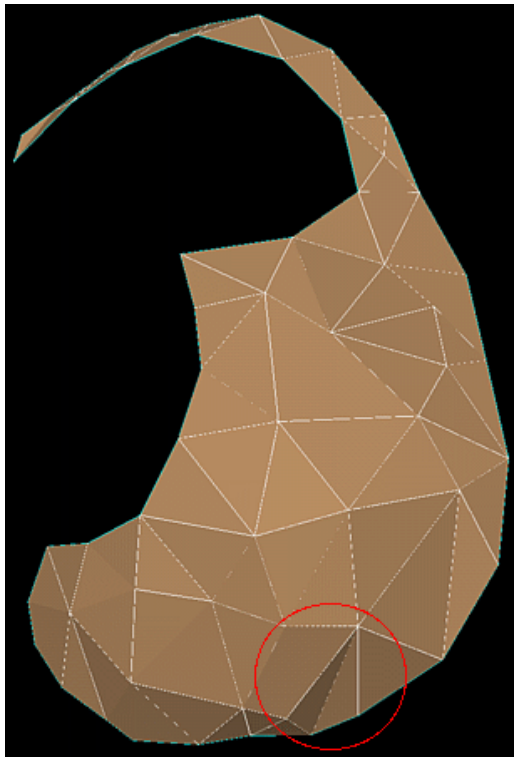
6.2.5 Krümmungsabhängige *unit_size*

Wie die in Abschnitt 6.2.3 getestete Krümmungstransformation ist die krümmungsabhängige *unit_size* eine Maßnahme zur Steuerung der Dreiecksgröße. Die adaptive Triangulierung *(adaptive meshing)* verringert den Approximationsfehler der Oberfläche durch das Dreiecksnetz. Die krümmungsabhängige *unit_size* kann im Gegensatz zur Krümmungstransformation mit der Drehkonstruktion des Kandidatenpunktes R gekoppelt werden.

Abbildung 6.29 erlaubt für Region 12 den direkten Vergleich zwischen zwei Versionen der Triangulierung durch Drehkonstruktion. Bei der zweiten Triangulierung wurde nach jedem ermittelten Kandidatenpunkt R ein Wert für die Abweichung des Dreiecks von der Objektoberfläche bestimmt. Falls die Abweichung über einem Grenzwert lag, wurde *unit_size* um ein Drittel verringert und ein neuer Kandidatenpunkt R berechnet. Die Region 12 besitzt auf der Begrenzung 56 Initialknotenpunkte. Ohne die adaptive Modifikation werden 16 Steiner-Punkte erzeugt. Die adaptive Version hat 3 Steiner-Punkte mehr in der Region untergebracht. Die Abbildung läßt erkennen, daß die Oberfläche mit den zusätzlichen Steiner-Punkten etwas differenzierter ausgebildet wird. In den flacheren Bereichen der Region bleiben die typischen Kantenlängen gleich.

In Abbildung 6.29 (b) markiert der rote Kreis zwei benachbarte Dreiecke, die durch Diagonalentausch zu einer besseren Oberflächenapproximation und kleineren Dreieckswin-

(a) Triangulierung ohne krümmungsabhängige Korrektur der Kantenlänge.



(b) Triangulierung mit krümmungsabhängiger Korrektur der Kantenlänge.

Abbildung 6.29: Triangulierungsergebnis bei krümmungskorrigierter Kantenlänge

keln führen. Die 2D-Retriangulierung in der (ϕ, h)-Ebene führt in diesem Fall zu keiner guten Oberflächentriangulierung, da die Flächennormalen der beiden Dreiecke zu stark von der Scanrichtung abweichen. Abbildung 6.30 zeigt, daß die Viereckdiagonale (gemeinsame Kante der beiden Dreiecke) in der (ϕ, h)-Ebene den kleinsten Winkel minimiert. Nach der Rückprojektion auf die Oberfläche haben sich die Winkelverhältnisse geändert. Im Raum maximiert die Diagonale den kleinsten Winkel.

6.2.6 Laufzeitverhalten

Die Implementierung ist auf einer leistungsfähigen Workstation durchgeführt worden. Die Laufzeiten zur Triangulierung einer Region stellen bei der Datenmenge der Testfigur kein Hindernis für interaktives Arbeiten dar, obwohl die Testimplementierung noch nicht geschwindigkeitsoptimiert wurde.

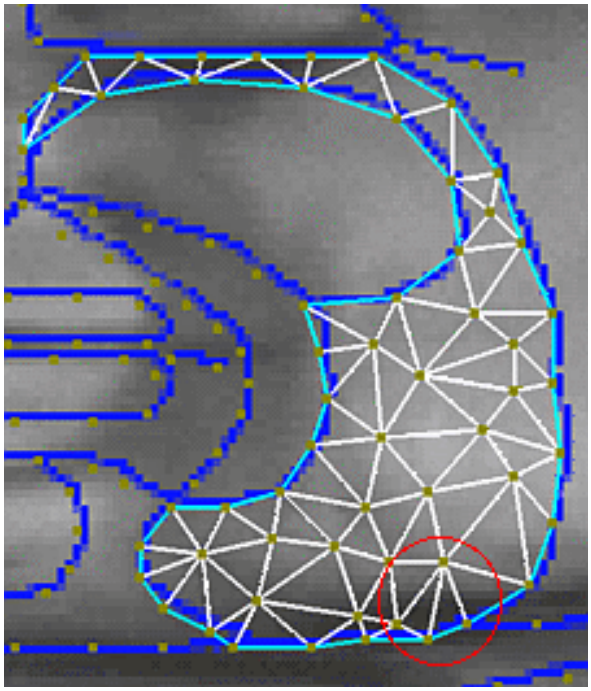


Abbildung 6.30: 2D-Darstellung der adaptiven Triangulierung
Die 2D-CDT der Region maximiert den kleinsten Winkel in der (ϕ, h)-Ebene. Die ungünstige Diagonale in 3D ist in 2D optimal. Das betreffende Viereck ist rot eingekreist.

Kapitel 7

Diskussion und Ausblick

Im abschließenden Kapitel werden aus den Testergebnissen des vorhergehenden Kapitels eine Reihe von Modifikationen für das Konzept zur rechnerunterstützten Generierung von Animationsnetzen abgeleitet. Danach folgt eine kurze Zusammenfassung der vorliegenden Arbeit. Im Ausblick werden längerfristige Entwicklungsziele der animationsorientierten Netzgenerierung genannt.

7.1 Modifikationen des Konzepts

Die Testimplementation kann Netze erzeugen, die die Anforderungen an ein Animationsnetz annähernd erfüllen. Durch manuelle Nachbearbeitung mit einem Editierwerkzeug könnte die Netzqualität in wenigen Schritten deutlich verbessert werden. Das Gesamtkonzept aus benutzerdefinierten Animationslinien, davon induzierter Partitionierung der Domain in Regionen, Initialisierung der Segmente und zwingender innerer Kantenzüge und Triangulierung der Regionen erweist sich als tragfähig zur Erzeugung von Animationsnetzen.

Zur Verbesserung der algorithmisch erzeugten Netze bieten sich eine Reihe von Modifikationen des Softwarekonzeptes an. Diese betreffen die Initialisierung der Segmente, den Triangulierungsalgorithmus, softwaretechnische Gesichtspunkte und Verbesserungen der Benutzerinteraktion.

Algorithmische Schnittpunktanalyse der Animationslinien mit anschließender Partitionierung der Domain in Regionen

Die automatische Bestimmung der Schnittpunkte der Animationslinien für deren Zerlegung in Segmente kann in der diskreten (ϕ, h)-Pixelebene problematisch sein. Linien, die im Kontinuierlichen einen gemeinsamen Punkt besitzen, können in der diskreten Ebene kein, ein oder mehrere gemeinsame Pixel besitzen. Darüber hinaus kann ein gemeinsames Pixel nur iterativ bestimmt werden. Eine geschlossene Repräsentation der Animationslinien, beispielsweise durch Splines, ermöglicht die exakte und effizientere Schnittpunktbestimmung der Raumkurven auf der Objektoberfläche.

Die Regionenanalyse beschäftigt sich mit der Topologie der Animationslinien. Das System aus Schnittpunkten, die durch Segmente verbunden sind, läßt sich als Polygonnetz

oder Graph interpretieren, sofern man die Segmente als Kanten betrachtet. Die algorithmische Regionenanalyse kann durch einfache Datenstrukturen und Operationen realisiert werden. Der entwickelte Entwurf, der die Erkennung von Regionen in Regionen (ergibt eine topologisch mehrfach verbundene Domain) und innerer Constraints einschließt, basiert auf einfachen Tabellen und Suchoperationen. Seine Darstellung wäre so umfangreich, daß darauf an dieser Stelle verzichtet wird.

Initialisierung der Segmente mit Knotenpunkten

Der Abschnitt 6.1 dokumentiert die Schwierigkeiten bei der Verteilung der Initialknotenpunkte auf diskreten Segmenten. Das Raumkoordinatensystem (ϕ, h, r) löst die (ϕ, h)-Koordinaten nur in einem groben diskreten Raster auf, während r einen Fließkommawert darstellt. Es ist problematisch, in diesem diskreten Koordinatensystem eine einfache und gute Approximation an die Krümmung der Kurve zu berechnen. Die erzielten Ergebnisse der krümmungsbasierten Initialisierung ließen sich durch fundierte Korrekturfunktionen verbessern. Trotzdem würden die Krümmungsverhältnisse bei kurzen Segmenten, die in der diskreten (ϕ, h)-Ebene nur aus wenigen Pixeln bestehen, weiterhin nur mit großen Ungenauigkeiten analysierbar sein.

Die geschlossene, kontinuierliche Repräsentation der Raumkurven durch Splines erlaubt robustere und exaktere Verfahren zur krümmungsbasierten Initialisierung. Aus der geschlossenen Darstellung läßt sich im allgemeinen zu jeder Stelle der Kurve exakt das Bogenmaß, der Krümmungswert und dessen Integral nicht-iterativ bestimmen. Die Darstellung ist skalierungsinvariant. Kurvendetails, die im Pixelraster nicht mehr aufgelöst werden können, bleiben erhalten.

Einfluß der Initialisierung der Segmente auf die Netzqualität

Bereits aus der Tatsache, daß das Gesichtsnetz in Abbildung 6.16 zu mehr als zwei Dritteln Initialknotenpunkte beinhaltet, läßt sich die Bedeutung der Initialknotenpunkte für das Ergebnis der Triangulierung erkennen.

Bei der Initialisierung der Segmente gelten die Randbedingungen:

- Die Verteilung der Knotenpunkte auf den Segmenten muß den Anfangs- und Endpunkt berücksichtigen. Die Länge der Intervalle zwischen den Initialknotenpunkten soll nicht zu stark von einem mittleren Wert abweichen, der identisch zur angestrebten mittleren Kantenlänge in den benachbarten Regionen ist.
- Die Initialisierung bildet einen Kantenzug, der die Raumkurve approximiert.

Die Beobachtungen in Abschnitt 6.2.1 führen zur Formulierung einer weiteren Randbedingung:

- Die Initialisierung benachbarter Segmente muß in gegenseitiger Abstimmung unter der Zielsetzung erfolgen, daß die gebildeten Regionen gut trianguliert werden können.

FARESTAMs Methode zur Initialisierung bei eng benachbarten zwingenden Linien ist ungeeignet, alle Randbedingungen zu erfüllen [Far93]. Abhängig vom Abstand zwischen zwei benachbarten Begrenzungslinien und den Krümmungsverhältnissen der Linien reduziert FARESTAM alle Kantenlängen in der Umgebung. Die Dreiecksgröße schrumpft, und

die Anzahl der Dreiecke in der Umgebung steigt. Die kleineren Dreiecke sind wohlgeformt. Die ersten beiden Randbedingungen zur Initialisierung der Segmente für Animationsnetze beschränken allerdings den Spielraum zur Auflösungserhöhung. Die Netzqualität in den eng begrenzten Bereichen soll nicht durch eine Vervielfachung der Initialknotenpunkte und deutliche Reduktion der Dreieckskantenlänge erreicht werden. Die Netzqualität soll in erster Linie durch Umverteilen der bestehenden Initialknotenpunkte und Setzen weniger zusätzlicher Knotenpunkte verbessert werden. Damit wird die algorithmische Initialisierung der Segmente zu einem komplizierten geometrischen und kombinatorischen Problem. Die Abhängigkeiten bei der Umverteilung der Knotenpunkte auf den Segmenten können mehrere benachbarte Regionentriangulierungen einschließen und sogar zyklisch sein.

Bereits für den trivialen Fall paralleler Segmente kann die algorithmische Behandlung der Segmentinteraktion nur mit mehreren Fallunterscheidungen realisiert werden. Der Entwurf bearbeitet Segmente, deren Abstand maximal zwei Kantenlängen beträgt. Nach der Abstandsmessung wird entschieden, ob zwischen den Segmenten ein oder zwei Streifen aus gleichschenkligen Dreiecken untergebracht werden sollen. Die Länge der Intervalle auf den Segmenten und *unit_size* in der Region werden dementsprechend gewählt. Falls ein Streifen vorgesehen ist, werden die Initialknotenpunkte auf den Segmenten versetzt positioniert, andernfalls gegenüberliegend.
Das triviale Verfahren kann keine komplizierteren geometrischen Konstellationen benachbarter Segmente berücksichtigen. Außerdem müßte es mit einer Kontrollstruktur versehen werden, die die Abarbeitungsreihenfolge für mehrere Segmente und ihre eingeschlossenen Regionen festlegt. Die Abarbeitung darf keine Endloszyklen bilden.

Der erste Arbeitsschritt eines universelleren Verfahrens zur Abstimmung der Position der Initialknotenpunkte auf benachbarten Segmenten muß die geometrische und topologische Analyse der Segmente und Regionen beinhalten. Hierzu kann die CDT der Regionen hilfreich sein. Im AFT-Algorithmus dient die CDT des ungemeshten Bereichs als effiziente Datenstruktur für die geometrische Dekomposition der Domain durch neue Dreiecke. Auch für die Analyse ermöglicht die CDT effiziente Operationen. Beispielsweise kann der Abstand zwischen zwei Segmenten sehr effizient über die Länge der Verbindungskanten genähert werden. Alle Segmentbereiche einer Region, deren Initialisierung aufeinander abgestimmt werden muß, sind durch ihre kurzen Verbindungskanten einfach identifizierbar. Diese Information wird abgespeichert und die Regionentriangulierung mit den unkorrigierten Initialpunkten gestartet. Die Regionentriangulierungen werden zum Animationsnetz zusammengesetzt. In einem Nachbearbeitungsschritt werden die markierten kritischen Initialknoten durch wiederholte Anwendung einer Glättungsoperation bearbeitet. Hierzu kann beispielsweise eine Abwandlung der Laplace-Glättung (Abschnitt 4.4.1) eingesetzt werden, die die Koordinaten des Knotenpunktes nur auf der Segmentlinie verschiebt. Die optimale Bearbeitungsreihenfolge und Iterationszahl muß experimentell ermittelt werden.

Das beschriebene Verfahren minimiert die Energie in einem Netzwerk aus Federelementen. Die Federkraft ist proportional zur Kantenlänge. Die Anfangs- und Endknotenpunkte der Segmente sind fest. Die übrigen Segmentknotenpunkte sind wie auf einer Schiene nur über der Segmentlinie beweglich. Die wiederholte Anwendung des Glättungsoperators minimiert die Summe der Federkräfte im System. Die Einbeziehung der Steiner-Punkte in den Minimierungsprozeß könnte die Gesamtqualität des Netzes weiter verbessern. Die mittlere Koordinate der umgebenden Punkte muß für einen Steiner-Punkt geeignet auf die

Oberfläche abgebildet werden. Für Segmentknotenpunkte muß eine sinnvolle Abbildung auf die zugrundeliegende Segmentkurve gefunden werden.

Speziell zur Behandlung paralleler Segmente mit engem Abstand bietet sich die Kombination des statistischen Verfahrens mit einer einfachen Methode zur Einführung zusätzlicher Segmentknotenpunkte an. Stellt sich während des AFT-Algorithmus heraus, daß über einer Segmentkante nur existierende Kandidaten des gegenüberliegenden Segments bevorzugt werden und zu Dreiecken mit niedriger Höhe führen (Situation in Abb. 6.18), dann wird die Kante gespalten. Diese Strategie steht im Widerspruch zur Forderung, daß begrenzende Kanten nicht gespalten werden dürfen. Es wird daher unbedingt erforderlich sein, die zur anderen Seite des Segments benachbarte Region zu retriangulieren, damit die Gesamttriangulierung gültig bleibt. Durch die Kantenspaltung können mit einer einfachen Operation notwendige zusätzliche Initialknotenpunkte gesetzt werden. Die Länge des Segmentintervalls springt am eingesetzten Punkt auf das halbe Niveau der mittleren Intervallänge. Im Nachbearbeitungsschritt wird die Ungleichmäßigkeit geglättet.

Die algorithmischen Verfahren werden zur Verbesserung der Netzqualität beitragen. Trotzdem wird es Situationen geben, in denen der Benutzer mit seiner hochentwickelten Fähigkeit zu konfigurativem Sehen bessere Lösungsvorschläge geben kann, als es die algorithmische Methode vermag. Die nächste Implementierung sollte daher über effiziente Werkzeuge zur Benutzerinteraktion verfügen und vollständig interaktiv konzipiert sein. Der Benutzer entscheidet im Zweifelsfall selbst, auf Grund der optischen Qualität der Triangulierung, welche Initialknotenpunkte er verschiebt, löscht oder hinzufügt. Danach wird die Triangulierung aktualisiert.

Abschließend sei noch angemerkt, daß alternative Triangulierungsverfahren zur AFT-Methode existieren, die weniger abhängig von der Initialisierung der Regionenbegrenzung sind. Die AFT-Methode tendiert in schmalen Bereichen einer Region zu ungünstigen Netzelementen, in denen der Abstand zwischen der Begrenzung kein ganzzahliges Vielfaches der Standarddreieckshöhe darstellt. Sie dekomponiert die Region elementweise. Falls das Dreieck aus der aktuellen Frontkante und zwei idealen Kanten nicht realisierbar ist oder in den nachfolgenden Schritten zu extrem entarteten Dreieckselementen führt, wird ein Dreieck mit einem existierenden Kandidatenpunkt erzeugt. Dieser Knotenpunkt stammt aus der CDT der ungemeshten Region und bezieht die Geometrie der Region in die Triangulierung ein. Eine Methode, bei der jedes neue Dreieckselement durch die geometrische Dekomposition der Region zustande kommt, bevorzugt keine "Ganzzahligkeit" der Regionengeometrie. CHEWs Methode zur Triangulierung von gekrümmten Oberflächen dekomponiert die Region global.

Gute Approximation der Oberfläche

Dieser Anforderungspunkt wird mit der entwickelten Konzeption ausreichend erfüllt. Lediglich in den Bereichen, die einen großen Winkel zur Projektionsebene (Zylindermantel) bilden, geht teilweise sichtbare Oberflächeninformation verloren. Die unzulänglich approximierte Oberfläche ist eine Folge degenerierter Dreiecke in diesen Bereichen.

Selektive Punktdichte

Die Animationslinien erweisen sich als praktikables Ausdrucksmittel für zwingende Kantenzüge und zur Definition von Regionen mit einheitlicher Punktdichte.

Qualität der Dreieckselemente

Die AFT-Methode erzeugt ideale Dreiecksnetze, in denen sich eine Flußrichtung der Elemente ausbildet, sofern die Region genügend groß in Relation zur Kantenlänge ist, die Begrenzung möglichst glatt ist und die Oberflächennormale wenig von der Scanrichtung des 3D-Laserscanners abweicht.

Schmale Bereiche zwischen zwei annähernd parallelen Abschnitten von begrenzenden Segmenten werden teilweise in ungünstige Dreiecke aufgeteilt. Die Ursache für dieses Verhalten liegt in der elementweisen Dekomposition. Wie bereits im Abschnitt "Einfluß der Initialisierung der Segmente auf die Netzqualität" ausgeführt, kann ein Triangulierungsverfahren, das die Region global geometrisch dekomponiert, unter engen Bedingungen gleichmäßigere Aufteilungen der Domain in Dreiecke finden.

Das (ϕ, h, r) Koordinatensystem der Scandaten hat die Verwendung der zweidimensionalen AFT-Methode in der (ϕ, h)-Ebene als Basismethode zur Oberflächentriangulierung nahegelegt. Bei der Implementation konnte auf frei erhältliche Software zur Berechnung der CDT zurückgegriffen werden. Die anschließend entwickelten dreidimensionalen Erweiterungen (Krümmungstransformation, Drehkonstruktion eines neuen Dreiecks im Raum) sollten gute Oberflächentriangulierungen ermöglichen.

Die Drehkonstruktion allein kann allerdings noch keine idealen Oberflächentriangulierungen der Scandaten sicherstellen. Drei weitere Komponenten arbeiten nur in der (ϕ, h)-Ebene und verhindern die Entstehung vieler wohlgeformter Oberflächendreiecke.

1. Die Entscheidungsfunktion zwischen dem neuen und dem existierenden Kandidatenpunkt arbeitet in der (ϕ, h)-Ebene.

2. Die Grundtriangulierung, die für die Wahl des existierenden Kandidatenpunktes verantwortlich ist, findet in zwei Dimensionen statt.

3. Die abschließende Triangulierung der Segmentknotenpunkte und Steiner-Punkte findet in der (ϕ, h)-Ebene statt.

Die konsequente Weiterentwicklung der AFT-basierten Konzeption zur animationsorientierten Triangulierung würde den Entwurf von 3D-Versionen dieser Komponenten beinhalten.

Mit der Entwicklung einer Oberflächen-CDT stünde aber bereits der Kernbestandteil von CHEWs Algorithmus zur Triangulierung von gekrümmten Oberflächen (S. 59) zur Verfügung. Wenige zusätzliche Kontrollstrukturen würden genügen, um CHEWs Algorithmus zu implementieren. Der schlankere Entwurf von CHEWs Algorithmus in Verbindung mit den im folgenden nocheinmal zusammengefaßten Vorteilen läßt diese Methode als geeigneteres Triangulierungsverfahren erscheinen. Die Vorteile von CHEWs Verfahren sind:

1. CHEWs Algorithmus unternimmt keine elementweise, sondern globale geometrische Dekomposition der Region.

2. Der Algorithmus garantiert wohlgeformte Dreiecke mit Winkelgrößen nicht unter 30° für alle neu eingeführten Kanten.

3. Er kann praktisch auf allen topologisch mehrfach verbundenen, mehrfach gefalteten Domains arbeiten und berücksichtigt Constraints.

4. Die Qualität der Triangulierung hängt nicht von heuristischen Strategien, wie der Reihenfolge der bearbeiteten Kanten ab.

Die Eigenschaft, Feldstrukturen auszubilden hat zu Beginn der Arbeit unter anderem zur Entscheidung für die AFT-Methode geführt. In einigen Regionen der Testfigur kann die implementierte AFT diese Strukturen ausbilden. Für den Fall, daß CHEWs Algorithmus in den großen flachen Regionen schlechtere Triangulierungen ohne Feldstruktur erzeugt, wird dem Benutzer in der nächsten Implementierung die freie Wahl der Triangulierungsmethode ermöglicht.

Koordinatensysteme

Das diskrete (ϕ, h)-Raster im zylindrischen (ϕ, h, r)-Scankoordinatensystem dient als Koordinatensystem zur Orientierung auf der Objektoberfläche. In der Testimplementation wird das (ϕ, h)-Koordinatensystem an den Schnittstellen zwischen den einzelnen Hilfsprogrammen, bei der Berechnung der CDT und bei verschiedenen geometrischen Hilfsroutinen (z.B. Überprüfung, ob sich ein Knotenpunkt links von einer Kante befindet) verwendet.

Bei Netzen mit kleinen Kantenlängen und in Regionen, deren Oberflächennormale stark von der Scanrichtung abweicht, bietet das Raster zu geringe Auflösung. In der nächsten Implementation soll daher auf der interpolierten, kontinuierlichen Oberfläche gearbeitet werden, die aus Scandaten verschiedener Auflösungsstufen gebildet werden kann. Die Repräsentation der Punkte wird durchgängig mit Hilfe kartesischer Fließkommakoordinaten erfolgen. Damit wird die Triangulierung skalierungsinvariant und beliebig kleine Dreiecke sind möglich. In der derzeitigen Implementation führen zu kleine Kantenlängen zu Laufzeitfehlern, wenn neu berechnete Knotenpunkte im groben (ϕ, h)-Raster mit existierenden Knotenpunkten zusammenfallen.

Benutzerfreundliche graphische Oberfläche

In der gegenwärtigen Testimplementation zerfällt die Software in mehrere Programme und ist nicht benutzerfreundlich gestaltet. Eine praxistaugliche Implementation des Triangulierungskonzepts integriert die einzelnen Programme zu einer Anwendung und stellt dem Benutzer effiziente Werkzeuge für die interaktive Netzerstellung unter einer graphischen Oberfläche zur Verfügung. Die Laufzeiten der nicht geschwindigkeitsoptimierten Testimplementierung machen eine interaktive Software realisierbar.

Die graphische Oberfläche umfaßt Werkzeuge zur Definition der Animationslinien und zur interaktiven Bearbeitung der Initialknotenpunkte in Bereichen, in denen die automatische Segmentinitialisierung keine zufriedenstellenden Triangulierungen ermöglicht hat. Zur benutzerfreundlichen und exakten Definition der Animationslinien auf der Objektoberfläche ist es, zusätzlich zu den in den vorherigen Abschnitten genannten Gründen, naheliegend, eine Splinerepräsentation zu wählen. Der Benutzer markiert die Stützpunkte der Linie mit dem Mauszeiger auf dem gewünschten Punkt der Oberfläche. Dabei ist es denkbar, eine 3D-Editiermöglichkeit zu schaffen, bei der der Oberflächenpunkt in der 3D-Ansicht gesetzt wird. Die 3D-Ansicht ist die Bildschirmdarstellung der Objektoberfläche aus beliebiger Kameraposition (siehe Abb. 2.8). Jede neu definierte Animationslinie stößt einen neuen Durchlauf der automatischen Regionenanalyse an. Die Triangulierung einer Region wird auf Initiative des Benutzers mit benutzerdefinierter *unit_size* unternommen. Jede neue Regionentriangulierung wird sofort in die Bildschirmdarstellung des Gesamtnetzes übernommen. In der Bildschirmdarstellung kann die Position einzelner Knotenpunkte mit dem Mauszeiger auf der Oberfläche verschoben werden.

Im Softwareprojekt zur interaktiven rechnerunterstützten Erstellung von Animationsnetzen besitzt die Konzeption der Benutzerschnittstelle eine tragende Rolle für die Entwicklung einer effizienten, praxistauglichen Applikation.

Softwaretechnische Gesichtspunkte

Die Modifikationen des Softwarekonzepts betreffen grundsätzliche Abläufe (andere Triangulierungsmethode) und Datenstrukturen (weitestgehender Verzicht auf diskrete (ϕ, h)-Koordinaten). Daher sind die existierenden Komponenten der Testimplementierung nicht weiterverwendbar. Die völlige Neuimplementierung kann als softwaretechnisch ausgerichtetes Projekt durchgeführt werden.

Zur effizienten Implementierung bietet sich die Einbindung verschiedener Software-Bibliotheken an. Die Bibliotheken sind geschwindigkeitsoptimiert und wurden intensiv getestet. Verfügbar sind Bibliotheken für typische Datenstrukturen [Näh], zur Grafikausgabe [JNW94], zur Bildschirmdarstellung dreidimensionaler Objekte [Wer94] und zur Erstellung grafischer Benutzeroberflächen. Es bleibt noch zu recherchieren, ob die Splinedarstellung und -verarbeitung der Animationslinien ebenfalls mit Hilfe einer Bibliothek realisierbar ist.

7.2 Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit der Erstellung von Dreiecksnetzen für die Computeranimation. Behandelt werden Objekte, die durch einen zylindrisch angeordneten 3D-Laserscandatensatz repräsentiert werden. Ziel der Arbeit ist die Entwicklung eines neuen Konzepts, mit dem bei der Überführung der Scandaten in eine datenreduzierte Dreiecksnetzrepräsentation Anforderungen der Animation optimiert werden.

Im problembeschreibenden Teil der Arbeit wird die Dreiecksnetzerstellung im Kontext der übrigen Arbeitsschritte zur Herstellung und Animation eines virtuellen Darstellers dargestellt. Daraus folgt die Formulierung eines Anforderungskatalogs für Animationsnetze. Der Anforderungskatalog stellt die Anforderungsdefinition eines Softwarekonzepts für

die rechnerunterstützte Erstellung von Animationsnetzen dar.

Im theoretischen Teil der Arbeit werden die Grundlagen der Triangulierung von Punktmengen erarbeitet. Mit dieser Grundlage werden Netzgenerierungsmethoden aus der Literatur der Computergrafik und Finite Elemente Methoden gemeinsam erschlossen. In der strukturierten Darstellung erfahren Methoden, die günstige Eigenschaften für eine animationsorientierte Triangulierung besitzen, eine detailliertere Darstellung.

Aus der Anforderungsdefinition und den theoretischen Grundlagen wird im Entwurfsteil der Arbeit ein Konzept zur rechnerunterstützten Erstellung von Animationsnetzen entworfen. Der Entwurf wird zur Testimplementation gebracht. Die Anforderungen der Animation werden vom Benutzer in Form von Animationslinien auf der Objektoberfläche formuliert. Die Software nutzt die Animationslinien zur automatischen Unterteilung der Oberfläche des Objekts in Regionen. Die Regionenbegrenzung wird durch initiale Knotenpunkte diskretisiert. Zur Verteilung der Knotenpunkte auf den Animationslinien wird eine äquidistante und eine krümmungsbasierte Initialisierungsmethode entwickelt.
Zur Regionentriangulierung wird die Advancing-Front-Methode verwendet. Sie wird um die Möglichkeit erweitert, regioneninterne zwingende Kantenzüge in der Triangulierung zu respektieren. Die Regionentriangulierungen werden zum Schluß zu einem Gesamtnetz zusammengefügt. Für eine gleichmäßige Triangulierung der gekrümmten Oberfläche wird die, in der Projektionsebene des Scanvorgangs arbeitende Advancing-Front-Triangulierung um zwei selbstentwickelte Methoden ergänzt. Hierbei handelt es sich um die Triangulierung im krümmungsbasierten Transformationsraum und die iterative Konstruktion neuer Dreiecke durch Anlegen an die gekrümmte Oberfläche.

Anhand der Testimplementation werden zahlreiche Versuche mit den unterschiedlichen Konfigurationen der Software unternommen. Die Testdaten stammen aus der Praxis. Die Triangulierungsergebnisse werden den manuell erstellten Netzen gegenübergestellt.

Aus der Untersuchung der Testergebnisse können zahlreiche Erkenntnisse zur Verbesserung einzelner Komponenten des Softwarekonzepts gewonnen werden. Basierend auf den Erkenntnissen der Arbeit kann nun die Implementierung einer einsatzfähigen Software zur effizienten, interaktiven Erstellung von Animationsnetzen auf der Basis von 3D-Laserscandaten unternommen werden.

7.3 Ausblick

Das unmittelbare Ziel der zukünftigen Arbeit stellt die Implementation des Softwarekonzepts für den praktischen Einsatz dar. Mit der Software wird eine effiziente Methode der Animationsnetzerstellung bereitstehen.

Das Animationsnetz bildet die Basis für den nächsten Arbeitsschritt bei der Erstellung eines virtuellen Darstellers. In der Modellierung werden die Bewegungen der Figur durch Transformation der Knotenpunkte des Animationsnetzes definiert. Die Arbeitsschritte der Skulpturierung und der Modellierung sind in ihrer Reihenfolge zwar strikt getrennt, aber bedingen sich doch. Die Brücke zwischen den in der Modellierung angestrebten Bewegungen und der Skulpturierung unter Berücksichtigung dieser Bewegungen schlägt der

Benutzer. Für eine weitere Automatisierung der Animationsnetzerstellung müssen die Arbeitsschritte von Skulpturierung und Modellierung gekoppelt werden. Hierzu müßte über einem allgemeinen Oberflächenmodell eine abstrakte Formulierungsmöglichkeit für die Bewegungen gefunden werden. Aus der Auswertung aller formulierten Bewegungen wird daraufhin automatisch ein animationsorientiertes, datenreduzierendes Dreiecksnetz erstellt.

Das abstrakte Bewegungsmodell kehrt die Reihenfolge der Modellierung und der Erstellung des Animationsnetzes um. Damit wird die Möglichkeit geschaffen, aus dem abstrakten Bewegungsmodell Animationsnetze variabler Auflösung zu erzeugen. Die Datenmenge von Szenen mit mehreren virtuellen Darstellern kann beträchtlich reduziert werden, wenn die Auflösung der einzelnen Animationsnetze in Abhängigkeit von der Kameraentfernung gewählt wird. Die Arbeiten zur Multiresolution Analysis von Polygonnetzen ([EDD+95] und [GGS95]) motivieren dieses Entwicklungsziel.

Abbildungsverzeichnis

Literaturverzeichnis

[AM91] S. Abramowski und M. Müller. *Geometrisches Modellieren.* Number 75 in Reihe Informatik. BI-Wiss.-Verlag, 1991.

[BCVA86] A. Mitchie, B. C. Vermuri und J. K. Aggarwal. Curvature-based representation of objects from range data. *Image and Vision Computing*, 4(2):107–114, May 1986.

[BE92] Marshall Bern und David Eppstein. *Computing in Euclidian Space*, chapter Mesh Generation and Optimal Triangulation, pages 23–90. World Scientific, 1992.

[Bru93] Ralf Bruggen. *3D-Computergrafik und -animation.* Addison-Wesley, 1993.

[BS87] I. A. Bronstein und K. A. Semendjajew. *Taschenbuch der Mathematik.* Verlag Harri Deutsch, 1987.

[Che89a] L. P. Chew. Constrained Delaunay Triangulations. *Algorithmica*, 4:97–108, 1989.

[Che89b] L. Paul Chew. Guaranteed Quality Triangular Meshes. Technical Report TR-89-983, Cornell University, Department of Computer Science, 1989.

[Che93] L. P. Chew. Guaranteed-Quality Mesh Generation for Curved Surfaces. In *Proceedings 9th Symp. Comp. Geometry*, pages 274–280. ACM, 1993.

[Chu92] Charles K. Chui. *An Introduction to Wavelets.* Academic Press Inc., 1992.

[dC83] Manfredo P. do Carmo. *Differentialgeometrie von Kurven und Flächen.* Vieweg, Braunschweig, 1983.

[DO91] Michael Dohmen und Walter Oberschelp. Mathematische Methoden für Bildverarbeitung und Computergrafik; Spezielle algorithmische Probleme der Computergrafik. Schriften zur Informatik und angewandten Mathematik 149, Lehrstuhl für Angewandte Mathematik insbesondere Informatik, Juli 1991.

[DZ91] Michael J. DeHaemer und Michael J. Zyda. Simplification of Objects Rendered by Polygonal Approximations. *Computing and Graphics*, 15(2):175–184, 1991.

[EDD+95] Matthias Eck, Tony DeRose, Tom Duchamp u.a. Multiresolution Analysis of Arbitrary Meshes. In *SIGGRAPH'95, Proceedings*, pages 173–182. University of Washington and Microsoft Research and Alias Research, 1995.

[Far93] S. Farestam. *Generating anisotropic unstructured grids for two dimensional manifolds.* PhD thesis, L'institut national polytechnique de toulouse, 1993.

[FGMS93] S. I. Feldman, D. M. Gay, M. W. Maimone und N. L. Schryer. A Fortran-to-C Converter. Technical Report 149, AT&T Bell Laboratories, Murray Hill, NJ 07974, March 1993.

[FS94] S. Farestam und R. B. Simpson. A framework for advancing front techniques of finite element mesh generation. Technical Report, Dep. of Comp. Science, University of Waterloo, september 1994.

[GB89] Duncan Gillies und Peter Burger. *Interactive Computer Graphics.* Addison Wesley, 1989.

[GGS95] M. H. Gross, R. Gatti und O. Staadt. Fast Multiresolution Surface Meshing. In *Proceedings of the IEEE Visualisation '95.* Computer Science Department ETH Zürich, 1995.

[Hag96] Yvonne Hager. 3D-Charakteranimation, Anforderungen der Performance Animation an die Modellierung. Diplomarbeit, Fachhochschule Köln, Fachbereich Photoingenieurwesen, Februar 1996.

[Han95] Fred Hantelmann. Schnelle Pipeline - Portabilität mit OpenGL. *iX Multiuser Multitasking Magazin*, pages 138–152, Oktober 1995.

[HDD+93] Hugues Hoppe, Tony DeRose, Tom Duchamp u.a. Mesh Optimization. In *SIGGRAPH'93, Proceedings*, pages 19–26. University of Washington, 1993.

[HDD+94] Hugues Hoppe, Tony DeRose, Tom Duchamp u.a. Piecewise Smooth Surface Reconstruction. University of Washington, 1994.

[HDDM92] H. Hoppe, Tony DeRose, Tom Duchamp und John McDonald. Surface reconstruction from unorganized points. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings*, number 2, pages 71–78, jul 1992.

[HL88] K. Ho-Le. Finite Element Mesh Generation Methods: A Review and Classification. *Computer Aided Design*, 20(1):27–38, Jan./Feb. 1988.

[HL92] Josef Hoschek und Dieter Lassser. *Grundlagen der geometrischen Datenverarbeitung.* Teubner, 2. neubearb. und erw. Auflage edition, 1992.

[Jäh91] Bernd Jähne. *Digitale Bildverarbeitung.* Springer Verlag, 1991.

[JDF94] Steven K. Feiner, James D. Foley, Andries van Dam. *Grundlagen der Computergrafik.* Addison Wesley Bonn Paris, 1994.

[JNW94] Tom Davis Jackie Neider und Mason Wao. *OpenGL Programming Guide.* Addison-Wesley Publishing Company, 1994. The official Guide to Learning OpenGL, SGI online book.

[KL96] Venkat Krishnamurthy und Marc Levoy. Fitting Smooth Surfaces to Dense Polygon Meshes. Dep. of Comp. Science, Stanford University, Stanford, 1996.

[Kry95] P. Krysl. Computational Complexity of the Advancing Front Triangulation. Technical Report, Dep. of Structural Mechanics, Czech Techn. University, 1995.

[KS93] P. Knupp und S. Steinberg. *The fundamentals of grid generation.* CRC Press, 1993.

[Law72] C. L. Lawson. Generation of a Triangular Grid With Applications to Contour Plotting. Tech. Memorandum 299, California Institute of Technology, Jet Propulsion Laboratory, 1972.

[Löh88a] R. Löhner. Generation of three-dimensional unstructured grids by the advancing-front method. In *Proc. AIAA 26th Aerospace Sciences Meeting, Reno*, 1988.

[Löh88b] R. Löhner. Some useful data structures for the generation of unstructured grids. *Communications in applied numerical methods*, 4:123–135, 1988.

[Löh94] R. Löhner. Surface Gridding from Discrete Data. In *4th international meshing roundtable.* SANDIA National Laboratories, 1994.

[LTW95] Yuencheng Lee, Demetri Terzopoulos und Keith Waters. Realistic Modelling for Facial Animation. In *SIGGRAPH 95.* University of Toronto and Digital Equipment Company, 1995.

[Men94] R. Mencl. Triangulierungen von ungeordneten Punktmengen auf Flächen und Körpern. Diplomarbeit, Universität Karlsruhe, Fakultät für Informatik, 1994.

[Men95] Robert Mencl. A Graph-Theoretic Approach to Surface Reconstruction. Technical Report 568/1995, Universität Dortmund - Fachbereich Informatik, 3 1995.

[Mül87] Horst Müller. Diskrete Algebraische Strukturen - Stichworte, Definitionen und Sätze. Arbeitsbericht 20, Institut für Mathematische Maschinen und Datenverarbeitung (Informatik), Friedrich-Alexander-Universität Erlangen-Nürnberg, 1987.

[Näh] Stefan Näher. LEDA - A Library of Efficient Data Types and Algorithms. ftp://ftp.th-darmstadt.de/pub/programming/languages/C++/class-libraries/LEDA/leda.html.

[Pöp95] Josef Pöpsel. Tex-Tour - Darstellung von texturierten 3d-Objekten. *c't Magazin für Computertechnik*, pages 344–348, November 1995.

[Rei91] Ulrich Reinhard. Kombinatorische Interpolation durch Delaunay-Triangulierung. Diplomarbeit, Institut für Informatik der Mathematischen Fakultät, Albert-Ludwigs-Universität Freiburg i. Brg., 1991.

[Ren96] Robert J. Renka. A Constrained Two-Dimensional Triangulation and the Solution of Closest Node Problems in the Presence of Barriers. *Transactions on Mathematical Software*, 22(1):1–8, March 1996. http://www.netlib.no/netlib/toms/751.

[RK94] D. Rypl und R. Krysl. Triangulation of 3D Surfaces. Technical Report, Dep. of Structural Mechanics, Czech Techn. University, 1994.

[SAS94] Jamshid Samareh-Abolhassani und John Stewart. Surface Grid Generation in a Parameter Space. *Journal of Computational Physics*, (113):112–121, 1994.

[SNTM91] V. Srinivasan, L. R. Nackman, J.-M. Tang und S.N. Meshkat. Automatic mesh generation using the symmetric axis tranformation of polygonal domains. In *IEEE Proceedings*, volume 80, pages 1485–1501, 9 1991.

[Spe95] Tom Sperlich. Digitale Kreaturen. *c't Magazin für Computertechnik*, pages 92–105, März 1995.

[SZL92] William J. Schroeder, Jonathan A. Zarge und William E. Lorensen. Decimation of Triangle Meshes. In *SIGGRAPH'95, Proceedings*, pages 65–70. General Electric Company Corporate Research and Development, 1992.

[TAG95] TAG/TRAUM_LAB*. Das unbekannte MdB. *http://www.khm.uni-koeln.de/projects/mdb/*, 1995.

[TM86] P. H. Todd und R. J. Y. McLead. Numerical Estimation of the Curvature of Surfaces. *Computer Aided Design*, 18:33–37, Jan/Feb 1986.

[Tur92] Greg Turk. Re-Tiling Polygonal Surfaces. In *SIGGRAPH'92, Proceedings*, pages 55–64. Department of Computer Science University of North Carolina, 1992.

[Wer94] Josie Wernecke. *The Inventor Mentor*. Addison Wesley, 1994.

[Wol90] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1990.

[WS92] D. J. Williams und M. Shah. A fast algorithm for active contours and curvature estimation. *Image Understanding*, 55(1):14–26, january 1992.

[WW92] Alan Watt und Mark Watt. *Advanced Animation and Rendering Techniques*. Addison-Wesley, 1992.