Von der Carl-Friedrich-Gauß-Fakultät für Mathematik und
Informatik der Technischen Universität Braunschweig
genehmigte Dissertation
zur Erlangung des Grades einer Doktor-Ingenieurin (Dr.-Ing.)

Kerstin Müller

NURBS und Unterteilungsflächen: Adaptive Visualisierung von Unterteilungsflächen und ihre Erweiterung auf nicht-uniforme Flächen.

Tag der mündlichen Prüfung: 14.02.2006

1.Referent: Prof. Dr. Dieter W. Fellner
2.Referent: Prof. Dr. Gerald E. Farin

eingereicht am 21.11.2005

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit „NURBS und Unterteilungsflächen: Adaptive Visualisierung von Unterteilungsflächen und ihre Erweiterung auf nicht-uniforme Flächen“ selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Braunschweig, den 21. November 2005

Kurzzusammenfassung

Zur Modellierung von Freiformflächen im CAD-Umfeld werden zur Zeit hauptsächlich nicht-uniforme rationale B-Spline-Flächen (NURBS) und Unterteilungsflächen verwendet. NURBS-Flächen sind die älteren sowie etablierteren Flächen und werden standardmäßig im Industriedesign verwendet. Unterteilungsflächen bieten eine flexiblere Topologie und nützliche Modellieroptionen, womit sie einen festen Platz in Modellierungspaketen wie z.B. Maya und Cinema4D gefunden haben.

Ziel dieser Arbeit ist es, die Vorteile von bikubischen NURBS- und Unterteilungsflächen in einem neuen, erweiterten Unterteilungsflächentyp (ESubs) zu vereinen, um damit ein breiteres Einsatzspektrum in Forschung und Industrie zu erhalten. Catmull-Clark- und bikubische NURBS-Flächen sind in den erweiterten Unterteilungsflächen durch die Verallgemeinerung enthalten. ESubs bieten jedoch Flächenformen an, die über die Möglichkeiten von NURBS- und Catmull-Clark-Flächen hinaus gehen. Zu den Kerneigenschaften des neuen Flächentyps ESubs gehören u.a. die Verwendung einer beliebigen, zwei-mannigfaltigen Topologie, die Generierung einer nicht-uniformen Fläche, die Möglichkeit Knotenintervalle einzeln an den Kanten zu modifizieren und die Verfügbarkeit von sogenannten „Special Features“. Für den praktischen Einsatz sind Limitpunktregeln unabdingbar, weshalb sie zusätzlich zu den Unterteilungsregeln im Rahmen dieser Arbeit entwickelt wurden. ESubs sind die einzigen nicht-uniformen Unterteilungsflächen als Verallgemeinerung von Catmull-Clark- und NURBS-Flächen, die Limitpunktregeln anbieten.

Der zweite Schwerpunkt dieser Arbeit ist die adaptive Visualisierung von Unterteilungsflächen. Sowohl bei der photorealistischen als auch bei der interaktiven Darstellung wird das Objekt adaptiv während des Renderingprozesses verfeinert. Dabei werden die Teile des Objektes, die zu einer Verbesserung des Bildes beitragen, in entsprechend höherer Unterteilungstiefe dargestellt. Die dazu entwickelten und implementierten Verfahren sind leicht auf weitere Unterteilungsflächentypen übertragbar und erlauben eine einfache Handhabung von Special Features sowie anderen Regelmodifikationen. Durch die adaptive Verfeinerung werden nur die notwendigen Flächenteile zur Generierung eines qualitativ hochwertigen Bildes berechnet, so dass Rechenzeit und Speicherplatz effizient genutzt werden.

Danksagung

Mein Dank richtet sich als erstes an Prof. Dr. D. W. Fellner, der mir die Möglichkeit gab, diese Arbeit an seinem Institut anzufertigen. Prof. Dr. G. Farin danke ich für sein Interesse an meiner Arbeit und die Übernahme des Koreferates. Meinen Eltern möchte ich für die ausgiebige Unterstützung während des Studiums und der Promotion danken.

Lars Reusche danke ich für die Bereitstellung seines Modellierungstools sowie für seine ausgezeichneten Arbeiten im Bereich Modellierung, Konvertierung und Trimming im Rahmen seiner Studien- und Diplomarbeit. Damit war ein professionelles Erstellen und Modifizieren der ESubs-Objekte erst effizient und anschaulich möglich. Bei Volker Settgast möchte ich mich für seine hervorragende Arbeit im Gebiet der interaktiven Visualisierung im Rahmen seiner Studienarbeit bedanken. Christoph Fünfzig danke ich für die konstruktiven Diskussionen und die vielen Tipps aus seinem umfangreichen Fachwissen in Computergraphik und Programmierung. Mein Dank gilt auch Torsten Techmann, der als Ansprechpartner für den MRT immer ein offenes Ohr für meine Fragen zum MRT hatte und die Dreieckskontrollnetze für die photorealistische Visualisierung implementierte. Rafael Janetzek danke ich für seine Implementierungen im Bereich Viereckskontrollnetze und adaptive photorealistische Visualisierung für ESubs. Für die konstruktiven Diskussionen möchte ich Andreas Müller danken. Seine aufschlussreichen und interessanten Vorlesungen im Bereich Kurven und Flächen waren sehr inspirierend und Grundlage für neue Ideen. Für die schönen Modelle und Szenenzusammenstellungen, die eine Arbeit im Bereich Computergraphik erst sehenswert machen, danke ich Volkswagen AG, Matthias Richter, Volker Settgast, Lars Reusche, Torsten Techmann, Rafael Janetzek und Christian Bode. Für die dazu passenden Texturen bedanke ich mich bei Andreas Halm sowie bei den AutorInnen von [MU], [Sio] und [Squ].

Mein Dank gilt auch meinen Kollegen an der TU Braunschweig, Andreas Müller, Arno Formella, Tanja Bongardt und Michael Braun fürs Korrekturlesen. Allen Genannten möchte ich für ihre Hilfsbereitschaft und ihr Engagement nochmals herzlich danken! Weiterhin möchte ich allen, die an dieser Stelle nicht namentlich erwähnt wurden, aber trotzdem zum Gelingen dieser Arbeit beigetragen haben, herzlich danken.

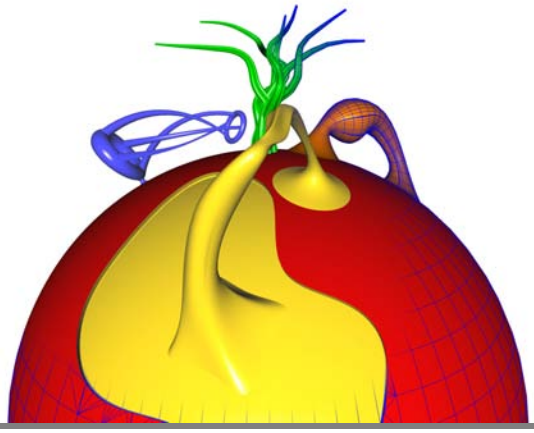
Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung



1.1 Einführung in die Thematik

Die Produkte eines erfolgreichen Freiformflächen-Designs begegnen uns im Alltag in vielerlei Ausprägungen: Beispielsweise findet man sie als wohl geformte Kaffeeautomaten oder als elegante Autokonsolen wieder. Im Entertainment-Bereich halten sie ebenso verstärkt Einzug, z.B. werden Hauptdarsteller in computergenerierten Filmen komplett mit Freiformflächen modelliert. Zur Generierung dieser Freiformflächenmodelle werden zur Zeit hauptsächlich nicht-uniforme rationale B-Spline-Flächen (NURBS) und Unterteilungsflächen verwendet.

Für den Bereich des Computer Aided Geometric Designs [BFK84] wurden zunächst von P. Bézier als auch von de Casteljau unabhängig voneinander die Bézier-Kurven und Flächen entwickelt [Béz88]. Ihre Forschungsergebnisse fanden Einzug in den meisten CAD-Systemen und dienten als Grundlage für eine Vielzahl von Weiterentwicklungen [FHK02]. Die darauf aufbauenden B-Spline-Flächen, eingeführt von J. Ferguson [Fer64], können als eine Verallgemeinerung von Bézier-Flächen angesehen werden, wie Gordon und Riesenfeld zeigten [GR74]. Die Erweiterung der B-Splines zu NURBS [PT95] hat sich in den heutigen CAD-Systemen zu einem Standard entwickelt und ist im Industriedesign unentbehrlich geworden. Definiert sind NURBS-Flächen durch ihre Parameterdarstellung, welche eine Auswertung der Fläche an beliebigen Parameterwerten erlaubt. Die dazu notwendigen Kontrollpunkte liegen in Form eines $n \times m$ Gitters vor.

Um den Nachteil dieser starren Topologie zu beheben, entwickelten Catmull und Clark [CC78] sowie Doo und Sabin [DS78] aus den Unterteilungsregeln der uniformen B-Spline-Flächen Unterteilungsregeln für allgemeine Topologien (zwei-mannigfaltig, mit und ohne Rand). Ihre Flächen entstehen durch das sukzessive Anwenden der Unterteilungsregeln auf das Kontrollnetz. Damit sind Unterteilungsflächen definiert als der Grenzwert einer unendlichen Folge von Unterteilungsschritten. Zusätzlich zur beliebigen Topologie können *Special Features* [HDD+94], wie z.B. scharfe Kanten und Spitzen, genutzt werden. Dadurch ist es dem Modellierer möglich, auf einfache Weise komplexe Modelle zu erzeugen und besondere Akzente im Objekt zu setzen. Die Verbesserung der Rechnerleistung begünstigte die Entwicklung der Unterteilungsflächen, so dass heutzutage auf eine Vielzahl von Arbeiten in diesem Gebiet zurückgegriffen werden kann [ZS99]. Neben den Loop-Unterteilungsflächen [Loo87], die auf Box-Splines basieren, zählen die Catmull-Clark-Unterteilungsflächen zu den am häufigsten eingesetzten Unterteilungsflächentypen der aktuellen Modellierungspakete wie z.B. Maya und Cinema4D.

Um mit NURBS ein komplexes Modell zu erhalten, werden mehrere NURBS-Patches generiert, passend *ausgeschnitten* (getrimmt) und geeignet aneinander gesetzt. Abbildung 1.1 zeigt ein solches Modell: Zur Modellierung dieses Teils der Autokarosserie kommen beispielsweise bereits 1578 (getrimmte) NURBS-Patches zum Einsatz.

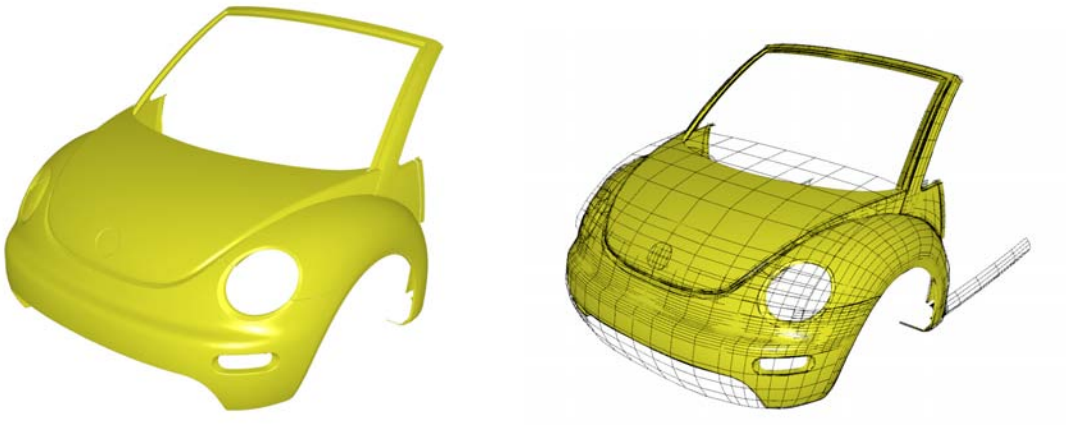


Abbildung 1.1: NURBS-Modell, bestehend aus 1578 getrimmten einzelnen NURBS-Patches. Die Kontrollnetze der einzelnen Patches sind schwarz markiert (rechts). Modell von Volkswagen AG aus [Reu05].

Zur Bildung komplexer Modelle sind Unterteilungsflächen (siehe Abbildung 1.2) nicht auf ein Trimming angewiesen. Eine freie Topologiewahl erlaubt es dem Modellierer, auf einfache Art aufwändige Modelle zu erzeugen. Damit fallen bei Unterteilungsflächen alle Probleme weg, die beim notwendigen Trimmen von NURBS entstehen. Mit Hilfe von Special Features können mit Unterteilungsflächen zusätzlich Akzente in das Modell eingefügt werden, die auf solche einfache Weise mit NURBS nicht zu erzielen sind. Auf Grund ihrer Vorteile in der Modellierung, sind Unterteilungsflächen verstärkt in den Modellierungspaketen des Entertainment-Bereiches zu finden. NURBS hingegen sind die älteren sowie im CAD-Bereich weit verbreiteten Freiformflächen. Sie bieten Vorteile durch ihre analytische Beschreibung. Ihr Einsatzschwerpunkt liegt im Industriedesign, wodurch die meisten Freiformflächenmodelle in diesem Gebiet als NURBS-Modelle vorliegen.

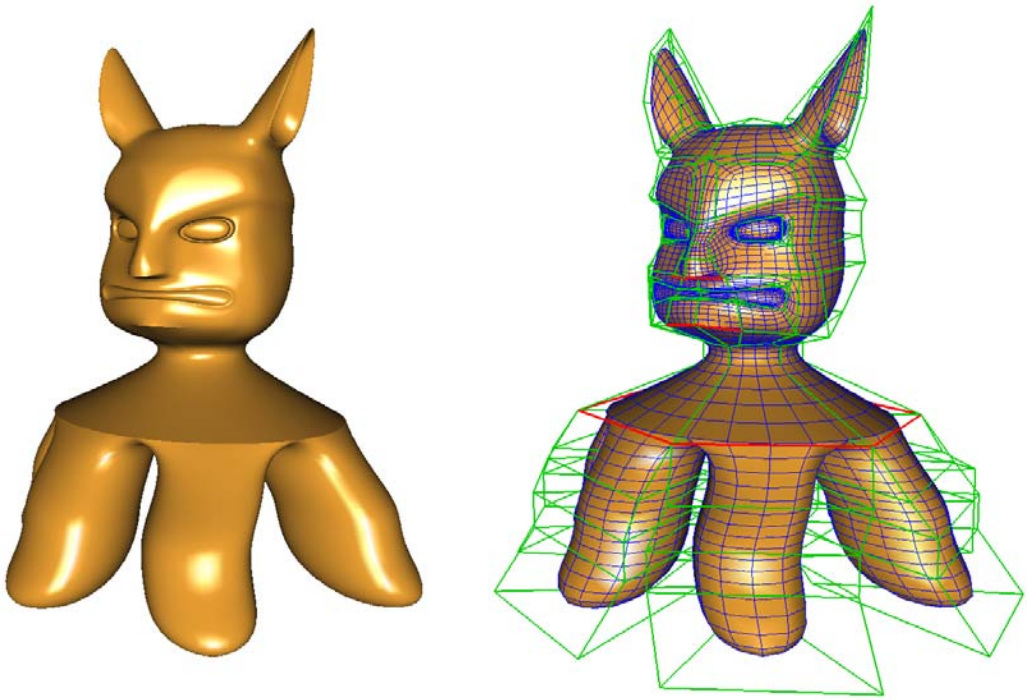


Abbildung 1.2: Unterteilungsfläche, gebildet aus *einem* Kontrollnetz mit Special Features. Die Special Features sind im grünen Kontrollnetz rot markiert (rechts).

1.2 NURBS-Flächen

In dieser Arbeit werden hauptsächlich bikubische, nicht-uniforme B-Spline-Flächen verwendet. Rationale Flächen können durch eine Homogenisierung erzielt werden. Eine bikubische NURBS-Fläche $S : [u_0, u_p] \times [v_0, v_q] \rightarrow \mathbb{R}^d$, mit $u_0, u_p, v_0, v_q \in \mathbb{R}$ und i.A. $d = 3$, ist durch topologisch gitterförmig angeordnete Kontrollpunkte und zwei Knotenvektoren festgelegt. Abbildung 1.3 zeigt das gitterförmige Kontrollnetz mit den dreidimensionalen Kontrollpunkten (grün markiert), die zwei Knotenvektoren und eine Tesselierung (blau eingezeichnet) der NURBS-Fläche. Die zwei Knotenvektoren der NURBS-Fläche $S(u,v)$ in u-Richtung $\{u_0, \ldots, u_p\}$ und in v-Richtung $\{v_0, \ldots, v_q\}$ bestehen aus je einer endlichen Folge monoton steigender reeller Zahlen u_i und v_j, die Knoten genannt werden. Als Knotenintervallwerte werden die Abstände $(u_{i+1} - u_i)$ und $(v_{j+1} - v_j)$, mit $i = 0, \ldots, p-1$ und $j = 0, \ldots, q-1$, bezeichnet. Ein Punkt $S(u,v)$ der NURBS-Fläche resultiert aus einer Konvexkombination der Kontrollpunkte. Die Gewichte für diese Konvexkombination lassen sich über die B-Spline-Basisfunktionen unter Berücksichtigung der Knotenvektoren bestimmen. Das Aussehen einer bikubischen NURBS-Fläche hängt somit von der Lage der Kontrollpunkte und den zwei Knotenvektoren ab. Statt Knotenintervallwerte wird im Folgenden die Kurzfassung Knotenintervalle benutzt (angelehnt an die Notation von [SSS98]). Bei

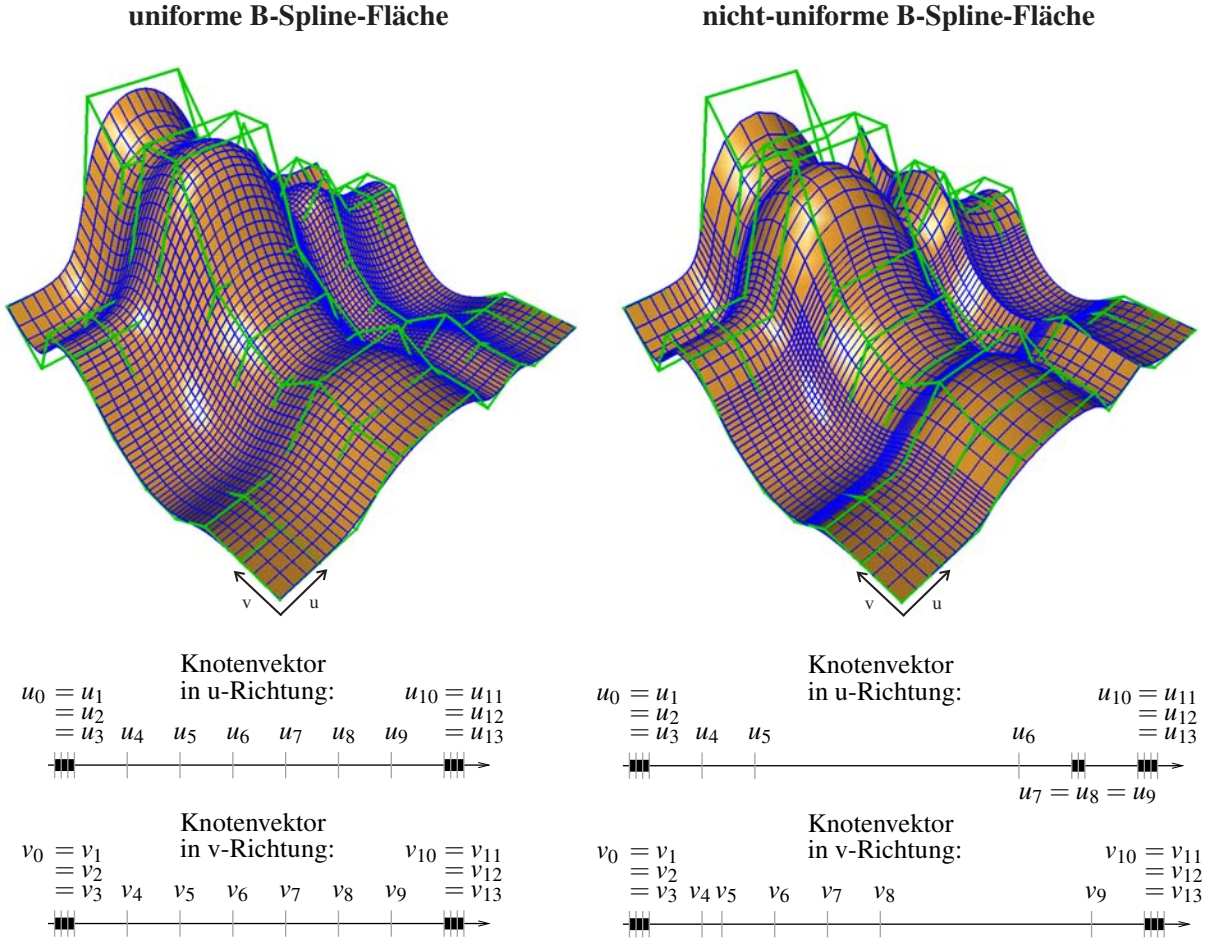


Abbildung 1.3: Bikubische, uniforme und nicht-uniforme B-Spline-Fläche mit Kontrollnetz (grün) und den zwei Knotenvektoren in u- und v-Richtung. Die Tesselierung (blau) der Fläche ist bei beiden Flächen an den gleichen Parameterwerten vorgenommen worden. Deutlich zu sehen ist die Auswirkung der nicht-uniformen Knotenvektoren (rechts) am lokalen Dehnen und Zusammenziehen der Fläche. Zwei innere Null-Knotenintervalle bewirken eine lokale Reduktion der Stetigkeit auf C^0, so dass eine scharfe Kante bei den Parameterwerten (u_7, v), $v \in [v_0, v_{13}]$ durch die Fläche läuft.

der meist verwendeten clamped NURBS-Fläche sind die drei ersten bzw. drei letzten Knotenintervalle der zwei Knotenvektoren im bikubischen Fall nullwertig [PT95], die übrigen Knotenintervalle heißen innere Knotenintervalle. Liegt eine uniforme Fläche vor, so sind die inneren Knotenintervalle identisch (siehe Abbildung 1.3 links). Nicht-uniforme Knotenintervalle bewirken eine nicht-uniforme Fläche und führen zu einem lokalen Dehnen bzw. Zusammenziehen der Fläche (siehe Abbildung 1.3 rechts). Durch Verringerung des Knotenintervalles kann das lokale Zusammenziehen der Fläche bis zu einer Reduktion der Stetigkeit gesteigert werden, wie in Abbildung 1.3 rechts im Knotenvektor in u-Richtung demonstriert. Die Möglichkeit, Knotenintervalle in den Knotenvektoren beliebig zu modifizieren, bietet somit dem Designer ein wichtiges Werkzeug zur individuellen Formgestaltung seines Produktes.

1.3 Unterteilungsflächen

Neben den bikubischen NURBS-Flächen kommen in der vorliegenden Arbeit primär approximierende Unterteilungsflächen zum Einsatz, die je nach verwendetem Unterteilungsschema Dreiecksnetz- bzw. Viereckснetz-erzeugend sind und beim Unterteilen neue Punkte ins Netz einfügen (Klassifizierungskriterien nach [ZS99]).

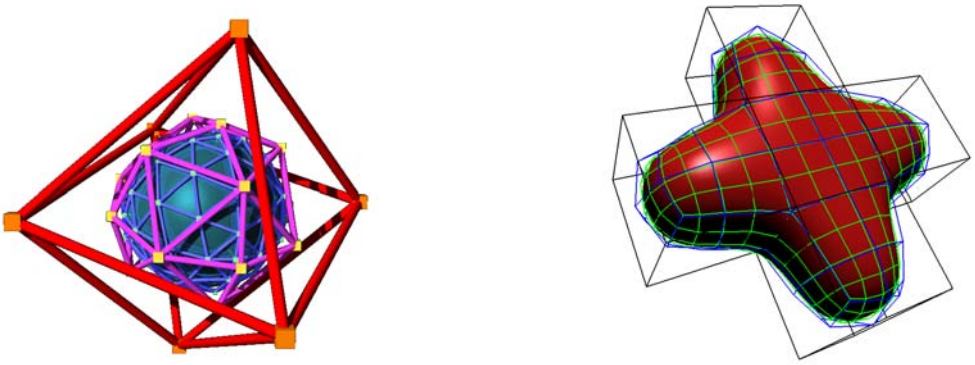


Abbildung 1.4: Loop- und Catmull-Clark-Unterteilungsflächen mit Kontrollnetzen in drei Unterteilungstiefen und zugehörige Limitflächen. Beide Unterteilungsflächen sind approximierend und fügen neue Punkte beim Unterteilen ins Netz ein. Loop-Flächen (links) sind Dreiecksnetz-erzeugend. Catmull-Clark-Flächen generieren ein Viereckснetz. Modell links von Torsten Techmann, Modell rechts von Volker Settgast.

Eine solche Unterteilungsfläche ist durch ihre Unterteilungsregeln definiert. Das Durchführen der Unterteilungsregeln auf dem Kontrollnetz der Fläche liefert ein verfeinertes Kontrollnetz. Abbildung 1.5 links zeigt zwei Unterteilungsschritte. Werden diese Unterteilungsregeln unendlich oft angewendet, so erhält man die Limitfläche, wie in Abbildung 1.5 mitte zu sehen. Für einen Kontrollpunkt des Kontrollnetzes lässt sich sein zugehöriger Punkt auf der Limitfläche bestimmen, die dazu benötigten Regeln heißen Limitpunktregeln. Ein Anwenden der Limitpunkregeln auf dem Kontrollnetz ergibt die Limitpunkte der Kontrollpunkte und führt somit zu einer Tesselierung der Limitfläche. Abbildung 1.5 rechts präsentiert die Tesselierung der Unterteilungsfläche in den drei gezeigten Unterteilungstiefen.

Die Kontrollpunkte des Kontrollnetzes verändern ihre Position beim Unterteilen, das Kontrollnetz konvergiert und „schrumpft“ somit zur Limitfläche (siehe Abbildung 1.5). Aus diesem Grund ist eine Tesselierung der Limitfläche, insbesondere während der Modellierungsphase, einer Darstellung mit Kontrollpunkten vorzuziehen. Mit Limitpunkten ist eine exakte Positionierung des zu modellierenden Objektes erst möglich. Abbildung 1.6 zeigt ein Beispiel, bei dem zwei Unterteilungsflächen an einem gegebenen Punkt verbunden wurden. Limitpunktregeln sind somit für den praktischen Einsatz im CAD-Umfeld unabdingbar.

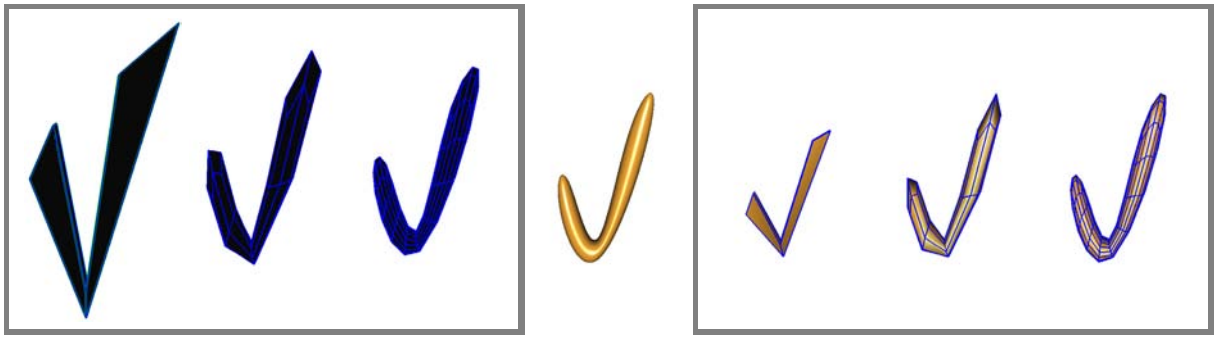


Abbildung 1.5: Linke Box: Das Anwenden der Unterteilungsregeln auf dem Kontrollnetz liefert ein verfeinertes Kontrollnetz. Mit jedem Unterteilungsschritt approximiert das Kontrollnetz die Limitfläche genauer. Mitte: Limitfläche. Rechte Box: Um eine Tesselierung der Limitfläche zu erhalten, werden in den drei Unterteilungstiefen die Limitpunktregeln angewendet, die für jeden Kontrollpunkt seinen Limitpunkt auf der Limitfläche berechnen.

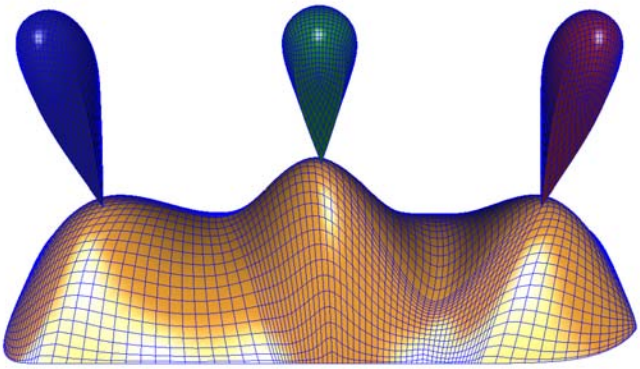


Abbildung 1.6: Darstellung von ESubs-Unterteilungsflächen in Form einer Tesselierung ihrer Limitfläche mit Hilfe ihrer Limitpunktregeln. Das rote, blaue und grüne Modell ist mit dem orangefarbenen Modell über gemeinsame Limitpunkte verbunden. Somit ist ihr Kontakt unabhängig von der Unterteilungstiefe.

1.4 Motivation

Um über die Vorteile von NURBS- und Catmull-Clark-Unterteilungsflächen zu verfügen, ist eine Verallgemeinerung beider Flächentypen in Form einer erweiterten Unterteilungsfläche wünschenswert. Die Entwicklung einer erweiterten Unterteilungsfläche, basierend auf den bikubischen NURBS-Unterteilungsregeln, ist somit analog zur Erweiterung der uniformen bikubischen B-Spline-Flächen zu Catmull-Clark-Flächen sinnvoll. Die Nutzung der Modellierwerkzeuge beider Flächentypen in einem neuen Flächentyp bietet entscheidende Vorteile im Modellierungsprozess und erlaubt zudem die Verwendung der bereits bestehenden Modelle beider Flächen (siehe auch Abbildung 1.7).

Als Verallgemeinerung von Catmull-Clark- und NURBS-Flächen existieren bislang zwei weitere Unterteilungsflächen, die allerdings keine Limitpunktregeln anbieten. Limitpunktregeln sind jedoch unverzichtbar um einen neuen Flächentyp im CAD-Bereich sowie in der adaptiven Visualisierung (siehe

auch Abbildung 1.8) zu verwenden. Daher ist es notwendig eine erweiterte Unterteilungsfläche (ESubs) mit Limitpunktregeln zu entwickeln und zu implementieren sowie in praktischen Anwendungen zu testen.

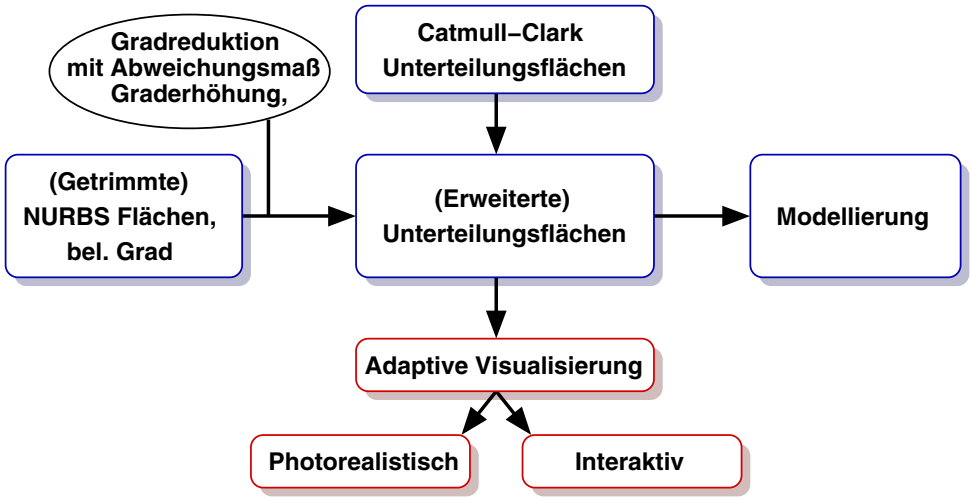


Abbildung 1.7: Überblick.

Nach der Modellierungsphase besteht der Bedarf nach einer effizienten Visualisierung der generierten Unterteilungsflächenobjekte (siehe Abbildung 1.7). Die Anzahl der Faces wächst exponentiell beim Unterteilen, daher ist es sinnvoll, nur die Teile des Objektes zu berechnen und darzustellen, die an der Beleuchtungsrechnung teilnehmen. Zur Erzeugung eines qualitativ hochwertigen Bildes unter effizienter Nutzung der Rechenleistung und des Speichers sollen nur die Objektteile berechnet und visualisiert werden, die zu einer Verbesserung der Bildqualität beitragen.

Abbildung 1.8 zeigt dazu ein adaptives Verfahren in schematischer Darstellung: Die Silhouettenbereiche werden weiter verfeinert als die übrigen Objektteile, die kameraabgewandten Faces werden hier lediglich in geringster Unterteilungstiefe präsentiert. Die Notwendigkeit von Limitpunktregeln bei der adaptiven Visualisierung unterstreicht Abbildung 1.8c: Werden Kontrollpunkte zur Darstellung des Objektes gewählt, so erhält man ein unerwünschtes Aussehen des Objektes. Die Verwendung von Limitpunkten liefert hingegen das erwartete Erscheinungsbild der Fläche, wie Abbildung 1.8d demonstriert.

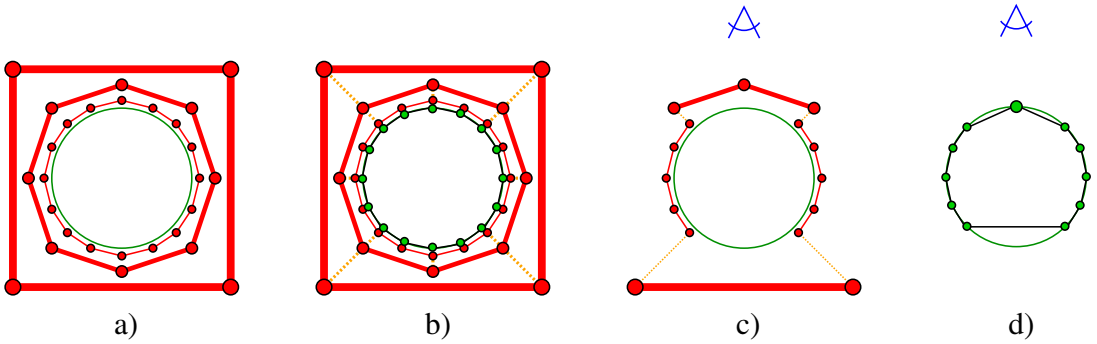


Abbildung 1.8: Schematische Darstellung einer Unterteilungsfläche: Die roten Linien stellen das Kontrollnetz dar. Die Liniendicke nimmt mit zunehmender Unterteilungstiefe ab. Die grünen Punkte sind die Limitpunkte und die gestrichelte orangefarbene Linie markiert die Zugehörigkeit eines Kontrollpunktes zu seinem Limitpunkt. Die schwarze Linie zeigt die zugehörige Tesselierung der Limitfläche (grün) an. Verwendet man zur adaptiven Visualisierung Kontrollpunkte, so hat das resultierende Objekt ein unerwünschtes Aussehen (c). Mit den Limitpunkten erhält man das erwartete Ergebnis (d) des Objektes.

1.5 Ziele

Ein Ziel dieser Arbeit ist es, die Vorteile von bikubischen NURBS und Unterteilungsflächen in einem neuen Unterteilungsflächentyp zu vereinen, um damit ein breiteres Einsatzspektrum in Forschung und Industrie zu erhalten. Zu den Kerneigenschaften des neuen Flächentyps ESubs gehören u.a. die Verwendung einer beliebigen Topologie (zwei-mannigfaltig, mit und ohne Rand), die Generierung einer nicht-uniformen Fläche, die Möglichkeit Knotenintervalle einzeln zu modifizieren und die Nutzung von Special Features. Für den praktischen Einsatz sind Limitpunktregeln unabdingbar, weshalb sie zusätzlich zu den Unterteilungsregeln entwickelt wurden.

Üblicherweise werden bei der Entwicklung einer neuen Unterteilungsfläche zunächst die Unterteilungsregeln hergeleitet. Aus diesen Unterteilungsregeln lässt sich eine Unterteilungsmatrix S bilden. Mit Hilfe der Unterteilungsmatrix S kann ein Unterteilungsschritt für einen Kontrollpunkt P^0 und seine benachbarten Kontrollpunkte durchgeführt werden. Wird S unendlich oft auf dieses partielle Kontrollnetz angewendet, so erhält man den Limitpunkt von P^0. Die Regeln zur Berechnung der Limitpunkte können aus der Unterteilungsmatrix mit Hilfe ihres größten Eigenwertes bestimmt werden [ZS99]. Da ESubs bei der geforderten Verwendung von unterschiedlichen Knotenintervallen auf gegenüber liegenden Seiten eines Vierecks im Allgemeinen nicht-stationär sind, verändert sich die Unterteilungsmatrix S bei jedem Unterteilungsschritt. Eine Eigenanalyse wird aus diesem Grunde bei nicht-stationären Verfahren nicht durchgeführt. Zur Bestimmung der Limitpunktregeln muss daher eine neue Herangehensweise gewählt werden, womit erstmals ein Verfahren zur Verfügung steht, das Limitpunktregeln für ein nicht-stationäres Schema liefert. Der Hauptvorteil von ESubs gegenüber den bereits vorhandenen nicht-uniformen Unterteilungsflächen (siehe Abschnitt 3.1) als Verallgemeinerung von NURBS- und Catmull-Clark-Flächen liegt somit in der Verfügbarkeit von Limitpunktregeln.

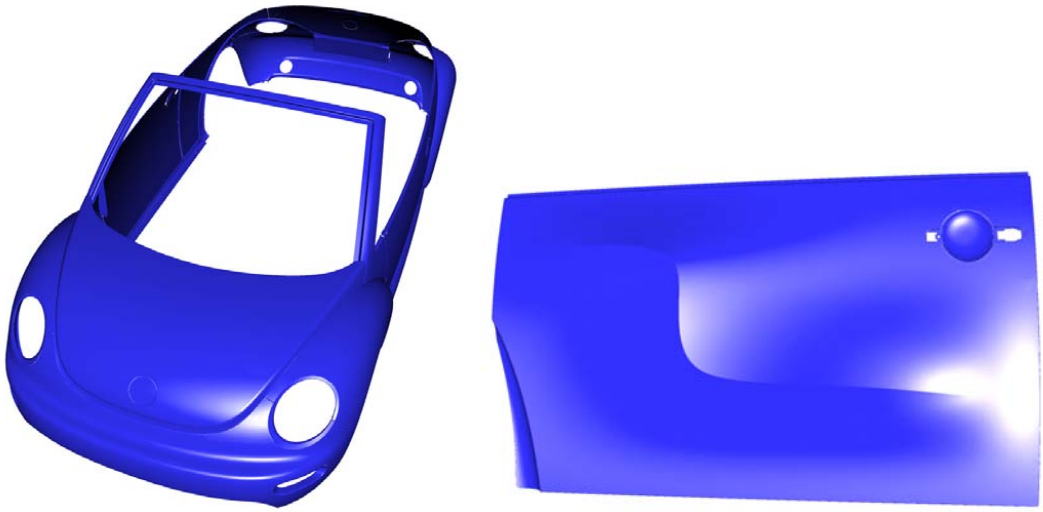


Abbildung 1.9: Zu ESubs konvertiertes getrimmtes NURBS-Modell (links). In einem weiteren Verarbeitungsschritt sind Special Features hinzugefügt worden (rechts). Modell von Volkswagen AG.

Mit ESubs lassen sich sowohl polygonale Objekte als auch Catmull-Clark-Modelle verwenden. Ebenso können bikubische NURBS-Modelle genutzt werden. NURBS-Flächen beliebigen Grades stehen nach einer Konvertierung mit einer gegebenen Abweichungstoleranz zu bikubischen Flächen ebenfalls zur Verfügung (siehe Abbildung 1.9). Ein Trimmen der konvertierten getrimmten NURBS-Modelle ist weiterhin möglich. Bei einer Neuerstellung eines ESubs-Modells ist ein Trimmen jedoch nicht mehr notwendig (siehe Abbildung 1.2). Dadurch entfällt die komplette Problematik des Trimmingprozesses. Stattdessen kann ein Modellierer zur Gestaltung seines Objektes auf eine beliebige Topologie zurückgreifen, Special Features setzen und Knotenintervalle modifizieren. Somit bieten ESubs die Möglichkeit, komplexe

Modelle mit *einem zusammenhängenden* Kontrollnetz zu erzeugen statt eine Vielzahl getrimmter Patches nutzen zu müssen. Die Bearbeitung eines zusammenhängenden Kontrollnetzes vereinfacht die Handhabung des Modells beispielsweise deutlich in der Texturierung und Modellierung. Die Tesselierung einer solchen Fläche liefert zudem ein gleichmäßigeres Dreiecksnetz als die Tesselierung vieler getrimmter Patches. Catmull-Clark und bikubische NURBS sind in den erweiterten Unterteilungsflächen durch die Verallgemeinerung bereits inbegriffen: Sind die Knotenintervalle uniform, so ist die resultierende Fläche eine Catmull-Clark-Fläche. Wird ein Viereckснetz mit regulärer Topologie und einer Knotenintervallbelegung analog zu NURBS verwendet, dann erhält man eine bikubische NURBS-Fläche.

Der zweite Schwerpunkt dieser Arbeit ist die adaptive Visualisierung von Unterteilungsflächen. Sowohl bei der photorealistischen als auch bei der interaktiven Darstellung wird das Objekt adaptiv *während* des Renderingprozesses verfeinert. Dabei werden die Teile des Objektes, die zu einer Verbesserung des Bildes beitragen, in entsprechend höherer Unterteilungstiefe dargestellt, um sowohl die Rechenleistung als auch den Speicher effizient zur Generierung eines qualitativ hochwertigen Bildes zu nutzen.

Bei der interaktiven Visualisierung liegt der Fokus auf einer geringen Rechenzeit zur Erhaltung einer akzeptablen Framerate. Bild für Bild wird für jedes Patch der Unterteilungsfläche eine geeignete Unterteilungstiefe ausgewählt. Auswahlkriterien zur Verfeinerung eines Patches sind dabei beispielsweise seine projizierte Patchgröße, die Silhouettenzugehörigkeit und seine Krümmung. Nur die notwendigen Vierecke bzw. Dreiecke werden dann zur Hardware gesendet und erzeugen das gewünschte Bild in der gewählten Qualitätsstufe. Durch ein dynamisches Verschieben der Unterteilungstiefen der einzelnen Patches ist es möglich einen gegebenen Frameratenbereich einzuhalten. Mit Hilfe einer konstanten Framerate kann die so genannte *Simulator Sickness* [HKB03] vermieden werden. Die patchweise Tesselierung der Fläche findet in einer speziellen Datenstruktur statt, welche *unabhängig* von der Größe des Ausgangskontrollnetzes ist und das Ausgangskontrollnetz *nicht verändert*. Die unveränderte Beibehaltung des Ausgangskontrollnetzes ist eine wichtige Voraussetzung für den Einsatz des Verfahrens in modernen Szenengraphsystemen wie OpenSG [RVB02], in welches der entwickelte Algorithmus integriert wurde.

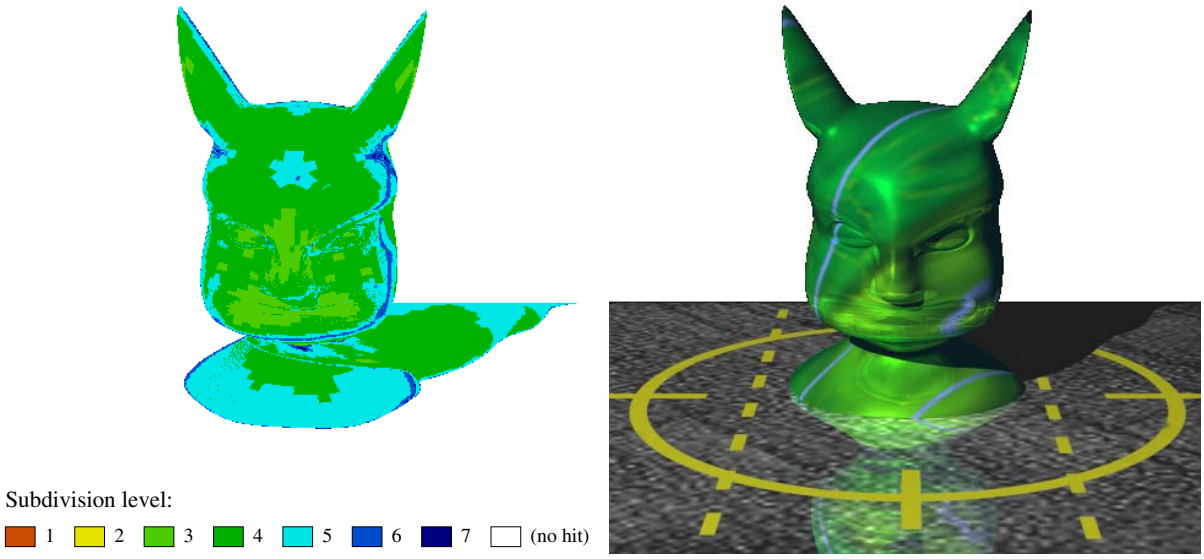


Abbildung 1.10: Adaptives Raytracing einer ESubs-Fläche in Abhängigkeit ihrer Krümmung, Silhouettenzugehörigkeit und projizierter Patchgröße: Resultierendes Originalbild (rechts) und Tiefenkarte (links), dargestellt für Sehstrahlen und Schattenfühler. Modell von Volker Settgast.

Bei der photorealistischen Visualisierung [Fel92] steht die Qualität des Bildes im Vordergrund. Im Rahmen der vorliegenden Arbeit wurden Unterteilungsflächen in den Raytracer und die Radiositybe-rechnung des Modular Rendering Tools (MRT) [Fel96] des Instituts für ComputerGraphik integriert. Wie auch bei der interaktiven Darstellung wurde beim photorealistischen Rendern der Ansatz der adap-

tiven Verfeinerung konsequent verfolgt. Bei der hierarchischen Radiosity ergibt sich die adaptive Verfeinerung des Unterteilungsflächenobjektes Blickpunkt unabhängig aus der Beleuchtungsberechnung: Lichtaustausch, Patchgröße und Krümmung bestimmen anhand der gegebenen Parameter die adaptive Verfeinerung. Zum Raytracen von Unterteilungsflächen wurde ein neues Verfahren entwickelt und implementiert. Ziel des neuen Algorithmus ist es ebenfalls, unter effizienter Nutzung der Rechenleistung und des Speichers, ein qualitativ hochwertiges Bild zu generieren. Die Unterteilungsfläche wird dazu *während* des Raytracings adaptiv verfeinert: Für jeden zu testenden Strahl wird die Unterteilungsfläche lokal *entlang des Strahls* in Abhängigkeit von Krümmung, projizierter Patchgröße und Silhouettenzugehörigkeit verfeinert. Abbildung 1.10 illustriert die adaptive Arbeitsweise des Algorithmus mit Hilfe einer Tiefenkarte und dem dazugehörigen Ergebnisbild. Dieser Algorithmus ist robust, schnell, einfach umzusetzen und für Catmull-Clark-, Loop- und ESubs-Flächen implementiert.

1.6 Überblick über die Dissertation

Die vorliegende Dissertation führt die notwendigen Grundlagen in Kapitel 2 ein. In diesem Kapitel werden elementare Kurven und Flächentypen definiert, weiterhin werden Unterteilungsregeln und Limitregeln für NURBS-Kurven und Flächen hergeleitet. Für Catmull-Clark- und Loop-Unterteilungsflächen, die zwei meist verwendeten Unterteilungsflächen, sind die Regeln ebenfalls angegeben. Kapitel 3 befasst sich mit der Herleitung der erweiterten Unterteilungsflächen, die im Rahmen dieser Dissertation entwickelt und implementiert wurden. Detailliert werden die Unterteilungs- und Limitpunktregeln beschrieben, Konvergenz und Stetigkeit werden untersucht und Special Features behandelt. Die erweiterten Unterteilungsflächen sind somit Themenschwerpunkt sowohl dieses Kapitels als auch des nächsten Kapitels 4, in dem die Modellierung mit ESubs und die Verwendung bestehender NURBS, Catmull-Clark und polygonaler Objekte im Mittelpunkt steht. Kapitel 5 wendet sich dem zweiten Themengebiet, der adaptiven Visualisierung von Unterteilungsflächen zu. Dazu wurden neue Algorithmen entwickelt und implementiert, die in diesem Kapitel ausführlich beschrieben werden. Nach einer allgemeinen Einführung in die Thematik steht in Abschnitt 5.1 die interaktive Darstellung im Mittelpunkt. Das neue Verfahren wird detailliert erläutert und analysiert. Visuelle Ergebnisse und aussagekräftige Messergebnisse runden die Untersuchung ab. Im anschließenden Abschnitt 5.2 wird die photorealistische Visualisierung behandelt: Die Methoden zur Radiosityberechnung mit Unterteilungsflächen werden zunächst kurz vorgestellt. Der Algorithmus zum Raytracen von Unterteilungsflächen wird detailliert beschrieben und seine Ergebnisse ausführlich erläutert. Das abschließende Kapitel 6 enthält eine Zusammenfassung und einen Ausblick der Arbeit. Der Beitrag der Dissertation wird darin nochmals präzisiert.

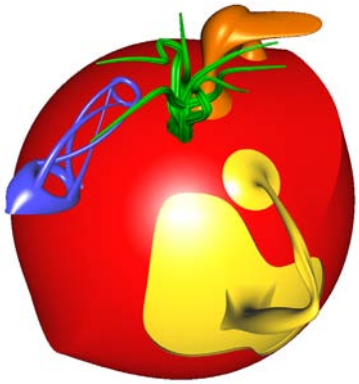
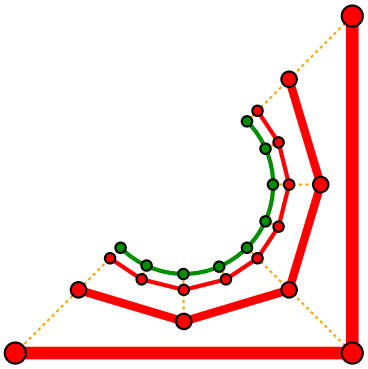


Abbildung 1.11: Zu ESubs konvertiertes getrimmtes NURBS-Modell, weiterverarbeitet mit den zusätzlichen Modellierungsoptionen der ESubs. Modell von Matthias Richter und Lars Reusche.

Kapitel 2

Kurven und Flächen



Zum Einstieg in die Thematik der Kurven und Flächen wird mit einer kurzen historischen Einführung in die Computer Aided Geometric Design (CAGD) Welt [BM99] begonnen. Im Jahre 1959 entwickelte P. de Casteljau im Laufe seiner Arbeit bei Citroën ein elegantes Verfahren [dC63], um aus n gegebenen Kontrollpunkten eine Kurve respektive Fläche zu konstruieren. P. de Casteljau bildete seine Kurven und Flächen mit Hilfe eines iterativen Prozesses aus Konvexkombinationen der Kontrollpunkte. Die gleiche Kurve, jedoch auf andere Art und Weise, leitete P. Bézier [Béz88] bei Renault aus dem Schnitt zweier, in einem Würfel liegenden Viertel-Zylinder her. Einen weiteren Meilenstein in der Geschichte des CAGDs legte C. de Boor 1972 mit der Entdeckung der Rekursionsformeln für B-Splines [dB72]. Die Konstruktion von Bézier-Kontrollpunkten aus B-Spline-Kontrollpunkten erforschten und publizierten 1977/1978 P. Sablonnière [Sab78] und W. Boehm [Boe77] unabhängig voneinander. Ein Verfahren zum Einfügen von Knoten und der Oslo Algorithmus folgten 1980 sowohl durch R. F. Riesenfeld [CLR80] als auch durch W. Boehm [Boe80]. P. de Casteljau entwickelte schließlich eine Bündelung und Zusammenfassung all dieser Kurven und Flächen zu einer vollständigen Repräsentation, den so genannten *Blossoms*. L. Ramshaw publizierte 1987 die Blossoms in [Ram87].

Das Kapitel beginnt mit einer Einführung der Blossoms, führt dann auf deren Grundlage Bézier und NURBS-Kurven sowie Flächen ein. Darauf aufbauend werden Unterteilungsverfahren vorgestellt. Das Kapitel schließt mit der Betrachtung der zwei meist verwendeten Unterteilungsflächen Catmull-Clark und Loop, die auch in dieser Arbeit zum Einsatz kommen.

2.1 Blossoms

Blossoms bieten die Möglichkeit, Bézier, B-Spline und NURBS-Kurven sowie Flächen auf einfache Art und Weise in einer Notation zu vereinen. Sie werden zunächst anschaulich als Verallgemeinerung des de Casteljau Algorithmus eingeführt, ihre wichtigen Eigenschaften und Charakteristiken werden im Anschluss erläutert.

Blossoms als Verallgemeinerung des de Casteljau Algorithmus

Bei dem herkömmlichen de Casteljau Algorithmus [Far94] ist ein Punkt auf einer Kurve im $\mathbb{E}^d$ mit dem Parameterwert $t \in \mathbb{R}$ und den Kontrollpunkten $b_0^0, \ldots, b_n^0 \in \mathbb{R}^d$, üblicherweise $d = 2,3$, definiert durch

$$b_i^r(t) = \frac{u_1 - t}{u_1 - u_0} \cdot b_i^{r-1}(t) + \frac{t - u_0}{u_1 - u_0} \cdot b_{i+1}^{r-1}(t) \tag{2.1}$$

mit $r = 1, \ldots, n, \quad i = 0, \ldots, n-r, \quad u_0 < u_1$ und $b_j^0(t) = b_j^0, \quad j = 0, \ldots, n.$

In schematischer Darstellung ergibt sich somit für einen Punkt b_0^3 mit dem Parameterwert $\tau \in \mathbb{R}$ und $n = 3$ (siehe auch Abbildung 2.1):

$$
\begin{array}{llll}
b_3^0 & & & \\
b_2^0 & b_2^1 := b_2^1(\tau) & & \\
b_1^0 & b_1^1 := b_1^1(\tau) & b_1^2 := b_1^2(\tau) & \\
b_0^0 & b_0^1 := b_0^1(\tau) & b_0^2 := b_0^2(\tau) & b_0^3 := b_0^3(\tau)
\end{array}
$$

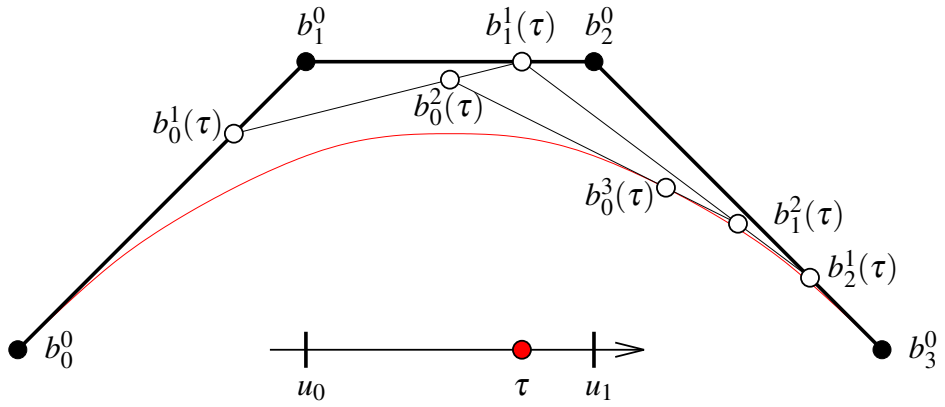


Abbildung 2.1: Herkömmlicher de Casteljau Algorithmus.

Der verallgemeinerte de Casteljau Algorithmus verwendet in jeder Iterationsstufe r, mit $r = 1, \ldots, n$, einen Parameter $t_r \in \mathbb{R}$. Damit lässt sich die Rekursionsgleichung aus 2.1 erweitern zu:

$$b_i^r[t_r, \ldots, t_0] = \frac{u_1 - t_r}{u_1 - u_0} \cdot b_i^{r-1}[t_{r-1}, \ldots, t_0] + \frac{t_r - u_0}{u_1 - u_0} \cdot b_{i+1}^{r-1}[t_{r-1}, \ldots, t_0] \tag{2.2}$$

$b_0^n[t_n, \ldots, t_0]$ ist somit eine n-variante Funktion und beschreibt ein Gebiet im $\mathbb{E}^d$. Einen Punkt b_0^3 mit den drei Parameterwerten τ_1, τ_2, τ_3 erhält man in schematischer Darstellung durch (siehe auch Abbildung 2.2):

$$
\begin{array}{llll}
b_3^0 & & & \\
b_2^0 & b_2^1 := b_2^1[\tau_1] & & \\
b_1^0 & b_1^1 := b_1^1[\tau_1] & b_1^2 := b_1^2[\tau_1, \tau_2] & \\
b_0^0 & b_0^1 := b_0^1[\tau_1] & b_0^2 := b_0^2[\tau_1, \tau_2] & b_0^3 := b_0^3[\tau_1, \tau_2, \tau_3]
\end{array}
$$

Das Verfahren reduziert sich auf den ursprünglichen de Casteljau Algorithmus, wenn stets der gleiche Parameter $t = t_1 = \ldots = t_n$ verwendet wird. Die resultierende Kurve $b[t, \ldots, t] = b(t)$ ist dann eine Bézier-Kurve, die durch die Kontrollpunkte $b_0^0, \ldots, b_n^0$ definiert ist. Die Funktion

$$b[t_1, \ldots, t_n] \quad \text{mit} \quad b : \mathbb{R}^n \to \mathbb{R}^d \tag{2.3}$$

gebildet aus der rekursiven Gleichung 2.2, heißt *Blossom* von $b(t)$.

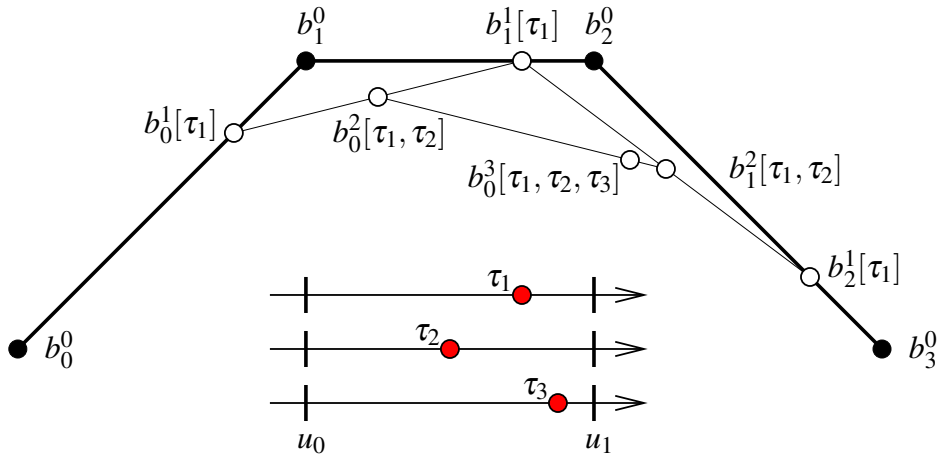


Abbildung 2.2: Verallgemeinerter de Casteljau Algorithmus.

Definition und Eigenschaften

Allgemeiner lassen sich Blossoms über symmetrische Polynome definieren:

Definition 1 (Blossom). *Das Blossom einer polynomialen Kurve*

$$b(t) = a_0 + a_1 t + \ldots a_n t^n, \qquad a_j \in \mathbb{R}^d, \quad j = 0,\ldots,n, \quad t \in \mathbb{R}$$

vom Grad n ist eine n-variante Funktion $b[t_1,\ldots,t_n]$, für die gilt:

1. *$b[t_1,\ldots,t_n]$ ist multiaffin:*

$$\alpha+\beta = 1 \quad \Rightarrow \quad b[\alpha\cdot x+\beta\cdot y, t_2,\ldots,t_n] = \alpha\cdot b[x,t_2,\ldots,t_n]+\beta\cdot b[y,t_2,\ldots,t_n]$$

2. *$b[t_1,\ldots,t_n]$ ist symmetrisch:*

$$b[t_1,\ldots,t_n] = b[\pi(t_1),\ldots,\pi(t_n)]$$

   für jede Permutation $\pi(t_1),\ldots,\pi(t_n)$ von $t_1,\ldots,t_n$.

3. *$b[t_1,\ldots,t_n]$ stimmt mit $b(t)$ in der Diagonalen überein:*

$$b[t,\ldots,t] = b(t)$$

Die Funktion $b[t_1,\ldots,t_n]$ (2.3) mit n Variablen erfüllt die drei Eigenschaften des Blossoms (siehe Definition 1):

- Die Multiaffinität ergibt sich direkt aus dem Algorithmus. Jeder Iterationsschritt ist eine affine Abbildung.
- Das Ergebnis des verallgemeinerten de Casteljau Algorithmus ist unabhängig von der Reihenfolge der Auswertung der n Parameter $t_1,\ldots,t_n$.
- Gilt $t = t_1 = \ldots = t_n$, so ist $b[t,\ldots,t] = b(t)$.

Blossom der Bézier-Kurve

Die Bézier-Kurve $b(t)$ mit $b : [0,1] \to \mathbb{R}^d$ sei in expliziter Darstellung, d.h. als Linearkombination der Bernsteinpolynome $B_j^n(t)$ [Far94], gegeben. Um die Formel des Blossoms $b[t_1,\ldots,t_n]$ der Bézier-Kurve $b(t)$ herzuleiten, wird im ersten Schritt $b(t)$ in die Form $a_0 + a_1 \cdot t + a_2 \cdot t^2 + \ldots + a_n \cdot t^n$ gebracht:

$$b(t) = \sum_{j=0}^{n} b_j^0 \cdot B_j^n(t) = \sum_{j=0}^{n} b_j^0 \binom{n}{j} \cdot t^j \cdot (1-t)^{n-j} \tag{2.4}$$

Durch Ersetzung der $(1-t)^{n-j}$ mit Hilfe des binomischen Satzes ergibt sich:

$$\begin{aligned} &= \sum_{j=0}^{n} b_j^0 \binom{n}{j} \cdot t^j \cdot \sum_{l=0}^{n-j} \binom{n-j}{l} \cdot 1^{n-j-l} \cdot (-t)^l \\ &= \sum_{j=0}^{n} \sum_{l=0}^{n-j} b_j^0 \cdot t^{j+l} \cdot (-1)^l \binom{n}{j} \cdot \binom{n-j}{l} \end{aligned}$$

Eine Umordnung nach $i = j + l$ liefert:

$$\begin{aligned} &= \sum_{i=0}^{n} t^i \underbrace{\sum_{j=0}^{n} b_j^0 \cdot (-1)^{i-j} \cdot \binom{n}{j} \cdot \binom{n-j}{i-j}}_{=:\, a_i} \\ &= \sum_{i=0}^{n} t^i \cdot a_i \end{aligned} \tag{2.5}$$

Im zweiten Schritt wird für die vereinfachte Normkurve $c(t) = t + t^2 + \ldots + t^n$ ihr Blossom $c[t_1,\ldots,t_n]$ ermittelt, mit dessen Hilfe das Blossom von $b(t)$ hergeleitet werden kann. Dazu sind die elementarsymmetrischen Funktionen nötig:

Definition 2 (elementarsymmetrische Funktionen). *Die Funktionen*

$$\begin{aligned} \sigma_1^n &= \Sigma_{1\le i\le n} && t_i \\ \sigma_2^n &= \Sigma_{1\le i<j\le n} && t_i \cdot t_j \\ \sigma_3^n &= \Sigma_{1\le i<j<k\le n} && t_i \cdot t_j \cdot t_k \\ &\vdots \\ \sigma_n^n &= \Pi_{1\le i\le n} && t_i \end{aligned}$$

heißen elementarsymmetrische Funktionen von $t_1,\ldots,t_n \in \mathbb{R}$. Alle Glieder von σ_l^n mit $l = 1,\ldots,n$ haben denselben Grad und sind somit homogen vom Grad l.

Satz 1 (Hauptsatz elementarsymmetrische Funktionen). *Jedes symmetrische Polynom $f(t_1,\ldots,t_n)$ lässt sich als Polynom $\varphi(\sigma_1^n,\ldots,\sigma_n^n)$ schreiben.*

Der Beweis des Satzes sowie weiterführendes zum Thema elementarsymmetrische Funktionen ist unter [vdW93] zu finden. Aus dem Hauptsatz der elementarsymmetrische Funktionen folgt, dass sich die Normkurve als Polynom über $\sigma_1^n,\ldots,\sigma_n^n$ schreiben lässt:

$$\begin{aligned} c(t) &= t + t^2 + \ldots + t^n = f(x_1,\ldots,x_n) \quad \text{mit } x_i = t^i,\ i = 1,\ldots,n \\ &= \alpha_1 \cdot \sigma_1^n + \ldots + \alpha_n \cdot \sigma_n^n \end{aligned} \tag{2.6}$$

Gesucht sind die α_i zur Bildung des Polynoms über $\sigma_1^n,\ldots,\sigma_n^n$. Aus der Definition der elementarsymmetrischen Funktionen und aus $t = t_1 = \ldots = t_n$ folgt $\sigma_i^n = \binom{n}{i}\cdot t^i$. Komponentenweise betrachtet gilt $t^i = \alpha_i \cdot \binom{n}{i}\cdot t^i$ und somit $\alpha_i = 1/\binom{n}{i}$. Eingesetzt in Formel (2.6) ergibt sich:

$$c(t) = \sigma_1^n/\binom{n}{1} + \ldots + \sigma_n^n/\binom{n}{n}$$

Damit ist $c[t_1,\ldots,t_n]$ gegeben durch:

$$c[t_1,\ldots,t_n] = \sigma_1^n/\binom{n}{1} + \ldots + \sigma_n^n/\binom{n}{n} \tag{2.7}$$

Da die elementarsymmetrischen Funktionen die Diagonaleigenschaft besitzen sowie multiaffin und symmetrisch sind, gelten diese Eigenschaften auch für $c[t_1,\ldots,t_n]$. Das Blossom der Normkurve $c(t) = t+t^2+\ldots+t^n$ ist folglich $c[t_1,\ldots,t_n]$ (Formel 2.7). Für ein Polynom $b(t) = a_0+a_1\cdot t+a_2\cdot t^2+\ldots+a_n\cdot t^n$ ergibt sich dann:

$$b[t_1,\ldots,t_n] = a_0 + a_1\cdot\sigma_1^n/\binom{n}{1} + \ldots + a_n\cdot\sigma_n^n/\binom{n}{n} \tag{2.8}$$

Ebenso wie bei der Normkurve besitzt auch $b[t_1,\ldots,t_n]$ die geforderten Eigenschaften des Blossoms aus Definition 1. $b[t_1,\ldots,t_n]$ aus (2.8) ist somit das Blossom von $b(t) = a_0 + a_1\cdot t + a_2\cdot t^2 + \ldots + a_n\cdot t^n$. Werden für die a_i die Koeffizienten der Bézier-Kurve gesetzt mit $a_i = \sum_{j=0}^n b_j^0\cdot(-1)^{i-j}\cdot\binom{n}{j}\cdot\binom{n-j}{i-j}$, $i = 0,\ldots,n$ aus (2.5), so ist $b[t_1,\ldots,t_n]$ (2.8) das Blossom der Bézier-Kurve $b(t)$.

In den meisten Fällen werden kubische Kurven bzw. bikubische Flächen zur Modellierung verwendet, ebenfalls basieren die in den folgenden Kapiteln vorgestellten Unterteilungsflächen auf bikubischen Flächen. Daher wird das Blossom einer kubischen Bézier-Kurve detailliert angegeben. Eine Umformung der kubischen Bézier-Kurve liefert die Koeffizienten $a_0,\ldots,a_3$:

$$\begin{aligned} b(t) &= \sum_{j=0}^{n} b_j^0\cdot B_j^n(t) \\ &= \underbrace{b_0^0}_{=:a_0} + \underbrace{(-3b_0^0+3b_1^0)}_{=:a_1}\cdot t + \underbrace{(3b_0^0-6b_1^0+3b_2^0)}_{=:a_2}\cdot t^2 + \underbrace{(-b_0^0+3b_1^0-3b_2^0+b_3^0)}_{=:a_3}\cdot t^3 \\ &= a_0 + a_1\cdot t + a_2\cdot t^2 + a_3\cdot t^3 \end{aligned}$$

Eingesetzt in Formel (2.8) ergibt sich das Blossom:

$$\begin{aligned} b[t_1,t_2,t_3] &= a_0 + (a_1,a_2,a_3)\cdot \begin{pmatrix} 1/\binom{3}{1} & & 0 \\ & 1/\binom{3}{2} & \\ 0 & & 1/\binom{3}{3} \end{pmatrix} \cdot \begin{pmatrix} \sigma_1^n \\ \sigma_2^n \\ \sigma_3^n \end{pmatrix} \\ &= b_0^0 + (-3b_0^0+3b_1^0, 3b_0^0-6b_1^0+3b_2^0, -b_0^0+3b_1^0-3b_2^0+b_3^0)\cdot \begin{pmatrix} 1/3 & & 0 \\ & 1/3 & \\ 0 & & 1 \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} t_1+t_2+t_3 \\ t_1\cdot t_2 + t_1\cdot t_3 + t_2\cdot t_3 \\ t_1\cdot t_2\cdot t_3 \end{pmatrix} \end{aligned}$$

Eine Verallgemeinerung des Definitionsbereich der Bézier-Kurve von $[0,1]$ auf $[u_0,u_1]$ mit $u_0 < u_1$, $u_0,u_1 \in \mathbb{R}$ liefert schließlich für $b(t) = \sum_{j=0}^{3} b_j^0 \cdot B_j^n(\frac{t-u_0}{u_1-u_0})$ das zugehörige Blossom:

$$b[t_1,t_2,t_3] = b_0^0 + (-3b_0^0+3b_1^0, 3b_0^0-6b_1^0+3b_2^0, -b_0^0+3b_1^0-3b_2^0+b_3^0) \cdot \begin{pmatrix} 1/3 & & 0 \\ & 1/3 & \\ 0 & & 1 \end{pmatrix} \cdot \begin{pmatrix} \frac{t_1-u_0}{u_1-u_0} + \frac{t_2-u_0}{u_1-u_0} + \frac{t_3-u_0}{u_1-u_0} \\ \frac{t_1-u_0}{u_1-u_0} \cdot \frac{t_2-u_0}{u_1-u_0} + \frac{t_1-u_0}{u_1-u_0} \cdot \frac{t_3-u_0}{u_1-u_0} + \frac{t_2-u_0}{u_1-u_0} \cdot \frac{t_3-u_0}{u_1-u_0} \\ \frac{t_1-u_0}{u_1-u_0} \cdot \frac{t_2-u_0}{u_1-u_0} \cdot \frac{t_3-u_0}{u_1-u_0} \end{pmatrix}$$

Blossom Hauptsatz und seine Anwendungen

Die Verbindung zwischen den symmetrischen Polynomen und der Bézier-Repräsentation wird im folgenden Hauptsatz deutlich. Der Beweis des Satzes sowie weitere Erläuterungen zu diesem Thema sind in [PBP02] zu finden.

Satz 2 (Hauptsatz). *Für jede polynomische Kurve $b(t)$ existiert genau ein n wertiges, multiaffines und symmetrisches Polynom $b[t_1,\ldots,t_n]$ mit der Diagonalen $b[t,\ldots,t] = b(t)$. Die Punkte*

$$b_i^0 = b[u_0^{\langle n-i\rangle}, u_1^{\langle i\rangle}], \quad i = 0,\ldots,n$$

sind die Bézier-Kontrollpunkte von $b(t)$ über $[u_0,u_1]$.

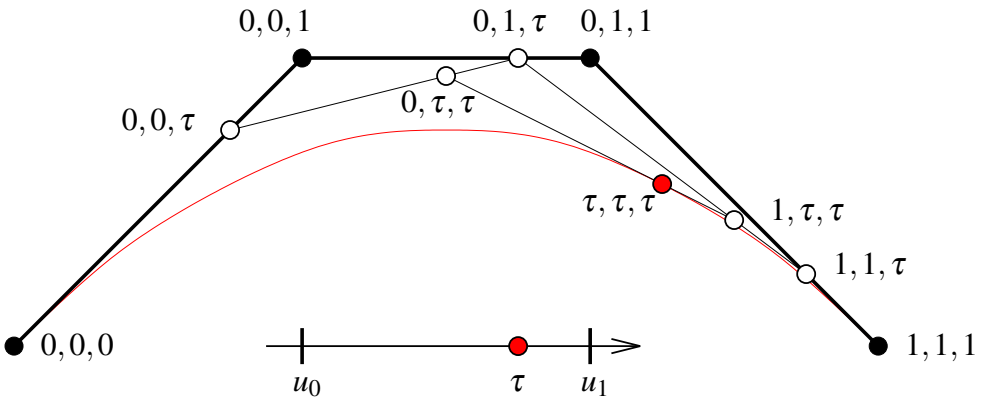


Abbildung 2.3: Bézier-Kurve $b(t)$ (rot markiert) über $[u_0,u_1]$ und Blossom Notation.

Aus diesem Satz resultiert (siehe Abbildung 2.3): $b(t)$ ist die Bézier-Kurve der Kontrollpunkte $b[u_0,u_0,u_0]$, $b[u_0,u_0,u_1]$, $b[u_0,u_1,u_1]$ und $b[u_1,u_1,u_1]$ über $[u_0,u_1]$. $b[t_1,t_2,t_3]$ ist das zugehörige Blossom von $b(t)$. Zur Notationsvereinfachung wird statt $b[u_0,u_0,u_1]$ die Kurzform $0,0,1$ verwendet (siehe Abbildung 2.3).

Zur Veranschaulichung der Unterteilungseigenschaften dient das Blossom $b[t_1,t_2,t_3]$ von $b(t)$ über $[u_0,u_4]$ aus Abbildung 2.4. Es sei u_2 ein beliebiger Punkt aus $[u_0,u_4]$, der das Intervall $[u_0,u_4]$ in die Abschnitte $[u_0,u_2]$ und $[u_2,u_4]$ unterteilt.

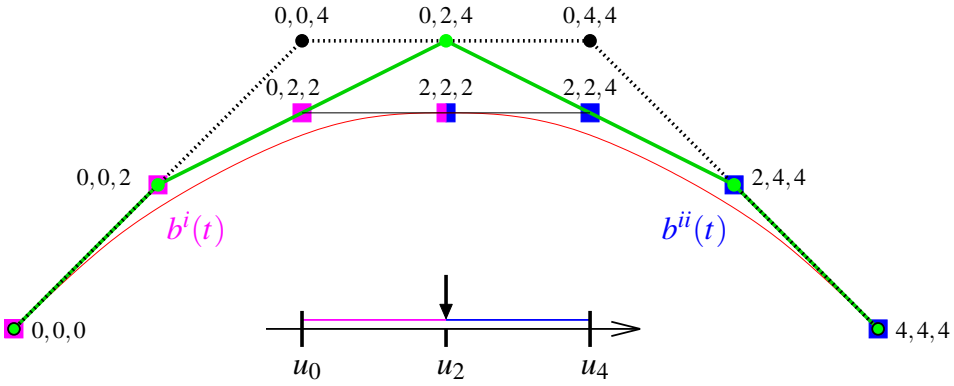


Abbildung 2.4: Blossom Notation: Bézier-Kurve $b^i(t)$ über $[u_0,u_2]$ und Bézier-Kurve $b^{ii}(t)$ über $[u_2,u_4]$ mit ihren jeweiligen magenta bzw. blau gefärbten Kontrollpunkten. Die grün markierten Punkte sind die Kontrollpunkte der korrespondierenden B-Spline-Kurve.

Gesucht sind die Kontrollpunkte der Bézier-Kurve über $[u_0,u_2]$ und $[u_2,u_4]$ mit:

$$b(t) = \begin{cases} b^i(t) & ,t \in [u_0,u_2] \\ b^{ii}(t) & ,t \in [u_2,u_4] \end{cases}$$

Aus der Anwendung des Hauptsatzes resultieren die Kontrollpunkte für $b^i(t)$, $t \in [u_0,u_2]$ mit $b[u_0,u_0,u_0]$, $b[u_0,u_0,u_2]$, $b[u_0,u_2,u_2]$, $b[u_2,u_2,u_2]$ sowie die Kontrollpunkte für $b^{ii}(t)$, $t \in [u_2,u_4]$ mit $b[u_2,u_2,u_2]$, $b[u_2,u_2,u_4]$, $b[u_2,u_4,u_4]$, $b[u_4,u_4,u_4]$. Für $u_2 \in]u_0,u_4[$ sind die beiden Bézier-Kurven an der zusammengesetzten Stelle $b[u_2,u_2,u_2] = b^i(u_2) = b^{ii}(u_2)$ wie auch $\forall t \in [u_0,u_4]$ C^2-stetig. Mit Hilfe von Bézier-Kurven vom Grad n, welche $n-1$ stetig zusammengefügt werden, erhält man den nächsten übergeordneten Kurventyp, die B-Spline-Kurven. Blossoms liefern auf einfache Art und Weise die zugehörigen Kontrollpunkte. In dem Beispiel aus Abbildung 2.4 sind dies die Punkte $b[u_0,u_0,u_0]$, $b[u_0,u_0,u_2]$, $b[u_0,u_2,u_4]$, $b[u_2,u_4,u_4]$, $b[u_4,u_4,u_4]$ über dem Knotenvektor $\{u_0,u_0,u_0,u_2,u_4,u_4,u_4\}$. Werden die Intervalle $[u_0,u_2]$ und $[u_2,u_4]$ weiter in $u_1 \in [u_0,u_2]$ und $u_3 \in [u_2,u_4]$ unterteilt (siehe Abbildung 2.5), so entsteht der Knotenvektor $\{u_0,u_0,u_0,u_1,u_2,u_3,u_4,u_4,u_4\}$ mit den dedizierten B-Spline-Kontrollpunkten $b[u_0,u_0,u_0]$, $b[u_0,u_0,u_1]$, $b[u_0,u_1,u_2]$, $b[u_1,u_2,u_3]$, $b[u_2,u_3,u_4]$, $b[u_3,u_4,u_4]$, $b[u_4,u_4,u_4]$. Blossoms ermöglichen demzufolge in einfacher Form das Einfügen von Knoten. Auf diesem Einfügen von Knoten basieren die Unterteilungsregeln der Unterteilungsflächen, welche in den nachfolgenden Kapiteln ausführlich behandelt werden.

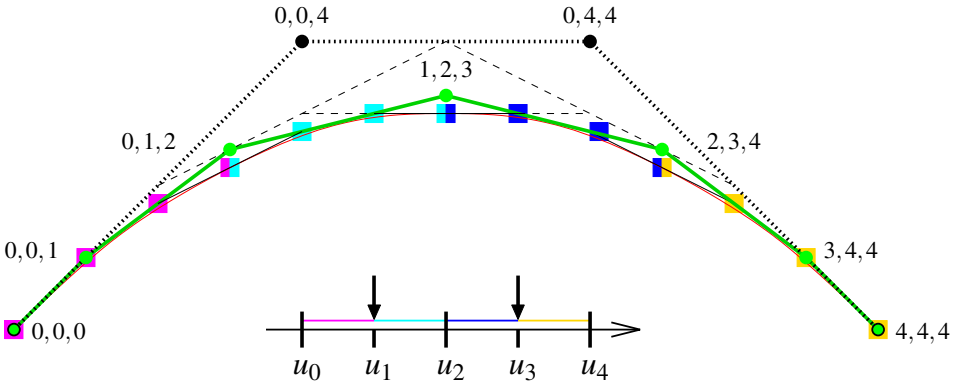


Abbildung 2.5: Blossom Notation: Unterteilen des Kontrollpunktepolygons durch Knoteneinfügen.

NURBS

Eine Erweiterung der B-Spline-Kurve zur NURBS-Kurve ist durch eine Homogenisierung möglich. Abbildung 2.6 zeigt dazu ein Beispiel: Die Kontrollpunkte aus dem $\mathbb{R}^2$ werden durch ihre Gewichte Teil des $\mathbb{R}^3$. Die B-Spline-Kurve wird im $\mathbb{R}^3$ aus den homogenen Kontrollpunkten gebildet. Durch eine Projektion der B-Spline-Kurve im $\mathbb{R}^3$ in die Hyperebene $w = 1$, der so genannten Inhomogenisierung, resultiert die rationale B-Spline-Kurve im $\mathbb{R}^2$.

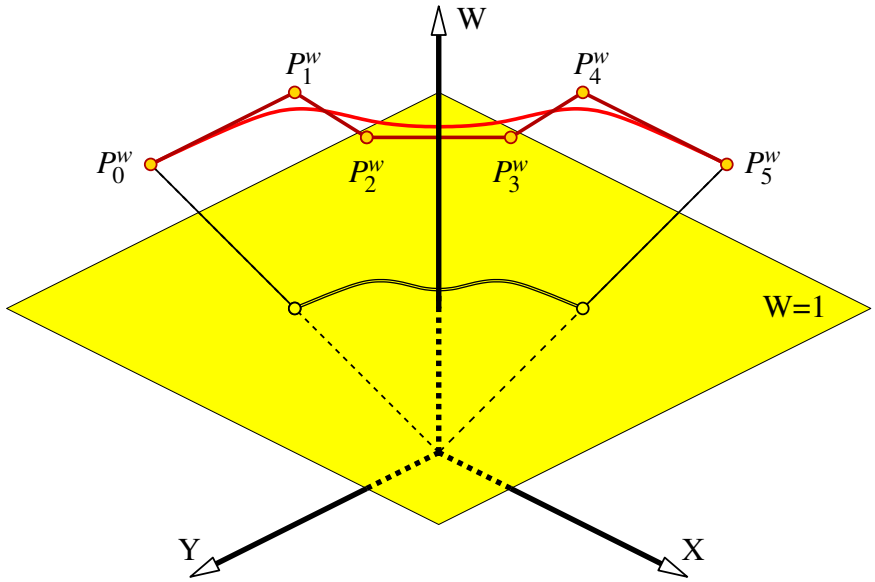


Abbildung 2.6: Geometrische Konstruktion einer rationalen B-Spline-Kurve

Blossom Fläche

Die bisherige Vorgehensweise für Kurven lässt sich auch auf Flächen übertragen, indem lediglich der Kurven-Parameter $t \in \mathbb{R}$ durch einen Flächen-Parameter $t = (u, v) \in \mathbb{R}^2$ ersetzt wird.

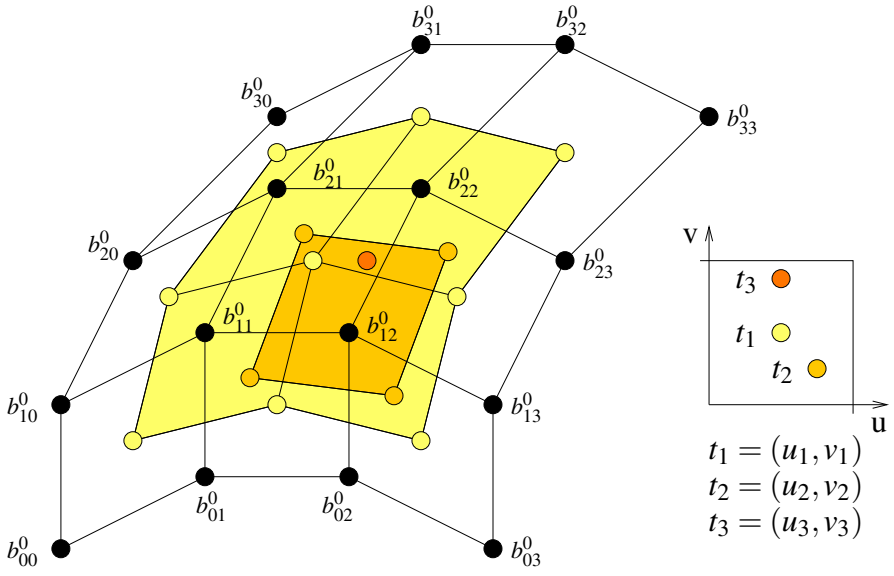


Abbildung 2.7: Verallgemeinerter de Casteljau Algorithmus für Bézier-Flächen.

Damit ergibt sich für ein bikubisches Blossom folgende schematische Darstellung (siehe Abbildung 2.7):

$$
\begin{array}{cccccccccccc}
b^0_{30} & b^0_{31} & b^0_{32} & b^0_{33} & & & & & & & & \\
b^0_{20} & b^0_{21} & b^0_{22} & b^0_{23} & \xrightarrow{t_1} & b^1_{20}[t_1] & b^1_{21}[t_1] & b^1_{22}[t_1] & & & & \\
b^0_{10} & b^0_{11} & b^0_{12} & b^0_{13} & & b^1_{10}[t_1] & b^1_{11}[t_1] & b^1_{12}[t_1] & \xrightarrow{t_2} & b^2_{10}[t_1,t_2] & b^2_{11}[t_1,t_2] & \\
b^0_{00} & b^0_{01} & b^0_{02} & b^0_{03} & & b^1_{00}[t_1] & b^1_{01}[t_1] & b^1_{02}[t_1] & & b^2_{00}[t_1,t_2] & b^2_{01}[t_1,t_2] & \xrightarrow{t_3} \quad b^3_{00}[t_1,t_2,t_3]
\end{array}
$$

2.2 NURBS-Kurven und Flächen

Ergänzend zu der anschaulichen Vorstellung der NURBS mit Hilfe der Blossoms, folgt in diesem Abschnitt ihre formale Einführung über B-Spline-Basisfunktionen [PT95]. Wichtige Eigenschaften der NURBS-Kurven und Flächen werden ebenfalls erläutert.

Definition 3 (B-Spline-Basis-Funktionen). *Die B-Spline-Basis-Funktionen N_i^p vom Grad p sind definiert durch*

$$N_i^0(u) = \begin{cases} 1 & ,u_i \le u < u_{i+1} \\ 0 & ,sonst \end{cases}$$

$$N_i^p(u) = \frac{u-u_i}{u_{i+p}-u_i} N_i^{p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} N_{i+1}^{p-1}(u)$$

mit $U = \{u_0, \ldots, u_m\}$, $u_i \le u_{i+1}$ ist der zugehörige Knotenvektor und $i = 0, \ldots, m-1$.

Definition 4 (NURBS-Kurve). *Es seien $\boldsymbol{P}_i$ die Kontrollpunkte der NURBS-Kurve und w_i die Gewichte der Kontrollpunkte P_i. Weiterhin seien $N_i^p(u)$ die B-Spline-Basisfunktionen vom Grad p aus Definition 3 über dem nicht-uniformen Knotenvektor*

$$U = \{u_0, \ldots, u_p, u_{p+1}, \ldots, u_{m-p-1}, u_{m-p}, \ldots, u_m\}$$

Eine NURBS-Kurve vom Grad p ist definiert durch

$$\boldsymbol{C}(u) = \frac{\sum_{i=0}^n N_i^p(u) w_i \boldsymbol{P}_i}{\sum_{i=0}^n N_i^p(u) w_i} \qquad u_p \le u \le u_{m-p}$$

mit $m = n+p+1$. Gilt $u_0 = \ldots = u_p$ sowie $u_{m-p} = \ldots = u_m$, so liegt eine clamped NURBS-Kurve vor. Der Anfangs- und Endkontrollpunkt wird interpoliert. Ist diese Bedingung nicht erfüllt, so ist die NURBS-Kurve unclamped.

Definition 5 (NURBS-Fläche). *Es seien $\boldsymbol{P}_{ij}$ die Kontrollpunkte eines gegebenen $n \times m$ Kontrollnetzes und $w_{i,j}$ die Gewichte der Kontrollpunkte P_{ij}. Weiterhin seien $N_i^p(u)$ und $N_j^q(v)$ die B-Spline-Basisfunktionen über den nicht-uniformen Knotenvektoren*

$$U = \{u_0, \ldots, u_p, u_{p+1}, \ldots, u_{r-p-1}, u_{r-p}, \ldots, u_r\}$$

$$V = \{v_0, \ldots, v_q, v_{q+1}, \ldots, v_{s-q-1}, u_{s-q}, \ldots, v_s\}$$

mit $r = n+p+1$ und $s = m+q+1$.
Eine NURBS-Fläche vom Grad p in u-Richtung und Grad q in v-Richtung ist definiert durch

$$\boldsymbol{S}(u,v) = \frac{\sum_{i=0}^n \sum_{j=0}^m N_i^p(u) N_j^q(v) w_{ij} \boldsymbol{P}_{ij}}{\sum_{i=0}^n \sum_{j=0}^m N_i^p(u) N_j^q(v) w_{ij}}$$

Eigenschaften:

Die wichtigsten und für diese Arbeit essentiellen Eigenschaften der NURBS-Kurven und Flächen sind im folgenden aufgeführt, weitere sind u.a. in [PT95] beschrieben:

- **Affine Invarianz:** Eine affine Transformation der Kurve bzw. Fläche kann durch eine affine Transformation der Kontrollpunkte ausgeführt werden. NURBS-Kurven und Flächen sind weiterhin invariant unter perspektivischen Projektionen.

- **Konvexe Hülle:** Für $u \in [u_i, u_{i+1}]$ liegt die NURBS-Kurve $C(u)$ in der konvexen Hülle der Punkte $P_{i-p}, \ldots, P_i$. Wenn $(u,v) \in [u_{i_0}, u_{i_0+1}] \times [v_{j_0}, v_{j_0+1}]$ gilt, dann befindet sich die NURBS-Fläche $S(u,v)$ innerhalb der konvexen Hülle der Punkte P_{ij}, $i_0 - p \leq i \leq i_0$, $j_0 - q \leq j \leq j_0$.

- **Stetigkeit:** Eine NURBS-Kurve vom Grad p und einem Knotenvektor mit Knoten der Vielfachheit 1 ist C^{p-1}-stetig. Liegt ein Knoten u_j, $j \in \{p+1, \ldots, m-p-1\}$ mit Vielfachheit $k > 1$ vor, so ist die NURBS-Kurve an der Stelle u_j C^{p-k}-stetig. Diese Eigenschaft lässt sich direkt auf NURBS-Flächen in u und v Richtung übertragen.

- **Variationsmindernde Eigenschaft:** Keine Ebene in 3D respektive Gerade in 2D hat mehr Schnittpunkte mit der NURBS-Kurve als mit dem Kontroll-Polygon. Für Flächen ist bislang keine korrespondierende Eigenschaft bekannt.

- **Reduktion zu Bézier-Kurven:** Eine NURBS-Kurve bzw. Fläche ohne innere Knoten, d.h. $n = p$ bzw. $n = p$ und $m = q$, ist eine Bézier-Kurve respektive Bézier-Fläche. Weiterhin kann eine NURBS-Kurve bzw. Fläche aus rationalen Bézier-Segmenten zusammengesetzt werden.

- **Lokalität:** Anders als bei den Bézier-Kurven und Flächen wirkt sich das Verändern eines Kontrollpunktes P_i bzw. P_{ij} nur auf das Kurvenstück $[u_i, u_{i+p+1}]$ respektive Flächenstück $[u_i, u_{i+p+1}] \times [v_j, v_{j+q+1}]$ aus.

2.3 Unterteilungsflächen

Zur Generierung von Freiformflächenmodelle werden zur Zeit hauptsächlich NURBS-Flächen [PT95] sowie Unterteilungsflächen [ZS99] verwendet. Der Einsatzschwerpunkt von NURBS-Flächen liegt im Industriedesign: Die Genauigkeit bei der Flächendarstellung, die sich durch die Parameterauswertung ergibt, liefert entscheidende Vorteile für diesen Anwendungsbereich. Unterteilungsflächen hingegen werden durch eine sukzessive Verfeinerung des Kontrollpunktnetzes gebildet. Die beliebige Topologie und die Möglichkeit, mit Special Features Akzente im Modell zu setzen, erlaubt es mit Unterteilungsflächen auf einfache Art und Weise komplexe Modelle zu erzeugen. Solche komplexen Modelle lassen sich mit NURBS nur durch das Aneinanderfügen vieler ausgeschnittener Teilstücke generieren, da lediglich eine feste Topologie in Form eines $n \times m$ Gitters zur Verfügung steht.

Catmull und Clark erweiterten 1978 in [CC78] die uniformen bikubischen B-Spline-Flächen um beliebige Valenzen und schufen damit einen neuen Oberflächentyp. Mit dieser Arbeit eröffneten sie einen neuen Forschungszweig, der einer Vielzahl von Weiterentwicklungen folgten (siehe z.B. [ZS99]). Die meisten Unterteilungs-Schemata lassen sich mit den folgenden Kriterien [ZS99] klassifizieren:

- Art der Verfeinerungsregeln (Vertex einfügen oder Corner-Cutting).

- Art des generierten Kontrollnetzes (Dreiecke oder Vierecke).

- Approximierendes oder interpolierendes Verfahren.

Die bislang weiteste Verbreitung fanden die Unterteilungsschemata von Catmull-Clark sowie von Loop. Beide Verfahren sind *approximierend* und gehören zur Klasse *Vertex einfügen*. Das Catmull-Clark-Verfahren liefert *Vierecke* beim Unterteilen und geht von bikubischen B-Splines aus, das Schema von Loop erzeugt *Dreiecke* und basiert auf Box-Splines.

Im Abschnitt 2.3.2 werden zunächst die Unterteilungsregeln für kubische NURBS hergeleitet. Darauf aufbauend werden in den Abschnitten 2.3.3 und 2.3.4 die Verfahren von Catmull-Clark und Loop vorgestellt. Dabei werden lediglich die Hauptregeln von Catmull-Clark und Loop kurz zusammengefasst, ein Überblick über die diversen Modifikationen der Regeln ist u.a. in [ZS99] zu finden. In den nächsten Abschnitten werden einige Begriffe benötigt, die im Folgenden definiert und erläutert werden (siehe auch [ZS99]).

2.3.1 Notation und Terminologie

Eine Unterteilungsfläche ist definiert durch ein *Kontrollnetz* M^0. Eine Anwendung der Unterteilungsregeln auf das Kontrollnetz M^0 liefert das verfeinerte Netz M^1. Die Folge der Kontrollnetze konvergiert gegen die *Limitfläche* $L(M^0)$. Die Punkte eines Kontrollnetzes M^l in Unterteilungstiefe l werden *Kontrollpunkte* P_i^l genannt. Die *Valenz eines Kontrollpunktes* P_i^l ist die Anzahl seiner inzidenten Kanten, die *Valenz eines Faces* ist die Anzahl seiner Eckpunkte. Die Anordnung der Kontrollpunkte in einem $n \times m$ Gitter wird als regulär bezeichnet. Eine Übertragung auf die beliebige Topologie der Unterteilungsflächen liefert: Besitzt ein Kontrollpunkt im Inneren eines Viereckснetztes Valenz vier bzw. hat ein Kontrollpunkt eines Dreiecksnetzes Valenz sechs, so liegt ein *regulärer* Kontrollpunkt vor. Ein Kontrollpunkt auf dem Rand des Viereckснetzes ist mit den Valenzen zwei und drei regulär. Bei einem Kontrollpunkt auf dem Rand eines Dreiecksnetzes liegen mit den Valenzen zwei, drei und vier reguläre Punkte vor. In den übrigen Fällen handelt es sich um *irreguläre* Kontrollpunkte, auch außerordentliche Punkte genannt.

Die *1-Nachbarschaft* $N^l(P_i^l)$ *eines Kontrollpunktes* P_i^l besteht aus den Kontrollpunkten der Faces, die P_i^l enthalten. Die *1-Nachbarschaft* $N^l(F^l)$ *eines Faces* F^l beinhaltet alle zu den Eckpunkten von F^l adjazenten Faces. Zur *Kantennachbarschaft* gehören die Kontrollpunkte zweier Faces, die eine gemeinsame Kante teilen.

Bei einem Unterteilungsschritt werden drei Arten von neuen Kontrollpunkten erzeugt: *Face-*, *Kanten-* und *Vertexpunkte*. Zur Berechnung der neuen Punkte werden unterschiedliche Ausschnitte aus dem Kontrollnetz benötigt. Diese Ausschnitte werden in schematischer Darstellung als *Masken* bezeichnet. Die Masken enthalten die Koeffizienten der Konvexkombinationen, die als Regelsatz dienen. Bei den vorgestellten Unterteilungsverfahren treten für die drei Arten von neuen Kontrollpunkten folgende Abhängigkeiten auf:

- Ein neuer Vertexpunkt P_i^{l+1} wird aus der 1-Nachbarschaft von P_i^l gebildet.
- Ein neuer Kantenpunkt wird aus seinen Kantennachbarn bestimmt.
- Ein neuer Facepunkt wird über die Eckpunkte seines Faces berechnet.

Diese Unterteilungsregeln werden auch als *glatte* Regeln (smooth rules) bezeichnet. Loop-Flächen kommen ohne Facepunkte aus, ESubs- und Catmull-Clark-Flächen generieren neue Vertex-, Face- und Kantenpunkte.

Durch die Anwendung alternativer Unterteilungsregeln können *Special Features* wie z.B. Spitzen, scharfe oder halbscharfe Kanten erzeugt werden. Dazu werden die ausgewiesenen Masken der glatten Regeln nicht vollständig genutzt. Nur ein entsprechend markierter Teilbereich wird für den Regelsatz herangezogen: Z.B. wird zur Berechnung eines neuen Vertexpunktes auf einer scharfen Kante nur seine inzidenten scharfen Kanten berücksichtigt statt seiner kompletten 1-Nachbarschaft. Auf diese Weise erhält man eine lokale Reduktion der Stetigkeit und die Möglichkeit, auffallende Akzente in ein Modell zu setzen.

Die neu berechneten Kontrollpunkte werden ihrem jeweiligen *Schema* entsprechend zu einem Netz verbunden. Bei Loop wird ein Dreiecksnetz generiert, bei den übrigen Verfahren, die im folgenden vorgestellt werden, wird immer ein Viereckснetz gebildet.

Der *Limitpunkt* von P_i^l ist durch $L_i = \lim_{l\to\infty} P_i^l$ definiert. Mit $\vec{N}_i$ wird die *Normale* von $L(M^0)$ an der Stelle L_i bezeichnet. Zur Berechnung des Limitpunktes wird die gleiche Maske wie zur Berechnung eines Vertexpunktes benutzt. Die Normale des Limitpunktes wird über das Kreuzprodukt der zwei zugehörigen Tangenten bestimmt. Die Tangenten werden ebenfalls mit der Vertex-Maske berechnet.

Zur Vereinfachung der Notation wird die Unterteilungstiefe l weg gelassen, wenn sie aus dem Zusammenhang klar ist (z.B. P_i statt P_i^l).

2.3.2 NURBS-Unterteilungsregeln

In diesem Abschnitt werden die Unterteilungs- und Limitregeln für kubische NURBS-Kurven und bikubische NURBS-Flächen hergeleitet, wobei sich die Rationalität aus der Homogenisierung ergibt (siehe Abbildung 2.6). Die Unterteilungsregeln von NURBS lassen sich auf die Unterteilungsregeln von uniformen B-Spline-Flächen reduzieren. Auf deren Grundlage wurden die Catmull-Clark-Flächen entwickelt, die Thema des nachfolgenden Kapitels sind. Für die Entwicklung der erweiterten Unterteilungsflächen (ESubs) dienen die NURBS-Unterteilungsregeln ebenfalls als Ausgangsbasis. Der Herleitungsweg für die Unterteilungsregeln für NURBS ist so gewählt worden, dass neben den eigentlichen NURBS-Regeln auch die ESubs- sowie regulären Catmull-Clark-Regeln auf einfache Weise abgeleitet werden können. Die Notation in den folgenden Abschnitten ist angelehnt an [SSS98] und [SZBN03].

2.3.2.1 Kurve

Im ersten Schritt wird die Verfeinerung einer kubischen, clamped B-Spline-Kurve betrachtet. Die Kurve ist durch die Kontrollpunkte P_i, $i = 0,\ldots,n$ und den Knotenvektor $U = \{u_{-2},\ldots,u_{n+2}\}$ definiert (siehe Abbildung 2.8). Der zugehörige Knotenintervallvektor ergibt sich aus den Einträgen $d_i = u_{i+1} - u_i$ als $K = \{d_{-2},\ldots,d_{n+1}\}$ mit $d_{-2} = d_{-1} = d_0 = 0$ und $d_{n-1} = d_n = d_{n+1} = 0$.

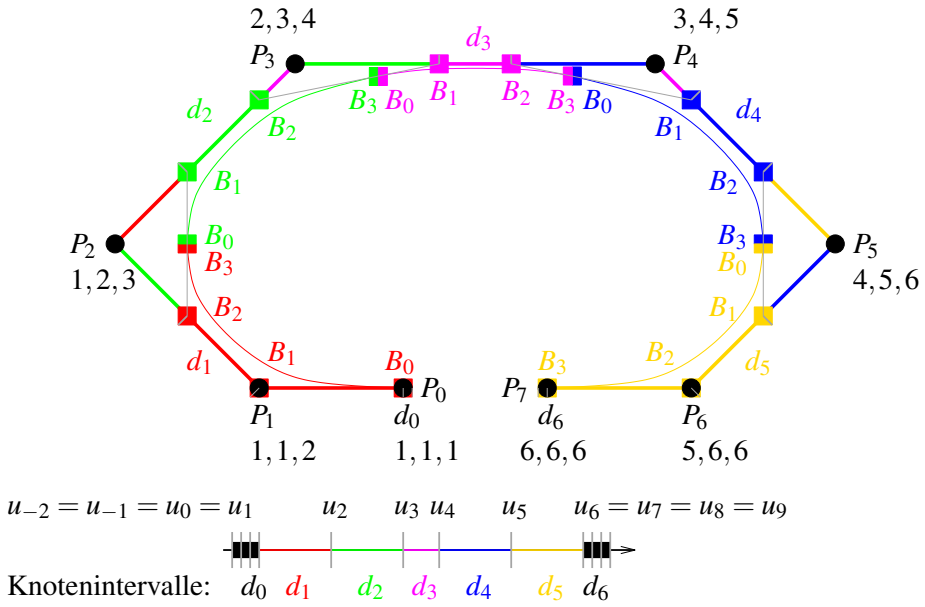


Abbildung 2.8: Kubische B-Spline-Kurve mit Knotenvektor. Die Bézier-Kurvensegmente mit ihren entsprechenden Kontrollpunkten sind farbig markiert.

Jede Kante des Kontrollpolygons bekommt ihr entsprechendes Knotenintervall zugewiesen (siehe Abbildung 2.8). Dieses Knotenintervall bestimmt den Definitionsbereich des zugehörigen Kurvensegmentes, welches in Bézier-Form durch die lokalen Bézier-Kontrollpunkte B_0, B_1, B_2, B_3 repräsentiert wird (siehe Abbildung 2.8). Aus diesen Knotenintervallen erhält jeder Vertex P_i einen lokalen Knotenintervallvektor

$$K_i = \{d_{i-2}, d_{i-1}, d_i, d_{i+1}\}$$

und einen lokalen Knotenvektor

$$\overline{K}^i = \{\overline{d}_{i-2}, \overline{d}_{i-1}, \overline{d}_i, \overline{d}_{i+1}, \overline{d}_{i+2}\}$$

mit $\overline{d}_k = \sum_{j=i-2}^{k-1} d_j$, $k = i-2, \ldots, i+2$ und $d_j = 0$ für $j \leq 0$ und $j \geq n-1$.

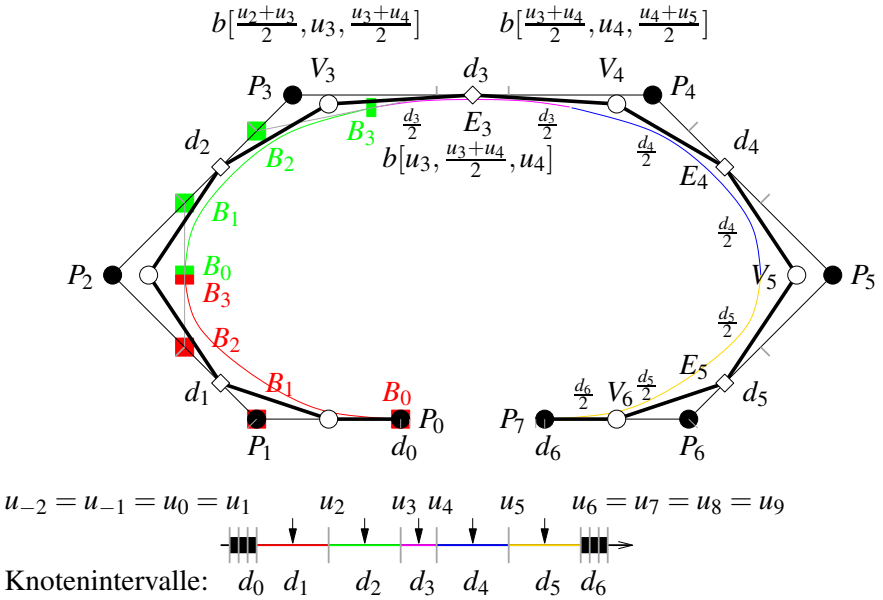


Abbildung 2.9: Verfeinerung einer kubischen B-Spline-Kurve durch Knoteneinfügen. Die Bézier-Kurvensegmente mit ihren zugehörigen Kontrollpunkten sind farbig markiert.

Eine Verfeinerung der nicht-uniformen B-Spline-Kurve durch mittiges Knoteneinfügen von $\{\ldots, u_i, u_{i+1}, \ldots\}$ zu $\{\ldots, u_i, \frac{u_i+u_{i+1}}{2}, u_{i+1}, \ldots\}$ mit Hilfe des zugehörigen Blossoms $b[t_1, t_2, t_3]$ liefert für jede Kante und jeden Vertex neue Kontrollpunkte (siehe Abbildung 2.9):

$$\begin{aligned} E_i &= b\left[u_i, \tfrac{u_i+u_{i+1}}{2}, u_{i+1}\right] \\ &= \frac{1}{2} \cdot (B_1 + B_2) \\ &= \frac{(d_{i+1} + d_i/2) \cdot P_i + (d_{i-1} + d_i/2) \cdot P_{i+1}}{d_{i-1} + d_i + d_{i+1}} \\ &=: \text{subdivCurveEdge}(P_i, P_{i+1}, d_{i-1}, d_i, d_{i+1}) \end{aligned} \tag{2.9}$$

$$\begin{aligned} V_i &= b\left[\tfrac{u_{i-1}+u_i}{2}, u_i, \tfrac{u_i+u_{i+1}}{2}\right] \\ &= \frac{1}{2} \cdot \left(\frac{d_i \cdot E_{i-1} + d_{i-1} \cdot E_i}{d_{i-1} + d_i} + P_i\right) \\ &=: \text{subdivCurveVertex}(P_{i-1}, P_i, P_{i+1}, d_{i-2}, \ldots, d_{i+1}) \end{aligned} \tag{2.10}$$

wobei B_1 und B_2 die lokalen Bézier-Kontrollpunkte von $P_{i-1}, P_i, P_{i+1}, P_{i+2}$ sind.

Werden diese Unterteilungsregeln unendlich oft für P_i und seine 1-Nachbarschaft angewendet, so erhält man den Limitpunkt von P_i. Den gleichen Punkt erhält man durch die Berechnung der lokalen NURBS-Kurve am Parameterwert $\overline{d}_i$ im lokalen Kurvensegment der Kontrollpunkte $P_{i-1}, P_i, P_{i+1}, P_{i+2}$ und dem lokalen Knotenvektor $\overline{K}^i$ von P_i.

Die Regeln zur Berechnung des Limitpunktes und des Tangentenvektors von P_i sind:

$$\begin{aligned} L_i &= b[u_i, u_i, u_i] \\ &= \frac{d_i}{d_{i-1}+d_i} \cdot \frac{d_i \cdot P_{i-1} + (d_{i-2}+d_{i-1}) \cdot P_i}{d_{i-2}+d_{i-1}+d_i} + \\ &\quad \frac{d_{i-1}}{d_{i-1}+d_i} \cdot \frac{(d_i+d_{i+1}) \cdot P_i + d_{i-1} \cdot P_{i+1}}{d_{i-1}+d_i+d_{i+1}} \\ &=: \text{limitCurveVertex}(P_{i-1}, P_i, P_{i+1}, d_{i-2}, \ldots, d_{i+1}) \end{aligned} \tag{2.11}$$

$$\begin{aligned} \vec{T}_i &= b[u_i, u_{i+1}, u_i] - b[u_{i-1}, u_i, u_i] \\ &= \frac{(d_i+d_{i+1}) \cdot P_i + d_{i-1} \cdot P_{i+1}}{d_{i-1}+d_i+d_{i+1}} - \frac{d_i \cdot P_{i-1} + (d_{i-2}+d_{i-1}) \cdot P_i}{d_{i-2}+d_{i-1}+d_i} \\ &=: \text{tangentCurveVertex}(P_{i-1}, P_i, P_{i+1}, d_{i-2}, \ldots, d_{i+1}) \end{aligned} \tag{2.12}$$

Aus diesen Unterteilungsregeln ergibt sich bei einer clamped Kurve $V_0 = P_0 = L_0, V_n = P_n = L_n$ und $E_0 = V_0, E_{n-1} = V_n$.

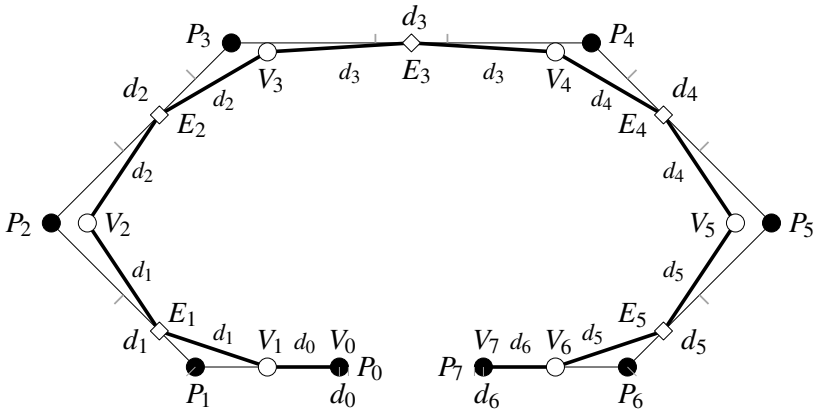


Abbildung 2.10: Die neuen Knotenintervalle werden nach dem Unterteilungsschritt den neuen Kanten zugewiesen. Zur Vereinfachung der Schreibweise wird bei der Zuweisung Faktor zwei verwendet, welches die Teilverhältnisse unverändert lässt.

2.3.2.2 Fläche

Die Unterteilungs- und Limitpunktregeln der Kurve können nun einfach auf eine Fläche (siehe Abschnitt 2.2) übertragen werden. Die Einträge der beiden dedizierten Knotenintervallvektoren der Fläche werden jeweils einer kompletten Reihe bzw. Spalte zugeordnet (siehe Abbildung 2.11).

Ebenso wie bei der NURBS-Kurve kann die NURBS-Fläche durch Bézier-Patches repräsentiert werden. Zur Herleitung der lokalen Bézier-Kontrollpunkte wird die Notation aus Abbildung 2.11 verwendet. Um die lokalen Bézier-Kontrollpunkte B und C des Faces 6 zu berechnen, sind die Kontrollpunkte $P_0, \ldots, P_7$ sowie der vertikale lokale Knotenintervallvektor $K_0^v = \{d_1, d_2, d_3, d_4\}$ und der horizontale lo-

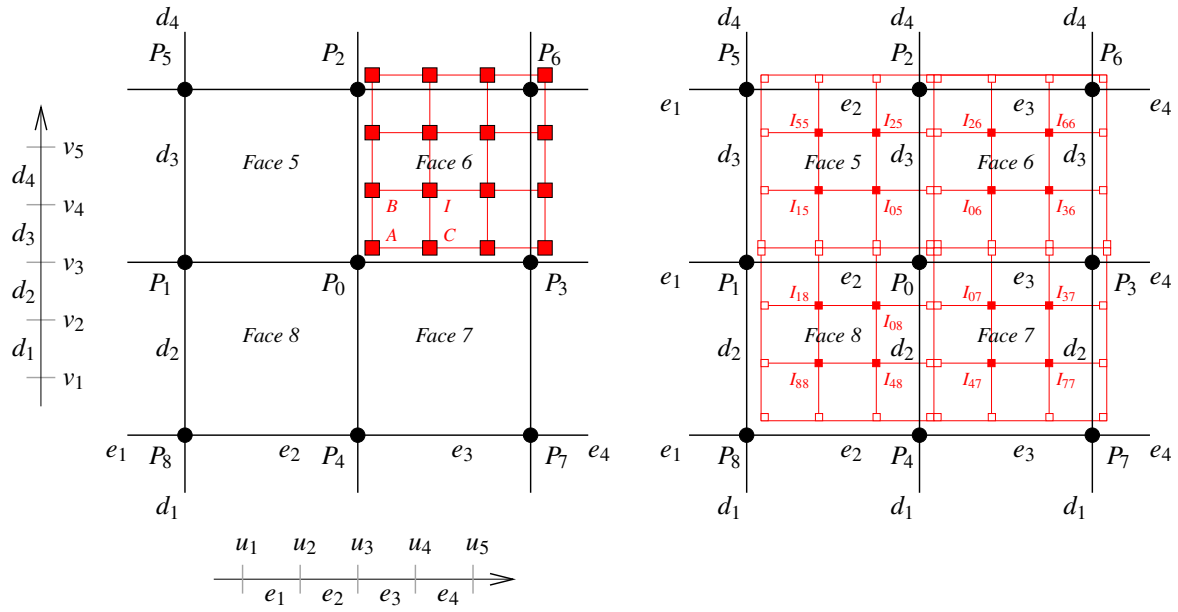


Abbildung 2.11: Teilstück eines Kontrollnetzes einer bikubischen NURBS-Fläche. Für Face 6 ist die Darstellung durch ein bikubisches Bézier-Patch angegeben.

kale Knotenintervallvektor $K_0^h = \{e_1, e_2, e_3, e_4\}$ des Punktes P_0 nötig.

$$\begin{aligned} B &= b[(u_3,v_3),(u_3,v_3),(u_3,v_4)] \\ &= ((d_3+d_4)\cdot \text{limitCurveVertex}(P_1,P_0,P_3,e_1,\ldots,e_4) \\ &\quad + d_2 \cdot \text{limitCurveVertex}(P_5,P_2,P_6,e_1,\ldots,e_4))/(d_2+d_3+d_4) \end{aligned} \tag{2.13}$$

$$\begin{aligned} C &= b[(u_3,v_3),(u_3,v_3),(u_4,v_3)] \\ &= ((e_3+e_4)\cdot \text{limitCurveVertex}(P_4,P_0,P_2,d_1,\ldots,d_4) \\ &\quad + e_2 \cdot \text{limitCurveVertex}(P_7,P_3,P_6,d_1,\ldots,d_4))/(e_2+e_3+e_4) \end{aligned} \tag{2.14}$$

Zur Bestimmung des lokalen Kontrollpunktes I reichen die Eckpunkte des Faces 6 und K_0^v, K_0^h aus:

$$\begin{aligned} I &= b[(u_3,v_3),(u_3,v_3),(u_4,v_4)] \\ &= \frac{(e_3+e_4)\cdot(d_3+d_4)\cdot P_0 + (e_3+e_4)\cdot d_2 \cdot P_2 + e_2 \cdot d_2 \cdot P_6 + e_2 \cdot (d_3+d_4)\cdot P_3)}{(e_4+e_3+e_2)\cdot(d_2+d_3+d_4)} \end{aligned} \tag{2.15}$$

Punkt I heißt *innerer* Bézier-Kontrollpunkt. Zur Vereinfachung der Notation über dem gesamten Kontrollnetz wird ein innerer Bézier-Punkt mit der Bezeichnung I_{ij} versehen, wobei j das zugehörige Face und i den entsprechenden Kontrollpunkt angibt, mit deren Hilfe I_{ij} aus den Punkten des Faces j und den lokalen Knotenintervallvektoren von Punkt P_i gebildet werden kann. Somit ist z.B. $I_{06} = I$ ein innerer Bézier-Punkt, welcher aus den Punkten des Faces 6 und den lokalen Knotenintervallvektoren des Kontrollpunktes P_0 berechnet wird.

Aus den inneren Bézier-Punkten rund um P_0 lässt sich der lokale Kontrollpunkt A berechnen:

$$\begin{aligned} A &= b[(u_3,v_3),(u_3,v_3),(u_3,v_3)] \\ &= \frac{e_3\cdot d_2\cdot I_{05} + e_2\cdot d_2\cdot I_{06} + e_2\cdot d_3\cdot I_{07} + e_3\cdot d_3\cdot I_{08}}{(e_2+e_3)\cdot(d_2+d_3)} \\ &=: \text{limitPoint}(P_0,\ldots,P_8,e_1,\ldots,e_4,d_1,\ldots,d_4) \end{aligned} \tag{2.16}$$

A ist der Limitpunkt zu P_0. Damit erhält man mit der Berechnungsvorschrift zu A ebenso die Formel zur Limitpunktbestimmung. Die restlichen lokalen Bézier-Kontrollpunkte lassen sich auf gleiche Art und Weise mit den entsprechenden lokalen Knotenintervallvektoren bestimmen.

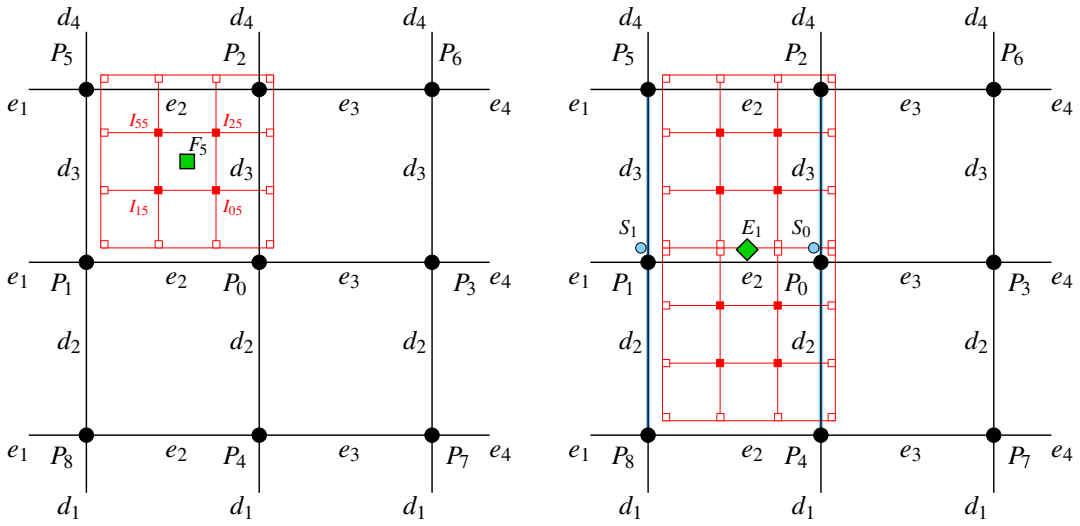


Abbildung 2.12: Unterteilung eines NURBS-Patches: Neuer Facepunkt (links) und neuer Kantenpunkt (rechts). Die Polygonzüge $\langle P_8, P_1, P_5\rangle$ und $\langle P_4, P_0, P_2\rangle$ sind blau unterlegt, sowie die Hilfspunkte S_1, S_0.

Mittiges Knoteneinfügen in beiden Knotenvektoren der Fläche liefert drei Arten von neuen Kontrollpunkten: Neue Facepunkte F_i , neue Vertexpunkte V_i und neue Kantenpunkte E_i (siehe Abbildung 2.12 und 2.13). Unter Anwendung der Notation aus den Abbildungen 2.11, 2.12 und 2.13 wird die Berechnung neuer Face-, Vertex- und Kantenpunkte anhand des Beispiels F_5, E_1 und V_0 gezeigt.

Der neue Facepunkt F_5 ergibt sich aus:

$$\begin{aligned} F_5 &= b\left[(u_3,v_3),(\tfrac{u_3+u_4}{2},\tfrac{v_3+v_4}{2}),(u_4,v_4)\right] \\ &= \frac{1}{4}\cdot(I_{05}+I_{15}+I_{55}+I_{25}) \end{aligned} \tag{2.17}$$

Ein Polygonzug, der aus den Kontrollpunkten des Kontrollnetzes gebildet wird und entlang bestehender Kanten des Kontrollnetzes verläuft, wird mit $\langle P_a,\ldots,P_z\rangle$ bezeichnet. Es sei S_1 der neue, durch Unterteilung entstandene Vertexpunkt der Kurve, welche aus dem Polygonzug $\langle P_8, P_1, P_5\rangle$ und dem lokalen vertikalen Knotenintervallvektor K_1^v von P_1 konstruiert wurde. Ebenso sei S_0 der neue Vertexpunkt der Kurve $\langle P_4, P_0, P_2\rangle$ mit K_0^v von P_0. Dann erhält man den neuen Kantenpunkt E_1 durch:

$$\begin{aligned} E_1 &= b[(u_3,v_2),(u_3,\tfrac{v_2+v_3}{2}),(u_3,v_3)] \\ &= \text{subdivCurveEdge}(S_1,S_0,e_1,e_2,e_3) \end{aligned} \tag{2.18}$$

Eine Unterteilung der Kanten der Kurve $\langle P_1, P_0, P_3\rangle$ mit dem lokalen Knotenintervallvektor K_0^h von P_0 liefert die Punkte M_1 und M_3 (siehe Abbildung 2.13). Auf gleiche Art und Weise erhält man die Punkte M_4 und M_2 durch Unterteilung der Kante der Kurve $\langle P_4, P_0, P_2\rangle$ mit dem lokalen Knotenintervallvektor K_0^v von P_0. Der neue Vertexpunkt der Fläche errechnet sich nun aus:

$$\begin{aligned} V_0 &= b[(\tfrac{u_2+u_3}{2},\tfrac{v_2+v_3}{2}),(u_3,v_3),(\tfrac{u_3+u_4}{2},\tfrac{v_3+v_4}{2})] \\ &= \frac{1}{4}\cdot\left(\frac{e_3\cdot d_2\cdot F_5+e_2\cdot d_2\cdot F_6+e_2\cdot d_3\cdot F_7+e_3\cdot d_3\cdot F_8}{(e_2+e_3)\cdot(d_2+d_3)}\right. \\ &\qquad \left.+\frac{e_3\cdot M_1+e_2\cdot M_3}{e_2+e_3}+\frac{d_2\cdot M_2+d_3\cdot M_4}{d_2+d_3}+P_0\right) \end{aligned} \tag{2.19}$$

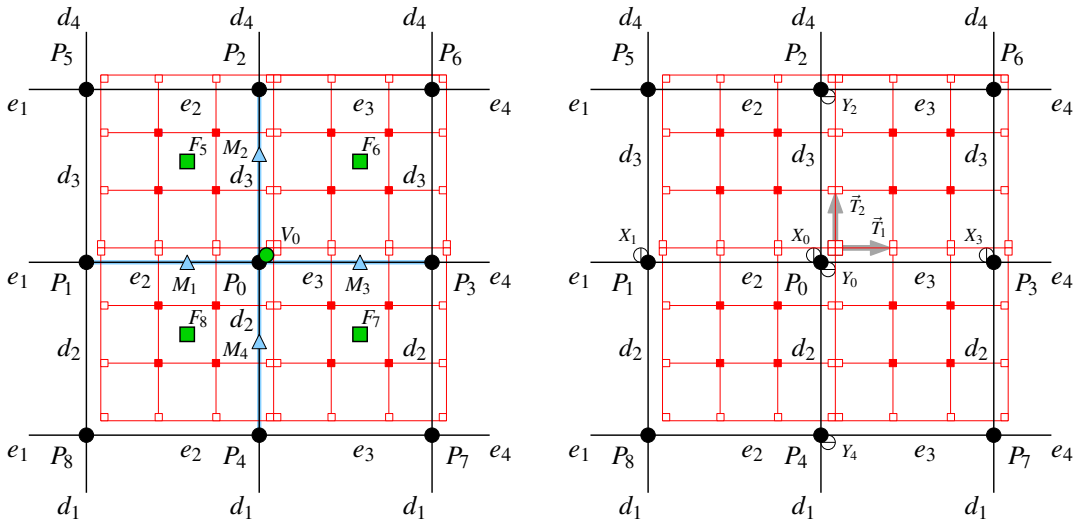


Abbildung 2.13: Links: Unterteilung eines NURBS-Patches: neuer Vertexpunkt. Die Polygonzüge $\langle P_1,P_0,P_3\rangle$ und $\langle P_4,P_0,P_2\rangle$ sind blau unterlegt, sowie die Hilfspunkte $M_1,\ldots,M_4$. Rechts: Bestimmung der Tangenten.

Wertet man die NURBS-Teilfläche, welche durch die Kontrollpunkte $P_0,\ldots,P_8$ und die lokalen Knotenintervallvektoren K_v^0,K_h^0 von P_0 definiert ist, für den Parameter (u_3,v_3) aus, so erhält man den Limitpunkt von P_0. Die Regel zur Bestimmung des Limitpunktes ist (siehe auch Formel (2.16)):

$$\begin{aligned} L &= b[(u_3,v_3),(u_3,v_3),(u_3,v_3)] \\ &= \frac{e_3\cdot d_2\cdot I_{05}+e_2\cdot d_2\cdot I_{06}+e_2\cdot d_3\cdot I_{07}+e_3\cdot d_3\cdot I_{08}}{(e_2+e_3)\cdot(d_2+d_3)} \\ &= A \\ &=: \text{limitPoint}(P_0,\ldots,P_8,e_1,\ldots,e_4,d_1,\ldots,d_4) \end{aligned} \tag{2.20}$$

Zur Berechnung der Limitnormale (siehe Abbildung 2.13) von P_0 sind die drei „vertikalen“ Kurven

$$\langle P_8,P_1,P_5\rangle,\langle P_4,P_0,P_2\rangle,\langle P_7,P_3,P_6\rangle$$

mit vertikalem, lokalen Knotenintervallvektor K_0^v von P_0 und die drei „horizontalen“ Kurven

$$\langle P_8,P_4,P_7\rangle,\langle P_1,P_0,P_3\rangle,\langle P_5,P_2,P_6\rangle$$

mit horizontalem lokalen Knotenintervallvektor K_0^h von P_0 erforderlich. Die Limitpunkte der vertikalen Kurven von $\langle P_1,P_0,P_3\rangle$ werden mit X_1,X_0,X_3 bezeichnet. Ebenso werden die Limitpunkte der horizontalen Kurven von $\langle P_4,P_0,P_2\rangle$ mit Y_4,Y_0,Y_2 benannt. Damit ergibt sich die Limitnormalen-Regel als:

$$\begin{aligned} \vec{N} &= \|\vec{T}_1\times\vec{T}_2\| \\ &= \|(b[(u_3,v_3),(u_3,v_3),(u_4,v_3)]-b[(u_3,v_3),(u_3,v_3),(u_3,v_3)])\times \\ &\quad (b[(u_3,v_3),(u_3,v_3),(u_3,v_4)]-b[(u_3,v_3),(u_3,v_3),(u_3,v_3)])\| \\ &= \|\text{tangentCurveVertex}(X_1,X_0,X_3,e_1,\ldots,e_4)\times \\ &\quad \text{tangentCurveVertex}(Y_4,Y_0,Y_2,d_1,\ldots,d_4)\| \\ &=: \text{limitNormal}(P_0,\ldots,P_8,e_1,\ldots,e_4,d_1,\ldots,d_4) \end{aligned} \tag{2.21}$$

Die neuen Knotenintervalle werden den neuen Kanten zugewiesen (siehe Abbildung 2.14).

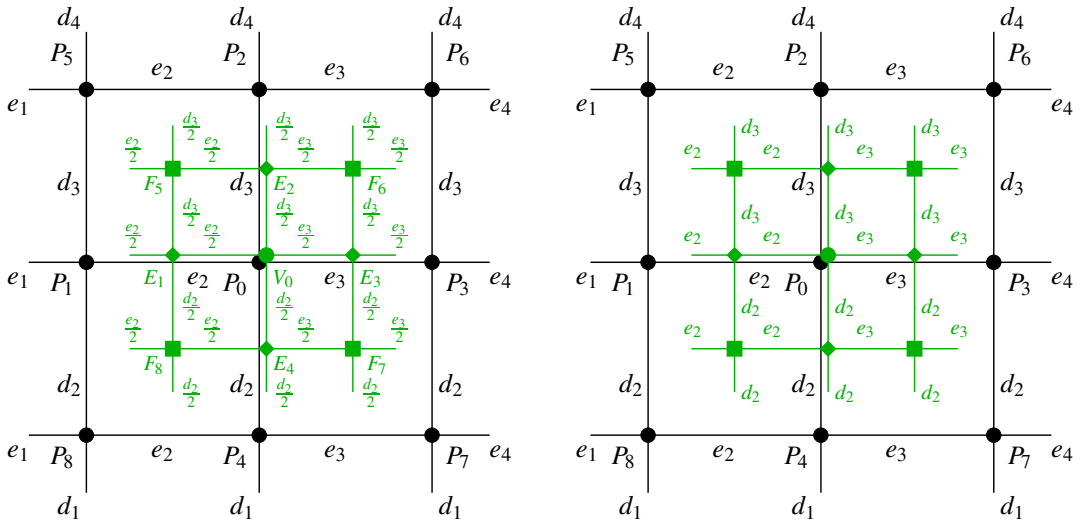


Abbildung 2.14: Die neuen Knotenintervalle werden zu den neuen Kanten zugewiesen (links). Zur Vereinfachung der Schreibweise wird bei der Zuweisung Faktor zwei verwendet, welches die Teilverhältnisse unverändert lässt (rechts).

2.3.3 Catmull-Clark-Unterteilungsflächen

Setzt man in Formel (2.17) bis (2.21) die Knotenintervalle uniform, so erhält man die Unterteilungs- sowie Limitpunkt- und Normalenregeln für uniforme B-Spline-Flächen (siehe Abbildung 2.16). Catmull und Clark erweiterten in [CC78] diesen Regelsatz: Statt eingeschränkter $n \times m$ Kontrollnetzstrukturen sind beliebige Topologien im Kontrollnetz verwendbar.

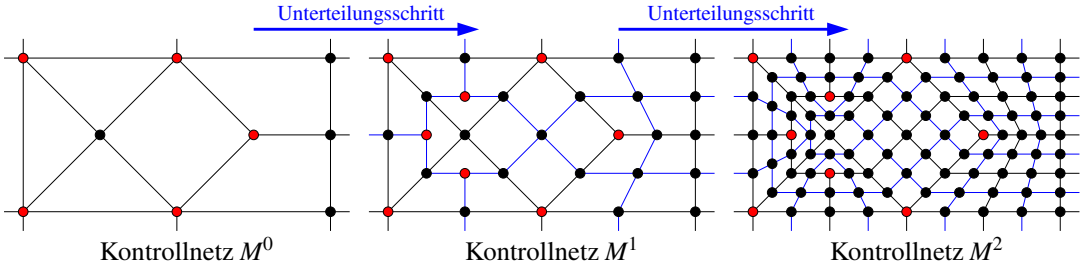


Abbildung 2.15: Catmull-Clark-Unterteilungsschema. Die rot markierten Punkte sind irregulär.

Das Unterteilungsschema von Catmull-Clark ist in Abbildung 2.15 dargestellt. Ausgehend von einem Ausgangskontrollnetz M^0 mit Vertices und Faces beliebiger Valenz, werden während eines Unterteilungsschrittes neue Face-, Kanten- und Vertexpunkte bestimmt. Die neuen Punkte werden wie in Abbildung 2.15 gezeigt verbunden. Dabei entstehen aus einem n-Eck n neue Vierecke. Der neue Facepunkt besitzt demzufolge Valenz n. Das resultierende Kontrollnetz M^1 ist ein reines Viereckesnetz. Beim Unterteilen der Fläche wächst die Faceanzahl exponentiell: aus m Viereckesfaces werden nach l Unterteilungsschritten $m \cdot 4^l$ neue Faces. Beim Unterteilen des Ausgangskontrollnetzes M^0 erhöht sich die Anzahl der irregulären Punkte um die Anzahl der n-Ecke, $n \neq 4$, in M^0. Für alle weiteren Unterteilungen des Kontrollnetzes M^l, $l > 0$, bleibt die Anzahl der irregulären Punkte konstant. Die jeweils neu berechneten Face- und Kantenpunkte sind immer regulär. Vertexpunkte behalten beim Unterteilen ihre Valenz. Wird ein Kontrollpunkt P_i^l des Kontrollnetzes M^l mit seiner 1-Nachbarschaft unendlich oft unterteilt, so erhält man mit $L_i = \lim_{l \to \infty} P_i^l$ seinen Limitpunkt auf der Unterteilungsfläche. Die Normale des Limitpunktes L_i wird durch $\vec{N}_i = \vec{T}_1 \times \vec{T}_2$ berechnet [HKD93], wobei $\vec{T}_1, \vec{T}_2$ die beiden Oberflächentangenten an der Stelle L_i sind. Ebenso wie die B-Spline-Fläche ist die resultierende Unterteilungsfläche C^2-stetig mit Ausnahme der irregulären Punkte, dort liegt lediglich eine C^1-Stetigkeit vor [ZS99].

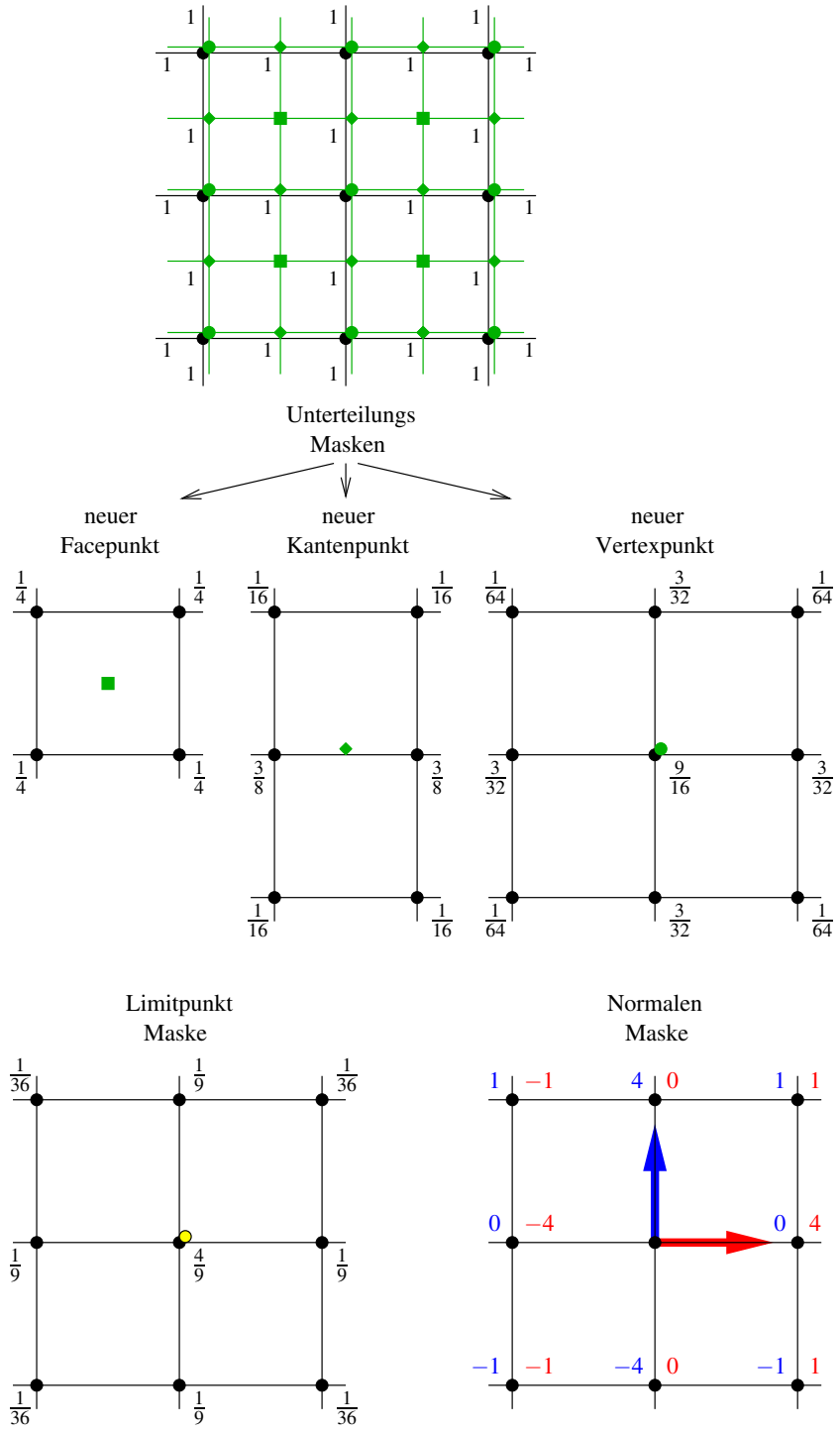


Abbildung 2.16: Unterteilungs- und Limitpunktregeln sowie Normalenberechnung bei uniformen B-Spline-Flächen.

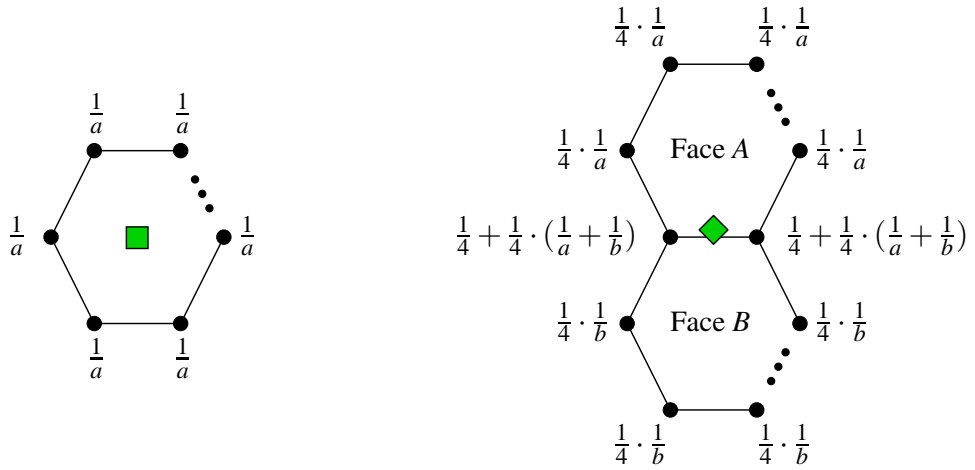


Abbildung 2.17: Masken für Face- und Kantenpunkte mit beliebigen Facevalenzen a und b.

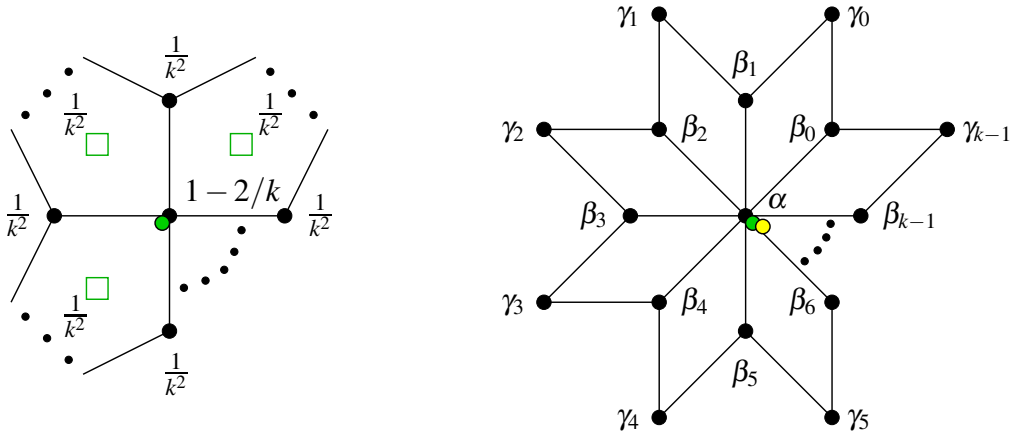


Abbildung 2.18: Berechnung eines Vertexpunktes mit Vertexvalenz k und beliebigen Facevalenzen (links). Die grünen Quadrate markieren die dazu benötigten neuen Facepunkte. Maske für Vertex- und Limitpunkte mit Vertexvalenz k in einem Viereckskontrollnetz (rechts).

Catmull-Clark-Regeln	α	β_i	γ_i
Unterteilung (Abb. 2.18)	$1-\frac{3}{2k}-\frac{1}{4k}$	$\frac{3}{2k^2}$	$\frac{1}{4k^2}$
Limitpunkt (Abb. 2.18)	$1-\frac{5}{k+5}$	$\frac{4}{(k+5)k}$	$\frac{1}{(k+5)k}$
Normale (Abb. 2.18)			
Tangente $\vec{T}_1$	0	$a(k)\cos\frac{2\pi(i+1)}{k}$	$\cos\frac{2\pi(i+1)}{k}+\cos\frac{2\pi(i+2)}{k}$
Tangente $\vec{T}_2$	0	$a(k)\cos\frac{2\pi i}{k}$	$\cos\frac{2\pi i}{k}+\cos\frac{2\pi(i+1)}{k}$

Tabelle 2.1: Gewichte für einen Unterteilungsschritt sowie Limitpunkt- und Tangentenberechnung bei Catmull-Clark-Unterteilungsflächen mit $a(k)=1+\cos\frac{2\pi}{k}+\cos\frac{\pi}{k}\sqrt{2(9+\cos\frac{2\pi}{k})}$.

Der Regelsatz für Catmull-Clark-Flächen ist in Abbildung 2.17 und 2.18 sowie Tabelle 2.1 aufgeführt. Neue Face-, Kanten- und Vertexpunkte werden durch Konvexkombinationen ihrer 1-Nachbarschaften berechnet, die entsprechenden Gewichte sind in Abbildung 2.17 und 2.18 präsentiert. Der neue Vertexpunkt von P_0^l mit Valenz k im Viereckskontrollnetz ergibt sich beispielsweise aus seiner 1-Nachbarschaft $P_{\beta i}^l, P_{\gamma i}^l$ durch:

$$P_0^{l+1} = \alpha \cdot P_0^l + \sum_{i=0}^{k-1} \beta_i \cdot P_{\beta i}^l + \gamma_i \cdot P_{\gamma i}^l$$

Die Regeln zur Limitpunkt- und Limitnormalenberechnung von P_i^l aus seiner 1-Nachbarschaft in M^l sind in Tabelle 2.1 aufgelistet [HKD93].

Special Features

Um besondere Akzente im Modell wie z.B. Falten, Knicke und Spitzen zu erhalten, bieten sich die Special Features der Unterteilungsflächen an. Zur Erzielung dieser Effekte nutzen die Unterteilungsregeln nicht die kompletten ausgewiesenen Masken, sondern beschränken sich auf einen geeigneten Teilbereich. Der Regelsatz für Special Features ist in Tabelle 2.2 sowie in Abbildung 2.19 aufgelistet. Mit diesem Regelsatz lassen sich z.B. Spitzen (Corner), glatt auslaufende scharfe Kanten (Darts) sowie scharfe und halbscharfe Kanten erzeugen. Bei einem Corner wird der entsprechend markierte Kontrollpunkt beim Unterteilen unverändert übernommen, daraus resultiert eine Spitze im Modell. Knicke im Modell entstehen durch einen als scharf markierten Kantenzug, nur die Kontrollpunkte auf den scharfen Kanten werden im Regelsatz verwendet. Halbscharfe Kanten können erzielt werden, indem $n-$mal der scharfe Regelsatz und anschließend der glatte Regelsatz angewendet wird. Um Darts zu erreichen werden in der 1-Nachbarschaft des Endvertex der scharfen Kante stets die glatten Regeln angewendet, wie in Abbildung 2.19 demonstriert.

scharfe Kanten und Ränder	α	β_i	γ_i
Unterteilung (Abb. 2.19)	$\frac{3}{4}$	$\frac{1}{8}$ für $i = g,h$, 0 sonst	0
Limitpunkt (Abb. 2.19)	$\frac{4}{6}$	$\frac{1}{6}$ für $i = g,h$, 0 sonst	0
Normale (Abb. 2.20)			
Tangente $\vec{T}_1$ $(k > 2)$	$4r(k)\left(\cos\frac{\pi}{k} - 1\right)$	$-r(k)\left(1 + 2\cos\frac{\pi}{k}\right)$ für $i = 0,k$, $\frac{4\sin i\frac{\pi}{k}}{\left(3+\cos\frac{\pi}{k}\right)k}$ sonst	$\frac{\left(\sin i\frac{\pi}{k} + \sin(i+1)\frac{\pi}{k}\right)}{\left(3+\cos\frac{\pi}{k}\right)k}$
Tangente $\vec{T}_2$	0	1 für $i = 0$, -1 für $i = k$, 0 sonst	0

Tabelle 2.2: Special Feature-Regeln für Catmull-Clark mit $r(k) = \frac{\cos\frac{\pi}{k}+1}{k\sin\frac{\pi}{k}\left(3+\cos\frac{\pi}{k}\right)}$

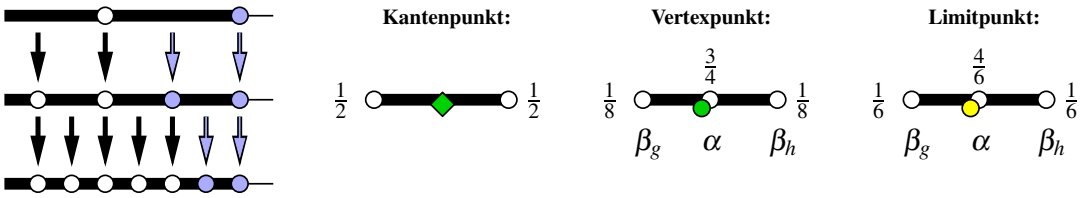


Abbildung 2.19: Regelsatz für eine scharfe Kante (schwarz markiert). Bei einer weich auslaufenden scharfen Kante werden in der 1-Nachbarschaft des Kantenendes die glatten Unterteilungsregeln (blau gekennzeichnet) benutzt (links). Der übrige Bereich der scharfen Kante verwendet die scharfen Regeln für neue Kantenpunkte, neue Vertexpunkte und Limitpunkte.

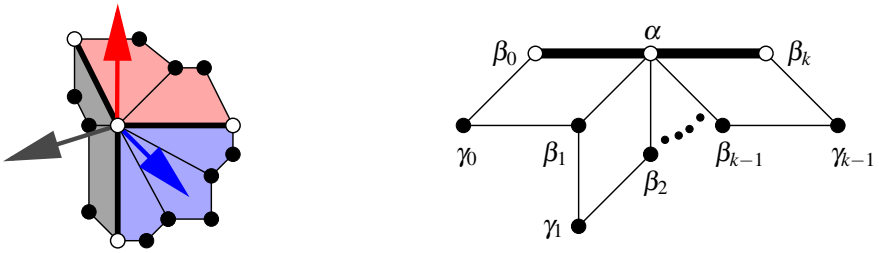


Abbildung 2.20: Maske für die Normalenberechnung bei Catmull-Clark-Flächen an einer scharfen Kante.

Im Gegensatz zu einem Limitpunkt, der aus den glatten Regeln resultiert, kann ein Limitpunkt, welcher mit Hilfe des Special Feature-Regelsatz berechnet wurde, mehrere Normalen besitzen (siehe Abbildung 2.20). Hat der Special Feature-Limitpunkt s inzidente scharfe Kanten, so sind ihm s glatte Regionen und demzufolge s Normalen zugeordnet. Die zwei dedizierten Tangenten für jede Region verlaufen entlang der scharfen Kante sowie quer zur scharfen Kante. Die entsprechenden Regeln sind in Tabelle 2.2 aufgelistet.

2.3.4 Loop-Unterteilungsflächen

Charles Loop stellte in [Loo87] eine Unterteilungsfläche des Typs *approximierend*, zur Klasse *Vertex einfügen* gehörend und *Dreiecksnetz erzeugend* vor [ZS99]. Loop's Fläche basiert auf Box-Splines und verwendet Dreiecksnetze beliebiger zwei-mannigfaltiger Topologie.

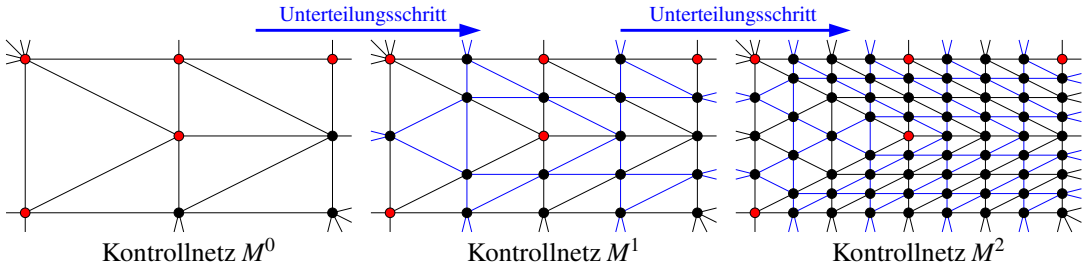


Abbildung 2.21: Loop-Unterteilungsschema.

Bei einem Unterteilungsschritt werden neue Kanten und Vertexpunkte berechnet, die wie in Abbildung 2.21 illustriert, verbunden werden. Aus einem Dreieck entstehen somit beim Unterteilen 4 neue Dreiecke. Die Faceanzahl wächst ebenfalls wie bei den Catmull-Clark-Flächen exponentiell: nach l Unterteilungsschritten eines Kontrollnetzes mit m Dreiecken liegen $m \cdot 4^l$ Dreiecke vor. Ebenso bleibt die Anzahl der irregulären Vertices beim Unterteilen konstant, d.h. die neu hinzu kommenden Vertices sind

immer regulär. Der Regelsatz zum Unterteilen der Fläche sowie zur Limitpunkt- und Normalenberechnung ist in Tabelle 2.3 und Abbildung 2.22 aufgelistet. Die resultierende Unterteilungsfläche ist C^2-stetig mit Ausnahme der irregulären Punkte, dort liegt lediglich eine C^1-Stetigkeit vor [ZS99].

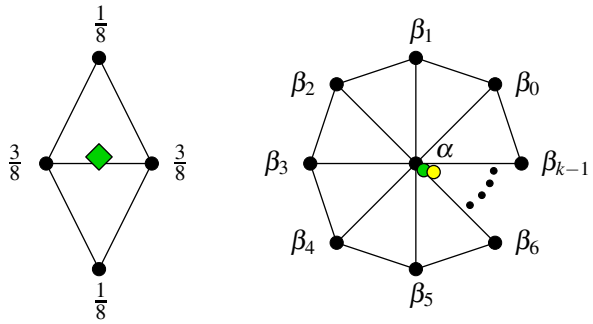


Abbildung 2.22: Masken für Kanten-, Vertex- und Limitpunkte mit Vertexvalenz k für Loop-Unterteilungsflächen.

Loop-Regeln	α	β_i
Unterteilung (Abb. 2.22)	$1-ka(k)$	$a(k)$
Limitpunkt (Abb. 2.22)	$1-\frac{k}{3/(8a(k))+k}$	$\frac{1}{3/(8a(k))+k}$
Normale (Abb. 2.22)		
Tangente $\vec{T}_1$	0	$\cos\frac{2\pi i}{k}$
Tangente $\vec{T}_2$	0	$\sin\frac{2\pi i}{k}$

Tabelle 2.3: Gewichte für einen Unterteilungsschritt sowie Limitpunkt- und Tangentenberechnung bei Loop-Unterteilungsflächen mit $a(k)=\frac{1}{k}\left(\frac{5}{8}-\left(\frac{3}{8}+\frac{1}{4}\cos\frac{2\pi}{k}\right)^2\right)$.

Special Features

Für Loop-Flächen stehen die gleichen Special Features wie bei den Catmull-Clark-Flächen zur Verfügung. Die Anwendungsweise ist die gleiche wie bereits für Catmull-Clark-Flächen beschrieben, ebenso werden auf den scharfen Kanten die selben Gewichte benutzt wie in Abbildung 2.19 illustriert. Der ent-

Abbildung 2.23: Maske für die Normalenberechnung bei Loop-Flächen an einer scharfen Kante.

sprechende Regelsatz zur Unterteilung, Limitpunktbestimmung und Normalenberechnung ist in Tabelle 2.4 mit Abbildung 2.23 aufgeführt.

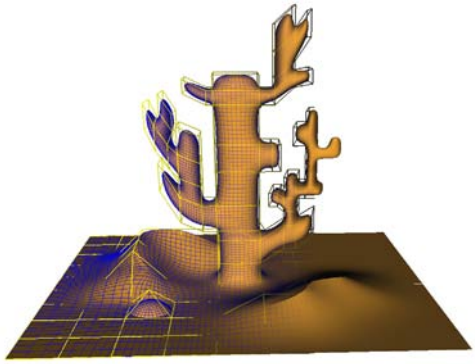
scharfe Kanten und Ränder	α	β_i
Unterteilung (Abb. 2.19)	$\frac{3}{4}$	$\frac{1}{8}$ für $i = g,h$, 0 sonst
Limitpunkt (Abb. 2.19)	$\frac{4}{6}$	$\frac{1}{6}$ für $i = g,h$, 0 sonst
Normale (Abb. 2.23)		
Tangente $\vec{T}_1$	0	1 für $i = 0$, -1 für $i = k$, 0 sonst
Tangente $\vec{T}_2$ $(k > 3)$	0	$\frac{\sin\frac{\pi}{k-1}}{2\cos\frac{\pi}{k-1}-2}$ für $i = 0,k$, $\sin\frac{\pi i}{k-1}$ sonst

Tabelle 2.4: Special Feature-Regeln für Loop-Unterteilungsflächen.

Die vorgestellten Catmull-Clark- und Loop-Regeln entsprechen dem originalen Regelsatz beider Flächen, die auch in den nachfolgenden Kapiteln dieser Arbeit verwendet wurden. Für beide Unterteilungsschemata existieren diverse Regelmodifikationen. Ein Überblick über diese Regeländerungen und Flächenvariationen ist in [ZS99] zu finden.

Kapitel 3

Erweiterte Unterteilungsflächen



Catmull und Clark erweiterten 1978 in [CC78] die Unterteilungsregeln für uniforme B-Spline-Flächen um beliebige Valenzen und schufen damit einen neuen Oberflächentyp. Es liegt daher als Weiterentwicklung nahe, die Unterteilungsregeln für NURBS ebenfalls um beliebige Valenzen zu ergänzen. Abschnitt 3.1 gibt einen Überblick über bereits bestehende Verfahren, klassifiziert sie und zeigt ihre Vor- und Nachteile auf. Anhand gebräuchlicher Anwendungen im CAD- und Computergraphik-Umfeld wird in Abschnitt 3.2 ein Anforderungsprofil für eine nicht-uniforme Unterteilungsfläche skizziert. Da bei den bereits vorhandenen nicht-uniformen Unterteilungsflächen wichtige Punkte im Anforderungsprofil fehlen, wird eine neue erweiterte Unterteilungsfläche (extended Subdivision Surface: ESubs) präsentiert. Der Regelsatz ist Thema des Abschnitts 3.3. Für reguläre sowie irreguläre Topologien werden die Unterteilungs- und Limitregeln detailliert vorgestellt. Einhergehend mit den Limitpunktregeln wird in Abschnitt 3.4 die Konvergenz der Unterteilungsregeln untersucht. Eine abschließende Betrachtung der Stetigkeitseigenschaften der Fläche in Abschnitt 3.5 rundet die Analyse der neuen Fläche ab. Der letzte Abschnitt 3.6 dieses Kapitels widmet sich den Special Features, die ebenso wie bei Catmull-Clark-Flächen ergänzende Modellierungsoptionen bieten. Die ESubs-Unterteilungsflächen wurden in [MRF06] vorgestellt.

3.1 Überblick und Klassifizierung bisheriger Ansätze

Im Bereich *nicht-uniformer Unterteilungsflächen* als Verallgemeinerung von NURBS- und Catmull-Clark-Flächen stehen die Flächentypen aus [SSS98] und [SZBN03] von Sederberg et. al. zur Verfügung. Nach den Klassifizierungskriterien aus [ZS99] gehören diese nicht-uniformen Unterteilungsflächen in die Kategorie *Vertex einfügend, approximierend* und *Viereckснetz erzeugend*. Um innerhalb dieser Kategorie eine deutlichere Abstufung zu erhalten, ist die Hinzunahme folgender Kriterien sinnvoll:

- uniformer oder nicht-uniformer Flächentyp
- Verfügbarkeit von Limitpunktregeln

Eine NURBS-Unterteilungsfläche (siehe Abschnitt 2.3.2), als Basisfläche für alle weiteren vorgestellten Flächen, erhält somit in der Klassifizierung die Zusatzattribute *nicht-uniformer Flächentyp* und *Limitpunktregeln verfügbar*.

Die Knotenintervallbelegung einer nicht-uniformen Unterteilungsfläche kann über die Möglichkeiten einer nicht-uniformen B-Spline-Fläche hinausgehen, daher ist folgende Definition nötig:

Definition 6 (konforme und nicht-konforme Faces). *Es sei M ein Viereckskontrollnetz mit Knotenintervallen auf seinen Kanten und F ein Face von M. Sind die Knotenintervalle auf den gegenüberliegenden Seiten von F gleich, so ist F ein konformes Face. Ist dies nicht der Fall, so heißt F nicht-konformes Face.*

Das Kontrollnetz der NURBS-Unterteilungsflächen besteht aus konformen Faces. Bei den allgemeinen nicht-uniformen Unterteilungsflächen können jedoch auch nicht-konforme Faces auftreten.

Sederberg et. al. erweiterten als erste in [SSS98] die Unterteilungsregeln von bikubischen NURBS-Flächen um beliebige Valenzen zu *Non-Uniform Recursive Subdivision Surfaces (NURSS)*. Im nicht-uniformen Kontrollnetz der NURSSes können die Knotenintervalle beliebig gesetzt werden, Special Features werden durch ausgewählte null wertige Knotenintervalle erzielt. Werden alle Knotenintervalle uniform gewählt, so reduziert sich die Fläche auf eine Catmull-Clark-Fläche. Ebenso erhält man bei der Verwendung eines Viereckskontrollnetzes mit regulärer Topologie und konformen Faces aus den NURSS eine NURBS-Fläche. NURSSes konvergieren gegen eine Grenzfläche, Limitpunktregeln können jedoch nicht angegeben werden.

In [SZBN03] präsentierten Sederberg et. al. eine weitere Form der nicht-uniformen Unterteilungsflächen: *Non-Uniform-Rational-Catmull-Clark-Flächen (NURCCs)* sind im Regelsatz *identisch* mit NURSSes, erlauben jedoch nur gleiche Knotenintervalle an gegenüberliegenden Viereckssseiten. Mit dieser Einschränkung erreichen sie stationäre Unterteilungsregeln, wodurch eine lokale Verfeinerung im Kontrollnetz möglich wird. NURCCes mit diesen lokalen Verfeinerungen über T-Vertices werden T-NURCCes genannt. Kontrollpunkte können somit beliebig im Kontrollnetz als T-Vertices eingefügt werden, ohne die Limitfläche zu verändern. Limitpunktregeln selbst sind jedoch bislang nicht verfügbar.

3.2 Motivation und Anforderungen

Um im CAD- und Computergraphik-Umfeld eingesetzt zu werden, sollte eine neue Fläche aus der Kategorie nicht-uniforme Unterteilungsflächen die folgenden Anforderungen erfüllen:

Verallgemeinerung von NURBS- und Catmull-Clark-Flächen : NURBS- sowie Catmull-Clark-Flächen sollen mit dieser Flächenkategorie darstellbar sein: Liegen im Kontrollnetz nur uniforme Knotenintervalle vor, so reduziert sich die Fläche zu einer Catmull-Clark-Fläche. Eine NURBS-Fläche ergibt sich, wenn in einem Viereckskontrollnetz eine reguläre Topologie mit konformen Faces vorliegt.

Nicht-konforme Faces : In Erweiterung zu der Knotenintervallbelegung von NURBS soll es möglich sein, unterschiedliche Knotenintervalle auf gegenüberliegenden Seiten eines Faces zu setzen. Dies erlaubt eine größere Flexibilität und Vielfalt in der Modellgenerierung. NURBS-Kontrollnetze mit unterschiedlichen Knotenvektoren können beispielsweise auf diese Weise zusammengesetzt werden. Weiterhin werden spezielle Effekte und Formen mit nicht-konformen Faces möglich.

Limitpunktregeln : Zum Einsatz der Fläche in der adaptiven Visualisierung, Modellierung sowie in CAD-Anwendungen sind Limitpunkte unabdingbar. Daher müssen neben den Unterteilungsregeln auch die entsprechenden Limitpunktregeln im Regelsatz dieser Flächenkategorie vertreten sein.

Special Features : Eine Erweiterung der Modelliervielfalt bieten Special Features mit z.B. scharfen Kanten und Spitzen. Dieses Zusatzwerkzeug ist bei aktuellen Unterteilungsflächen Standard und muss auch bei dieser Flächenkategorie fester Bestandteil des Regelsatzes sein.

Keine der bislang verfügbaren nicht-uniformen Unterteilungsflächen erfüllt alle diese Kriterien. Es ist daher ein Bedarf vorhanden, eine Fläche zu entwickeln, die diesen Anforderungen entspricht. Eine Zusammenfassung der vorgestellten Flächen mit ihren jeweiligen Eigenschaften ist in Tabelle 3.1 zu finden.

	beliebige Topologie möglich	nicht-uniforme Fläche darstellbar	nicht-konforme Faces möglich	Special Features integriert	Limitpunkt-regeln verfügbar
NURBS		✔			✔
Catmull-Clark	✔			✔	✔
NURSS	✔	✔	✔	durch null Knotenintervalle	
T-NURCCs	✔	✔		durch null Knotenintervalle	
ESubs	✔	✔	✔	✔	✔

Tabelle 3.1: Zusammenfassung der relevanten Flächen.

Abbildung 3.1 verdeutlicht den Zusammenhang zwischen ESubs, NURBS und Catmull-Clark: Uniforme B-Spline-Flächen sind die gemeinsame Schnittmenge von NURBS- und Catmull-Clark-Flächen. ESubs enthalten sowohl NURBS- als auch Catmull-Clark-Flächen. Im nachfolgenden Abschnitt 3.3 wird der Regelsatz für die nicht-uniforme Unterteilungsfläche ESubs unter Berücksichtigung der gestellten Kriterien hergeleitet.

3.3 Regelsatz für erweiterte Unterteilungsflächen

Ausgehend von einem Ausgangskontrollnetz M^0 mit beliebiger Topologie (zwei-mannigfaltig, mit und ohne Rand), werden bei einem Unterteilungsschritt mit dem ESubs-Regelsatz neue Face-, Kanten- und Vertexpunkte berechnet. Diese neuen Punkte werden in gleicher Weise wie bei dem Catmull-Clark-Schema miteinander verbunden. Das resultierende Kontrollnetz M^1 ist ein Viereckснetz. Bei weiteren n Unterteilungsschritten entstehen aus m Vierecken des Kontrollnetzes M^1 $m \cdot 4^n$ neue Faces. Es liegt ein exponentielles Wachstum der Faceanzahl vor. Die Anzahl der irregulären Punkte erhöht sich beim Unterteilen des Ausgangskontrollnetzes M^0 um die Anzahl der Polygone, welche keine Vierecke sind. Alle weiteren Unterteilungsschritte erhöhen die Anzahl der irregulären Punkte nicht. Für M^n, $n > 0$ bleibt die Anzahl der irregulären Punkte somit konstant.

Ebenso wie bei den Unterteilungsflächen von Sederberg et. al. erhalten die Kanten des ESubs-Kontrollnetzes Knotenintervalle. Sind diese Knotenintervalle uniform, so liefern die ESubs-Regeln eine Catmull-Clark-Fläche. Liegen die Knotenvektoren einer NURBS-Fläche zugrunde und sind somit die Knotenintervalle auf den gegenüberliegenden Seiten eines Faces gleich, so erhält man bei regulärer Topologie eines Viereckснetzes eine bikubische NURBS-Fläche. Ohne diese Einschränkungen sind die ESubs im allgemeinen nicht-uniform und nicht-stationär. Der Regelsatz für ESubs, bestehend aus neuen Face-, Kanten-, und Vertexregeln sowie den Limitpunktregeln, basiert auf dem Regelsatz für bikubische NURBS-Flächen. Abbildung 3.1 zeigt den Zusammenhang zwischen der neu zu entwickelnden nicht-uniformen Unterteilungsfläche ESubs sowie ihren Basis-Flächen Catmull-Clark und NURBS.

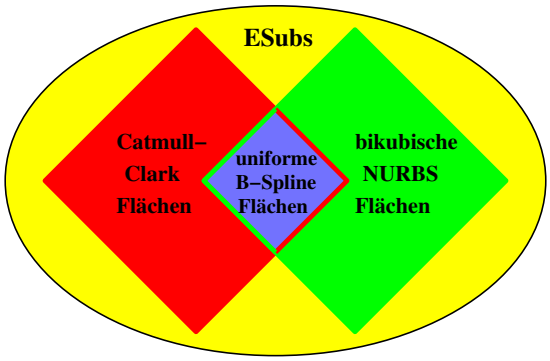


Abbildung 3.1: Zusammenhang NURBS-, Catmull-Clark-, uniforme B-Spline- und ESubs-Flächen.

Um für ein neues Schema den Regelsatz zu bestimmen, wird üblicherweise folgendermaßen vorgegangen: In Abhängigkeit der gewünschten Eigenschaften der neuen Fläche werden die Verfeinerungsregeln spezifiziert und die entsprechende Unterteilungsmatrix S generiert. Mit Hilfe der Eigenanalyse von S können dann die Limitpunktregeln der neuen Unterteilungsfläche ermittelt werden. Wird ein nicht-stationäres Unterteilungsschema benutzt, so ändert sich die Unterteilungsmatrix bei jeder Iteration. Eine Eigenanalyse kommt aus diesem Grunde bei nicht-stationären Verfahren nicht zum Einsatz. In dieser Arbeit wurde daher eine neue Herangehensweise gewählt, um die Limitpunktregeln zu berechnen.

Zur Bestimmung des Regelsatzes der nicht-uniformen, nicht-stationären ESubs wurde ein neues Verfahren entwickelt: Im ersten Schritt werden die neuen Face-, Kanten- und *vorgegebene* Limitpunkte berechnet. Dazu wird jedem Vertex mit Valenz vier zwei Knotenvektoren zugeordnet, die aus den Knotenintervallen der Kontrollnetzkanten ermittelt werden. Mit Hilfe dieser lokalen Knotenvektoren können analog zu den NURBS-Flächen lokale Bézier-Kontrollpunkte rund um den Vertex berechnet werden. Diese lokalen Bézier-Kontrollpunkte dienen als temporäre Variablen, um die neuen Face-, Kanten- und Limitpunkte zu berechnen. Der noch fehlende Vertexpunkt wird aus den umgebenden neuen Face- und Kantenpunkten sowie seinem *Limitpunkt* berechnet. Auf diese Weise wird sichergestellt, dass der neue Vertexpunkt gegen seinen vorgegebenen Limitpunkt konvergiert. Dies ist nicht nur eine neue Methode, um Vertexpunkte zu berechnen, sondern auch die erste die Limitpunktregeln für ein nicht-stationäres Schema liefert.

Während die Anzahl der regulären Punkte beim Unterteilen exponentiell wächst, bleibt die Anzahl der irregulären Punkte nach einem Unterteilungsschritt konstant. Für diese konstante Anzahl von irregulären Punkten wird der Catmull-Clark-Regelsatz verwendet.

Der Regelsatz für ESubs wird zunächst für reguläre Topologien hergeleitet und anschließend auf irreguläre Gebiete erweitert.

3.3.1 Reguläre Topologie

Ebenso wie bei den NURBS-Unterteilungsflächen (siehe Abschnitt 2.3.2), sind bei den ESubs jedem regulären Kontrollpunkt P_i ein horizontaler und ein vertikaler Knotenintervallvektor zugeordnet. In Abbildung 3.2 sowie 3.3 sind dies für P_0 die Knotenintervallvektoren

$$K_0^v = \{d_1, d_2, d_3, d_4\} \text{ (vertikal) und } K_0^h = \{e_1, e_2, e_3, e_4\} \text{ (horizontal)}.$$

Bei NURBS-Unterteilungsflächen müssen diese lokalen Knotenintervallvektoren partiell in ihrer Nachbarschaft gleich sein: Ist ein Kontrollpunkt P_j horizontal benachbart zu P_i, so haben beide Kontrollpunkte gleiche vertikale Knotenintervallvektoren. Analog gilt: Ist P_j vertikaler Nachbar zu P_i, so sind ihre horizontalen Knotenintervallvektoren identisch. ESubs kommen ohne diese Voraussetzung aus, so dass die entsprechenden Knotenintervalle ohne diese Einschränkung gewählt werden können. In Abbildung 3.2

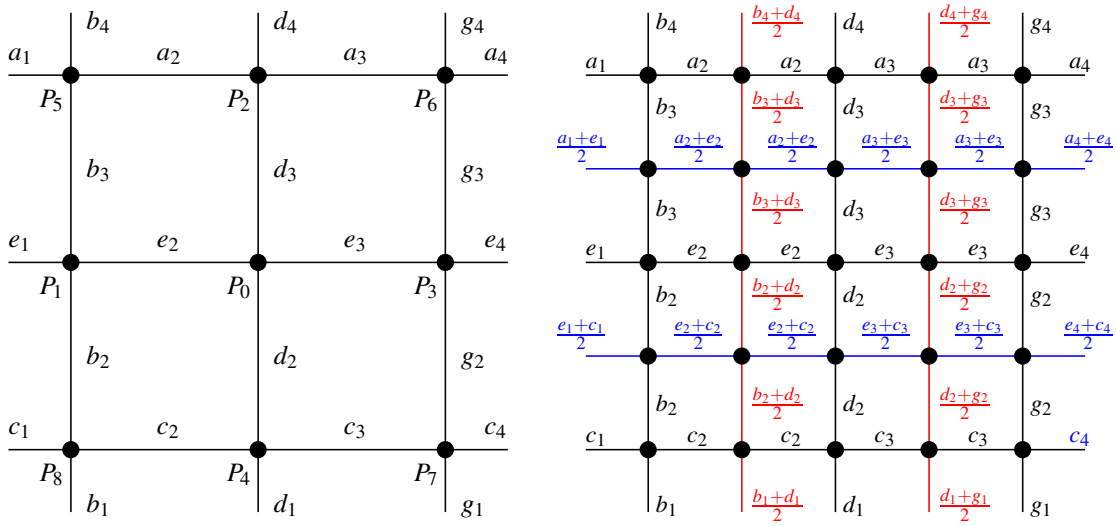


Abbildung 3.2: Reguläres ESubs-Kontrollnetz mit Knotenintervallen an den Kanten (links) und nach einer Unterteilung (rechts). Die neu hinzugekommenen, vertikalen Kanten sind mit ihren entsprechenden Knotenintervallen rot markiert, analog dazu sind die horizontalen Kanten mit Knotenintervallen blau eingefärbt.

sowie 3.3 hat z.B. der zu P_0 horizontal benachbarte Kontrollpunkt P_1 den vertikalen Knotenintervallvektor $K_1^v = \{b_1, b_2, b_3, b_4\}$, so dass $K_1^v \neq K_0^v$ gilt. Bei dem zu P_0 vertikalen Nachbarn P_2 mit seinem horizontalen Knotenintervallvektor $K_2^h = \{a_1, a_2, a_3, a_4\}$ ist ebenfalls ein unterschiedlicher horizontalen Knotenintervallvektor $K_2^h = \{a_1, a_2, a_3, a_4\} \neq K_0^h$ vorhanden. ESubs erlauben somit nicht-konforme Faces, wie in Abbildung 3.2 in schematischer Darstellung zu sehen.

Nach einem Unterteilungsschritt erfolgt die Zuweisung der Knotenintervalle zu den Kanten analog zu der Vorgehensweise der NURBS-Unterteilungsfläche (siehe Abbildung 2.14). Bei nicht-konformen Faces werden die Knotenintervalle auf den neuen Kanten durch Mittelung der beiden horizontalen respektive vertikalen Knotenintervalle berechnet (siehe Abbildung 3.2).

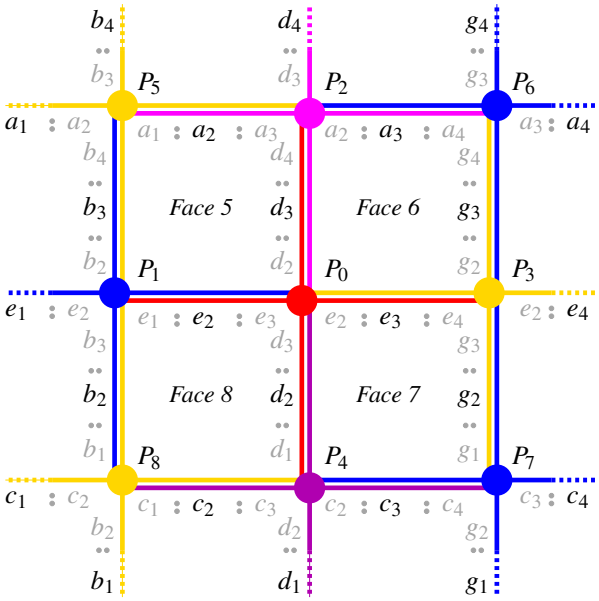


Abbildung 3.3: Lokale Knotenintervallvektoren eines regulären ESubs-Kontrollnetzes.

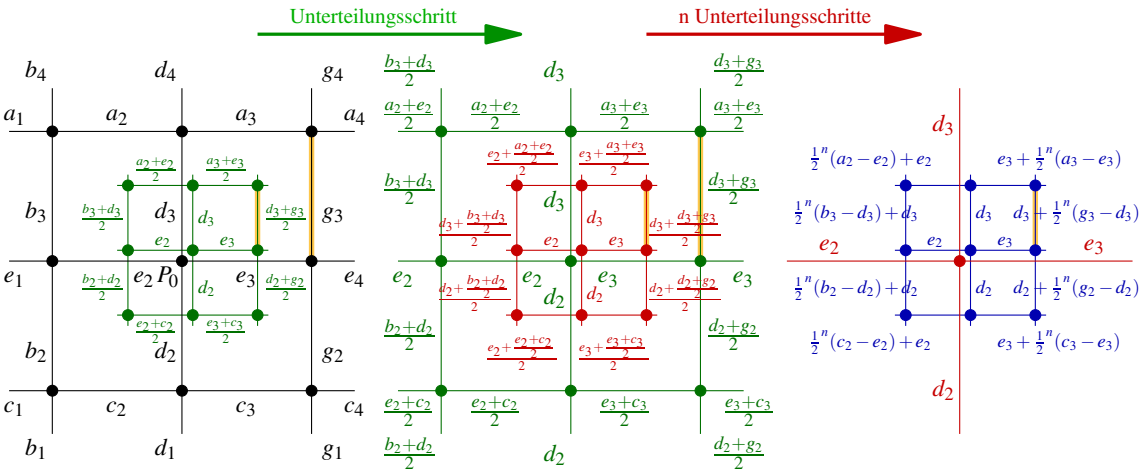


Abbildung 3.4: Knotenintervallzuweisung bei Unterteilung. Für $n \to \infty$ konvergieren die Knotenvektoren in der 1–Nachbarschaft zum Knotenvektor des eingeschlossenen Punktes.

Es sei $P_0 = P_0^0$ ein Vertex des Kontrollnetzes M^0 mit Valenz vier, P_0^n sei aus n Unterteilungen des Vertex P_0 gebildet worden. Die Knotenintervallvektoren des Kontrollpunktes P_0 ändern sich nach einem Unterteilungsschritt zu $K_0^v = \{d_2, d_2, d_3, d_3\}$, $K_0^h = \{e_2, e_2, e_3, e_3\}$ und bleiben bei weiteren Unterteilungen für P_0^n, mit $n > 0$, konstant (siehe Abbildung 3.4). Die Knotenintervalle in der 1–Nachbarschaft von P_0 sind ebenfalls in Abbildung 3.4 dargestellt. Für ein solches Knotenintervall gilt nach n Unterteilungsschritten, aufgezeigt am Beispiel des Knotenintervalles der orange unterlegten Kante aus Abbildung 3.4:

$$
\begin{aligned}
n = 0: &\qquad g_3 \\
n = 1: &\qquad \frac{d_3 + g_3}{2} = d_3 \cdot \frac{1}{2} + g_3 \cdot \frac{1}{2} &&= d_3 + \frac{1}{2^1} \cdot (g_3 - d_3) \\
n = 2: &\qquad \frac{d_3 + \frac{d_3 + g_3}{2}}{2} = d_3 \cdot \left(\frac{1}{2} + \frac{1}{4}\right) + g_3 \cdot \frac{1}{4} &&= d_3 + \frac{1}{2^2} \cdot (g_3 - d_3) \\
n = 3: &\qquad \frac{d_3 + \frac{d_3 + \frac{d_3 + g_3}{2}}{2}}{2} = d_3 \cdot \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right) + g_3 \cdot \frac{1}{8} &&= d_3 + \frac{1}{2^3} \cdot (g_3 - d_3) \\
&\vdots \\
n: &\qquad \frac{d_3 + \ldots \frac{d_3 + \frac{d_3 + \frac{d_3 + g_3}{2}}{2}}{2}}{2} = d_3 \cdot \sum_{i=1}^{n} \frac{1}{2^i} + g_3 \cdot \frac{1}{2^n} &&= d_3 + \frac{1}{2^n} \cdot (g_3 - d_3)
\end{aligned}
$$

Nach n Unterteilungsschritten erhält man durch wiederholtes Mitteln in der 1-Nachbarschaft von P_0^n die Knotenintervalle

$$
\begin{aligned}
&d_3 + \frac{1}{2^n}(b_3 - d_3), \qquad e_2 + \frac{1}{2^n}(a_2 - e_2), \qquad e_3 + \frac{1}{2^n}(a_3 - e_3), \qquad d_3 + \frac{1}{2^n}(g_3 - d_3), \\
&d_2 + \frac{1}{2^n}(g_2 - d_2), \qquad e_3 + \frac{1}{2^n}(c_3 - e_3), \qquad e_2 + \frac{1}{2^n}(c_2 - e_2), \qquad d_2 + \frac{1}{2^n}(b_2 - d_2).
\end{aligned}
$$

Die Knotenintervallvektoren in der 1–Nachbarschaft von P_0 konvergieren somit mit exponentieller Geschwindigkeit zu den beiden lokalen Knotenintervallvektoren von P_0.

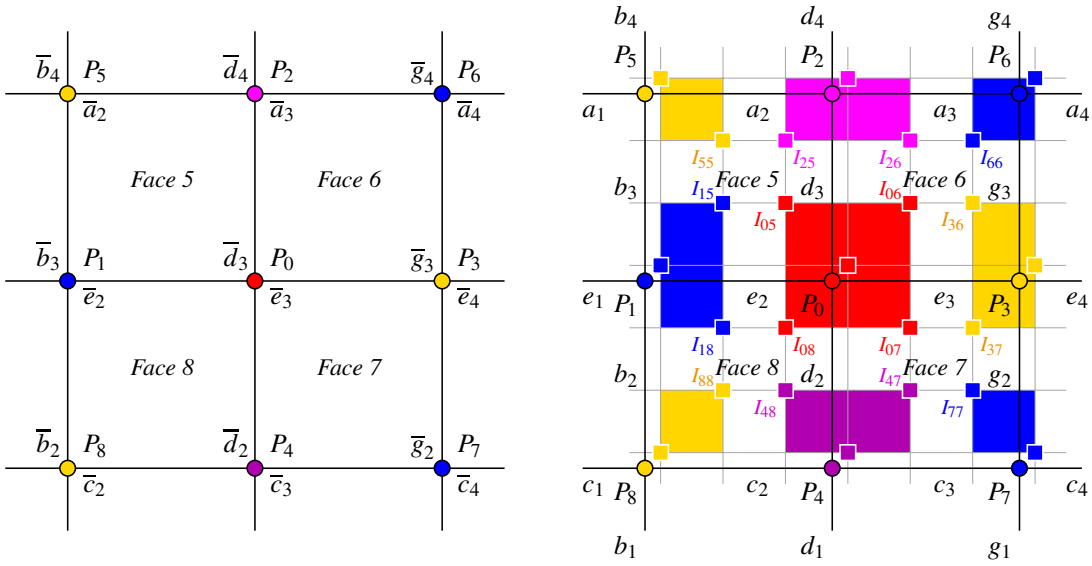


Abbildung 3.5: Aus den Knotenintervallen lassen sich die lokalen Knotenvektoren bestimmen (links). Für P_0 ist sein vertikaler Knotenvektor durch $\overline{K}_0^v = \{\overline{d}_1 = 0, \overline{d}_2, \overline{d}_3, \overline{d}_4, \overline{d}_5\}$ gegeben und sein horizontaler Knotenvektor durch $\overline{K}_0^h = \{\overline{e}_1 = 0, \overline{e}_2, \overline{e}_3, \overline{e}_4, \overline{e}_5\}$ definiert. Mit Hilfe dieser lokalen Knotenvektoren lassen sich die entsprechenden lokalen Bézier-Kontrollpunkte berechnen (rechts).

Die Herleitung des Regelsatzes für reguläre Topologien erfolgt analog zu den NURBS-Unterteilungsregeln. Dazu wird lokal von einer NURBS-Fläche ausgegangen, welche durch einen Punkt P_i, seinen beiden Knotenintervallvektoren K_i^v, K_i^h und seiner 1–Nachbarschaft definiert ist. Ebenso wie bei den NURBS-Unterteilungsflächen werden zunächst die lokalen Bézier-Kontrollpunkte berechnet, auf deren Grundlage die Unterteilungsregeln aufgebaut sind.

Die inneren Bézier-Kontrollpunkte I_{ij} rund um einen Punkt P_i werden aus den beiden Knotenintervallvektoren K_i^v, K_i^h von P_i und den Eckpunkten des Faces j bestimmt. Damit lässt sich I_{05} (siehe Abbildung 3.5) aus den beiden Knotenintervallvektoren $K_0^v = \{d_1, d_2, d_3, d_4\}, K_0^h = \{e_1, e_2, e_3, e_4\}$ des Punktes P_0 und den Eckpunkten des Faces 5 mit Hilfe des lokalen Blossoms b wie folgt berechnen:

$$\begin{aligned} I_{05} &= b[(\overline{e}_3, \overline{d}_3), (\overline{e}_3, \overline{d}_3), (\overline{e}_2, \overline{d}_4)] \\ &= \frac{(e_1+e_2)\cdot(d_3+d_4)\cdot P_0 + e_3\cdot(d_3+d_4)\cdot P_1 + e_3\cdot d_2\cdot P_5 + (e_1+e_2)\cdot d_2\cdot P_2}{(e_2+e_3+e_4)\cdot(d_2+d_3+d_4)} \end{aligned} \tag{3.1}$$

Die weiteren lokalen inneren Bézier-Kontrollpunkte des Faces 5 lassen sich analog in Abhängigkeit ihrer jeweiligen Knotenintervallvektoren ermitteln:

$$\begin{aligned} I_{15} &= b[(\overline{e}_2, \overline{b}_3), (\overline{e}_2, \overline{b}_3), (\overline{e}_3, \overline{b}_4)] \\ &= \frac{(e_2+e_3)\cdot(b_3+b_4)\cdot P_1 + (e_2+e_3)\cdot b_2\cdot P_5 + e_1\cdot b_2\cdot P_2 + e_1\cdot(b_3+b_4)\cdot P_0}{(e_1+e_2+e_3)\cdot(b_2+b_3+b_4)} \end{aligned} \tag{3.2}$$

$$\begin{aligned} I_{55} &= b[(\overline{a}_2, \overline{b}_4), (\overline{a}_2, \overline{b}_4), (\overline{a}_3, \overline{b}_3)] \\ &= \frac{(a_2+a_3)\cdot(b_3+b_2)\cdot P_5 + a_1\cdot(b_2+b_3)\cdot P_2 + a_1\cdot b_4\cdot P_0 + (a_2+a_3)\cdot b_4\cdot P_1}{(a_1+a_2+a_3)\cdot(b_2+b_3+b_4)} \end{aligned} \tag{3.3}$$

$$\begin{aligned} I_{25} &= b[(\overline{a}_3, \overline{d}_4), (\overline{a}_3, \overline{d}_4), (\overline{a}_2, \overline{d}_3)] \\ &= \frac{(a_1+a_2)\cdot(d_2+d_3)\cdot P_2 + (a_1+a_2)\cdot d_4\cdot P_0 + a_3\cdot d_4\cdot P_1 + a_3\cdot(d_2+d_3)\cdot P_5}{(a_1+a_2+a_3)\cdot(d_2+d_3+d_4)} \end{aligned} \tag{3.4}$$

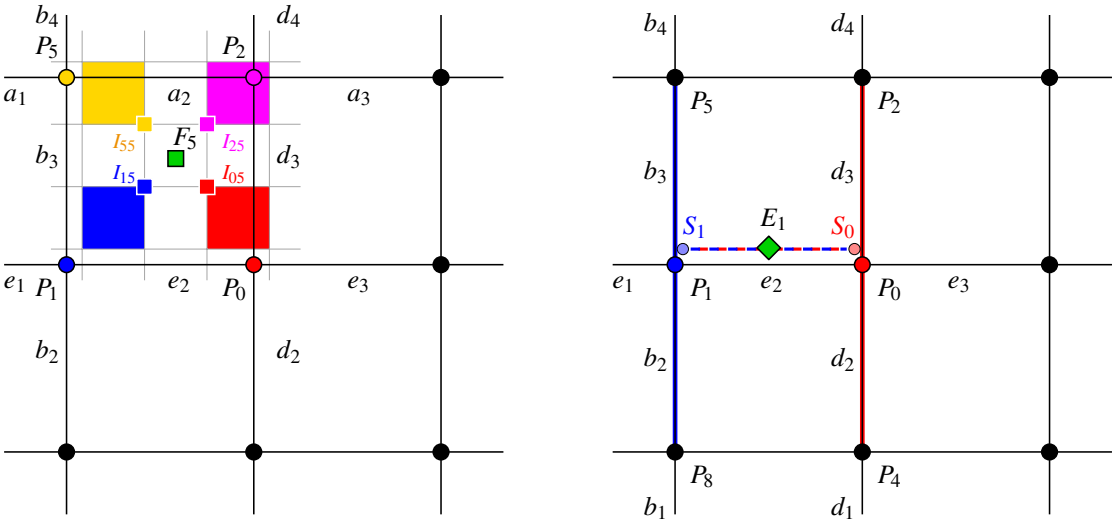


Abbildung 3.6: Neuer Facepunkt, berechnet aus dem gewichteten Mittel der vier lokalen Bézier-Kontrollpunkte (links). Neuer Kantenpunkt, ermittelt durch lokale Unterteilung von Kurvensegmenten (rechts): Unterteilung der Kurve $\langle P_8, P_1, P_5 \rangle$ (blau unterlegt) und der Kurve $\langle P_4, P_0, P_2 \rangle$ (rot markiert) mit den jeweils angegebenen Knotenintervallen ergibt die Hilfspunkte S_1, S_0. Die lokale Unterteilung von S_1 und S_0 liefert den gesuchten neuen Kantenpunkt.

Mit Hilfe dieser lokalen inneren Bézier-Kontrollpunkte I_{ij} erhält man den neuen Facepunkt auf gleiche Weise wie bei den NURBS-Unterteilungsflächen beschrieben (siehe auch Formel 2.17):

$$F_5 = \frac{1}{4} \cdot (I_{05} + I_{15} + I_{55} + I_{25}) \tag{3.5}$$

Der Unterschied zur NURBS-Unterteilung besteht lediglich in den lokal unterschiedlichen Knotenintervallvektoren, woraus unterschiedliche innere Bézier-Kontrollpunkte aus den lokal definierten NURBS-Flächen resultieren. Werden die gleichen Knotenintervalle wie bei einer NURBS-Fläche eingesetzt, so erhält man über die Faceregeln der ESubs die gleichen neuen Facepunkte wie bei den NURBS-Unterteilungsflächen. Ebenso reduzieren sich die Faceregeln der ESubs bei uniformer Knotenintervallwahl zu den Catmull-Clark-Faceregeln.

Zur Berechnung eines neuen Kantenpunktes sind zwei Hilfspunkte S_0, S_1 nötig: S_1 ist der neue Vertexpunkt der Kurve $\langle P_8, P_1, P_5 \rangle$ mit dem lokalen Knotenintervallvektor K_1^v von P_1.

$$\begin{aligned} S_1 &= b[\tfrac{\overline{b}_2+\overline{b}_3}{2}, \overline{b}_3, \tfrac{\overline{b}_3+\overline{b}_4}{2}] \\ &= \text{subdivCurveVertex}(P_8, P_1, P_5, b_1, b_2, b_3, b_4) \end{aligned} \tag{3.6}$$

Weiterhin sei S_0 der neue Vertexpunkt der Kurve $\langle P_4, P_0, P_2 \rangle$ mit dem lokalen Knotenintervallvektor K_0^v von P_0.

$$\begin{aligned} S_0 &= b[\tfrac{\overline{d}_2+\overline{d}_3}{2}, \overline{d}_3, \tfrac{\overline{d}_3+\overline{d}_4}{2}] \\ &= \text{subdivCurveVertex}(P_4, P_0, P_2, d_1, d_2, d_3, d_4) \end{aligned} \tag{3.7}$$

Dann erhält man den neuen Kantenpunkt E_1 ebenso wie bei den NURBS-Unterteilungsflächen durch:

$$E_1 = \text{subdivCurveEdge}(S_1, S_0, e_1, e_2, e_3) \tag{3.8}$$

Der Unterschied zu NURBS-Unterteilungsflächen zeigt sich in den unterschiedlichen Knotenvektoren zur Berechnung von S_0 und S_1. Werden wie bei NURBS-Unterteilungsflächen gefordert gleiche Knotenvektoren benutzt, so reduzieren sich die ESubs-Kantenregeln zu NURBS-Kantenregeln. Aus der Wahl uniformer Knotenintervalle bei ESubs resultieren wiederum die Kantenregeln für Catmull-Clark.

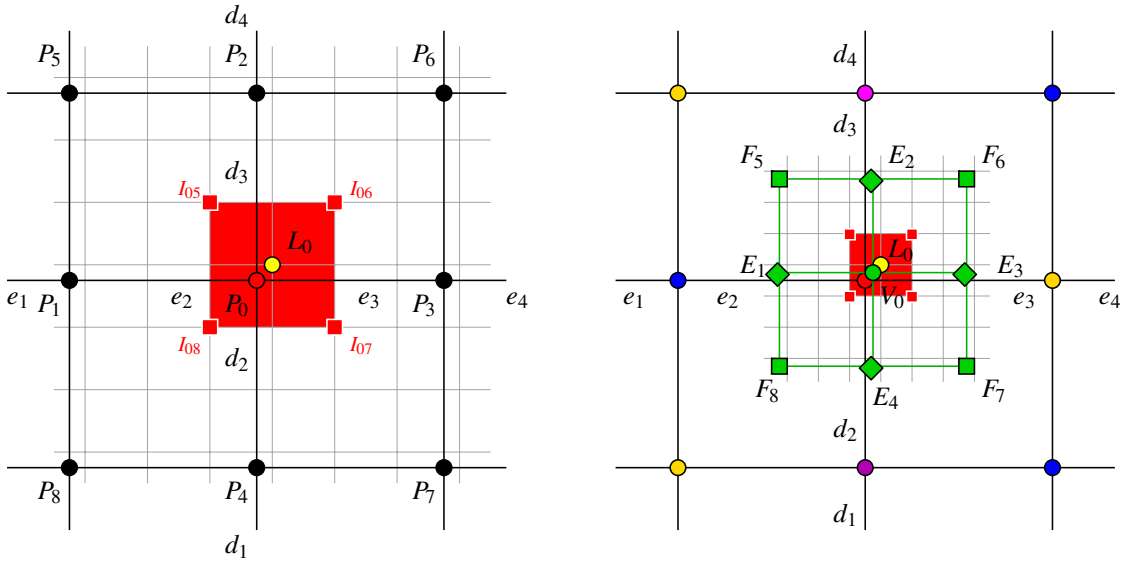


Abbildung 3.7: Links: Limitpunkt von P_0, bestimmt aus den lokalen Bézier-Kontrollpunkte von P_0. Rechts: neuer Vertexpunkt von P_0, ermittelt aus der unterteilten 1–Nachbarschaft und seinem Limitpunkt.

Um den Regelsatz für ESubs mit regulärer Topologie zu komplettieren sind die Vertex- sowie Limitpunktregeln zu ergänzen. Daher werden zunächst die Limitpunktregeln am Beispiel des Kontrollpunktes P_0 aus Abbildung 3.7 hergeleitet. Bei den ESubs wird lokal von einer NURBS-Fläche ausgegangen, welche durch P_0, seinen beiden Knotenvektoren $\overline{K}_0^h = \{\overline{e}_1 = 0, \overline{e}_2, \overline{e}_3, \overline{e}_4\}$, $\overline{K}_0^v = \{\overline{d}_1 = 0, \overline{d}_2, \overline{d}_3, \overline{d}_4\}$, und seiner 1–Nachbarschaft $P_1 \dots P_8$ definiert ist. Dadurch sind die Limitpunktregeln von P_0 mit Hilfe der lokalen NURBS-Fläche gegeben durch das Blossom:

$$\begin{aligned} L_{P_0} &= b[(\overline{e}_3, \overline{d}_3), (\overline{e}_3, \overline{d}_3), (\overline{e}_3, \overline{d}_3)] \\ &= \frac{e_3 \cdot d_2 \cdot I_{05} + e_2 \cdot d_2 \cdot I_{06} + e_2 \cdot d_3 \cdot I_{07} + e_3 \cdot d_3 \cdot I_{08}}{(e_2 + e_3) \cdot (d_2 + d_3)} \\ &=: \text{limitPoint}(P_0, \dots, P_8, e_1, \dots, e_4, d_1, \dots, d_4) \end{aligned} \tag{3.9}$$

Formel (3.9) gibt somit die Limitpunktregel für ESubs an. Herzuleiten ist nun eine entsprechende Vertexregel, so dass ein unendliches Unterteilen von P_0 zu L_{P_0} aus Formel (3.9) führt. Nach einem Unterteilungsschritt des Teil-Kontrollnetzes $P_0 \dots P_8$ liegen die neuen Kantenpunkte $E_1 \dots E_4$, die neuen Facepunkte $F_5 \dots F_8$ und der noch zu bestimmende neue Vertexpunkt V_0 vor. Punkt V_0 hat die Knotenintervallvektoren $\{d_2, d_2, d_3, d_3\}$, $\{e_2, e_2, e_3, e_3\}$ und die lokalen Knotenvektoren

$$\{\overline{d}_2, \frac{\overline{d}_2 + \overline{d}_3}{2}, \overline{d}_3, \frac{\overline{d}_3 + \overline{d}_4}{2}, \overline{d}_4\}, \qquad \{\overline{e}_2, \frac{\overline{e}_2 + \overline{e}_3}{2}, \overline{e}_3, \frac{\overline{e}_3 + \overline{e}_4}{2}, \overline{e}_4\}.$$

Der Limitpunkt von V_0 lässt sich ebenso wie der Limitpunkt von P_0 berechnen, unter Berücksichtigung der 1-Nachbarschaft von V_0, seiner lokalen Knotenintervallvektoren und seinem entsprechenden Blossom:

$$\begin{aligned} L_{V_0} &= b[(\overline{e}_3, \overline{d}_3), (\overline{e}_3, \overline{d}_3), (\overline{e}_3, \overline{d}_3)] \\ &= \text{limitPoint}(V_0, E_1, \dots, E_4, F_5, \dots, F_8, e_2, e_2, e_3, e_3, d_2, d_2, d_3, d_3) \end{aligned}$$

Durch Ausformulierung des Blossoms und Umordnung nach $V_0, E_1, \ldots, E_4, F_5, \ldots, F_8$ ergeben sich die Gewichte β_i, $i = 0, \ldots, 8$, für die Kanten- und Facepunkte sowie für den Vertexpunkt zur Berechnung von L:

$$L_{V_0} = \beta_0 \cdot V_0 + \sum_{i=1}^{4} \beta_i \cdot E_i + \sum_{i=5}^{8} \beta_i \cdot F_i \tag{3.10}$$

mit:

$$\begin{aligned}
\beta_0 &= \frac{36 \cdot d2 \cdot e3 \cdot d3 \cdot e2}{(2 \cdot e2 + e3) \cdot (2 \cdot d3 + d2) \cdot (e2 + 2 \cdot e3) \cdot (2 \cdot d2 + d3)} \\
\beta_1 &= \frac{6 \cdot d2 \cdot e3 \cdot e3 \cdot d3}{(e2 + e3) \cdot (2 \cdot e2 + e3) \cdot (2 \cdot d3 + d2) \cdot (2 \cdot d2 + d3)} \\
\beta_2 &= \frac{6 \cdot d2 \cdot d2 \cdot e3 \cdot e2}{(d3 + d2) \cdot (e2 + 2 \cdot e3) \cdot (2 \cdot e2 + e3) \cdot (2 \cdot d3 + d2)} \\
\beta_3 &= \frac{6 \cdot d2 \cdot e2 \cdot e2 \cdot d3}{(e2 + e3) \cdot (2 \cdot d3 + d2) \cdot (e2 + 2 \cdot e3) \cdot (2 \cdot d2 + d3)} \\
\beta_4 &= \frac{6 \cdot e3 \cdot d3 \cdot d3 \cdot e2}{(d3 + d2) \cdot (2 \cdot d2 + d3) \cdot (2 \cdot e2 + e3) \cdot (e2 + 2 \cdot e3)} \\
\beta_5 &= \frac{d2 \cdot d2 \cdot e3 \cdot e3}{(e3 + 2 \cdot e2) \cdot (2 \cdot d3 + d2) \cdot (e2 + e3) \cdot (d3 + d2)} \\
\beta_6 &= \frac{d2 \cdot d2 \cdot e2 \cdot e2}{(2 \cdot d3 + d2) \cdot (2 \cdot e3 + e2) \cdot (e2 + e3) \cdot (d3 + d2)} \\
\beta_7 &= \frac{d3 \cdot d3 \cdot e2 \cdot e2}{(2 \cdot e3 + e2) \cdot (2 \cdot d2 + d3) \cdot (e2 + e3) \cdot (d3 + d2)} \\
\beta_8 &= \frac{e3 \cdot e3 \cdot d3 \cdot d3}{(2 \cdot d2 + d3) \cdot (e3 + 2 \cdot e2) \cdot (e2 + e3) \cdot (d3 + d2)}
\end{aligned}$$

Der Limitpunkt und die Limitpunktberechnung sind unabhängig von der gewählten Kontrollnetztiefe, d.h. der Limitpunkt L_{P_0} von P_0 ist gleich dem Limitpunkt L_{V_0} von V_0:

$$\begin{aligned}
L = L_{P_0} &= L_{V_0} \\
&= \beta_0 \cdot V_0 + \sum_{i=1}^{4} \beta_i \cdot E_i + \sum_{i=5}^{8} \beta_i \cdot F_i
\end{aligned}$$

Durch Auflösen nach V_0 erhält man schließlich die gesuchte Vertexregel:

$$V_0 = \frac{1}{\beta_0} \cdot \left(L - \sum_{i=1}^{4} \beta_i \cdot E_i - \sum_{i=5}^{8} \beta_i \cdot F_i\right) \tag{3.11}$$

Vertex V_0 konvergiert mit zunehmender Unterteilungstiefe zum Limitpunkt L, welches in Abschnitt 3.4 bewiesen wird.

Der Unterschied zur NURBS-Unterteilung besteht in den lokal unterschiedlichen Knotenintervallvektoren von P_0 bis P_8, wodurch unterschiedliche neue Face- und Kantenpunkte berechnet werden. Werden die gleichen Knotenintervalle wie bei einer NURBS-Fläche eingesetzt, so erhält man gleiche Face- sowie Kantenpunkte und die ESubs-Vertexregel reduziert sich auf die gleiche Vertex-Unterteilungsregel wie bei den NURBS-Unterteilungsflächen. Ebenso erhält man bei uniformer Knotenintervallwahl die Catmull-Clark-Vertexregeln.

3.3.2 Irreguläre Topologie

Während die Anzahl der regulären Punkte beim Unterteilen um Faktor vier wächst, bleibt die Anzahl der irregulären Punkte nach einem Unterteilungsschritt konstant. Für diese konstante Anzahl von irregulären Punkten sowie ihrer 1–Nachbarschaft wird der Catmull-Clark-Regelsatz verwendet. Der Catmull-Clark-Vertexpunkt und seine 1–Nachbarschaft konvergieren für Unterteilungstiefe $n \to \infty$ gegen den Limitpunkt des Catmull-Clark-Vertexpunktes [ZS99]. Beim Unterteilen verkleinert sich somit der Bereich des Catmull-Clark-Einflussgebietes (siehe Abbildung 3.8).

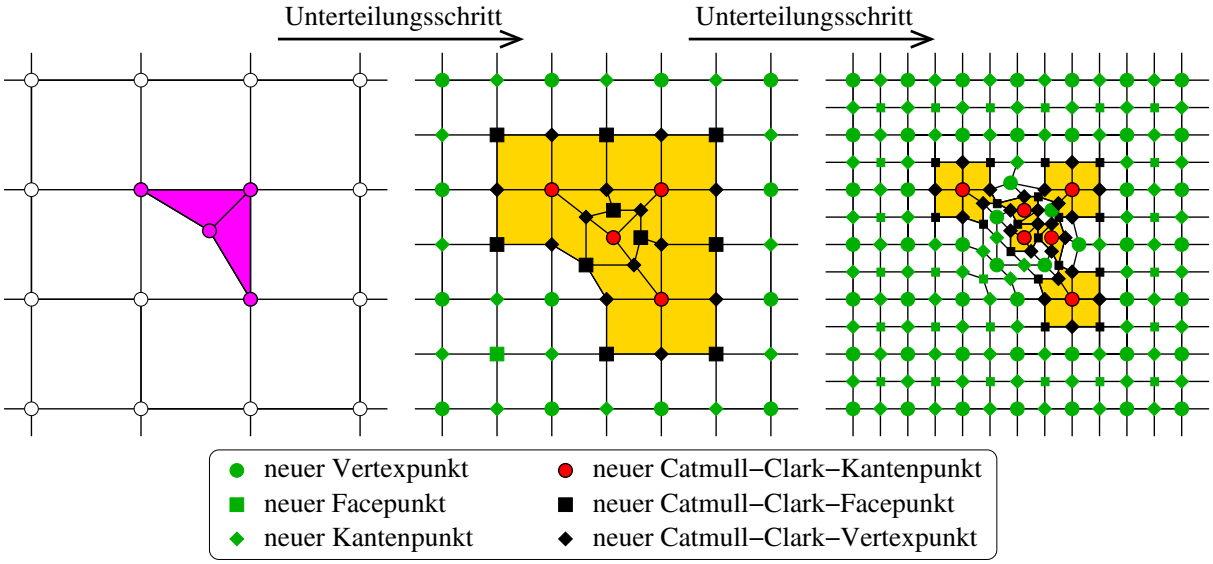


Abbildung 3.8: Catmull-Clark-Bereiche bei den ESubs. Links: Kontrollnetz M^0 der ESubs, irreguläre Bereiche sind magenta markiert. Mitte: Nach einem Unterteilungsschritt, die mit Catmull-Clark-Regeln entstandenen irregulären Gebiete inklusive ihrer 1–Nachbarschaft sind gelb markiert. Rechts: Die Catmull-Clark-Gebiete schrumpfen mit zunehmender Unterteilungstiefe.

Die Knotenintervallzuweisung beim Unterteilen erfolgt in den irregulären Gebieten wie in Abbildung 3.9 beschrieben. Die neu entstandenen regulären Vertices aus den ehemals Catmull-Clark-Gebieten werden mit dem regulären Regelsatz der ESubs unter Verwendung der jeweiligen lokalen Knotenvektoren weiter behandelt. Dabei ist zu beachten, dass bei der Bestimmung des Knotenintervallvektors eines regulären Punktes mit Kante zu einem irregulären Punkt ein Sonderfall auftritt. Unter Verwendung der Notation des vorherigen Abschnittes 3.3.1 sei beispielsweise P_0 ein regulärer Punkt und P_3 ein irregulärer Nachbar. Dann ist der horizontale Knotenvektor von P_0 nicht vollständig bestimmbar, da e_4 nicht eindeutig zu wählen ist. Daher wird in diesem Falle der horizontale Knotenintervallvektor von P_0 festgelegt

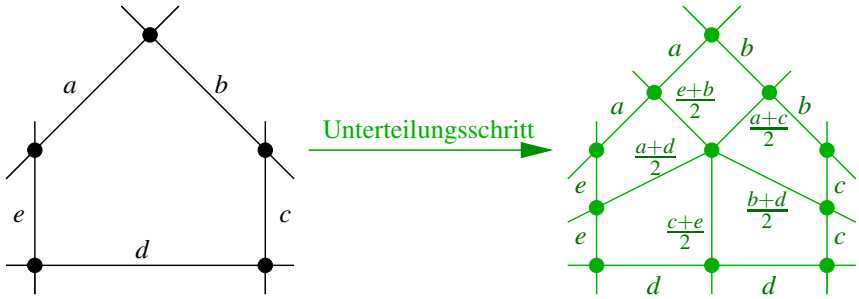


Abbildung 3.9: Knotenintervallzuweisung nach Unterteilung bei n-Eck.

als $\{e_1, e_2, e_3, e_3\}$, d.h. das Knotenintervall zwischen regulärem und irregulärem Punkt wird zweifach im Knotenintervallvektor gesetzt. In Abschnitt 3.5 wird die Wahl dieses Knotenintervallvektors untersucht und verdeutlicht.

Die Punkte in der 1–Nachbarschaft eines irregulären Punktes sind durch Catmull-Clark-Regeln entstanden. Beim nächsten Unterteilungsschritt werden diese Nachbarschaftspunkte mit Hilfe der ESubs-Regeln unterteilt und gehen in den 2–Nachbarschaftsring des irregulären Punktes über. Somit werden für alle regulären Punkte die ESubs-Limitpunktregeln verwendet, für die irregulären Punkte stehen die Catmull-Clark-Limitpunktregeln zur Verfügung. Dies ist gültig, da die 1–Nachbarschaft eines irregulären Punktes stets ausschließlich aus Catmull-Clark-Punkten besteht. An den irregulären Punkten liegt somit lokal eine Catmull-Clark-Fläche vor, für die lokal alle Eigenschaften dieser Fläche gelten.

3.4 Konvergenz

Generell sind Unterteilungsflächen definiert als Grenzfläche einer Folge von Kontrollnetzverfeinerungen, welche aus der sukzessiven Anwendung der Unterteilungsregeln resultiert. Die Konvergenz einer Unterteilungsfläche gegen eine Limitfläche ist somit ein essentielles Kennzeichen dieses Flächentyps. Bei der Bildung einer neuen Unterteilungsfläche gilt es, diese Eigenschaft zu überprüfen.

Wie im Allgemeinen üblich, sind nur positive Knotenintervalle zulässig. Unter dieser Voraussetzung sind die ESubs-Limitpunktregeln sowie die Kanten- und Faceregeln Konvexkombinationen. Die ESubs-Vertexregeln können in Abhängigkeit der gewählten Knotenintervalle als Konvexkombination sowie als Nicht-Konvexkombinationen vorliegen. Eine Umordnung der Vertexregel (3.11) zu $V_0 = \sum_0^8 \alpha_i P_i$ nach der 1–Nachbarschaft P_i liefert die Gewichte für P_i zur Unterteilung.

Definition 7 (konvex-gültig). *Es sei P_i, $i = 1,\ldots,8$ die 1–Nachbarschaft von P_0 mit ihren dedizierten horizontalen und vertikalen Knotenintervallvektoren K_i^v, K_i^h. Weiterhin sei $V_0 = \sum_0^8 \alpha_i P_i$ die Unterteilungsregel für Vertex P_0, gebildet aus den Knotenintervallvektoren K_i^v, K_i^h, $i = 0,\ldots,8$.*

Die Knotenintervallvektoren in der 1–Nachbarschaft von P_0 heißen **konvex-gültig**, *wenn $\alpha_i \geq 0$ für alle $i = 0,\ldots,8$ gilt.*

Wenn konvex-gültige Knotenintervallvektoren vorliegen, dann ist die Vertexregel eine Konvexkombination. Nicht konvex-gültige Knotenintervallvektoren sind bei ESubs möglich, bilden aber einen Son-

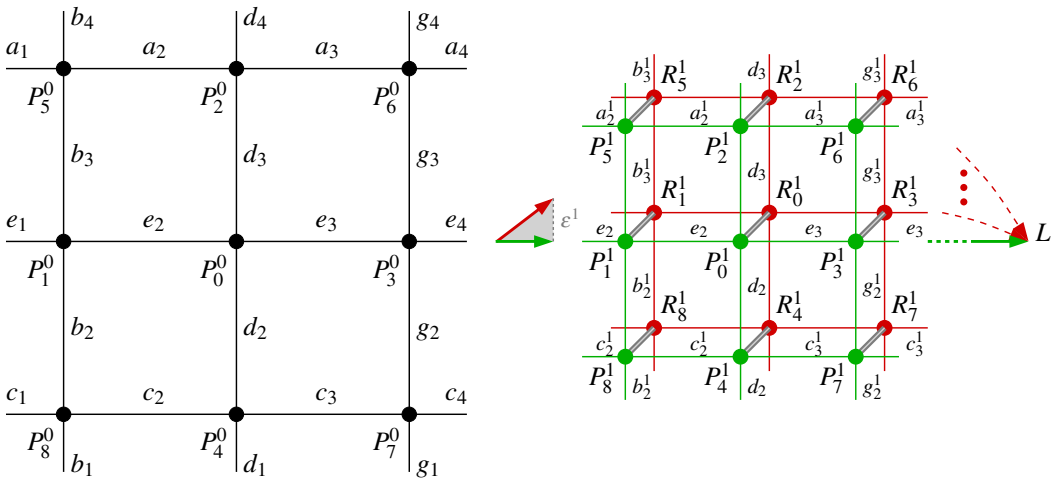


Abbildung 3.10: Aus den Punkten P_i^0, $i = 0,\ldots,8$ sind mit Hilfe des ESubs-Regelsatzes die Punkte P_i^1, $i = 0,\ldots,8$ (grün dargestellt) generiert worden. Unter Verwendung der NURBS-Unterteilungsregeln mit den Knotenvektoren von P_0^0 ergeben sich die Punkte R_i^1, $i = 0,\ldots,8$ (rot markiert).

derfall, da sie die Konvexe-Hülle-Eigenschaft verlieren. Sie sind daher als eine Art Special Feature anzusehen und werden am Ende des Abschnitts näher untersucht.

Im Folgenden wird von konvex-gültigen Knotenvektoren ausgegangen. Der komplette ESubs-Regelsatz liegt somit als Konvexkombinationen vor und besitzt daraus resultierend die Konvexe-Hülle-Eigenschaft.

Es sei P_0^0 ein Punkt des Ausgangskontrollnetzes M^0 und P_i^0, $i = 1,\ldots,8$ seine 1–Nachbarschaft. Punkt P_0^n bezeichnet den n-fach unterteilten Vertex P_0^0, die 1–Nachbarschaft von P_0^n in Tiefe n ist P_i^n, $i = 1,\ldots,8$. Der Notation aus Abbildung 3.10 folgend, sind die Knotenintervallvektoren von P_0^0 durch $K_0^v = \{d_1,d_2,d_3,d_4\}$ und $K_0^h = \{e_1,e_2,e_3,e_4\}$ gegeben. Nach einem Unterteilungsschritt bleiben die Knotenintervallvektoren von P_0^n für alle $n \geq 1$ konstant auf $K_0^v = \{d_2,d_2,d_3,d_3\}$ und $K_0^h = \{e_2,e_2,e_3,e_3\}$. Die Knotenintervallvektoren in der 1–Nachbarschaft von P_0^n unterscheiden sich durch ihre Zugehörigkeit zu Face- oder Kantenpunkten (siehe Abbildung 3.4 und 3.10). Für den neuen Kantenpunkt P_1^n und den neuen Facepunkt P_5^n sind die Knotenintervallvektoren für $n > 1$ beispielsweise

$$K_1^v = \{b_2^n,b_2^n,b_3^n,b_3^n\}, K_1^h = \{e_2,e_2,e_2,e_3\} \text{ und } K_5^v = \{b_2^n,b_3^n,b_3^n,b_3^n\}, K_5^h = \{a_2^n,a_2^n,a_2^n,a_3^n\} \tag{3.12}$$

mit:

$$\begin{aligned} b_2^n &= d_2 + \tfrac{1}{2^n}\cdot(b_2 - d_2), & \lim_{n\to\infty} b_2^n &= d_2 \\ b_3^n &= d_3 + \tfrac{1}{2^n}\cdot(b_3 - d_3), & \lim_{n\to\infty} b_3^n &= d_3 \\ a_2^n &= e_2 + \tfrac{1}{2^n}\cdot(a_2 - e_2), & \lim_{n\to\infty} a_2^n &= e_2 \\ a_3^n &= e_3 + \tfrac{1}{2^n}\cdot(a_3 - e_3), & \lim_{n\to\infty} a_3^n &= e_3 \end{aligned}$$

Mit $n \to \infty$ gilt somit für die Knotenintervallvektoren des Kantenpunktes P_1^n

$$\lim_{n\to\infty} K_1^v = \{d_2,d_2,d_3,d_3\} = K_0^v, \qquad \lim_{n\to\infty} K_1^h = \{e_2,e_2,e_2,e_3\} \tag{3.13}$$

und für die Knotenintervallvektoren des Facepunktes P_5^n

$$\lim_{n\to\infty} K_5^v = \{d_2,d_3,d_3,d_3\}, \qquad \lim_{n\to\infty} K_5^h = \{e_2,e_2,e_2,e_3\} \tag{3.14}$$

Die Konvergenz der Knotenintervallvektoren der übrigen Kantenpunkte P_2^n, P_3^n, P_4^n und Facepunkte P_6^n, P_7^n, P_8^n in der 1–Nachbarschaft von P_0^n verläuft analog. Damit ist direkt ersichtlich (siehe auch Abbildung 2.14):

Lemma 1. *Die Knotenintervallvektoren in der 1–Nachbarschaft von P_0^n einer erweiterten Unterteilungsfläche konvergieren mit $O(\frac{1}{2^n})$ zu NURBS-Knotenintervallvektoren.*

Beweis:
Aus Formel (3.12), (3.13) und (3.14) ergibt sich direkt der Beweis des Lemmas.

■

Aus Lemma 1 folgt unmittelbar, dass auch der ESubs-Regelsatz gegen den NURBS-Regelsatz konvergiert. Um zu überprüfen, ob P_0^n für $n \to \infty$ gegen den Punkt $L = \sum_{i=0}^{8} \beta_i \cdot P_i^n$, $\forall n \geq 0$ aus dem ESubs-Regelsatz konvergiert, ist Definition 8 nötig:

Definition 8 (Konvergenzradius). *Es sei P_0^n ein Punkt des Kontrollnetzes M^n und P_i^n, $i = 1,\ldots,8$ seine 1–Nachbarschaft. Punkt L von P_0^n ist definiert im ESubs-Regelsatz (Formel (3.10)) durch P_i^n, $i = 0,\ldots,8$ und K_0^v, K_0^h. Dann heißt*

$$D^n := \max_{i=0,\ldots,8} |L - P_i^n|$$

der **Konvergenzradius** *von Punkt P_0^n in Unterteilungstiefe n.*

Strebt der Konvergenzradius D^n für $n \to \infty$ gegen null, so konvergiert P_0^n für $n \to \infty$ gegen L. Dies ist der Fall, wenn eine Konstante $0 < c < 1$ existiert, so dass gilt:

$$\exists n_0 \in \mathbb{N}_0 \quad \forall n \geq n_0 : \quad D^{n+1} < c \cdot D^n$$

Damit lässt sich der Satz zur Konvergenz der ESubs formulieren:

Satz 3 (Satz zur Konvergenz der ESubs). *Es sei P_0^0 ein Punkt des Ausgangskontrollnetzes M^0 und P_i^0, $i = 1,\ldots,8$ seine 1–Nachbarschaft. Punkt P_0^n bezeichnet den n-fach unterteilten Vertex P_0^0, erhalten durch den ESubs-Regelsatz. Die 1–Nachbarschaft von P_0^n in Tiefe n, erzeugt durch den ESubs-Regelsatz, sei P_i^n, $i = 1,\ldots,8$. Es sei Punkt $L = \sum_{i=0}^{8} \beta_i \cdot P_i^n$ von P_0^n durch den ESubs-Regelsatz definiert und D^n sei der Konvergenzradius von P_0^n aus Definition 8.*
Dann gibt es eine Konstante c für die gilt:

$$\exists n_0 \in \mathbb{N}_0 \quad \forall n \geq n_0 : \quad D^{n+1} < c \cdot D^n \quad \textit{mit } 0 < c < 1 \tag{3.15}$$

und damit

$$L = \lim_{n \to \infty} P_0^n.$$

Beweis:
Die Knotenintervallvektoren in der 1–Nachbarschaft von P_0^n seien konvex-gültig. Der Fall nicht konvex-gültige Knotenintervallvektoren wird nachfolgend behandelt. Wegen den konvex-gültigen Knotenintervallvektoren sind die ESubs-Unterteilungsregeln Konvexkombinationen, so dass bereits gilt:

$$D^{n+1} \leq D^n$$

Für NURBS-Unterteilungsflächen existiert die in Formel (3.15) geforderte Konstante c mit $c_R < 1$.

Lemma 2. *Die Punkte, die mit den NURBS-Unterteilungsregeln aus P_i^n, $i = 0,\ldots,8$, $n \geq 0$, und den Knotenvektoren K_0^v, K_0^h von P_0^n generiert werden, seien R_i^{n+1}, $i = 0,\ldots,8$ (siehe Abbildung 3.10). Weitere m Unterteilungsschritte mit dem NURBS-Regelsatz auf R_i^{n+1} werden mit $(R_i^{n+1})^m$ bezeichnet. Punkt L ist wie in Satz 3 beschrieben definiert. Für $m \to \infty$ konvergiert $(R_i^{n+1})^m$ zu L und es gilt*

$$L = \sum_{i=0}^{8} \beta_i \cdot (R_i^{n+1})^m, \quad \forall n \geq 0 \textit{ und } m > 0$$

wobei die β_i, $i = 0,\ldots,8$ aus den Knotenintervallvektoren K_0^v, K_0^h von P_0^{n+1} gebildet werden. Die Knotenintervallvektoren K_0^v, K_0^h bleiben nach dem ersten Unterteilungschritt für alle weiteren Unterteilungen konstant. Somit erhält man für P_i^{n+1}, R_i^{n+1}, $(R_i^{n+1})^m$ mit $n \geq 0$ die gleichen β_i und es folgt:

$$L = \sum_{i=0}^{8} \beta_i \cdot P_i^{n+1} = \sum_{i=0}^{8} \beta_i \cdot R_i^{n+1} = \sum_{i=0}^{8} \beta_i \cdot (R_i^{n+1})^m, \quad \forall n \geq 0 \textit{ und } m > 0$$

Beweis:
Der Beweis des Lemmas folgt unmittelbar aus der Definition des NURBS- sowie ESubs-Regelsatzes (siehe auch Abbildung 3.11).

■

Um zu zeigen, dass die Konstante c aus Formel (3.15) existiert und somit P_0^n für $n \to \infty$ gegen L konvergiert, wird o. B. d. A. angenommen, dass L der Nullpunkt ist. Somit gilt

$$D^n := \max_{i=0,\ldots,8} |P_i^n| \quad \text{und} \quad |P_i^{n+1}| \leq D^{n+1},\ i = 0,\ldots,8$$

und für alle $i = 0,\ldots,8$ ist eine Konstante c mit $0 < c < 1$ zu finden, so dass

$$|P_i^{n+1}| < c \cdot D^n$$

erfüllt ist. Wegen $L = \mathbf{0}$,

$$L = \mathbf{0} = \sum_{j=0}^{8} \beta_j \cdot P_j^{n+1} = \beta_i \cdot P_i^{n+1} + \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot P_j^{n+1}$$

lässt sich $|P_i^{n+1}|$ schreiben als:

$$\begin{aligned} |P_i^{n+1}| &= \left| -\frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot P_j^{n+1} \right| \\ &= \left| \mathbf{0} - \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot P_j^{n+1} \right| \end{aligned}$$

Aus Lemma 2 folgt mit $\mathbf{0} = L = \sum_{j=0}^{8} \beta_j \cdot R_j^{n+1} = \frac{1}{\beta_i} \cdot \sum_{j=0}^{8} \beta_j \cdot R_j^{n+1}$:

$$\begin{aligned} &= \left| \frac{1}{\beta_i} \cdot \sum_{j=0}^{8} \beta_j \cdot R_j^{n+1} - \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot P_j^{n+1} \right| \\ &= \left| R_i^{n+1} + \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot (R_j^{n+1} - P_j^{n+1}) \right| \\ &\leq |R_i^{n+1}| + \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot |R_j^{n+1} - P_j^{n+1}| \end{aligned}$$

Da für NURBS-Unterteilungsflächen die geforderte Konstante c aus Formel (3.15) in Satz 3 durch $c_R < 1$ bereits existiert, gilt mit $D_R^n := \max_{i=0,\ldots,8} |L - R_i^n|$ analog zur Definition 8, sowie o. B. d. A. $L = \mathbf{0}$ und $|R_i^{n+1}| \leq D_R^{n+1} < c_R \cdot D^n$:

$$|P_i^{n+1}| < c_R \cdot D^n + \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j\neq i}}^{8} \beta_j \cdot |R_j^{n+1} - P_j^{n+1}|$$

Eine Abstandsabschätzung von $|R_j^{n+1} - P_j^{n+1}|$ liefert Lemma 3:

Lemma 3. *Es sei P_0^0 ein Punkt des Ausgangskontrollnetzes M^0 und P_i^0, $i = 1,\ldots,8$ seine 1–Nachbarschaft. Punkt P_0^n bezeichnet den n-fach unterteilten Vertex P_0^0, erhalten mit dem ESubs-Regelsatz. Die 1–Nachbarschaft von P_0^n in Tiefe n erzeugt mit dem ESubs-Regelsatz sei P_i^n, $i = 1,\ldots,8$. Die Punkte, die mit den NURBS-Unterteilungsregeln aus P_i^n, $i = 0,\ldots,8$, $n \geq 0$, und den Knotenvektoren K_0^v, K_0^h von P_0^n generiert werden, seien R_i^{n+1}, $i = 0,\ldots,8$. D^n sei der Konvergenzradius von P_0^n aus Definition 8.*

Dann ist $|R_i^{n+1} - P_i^{n+1}|$, $i = 0,\ldots,8$, $n \geq 0$, beschränkt und es existiert für jedes $i = 0,\ldots,8$ eine von der Unterteilungstiefe n unabhängige Konstante H_i, so dass gilt:

$$|R_i^{n+1} - P_i^{n+1}| \leq \frac{1}{2^n} \cdot H_i \cdot D^n$$

Beweis:
Der Beweis des Lemmas ist im Anhang A zu finden.

■

Somit ergibt sich für $|P_i^{n+1}|$:

$$\begin{aligned} |P_i^{n+1}| &< c_R \cdot D^n + \frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j \neq i}}^{8} \beta_j \cdot \frac{1}{2^n} \cdot H_j \cdot D^n \\ &= (c_R + \frac{1}{2^n} \cdot \underbrace{\frac{1}{\beta_i} \cdot \sum_{\substack{j=0 \\ j \neq i}}^{8} \beta_j \cdot H_j}_{=\gamma_i}) \cdot D^n \\ &= (c_R + \frac{1}{2^n} \cdot \gamma_i) \cdot D^n \end{aligned} \tag{3.16}$$

Wegen Lemma 3 und den ESubs-Limitpunktregeln folgt unmittelbar dass γ_i unabhängig von der Unterteilungstiefe n ist. Aus den γ_i, $i = 0, \ldots, 8$ ergibt sich γ_{max} mit:

$$\gamma_{max} := \max_{i=0,\ldots,8} (\gamma_i) \tag{3.17}$$

Somit erhält man:

$$\exists n_0 \quad \forall n \geq n_0 : \quad \frac{1}{2^n} \cdot \gamma_{max} < 1 - c_R \text{ und } c_R + \frac{1}{2^n} \cdot \gamma_{max} < 1$$

Damit lassen sich c^n und Konstante c bestimmen:

$$\begin{aligned} c^n &:= c_R + \frac{1}{2^n} \cdot \gamma_{max} \\ c &:= \max_{n \geq n_0} (c^n) \end{aligned}$$

Somit gilt:

$$\forall n \geq n_0 : \quad |P_i^{n+1}| < c^n \cdot D^n \text{ mit } c^n < 1$$

und

$$\forall n \geq n_0 : \quad |P_i^{n+1}| < c \cdot D^n \text{ mit } c < 1$$

Wegen $L = \mathbf{0}$ und $D^{n+1} := \max_{i=0,\ldots,8} |P_i^{n+1}|$ folgt unmittelbar

$$\forall n \geq n_0 : \quad D^{n+1} < c^n \cdot D^n \text{ mit } c^n < 1$$

wobei für $n \to \infty$ gilt: $c^n \to c_R$. Ebenso ist die Forderung aus Formel (3.15) von Satz 3 erfüllt mit

$$\exists n_0 \in \mathbb{N}_0 \quad \forall n \geq n_0 : \quad D^{n+1} < c \cdot D^n \text{ mit } 0 < c < 1$$

Daraus resultiert direkt die Konvergenz der ESubs bei Verwendung konvex-gültiger Knotenvektoren.

Auch im Falle nicht konvex-gültiger Knotenvektoren konvergiert die Fläche gegen den berechneten Limitpunkt L: Die nicht konvex-gültigen Knotenvektoren ergeben sich aus einer zu großen Differenz zwischen den Knotenintervalle auf gegenüberliegenden Faceseiten. Da die Knotenintervallvektoren in der 1–Nachbarschaft von P_0^n mit $O(\frac{1}{2^n})$ zu NURBS-Knotenintervallvektoren (siehe z.B. Abbildung 2.14) konvergieren, vermindert sich die Differenz und geht für $n \to \infty$ gegen Null. Damit existiert ein n_k, so dass für alle Unterteilungstiefen $n \geq n_k$ konvex–gültige Knotenvektoren vorliegen. Das Vorgehen zur Folgerung der Konvergenz bei konvex-gültigen Knotenvektoren kann somit bei nicht konvex-gültigen Knotenvektoren nach n_k Unterteilungsschritten angewendet werden.
Der Satz zur Konvergenz der ESubs ist somit für positive Knotenintervalle bewiesen.

■

Korollar 1. *Die Konstante c aus Formel (3.15) lässt sich in Abhängigkeit der Unterteilungstiefe n ausdrücken durch*

$$c^n = c_R + \frac{1}{2^n} \cdot \gamma_{max}$$

wobei c_R die NURBS-Konstante aus Formel (3.15) der NURBS-Unterteilungsflächen ist. γ_{max} ist definiert in Formel (3.17).
Die Konstanten c_R und γ_{max} sind unabhängig von der Unterteilungstiefe n. Somit gilt:

$$\exists n_0 \in \mathbb{N}_0 \quad \forall n \geq n_0 : \quad D^{n+1} < c^n \cdot D^n \; mit \; 0 < c^n < 1$$

und

$$c_R = \lim_{n \to \infty} c^n.$$

Beweis:
Der Beweis des Korollars ergibt sich direkt aus dem Beweis des Satzes 3. ■

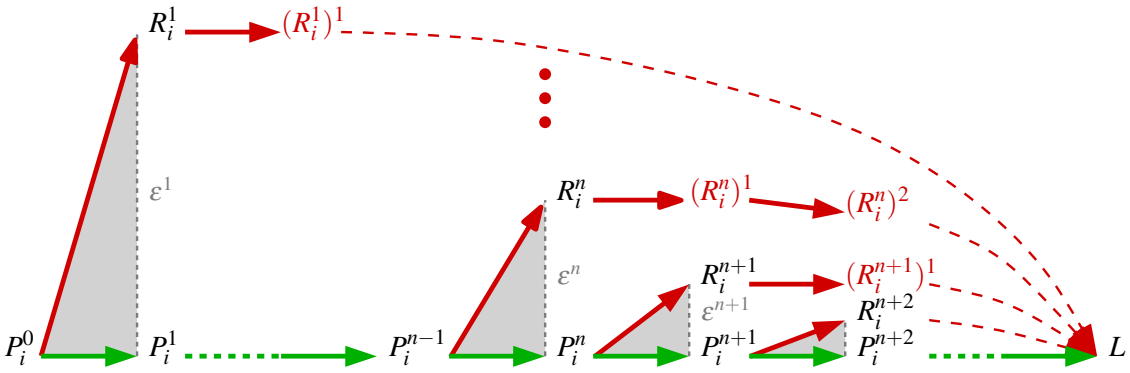


Abbildung 3.11: Konvergenz von P_i^n, R_i^n und $(R_i^n)^m$ nach L.

3.5 Stetigkeit

Zur Reduktion der Fallanzahl wird die Stetigkeitsbetrachtung der ESubs nach einem Unterteilungsschritt durchgeführt, wenn nur noch Vierecke im Kontrollnetz vorliegen. Der Grad der Stetigkeit hängt von der Vertexvalenz sowie von den Knotenintervallen ab. Null-Knotenintervalle und Special Features führen zu einer gewünschten Reduktion der Stetigkeit und werden im nachfolgenden Abschnitt 3.6 untersucht. Im Folgenden wird von positiven Knotenintervallen und Flächen ohne Special Features ausgegangen.

Irreguläre Vertices mit Valenz ungleich vier sind lokale Catmull-Clark-Punkte und besitzen demzufolge C^1-Stetigkeit [ZS99]. Faces mit regulärer Topologie und konformen Faces in ihrer 1–Nachbarschaft definieren lokal eine bikubische NURBS-Fläche. Somit liegt bei diesen Patches eine C^2-Stetigkeit vor. Die verbleibenden Gebiete haben reguläre Topologie mit nicht-konformen Faces und bedürfen einer eingehenderen Betrachtung.

Abbildung 3.12 zeigt die Knotenintervalle eines nicht-konformen Faces nach zwei sowie drei Unterteilungsschritten. Die Vertices im Inneren des nicht-konformen Faces, außerhalb der 1–Nachbarschaft des Randes, besitzen lokal uniforme Knotenvektoren. In Unterteilungstiefe zwei gilt dies für einen Vertex, in Tiefe drei besitzen 25 Vertices einen uniformen Knotenvektor. Die Face-, Kanten- und Vertexregeln der ESubs reduzieren sich bei einer solchen uniformen Knotenvektorenstruktur auf die Unterteilungsregeln einer uniformen B-Spline-Fläche. Mit zunehmender Unterteilungstiefe wächst das Gebiet im Inneren des nicht-konformen Faces mit lokal uniformen Knotenvektoren, wodurch sich das Gebiet der

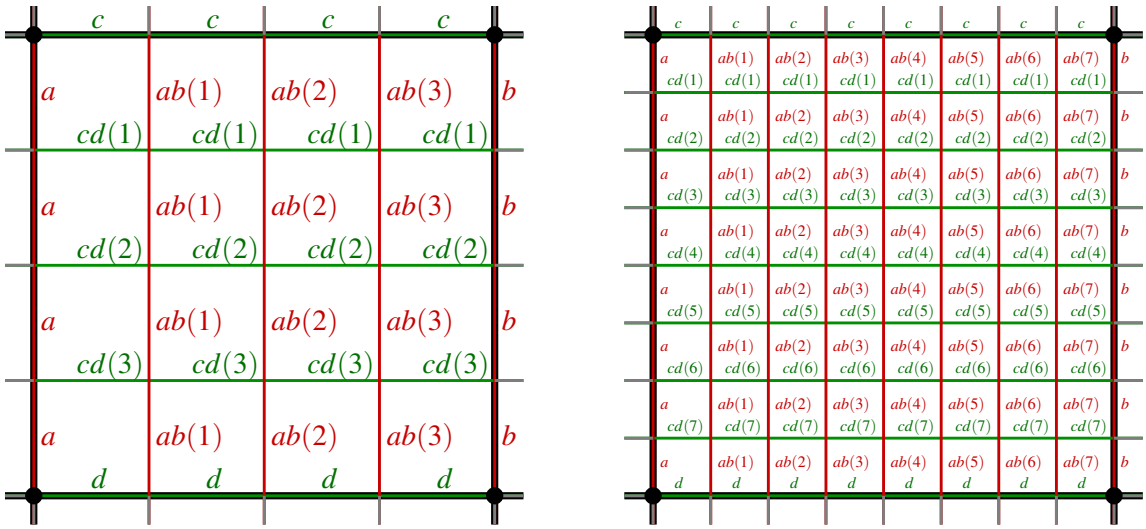


Abbildung 3.12: Knotenintervalle in einem nicht-konformen Face nach $n = 2$ (links) und $n = 3$ (rechts) Unterteilungsschritten mit $ab(i) = \frac{i}{2^n} \cdot a + \frac{2^n - i}{2^n} \cdot b$, $cd(i) = \frac{i}{2^n} \cdot c + \frac{2^n - i}{2^n} \cdot d$, $1 \leq i < 2^n$.

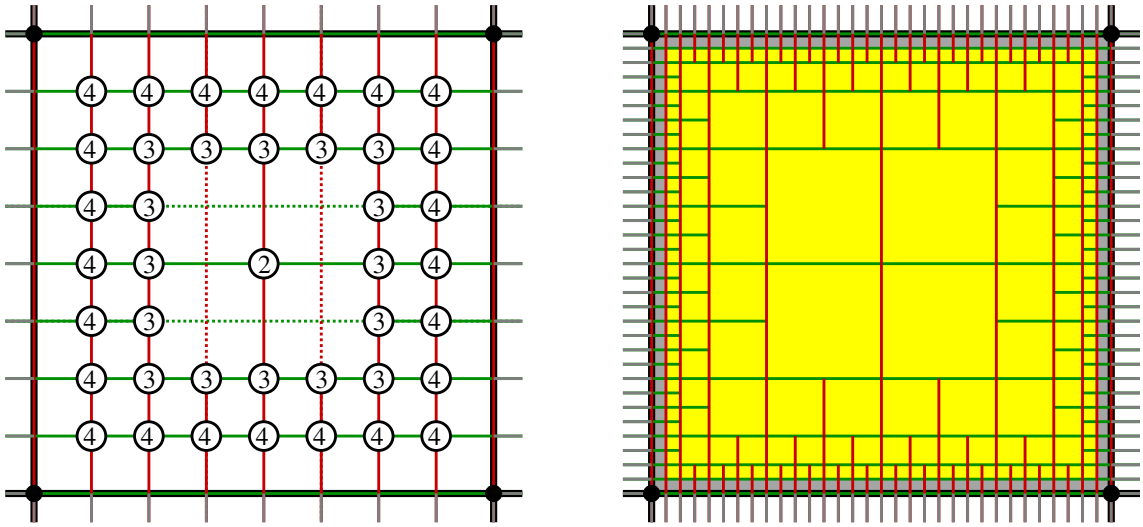


Abbildung 3.13: Die lokalen Knotenvektoren im Inneren des nicht-konformen Faces werden in der angegebenen Unterteilungstiefe uniform (links). Der ESubs-Regelsatz reduziert sich bei einer solchen uniformen Knotenintervallstruktur auf die uniformen B-Spline-Unterteilungs- und Limitpunktregeln. Die Folge der resultierenden uniformen B-Spline-Patches (gelb markiert) konvergiert gegen den Rand des nicht-konformen Faces (rechts).

uniformen B-Spline-Fläche vergrößert (siehe Abbildung 3.13). Für Unterteilungstiefe $n \to \infty$ konvergiert die uniforme B-Spline-Fläche im Inneren des nicht-konformen Faces gegen ein offenes Flächenpatch. Somit ist das offene innere Gebiet eines nicht-konformen Face-Patches eine uniforme B-Spline-Fläche mit C^2-Stetigkeit.

Es verbleibt noch den Rand des nicht-konformen Faces zu untersuchen. Ebenso wie bei den NURSS-Flächen [SSS98] von Sederberg et. al. liegt an den Rändern der nicht-konformen Faces der ESubs ein nicht-stationäres, nicht-uniformes Unterteilungsschema vor. Aus diesem Grunde ist es schwierig, eine Eigenanalysis durchzuführen um die Ordnung der Stetigkeit herauszufinden. Analog zu Sederberg et. al. wird ein typisches Beispiel mit uniformer, nicht-uniformer und nicht-konformer Knotenintervall-Konfiguration bezüglich eines Vertex V_0 untersucht (siehe Abbildung 3.14).

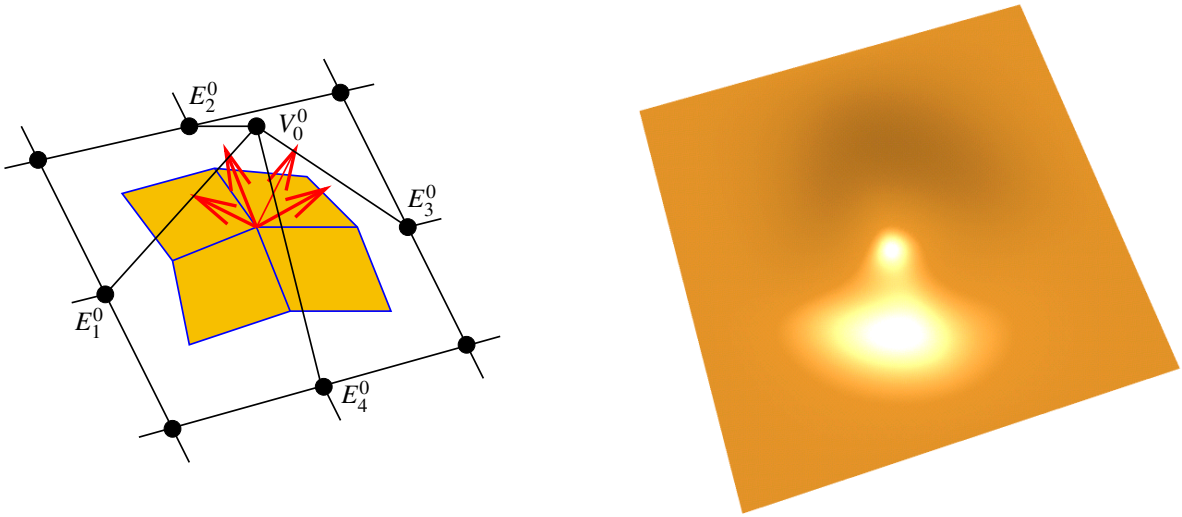


Abbildung 3.14: Schematische Darstellung des zur Messung benutzten Modells (links) und resultierendes Bild der nicht-konformen Face-Konfiguration (rechts).

Tiefe	uniforme Knotenintervalle	nicht-uniform	nicht-konform
1	0.31433856261350751000	0.28530563486070143000	0.29354939434014776000
2	0.20945458832778499000	0.19250866192863764000	0.20131895858790597000
3	0.11728565979642359000	0.10846708183561594000	0.11480215365071569000
4	0.06159875354654580100	0.05715500522150012600	0.06098495866437304800
5	0.03150700715055997800	0.02929276326926552300	0.03141065546708451900
6	0.01592596159664678600	0.01482755162634473400	0.01594001939504619700
7	0.00800550624716202430	0.00746164387808778520	0.00802811009646413330
8	0.00401330881872325430	0.00374423283180049360	0.00402726600271096580
9	0.00200928374829368540	0.00187619659582988340	0.00201600349693378590

Tabelle 3.2: Maximale Winkel im Bogenmaß für die drei Knotenintervallkonfigurationen uniform, nicht-uniform und nicht-konform.

Für die Unterteilungstiefen $n = 1,\ldots,9$ wird der maximale Winkel zwischen den 4 Normalen der Ebenen, welche durch (E_1^n, V_0^n, E_2^n), (E_2^n, V_0^n, E_3^n), (E_3^n, V_0^n, E_4^n) und (E_4^n, V_0^n, E_1^n) definiert sind, gemessen. Dabei wird getestet, wie schnell die benachbarten Faces des betrachteten Vertex koplanar werden. In der Tabelle 3.2 sind die maximalen Winkel im Bogenmaß für die drei Knotenintervallkonfigurationen uniform, nicht-uniform und nicht-konform aufgelistet.

Es ist zu beobachten, dass die nicht-konforme Face-Konfiguration analog zu der uniformen und NURBS-Konfiguration zu einer Ebene konvergiert. Dieses Beispiel sowie weitere Testfälle lassen mindestens eine G^1-Stetigkeit an den Rändern der nicht-konformen Faces erwarten. Nicht konvex-gültige und nicht überlappungsfreie Knotenintervallkonfigurationen (siehe Definition 7 und Abschnitt 4.3.1) wurden bei diesen Tests nicht berücksichtigt, da sie dem Bereich Special Features angehören.

3.6 Special Features

Um die Modellierungsvielfalt von Unterteilungsflächen zu erhöhen, wurden so genannte *Special Features* [HDD+94] eingeführt. Diese Zusatzfeatures sind inzwischen ein Standardwerkzeug bei der Modellierung mit Unterteilungsflächen, womit auf einfache Art und Weise lokale Stetigkeitsreduktionen zum Setzen von Akzenten im Modell durchgeführt werden können. Dieselben Effekte lassen sich mit NURBS-Flächen nur auf umständliche Weise über ein Zusammenfügen mehrerer getrimmter Patches erzielen. Auch bei den erweiterten Unterteilungsflächen sind Special Features integriert. Es stehen zwei unterschiedliche Arten von Special Features zur Verfügung, die sich ergänzen und kombinieren lassen.

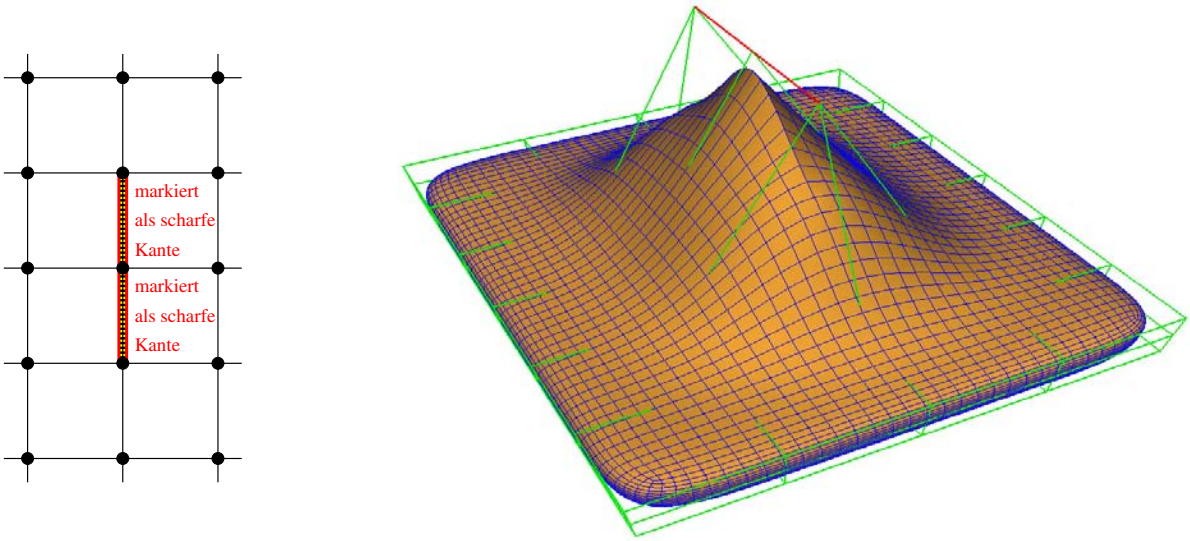


Abbildung 3.15: Scharfe Kante, generiert mit dem Special Feature-Regelsatz: Kontrollnetz in schematischer Darstellung (links) und resultierendes Bild (rechts) mit grünem Kontrollnetz und blauer Tesselierung der Limitfläche. Die betreffende Kante ist auf beiden Abbildungen rot markiert.

Üblicherweise resultieren die Special Features der Unterteilungsflächen aus zusätzlichen Verfeinerungs- und Limitregeln. Diese Feature-Regeln lassen sich aus den herkömmlichen Masken der Unterteilungs- und Limitregeln herleiten: Nur ein ausgewählter Teil der Masken wird mit modifizierten Gewichten genutzt, um eine lokale Reduktion der Stetigkeit zu erzielen. Der Special Feature-Regelsatz der ESubs basiert auf dem entsprechenden Regelsatz der Catmull-Clark-Flächen, wobei die uniformen Regeln um die nicht-uniformen Fälle erweitert werden. Um beispielsweise den Special Feature Typ *scharfe Kante* bei Catmull-Clark-Flächen zu generieren, werden die Unterteilungs- und Limitregeln der uniformen B-Spline-Kurven (siehe Abschnitt 2.3.3) auf den als scharf markierten Kantenzug angewendet. Analog dazu werden bei den ESubs der Regelsatz von NURBS-Kurven (siehe Abschnitt 2.3.2.1) für scharfe Kanten eingesetzt. Um ein glattes Auslaufen einer scharfen Kante zu erreichen, werden die herkömmlichen glatten Regeln beim letzten Vertex auf der scharfen Kante inklusive seiner 1–Nachbarschaft angewendet. Laufen mindestens drei scharfe Kanten in einem Vertex zusammen, so wird bei Catmull-Clark wie auch bei den ESubs-Flächen eine Spitze generiert, bei dem der Vertex interpoliert wird. Weitere Special Features der Catmull-Clark-Flächen können analog an ESubs-Flächen angepasst werden und in den Regelsatz der ESubs integriert werden. Beispiele für solche Special Features mit ESubs sind in den Abbildungen 3.15, 3.17, 3.18 und 3.19 gezeigt. Scharfe Kanten können dabei durch reguläre sowie irreguläre Bereiche im Kontrollnetz laufen. Eine Modifikation des Verlaufs des scharfen Kantenzugs ist über die Knotenintervalle auf der scharfen Kante sowie über die Knotenintervalle inzident zur glatt auslaufenden scharfen Kante möglich.

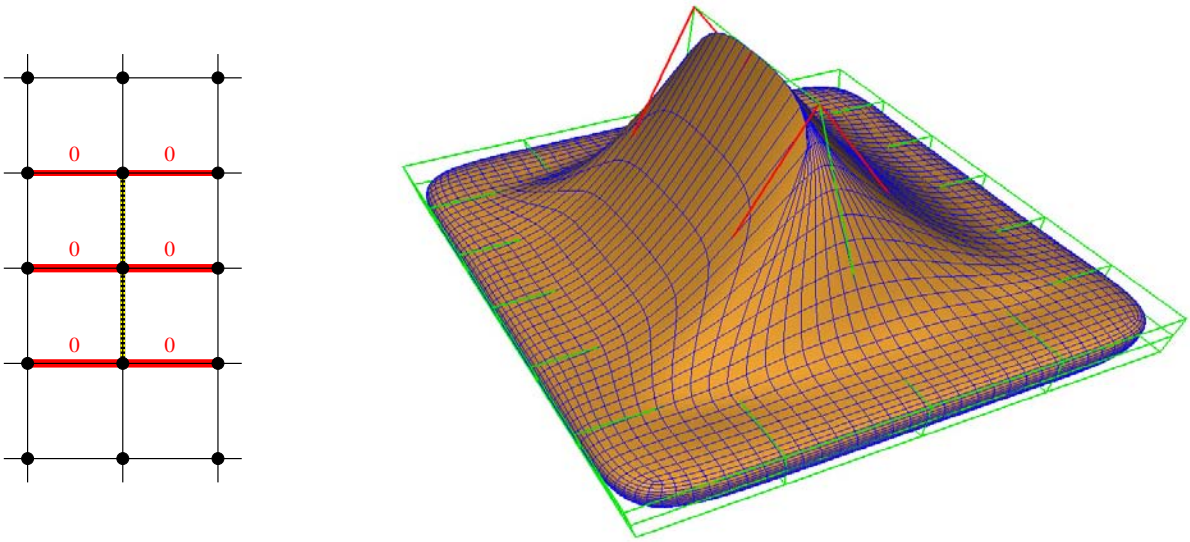


Abbildung 3.16: Scharfe Kante, generiert mit Null-Knotenintervallen: Kontrollnetz in schematischer Darstellung (links) und resultierendes Bild (rechts). Die Kanten mit null Knotenintervallen sind auf beiden Abbildungen rot markiert.

Sederberg et. al. [SSS98] stellten eine weitere Methode zur Generierung von Special Feature Effekten vor. Durch ausgewähltes Nullsetzen von Knotenintervallen konnten sie ebenfalls Formen wie z.B. scharfe Kanten und Spitzen erzeugen.

Auch für ESubs ist dieses Verfahren anwendbar: Ein Nullsetzen der Knotenintervalle entsprechender inzidenter Kanten (siehe Abbildung 3.16 rot markiert) zu einer ausgewiesenen Kante (siehe Abbildung 3.16 gelb/schwarz unterlegt) erzeugt eine scharfe Kante. Eine Spitze erreicht man mit dieser Methode durch das Nullsetzen aller Knotenintervalle in der 1-Nachbarschaft eines zu interpolierenden Vertex (siehe Abbildung 3.17). In der Anwendung sind jedoch die Special Features mit eigenem Regelsatz dem Null-Knotenintervall-Ansatz vorzuziehen, da mit ersterem Verfahren eine bessere Tesselierung der Fläche durch einfaches Unterteilen resultiert. Dies wird beim Vergleich der Abbildungen 3.15 und 3.16 deutlich. Beide Verfahren können jedoch geeignet kombiniert werden und bieten zusammen mit den nicht-konvexen Knotenintervallen eine Vielzahl von Möglichkeiten, Modelle zu gestalten (siehe Abbildung 3.18-3.20).

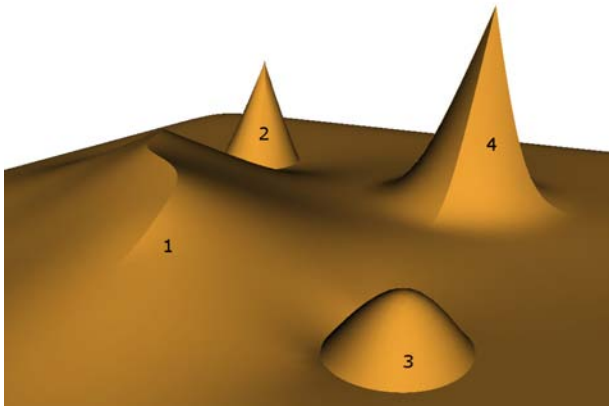


Abbildung 3.17: Mischung unterschiedlicher Special Features: Scharfe Kante mit glatt auslaufenden Enden (1), Spitze generiert durch Null-Knotenintervalle (2), Kegel (3), Spitze erzeugt durch drei in einem Vertex inzidente scharfe Kanten (4).

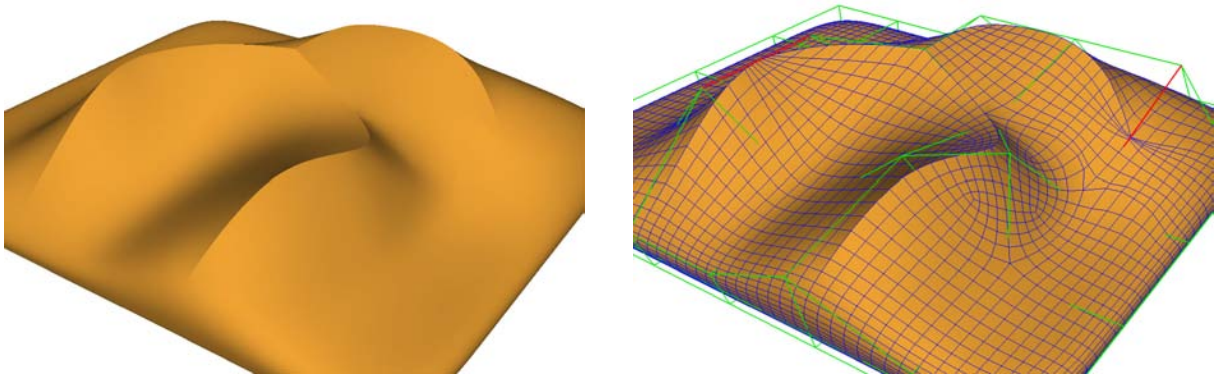


Abbildung 3.18: Nicht-uniforme Knotenintervalle (rot markiert) auf der scharfen Kante erlauben eine präzise Kontrolle des Verlaufs der scharfen Kante. Ebenso beeinflusst eine Modifikation der Knotenintervalle (rot markiert) auf den Kanten inzident zur glatt auslaufenden scharfen Kante die Gestalt der scharfen Kante. Der scharfe Kantenzug verläuft im Beispiel durch reguläre und irreguläre Vertices.

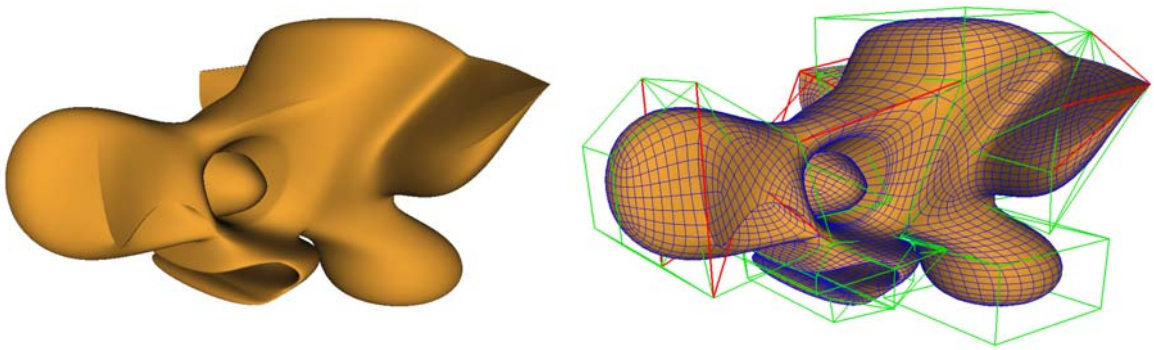


Abbildung 3.19: Beispiel für ein erweitertes Unterteilungsflächenmodell mit scharfen Kanten: Die scharfen Kanten sind im grünen Kontrollnetz rot markiert. Modell von Lars Reusche.

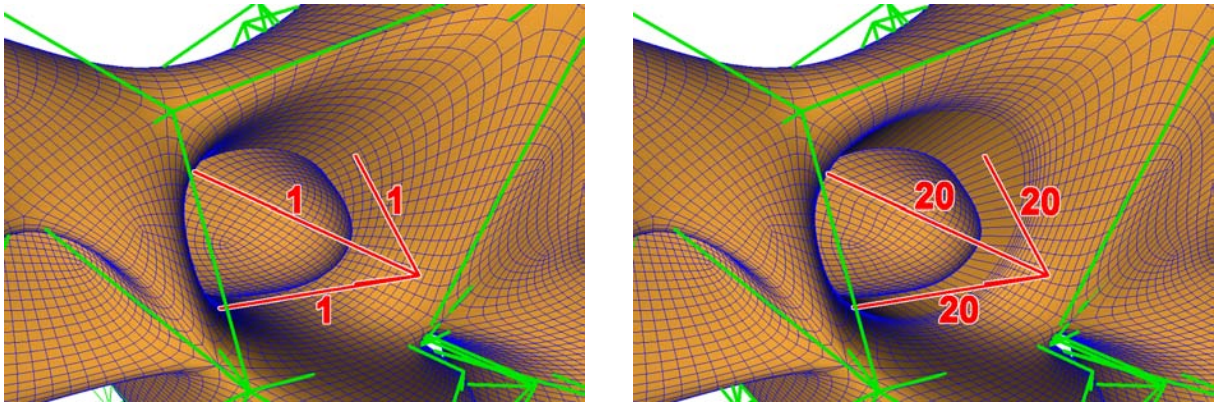


Abbildung 3.20: Wirkung konvexer (links) und nicht-konvexer (rechts) Knotenintervallkonfigurationen in Detailsicht. Die Knotenintervallwerte der rot markierten Kanten sind ebenfalls rot gekennzeichnet.

3.7 Rationale erweiterte Unterteilungsflächen

Analog zu NURBS können *rationale* erweiterte Unterteilungsflächen durch Homogenisierung erzielt werden. Die Kontrollpunkte $P_i \in \mathbb{R}^3$ der erweiterten Unterteilungsfläche erhalten Gewichte $w_i \neq 0$ in einem Homogenisierungsschritt:

$$\text{Homogenisierung} : \mathbb{R}^3 \longrightarrow \mathbb{R}^4, \qquad P = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \longmapsto P^w = \begin{pmatrix} x \cdot w \\ y \cdot w \\ z \cdot w \\ w \end{pmatrix}.$$

Zur Berechnung und Weiterverarbeitung der Fläche kommen statt der Kontrollpunkte P_i die homogenisierten Kontrollpunkte P_i^w zum Einsatz. Die Unterteilungsregeln, Special Feature Regeln und die Limitregeln werden mit den vierdimensionalen Kontrollpunkten P_i^w ebenso wie in den Abschnitten 3.3 und 3.6 beschrieben durchgeführt. Die resultierenden dreidimensionalen Kontrollpunkte und Limitpunkte der rationalen Fläche erhält man durch eine Projektion der vierdimensionalen Punkte in die Hyperebene $w = 1$:

$$\text{Inhomogenisierung} : \mathbb{R}^4 \longrightarrow \mathbb{R}^3, \qquad P^w = \begin{pmatrix} x_w \\ y_w \\ z_w \\ w \end{pmatrix} \longmapsto P = \begin{pmatrix} \frac{x_w}{w} \\ \frac{y_w}{w} \\ \frac{z_w}{w} \end{pmatrix}.$$

Somit stehen die erweiterten Unterteilungsflächen auch als rationale Flächen zur Verfügung. Bikubische *rationale* nicht-uniforme B-Spline Flächen können auf diese Weise mit den erweiterten Unterteilungsflächen repräsentiert werden.

Kapitel 4

Konvertierung, Approximierung und Modellierung

Beim Modellieren mit erweiterten Unterteilungsflächen hat der Designer die Wahl auf Werkzeuge aus der Polygonmodellierung zurückzugreifen und Vorgehensweisen aus der NURBS-Modellierung zu nutzen. Beliebige polygonalen Grundobjekte, wie z.B. ein Würfel, dienen als Ausgangskontrollnetze für einen Modellierungsprozess mit Unterteilungsflächen. Auf diesen Ausgangskontrollnetzen können topolgieverändernde Operationen ausgeführt werden, die zu komplexeren Modellen führen. Catmull-Clark-Modelle können ebenfalls mit ESubs verwendet werden. Abschnitt 4.1.1 widmet sich der Konvertierung von polygonalen Objekten und Catmull-Clark-Objekten zur Weiterverarbeitung mit ESubs.

Darüber hinaus können mit den ESubs NURBS-Modelle genutzt werden: Nach einer Konvertierung der NURBS-Fläche mit beliebigem Grad zu einer bikubischen NURBS-Fläche mit Hilfe eines gegebenen Abweichungsmaßes, stehen auch beliebige NURBS-Modelle als ESubs-Modelle zur Verfügung. Abschnitt 4.1.2 erläutert die Verwendung von NURBS-Objekten. Um mit NURBS-Flächen komplexe Modelle trotz eingeschränkter Topologie zu erzielen, sind NURBS-Objekte im Allgemeinen getrimmt. Um diese getrimmten Flächen mit ESubs korrekt darzustellen, müssen die konvertierten NURBS-Modelle ebenfalls getrimmt werden. Abschnitt 4.2 beschreibt die Integration der Trimminformationen ins Kontrollnetz und das Trimming mit ESubs. Zur Weiterverarbeitung bieten ESubs dem Modellierer Operationen an, die über die Möglichkeiten von NURBS- und Catmull-Clark-Flächen hinausgehen. Abschnitt 4.3 zeigt eine Reihe von Modellierungsoptionen auf, die im praktischen Einsatz von ESubs Verwendung finden.

4.1 Konvertierung mit Abweichungsmaß

ESubs basieren auf NURBS- sowie Catmull-Clark-Flächen. Bei Verwendung eines Kontrollnetzes mit uniformen Knotenintervallen reduziert sich die ESubs-Fläche zu einer Catmull-Clark-Fläche. Bei regulärer Topologie im Vierecks-Kontrollnetz sowie konformen Faces resultiert eine bikubische NURBS-Fläche.

Umgekehrt lässt sich eine Catmull-Clark-Fläche mit ESubs darstellen. Ebenso lässt sich eine bikubische NURBS-Fläche durch ESubs repräsentieren. Dies sind die Voraussetzungen für die Konvertierung mit Abweichungsmaß von NURBS- und Catmull-Clark-Flächen zu ESubs-Flächen.

4.1.1 Konvertierung von polygonalen Objekten und Catmull-Clark-Objekten

Um Catmull-Clark-Objekte mit ESubs nutzen zu können, müssen uniforme Knotenintervalle an den Kanten des Catmull-Clark-Kontrollnetzes ergänzt werden. Der ESubs-Regelsatz zusammen mit dem erweiterten Catmull-Clark-Kontrollnetz liefert eine identische Fläche zur Catmull-Clark-Fläche.

Zum Aufbau eines neuen ESubs-Objektes aus einem polygonalen Grundkörper muss das Kontrollnetz, das aus dem polygonalen Objekt besteht, ebenfalls um Knotenintervalle erweitert werden. Dazu bieten sich uniforme Knotenintervalle an, die in weiteren Modellierungsschritten modifiziert werden können (siehe Abschnitt 4.3).

4.1.2 Konvertierung und Approximierung von NURBS-Modellen

Die meisten NURBS-Modelle liegen als clamped NURBS-Flächen vor. Daher wird in diesem Abschnitt von clamped Flächen ausgegangen. Unclamped NURBS-Modelle können wie z.B. in [PT95] beschrieben zu clamped NURBS-Flächen umformuliert werden.

4.1.2.1 Bikubische NURBS-Modelle als erweiterte Unterteilungsflächen

Das Kontrollnetz einer bikubischen NURBS-Fläche, ergänzt um seine Knotenintervalle (siehe Abbildung 4.1), kann direkt als Kontrollnetz für eine ESubs-Fläche genutzt werden und liefert eine zur NURBS-

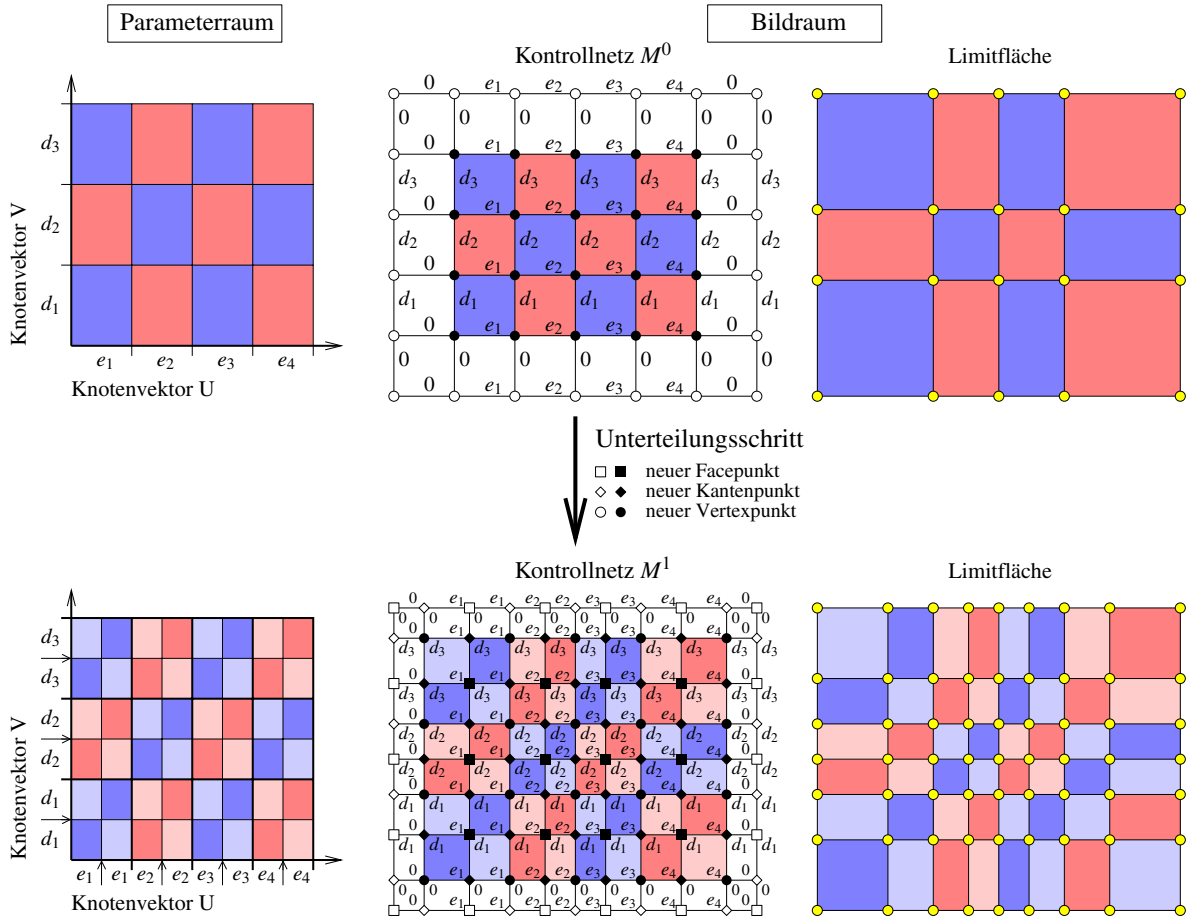


Abbildung 4.1: Parameterraum und Bildraum mit Kontrollnetz sowie Limitfläche einer zu ESubs konvertierten bikubischen NURBS-Fläche in schematischer Darstellung (oben). Ein Unterteilungsschritt der ESubs-Fläche erfolgt analog zu einem Unterteilungsschritt bei NURBS-Unterteilungsflächen (unten).

Fläche identische Fläche. Die Knotenintervalle auf den Kanten des Kontrollnetzes markieren den Parameterbereich, zu dem ein Face des Kontrollnetzes gehört (in Abbildung 4.1 jeweils blau/rot markiert). Zur Unterteilung werden analog zu den NURBS-Unterteilungsflächen Knoten an den Stellen $(\overline{e}_i + \overline{e}_{i+1})/2$ und $(\overline{d}_j + \overline{d}_{j+1})/2$ eingefügt. Aus diesem Einfügeprozess resultieren die neuen Face-, Kanten- und Vertexpunkte wie in Abbildung 4.1 dargestellt.

Im Gegensatz zum Catmull-Clark-Unterteilungsschema wird bei den ESubs analog zu den NURBS-Unterteilungsflächen der Rand der Fläche mit Null-Knotenintervallen (siehe Abbildung 4.1) nicht unterteilt. Ein Face des Kontrollnetzes ist im Randbereich, wenn sich Eckpunkte des Faces auf dem Rand befinden. Eine Kante ist im Randbereich, wenn genau ein Vertex der Kante auf dem Rand liegt. Die neuen Facepunkte und die neuen Kantenpunkte im Randbereich ziehen sich analog zu den NURBS-Flächen auf Grund der Null-Knotenintervalle zum Rand hin und bilden somit als unterteiltes Kontrollnetz M^1 die gleiche NURBS-Fläche wie mit M^0. Der Parameterraum wird bei einem Unterteilungsschritt zur Vereinfachung um Faktor zwei skaliert: Statt der Knotenintervalle $d_i/2$ und $e_j/2$ werden zur besseren Übersicht die Knotenintervalle d_i und e_j verwendet. Diese Vorgehensweise ist o.B.d.A. korrekt, da durch die Skalierung die Teilverhältnisse beibehalten werden. Beim fortwährenden Unterteilen konvergieren die Punkte im Randbereich zum Rand. Die Limitpunkte (gelb markiert in Abbildung 4.1) werden nur für die inneren Vertexpunkte (in Abbildung 4.1 schwarz markiert) berechnet, da die Limitpunkte des Randes mit den Limitpunkten der Vertices im Randbereich identisch sind. Es ergibt sich somit für ESubs analog zu NURBS-Flächen eine andere Tesselierung der Limitfläche im Randbereich als bei Catmull-Clark-Flächen.

4.1.2.2 NURBS-Flächen beliebigen Grades als erweiterte Unterteilungsflächen

NURBS-Flächen mit beliebigem Grad können mit einem benutzerdefinierten Abweichungsmaß TOL zu einer bikubischen Fläche konvertiert werden. Die resultierende Fläche liefert somit eine Approximation der Ausgangsfläche. Das Abweichungsmaß stellt sicher, dass die konvertierte Fläche $S^{konv}(u,v)$ von der originalen Fläche $S(u,v)$ nicht mehr als TOL abweicht:

$$|S(u,v) - S^{konv}(u,v)| < TOL$$

Zur Konvertierung der NURBS-Fläche $S(u,v)$ werden zunächst die Kurvensegmente in Richtung des ersten Parameters und anschließend in Richtung des zweiten Parameters mit Abweichungsmaß konvertiert (siehe Abbildung 4.2). Der Algorithmus zur Konvertierung dieser NURBS-Kurven aus dem Kontrollnetz erhält als Eingabe ein Abweichungsmaß TOL, die Kontrollpunkte K der NURBS-Kurve mit Grad p aus dem Kontrollnetz der NURBS-Fläche und den Knotenvektor des Kurvensegmentes

$$U = \{\underbrace{u_0,\ldots,u_0}_{p},\underbrace{u_1,\ldots,u_1}_{m_1},\ldots,\underbrace{u_s,\ldots,u_s}_{m_s},\underbrace{u_{s+1},\ldots,u_{s+1}}_{p}\}$$

Die NURBS-Kurve und somit auch das Kurvensegment sei o.B.d.A. clamped.
Gilt Grad $p < 3$, so wird eine verlustfreie Graderhöhung durchgeführt:

$i := 1, U^{konv} := U, K^{konv} := K$
repeat

- Extrahiere i-tes Bézier-Segment aus der NURBS-Kurve mit Kontrollpunkten K^{konv} durch $(m_i - p)$-faches Knoteneinfügen von u_i in Knotenvektor U^{konv}.
- Führe Graderhöhung des i-ten Bézier-Segmentes durch.
- Entferne überflüssige Knoten aus U^{konv} zwischen $(i-1)$-ten und i-ten Bézier-Segmentes.
- $i := i + 1$

until $i > s + 1$

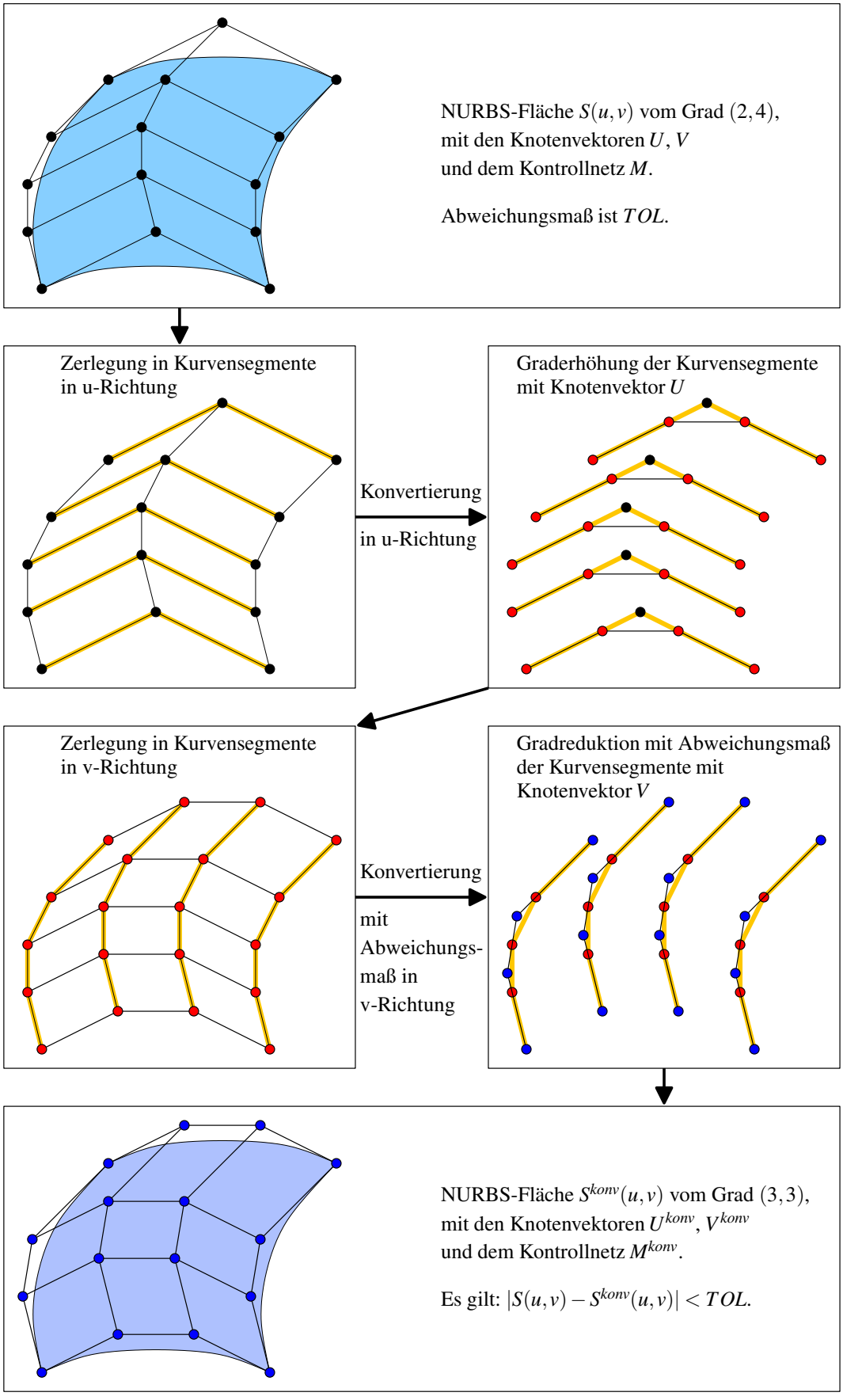


Abbildung 4.2: Konvertierung mit Abweichungsmaß einer NURBS-Fläche vom Grad $(2,4)$ zu einer bikubischen Fläche.

Liegt Grad $p > 3$ vor, so ist eine Gradreduktion mit Hilfe des Abweichungsmaßes TOL nötig:

repeat
$i := 1$, $U^{konv} := U$, $K^{konv} := K$, fertig:=true
Setze Einträge in Abweichungsvektor $A[]$ auf null.
repeat
- Extrahiere i-tes Bézier-Segment aus der NURBS-Kurve mit Kontrollpunkten K^{konv} durch $(m_i - p)$-faches Knoteneinfügen von u_i in Knotenvektor U^{konv}.
- Führe Gradreduktion des i-ten Bézier-Segmentes durch.
  Speichere maximal auftretende Abweichung ab:
  $|\text{Bézier-Segment}_i(u) - \text{Bézier-Segment}_i^{konv}(u)| \leq \text{maxA}_i =: A[i], \quad u \in [u_{i-1}, u_i]$
- Entferne überflüssige Knoten aus U^{konv} zwischen $(i-1)$-ten und i-ten Bézier-Segmentes.
  Speichere auftretende Abweichung in Abweichungsvektor $A[]$ ab.
- i:=i+1

until $i > s+1$
// Überprüfe Abweichungsvektor:
for $(i = 1;\ i \leq s+1;\ ++i)$ {
  if $(A[i] > TOL)$ {
    // Abweichung ist zu groß, zur Korrektur:
    Füge Knoten $\frac{u_{i-1}+u_i}{2}$ in Knotenvektor U $(m_i - p - 2)$-fach ein.
    fertig:=false.
  }
}
until fertig

Der Pseudo-Code gibt die Gradreduktion mit Abweichungsmaß einer NURBS-Kurve an. Zur Gradreduktion einer NURBS-Fläche werden die Kurvensegmente in Richtung des ersten Parameters und anschließend in Richtung des zweiten Parameters mit Abweichungsmaß gradreduziert. Bei der Gradreduktion in Richtung eines Parameters wird vor der Überprüfung des Abweichungsvektors für alle Kurvensegmente in Richtung des betrachteten Parameters eine Gradreduktion durchgeführt (innere repeat-until Schleife). Für eine Gradreduktion einer NURBS-Fläche mit Abweichungsmaß sind somit zwei Abweichungsvektoren zu berücksichtigen. Nach der Konvertierung der Kurvensegmente in Richtung des ersten und des zweiten Parameters liegt eine bikubische NURBS-Fläche vor, die sich innerhalb des angegebenen Abweichungsmaßes TOL bewegt. Das Kontrollnetz der konvertierten Fläche, erweitert um seine Knotenintervalle, ergibt zusammen mit dem ESubs-Regelsatz eine Fläche, die von der originalen NURBS-Fläche ebenfalls nicht mehr als TOL abweicht. Details zum Konvertierungsalgorithmus sind in [PT95] zu finden. Die Konvertierung mit Abweichungsmaß von NURBS beliebigen Grades zu bikubischen NURBS wurde im Rahmen einer Studienarbeit [Reu03] von Lars Reusche umgesetzt.

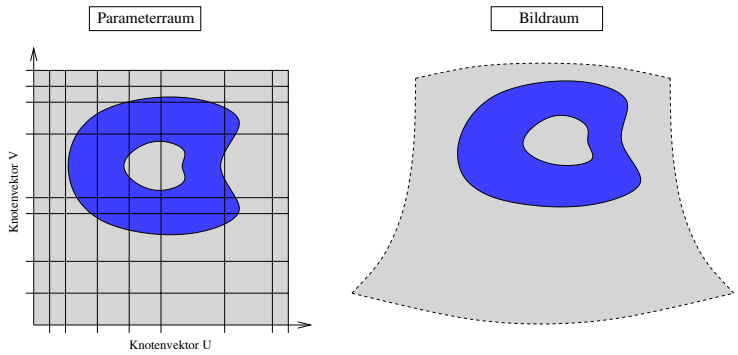


Abbildung 4.3: Trimming einer NURBS-Fläche.

4.2 Trimming von konvertierten NURBS-Modellen

Komplexe NURBS-Modelle bestehen im Allgemeinen aus mehreren getrimmten NURBS-Flächen. Um diese getrimmten NURBS-Flächen mit ESubs darstellen zu können, werden sie zunächst, wie im vorigen Abschnitt beschrieben, zu bikubischen NURBS-Flächen mit Abweichungsmaß konvertiert. Zum Trimmen der ESubs-Fläche sei M^0 das Ausgangskontrollnetz der konvertierten bikubischen NURBS-Fläche, wobei M^0 die Knotenintervalle aus den Knotenvektoren U und V der konvertierten NURBS-Fläche enthält. Der Definitionsbereich von U und V sei $[u_0,u_n]$ und $[v_0,v_m]$. Weiterhin sei $c(t)$ die Trimmkurve mit $c : [t_0,t_q] \rightarrow [u_0,u_n] \times [v_0,v_m]$. Im Beispiel von Abbildung 4.3 bzw. 4.4 besteht $c(t)$ aus zwei Kurvenstücken.

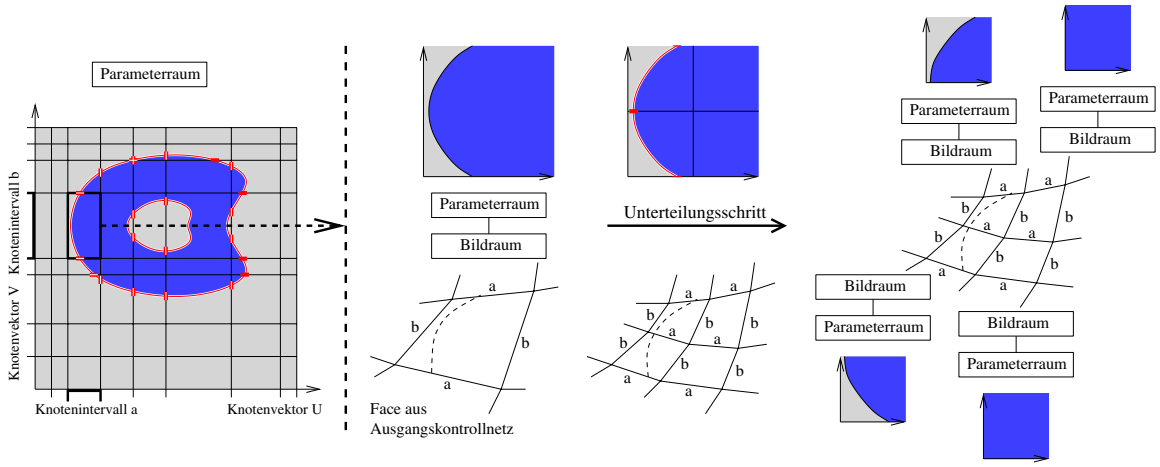


Abbildung 4.4: Trimming einer ESubs-Fläche. Die Trimminformationen entstammen dem getrimmten konvertierten NURBS-Modell.

Im ersten Schritt des Trimmings wird die Trimmkurve anhand der Schnittgeraden (u_i,y) und (x,v_j) mit $0 \leq i \leq n$, $x \in [u_0,u_n]$ und $0 \leq j \leq m$, $y \in [v_0,v_m]$, in Teilkurven zerlegt (siehe Abbildung 4.4 links). Jede Teiltrimmkurve verläuft somit in einem Teilbereich $[u_i,u_{i+1}] \times [v_j,v_{j+1}]$, $i = 0,\ldots,n-1$, $j = 0,\ldots,m-1$. Diese Parameterbereiche inklusive ihrer Teiltrimmkurven werden den Faces des Kontrollnetzes M^0 nach einer Skalierung zu $[0,1] \times [0,1]$ zugewiesen (siehe Abbildung 4.4 mitte).

Bei einem Unterteilungsschritt von Tiefe l nach Tiefe $l+1$ werden die Parameterbereiche der Faces analog zum Unterteilungsschema der Fläche unterteilt. Die Trimmkurven werden dabei entlang $(x,0.5)$ und $(0.5,y)$ mit $x,y \in [0,1]$ in Teilkurven zerlegt. Die vier Teilparameterbereiche inklusive ihrer Trimmkurven werden zu $[0,1] \times [0,1]$ skaliert und den jeweiligen vier neuen Faces in Unterteilungstiefe $l+1$ zugeordnet.

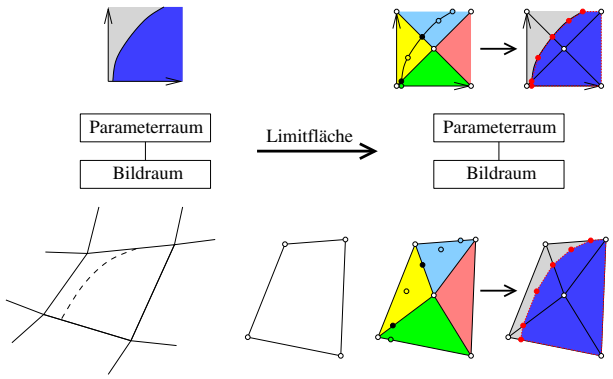


Abbildung 4.5: Tesselierung der getrimmten ESubs-Limitfläche.

In gewünschter Unterteilungstiefe l werden für jedes Face des Kontrollnetzes M^l seine vier Limitpunkte zur Darstellung bestimmt. Für Faces mit Trimmkurven ist eine zusätzliche Berechnung notwendig: Der Parameterraum des Faces wird mit Hilfe des Mittelpunktes $(0.5, 0.5)$ in vier Dreiecke zerlegt (siehe Abbildung 4.5). Die Trimmkurve wird abgetastet und durch einen Polygonzug ersetzt. Alle Abtastpunkte liegen in einem der vier Dreiecke bzw. auf den Kanten der Dreiecke. Für diese Abtastpunkte werden die baryzentrischen Koordinaten bezüglich des Dreiecks, in dem sie liegen, berechnet. Im Bildraum werden die vier Limitpunkte $L_{0,0}, L_{0,1}, L_{1,1}, L_{1,0}$ bestimmt, wobei $L_{i,j}$ der entsprechende Limitpunkt zum lokalen Parameter (i, j) des Faces ist. Der Mittelpunkt ergibt sich aus $\frac{1}{4}(L_{0,0} + L_{0,1} + L_{1,1} + L_{1,0})$. Die vier Dreiecke im Bildraum lassen sich analog zu den Dreiecken im Parameterraum aus $L_{0,0}, L_{0,1}, L_{1,1}, L_{1,0}$ und dem Mittelpunkt ermitteln (siehe Abbildung 4.5). Die Abtastpunkte können nun durch die baryzentrischen Koordinaten und die entsprechenden Dreiecke in den Bildraum übertragen werden. Mit den Abtastpunkten im Bildraum und den Limitpunkten $L_{i,j}$ lässt sich das Polygon bestimmen, das zur Fläche gehört und beispielsweise zu visualisieren ist (blau markiert in Abbildung 4.5). Das Trimming der konvertierten Modelle wurde von Lars Reusche im Rahmen seiner Diplomarbeit [Reu05] umgesetzt.

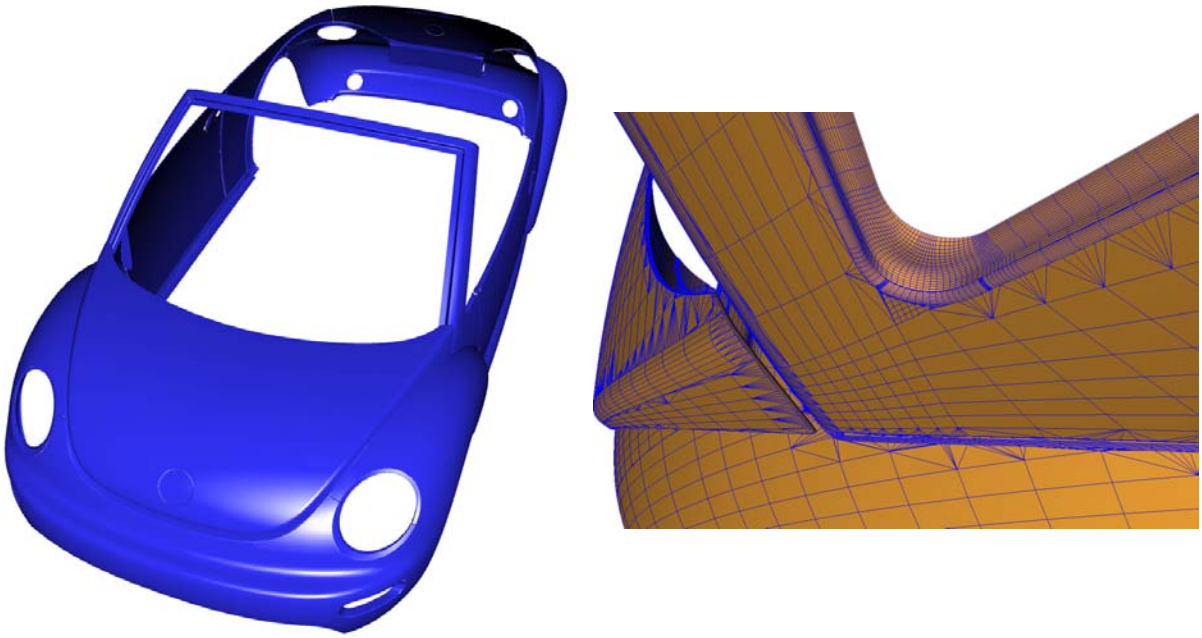


Abbildung 4.6: Getrimmtes NURBS-Modell konvertiert zu getrimmtem ESubs-Modell. Modell von Volkswagen AG aus [Reu05].

4.3 Modellierung

ESubs bieten dem Modellierer Operationen an, die über die Möglichkeiten von (bikubischen) NURBS- und Catmull-Clark-Flächen hinausgehen. Als Erweiterung von NURBS steht bei den ESubs eine beliebige Topologie zur Verfügung, die eine Generierung komplexer Modelle ohne Zuhilfenahme von Trimming erlaubt. Der Einsatz von Special Features bringt zusätzliche Akzente ins Modell, die mit konventionellen NURBS-Flächen wesentlich umständlicher nachzumodellieren sind. Als Weiterentwicklung zur uniformen Catmull-Clark-Unterteilungsfläche können bei ESubs Knotenintervalle beliebig gesetzt werden, so dass eine nicht-uniforme Unterteilungsfläche inklusive nicht-konformer Faces erzeugt werden kann.

4.3.1 Modellierung mit Knotenintervallen

In diesem Abschnitt des Modellierungskapitels werden die Möglichkeiten erläutert, die sich durch Modifikationen der Knotenintervalle ergeben. Eine Änderung eines Knotenintervallwertes bei einer NURBS-Fläche in einem uniformen Knotenvektor bewirkt ein „dehnen“ bzw. „zusammenziehen“ der Fläche im Einflussbereich des modifizierten Knotenintervalls. Durch diese Knotenintervallmodifikation ergibt sich

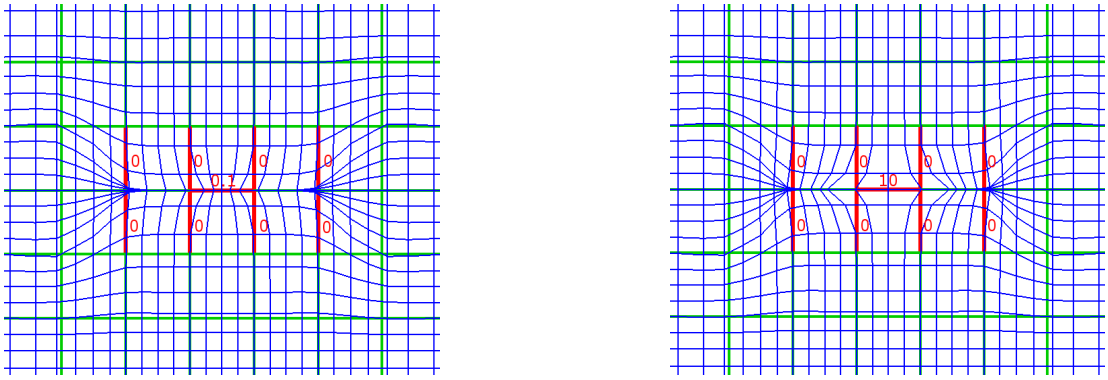


Abbildung 4.7: Kontrollpunkte in Tiefe 2 bei regulärem Einheitskontrollnetz. Verwendet wurden die angegebenen Knotenintervallkonfigurationen.

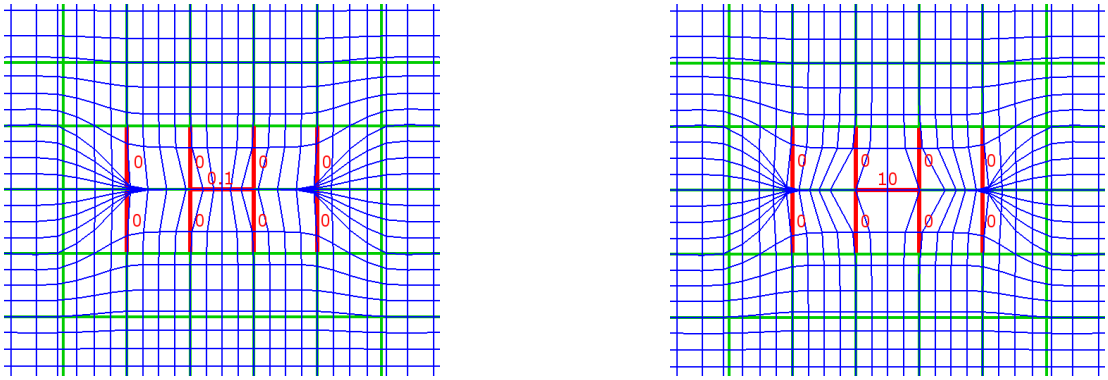


Abbildung 4.8: Limitpunkte in Tiefe 2 bei regulärem Einheitskontrollnetz. Benutzt wurden die angegebenen Knotenintervallkonfigurationen.

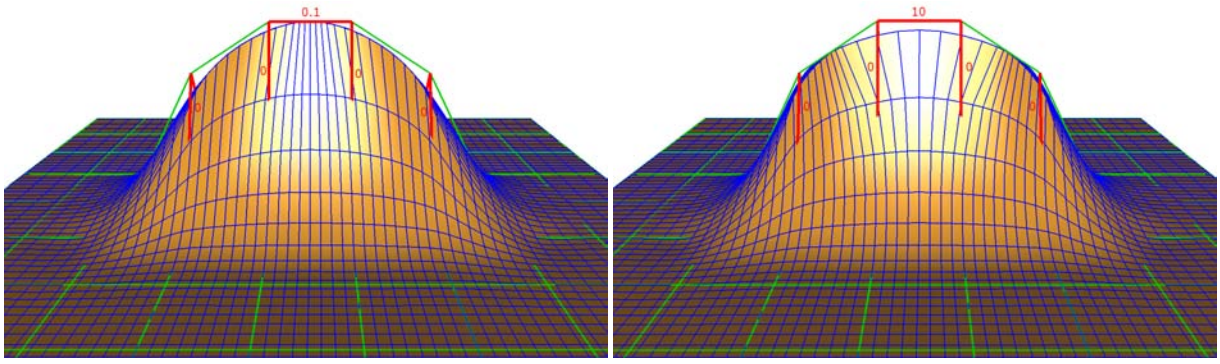


Abbildung 4.9: Tesselierte Limitfläche mit den Knotenintervallkonfigurationen aus Abbildung 4.7 und Abbildung 4.8.

somit eine nicht-uniforme Fläche. Ein „Zusammenziehen“ der Fläche kann mit Nullsetzen von Knotenintervallen im Knotenvektor verstärkt werden, womit eine Reduktion der Stetigkeit der NURBS-Fläche erzielt wird.

Analog dazu verhält sich die ESubs-Fläche: Bei regulärer Topologie mit uniformen Knotenintervallen führt eine Knotenintervalländerung einer kompletten „Reihe“ bzw. „Spalte“ von Knotenintervallen zu einer nicht-uniformen Fläche. Durch diese Knotenintervalländerung einer ganzen Reihe bzw. Spalte entstehen wiederum nur konforme Faces.

ESubs bieten zudem die Möglichkeit lokal einzelne Knotenintervalle zu modifizieren, woraus nicht-konforme Faces resultieren. Diese nicht-konformen Faces bieten dem Modellierer ein erweitertes Spektrum verfügbarer Modellierungsoptionen an. Mit nicht-konformen Faces sind Null-Knotenintervalle zur Generierung von Special Features möglich, wie bereits in Abschnitt 3.6 erläutert. Die Abbildungen 4.7 bis 4.9 zeigen ein Beispiel für eine scharfe Kante, die durch Null-Knotenintervalle generiert wurde. Zusätzlich lässt sich durch eine gezielte Modifikation der Knotenintervalle Einfluss auf den Verlauf der scharfen Kante nehmen.

Mit einer Abänderung einzelner Knotenintervalle, wodurch nicht-konforme Faces entstehen, kann somit lokal eine „Dehnung“ bzw. ein „Zusammenziehen“ der Fläche erfolgen. Die ESubs-Limitpunkte eines Vertex mit seinen beiden lokalen Knotenvektoren sind durch die NURBS-Limitpunktregel definiert und geben auf diese Weise die lokale Verzerrung der Fläche vor. Bei nicht-konformen Faces mit extremen Verzerrungen kann es zu überlappenden Knotenintervallkonfigurationen kommen. Ein Modellierungstool muss daher im Einheitskontrollnetz überprüfen, ob eine Überlappung bei den Kontrollpunkten vorliegt (siehe Abbildung 4.10) und einen entsprechenden Hinweis ausgeben. Überlappende Knotenintervallkonfigurationen können als eine Art Special Feature genutzt werden, beispielsweise lassen sich damit auf einfache Weise wellenförmige Gebilde erzeugen (siehe Abbildung 4.12). Sie sind daher nicht komplett zu unterbinden, sondern als Modellierungsoption anzusehen.

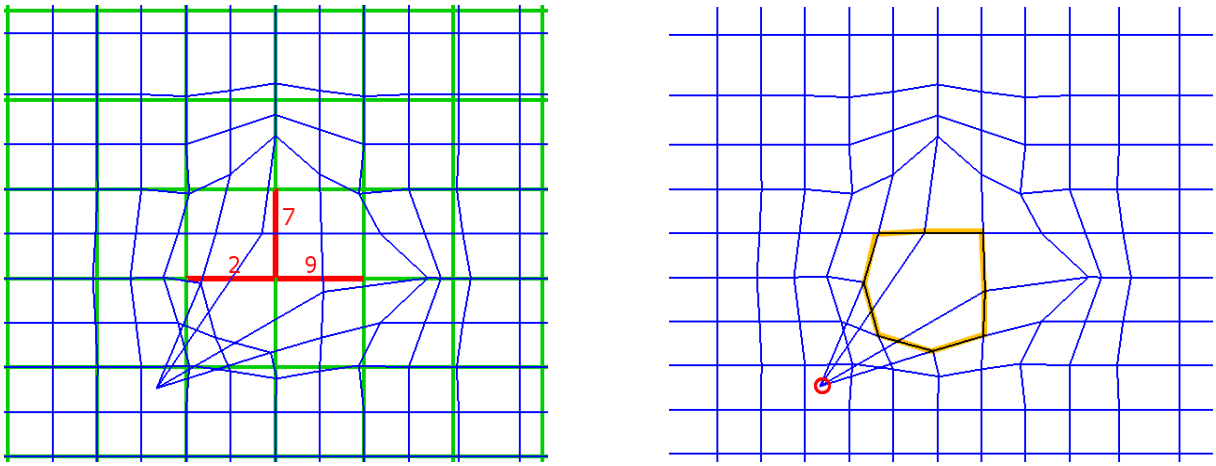


Abbildung 4.10: Reguläres Einheitskontrollnetz (grün) mit angegebener Knotenintervallkonfiguration, siehe auch Abbildung 4.11 und 4.12. Die übrigen Knotenintervalle haben den Wert 1. Das resultierende Kontrollnetz nach einem Unterteilungsschritt ist blau gekennzeichnet. Bei der gewählten Knotenintervallkonfiguration tritt für einen neuen Vertexpunkt (roter Kreis) eine Überlappung im Kontrollnetz auf. Der neue Vertex liegt nicht innerhalb des Polygonzugs seiner 1-Nachbarschaft (orange markiert). Zur besseren Unterstützung des Anwenders sollte ein Modellierungstool einen Hinweis auf diesen Fall ausgeben.

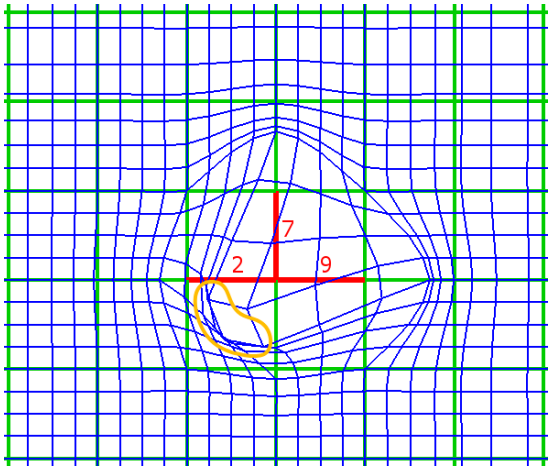


Abbildung 4.11: Tesselierte Limitfläche (blau) eines regulären Einheitskontrollnetzes (grün) mit angegebener Knotenintervallkonfiguration. Im orangefarbenen eingekreisten Bereich findet eine Überlappung statt.

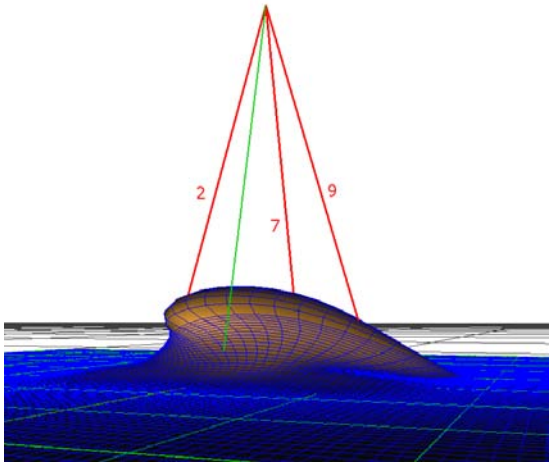


Abbildung 4.12: Tesselierte Limitfläche (blau) mit der gleichen Knotenintervallkonfiguration wie Abbildung 4.11. Mit Hilfe einer überlappenden Knotenintervallkonfiguration können auf einfache Weise wellenförmige Gebilde erzeugt werden.

4.3.2 Beispiele für die Modellierung mit ESubs

ESubs-Modelle lassen sich aus beliebigen NURBS-, Catmull-Clark- bzw. Polygonmodellen generieren (siehe Abbildung 4.13 bis 4.19). Bei einer Neukreation eines Objektes wird bei den ESubs analog zum Modellierungsprozess von NURBS und Unterteilungsflächen verfahren. In diesem Abschnitt sind dazu einige Beispiele aufgeführt.

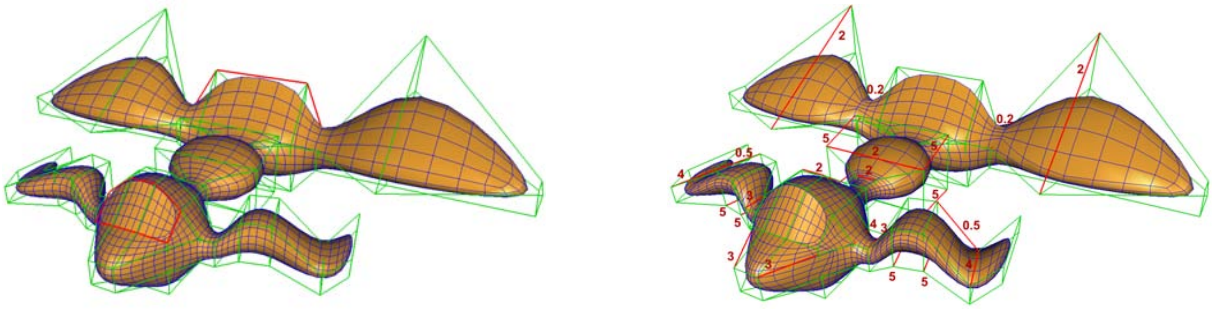


Abbildung 4.13: Catmull-Clark-Modell mit scharfen Kanten (rot markiert, links) konvertiert zu einem ESubs-Modell, bei dem zusätzlich Knotenintervalle modifiziert wurden (rechts, rot gekennzeichnet).

Die Abbildungen 4.14 und 4.15 zeigen ein Beispiel für ein ESubs-Modell, das aus einem planaren Kontrollnetz mit uniformen Knotenintervallen erzeugt wurde. Mittels diverser topologieverändernden Netzoperationen wie beispielsweise das *Splitten* eines Faces oder das *Extruden* eines Faces lässt sich das Kontrollnetz des Ausgangsobjektes beliebig modifizieren. Translationen von Vertices, Kanten und Faces des Kontrollnetzes führen schließlich mit den topologieverändernden Netzoperationen zu komplexen Objekten. Der Kaktusbaum in Abbildung 4.14 ist auf diese Weise aus einer Ebene entstanden. Alle Knotenintervalle des Modells sind bislang uniform, so dass eine Catmull-Clark-Fläche repräsentiert wird. Um Akzente ins Modell zu setzen, können die Knotenintervalle genutzt werden. Dadurch lassen sich nicht-uniforme Flächenteile oder nicht-konforme Faces bilden. Abbildung 4.15 zeigt die Wirkungsweise unterschiedlicher Knotenintervall-Belegungen.

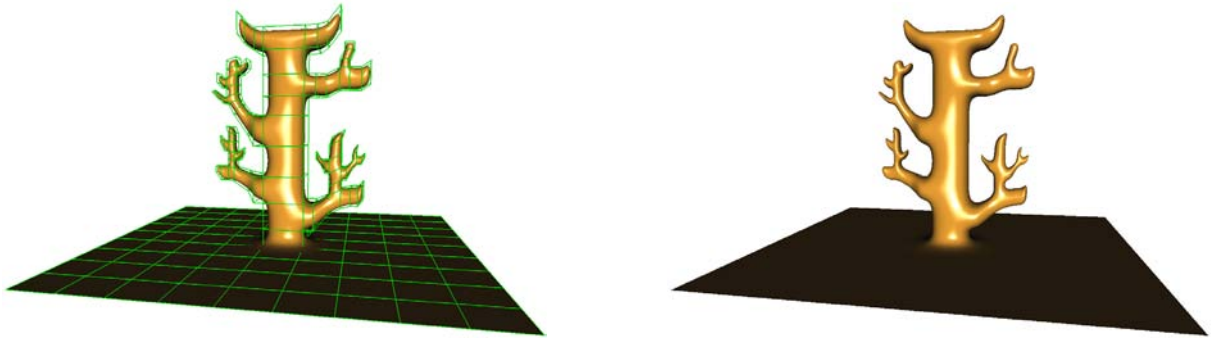


Abbildung 4.14: Durch diverse Modellierungsoperationen auf einem planaren Kontrollnetz mit uniformen Knotenintervallen ist ein komplexeres Modell entstanden. Wegen der uniformen Knotenintervalle repräsentiert die ESubs-Fläche eine Catmull-Clark-Fläche (links mit Kontrollnetz, rechts resultierende Fläche).

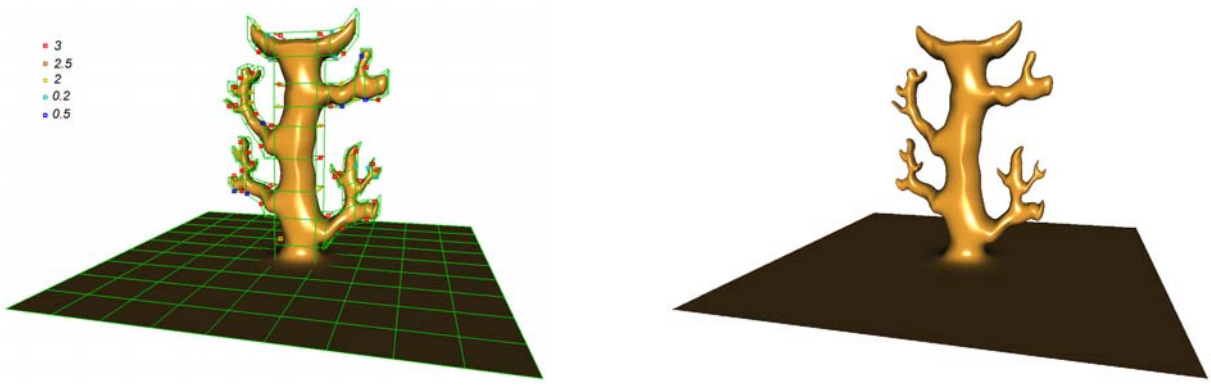


Abbildung 4.15: Eine Änderung vom uniformen Knotenintervallwert 1 zu den angegebenen Werten auf den rot markierten Kanten (links) führt zu nicht-uniformen Flächenteilen und nicht-konformen Faces. Dadurch sind entsprechende Akzente im Modell möglich, die dem Modellierer erlauben individuelle Gestaltungspunkte zu setzen.

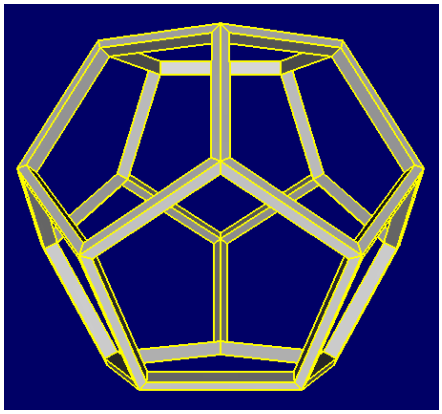


Abbildung 4.16: Beispiel für ein Polygonmodell. Modell von Pytha Lab GmbH.

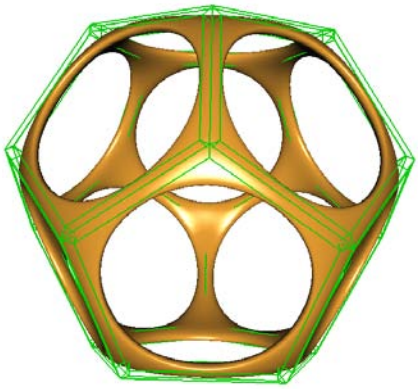


Abbildung 4.17: Polygonmodell ergänzt um uniforme Knotenintervalle und verwendet als ESubs-Kontrollnetz.

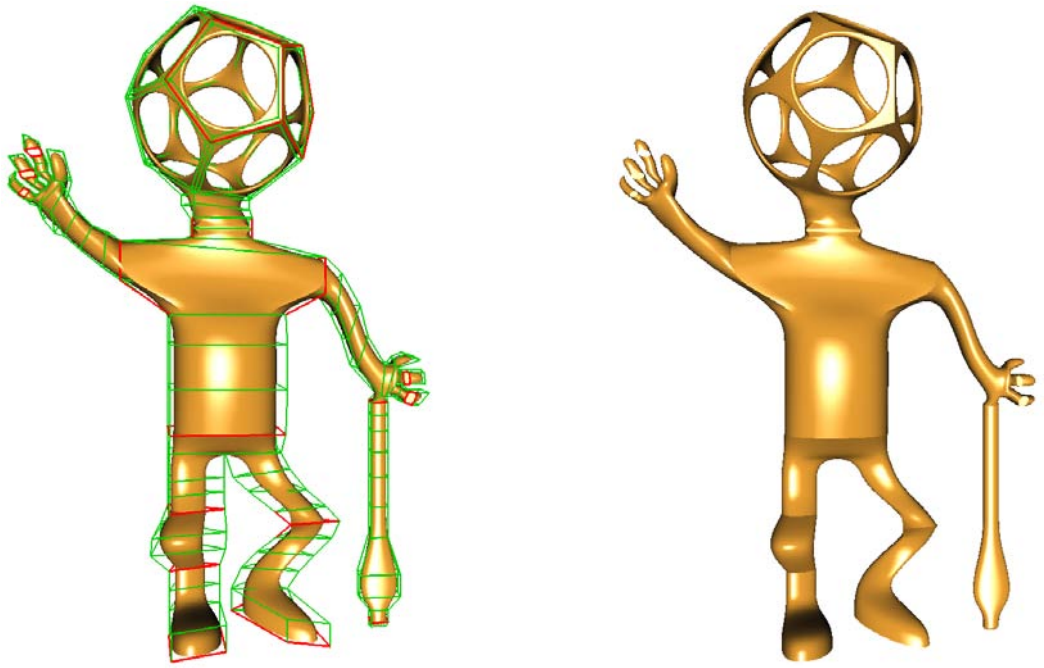


Abbildung 4.18: Anwendung von Modellierungsoperationen auf das Polygonmodell und Setzen von scharfen Kanten (rot markiert im Ausgangskontrollnetz). Die resultierende ESubs-Fläche ist wegen den uniformen Knotenintervallen eine Catmull-Clark-Fläche.

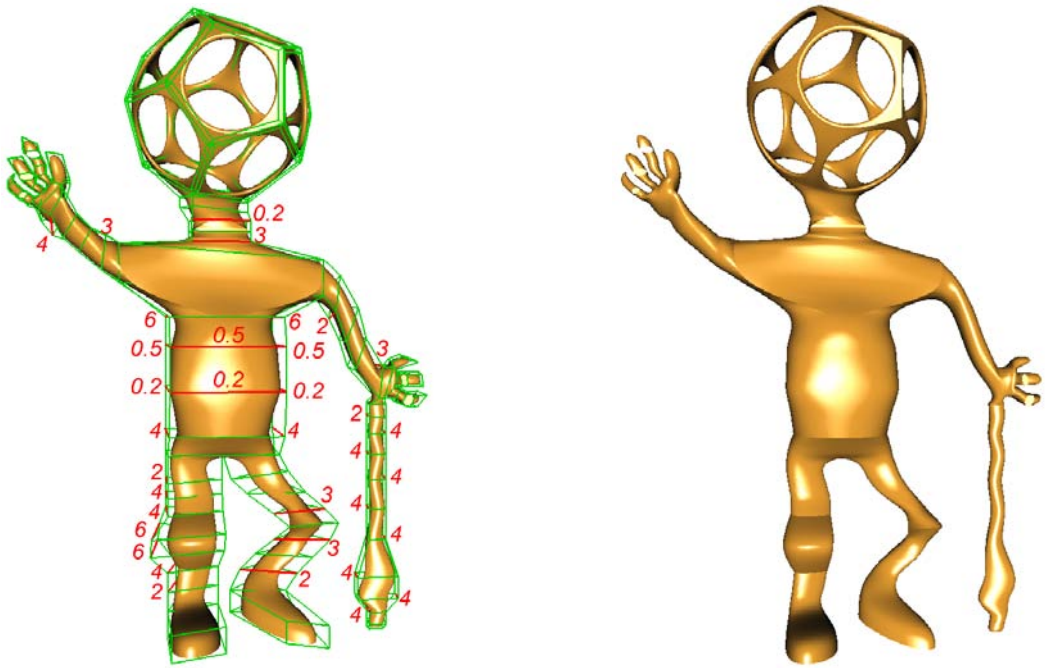


Abbildung 4.19: Die Knotenintervalle der rot markierten Kanten werden auf die angegebenen Werte gesetzt, die übrigen Knotenintervalle bleiben uniform auf Wert 1. Durch Änderung der Knotenintervalle ergeben sich nicht-uniforme Flächenteile und nicht-konforme Faces, mit deren Hilfe die Fläche zusätzlich gestaltet werden kann.

Polygonmodelle, ergänzt um uniforme Knotenintervalle, stehen ebenfalls als Ausgangskontrollnetze für ESubs zur Verfügung, wie Abbildung 4.16 und 4.17 demonstriert. Eine identische Darstellung des Polygonmodells mit ESubs wird durch das scharf Setzen aller Kanten des Kontrollnetzes erzielt (siehe Abbildung 4.16), ein glatt Setzen aller Kanten liefert das gewünschte abgerundete Objekt (siehe Abbildung 4.17). In nachfolgenden Modellierungsprozessen können diese Objekte, mit Unterstützung der besonderen Eigenschaften der ESubs, weiterverarbeitet werden (siehe Abbildung 4.18 und 4.19). Am Beispiel von Abbildung 4.19 sind Einsatzmöglichkeiten der ESubs in der Modellierung durch Kombinationen von Special Features, nicht-uniformen Knotenintervallen und nicht-konformen Faces demonstriert.

Catmull-Clark-Modelle erhalten ebenso wie Polygonmodelle uniforme Knotenintervalle zugewiesen. Sie können identisch mit ESubs dargestellt sowie weiter bearbeitet werden (siehe Abbildung 4.13).

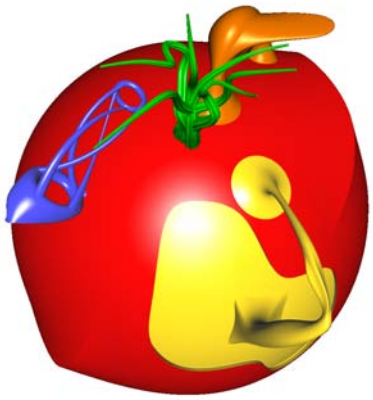


Abbildung 4.20: Zu ESubs konvertierte getrimmte NURBS-Flächen. Die konvertierten Flächen wurden in einem weiteren Modellierungsprozess modifiziert und mit Special Features versehen. Modell von Matthias Richter und Lars Reusche.

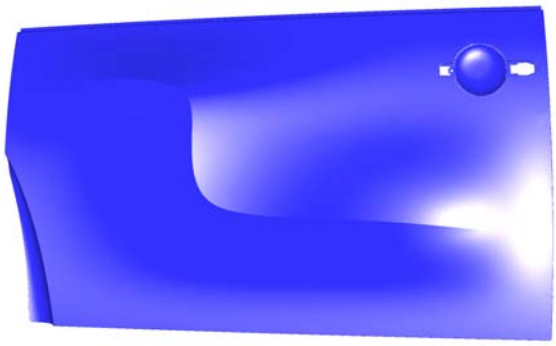


Abbildung 4.21: Zu ESubs konvertiertes getrimmtes NURBS-Modell, in einem anschließenden Modellierungsprozess erweitert um Special Features. Modell von Volkswagen AG [VW]

NURBS-Flächen beliebigen Grades können zu bikubischen NURBS-Flächen graderhöht bzw. mit einem gegebenen Abweichungsmaß gradreduziert werden (siehe Abschnitt 4.1.2). Das resultierende Kontrollnetz, ergänzt um die Knotenintervalle der beiden NURBS-Knotenvektoren, dient als Ausgangskontrollnetz für die ESubs-Fläche. Die Abbildungen 4.20 und 4.21 zeigen ESubs-Flächen, die aus konvertierten getrimmten NURBS-Flächen gebildet wurden und in einem weiteren Modellierungsschritt mit Special Features versehen wurden. Mit ESubs können auch konvertierte NURBS-Flächen mit unterschiedlichen Knotenvektoren zusammengesetzt werden, wie in Abbildung 4.22 gezeigt.

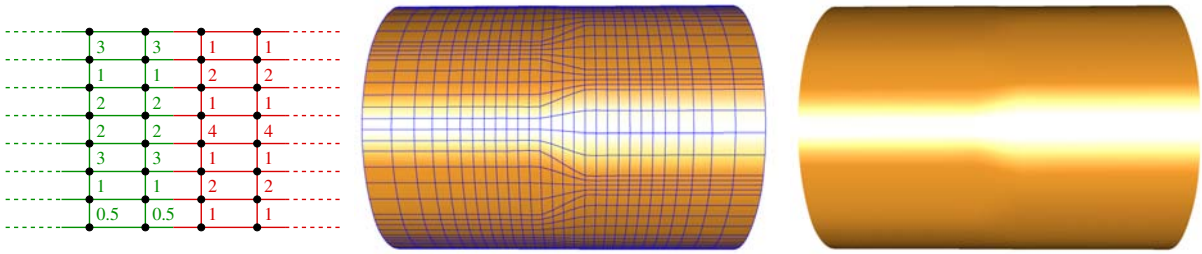


Abbildung 4.22: Zusammenfügen zweier NURBS-Flächen (grün und rot markiert) mit unterschiedlichen Knotenvektoren, wodurch eine Spalte nicht-konformer Faces entsteht. In schematischer Darstellung: Kontrollnetz mit Knotenintervallen (links), tesselierte Fläche (mitte) und resultierende Fläche (rechts).

Kapitel 5

Visualisierung

Nach Einführung der Unterteilungsflächen am Beispiel Loop, Catmull-Clark und ESubs, steht nun die Visualisierung im Mittelpunkt des Interesses. Ziel jeder Visualisierung ist es, in akzeptabler Zeit ein optisch möglichst ansprechendes Bild zu generieren. Dabei steht bei der interaktiven Darstellung die akzeptable Zeit im Vordergrund, während bei der photorealistischen Darstellung die Qualität des Bildes den Schwerpunkt setzt.

Diesen Zielen gegenüber steht das exponentielle Wachstum der Unterteilungsflächen im Zuge weiterer Verfeinerungsschritte. Wird Unterteilungstiefe l als ausreichende Tiefe akzeptiert, so sind beim Oberflächentyp Loop oder Catmull-Clark mit einem Dreiecks- bzw. Viereckskontrollnetz bereits (Anzahl Faces des Kontrollnetzes)$\cdot 4^l$ Dreiecke respektive Vierecke darzustellen. Nimmt man als Beispiel ein Kontrollnetz mittlerer Größe mit 1000 Faces, so liefert eine Unterteilungstiefe von 4 bereits 256000 Polygone. Mehrere Objekte in dieser Größenordnung würden eine feine Unterteilung schlicht unmöglich machen. Um also ein qualitativ hochwertiges Bild zu erhalten, muss erkannt werden, welche Teile der Unterteilungsfläche eine größere Verfeinerung bedürfen, welche an der Beleuchtungsrechnung nicht teilnehmen und welche ein optisch zufrieden stellendes Resultat mit geringerer Unterteilungstiefe liefern. Auf diese Weise werden adaptiv nur die notwendigen Unterteilungen zur Generierung eines qualitativ hochwertigen Bildes durchgeführt, es findet also eine Auswahl aus den $\sum_{i=0}^{l}$(Anzahl Faces des Kontrollnetzes)$\cdot 4^i$ Polygonen statt. Die vorhandene Rechenzeit wird somit optimal zur Berechnung und Darstellung der notwendigen Polygone investiert. Die entwickelten Verfahren zur adaptiven Visualisierung von Unterteilungsflächen wurden in [MH00], [SMFF03], [SMFF04] und [MTF03] veröffentlicht.

5.1 Interaktive Darstellung

Ein Bereich der Computergraphik, der mit wachsenden Rechnerkapazitäten zunehmend an Bedeutung gewinnt, widmet sich der interaktiven Darstellung von Objekten. Ausgehend von dem Gedanken, dass der Nutzer einer virtuellen Welt den Eindruck einer real existierenden Welt erfahren soll, gewinnt die Wahl der verwendeten Modelle eine zentrale Rolle. Ist man kein Anhänger des Kubismus, so beinhaltet eine realistisch wirkende Umgebung eine Vielzahl Freiformflächen. Unterteilungsflächen – als ein möglicher Freiformflächentyp – bieten insbesondere im VR-Umfeld ihre gewohnten Vorteile wie z.B. beliebige Wahl der Topologie während der Modellierung. Inzwischen sind Unterteilungsflächen auch in den meisten Standard-Modellierungstools wie z.B. 3ds Max, Maya, Softimage, Cinema4D integriert, so dass die Möglichkeit besteht vielfältige VR Modelle zu generieren.

Das Problem des exponentiellen Wachstums der Polygonanzahl bei steigender Verfeinerungstiefe stellt jedoch insbesondere für die interaktive Darstellung eine echte Herausforderung dar.

Existierende Lösungsansätze lassen sich in vier Klassen einordnen:

1. Direkte Berechnung an allgemeinen Parameterwerten.
2. Rekursive Berechnung.
3. Nutzung vorberechneter Tabellen.
4. Berechnung in Hardware.

Auch bei Unterteilungsflächen ist es möglich, patchweise Limitpunkte und Normalen an beliebigen Parameterwerten zu berechnen [Sta99], [Sta98]. Zorin und Kristjansson [ZK02] erweiterten die Arbeit von Stam [Sta99], [Sta98] um stückweise glatte Flächen. Mit dieser direkten Berechnungsart ist eine beliebige Form der Tesselierung durchführbar, welche auch in dem Modellierungspaket Maya implementiert wurde.

Die rekursive Berechnung, basierend auf der wiederholten Anwendung der Unterteilungsregeln, ist die intuitivste Art und Weise eine Unterteilungsfläche zu berechnen. Pulli und Seagal [PS96] stellen dazu eine Lösung vor, bei der auf einem Dreieckskontrollnetz zunächst Paare gebildet werden, die anschließend mit ihrer *Sliding Window* Methode tesseliert werden. Dabei werden alle Dreiecks-Paare mit Hilfe von drei Arrays je erreichter Unterteilungsstufe bis zu einer gewünschten Tiefe unterteilt. Die resultierenden Limitpunkte und Normalen werden in Form von Triangle-Strips ausgegeben. Dieses Konzept wurde um eine adaptive Verfeinerung erweitert und in [MH00] vorgestellt (siehe auch Kapitel 5.1.4.1). Havemann präsentierte in [Hav02] ein rekursives Schema für Catmull-Clark-Unterteilungsflächen basierend auf Unterteilungsringen. Alle genannten Verfahren lassen das Ausgangskontrollnetz unverändert. Im Gegensatz dazu führen Sovakar et. al. [SvSK03] die adaptiven Verfeinerungsschritte direkt auf dem Kontrollnetz aus. Um eine Vergröberung sowie eine korrekte Unterteilung auf gleicher Tiefe zu gewährleisten, können in einem Vertex der Tiefe l die Vertexpositionen für die Tiefen 0 bis l gespeichert werden.

Statt rekursiv bis zur gewünschten Unterteilungstiefe zu gelangen, kann man auch vorberechnete Tabellen nutzen. Mit Hilfe dieser Tabellen lassen sich die Limitpunkte und Normalen für eine gegebene Tiefe direkt aus dem Ausgangskontrollnetz berechnen [BS02].

Eine Implementierung in Hardware wurde bereits von namhaften Graphikkartenherstellern angekündigt, sie ist bislang jedoch noch nicht verfügbar. Ein Vorschlag zur Realisierung in Hardware wurde von Bischoff et. al. via Vorwärtsdifferenzen in [BKS00] beschrieben.

Betrachtet man die interaktive Darstellung von Unterteilungsflächen im Kontext der vorliegenden Arbeit, so muss das zu verwendende Verfahren den folgenden Ansprüchen genügen: Die Rechenzeit zur Generierung eines Bildes soll optimal genutzt werden, um möglichst hochwertigen Bilder in akzeptabler Framerate mit Hilfe eines adaptiven Verfahrens zu erzielen. Unterteilungsflächen basierend auf Dreiecks- sowie Viereckskontrollnetzen sollen im Algorithmus ohne größere Umstellungen integrierbar sein. Der Speicherplatz soll vom Tesselierungsalgorithmus konstant und unabhängig von der Größe des Ausgangskontrollnetzes effizient genutzt werden. Die Laufzeit muss linear zur Anzahl der darzustellenden Polygone sein. Berechnete Werte müssen durch ein geeignetes Caching effizient verfügbar sein. Um einen Einsatz in einem Szenengraph-System zu gewährleisten, darf das Ausgangskontrollnetz nicht verändert werden.

5.1.1 Grundidee

Um das Konzept der adaptiven Verfeinerung (siehe auch [LE97], [Hop97]) zu realisieren, erhält jedes Patch des Ausgangskontrollnetzes anhand der Kriterien Krümmung, projizierte Größe auf die Bildebene und eventuelle Zugehörigkeit zur Silhouette eine eigene Unterteilungstiefe (siehe Kapitel 5.1.2). Kameraabgewandte Patches werden aussortiert und dem Rendering Prozess nicht zugeführt. Nach der

Zuweisung der Unterteilungstiefen wird für jedes Patch seine 1-Nachbarschaft aus dem Ausgangskontrollnetz ausgelesen und in einer dedizierten Datenstruktur abgelegt. Innerhalb dieser Datenstruktur wird das Patch bis zur angegebenen Tiefe unterteilt (siehe Kapitel 5.1.4). Das Ausgangskontrollnetz wird dabei nicht verändert. Anschließend werden die Limitpunkte und Normalen des Patches berechnet. Haben die Kantennachbarn des Patches eine höhere Unterteilungstiefe, so wird die Tesselierung im Randkurvenbereich angepasst damit keine Lücken entstehen (siehe Kapitel 5.1.5). Die resultierenden Limitpunkte und Normalen ergeben die tesselierte Oberfläche in der gewünschten Feinheit (siehe Kapitel 5.1.6). Das im Abschnitt 5.1.2 und 5.1.4.2 beschriebene Verfahren implementierte Volker Settgast im Rahmen seiner Studienarbeit [Set04] und des Projektes OpenSG PLUS [Opeb], gefördert durch bmb+f 01IRA 02G. Das Teilprojekt 01IRA 02G koordinierte und führte Christoph Fünfzig [FF04] durch. Der Algorithmus zur interaktiven Darstellung nutzt als Datenstruktur für die Kontrollnetze der Unterteilungsflächen das am Lehrstuhl für Computergrahik und Multimedia der RWTH-Aachen entwickelte OpenMesh [BSBK02], [Bot].

Da aus Effizienzgründen eine statische Datenstruktur bei der Tesselierung zum Einsatz kommt, ist es nötig eine maximale Unterteilungstiefe *maxsd* und eine maximale Valenz *maxval* der Vertices anzugeben. Für die meisten Modelle reichen $maxsd = 5$ und $maxval = 50$ aus.

Im Folgenden wird der Algorithmus zunächst für Viereckskontrollnetze beschrieben, in Kapitel 5.1.11 wird eine Übertragung auf Dreieckskontrollnetze erläutert.

5.1.2 Bestimmung der Tiefenwerte

Der Tiefenwert eines Patches setzt sich zusammen aus seiner:

- Krümmung
- Sichtbarkeit
- projizierten Größe auf die Bildebene
- Lage (Silhouette/innerer Bereich)

Die Auswertung dieser Kriterien muss möglichst schnell erfolgen, um akzeptable Frameraten zu ermöglichen. Daher wird auf eine exakte Berechnung verzichtet. Statt dessen kommt eine Approximation zum Einsatz. Die Zuordnung der Tiefenwerte erfolgt in zwei Stufen: Die Krümmung muss nur bei einer Änderung des Kontrollnetzes berechnet werden, während die restlichen Kriterien kameraabhängig sind und somit in einer interaktiven Darstellung Bild für Bild bei jeder Kamerabewegung neu ermittelt werden.

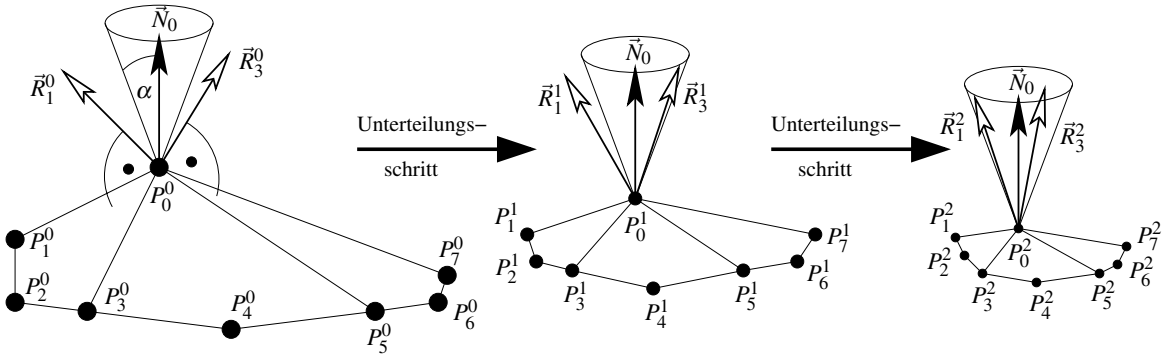


Abbildung 5.1: Normalenkegel mit Winkel α in Unterteilungstiefe 0, 1 und 2: In Tiefe 2 liegen die dargestellten Normalen der Nachbarfaces innerhalb des Normalenkegels.

Zur Krümmungsapproximation wird für jeden Kontrollpunkt mit Hilfe der Normalenkegel-Technik [SAE93] eine Unterteilungstiefe berechnet. Es sei P_0^0 der zu untersuchende Kontrollpunkt mit Valenz val, P_0^l ist Kontrollpunkt P_0^0 nach l Unterteilungsschritten. Die 1-Nachbarschaft von P_0^l in Unterteilungstiefe l sei P_j^l, $j = 1,\ldots,val$. $\vec{N}_0$ sei die Limitnormale zu $L(P_0^l)$ und $\vec{R}_i^l$ sind die Normalen der benachbarten Faces $i = 1,..,val$ in Unterteilungstiefe l. Um $\vec{N}_0$ wird ein Zylinder mit gegebenem Öffnungswinkel α gelegt, die Spitze des Zylinders Zyl^l liegt auf P_0^l (siehe Abbildung 5.1). Gesucht ist die minimale Unterteilungstiefe l, so dass alle $\vec{R}_i^l$, $i = 1,..,val$ innerhalb Zyl^l liegen:

$$\forall i = 1,..,val : \quad \vec{N}_0 \cdot \vec{R}_i^l < \cos\alpha \tag{5.1}$$

Damit erhält man eine Fläche, deren Krümmung an den Limitpunkten $L(P_j^0)$ für alle P_j^0 aus M^0 kleiner ist als durch das approximierende Maß α vorgegeben. Hat der betrachtete Kontrollpunkt r inzidente scharfe Kanten, so wird die Normalenkegelberechnung für jeden glatten Bereich B_i, $i = 1,..,r$ durchgeführt. Der zu untersuchende Kontrollpunkt erhält somit für jeden Bereich eine minimale Unterteilungstiefe zugeordnet (siehe Abbildung 5.2). Daraus ergibt sich aus dem Krümmungskriterium für jedes Face eine Unterteilungstiefe aus der maximale Tiefe seiner Kontrollpunkte.

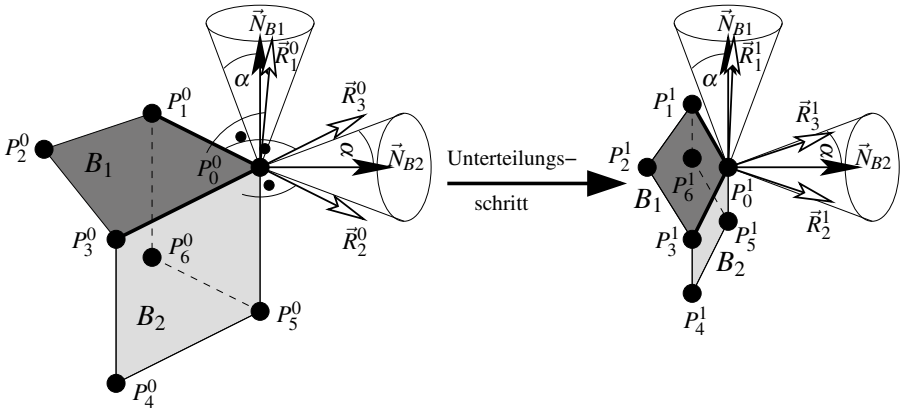


Abbildung 5.2: Scharfe Kante mit zwei glatten Bereichen B_1, B_2 und somit zwei Normalenkegeln: In Tiefe 1 liegen die Normalen der Nachbarfaces aus B_2 (hellgrau) im dedizierten Normalenkegel, in B_1 (dunkelgrau) wird Tiefe 0 benötigt.

Es sei *cameraposition* die aktuelle Position der Kamera, L_i sei der Limitpunkt und $\vec{N}_i$ die Normale von Kontrollpunkt P_i^0 und

$$\overrightarrow{camray}_i = \frac{L_i - cameraposition}{\|L_i - cameraposition\|}.$$

Um die kameraabhängigen Kriterien zu erfüllen, wird zunächst eine Klassifizierung für alle Kontrollpunkte P_i^0 aus M^0 anhand eines gegebenen ε vorgenommen (siehe Abbildung 5.3):

$$\begin{aligned}
&\text{Back-Vertex} && : && \varepsilon && < && \vec{N}_i \cdot \overrightarrow{camray}_i \\
&\text{Front-Vertex} && : && -\varepsilon && > && \vec{N}_i \cdot \overrightarrow{camray}_i \\
&\text{Silhouetten-Vertex} && : && -\varepsilon && \leq && \vec{N}_i \cdot \overrightarrow{camray}_i \quad \leq \quad \varepsilon
\end{aligned}$$

Anschließend erfolgt eine Traversierung der Faces mit Einordnung nach:

Back-Face	:	Alle Vertices des Faces sind Back-Vertices,
Front-Face	:	Alle Vertices des Faces sind Front-Vertices,
Silhouetten-Face	:	sonst

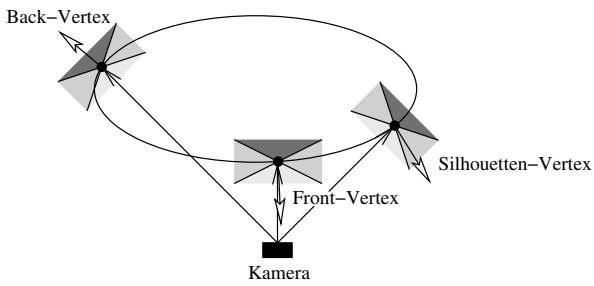


Abbildung 5.3: Front-, Silhouetten- und Back-Vertex.

Für alle Faces F_i^0 aus M^0 erfolgt eine kameraabhängige Tiefenzuweisung anhand der Klassifizierung Front-, Back- und Silhouetten-Faces. Berücksichtigt wird dabei die bereits berechnete Tiefe t_i^{cur} aus dem Krümmungskriterium für jedes Face F_i^0. Somit erhalten Front-Faces und Silhouette Faces als Tiefenwerte:

$$\begin{aligned} \text{Tiefe Front-Face} &= t_i^{cur} \\ \text{Tiefe Silhouetten-Face} &= \frac{t_i^{cur} + maxsd}{2} \end{aligned}$$

Back-Faces treten nicht in den Renderingprozess ein.

Um sowohl eine zu grobe als auch eine zu feine Unterteilung auszuschließen, wird die projizierte Größe aller sichtbaren Patches in der Bildebene überprüft. Wiederum kommt eine schnelle Approximation statt einer exakten, aber langsameren Berechnung zum Einsatz. Um das zu untersuchende Face F wird eine minimal umhüllende Kugel mit Radius R gelegt. Aus den gegebenen Werten *ProjectedSizeMin* und *ProjectedSizeMax* wird der minimale und maximale akzeptierte Radius R_{min} und R_{max} für die gewünschten Subpatches von F berechnet. Berücksichtigt werden dabei neben dem Abstand d zur Kamera und ihrem Öffnungswinkel auch die Bildauflösung:

$$\begin{aligned} \text{Überdeckung eines Pixels in x-Richtung} &: \beta_x = \frac{2 \cdot \text{Öffnungswinkel in x-Richtung}}{x - \text{Auflösung}} \\ \text{Überdeckung eines Pixels in y-Richtung} &: \beta_y = \frac{2 \cdot \text{Öffnungswinkel in y-Richtung}}{y - \text{Auflösung}} \\ &\quad \gamma = \max(\beta_x, \beta_y) \\ \text{Gewünschter minimaler Radius} &: R_{min} = d \cdot H_{min}, \\ &\quad H_{min} = \tan(ProjectedSizeMin \cdot \gamma/2) \\ \text{Gewünschter maximaler Radius} &: R_{max} = d \cdot H_{max}, \\ &\quad H_{max} = \tan(ProjectedSizeMax \cdot \gamma/2) \end{aligned}$$

H_{min} und H_{max} können vorberechnet werden, so dass zur Bestimmung von R_{min} und R_{max} jeweils nur eine Multiplikation notwendig ist. Es sei l die bislang zugewiesene Unterteilungstiefe des Faces F. Der approximierte Radius der Subpatches von F in Tiefe l ist $R \cdot 1/2^l$. Wenn $R \cdot 1/2^l$ größer als R_{max} ist, dann wurde die Darstellung zu grob gewählt und s weitere Unterteilungsschritte sind minimal notwendig bis $R \cdot 1/2^{l+s} \leq R_{max}$ gilt. Ist hingegen $R \cdot 1/2^l$ kleiner als R_{min}, so wurde eine unnötig feine Anzeige festgelegt und die Unterteilungstiefe kann auf $l - s$ verringert werden, so dass $l - s$ maximal ist und $R \cdot 1/2^{l-s} \geq R_{min}$ gilt. Im Falle $R_{min} \leq R \cdot 1/2^l \leq R_{max}$ ist keine Veränderung des Tiefenwertes notwendig.

5.1.3 Analyse der Berechnung der Tiefenwerte

Bei der Analyse der krümmungsgetriebenen Tiefenzuweisung wird der ungünstigste Fall (*worst case*) angenommen: Alle Normalen der benachbarten Faces bleiben bis zur letzten untersuchten Unterteilungstiefe außerhalb des Normalenkegels und müssen daher bis zuletzt berücksichtigt werden. Der Cosinus des Maßes α des Krümmungskriteriums aus Formel (5.1) wurde vorberechnet. Somit ergibt sich aus Formel (5.1) folgende Anzahl von Operationen:

$$\sum_{i=0}^{\sharp M_V^0-1} (l_i+1)\cdot val_i \cdot (3 \text{ Multiplikationen} + 2 \text{ Additionen} + 1 \text{ Vergleich})$$

mit $\sharp M_V^0$ ist die Anzahl der Vertices in Kontrollnetz M^0, l_i ist der angestrebte Tiefenwert von P_i^0 und val_i ist die Valenz von P_i^0. Ein Durchlauf durch alle Faces des Ausgangskontrollnetzes sichert die maximale Tiefe pro Face und bedarf $3 \cdot \sharp M_F^0$ Vergleiche.

Die weiteren kameraabhängigen Kriterien sind zeitkritisch, da sie Bild für Bild durchgeführt werden. Die Zeit zur Tiefenberechnung kommt zur Darstellungs- und Tesselierungsdauer noch hinzu und darf daher nur einen geringen Prozentsatz an der Gesamtlaufzeit ausmachen. Sie muss also effektiv sein, damit sich der Berechnungsaufwand für eine adaptive Darstellung gegenüber einer einfachen uniformen Visualisierung lohnt.
Die Klassifizierung der Vertices respektive Faces in Front, Back und Silhouette benötigt

$$(\sharp M_V^0) \cdot (6 \text{ Multiplikationen} + 7 \text{ Additionen} + 1 \text{ Division} + 1 \text{ Wurzeloperation})$$

und maximal

$$(\sharp M_V^0) \cdot (3 \text{ Vergleiche}).$$

Die dedizierte Tiefenzuweisung erfordert bei den Front-Faces keine Rechenoperationen, bei den Silhouetten-Faces fallen zusätzlich noch je 1 Addition und 1 Multiplikation an, Back-Faces nehmen ab jetzt nicht mehr am Rendering Prozess teil.

Bei der Berücksichtigung der projizierten Größe der Patches sind zur Berechnung der jeweiligen R_{min}, R_{max} je sichtbarem Patch

$$(5 \text{ Multiplikationen} + 5 \text{ Additionen} + 1 \text{ Wurzeloperation})$$

nötig, Abstandsberechnung von Kamera zum Patch inklusive. Um für jedes sichtbare Patch einen neuen Tiefenwert aus dem bisherigen Wert l zu erhalten, sind ohne Tiefenkorrektur

$$l \text{ Multiplikationen} + 2 \text{ Vergleiche}$$

und mit Tiefenkorrektur

$$(l+s) \text{ Multiplikationen} + (s+1) \text{ Vergleiche})$$

durchzuführen.

5.1.4 Patchweise Tesselierung

Nachdem für jedes Patch seine notwendige Unterteilungstiefe bestimmt wurde (siehe Abbildung 5.4), wird jedes sichtbare Patch entsprechend seiner zugeordneten Unterteilungstiefe tesseliert. Das Ausgangskontrollnetz wird dabei nicht verändert, stattdessen wird eine separate Datenstruktur genutzt, die unabhängig von der Größe des Ausgangskontrollnetzes ist. Für jedes Patch wird dazu zunächst die 1-Nachbarschaft aus dem Ausgangskontrollnetz ausgelesen (siehe Abb. 5.5).

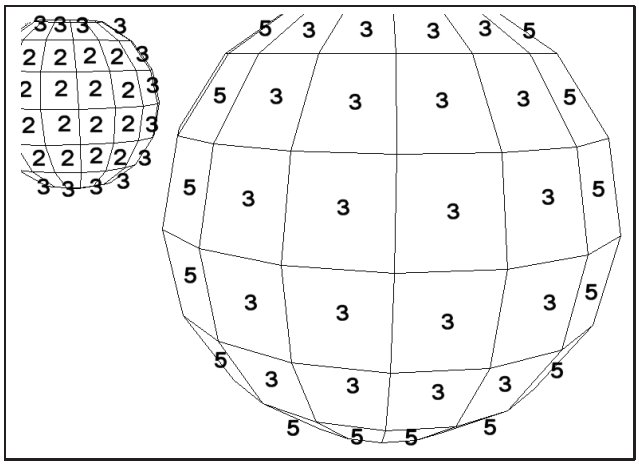


Abbildung 5.4: Resultierende Tiefenverteilung.

Zur Unterteilung bis zur angegebenen Tiefe stehen zwei verschiedene Methoden zur Verfügung:
i) Tesselierung mit Hilfe von *drei rotierenden Arrays je Unterteilungstiefe* (Abschnitt 5.1.4.1) und
ii) Tesselierung mit zwei *Slates* (Abschnitt 5.1.4.2).
Der wesentliche Unterschied der beiden rekursiven Unterteilungsverfahren besteht darin, dass in *i)* depth-first und in *ii)* breadth-first vorgegangen wird. Methode *i)* wurde von Pulli und Seagal in [PS96] mit einer uniformen Tesselierung vorgestellt. Die Methode *ii)* wurde im Rahmen dieser Arbeit entwickelt und in [SMFF03] präsentiert. Im Vergleich zur Methode von Pulli und Seagal ist Methode *ii)* leichter zu implementieren und einfacher zu erweitern im Hinblick auf adaptive Unterteilung und Einsatz von scharfen Kanten bzw. einem veränderten Regelsatz. Beide Verfahren nutzen den Umstand, dass nur die vier Eckpunkte des Ausgangsfaces irregulär sein können. Bei weiterer Unterteilung kommen nur reguläre Bereiche hinzu, so dass die zu wählende Datenstruktur lediglich in diesen vier Eckpunkten flexibel sein muss: Ist die Valenz *val* eines Eckpunktes größer als vier und damit irregulär, so werden $val - 4$ seiner benachbarten Vertices, welche nicht mehr in die reguläre Datenstruktur passen, in einem *Corner-Array* abgelegt. Auf diese Weise werden die irregulären Eckpunkte in die reguläre Datenstruktur integriert.

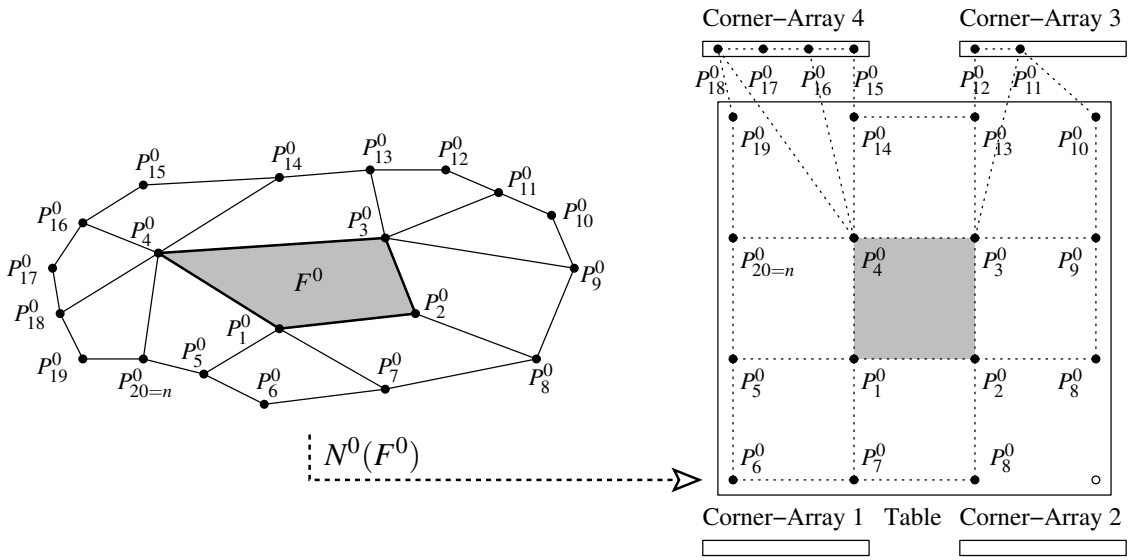
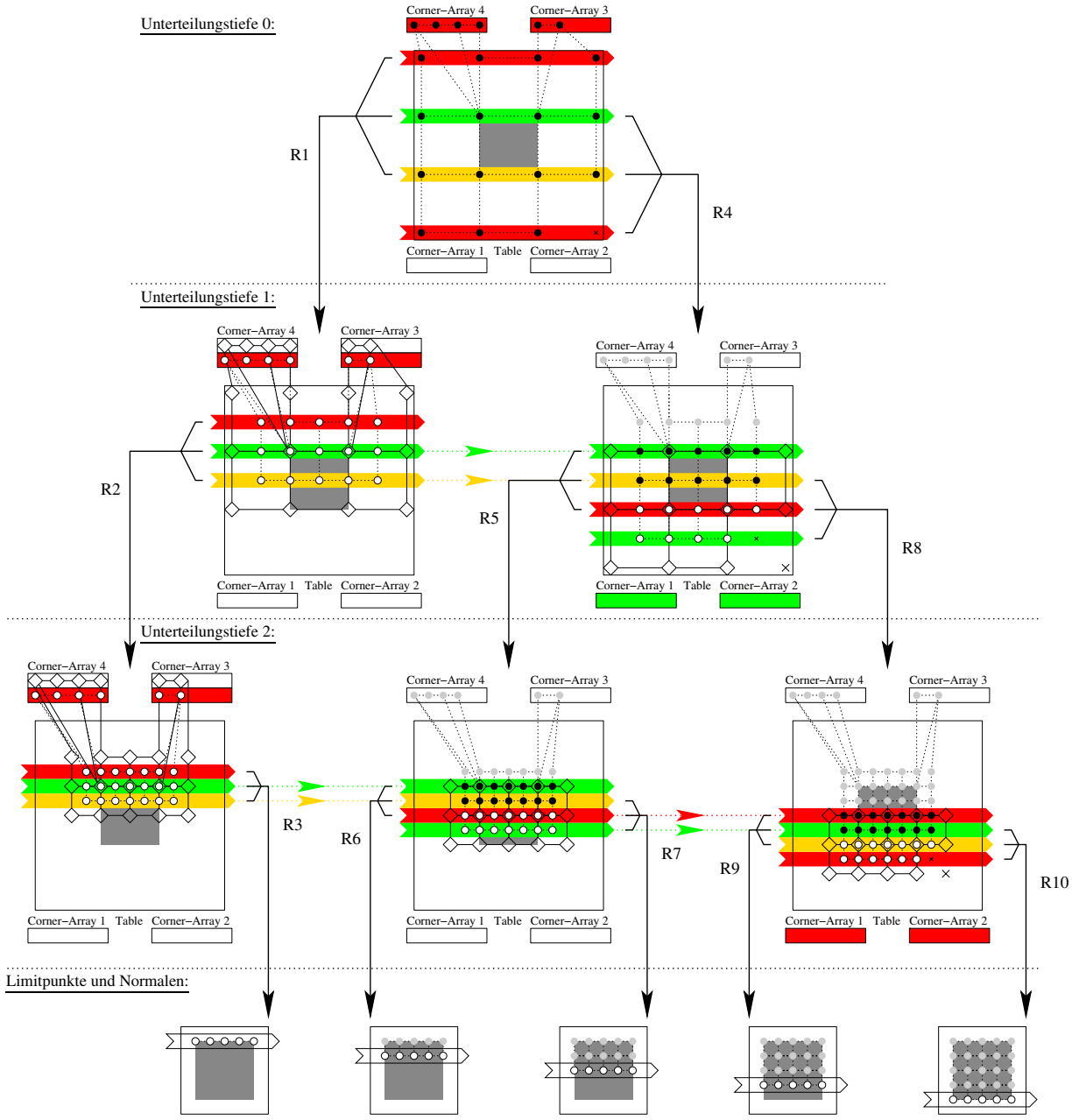


Abbildung 5.5: Aufsammeln der 1-Nachbarschaft von F^0 aus dem Ausgangskontrollnetz.

5.1.4.1 Tesselierung mit Arrays

Zur Tesselierung des zu untersuchenden Patches bis zu einer gewünschten Tiefe t mit Methode *i)* basierend auf Pulli und Seagal [PS96], werden lediglich drei Arrays pro Unterteilungstiefe benötigt. Liegt in einem der Eckpunkte des Patches eine Valenz größer als vier vor, so sind zusätzliche Corner-Arrays für jede erreichte Unterteilungstiefe nötig. Abbildung 5.6 verdeutlicht die Arbeitsweise des Algorithmus.



Legende:

- Kontrollpunkte der vorheriger Unterteilungstiefe zur Berechnung der neuen Kontrollpunkte
- neu berechnete Kontrollpunkte
- bereits berechnete Kontrollpunkte, aktuell nicht benutzt
- bereits berechnete Kontrollpunkte in Benutzung
- × fehlender Kontrollpunkt auf Grund Valenz < 4

Abbildung 5.6: Tesselierung mit Hilfe von drei Arrays je Unterteilungstiefe.

Die verwendeten drei Arrays pro Unterteilungstiefe sind mit den Farben rot, gelb und grün markiert (siehe Abbildung 5.6). Innerhalb einer Unterteilungstiefe können die Arrays überschrieben werden wie z.B. Array rot in Unterteilungstiefe 0 nach dem ersten rekursiven Aufruf. Weiterhin kann ihr Inhalt innerhalb einer Tiefe vom Nachbaraufruf wiederverwendet werden, dargestellt z.B. in Tiefe 1 durch die grünen und gelben Pfeile.

Der Algorithmus startet nach dem Aufsammeln der 1-Nachbarschaft des zu tesselierenden Patches mit der Initialisierung der drei Arrays rot (oberstes Array), grün und gelb mit den Kontrollpunkten. Es wird depth-first vorgegangen; innerhalb eines Patches wird kein Punkt doppelt berechnet. Exemplarisch wird im Folgenden eine Tesselierung bis Tiefe zwei betrachtet, wie in Abb. 5.6 angegeben. Die Corner-Arrays 4 und 3 gehören dabei zum Array rot. Die rekursiven Aufrufe R1 und R2 liefern die Erstbelegungen der drei Arrays bis zur angegebenen Tiefe $t = 2$. Bei Erreichen der Tiefe $t = 2$ wird die erste Reihe der Limitpunkte und Normalen aus den drei Arrays der Tiefe t berechnet (R3). Als Nächstes wird zurück in Ausgangstiefe $t = 0$ Array rot neu initialisiert. Der erneute rekursive Aufruf mit den drei Arrays liefert die Ausgangsdaten zur weiteren Verfeinerung: In Tiefe $t = 1$ können die bereits belegten Arrays grün und gelb nochmals genutzt werden, so dass zunächst nur Array rot mit neuen Werten überschrieben wird. Mit diesen drei Arrays erfolgt ein weiterer rekursiver Aufruf (R5): Wiederum können Array grün und gelb aus der vorherigen Berechnung in Tiefe $t = 3$ wiederverwendet werden, Array rot wird mit neuen Werten belegt. Aus diesen drei Arrays werden dann nach Aufruf R6 die nächste Reihe Limitpunkte und Normalen berechnet. Anschließend wird Array grün mit neuen Werten überschrieben und wiederum mit Aufruf R7 aus den drei Arrays eine weitere Reihe Limitpunkte und Normalen berechnet. Zurück in Tiefe $t = 1$ wird Array grün neu belegt und rekursiver Aufruf R8 gestartet. In Tiefe $t = 3$ kann nun Array rot und gelb übernommen werden, Array gelb wird neu berechnet. Nach Aufruf R8 wird die vorletzte Reihe der Limitpunkte und Normalen berechnet. Als Letztes wird Array rot in Tiefe $t = 3$ neu beschrieben und nach Aufruf R10 die letzte Reihe Limitpunkte und Normalen bestimmt.

5.1.4.2 Tesselierung mit Slates

Bei dieser Methode zur Tesselierung eines Patches bis zu einer gegebenen Tiefe besteht die zugrundelegende Datenstruktur aus zwei Slates. Ein Slate ist zusammengesetzt aus einem 2D-Array Table und vier Corner-Arrays, welche die Kontrollpunkte bei Valenz größer vier aufnehmen. Die zwei Slates sind statisch alloziiert und können für jedes zu tesselierende Patch wiederverwendet werden.

Im ersten Schritt erfolgt das Einlesen der 1-Nachbarschaft $N^0(F^0)$ des zu bearbeitenden Patches aus dem Ausgangskontrollnetz in das erste Slate. Liegen an den Eckpunkten des Faces Valenzen kleiner gleich vier vor, so wird die komplette 1-Nachbarschaft im Table des Slates abgespeichert. Treten Valenzen größer als vier auf, dann werden die restlichen Kontrollpunkte in den entsprechenden Corner-Arrays abgelegt.

Der Algorithmus **tesselate** startet mit *zu erreichende Unterteilungstiefe* und *Slate 1* als Startparameter und geht breadth-first wie folgt vor:

```
tesselate (subdivision depth, Slate i)
  if subdivision depth > 0 then
    perform one subdivision step and
    save the new points in Slate j = (i+1) mod 2;
    tesselate (subdivision depth - 1 , Slate j);
  else
    compute limit points and normals
    from Slate i;
  end if;
```

Mit Hilfe der beiden Slates werden die Kontrollpunkte für das betrachtete Patch bis zur gewünschten Unterteilungstiefe berechnet. Aus diesen Kontrollpunkten können dann die zugehörigen Limitpunkte und Normalen bestimmt werden. Das Ausgangskontrollnetz wird dabei nicht verändert; die Unterteilungen und Limitberechnungen werden lediglich mit den zwei Slates durchgeführt. Abbildung 5.7 zeigt die Arbeitsweise der zwei Slates.

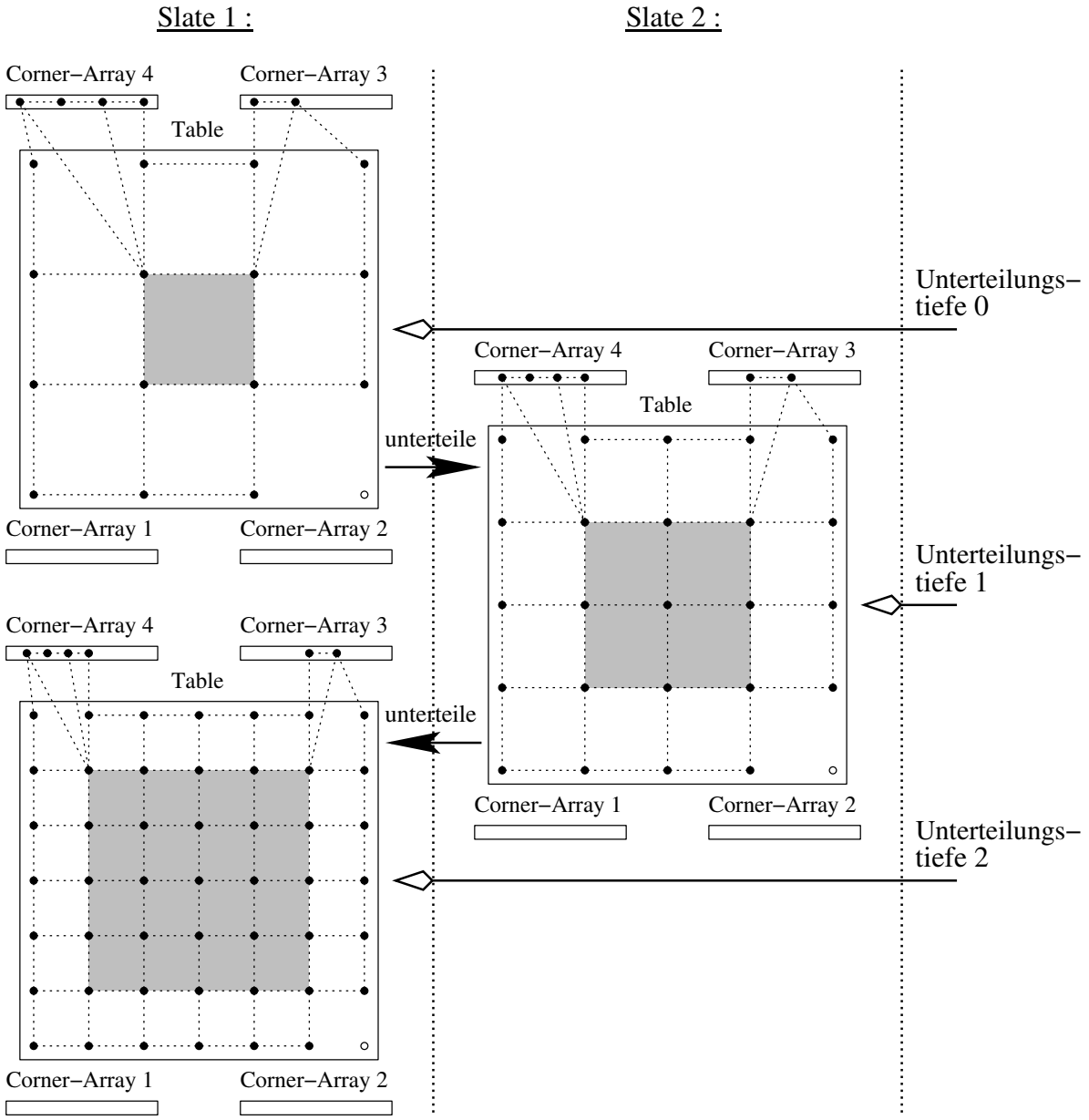


Abbildung 5.7: Tesselierung mit Hilfe von zwei Slates.

5.1.5 Lückenschließung

Auf Grund der adaptiven Tesselierung der Unterteilungsfläche, können Nachbarpatches unterschiedliche Unterteilungstiefen erhalten. Abbildung 5.8 zeigt ein Beispiel in schematischer Form. Zwischen den verschieden tesselierten Patches können Lücken auftreten, gekennzeichnet mit einer gestrichelten Linie. Um solche Lücken zu vermeiden, wird die Tesselierung des Patches mit geringerer Unterteilungstiefe an

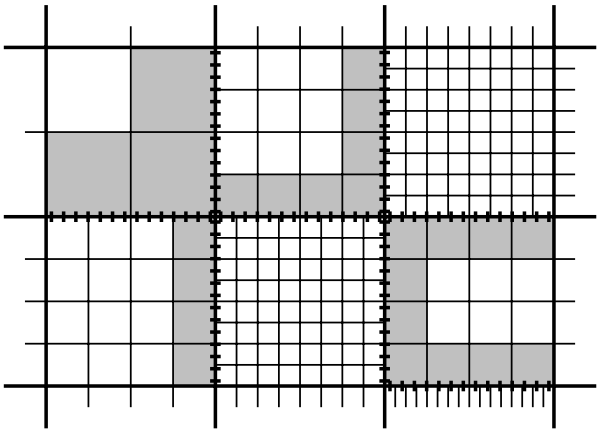


Abbildung 5.8: Schematische Darstellung von sechs Patches in unterschiedlichen Unterteilungstiefen: Lücken können an den gestrichelten Linien entstehen, daher müssen die grau markierten Vierecke durch eine geeignete Tesselierung ersetzt werden.

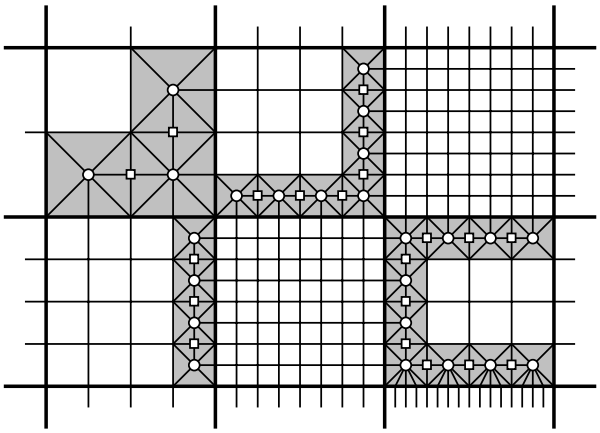


Abbildung 5.9: Schematische Darstellung von sechs Patches in unterschiedlichen Unterteilungstiefen nach der Lückenbeseitigung.

die Tesselierung des Patches mit höherer Unterteilungstiefe angepasst. Dazu werden die End-Vierecke, welche am Rand zum feiner tesselierten Nachbarn liegen (in Abbildung 5.8 grau markiert) durch einen zum Nachbarn anschließenden geeigneten Triangle-Fan ersetzt (siehe Abbildung 5.9).

Ein solcher Triangle-Fan wird gebildet aus dem Facepunkt (markiert durch einen kleinen Kreis), hervorgegangen aus einem weiteren Unterteilungsschritt des betrachteten Vierecks, den Eckpunkten des Vierecks und den Punkten, welche auf dem Rand zum tiefer tesselierten Nachbarn liegen. Wurden die benachbarten End-Vierecke im selben Patch auf gleiche Weise ersetzt, so wird die gemeinsame Kante unterteilt mit Hilfe eines Unterteilungsschrittes und der resultierende Punkt (markiert mit einem kleinen Quadrat) im Triangle-Fan eingefügt. Der zusätzliche Unterteilungsschritt zur Berechnung der Face- und Kantenpunkte ist notwendig, um einen glatten Übergang zu erreichen. Abbildung 5.10 (mitte) zeigt eine Lückenschließung ohne weiteren Unterteilungsschritt: Als Facepunkt wurde der Mittelpunkt des End-Vierecks gewählt, weitere Kantenpunkte wurden nicht eingefügt. Durch dieses Vorgehen können ungewünschte Dellen und Kanten in der Oberfläche auftauchen. Dem gegenüber zeigt Abbildung 5.10 (rechts) ein Beispiel für das verwendete Verfahren mit glatter Lückenschließung.

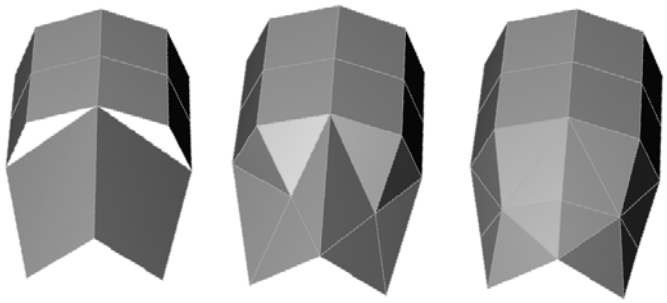


Abbildung 5.10: Lücke durch unterschiedlich tesselierte Patches (links). Lückenschließung ohne zusätzlichen Unterteilungsschritt: Artefakte in Form von ungewollten Dellen und Kanten am Übergang (mitte). Lückenschließung mit zusätzlichem Unterteilungschritt: Glatter Übergang (rechts).

5.1.6 Ausgabe der berechneten Werte

Nachdem jedes Face seine spezifische Unterteilungstiefe erhalten hat (siehe Abschnitt 5.1.2), übernehmen die beiden Slates patchweise die Berechnung der Kontrollpunkte bis zur angegebenen Tiefe des Patches (siehe Abschnitt 5.1.4.2). Aus diesen Kontrollpunkten lassen sich für jedes Patch die Limitpunkte sowie Normalen bestimmen.

Dabei können zwei Fälle auftreten (siehe Abbildung 5.11): Im Fall a) sind die Unterteilungstiefen der Kantennachbarn kleiner gleich der Unterteilungstiefe des betrachteten Patches. Eine Lückenschließung zu den Nachbarpatches ist somit nicht notwendig. Die Limitpunkte und Normalen können direkt aus den

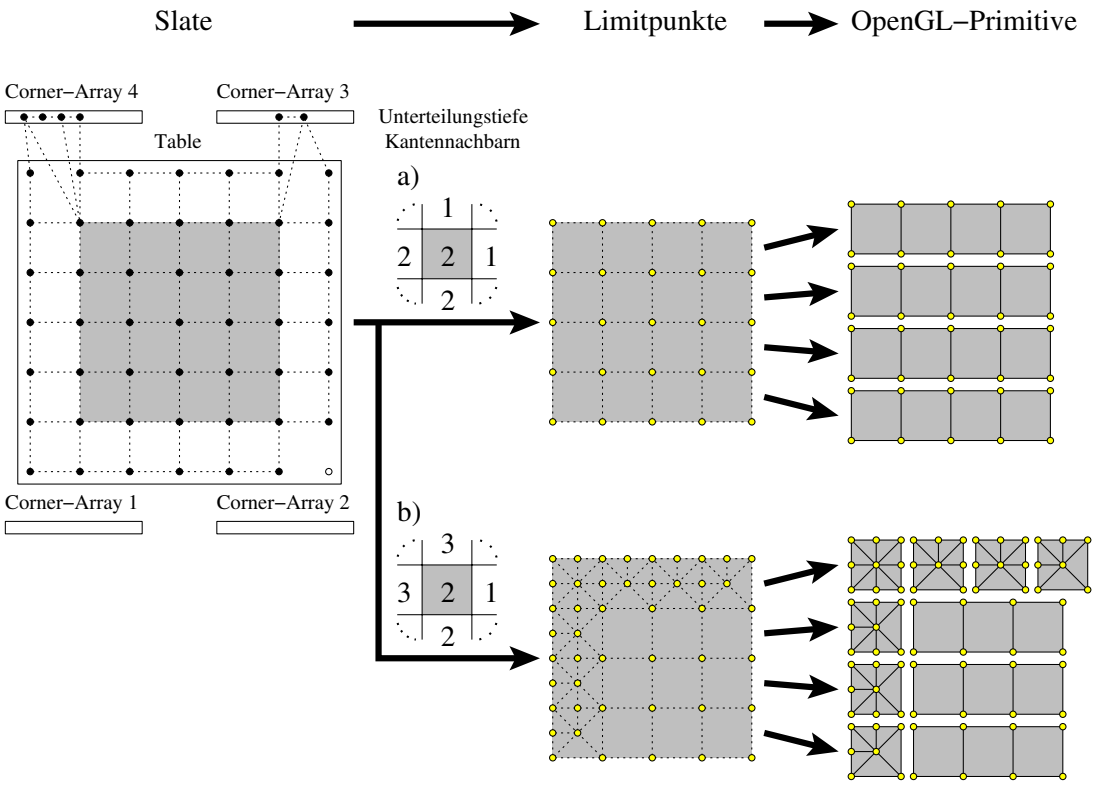


Abbildung 5.11: Ausgabe mit OpenGL-Primitiven.

Kontrollpunkten des Slates in der gewünschten Tiefe berechnet werden. Wegen der regulären Anordnung der Limitpunkte bietet es sich zur Ausgabe mit OpenGL an, Quad-Strips als OpenGL-Primitive zu verwenden.

Bei Fall *b*) hat mindestens ein Kantennachbar des aktuellen Patches eine größere Unterteilungstiefe als das aktuelle Patch. Eine Lückenschließung ist somit erforderlich. Wie im vorigen Abschnitt 5.1.5 beschrieben, müssen zur Lückenschließung zusätzliche Limitpunkte berechnet werden. Bei der OpenGL Ausgabe müssen die betroffenen Vierecke im Randbereich durch Triangle-Fans ersetzt werden und die Quad-Strips entsprechend verkürzt ausgegeben werden.

Statt die Limitpunkte und Normalen immer wieder neu on-the-fly zu berechnen, ist ein Caching sinnvoll. Die bereits berechneten Limitpunkte und Normalen werden dazu für jedes Patch in einem 2D-Array abgelegt. Stehen die Limitpunkte nicht in ausreichender Tiefe für ein Patch zur Verfügung, so können die fehlenden Punkte und Normalen mit Hilfe der Slates neu berechnet werden. Liegen die Punkte und Normalen im Cache in Tiefe größer gleich der angeforderten Unterteilungstiefe vor, dann können die gewünschten Limitpunkte und Normalen wie in Abbildung 5.12 beschrieben aus dem Cache entnommen werden.

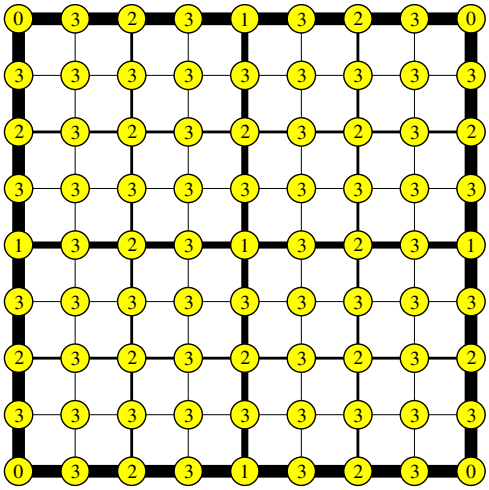


Abbildung 5.12: Caching der Limitpunkte eines Patches in einem 2D-Array in Tiefe 3 (schematische Darstellung). Die Nummern auf den Limitpunkten geben die Unterteilungstiefe an, in der sie berechnet wurden. Zur Darstellung des tesselierten Patches in Unterteilungstiefe $l \leq 3$ werden die Limitpunkte mit Nummer $0, \ldots, l$ genutzt.

5.1.7 Analyse der Laufzeit und des Speicherverbrauchs

Bei der Analyse der Laufzeit und des Speicherplatzverbrauches gilt die Voraussetzung, dass die maximale Unterteilungstiefe eines Patches kleiner gleich *maxsd* ist und die maximale Vertexvalenz des Kontrollnetzes *maxval* nicht überschreitet.

5.1.7.1 Analyse des Speicherverbrauchs

Für den gesamten Unterteilungsprozess des kompletten Modells bis zur maximal erlaubten Tiefe *maxsd* reichen die zwei statisch alloziierten Slates aus. Ein Slate besteht aus einem 2D-Array mit

$$(2^{maxsd}+3)^2 \qquad \text{3D-Vektoren}$$

und vier Corner-Arrays mit

$$(maxval-4)\cdot 2 \qquad \text{3D-Vektoren.}$$

Damit enthält ein Slate

$$(2^{maxsd}+3)^2+4\cdot(maxval-4)\cdot 2$$

3D-Vektoren. Insgesamt ergeben sich somit für den kompletten Unterteilungsprozess mit zwei Slates ein Bedarf von

$$2\cdot((2^{maxsd}+3)^2+4\cdot(maxval-4)\cdot 2)\cdot 3$$

Floats. Diese Anzahl ist unabhängig von der Größe des Kontrollnetzes und hängt nur von der maximalen Unterteilungstiefe und von der maximalen Vertexvalenz ab. Die berechneten Limitpunkte und Limitnormalen können optional gecachet werden. In diesem Fall ist für die gecachten Limitpunkte und Limitnormalen zusätzlich Speicherplatz nötig.

5.1.7.2 Analyse der Laufzeit

Um ein Patch in Unterteilungstiefe l darzustellen, sind die Kontrollpunkte bis Tiefe l sowie $n=(2^l+1)^2$ Limitpunkte und die selbe Anzahl an Limitnormalen zu berechnen. Für die dazu benötigte Laufzeit gilt:

Lemma 4. *Es sei M^0 eine Viereckskontrollnetz und F ein Face aus M^0 mit Unterteilungstiefe l. Die Laufzeit zur Berechnung der Limitpunkte und Limitnormalen von F in Unterteilungstiefe l ist linear zur Anzahl $n=(2^l+1)^2$ der Limitpunkte $O(n)$.*

Beweis:
Die Laufzeit zur Berechnung eines neuen Vertex, eines Limitpunktes sowie einer Limitnormale hängt von der Valenz des Vertex ab. Da für die Valenz des Vertex gilt:

$$\text{Valenz von Vertex } V \leq maxval,$$

existiert eine obere Schranke Z mit

$$A \in \{\text{neuer Vertexpunkt}, \text{neuer Kantenpunkt}, \text{neuer Facepunkt}, \text{Limitpunkt}, \text{Limitnormale}\}$$

und

$$\text{Laufzeit zur Berechnung von } A \leq Z.$$

Um die Laufzeit zur Berechnung der Limitpunkte und Limitnormalen in Tiefe l von Face $F \in M^0$ zu bestimmen, muss die Anzahl SN aller dazu notwenigen Unterteilungspunkte ermittelt werden:

$$SN=\sum_{i=1}^{l}((2^i+3)^2+C) \tag{5.2}$$

mit $C=$ Anzahl der Vertices in den vier Corner-Arrays des Slates. Wegen $C \leq 4\cdot(maxval-4)\cdot 2$ lässt sich SN abschätzen durch:

$$SN \leq \sum_{i=1}^{l}((2^i+3)^2+4\cdot(maxval-4)\cdot 2)$$

Aus diesen Kontrollpunkten können die LN Limitpunkte und Limitnormalen berechnet werden:

$$LN=2\cdot(2^l+1)^2 \tag{5.3}$$

Somit ergibt sich die Gesamtanzahl CN der zu berechnenden Punkte und Normalen aus Formel (5.2) und Formel (5.3):

$$\begin{aligned} CN &= SN + LN \\ &= \sum_{i=1}^{l}((2^i+3)^2+C)+2\cdot(2^l+1)^2 \\ &= \frac{10}{3}\cdot 2^{2l}+16\cdot 2^l+l\cdot(9+C)-\frac{34}{3} \\ &\leq \frac{10}{3}\cdot 2^{2l}+16\cdot 2^l+l\cdot(9+4\cdot(maxval-4)\cdot 2)-\frac{34}{3} \end{aligned}$$

und damit eine obere Schranke für die Laufzeit LZ zur Berechnung der Limitpunkte und Limitnormalen:

$$LZ \leq (\frac{10}{3}\cdot 2^{2l}+16\cdot 2^l+l\cdot(9+4\cdot(maxval-4)\cdot 2)-\frac{34}{3})\cdot Z$$

Dies ist linear zur Anzahl $n=(2^l+1)^2$ der Limitpunkte:

$$O(\frac{10}{3}\cdot(\sqrt{n}-1)^2+16\cdot(\sqrt{n}-1)+(9+C)\cdot\ln(\sqrt{n}-1)-\frac{34}{3}) = O(n)$$

■

5.1.8 Messergebnisse

Abbildung 5.13: Testszene I: Das Kontrollnetz besteht aus 3932 Faces. Modelle von Volker Settgast, Matthias Richter.

Das beschriebene Verfahren adaptive Tesselierung mit Slates ist in OpenSG [RVB02, Opea] integriert worden. Ziel des 1999 gegründeten OpenSG Projektes ist es, ein neues verbessertes Szenengraph-System in OpenSource-Form zu entwickeln. OpenSG setzt auf OpenGL auf und ist gleichermaßen plattformübergreifend einsetzbar. Mit der Integration des Verfahrens in OpenSG stehen die Vorteile eines Szenengraphsystems zur Verfügung, wie z.B. Anbindung an eine Cave und Nutzung eines Rendering Clusters ohne zusätzlichen Aufwand. Im Vergleich ist die Performance bei der direkten Ausgabe mit OpenGL etwas besser als in OpenSG (siehe Abbildung 5.14). Die Vorteile eines Szenengraph Systems überwiegen jedoch, so dass eine etwas schlechtere Performance in Kauf genommen wird. Daher beziehen sich die nachfolgenden Messreihen in den Abbildungen 5.18 und 5.20 auf die OpenSG Version.

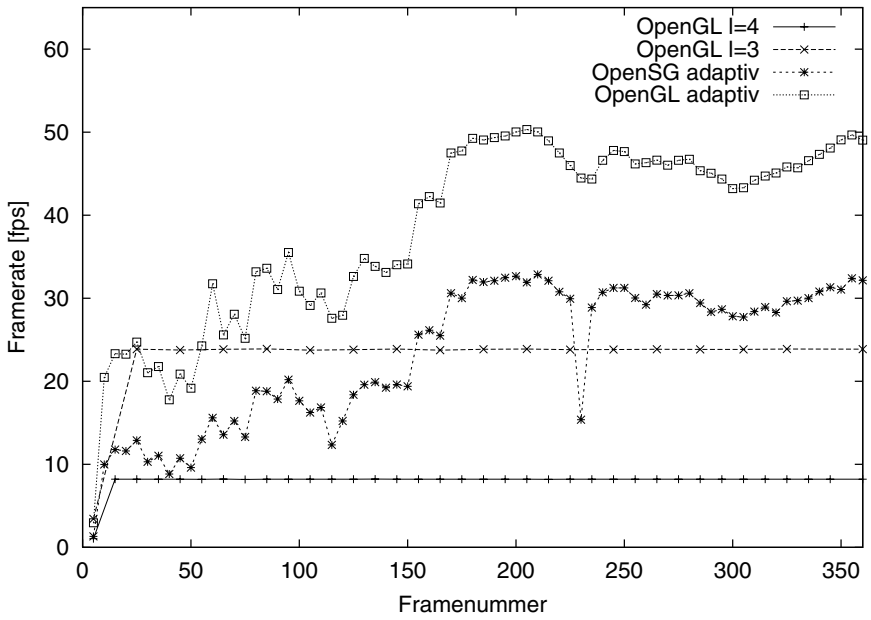


Abbildung 5.14: Frameraten für Testszene I (siehe Abbildung 5.13) mit Unterteilungstiefen 0 bis 4 bei den adaptiven Verfahren. Auf Grund des höheren Verwaltungsaufwandes eines Szenengraphsystemes ist die direkte Ausgabe der adaptiven Tesselierung mit OpenGL schneller als die Ausgabe mit OpenSG. Im Vergleich zur uniformen Tesselierung mit direkter OpenGL Ausgabe gewinnen beide adaptiven Versionen bei gleicher Darstellungsqualität. Messung von Volker Settgast.

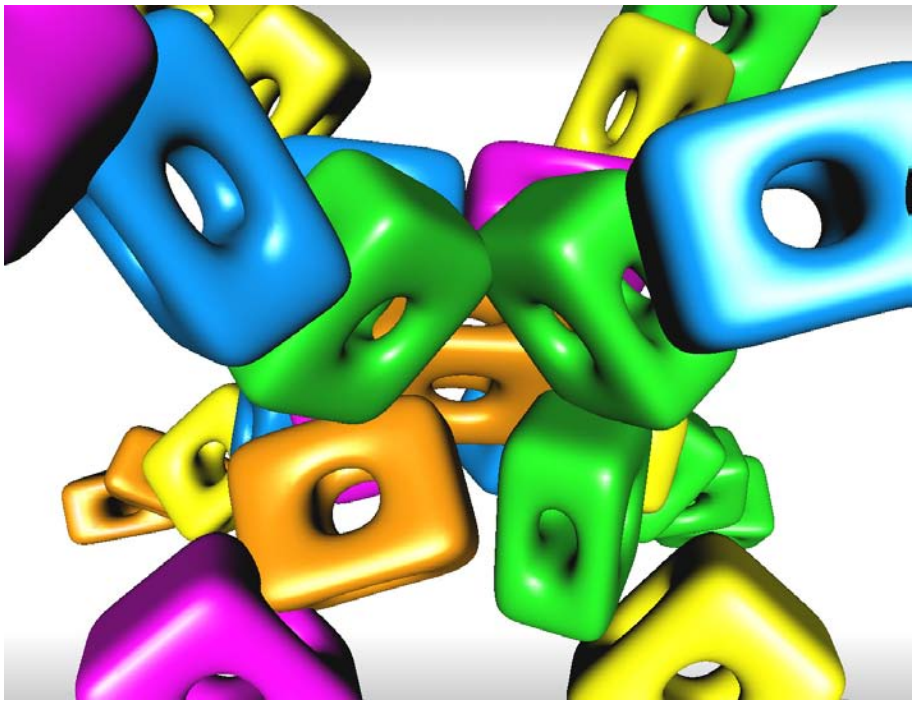


Abbildung 5.15: Testszene II: 40 Instanzen eines Objektes mit jeweils 68 Faces in sternförmiger Anordnung. Modell von Volker Settgast aus [Set04].

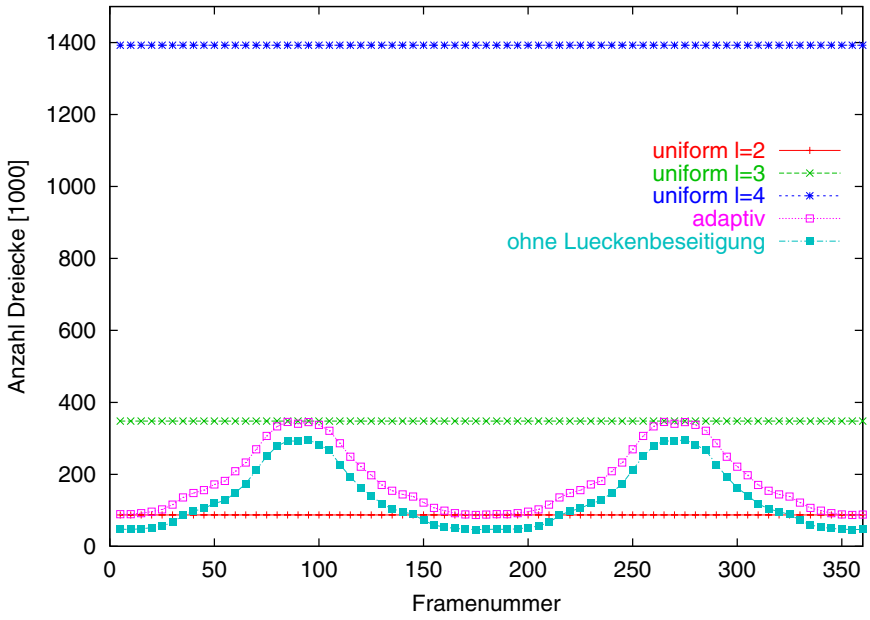


Abbildung 5.16: Anzahl der darzustellenden Dreiecke bei zweifacher Rotation um die Testszene II (siehe Abbildung 5.15) mit variablem Abstand zwischen Kamera und Testszene bei adaptiver und uniformer Tesselierung. Messung von Volker Settgast aus [Set04].

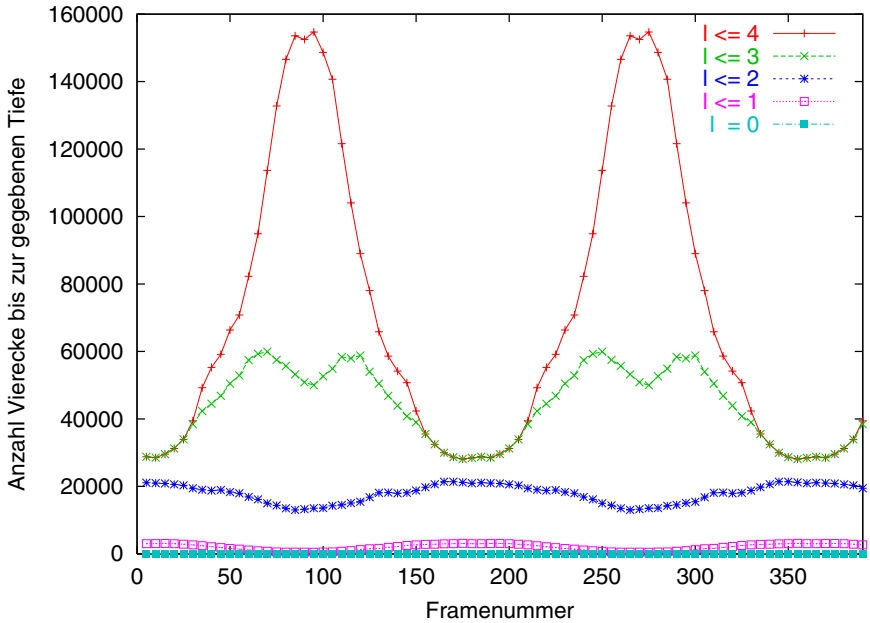


Abbildung 5.17: Verteilung der Unterteilungstiefen bei der adaptiven Tesselierung ohne Lückenschließung bezüglich der Messreihe aus Abbildung 5.16. Messung von Volker Settgast aus [Set04].

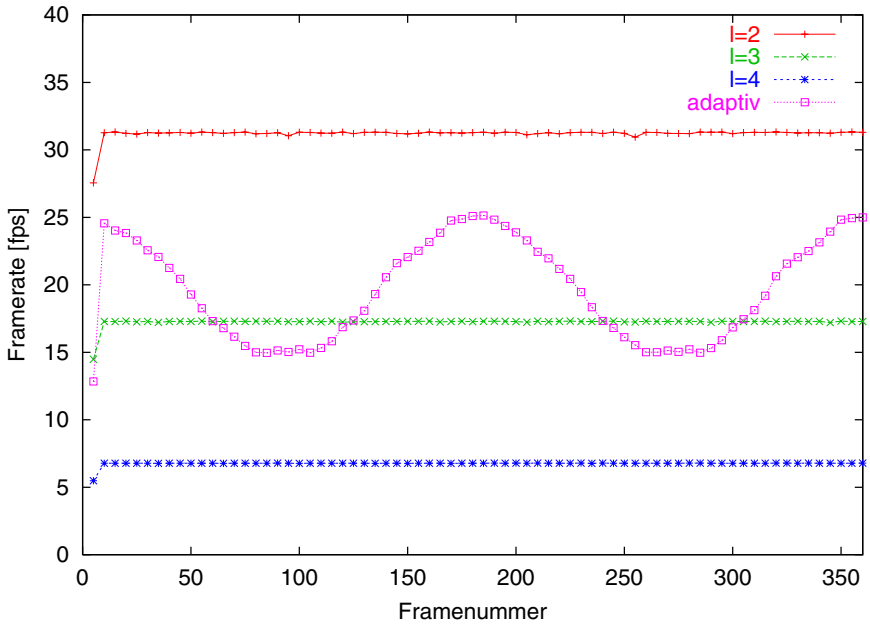


Abbildung 5.18: Frameraten zur Testszene II (siehe Abbildung 5.15) bezüglich der darzustellenden Dreiecke aus der Messreihe von Abbildung 5.16. Messung von Volker Settgast aus [Set04].

Alle Testreihen wurden auf einem Standard PC mit Intel P4 1.7 Ghz, 512 MB Arbeitsspeicher, Geforce 4 TI 4600 Graphikkarte durchgeführt. Die Testszene besteht aus 40 Instanzen eines Objektes mit jeweils 68 Faces im Kontrollnetz. Die Objekte sind sternförmig angeordnet (siehe Abbildung 5.15). Während der Laufzeitmessung wird die Kamera zweimal in jeweils 180 Frames um die Testzene rotiert, wobei der Abstand der Szene zur Kamera variiert um die Korrektheit der adaptiven Tesselierung zu überprüfen.

Wie bei OpenGL üblich, wird ein Viereck intern durch zwei Dreiecke ersetzt. Abbildung 5.16 zeigt daher die Anzahl der darzustellenden Dreiecke im Verlauf der Kamerarotation. Bei der uniformen Tesselierung bleibt die Anzahl der zu visualisierenden Dreiecke konstant, bei der adaptiven Methode schwankt die Anzahl je nach Kameraposition zur Szene. Verringert sich die Distanz der Objekte zur Kamera, so werden die Objekte feiner dargestellt und eine größere Dreiecksanzahl ist zur Repräsentation der Szene notwendig (siehe z.B. Frame 270 mit ca. 350.000 Dreiecken). Bei größerer Distanz reicht eine gröbere Darstellung aus (siehe z.B. Frame 180 mit 150.000 Dreiecken). Zur besseren Differenzierung ist in Abbildung 5.16 die Dreiecksanzahl ohne Lückenfüllen angeführt. Diese Dreiecksanzahl resultiert aus den Quad-Strips der tesselierten Patches, die bis zur jeweiligen bestimmten Tiefe unterteilt wurden. Der adaptive Algorithmus wählt bei dieser Testreihen nach den Gegebenheiten der Szene Unterteilungstiefen zwischen null und vier individuell für jedes Patch (siehe Abschnitt 5.1.2) aus. Analog zum Anstieg der Dreiecksanzahl in Abbildung 5.16 zeigt sich in Abbildung 5.17 bei Verringerung der Distanz zwischen Szene und Kamera ein Anstieg der Viereckszahl in den höheren Unterteilungstiefen, ebenso reichen bei größerem Abstand niedrigere Unterteilungstiefen aus.

Von der Anzahl der Dreiecke respektive Vierecke hängt die Laufzeit direkt ab. Abbildung 5.18 gibt die Laufzeit für die jeweiligen Verfahren an. Die Laufzeit des adaptiven Verfahrens schwankt in Abhängigkeit der Dreiecksanzahl zwischen 15 und 25 Bilder pro Sekunde. Um vergleichbare Bildqualität mit uniformer Tesselierung zu erreichen ist mindestens Tiefe 3 nötig (siehe auch Tiefenverteilung in Abbildung 5.17), was zu einer deutlich schlechteren Performance im Vergleich zum adaptiven Ansatz führt.

5.1.9 Frameratenkontrolle

Zur Vermeidung der so genannten *Simulator Sickness* [HKB03] ist eine konstante Framerate erforderlich. Eine Schwankung in der Framerate kann auftreten durch eine Veränderung der Anzahl der Primitive, die zur Graphikhardware gesendet werden. Tritt beispielsweise ein detailreiches Objekt in den Fokus der interaktiven Betrachtung, so verringert sich die Framerate solange das Objekt im Fokus bleibt. Eine Frameratenkontrolle verbunden mit einer adaptiven Visualisierung sorgt für eine qualitativ hochwertige Darstellung der Szene bei gleichzeitiger Einhaltung einer vorgegebenen Framerate.

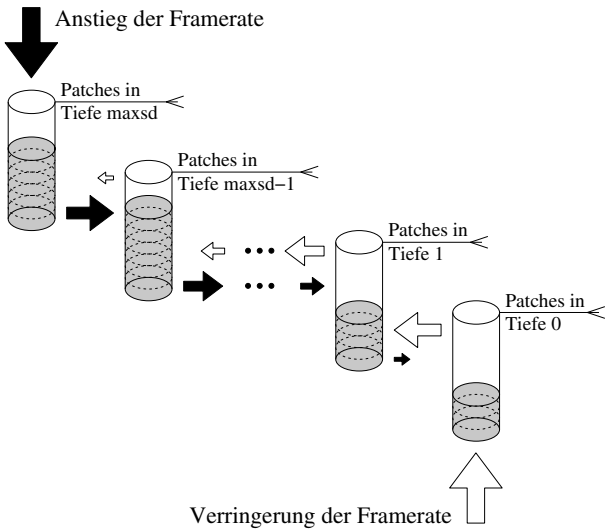


Abbildung 5.19: Frameratenkontrolle mit Feedback-Mechanismus.

Eine einfache Methode zur Vermeidung der Schwankungen und zur Kontrolle der Framerate ist der Feedback-Mechanismus [FS93]. Dazu wird die Zeit für einen Frame gemessen. Ist der Frame zu lang, so wird die Anzahl der Primitive reduziert indem die Objekte gröber dargestellt werden. Ist hingegen der Frame zu kurz und somit freie Zeit vorhanden, so können die Objekte in verfeinerter Auflösung präsentiert werden. Alternativ kann die verfügbare Zeit mit einer Warteschleife aufgefüllt werden.

Unterteilungsflächen liefern durch ihre Konstruktionsweise eine optimale Steuerung ihrer Primitivenanzahl (siehe Abbildung 5.19). Im Falle einer zu geringen Framerate findet eine dynamische Vergröberung der Unterteilungsfläche statt: Patches mit höchster zugeordneter Unterteilungstiefe $maxtiefe \leq maxsd$ werden sukzessive $maxtiefe - 1$ zugeordnet bis die Framerate wieder im gewünschten Bereich ist. Sollte die Reduzierung der Tiefe von $maxtiefe$ nach $maxtiefe - 1$ nicht ausreichen, so wird mit einer Reduktion von $maxtiefe - i$ nach $maxtiefe - i - 1$, mit $i = 1, \ldots, maxtiefe - 1$, fortgefahren. Steht wieder mehr Zeit zur Verfügung, so wird sukzessive die Tiefenreduktion zurückgenommen. Liegt die Framerate über dem angestrebten Bereich, so können Patches mit der kleinsten Unterteilungstiefe $mintiefe \geq 0$ schrittweise einer höheren Unterteilungstiefe zugeordnet werden. Die verbleibende Zeit zur Einhaltung der Framerate kann zusätzlich mit einer Warteschleife gefüllt werden, bis die optimale Darstellungsqualität gefunden wurde.

Abbildung 5.20 zeigt eine Messreihe mit und ohne Frameratenkontrolle für die Testszene II (siehe Abbildung 5.15). Für die ersten 600 Bilder wurde eine Framerate von 14 bis 15 Bilder pro Sekunde vorgegeben. Um diese Framerate einzuhalten und optimal auszunutzen strebt der Algorithmus in einem dynamischen Prozess die maximale Unterteilungstiefe 4 an. Die nachfolgenden Bilder > 600 erhielten als Framerate die Vorgabe 39 bis 40 Bilder pro Sekunde zu erreichen. Daraus resultierte eine Tiefenverteilung von 0 bis 2. Beim Umschalten zwischen den beiden Frameraten ergab sich eine Anpassungsphase von ca. 100 Bilder bzw. ca. 3 Sekunden. Insgesamt gesehen wurden die vorgegebenen Frameraten mit akzeptabler Präzision eingehalten.

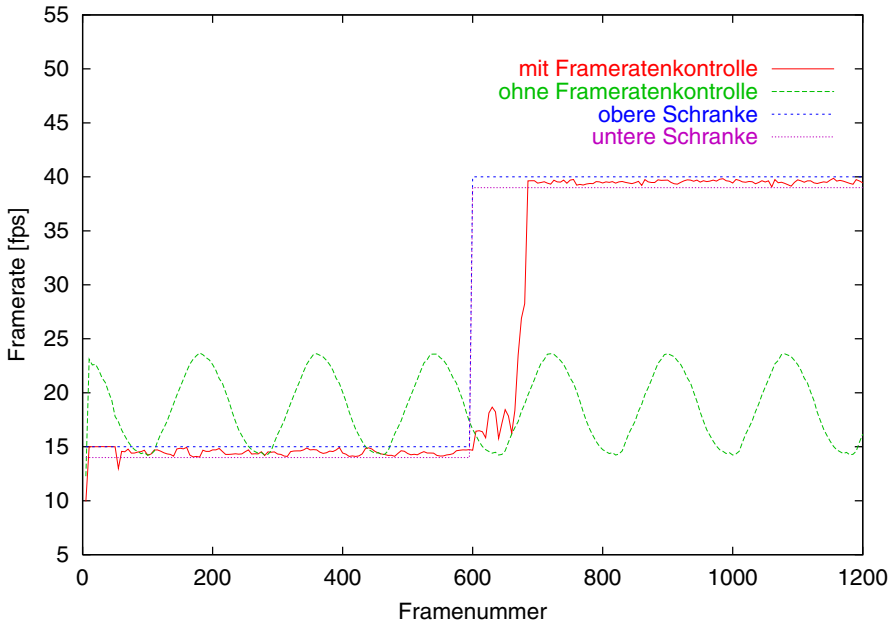


Abbildung 5.20: Messung zur Frameratenkontrolle mit Testszene II (siehe Abbildung 5.15). Um die Testszene wurde mehrfach rotiert. Messung von Volker Settgast aus [Set04].

5.1.10 Visuelle Ergebnisse

Abbildung 5.21 demonstriert die adaptive Arbeitsweise des Algorithmus. Zur Verdeutlichung der adaptiven Tesselierung ist das Objekt im Wireframe-Modus dargestellt, Abbildung 5.22 zeigt das gleiche Objekt mit Smooth-Shading. Im Wireframe-Modus ist deutlich zu erkennen, dass Patches im Silhouettenbereich sowie stärker gekrümmte Objektteile mit höherer Unterteilungstiefe angezeigt werden. Zwischen den unterschiedlich tesselierten Patches wurde eine Lückenschließung durchgeführt um Risse im Objekt zu verhindern.

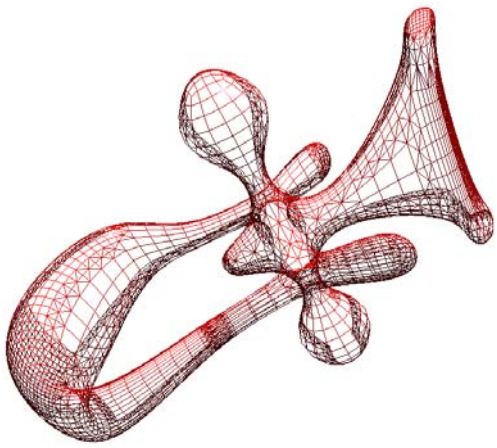


Abbildung 5.21: Unterteilungsfläche in Wireframe-Darstellung zur Verdeutlichung der adaptiven Tesselierung. Modell von Volker Settgast.

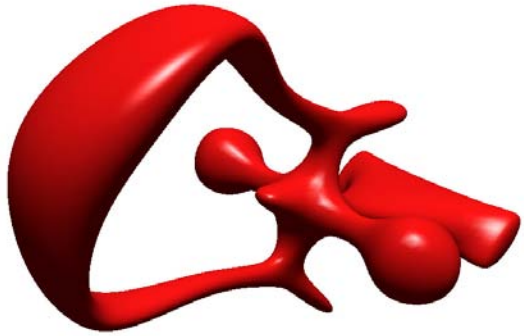


Abbildung 5.22: Gleiches Objekt im Smooth-Shaded-Modus.

Abbildung 5.23: Einfaches Unterteilungsflächenmodell mit scharfen Kanten.

Abbildung 5.24: Effekt der Backface-Culling-Heuristik des adaptiven Verfahrens: Links das originale Bild, rechts die Auswirkung der Backface-Culling-Heuristik demonstriert durch eine zweite, rückseitige Kamera. Modell von Volker Settgast.

Den Effekt des Aussortierens kameraabgewandter Patches (Backface-Culling) zeigt Abbildung 5.24: Im Bild 5.24 links werden die nicht sichtbaren Patches des Objektes weggelassen. Dies verifiziert eine zweite Kamera im Bild 5.24 rechts aus rückseitiger Perspektive. Die Anzahl der darzustellenden Dreiecke kann somit merklich reduziert werden, wodurch sich die Performance der Applikation erhöht.

Abbildung 5.26 zeigt ein Beispiel für eine Applikation mit Unterteilungsflächen in OpenSG. In Abbildungen 5.25 sind Loop-Unterteilungsflächen verwendet worden, welche Dreieckskontrollnetze verwenden. Durch die Integration der interaktiven Darstellung von Unterteilungsflächen mittels eines adaptiven Verfahrens in OpenSG stehen die Möglichkeiten eines Szenengraphsystemes zur Verfügung.

Abbildung 5.25: Loop-Unterteilungsfläche mit OpenSG. Freies Modell von Viewpoint Animation Engineering.

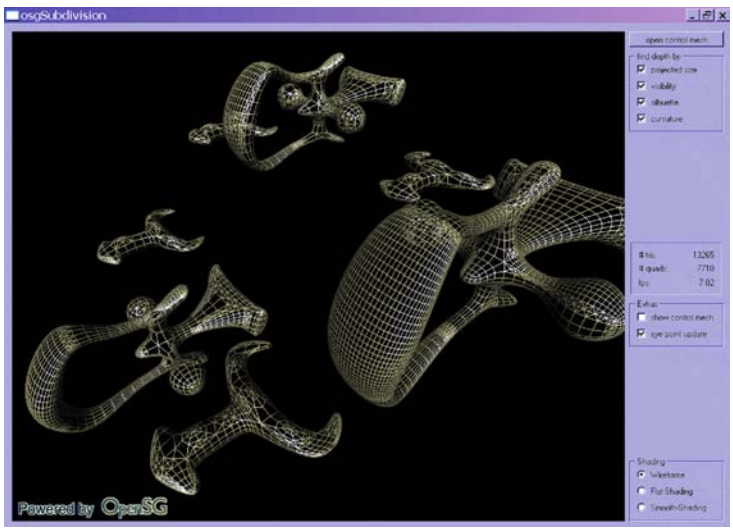


Abbildung 5.26: OpenSG-Anwendung mit Catmull-Clark-Unterteilungsflächen. Modelle von Volker Settgast, Matthias Richter.

Abbildung 5.27 zeigt eine OpenSG-Applikation mit Unterteilungsflächen in der Cave. Weitere Anwendungen mit Stereo-Darstellung sowie die Nutzung von Render-Cluster sind mit OpenSG ohne Zusatzaufwand möglich.

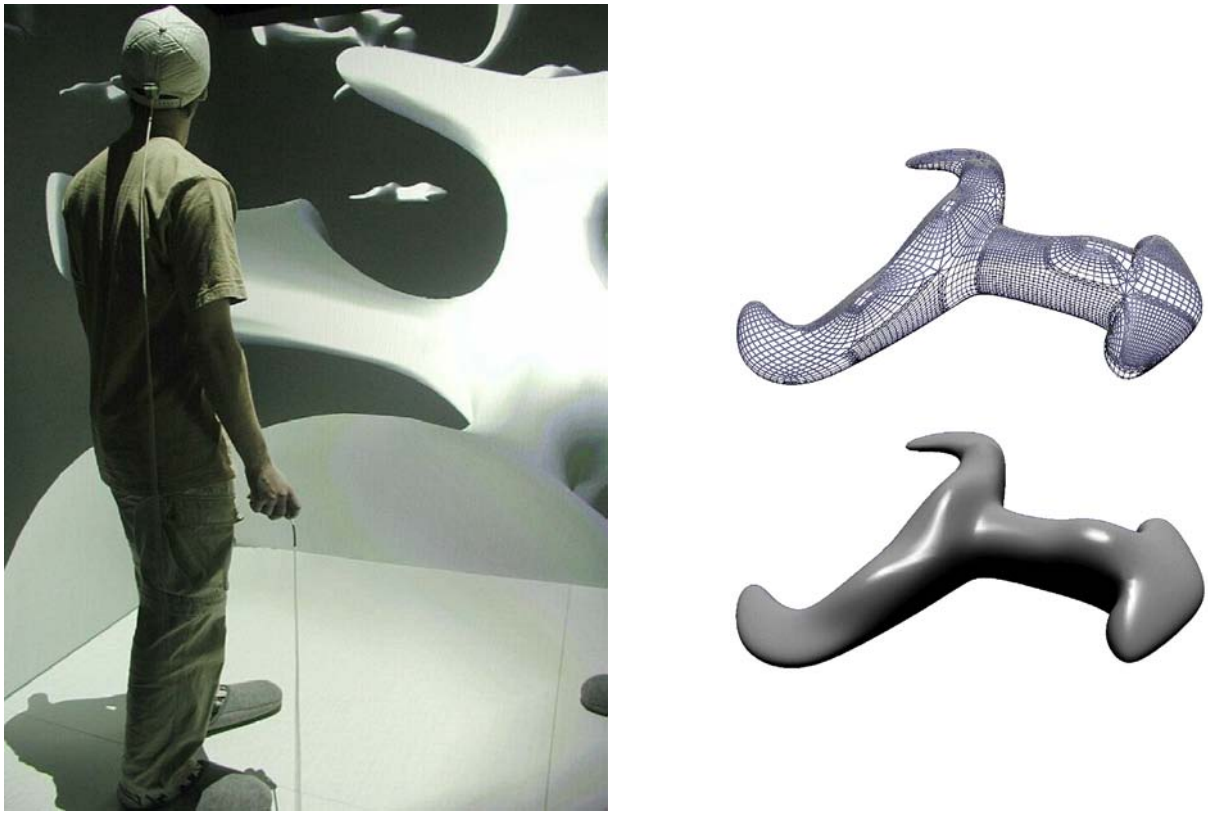


Abbildung 5.27: Unterteilungsflächen in der Cave mit OpenSG. Modelle von Volker Settgast, Matthias Richter. Fotografie von Norbert Schenk, der Anwender in der Cave ist Lars Reusche.

5.1.11 Übertragung auf Dreieckskontrollnetze

Das vorgestellte Verfahren zur interaktiven Visualisierung von Unterteilungsflächen, basierend auf Viereckskontrollnetze, ist auch übertragbar auf Unterteilungsflächen mit Dreiecksnetzen (siehe Abbildung 5.25). Damit ist es möglich, Loop-Unterteilungsflächen on-the-fly darzustellen. Abschnitt 5.1.11.1 zeigt die Grundidee, in 5.1.11.2 wird der nötige Preprocessing-Schritt demonstriert. Abschnitt 5.1.11.3 erläutert schließlich die Übertragung des Algorithmus vom Vierecks- zum Dreieckskontrollnetz.

5.1.11.1 Grundidee

Um ein Dreieckskontrollnetz mit dem beschriebenen Verfahren zu bearbeiten, werden in einem Vorverarbeitungsschritt zwei kantenbenachbarte Dreiecke zu einem Pseudo-Viereck zusammengefasst. Diesen Paarfindungsschritt erledigt ein Greedy-Algorithmus, welcher in der Praxis gute Ergebnisse liefert. Der Algorithmus generiert eine Liste von Pseudo-Vierecken und eine prozentual geringe Anzahl von Einzel-Dreiecken. Auf Grund der geringen Anzahl der Einzel-Dreiecke wird für diese Dreiecke kein Sonderfall generiert. Stattdessen erhalten sie einen Dummy-Partner zugeordnet, der eine analoge Weiterverarbeitung ermöglicht. Die Einzel-Dreiecke mit ihrem Dummy-Partner werden dann ebenfalls der Pseudo-Viereckliste beigefügt. Der Algorithmus zur interaktiven Visualisierung von Dreieckskontrollnetzen arbeitet mit dieser Pseudo-Viereckliste analog zum Viereckskontrollnetzverfahren.

5.1.11.2 Paarfindungsalgorithmus

Um ihre patchweise Tesselierung mit einem Dreieckskontrollnetz zu nutzen, generierten Pulli und Seagal [PS96] mit Hilfe eines Greedy-Algorithmus aus kantenbenachbarten Dreiecks-Faces im Dreieckskontrollnetz eine Liste von Pseudo-Vierecke. Der Algorithmus verwendet eine Heuristik, die Ergebnisse nahe am Optimum [PS96] liefert.

Ein Dreieck des Dreieckskontrollnetzes heißt frei, wenn es vom Paarfindungsalgorithmus noch keinen Partner zugeordnet bekam. Zu Beginn sind alle Dreiecke des Kontrollnetzes frei. Der Algorithmus arbeitet mit vier Listen $S_0,\ldots,S_3$, in denen er die freien Dreiecke verwaltet. Im ersten Schritt erfolgt eine Einordnung der Dreiecke des Kontrollnetzes in die Listen $S_0,\ldots,S_3$:

S_0 : 0 oder 1 freier Kantennachbar.

S_1 : 2 freie Kantennachbarn, einer davon ist in $S_0 \cup S_1$.

S_2 : 2 freie Kantennachbarn, beide sind in S_3.

S_3 : 3 freie Kantennachbarn.

Bei geschlossenen Kontrollnetzen besitzen die Dreiecke drei freie Kantennachbarn, alle Dreiecke sind demzufolge in Liste S_3 zu finden. Offene Kontrollnetze hingegen können Dreiecke am Rand haben mit nur einem oder zwei freien Kantennachbarn. Eine Überprüfung bei der Einordnung ist somit nötig.

Die Priorität in den Listen ist absteigend geordnet: Elemente der Liste S_0 haben die höchste Priorität und Elemente der Liste S_3 besitzen die niedrigste Priorität. Der Algorithmus zur Paarfindung startet mit $S_0,\ldots,S_3$ als Eingabe und erzeugt eine Liste mit Dreiecks-Paaren sowie eine Liste mit Einzel-Dreiecken, die einen Dummy-Partner erhalten.

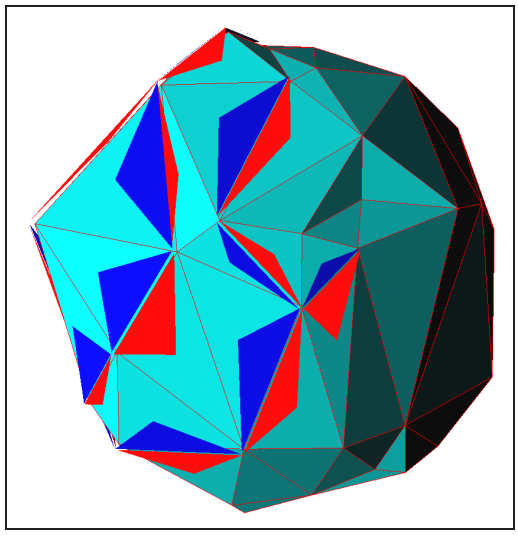


Abbildung 5.28: Paarfindungsalgorithmus beim Zusammenstellen von Pseudo-Vierecken: Die jeweils rot-blau markierten benachbarten Dreiecke ergeben ein Pseudo-Viereck, die unmarkierten Dreiecke befinden sich noch in den Listen $S_0,\ldots,S_3$ zur weiteren Verarbeitung.

Nach Durchlaufen des Paarfindungsalgorithmus sind die meisten Dreiecke zu Pseudo-Vierecken zusammengefasst. Tests in der Praxis haben ergeben, dass nur ca. ein Prozent als Einzel-Dreiecke übrig bleiben und mit einem Dummy-Partner versehen werden müssen. Die Einzel-Dreiecke mit Dummy-Partner werden mit einem Flag versehen und der Pseudo-Viereckslisteliste hinzugefügt.

Pairing of Triangles $(S_0,\ldots,S_3)$
while $S_0,\ldots,S_3$ are not empty **do**
Take a triangle f from the nonempty set with the highest priority
// Look for a free partner g for f:
if f has free neighbors **then**
if f has a neighbor in S_0 **then**
take it as g
else
let g be the neighbor with the highest priority
end if;
remove f,g from $S_0,\ldots,S_3$
insert (f,g) into the pair list
else
remove f from $S_0,\ldots,S_3$
mark f as unpaired *singleton*
end if;
// Increase neighbor priorities:
move neighbors of f (and g) from S_3 to S_2
move neighbors of f (and g) from S_2 and S_1 to S_0
if any triangle moved from S_3 to S_2 **then**
move all triangles from S_2 with neighbors in S_0, S_1 or S_2 to S_1
end if;
end while;

5.1.11.3 Einbettung in den Viereckskontrollnetz-Algorithmus

Der Algorithmus für Dreieckskontrollnetze arbeitet primär mit der Pseudo-Viereckskliste, welche im Vorverarbeitungsschritt erzeugt wurde (siehe vorigen Abschnitt 5.1.11.2). Zur interaktiven Darstellung ordnet das Verfahren Bild für Bild jedem Pseudo-Viereck eine geeignete Unterteilungstiefe zu. Die Bestimmung der Tiefenwerte verläuft analog zum Viereckskontrollnetzverfahren (siehe Abschnitt 5.1.2). Zwei gepaarte Dreiecke erhalten somit die gleiche Unterteilungstiefe (siehe Abbildung 5.31).

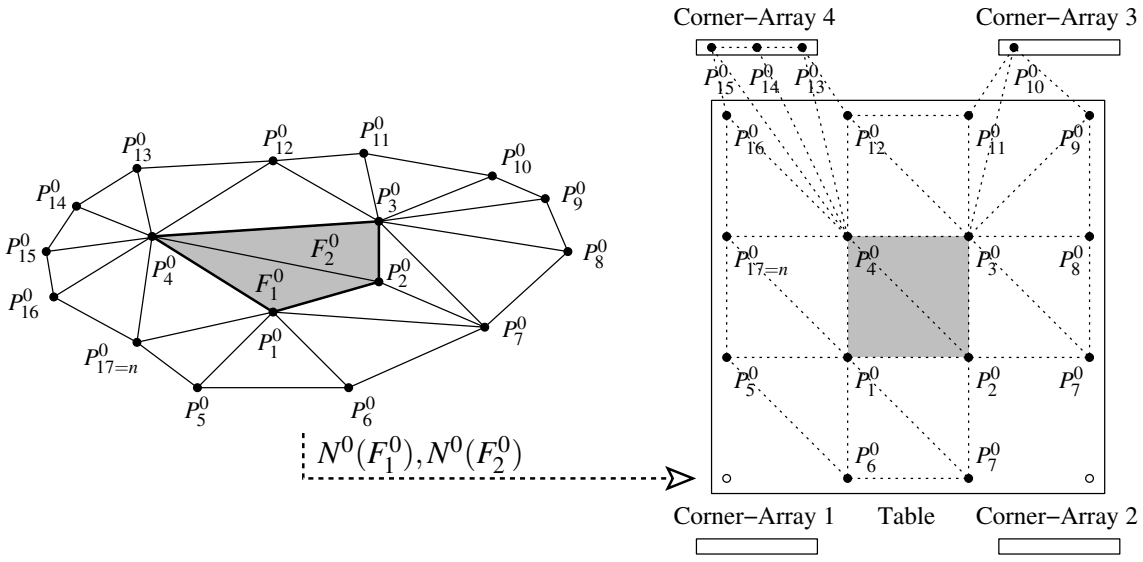


Abbildung 5.29: Aufsammeln der 1-Nachbarschaft eines Pseudo-Vierecks.

Die Tesselierung wird patchweise durchgeführt, wobei ebenso wie beim Viereckverfahren vorgegangen wird: Die beiden Dreiecke des Pseudo-Vierecks mit ihren 1-Nachbarschaften werden aus dem Kontrollnetz aufgesammelt und in ein Slate übertragen (siehe Abbildung 5.29). Die Unterteilung bis zur angegebenen Tiefe des Pseudo-Vierecks erfolgt gleichermaßen wie beim Viereckalgorithmus mit zwei Slates (siehe Abbildung 5.30).

Zwischen den unterschiedlich tesselierten Patches können Lücken auftreten, daher ist eine Lückenschließung 5.1.5 angelehnt an die Viereckmethode notwendig (siehe Abbildung 5.31). Die Ausgaberoutine unterscheidet zwischen echten Pseudo-Vierecken und Einzel-Dreiecken mit Dummy-Partner. Bei Einzel-Dreiecken werden die Dummy-Partner nicht berücksichtigt, während bei den echten Pseudo-Vierecken beide Dreiecke ausgegeben werden. Statt den beim Viereckverfahren verwendeten Quad-Strips kommen beim Dreieckverfahren Triangle-Strips zum Einsatz. Die restliche Ausgaberoutine verläuft analog zur Viereckroutine (siehe Abbildung 5.32).

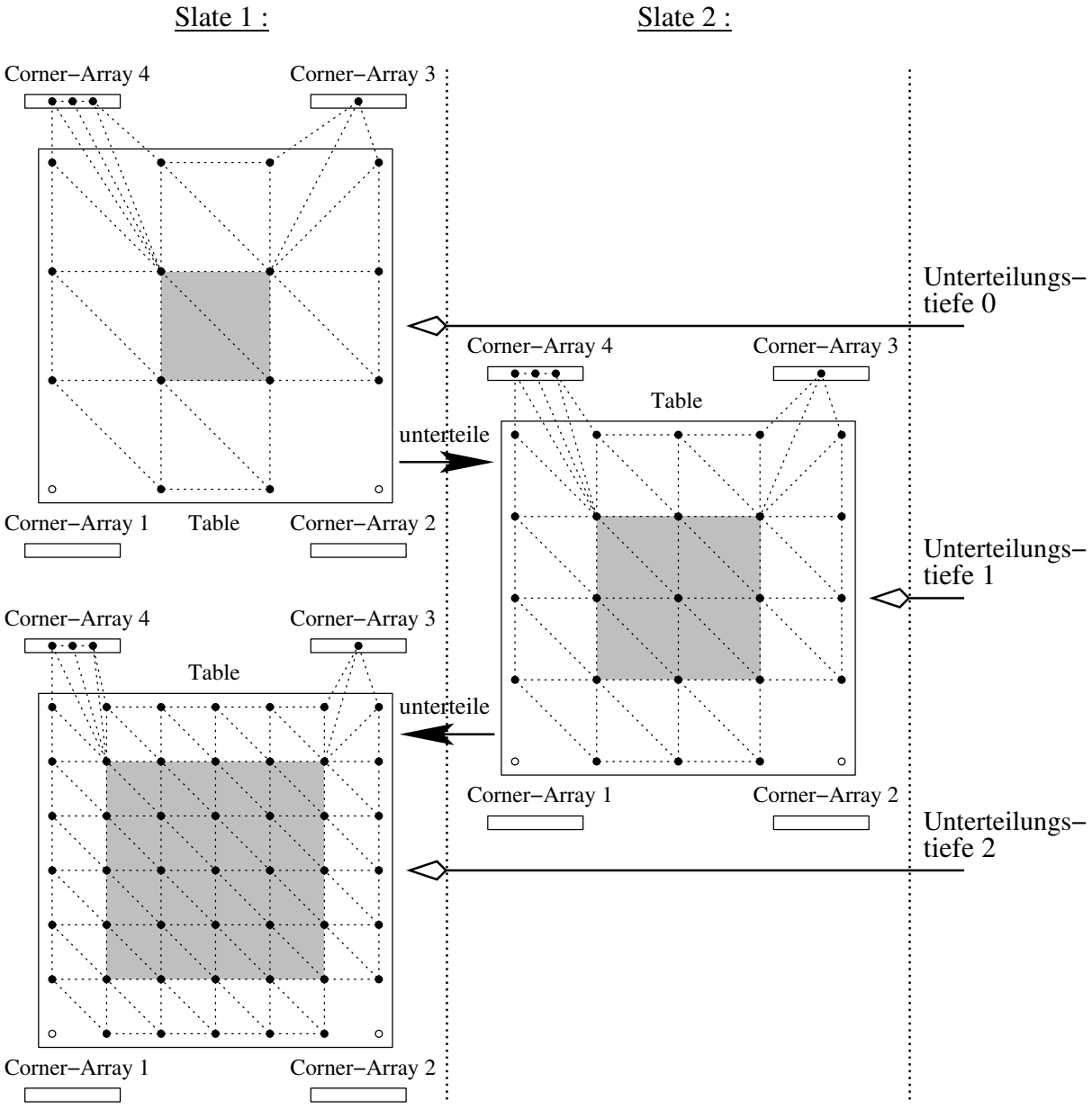


Abbildung 5.30: Tesselierung eines Pseudo-Vierecks mit Hilfe von zwei Slates.

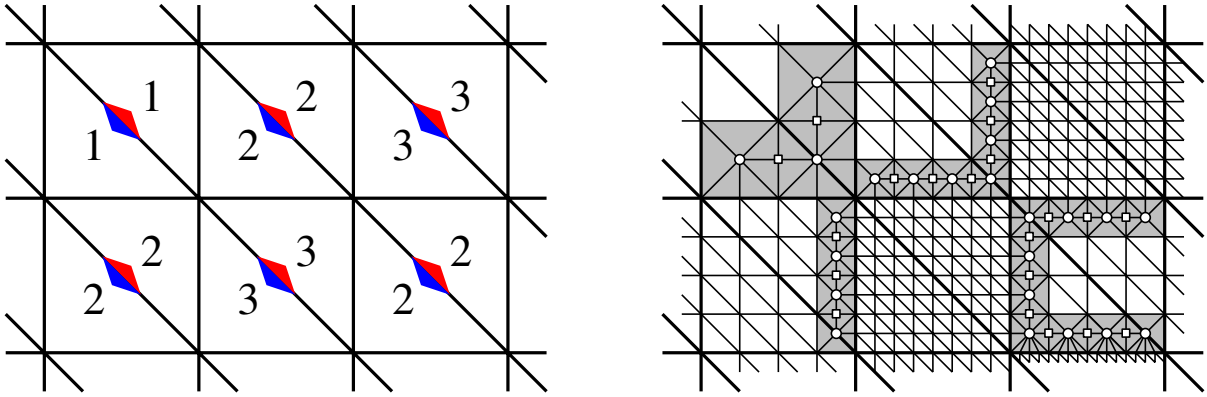


Abbildung 5.31: Schematische Darstellung von sechs Pseudo-Vierecken mit unterschiedlichen Unterteilungstiefen (links), so dass eine Lückenschließung erforderlich ist (rechts).

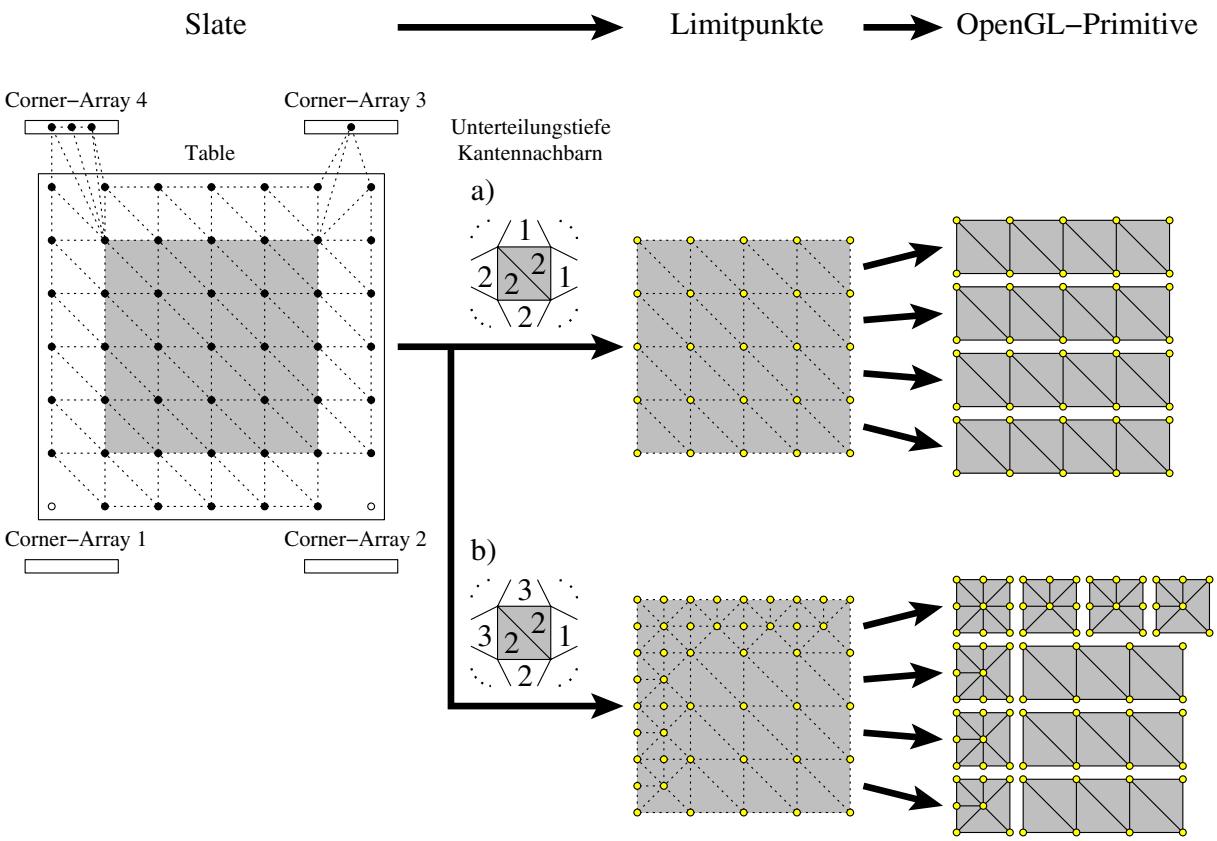


Abbildung 5.32: Ausgabe mit OpenGL-Primitiven eines tesselierten Pseudo-Vierecks.

Ein Caching der bereits berechneten Limitwerte ist ebenfalls analog zum Verfahren aus Abschnitt 5.1.6 möglich.

Die visuellen Ergebnisse zeigen die Abbildungen 5.25, 5.33 und 5.34 am Beispiel einiger Screenshots einer interaktiven Visualisierung mit Loop-Unterteilungsflächen in einer OpenSG-Applikation.

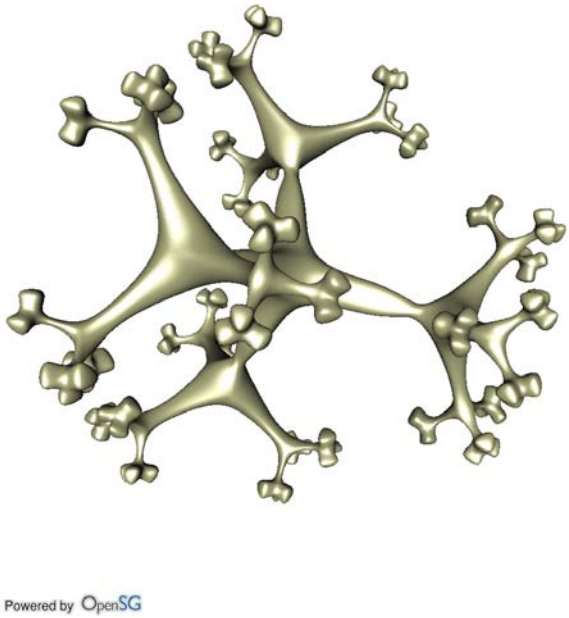


Abbildung 5.33: Beispiel für Loop-Unterteilungsflächen mit OpenSG. Modell von Volker Settgast.

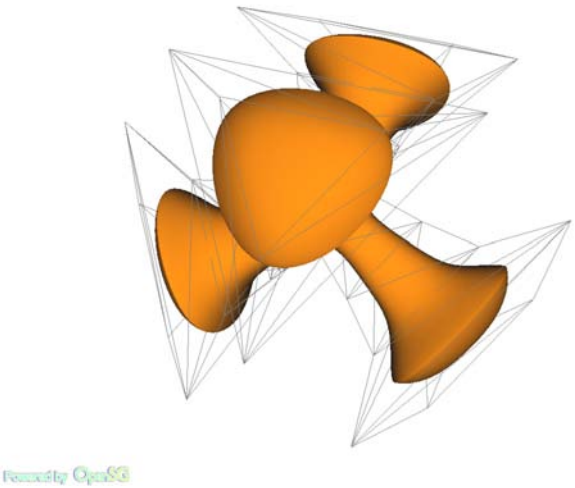


Abbildung 5.34: Das Dreieckskontrollnetz der Loop-Unterteilungsfläche ist grau markiert und enthält scharfe Kanten. Modell von Volker Settgast.

5.2 Photorealistische Darstellung

Die Vorteile von Unterteilungsflächen in der Freiformflächenmodellierung führten zu ihrem Einzug in Modellierungstools wie z.B. Maya und Cinema4D. Damit ist der Bedarf nach effizienten Rendering-Algorithmen vorhanden. Während das Thema interaktive Visualisierung bereits bearbeitet wurde (siehe Abschnitt 5.1), rückte der Bereich photorealistische Darstellung von Unterteilungsflächen bislang weniger in den Fokus des allgemeinen Interesses. Als Standardlösung wird meist das betreffende Unterteilungsflächenobjekt tesseliert und als Polygonmodell gerendert. Ein Vorteil dieses Verfahrens ist die universelle Einsetzbarkeit: Für neue Freiformflächentypen müssen keine eigenen Renderingmethoden entwickelt und implementiert werden. Die feste Tesselierung hat jedoch auch Nachteile: Ein Unterteilungsflächenobjekt muss für jeden Einsatz in einer Szene angepasst tesseliert werden, um eine zu grobe oder zu feine Tesselierung zu vermeiden. Eine zu grobe Tesselierung liefert eine schlechte Darstellungsqualität, während eine zu feine Unterteilung zu einer unakzeptablen Laufzeit führen kann. Eine effektivere Lösung bietet ein adaptiver Ansatz: Während des Renderingprozesses werden nur die Teile des Unterteilungsflächenobjektes berechnet und verfeinert, die an der Beleuchtungsrechnung teilnehmen und zu einer Verbesserung der Bildqualität beitragen. Abschnitt 5.2.1 widmet sich der Darstellung von Unterteilungsflächen mit Radiosity, im Abschnitt 5.2.2 wird ein neues Verfahren zum Raytracen von Unterteilungsflächen vorgestellt. Die Methoden zur Radiosityberechnung sowie der Algorithmus zum Raytracen von Unterteilungsflächen sind im Modular Rendering Tool (MRT) [Fel96] integriert. Das Softwarepaket MRT wurde am Institut für Computergrafik der TU Braunschweig entwickelt und dient als Plattform für innovative Renderingverfahren. Die Algorithmen zur photorealistischen Darstellung von Unterteilungsflächen der folgenden Abschnitte benötigen eine Datenstruktur für die Kontrollnetze der Unterteilungsflächen, die eine partielle Verfeinerung der Kontrollnetze unterstützt. Eine solche Datenstruktur für Kontrollnetze, welche auf einer hierarchischen, erweiterbaren Klassenstruktur basiert, wurde von Torsten Techmann für Dreiecksnetze [TF04] und von Rafael Janetzek [Jan05] für Viereckснetze implementiert. Sie stehen im MRT für Unterteilungsflächen sowie für polygonale Flächen zur Verfügung.

5.2.1 Radiosity

Beim (hierarchischen) Radiosityverfahren [Sch00] wird in einer dreidimensionalen Szene ein Energieaustausch simuliert. Dazu werden alle Objekte in einer gegebenen Qualitätsstufe tesseliert und in einem polygonalen Netz (boundary representation kurz BRep) abgelegt [BFS96]. Wenn die berechneten Formfaktoren unter Berücksichtigung der hierarchischen Struktur über einem gegebenen Schwellenwert liegen, wird rekursiv unterteilt solange die betrachteten Patches eine ausreichende Größe besitzen. Bei gekrümmten Objekten müssen zusätzlich die unterteilten Faces des BReps an das Objekt, das sie repräsentieren, angepasst werden [Sch97]. Diese Anpassung erfolgt innerhalb der Methode *adjustVertex* im MRT. Die Radiosityberechnung des MRT liefert eine adaptiv verfeinerte Szene bestehend aus BReps, die beispielsweise interaktiv visualisiert werden kann.

Um ein neues Objekt respektive Unterteilungsfläche in das Radiosityverfahren des MRT zu integrieren, muss das neue Objekt durch ein BRep repräsentierbar sein, welches lokal verfeinert werden kann. Die Radiosityberechnung startet mit einer groben Approximation der Fläche und führt in einem iterativen Prozess eine adaptive Verfeinerung in einem 2-Schrittverfahren durch. Dazu werden im ersten Schritt die notwendigen Unterteilungen auf dem BRep vorgenommen, die sich aus der Radiosityberechnung ergeben. Im zweiten Schritt werden diese neu erzeugten Vertices an die Limitfläche der Unterteilungsfläche angepasst.

Da der BRep lediglich die adaptiv tesselierte Limitfläche enthält, können für neue Vertices keine Limitpunkte mit Hilfe des BReps berechnet werden. Zur Ausführung der Methode adjustVertex ist daher ein weiteres Kontrollnetz nötig, das die Kontrollpunkte in den angeforderten Tiefen zur Berechnung der Limitpunkte bereit stellt. Im MRT wird dazu ein hierarchische Unterteilungsflächennetz verwendet.

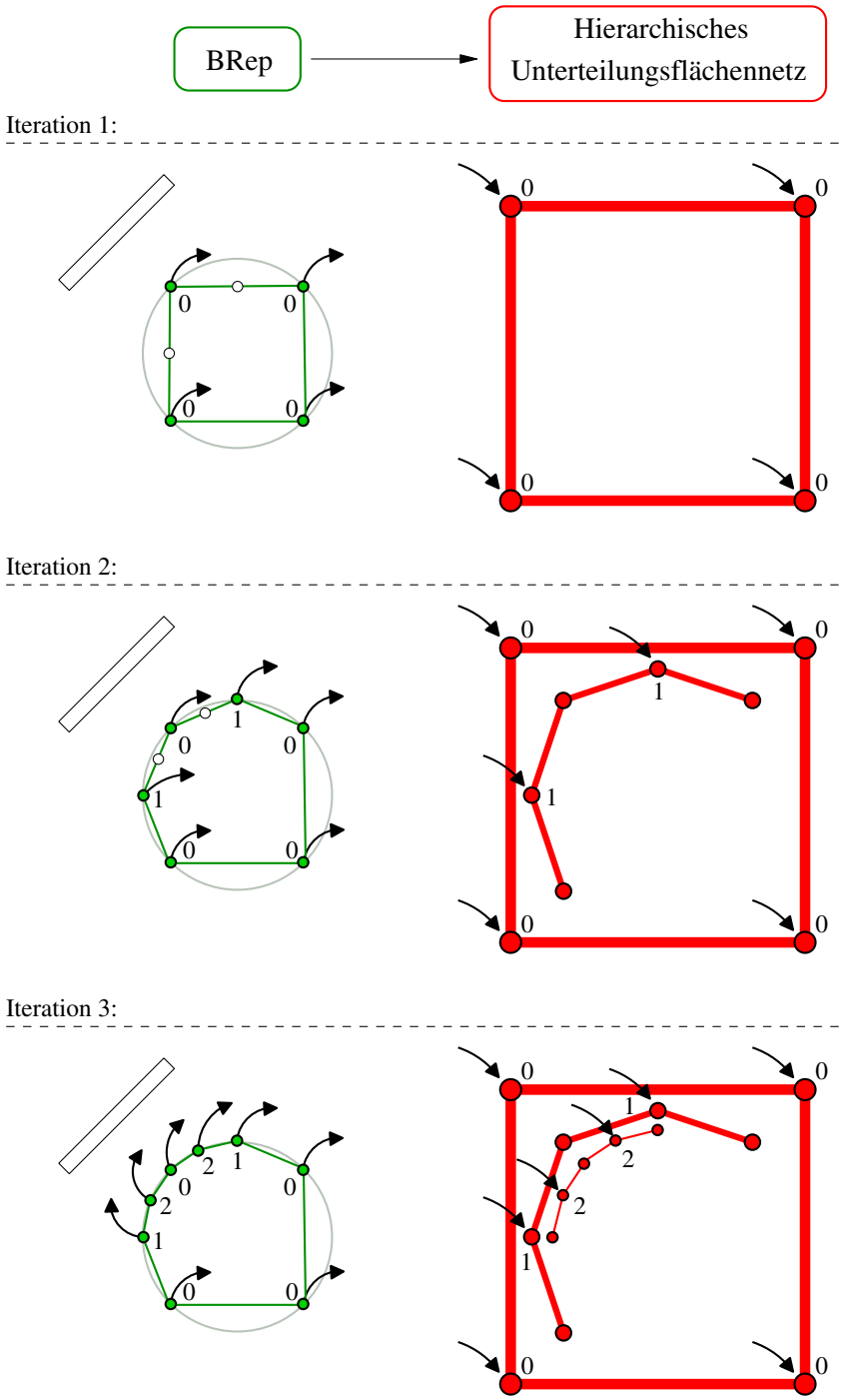


Abbildung 5.35: Adaptive Verfeinerung einer Unterteilungsfläche während der Radiosityberechnung in schematischer Darstellung: Nach Generierung der Approximation der Fläche in Unterteilungstiefe 0 im BRep und Setzen der Pointer in den Vertices des BRep zu den korrespondierenden Vertices im hierarchischen Unterteilungsnetz startet der rekursive Prozess. Neue Vertices werden sukzessive im BRep eingefügt (weiß markierte Punkte). Zur besseren Flächenapproximation werden ihre Limitpunkte benötigt. Dazu werden von den umgebenden alten Vertices im BRep die Pointer zum hierarchischen Unterteilungsflächennetz genutzt um zur korrespondierenden Stelle im hierarchischen Unterteilungsflächennetz zu gelangen. Die entsprechende 1-Nachbarschaft im Unterteilungsflächennetz wird angelegt und der Limitpunkt berechnet. Der neue Vertex im BRep erhält somit seinen Limitpunkt und einen Pointer auf seinen Partner im hierarchischen Unterteilungsflächennetz.

Jeder Vertex des BReps besitzt einen Pointer auf seinen Partnervertex im hierarchischen Unterteilungsflächennetz. Ein neu generierter Vertex im BRep findet über die Pointer seiner Nachbarn im BRep seine korrespondierende Stelle im hierarchischen Unterteilungsflächennetz und generiert dort seine entsprechende 1-Nachbarschaft. Daraus kann sein Limitpunkt berechnet werden. Abschließend erhält der neue Vertex im BRep einen Pointer auf seinen Partner im Unterteilungsflächennetz (siehe Abbildung 5.35). Unterteilungsflächen unterstützen durch ihren Regelsatz und ihr Unterteilungsschema bereits nativ die geforderte adaptive Verfeinerung des hierarchischen Radiosityverfahrens und sind somit auf einfache Weise in das Softwarepaket MRT zu integrieren [Sch97]. Für das Radiosityverfahren des MRT sind die Unterteilungsflächen Loop, Catmull-Clark und ESubs implementiert. Die Abbildungen 5.36 bis 5.39 zeigen Beispiele für photorealistische Darstellungen von Unterteilungsflächen mit Radiosity im MRT.

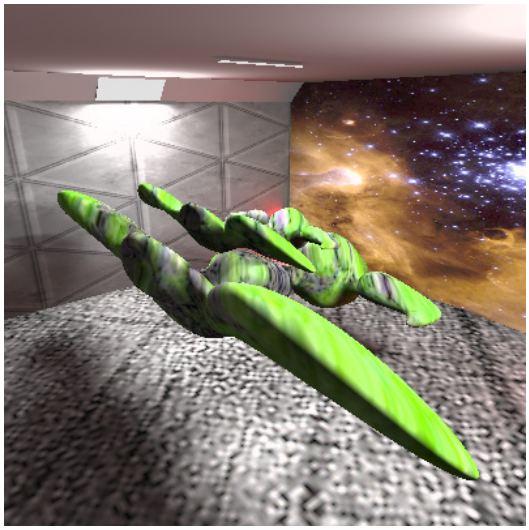


Abbildung 5.36: Radiosityszene mit Catmull-Clark-Unterteilungsflächen [Bod05].

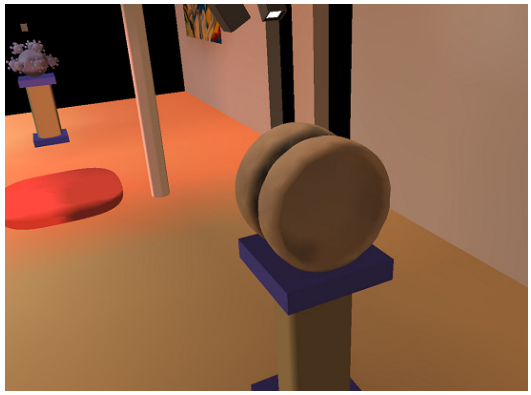


Abbildung 5.37: Radiosityszene mit Loop-Unterteilungsflächen. Museumsszene von Stephan Schäfer.

Abbildung 5.38: Radiosityszene mit uniformen ESubs ohne Special Features [Jan05].

Abbildung 5.39: ESubs mit nicht-uniformen und nicht-konformen Knotenintervallen sowie Special Features [Jan05].

5.2.2 Raytracing

Für die im Vergleich zu den Unterteilungsflächen älteren NURBS-Flächen existieren diverse Raytracingverfahren. Woodward schlägt beispielsweise in [Woo89] eine rekursive Unterteilungsmethode vor, die hauptsächlich in einer Ebene orthogonal zum Strahl arbeitet. Ein numerischer Ansatz ist z.B. in [Tot85] aufgeführt. Weitere Verfahren zum Raytracen von parametrischen Flächen finden sich in [NSK90], [LG90]. In [Sta99], [Sta98] und [BMZ04] werden Parametrisierungen für Unterteilungsflächen vorgestellt. Es erscheint jedoch wenig sinnvoll, diese Parametrisierungen zum Raytracing zu nutzen, da dieses Vorgehen im Allgemeinen zu aufwändig ist (siehe zum Vergleich auch [CSS97]).
Zum Raytracen von Unterteilungsflächen existieren bislang zwei Verfahrenstypen:

- Standardmethode: Die Unterteilungsfläche wird vor dem Raytracen tesseliert und das resultierende Polygonmodell wird gerendert.
- Keine Vortesselierung: Der Raytracer arbeitet direkt mit der Unterteilungsfläche.

Bei der Standardmethode verfeinert ein Tesselator die Unterteilungsfläche uniform oder adaptiv nach einem gegebenen Krümmungsmaß. Das resultierende Polygonnetz kann dann mit einem Raytracer gerendert werden, der lediglich in der Lage sein muss, Polygonmodelle zu verarbeiten. Auf diese Weise ist es möglich, eine Vielzahl von Freiformflächentypen zu visualisieren. Spezielle Schnittpunktalgorithmen für einen neuen Flächentyp müssen nicht entwickelt und implementiert werden. Für ein qualitativ hochwertiges Bild ist eine ausreichend feine Tesselierung erforderlich, wobei insbesondere die Bereiche der Objekt- und Schattensilhouetten eine höhere Unterteilungstiefe benötigen. Da die Tesselierung des Objektes bei der Standardmethode unabhängig vom Raytracingvorgang und somit blickpunktunabhängig durchgeführt wird, muss das komplette Modell mit der Unterteilungstiefe der Silhouettenbereiche in der angestrebten Qualitätsstufe verfeinert werden. Dadurch werden jedoch auch viele Bereiche des Objektes unnötig berechnet, beispielsweise ganze Teile des Objektes die an der Beleuchtungsrechnung nicht teilnehmen. Durch das exponentielle Wachstum der Polygonanzahl beim Unterteilen erhält man mit der Standardmethode schnell eine große Polygonanzahl. Daher ist für jede Szene eine geeignete Unterteilungstiefe für die jeweilige Unterteilungsfläche zu wählen, um eine zu grobe Tesselierung und damit eine schlechte Bildqualität sowie eine zu feine Tesselierung mit zu hoher Laufzeit bei der Bildberechnung zu vermeiden.

Die Verfahren ohne Vortesselierung führen die adaptive Verfeinerung während dem Raytracing-Prozess durch. Für jeden Strahl wird eine geeignete Unterteilungstiefe im Schnittpunktbereich des Objektes bestimmt. Damit werden nur die Objektteile berechnet, die auch tatsächlich an der Beleuchtungsrechnung teilnehmen. Es werden somit nur die Teile des Objektes in szenenspezifischer Feinheit dargestellt, die für ein qualitativ hochwertiges Bild notwendig sind. Kobbelt et.al. präsentierten in [KDS98] ein Verfahren zum Raytracen von Unterteilungsflächen ohne Vortesselierung. Ihr Verfahren nutzt eine *Bounding-Envelope*-Technik [Kob98], bei der die Unterteilungsfläche zwischen einem „größeren“ und einem „kleineren“ Kontrollnetz eingeschachtelt ist. Der Algorithmus verfeinert sukzessive entlang des Strahls beim Eintreten in die Bounding Envelope's bis zu einer geeigneten Tiefe. Die Methode ist schnell und verbraucht im Vergleich zur uniformen Vortesselierung in Silhouettentiefe relativ wenig Speicherplatz. Um die zwei einschachtelnden Kontrollnetze zu berechnen, müssen in einem Vorberechnungschritt für jede vorkommende Valenz Minima und Maxima der Basisfunktionen jedes Vertex in der 1-Nachbarschaft eines Patches bestimmt werden. Erweiterungen des Regelsatzes in Form von beispielsweise scharfen Kanten ziehen umfangreiche Vorberechnungen nach sich. Für jeden praktisch auftretenden Fall in jeder Valenz mit jeder scharfen Kantenkombination müssen die Basisfunktionen berechnet werden. Dies führt zu einer großen unpraktikablen Anzahl von Fällen, welches eine Implementierung im Hinblick auf Erweiterungen des Regelsatzes und Special Features erschwert.

Ein weiteres Verfahren ohne Vortesselierung basiert auf der Methode von Woodward [Woo89] für Bézier-Patches. Dieses Verfahren arbeitet hauptsächlich zweidimensional in einer Ebene ortoghonal zum

Strahl und nutzt die Eigenschaft, dass ein Patch sich in der konvexen Hülle seiner 1-Nachbarschaft befindet. Um zu testen ob ein Strahl ein Patch schneidet, projiziert der Algorithmus die 1-Nachbarschaft des Patches mit Hilfe einer Parallelprojektion in eine Ebene orthogonal zum Strahl. Der Aufpunkt der Ebene ist der Strahlursprung. Liegt der Strahlursprung innerhalb der konvexen Hülle der projizierten 1-Nachbarschaft, so wird das Patch unterteilt und mit den Subpatches rekursiv fortgefahren bis entweder ein Schnittpunkt in gewünschter Tiefe gefunden wurde oder ein Schnitt auszuschließen ist. Im Vergleich zum originalen Woodward-Verfahren mit Bézier-Patches sind die konvexen Hüllen der Unterteilungsflächen-Patches relativ groß im Verhältnis zum eingeschachtelten Objekt. Daher sind viele Schnittpunkttests erforderlich, welche wiederum die Laufzeit erhöhen. Dieses Verfahren ist jedoch einfach zu implementieren und unterstützt alle Special Features sowie Regelsatzerweiterungen bei Beibehaltung der Konvexen-Hülle-Eigenschaft.

Ziel des neuen, im Rahmen dieser Arbeit entwickelten und implementierten Verfahrens zum Raytracen von Unterteilungsflächen ist es, ein qualitativ hochwertiges Bild unter effizienter Nutzung der zur Verfügung stehenden Rechenzeit zu generieren. Die Unterteilungsfläche wird dazu während des Raytracingprozesses automatisch adaptiv verfeinert. Dabei berechnet der Algorithmus nur die Objektteile, die potentiell an der Beleuchtungsrechnung teilnehmen. Die gewählte Unterteilungstiefe für jeden Strahl hängt von den folgenden Kriterien ab:

i) Lage des Schnittpunktbereiches (Silhouette/Innen),

ii) Größe des dedizierten Patches projiziert auf die Bildebene,

iii) Krümmung des Patches.

Das neue Verfahren kann neben dem herkömmlichen Regelsatz auch Special Features darstellen. Weitere Unterteilungsflächentypen können ebenfalls mit diesem Verfahren visualisiert werden. Für Loop, Catmull-Clark und ESubs ist der ShaoLin-Algorithmus implementiert und in den MRT integriert worden.

5.2.2.1 Grundidee

In der Vorbereitungsphase findet eine Einordnung jedes Patches mit seinem Bounding-Volume (BV) in ein frei zu wählendes Raumunterteilungsschema statt. Wenn der Strahl das BV des Patches schneidet, dann startet der eigentliche Schnittpunkttest zwischen Patch und Strahl. Der Algorithmus arbeitet konzeptionell in einer Ebene, die durch die Eckpunkte des Patches gebildet wird. Die Parallelprojektion in Richtung der Normale dieser Ebene wird im Folgenden mit *schattenwerfen* bezeichnet. Wenn der Strahl einen Schatten auf das Patch wirft und der Strahl *nahe* am Patch ist, dann ist ein Schnittpunkt möglich. In diesem Fall wird das Patch rekursiv unter Berücksichtigung der Unterteilungstiefenkriterien unterteilt bis ein Schnittpunkt gefunden wurde oder es sichergestellt ist, dass ein Schnittpunkt nicht existiert. Der Name des Verfahrens **SHAOLIN** kommt von seinem Hauptmerkmal, dem Schattenwurf des Strahles: **Sha**dow **of** the **Lin**e.

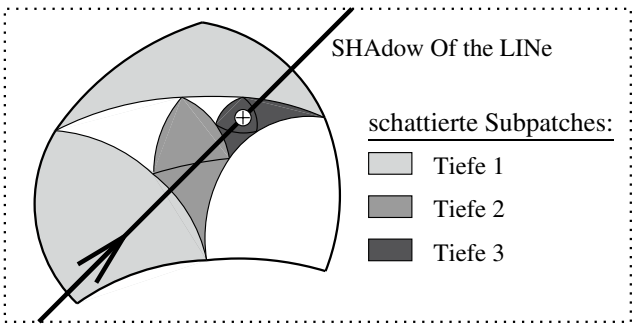


Abbildung 5.40: Grundidee des ShaoLin-Algorithmus.

Das Verfahren wird im Folgenden für Loop-Unterteilungsflächen vorgestellt. Nach einigen Vorbemerkungen zu notwendigen Begriffsdefinitionen im Abschnitt 5.2.2.2 wird in Abschnitt 5.2.2.4 der ShaoLin-Algorithmus erläutert. Abschnitt 5.2.2.5 untersucht den Algorithmus näher. Möglichkeiten zur Optimierung präsentiert Abschnitt 5.2.2.6. Eine Übertragung des Verfahrens auf Vierecks-Unterteilungsflächen zeigt Abschnitt 5.2.2.7. Die Ergebnisse in Form von Laufzeitmessungen sowie die visuellen Resultate sind Thema des Abschnittes 5.2.2.8.

5.2.2.2 Vorbemerkungen

Lückenschließung:

Es seien A^l und B^l zwei kantenbenachbarte Faces des Kontrollnetz M^l in Unterteilungstiefe l (siehe Abbildung 5.41). Eine Tesselierung von B^l und A^l in verschiedenen Tiefen liefert im Allgemeinen eine Lücke zwischen den unterschiedlich tesselierten Patches. Um eine geschlossene Fläche zu erhalten, wird beim Raytracen ein Lückenpolygon eingefügt.

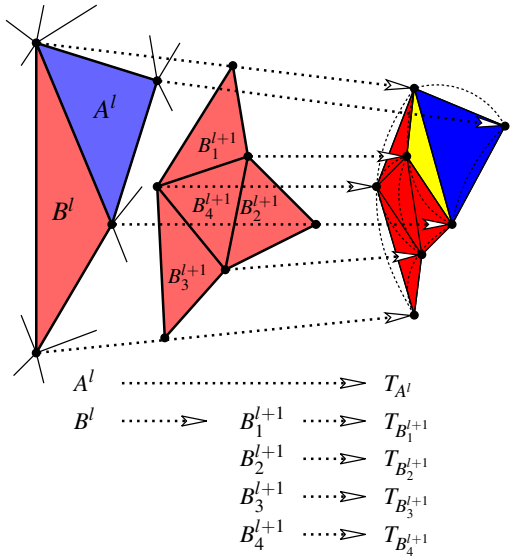


Abbildung 5.41: Zwei kantenbenachbarte Faces A^l und B^l aus Kontrollnetz M^l. Ein Unterteilen von Face B^l liefert die vier Subfaces $B_1^{l+1}, B_2^{l+1}, B_3^{l+1}, B_4^{l+1}$ aus M^{l+1}. Die Patches $L_{B_1^{l+1}}$, $L_{B_2^{l+1}}$, $L_{B_3^{l+1}}$, $L_{B_4^{l+1}}$ und L_{A^l} (gekennzeichnet durch gestrichelte Kurven) sind in Tiefe $l+1$ und l tesseliert (siehe die rot markierten Sehnendreiecke $T_{B_1^{l+1}}$, $T_{B_2^{l+1}}$, $T_{B_3^{l+1}}$, $T_{B_4^{l+1}}$ und das blau gefärbte Sehnendreieck T_{A^l}).
Eine Lücke (gelb markiertes Dreieck) tritt zwischen den unterschiedlich tesselierten Patches auf. Daher ist es erforderlich, ein Polygon in die Lücke einzufügen, um eine geschlossene Oberfläche zu erhalten.

Konvexe-Hülle-Eigenschaft:

Bemerkung 1 (konvexe Hülle). *Unter der Voraussetzung, dass die Unterteilungsregeln aus Konvexkombinationen der jeweiligen 1-Nachbarschaften der neuen Face-, Kanten- und Vertexpunkten bestehen, gilt: Die konvexe Hülle der 1-Nachbarschaft eines Faces F enthält ihr Patch L_F.*

Beweis:
Der Beweis folgt direkt aus den Voraussetzungen.

■

Aus Bemerkung 1 folgt unmittelbar: Wenn eine Ebene Q eine konvexe Hülle, gebildet aus der 1-Nachbarschaft eines Faces F, nicht schneidet, dann schneidet sie auch nicht das Patch L_F. Diese Eigenschaft ist essentiell für die korrekte Arbeitsweise des ShaoLin-Algorithmus.

Es sei F ein Face aus dem Kontrollnetz mit den Eckpunkten P_0, P_1, P_2. Weiterhin sei $T_F = \triangle L_{P_0} L_{P_1} L_{P_2}$ und Q sei eine Ebene orthogonal zu T_F. Wenn Ebene Q das Sehnendreieck T_F schneidet, dann schneidet Q auch mindestens eines der Sehnendreiecke der Subfaces von F. Dies ist der Fall, da L_{P_0}, L_{P_1}, L_{P_2} sich auch bei weiteren Unterteilungsschritten nicht ändern.

Patchrandkurve:

In einem regulären Dreieckskontrollnetz einer Loop-Unterteilungsfläche ist ein Patch ein Bézier-Dreieck vom Grad 4 (siehe beispielsweise [Sta99]). Die Randkurve eines regulären Patches (siehe Abbildung 5.42, links) ist demzufolge eine Bézier-Kurve vom Grad 4

$$b(t) = \sum_{i=0}^{4} B_i \cdot \mathbf{B}_i^4(t)$$

mit den Bézier-Kontrollpunkten:

$$B_0 = \frac{1}{12}\left(P_5^l + P_8^l + P_2^l + P_1^l + P_3^l + P_7^l + 6P_4^l\right) \tag{5.4}$$

$$B_1 = \frac{1}{12}\left(\frac{1}{2}P_5^l + \frac{1}{2}P_1^l + \frac{3}{2}P_8^l + \frac{3}{2}P_3^l + 6P_4^l + 2P_7^l\right) \tag{5.5}$$

$$B_2 = \frac{1}{12}\left(2P_8^l + 2P_3^l + 4P_4^l + 4P_7^l\right) \tag{5.6}$$

$$B_3 = \frac{1}{12}\left(\frac{1}{2}P_6^l + \frac{1}{2}P_{10}^l + \frac{3}{2}P_8^l + \frac{3}{2}P_3^l + 6P_7^l + 2P_4^l\right) \tag{5.7}$$

$$B_4 = \frac{1}{12}\left(P_8^l + P_3^l + P_4^l + P_9^l + P_{10}^l + P_6^l + 6P_7^l\right) \tag{5.8}$$

Aus den Eigenschaften der Bézier-Dreiecke bzw. der Bézier-Kurven folgt unmittelbar, dass B_0 der Limitpunkt von Kontrollpunkt P_4^l ist. Ebenso ist B_4 der Limitpunkt von P_7^l (siehe Abbildung 5.42, links). B_1 liegt auf der Geraden $B_0 + \alpha b'(0)$ mit $\alpha > 0$, die durch die Tangente der Randkurve an der Stelle $t = 0$ bestimmt wird. Gleichermaßen befindet sich B_3 auf der durch die Tangente der Randkurve an der Stelle $t = 1$ definierten Geraden $B_4 + \beta b'(1)$ mit $\beta < 0$. Weiterhin gilt: Die Randkurve $b(t)$ verläuft innerhalb der konvexen Hülle der Bézier-Kontrollpunkte $B_0, \ldots, B_4$.

Es sei E eine Ebene und $\mathscr{P}_E$ bezeichnet die Parallelprojektion in die Ebene E in Richtung des Normalenvektors von E. Dann gilt wegen der affinen Invarianz:

$$\sum_{i=0}^{4} \mathscr{P}_E(B_i) \cdot \mathbf{B}_i^4(t) = \mathscr{P}_E\left(\sum_{i=0}^{4} B_i \cdot \mathbf{B}_i^4(t)\right)$$

Für die korrekte Arbeitsweise des ShaoLin-Algorithmus ist diese Eigenschaft entscheidend.

Haben die beiden Vertices der betrachteten Kante eine Valenz ungleich sechs, so liegt für die zugehörige Randkurve ein irregulärer Fall vor. Im irregulären Fall ist es im Allgemeinen nicht möglich, die Randkurve durch eine Bézier-Kurve darzustellen. Stam stellt in [Sta99] eine Parametrisierung für Loop-Unterteilungsflächen in irregulären Fällen vor. Mit dieser Parametrisierung kann zwar die Randkurve beschrieben werden, eine konvexe Hülle für die Randkurve ist jedoch nicht gegeben. Da die Konvexe-Hülle-Eigenschaft für den ShaoLin-Algorithmus unabdingbar ist, muss eine andere Lösung gewählt werden. Das ShaoLin-Verfahren approximiert daher die Randkurve mit Hilfe einer Bézier-Kurve. Diese liefert über ihre Kontrollpunkte die erforderliche konvexe Hülle. Um die Randkurve möglichst eng zu approximieren, müssen folgende Eigenschaften erfüllt sein (siehe auch Abbildung 5.42, rechts):

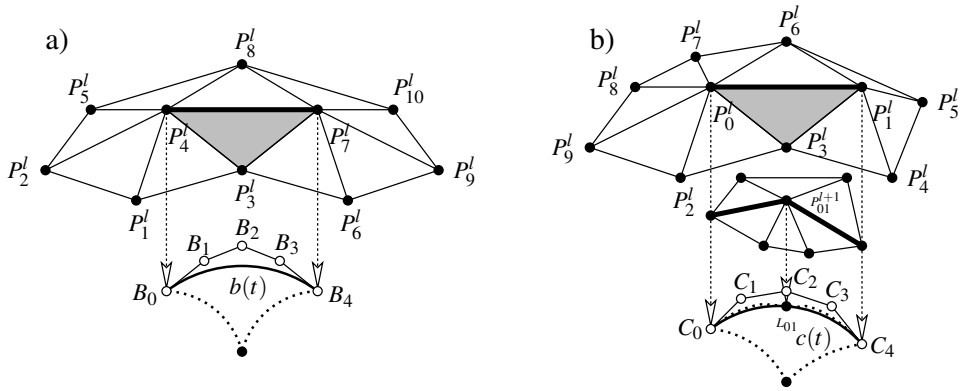


Abbildung 5.42: Fall *a)*: Randkurve eines Patches mit regulärem Dreieck-Kontrollnetz: Die fett markierte Kante des Kontrollnetzes gehört zu der eingezeichneten Randkurve $b(t)$.
Fall *b)*: Randkurve eines Patches mit irregulärem Dreieck-Kontrollnetz: Die Bézier-Kurve $c(t)$ approximiert die Randkurve (punktierte Kurve), welche zur fett markierten Kante im Kontrollnetz gehört. Mit dieser Kante wurde ein Unterteilungsschritt durchgeführt, um den neuen Kantenpunkt P_{01}^{l+1} zu erhalten, L_{01} ist sein zugehöriger Limitpunkt. Aus der 1-Nachbarschaft der fett markierten Kante in M^l und L_{01} können die Bézier-Kontrollpunkte der approximierenden Kurve $c(t)$ bestimmt werden.

P1: $c(0) = L_0$, $c(1) = L_1$,

P2: Die Tangente der Randkurve in L_0 besitzt die gleiche Richtung wie $c'(0)$, ebenso hat die Tangente der Randkurve in L_1 die selbe Richtung wie $c'(1)$.

P3: $c(0.5)$ ist gleich dem Limitpunkt L_{01} des Kontrollpunktes P_{01}^{l+1}.

Unter Berücksichtigung der gewünschten Eigenschaften lassen sich die die Bézier-Kontrollpunkte $C_0,\ldots,C_4$ der gesuchten Kurve $c(t)$ bestimmen:

$$C_0 = L_0 \tag{5.9}$$
$$C_1 = L_0 + \frac{1}{12} tangent_0 \tag{5.10}$$
$$C_2 = \frac{8}{3}\left(L_{01} - \frac{1}{16}C_0 - \frac{1}{4}C_1 - \frac{1}{4}C_3 - \frac{1}{16}C_4\right) \tag{5.11}$$
$$C_3 = L_1 - \frac{1}{12} tangent_1 \tag{5.12}$$
$$C_4 = L_1 \tag{5.13}$$

wobei $tangent_0$ der Tangentialvektor der Randkurve in L_0 ist und $tangent_1$ den Tangentialvektor in L_1 repräsentiert.

5.2.2.3 ShaoLin-Toolbox

Der ShaoLin-Algorithmus erhält als Eingabe ein Startpatch S. Nach l Unterteilungen des Startpatches liegen 4^l Subpatches S_j vor. Die Randkurven der Subpatches S_j, die Teil der Randkurve des Startpatches S sind, heißen Außenrandkurven (siehe Abbildung 5.43). Subpatches S_j ohne Außenrandkurven werden als Innenpatches bezeichnet. Analog dazu werden Subpatches S_j mit Außenrandkurven Außenpatches genannt. In Unterteilungstiefe l des Startpatches S sind $4^l - 3\cdot 2^l + 3$ Innenpatches und $3\cdot 2^l - 3$ Außenpatches mit $3\cdot 2^l$ Außenrandkurven vorhanden.

Der ShaoLin-Algorithmus benötigt drei Ebenen für den Schnittpunkttest zwischen einem Strahl R und einem Startpatch S, die im Folgenden eingeführt werden (siehe Abbildung 5.44).

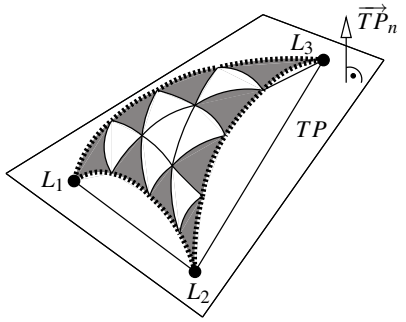


Abbildung 5.43: Zweifaches Unterteilen des Startpatches S liefert als Subpatches 7 Innenpatches (weiß markiert) und 9 Außenpatches (grau gekennzeichnet). Die Außenrandkurven sind schraffiert dargestellt. Die Ebene TP ist durch die Ecklimitpunkte L_1, L_2, L_3 des Startpatches S definiert.

Definition 9 (Ebene TP). *Es sei S ein Startpatch mit den Ecklimitpunkten L_1, L_2, L_3.*
Dann ist die Dreiecksebene TP definiert durch die drei Punkte L_1, L_2, L_3. Der Normalenvektor von TP heißt $\overrightarrow{TP}_n$.

Definition 10 (Ebene RP). *Es sei R ein Strahl mit Aufpunkt R_{org} und Richtung $\vec{R}_{dir}$. Die Dreiecksebene TP sei durch Startpatch S gegeben.*
Die Strahlebene RP ist durch den Aufpunkt R_{org} und die beiden Richtungsvektoren $\vec{R}_{dir}$ und $\overrightarrow{TP}_n$ definiert. Wenn $\vec{R}_{dir}$ und $\overrightarrow{TP}_n$ linear abhängig sind, dann kann der zweite Richtungsvektor beliebig gesetzt werden. Das RP-Koordinatensystem ist festgelegt durch:

$$\overrightarrow{RP}_y = \overrightarrow{TP}_n, \qquad \overrightarrow{RP}_z = \frac{\vec{R}_{dir} \times \overrightarrow{TP}_n}{|\,\vec{R}_{dir} \times \overrightarrow{TP}_n\,|}, \qquad \overrightarrow{RP}_x = \frac{\overrightarrow{TP}_n \times \overrightarrow{RP}_z}{|\,\overrightarrow{TP}_n \times \overrightarrow{RP}_z\,|}$$

Definition 11 (Ebene CP). *Es sei S ein Startpatch, R ein Strahl und RP die zugehörige Ebene.*
Dann heißt eine Ebene mit Aufpunkt R_{org} und den Richtungsvektoren $\vec{R}_{dir}$, $\overrightarrow{RP}_z$ Kreuzungsebene CP.

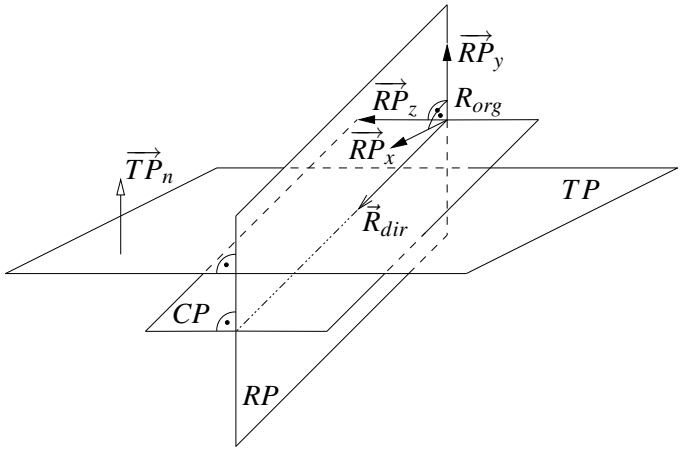


Abbildung 5.44: Ebenen des ShaoLin-Algorithmus: Dreiecksebene TP, Strahlebene RP und die dazu orthogonale Kreuzungsebene CP.

Es sei S ein Startpatch und TP die resultierende Dreiecksebene. Dann bezeichnet $\mathscr{P}_S : I\!R^3 \to TP$ die Parallelprojektion in die Ebene TP in Richtung $\overrightarrow{TP}_n$. Strahl R *wirft einen Schatten auf* S wenn

$$\mathscr{P}_S(S) \cap P_S(R) \neq \emptyset$$

gilt. Ein Schnitt zwischen Startpatch S und Strahl R ist nur dann möglich, wenn S schattiert ist.

Bei der tesselierten Flächendarstellung ergeben sich näher zu untersuchende Modifikationen: Es sei S das betrachtete Startpatch und $T = \triangle L_1L_2L_3$ das Sehnendreieck des Startpatches. H_1, H_2, H_3 seien die drei konvexen Hüllen der drei Randkurven von S. Ein Startpatch, welches die Gleichung

$$\mathscr{P}_S(S) \subseteq \bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T \tag{5.14}$$

erfüllt, heißt *gültiges* Patch. Das in TP projizierte Randkurvengebiet umfasst den Bereich in TP, der außerhalb des Sehnendreiecks T sowie zwischen der projizierten Randkurve und T liegt. Wenn Formel (5.14) gilt, dann ist das projizierte Randkurvengebiet ein Teil von $\bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T$. Ist Formel (5.14) nicht erfüllt, so sind weitere Unterteilungsschritte erforderlich. Der Algorithmus startet dann mit den gültigen Subpatches von S als Startpatches. Die Subpatches werden zu gültigen Patches, da die Folge der Subpatches mit zunehmender Unterteilungstiefe zu einer glatten Oberfläche konvergieren. Die Fläche von S ist glatt, da im Inneren eines Patches keine scharfe Kanten existieren. Abschnitt 5.2.2.5 behandelt den Spezialfall, dass Formel (5.14) für ein Startpatch S nicht erfüllt ist. In Kontrollnetzen aus realistischen Modellierungsprozessen treten im Allgemeinen nur gültige Startpatches auf, ungültige Startpatches kamen bislang nicht vor (siehe auch Abschnitt 5.2.2.5). Daher wird im Folgenden angenommen, dass Formel (5.14) erfüllt ist. Somit ist ein Schnitt zwischen Startpatch S und Strahl R nur dann möglich, wenn T oder $H_{1,2,3}$ schattiert sind, das bedeutet:

$$\left(\bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T\right) \cap \mathscr{P}_S(R) \neq \emptyset$$

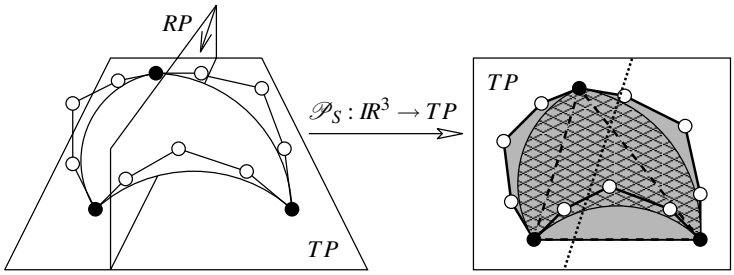


Abbildung 5.45: Parallelprojektion in Dreiecksebene TP entlang $\overrightarrow{TP}_n$: $\mathscr{P}_S(S)$ (schraffiert) ist Teil von $\bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T$ (grau gekennzeichnet).

Es seien S_j, $j = 1,\ldots,4^l$, die Subpatches von Startpatch S in Unterteilungstiefe l und T_j die zugehörigen Sehnendreiecke der Subpatches S_j. H_k, $k = 1,\ldots,3 \cdot 2^l$ seien die konvexen Hüllen der Außenrandkurven der Subpatches. Die Parallelprojektion $\mathscr{P}_S$ bezieht sich für alle Subpatches S_j immer auf das Startpatch S: Projektionsrichtung und Zielebene ist für alle Subpatches von S die Ebene TP mit Richtung $\overrightarrow{TP}_n$, wobei TP durch Startpatch S definiert ist. Ein Schnitt zwischen S respektive den Subpatches S_j und Strahl R ist nur dann möglich, wenn mindestens ein Sehnendreieck T_j oder eine konvexe Hülle H_k schattiert ist, das heißt wenn gilt:

$$\left(\left(\bigcup_{j=1,\ldots,4^l} \mathscr{P}_S(T_j)\right) \cup \left(\bigcup_{k=1,\ldots,3\cdot 2^l} \mathscr{P}_S(H_k)\right)\right) \cap \mathscr{P}_S(R) \neq \emptyset$$

Um zu überprüfen, ob ein Sehnendreieck $T = \triangle L_1L_2L_3$ schattiert ist, müssen die Eckpunkte von T getestet werden: Liegen die Eckpunkte von T in beiden Halbräumen von Ebene RP, so ist T schattiert. Die Einordnung eines Punktes in die beiden Halbräume kostet 2 Additionen, 3 Subtraktionen, 3 Multiplikationen und 1 Vergleich. Eine Klassifizierung der vier Sehnendreiecke eines unterteilten Patches erhält man somit durch maximal 12 Additionen, 18 Subtraktionen, 18 Multiplikationen und 6 Vergleiche.

Ein weiterer Vortest im ShaoLin-Algorithmus verwendet die Konvexe-Hülle-Eigenschaft des Patches S (siehe Bemerkung 1). Es sei K_S die konvexe Hülle der 1-Nachbarschaft des Faces von Patch S. K_S enthält S und wird im ShaoLin-Algorithmus als konvexe Hülle für S verwendet. Wenn $CP \cap K_S \neq \emptyset$ gilt,

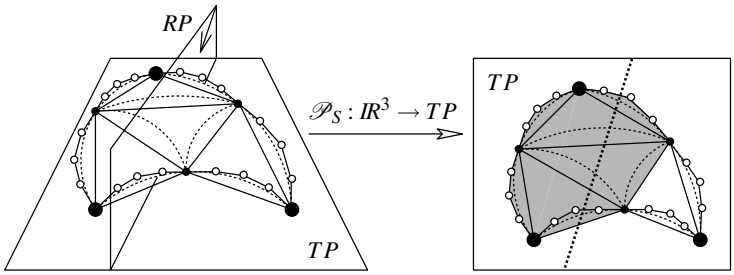


Abbildung 5.46: Nach einem Unterteilungsschritt: Parallelprojektion in Dreiecksebene TP entlang $\overrightarrow{TP_n}$, TP ist durch Startpatch S definiert. Die schattierten Sehnendreiecke der Subpatches von S mit ihren zugehörigen konvexen Hüllen der Außenrandkurven sind grau markiert.

dann *kreuzt* K_S die Ebene CP. Analog zum Schattierungstest wird überprüft, ob sich Kontrollpunkte der konvexen Hülle K_S im rechten *und* linken Halbraum von Ebene CP befinden. Ein Schnitt zwischen Startpatch S und Strahl R ist nur dann möglich, wenn K_S die Ebene CP kreuzt. Nach l Unterteilungsschritten des Startpatches S liegen die Subpatches $S_j, j = 1,\ldots 4^l$ und ihre zugehörigen konvexen Hüllen K_j vor. Wiederum ist ein Schnitt zwischen Startpatch S und Strahl R nur dann möglich, wenn zumindest ein K_j die Ebene CP kreuzt: $CP \cap (\bigcup_{j=1,\ldots,4^l} K_j) \neq \emptyset$.

5.2.2.4 ShaoLin-Algorithmus

Nach der Einführung der notwendigen Definitionen, widmet sich dieser Abschnitt der Arbeitsweise des ShaoLin-Algorithmus (siehe Teilabschnitt 5.2.2.4.1), zeigt dass in allen Fällen eine geschlossene Fläche vorliegt (siehe Teilabschnitt 5.2.2.4.2), erläutert den Schnittpunkttest (siehe Teilabschnitt 5.2.2.4.3) und präsentiert den vorgestellten Algorithmus im Pseudocode (siehe Teilabschnitt 5.2.2.4.4).

5.2.2.4.1 Arbeitweise des ShaoLin-Algorithmus

Im Allgemeinen ist der Schattentest *eines Patches* S mit einem Strahl R sehr kostenintensiv. Aus diesem Grund wird stattdessen das Sehnendreieck T_S von S und die konvexen Hüllen der Außenrandkurven getestet, welches wesentlich schneller überprüft werden kann (siehe Abschnitt 5.2.2.3). Wenn das Sehnendreieck T_S oder die konvexen Hüllen der Außenrandkurven schattiert sind, dann ist ein Schnitt möglich. Ein weiterer Test überprüft, ob die konvexe Hülle des Patches K_S die Ebene CP kreuzt. Der ShaoLin-Algorithmus nutzt diese zwei Vortests, um zu entscheiden ob ein Schnittpunkt vorliegen kann und eine weitere Untersuchung des Patches sinnvoll ist.

Wenn das Startpatch S beide Tests bestanden hat, wird S in seine vier Subpatches $S_1,\ldots,S_4$ unterteilt. Anschließend wird überprüft, welche Sehnendreiecke der Subpatches $S_1,\ldots,S_4$ und welche konvexen

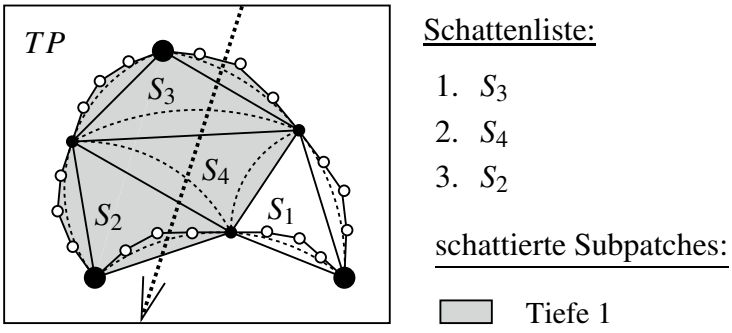


Abbildung 5.47: Schattenliste nach einem Unterteilungsschritt des Startpatches S.

Hüllen ihrer Außenrandkurven schattiert sind. Da eine geschlossene tesselierte Fläche vorliegt, müssen die inneren Randkurven der Subpatches nicht betrachtet werden. Es reicht aus die konvexen Hüllen der Außenrandkurven zu kontrollieren um sicherzustellen, dass die Übergänge zu den benachbarten Patches von S lückenfrei sind.

In einer Schattenliste sind die schattierten Subpatches von S aufgeführt, sortiert nach ihrem Abstand zum Strahlaufpunkt. Der Algorithmus arbeitet die Schattenliste in einem rekursiven Prozess durch bis ein Schnittpunkt gefunden wurde oder das Ende der Schattenliste erreicht ist. Die Parameter für diesen rekursiven Aufruf sind das aktuelle Subpatch, der Strahl und die derzeitige Unterteilungstiefe.

Für das aktuelle Subpatch wird getestet, ob seine konvexe Hülle die Ebene CP kreuzt. Ist dies der Fall, so ist ein Schnitt möglich und es muss entschieden werden, ob eine weitere Unterteilung nötig ist. Um ein qualitativ hochwertiges Bild mit möglichst geringer Laufzeit und Speicherplatzverbrauch zu erhalten, werden drei Verfeinerungskriterien berücksichtigt:

- Silhouettenzugehörigkeit: Wenn das untersuchte Subpatch zur Silhouette gehört, muss es weiter verfeinert werden als innere Bereiche des Objektes. Ein Sehnendreieck eines Patches mit Normale $\vec{N}$ gehört zum Silhouettenbereich, wenn gilt:

$$sil := \mid \vec{N} \cdot \vec{R}_{dir} \mid \leq \varepsilon_s, \qquad 0 \leq \varepsilon_s \leq 1$$

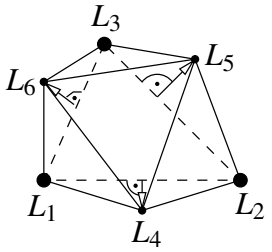
  wobei ε_s die Größe des Silhouettenbereiches spezifiziert.

- Krümmung: Teile des Objektes mit stärkerer Krümmung benötigen eine feinere Unterteilung, um glatt auszusehen. Statt einer exakten aber teuren Krümmungsberechnung wird eine schnelle Approximation verwendet: Das betrachtete Subpatch mit seinem Sehnendreieck $\triangle L_1L_2L_3$ wird zunächst unterteilt, wodurch vier neue Sehnendreiecke mit drei neuen Limitpunkten L_4, L_5, L_6 entstehen. Es sei cv der maximale Abstand zwischen einem neuen Limitpunkte L_i, $i = 4,5,6$, und der zugehörigen Kante des Sehnendreiecks $\triangle L_1L_2L_3$. Es sei

$$cv := \max_{i=1,2,3} \left\{ \left| \alpha_i - \frac{\beta_i \cdot \alpha_i}{\beta_i \cdot \beta_i} \cdot \beta_i \right| \right\},$$

$$\text{mit} \quad \alpha_i = L_j - L_i, \;\; j = i+3$$

$$\beta_i = L_{(i+1) \bmod 3} - L_i$$



  $\varepsilon_c \geq 0$ spezifiziert das Maß für die Krümmung. Wenn $cv \geq \varepsilon_c$ ist, dann ist eine weitere Verfeinerung des Subpatches notwendig.

- Größe des Subpatches: Eine weitere Verfeinerung des aktuellen Subpatches ist nicht erforderlich, wenn eine Unterteilung des Subpatches Sehnendreiecke liefert, die projiziert in die Bildebene IP kleiner als die Pixelgröße werden. Um glatte Reflexionen zu erhalten, darf die maximale Größe der projizierten Sehnendreiecke ein gegebenes Maximum nicht überschreiten.

Um alle Verfeinerunskriterien zu berücksichtigen, werden ein kombiniertes ε, sowie die Parameter *minSize* und *maxSize* verwendet. Der folgende Test kombiniert die Kriterien zur adaptiven Unterteilung:

if *projected size of the patch on* $IP <$ *minSize* **then**
  stop refinement;
else
  if *projected size of the patch on* $IP >$ *maxSize* **then**
    further refinement necessary;
  else

```
        if cv/sil < ε then
            stop refinement;
        else
            further refinement necessary;
        end if;
    end if;
end if;
```

Entscheidet der Algorithmus, dass ein weiterer Unterteilungsschritt notwendig ist, dann wird das aktuelle Patch unterteilt und überprüft, welche der neuen Subpatches schattiert sind. Diese schattierten Subpatches werden, geordnet nach ihrem Abstand zum Strahlaufpunkt, in die Schattenliste eingetragen. An dieser Stelle des Algorithmus erfolgt ein rekursiver Aufruf mit dem nächsten Element aus der Schattenliste.

5.2.2.4.2 W-Fall

Es seien A und B Subpatches des Startpatches S. Im Beispielfall von Abbildung 5.48a nimmt der Algorithmus nur B in die Schattenliste auf, obwohl $\mathscr{P}_S(A) \cap \mathscr{P}_S(R) \neq \emptyset$ vorliegt und damit das *Patch A* schattiert ist. Da das *getestete Sehnendreieck* T_A von A nicht schattiert ist, wird A jedoch nicht der Schattenliste zugeordnet.

Nach der Zuordnung von B in die Schattenliste erfolgt im nächsten Schritt der rekursive Aufruf für das erste Element in der Schattenliste, im betrachteten Beispiel (siehe Abbildung 5.48a) ist dies Patch B. Wiederum wird getestet, ob die konvexe Hülle K_B von Patch B die Ebene CP kreuzt. Wurde der Test erfolgreich absolviert, so unterteilt der Algorithmus B in die Subpatches B_1, B_2, B_3, B_4. Im Allgemeinen ist $\mathscr{P}_S(T_B) \neq \bigcup_{i=1,\ldots,4} \mathscr{P}_S(T_{B_i})$, somit kann der Schnitt zwischen $G := \mathscr{P}_S(T_B) \setminus \bigcup_{i=1,\ldots,4} \mathscr{P}_S(T_{B_i})$ und der Projektion des benachbarten Patches $\mathscr{P}_S(A)$ nicht leer sein. Dies ist der Fall, wenn die Projektion des Strahls $\mathscr{P}_S(R)$ den Bereich $\bigcup_{i=1,\ldots,4} P_S(T(B_i))$ an seinen Außenkanten verlässt und wieder eintritt. Im Beispiel von Abbildung 5.48b verlässt $\mathscr{P}_S(R)$ den Bereich $\bigcup_{i=1,\ldots,4} P_S(T(B_i)$ über die Außenkante von T_{B_2} (dick markiert) und tritt über die Außenkante von T_{B_1} (dick gekennzeichnet) wieder ein. Im Folgenden wird dieses Aus- und Eintreten des Strahles *W-Fall* genannt.

Bei einem W-Fall überquert die Projektion des Strahls $\mathscr{P}_S(R)$ ein unzugeordnetes Gebiet, im Beispiel von Abbildung 5.48b ist das Gebiet zwischen T_{B_2} und T_{B_1} betroffen (schraffiert dargestellt). Dieses unzugeordnete Gebiet ist Teil des projizierten benachbarten Patches A. Um diese Lücke zu schließen, *re-*

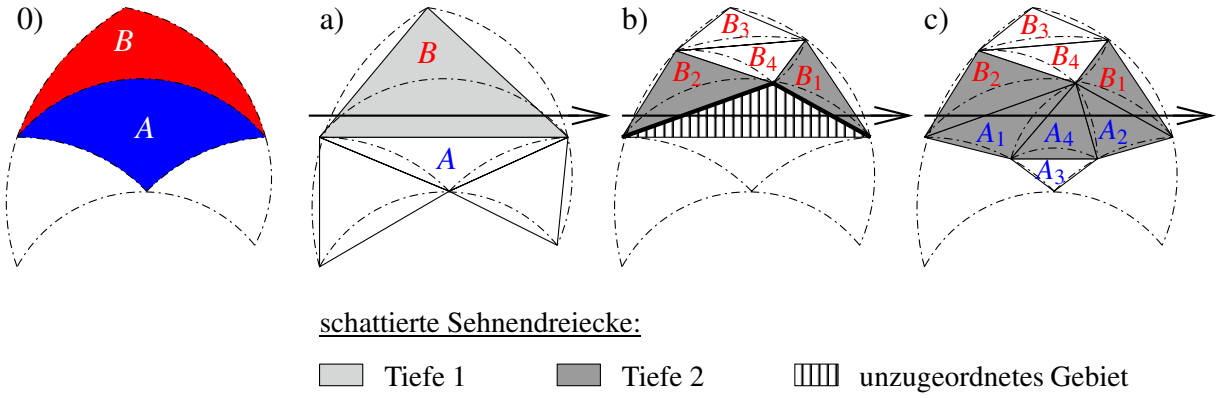


Abbildung 5.48: W-Fall: Ein unzugeordnetes Gebiet tritt in Unterteilungstiefe 2 zwischen T_{B_2} und T_{B_1} auf. Durch eine Reaktivierung des Patches A wird die Lücke geschlossen.

aktiviert der Algorithmus Patch A und ordnet A der Schattenliste zu mit dem Zusatzvermerk *aus W-Fall*. Im behandelten Beispiel enthält die Schattenliste jetzt die Elemente B_1, A (W-Fall), B_2.

Wenn im Verlauf des Algorithmus der rekursive Aufruf für A erreicht wird, führt der Zusatzvermerk W-Fall zu einem Überspringen der Ausschlusstests für Patch A: Auf den Vortest, ob K_A Ebene CP kreuzt, wird verzichtet. Ebenso entfällt eine Auswertung der drei Verfeinerungskriterien Silhouettenzugehörigkeit, Krümmung und Patchgröße. Stattdessen wird A direkt unterteilt. Die resultierenden Subpatches von A werden wieder regulär weiter behandelt. Im Beispiel von Abbildung 5.48 sind A_1, A_4, A_3 schattiert und werden der Schattenliste zugeordnet. Der Algorithmus fährt mit der Untersuchung der Elemente der Schattenliste unter Beachtung möglicher W-Fälle fort. Das ShaoLin-Verfahren überquert somit immer eine geschlossene, tesselierte Fläche. Im behandelten Beispiel werden die Sehnendreiecke von B_1, A_1, A_4, A_3 B_2 abgearbeitet, die auch im W-Fall eine geschlossene Fläche ergeben. Ein potentieller Schnittpunkt kann somit nicht verloren gehen.

5.2.2.4.3 Schnittpunkttest

Wenn die Verfeinerungskriterien entschieden haben, dass für das aktuelle Subpatch keine weiteren Unterteilungen nötig sind, wird ein Schnittpunkttest mit dem aktuellen Subpatch durchgeführt. Dazu testet der Algorithmus, ob ein Schnitt zwischen dem Sehnendreieck des Subpatches und dem Strahl R vorliegt. Ist das betrachtete Subpatch ein Außenpatch bezüglich des Startpatches S, so müssen zusätzlich die Außenrandkurvenbereiche untersucht werden. Es sei VP eine Ebene mit Aufpunkt R_{org} und Normale $\vec{R}_{dir}$. Um die Außenbereiche zu testen, werden die Bézier-Kontrollpunkte der Außenrandkurve in die Ebene VP durch eine Parallelprojektion entlang $\vec{R}_{dir}$ projiziert. Der Strahl R schneidet den Außenrandbereich nur dann, wenn R_{org} innerhalb der konvexen Hülle der projizierten Bézier-Kontrollpunkte in VP liegt.

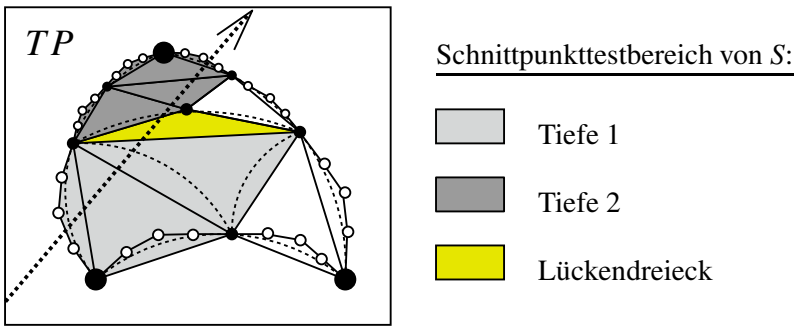


Abbildung 5.49: Auf Schnittpunkt getestete Bereiche von S: Sehnendreiecke, konvexe Hüllen der Außenrandkurven und Lückendreieck.

Hat der Algorithmus einen Schnittpunkt gefunden, so muss die dazugehörige Normale berechnet werden. Dazu werden die baryzentrischen Koordinaten (α, β, γ) des Schnittpunktes bezüglich des untersuchten Sehnendreiecks $\triangle L_1L_2L_3$ bestimmt. Es seien $\vec{N}_1$, $\vec{N}_2$, $\vec{N}_3$ die Normalen der Eckpunkte des Sehnendreiecks. Dann ergibt sich die Normale des Schnittpunktes durch $\alpha \cdot \vec{N}_1 + \beta \cdot \vec{N}_2 + \gamma \cdot \vec{N}_3$.

Wenn der Schnittpunkttest keinen Schnittpunkt mit dem aktuellen Subpatch gefunden hat, fährt der Algorithmus mit der Untersuchung des nächsten Elementes in der Schattenliste fort. Wenn das nächste getestete Subpatch eine andere Unterteilungstiefe als das zuletzt geprüfte Subpatch hat, kann eine Lücke auftreten. Zum Füllen dieser Lücke wird ein Lückenpolygon eingefügt (siehe Abbildung 5.49), welches ebenfalls in den Schnittpunkttest involviert wird. Die Berücksichtigung der Außenrandbereiche des Startpatches und die Überprüfung der eingefügten Lückenpolygone garantieren zusammen mit der gewählten Tesselierung eine geschlossene Fläche entlang des Strahls für den ShaoLin-Algorithmus.

5.2.2.4.4 ShaoLin-Pseudocode

Der ShaoLin-Algorithmus erhält als Parameter ein Startpatch *patch*, einen Strahl *ray* und die aktuelle Unterteilungstiefe *level*, welche beim Aufruf des ShaoLin-Algorithmus auf Null gesetzt ist.

```
bool ShaoLin (patch, ray, level)
  if control points of patch cross CP then
    if further refinement necessary then
      Subdivide patch;
      Compute shadow and shadowlist;
      for each shadow-patch_i in the shadowlist do
        if shadow-patch_i was activated by w-case then
            // shadow-patch_i is not a subpatch of patch!
          nextlevel=level;
        else
          nextlevel=level + 1;
        end if;
        if ShaoLin (shadow-patch_i, ray, nextlevel) then
          return true;
        end if;
      end for;
      return false;
    else
        // intersection possible
      if last tested level ≠ level then
        Insert gap polygon between
        the different subdivision levels
        if ray intersects gap polygon then
          return true;
        end if;
      end if;
      if ray intersects chord triangle then
        return true;
      end if;
      if patch is outer patch &&
        ray intersects its outer border curve area(s) then
        return true;
      end if;
    end if;
  else
      // intersection not possible
    return false;
  end if;
```

5.2.2.5 Untersuchung des Algorithmus

Um die korrekte Arbeitsweise des Algorithmus zu garantieren, müssen die folgenden Eigenschaften erfüllt sein:

a) Die Projektion des Randkurvenbereiches entlang $\overrightarrow{TP_n}$ in die Ebene TP liegt innerhalb des projizierten Bounding-Volumes der Randkurve.

b) $\mathscr{P}_S(S) \subseteq \bigcup_{i=1,2,3} P_S(H_i) \cup T_S$

Zu Eigenschaft a): Bei der Verwendung eines regulären Kontrollnetzes liefert die konvexe Hülle aus den Bézier-Kontrollpunkten der Randkurve das gewünschte Bounding-Volume (siehe Abbildung 5.42). Im irregulären Fall existiert im Allgemeinen keine solche Bézier-Kurve. Daher wird eine approximierende Bézier-Kurve $c(t)$ genutzt, welche die Eigenschaften **P1** bis **P3** (siehe Abschnitt 5.2.2.2) erfüllt. Die Kontrollpunkte von $c(t)$ dienen zur Bestimmung der konvexen Hülle und somit des Bounding-Volumes der Randkurve. Da der Algorithmus lediglich die Projektion des Bounding-Volumes benötigt und keine exakte Parametrisierung der Randkurve, ist die Verwendung der approximierenden Randkurve $c(t)$ mit ihren Kontrollpunkten eine gut funktionierende Heuristik. Andere Bounding-Volumes für die Randkurve sind ebenfalls einsetzbar, beispielsweise kann die Randkurve geeignet abgetastet und vergrößert werden.

Zu Eigenschaft b): Die Anforderung b) ist für gebräuchliche Modelle erfüllt (siehe auch Formel (5.14)), die in einen herkömmlichen Modellierungsprozess erzeugt wurden. Abbildung 5.50 veranschaulicht ein Modell, das die geforderte Eigenschaft b) nicht erfüllt. Zur besseren Übersicht zeigen die Abbildungen 5.50 und 5.51, jeweils auf der linken Seite, einen Querschnitt des Modells. Die extreme Positionierung der Kontrollpunkte des Startpatches führt zu einer Selbstüberlappung der Projektion von S auf die Ebene TP. Somit liegt $\mathscr{P}_S(H_i)$ innerhalb von $\mathscr{P}_S(S)$ und es gilt für dieses Patch $\mathscr{P}_S(S) \supset \bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T_S$.

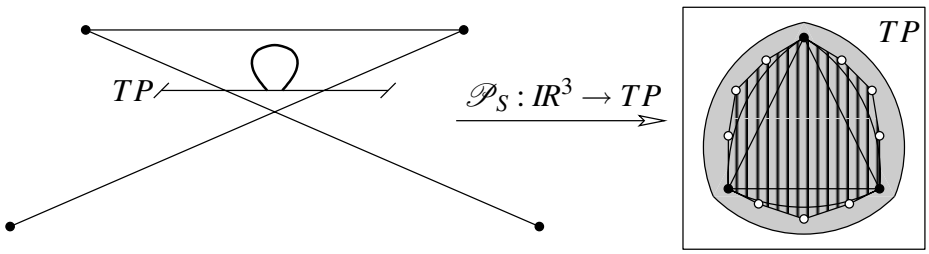


Abbildung 5.50: Spezialfall Eigenschaft b) ist nicht erfüllt, somit gilt $\mathscr{P}_S(S) \not\subseteq \bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T$. Querschnitt durch das Modell (links) und auf Ebene TP projiziertes Modell (rechts) mit $\bigcup_{i=1,2,3} \mathscr{P}_S(H_i) \cup T$ (schraffiert) befindet sich innerhalb $\mathscr{P}_S(S)$ (grau gekennzeichnet).

Da sich innerhalb eines Patches keine scharfen Kanten befinden, führt eine zunehmende Unterteilung des Startpatches S zu lokal flacheren Kontrollnetzen und Subpatches. Es sei S ein Startpatch und $A_j, j = 1,\ldots,4^l$ seine Subpatches in Unterteilungstiefe l. Wegen der Glattheit innerhalb eines Patches existiert eine Unterteilungstiefe l, so dass nach l Unterteilungen von S keine Selbstüberlappungen für die $A_j, j = 1,\ldots,4^l$ bei der Projektion auf ihre zugehörigen Ebenen TP_{A_j} vorliegen (siehe Abbildung 5.51). Somit ist $\mathscr{P}_{A_j}(A_j) \subseteq \bigcup_{i=1,2,3} \mathscr{P}_{A_j}(H_i) \cup T_j$ für jedes Patch A_j erfüllt und der Algorithmus kann die Patches A_j als Startpatches verwenden.

Um zu überprüfen, ob Eigenschaft b) erfüllt ist, werden die Kontrollpunkte des Startpatches S in seine Dreiecksebene TP projiziert. Bildet das projizierte lokale Kontrollnetz von S in TP einen planaren Graph, so ist Eigenschaft b) sicher erfüllt. Die Umkehrung gilt im Allgemeinen nicht, ein nicht-planares Kontrollnetz kann ein Patch generieren, das Anforderung b) genügt. Liegt ein nicht-planares Kontrollnetz

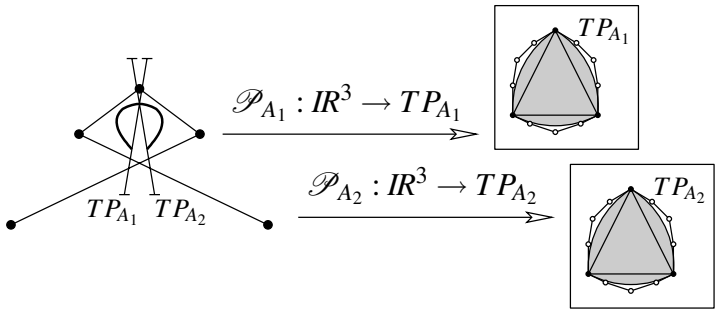


Abbildung 5.51: Spezialfall aus Abbildung 5.50 nach einem Unterteilungsschritt: Eine Unterteilung von Startpatch S liefert die vier Subpatches $A_1,\ldots,A_4$. Im Querschnitt durch das Modell (links) sind die beiden Subpatches A_1 und A_2 mit ihren dazugehörigen Ebenen TP_{A_1} and TP_{A_2} dargestellt. Die Projektion der Subpatches (grau markiert, rechts) in ihre jeweilige Dreiecksebenen erfüllt die Anforderung $\mathscr{P}_{A_j}(A_j) \subseteq \bigcup_{i=1,2,3} \mathscr{P}_{A_j}(H_i) \cup T_{A_j}$, $j = 1,\ldots,4$.

in der Dreiecksebene vor, so wird S unterteilt und die Subpatches von S in ihren jeweiligen Dreiecksebenen getestet. Der Vorgang wird rekursiv wiederholt, bis für die Subpatches ein planares projiziertes Kontrollnetz vorliegt. Mit diesen Subpatches kann der ShaoLin-Algorithmus als Startpatches beginnen. Der rekursive Prozess terminiert, da die Folge der Subpatches gegen eine glatte Fläche konvergiert. Diese Überprüfungsmethode von Eigenschaft b) ist jedoch mehr von theoretischem Interesse, da die Anforderung b) in praktischen Modellen normalerweise immer erfüllt ist.

5.2.2.6 Optimierungen

Um den ShaoLin-Algorithmus zu beschleunigen muss überprüft werden, an welchen Stellen unnötige Tests durchgeführt werden, die nichts zum korrekten Ablauf beitragen und somit vermieden werden können.

Eine Optimierung kann beispielsweise durch die Beschränkung der Schattierungstests erzielt werden: Wenn nur ein Subpatch eines Patches schattiert ist, müssen die Außenrandkurvenbereiche der Subpatches an der gegenüberliegenden Kante des Patches nicht überprüft werden (siehe Abbildung 5.52). Im Silhouettenbereich entfällt das Testen der Außenrandkurvenbereiche komplett, da jeweils die höchst mögliche Unterteilungstiefe des Objektes bei allen Patches in diesem Bereich erreicht wird. Somit liegt im Silhouettenbereich eine geschlossene Fläche ohne Lückenpolygone vor.

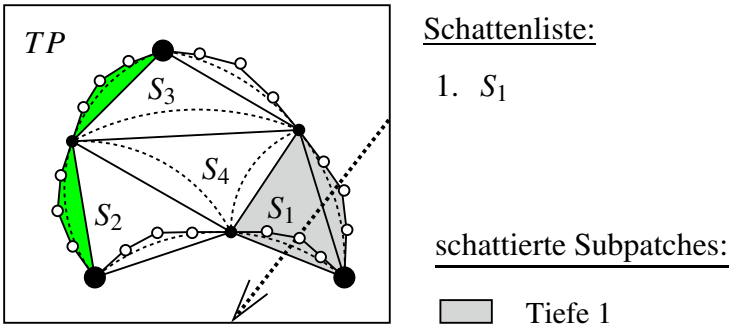


Abbildung 5.52: Optimierung beim Schattentest: Ist nur ein Subpatch schattiert (S_1), so müssen die Außenrandkurvenbereiche auf der gegenüberliegenden Kante (grün markiert) nicht getestet werden.

5.2.2.7 Übertragung auf Viereckskontrollnetze

Der ShaoLin-Algorithmus kann auf andere Unterteilungsflächentypen übertragen werden, wenn der neue Flächentyp die Voraussetzungen aus Abschnitts 5.2.2.5 erfüllt.

Um das ShaoLin-Verfahren zu nutzen, muss zunächst die Ebene TP für den neuen Flächentyp definiert werden. Für einen Vertreter der Viereckskontrollnetz-Flächen, wie z.B. Catmull-Clark- und ESubs-Unterteilungsflächen, sei S das zu untersuchende Startpatch mit den Eckpunkten $L_1,\ldots,L_4$. Die Ebene TP kann beispielsweise durch den Aufpunkt $1/4(L_1+L_2+L_3+L_4)$ und die Normale $\sum_{i=0}^{3}\Delta L_i \times \Delta L_{(i+1) \bmod 4}$ festgelegt werden. Die übrigen benötigten Ebenen und die Bounding-Volumes für die Randkurven können basierend auf Abschnitt 5.2.2.2 bestimmt werden. Der Algorithmus arbeitet mit schattierten Vierecken analog zu der in Abschnitt 5.2.2.4 beschriebenen Technik mit schattierten Dreiecken. Die Vorgehensweise und die rekursive Schnittfindungstechnik sind für Loop- und Catmull-Clark- sowie ESubs-Flächen gleich. Das ShaoLin-Verfahren ist im Rahmen einer Studienarbeit [Bod05] von Christian Bode für Catmull-Clark-Unterteilungsflächen umgesetzt worden. Die Abbildungen 5.53 bis 5.56 zeigen Beispiele für Catmull-Clark-Flächen, welche mit dem ShaoLin-Verfahren für Viereckskontrollnetze visualisiert wurden. Ein weiteres Beispiel für Unterteilungsflächen mit Viereckskontrollnetzen sind die ESubs aus Kapitel 3. Im Rahmen einer Diplomarbeit [Jan05] wurde das ShaoLin-Verfahren für ESubs-Unterteilungsflächen von Rafael Janetzek umgesetzt. Die Abbildungen 5.57 und 5.58 zeigen dazu Beispiele.

Abbildung 5.53: Beispielszene für Catmull-Clark-Unterteilungsflächen. Kopfmodell von Volker Settgast, Schachfiguren von René Berndt.

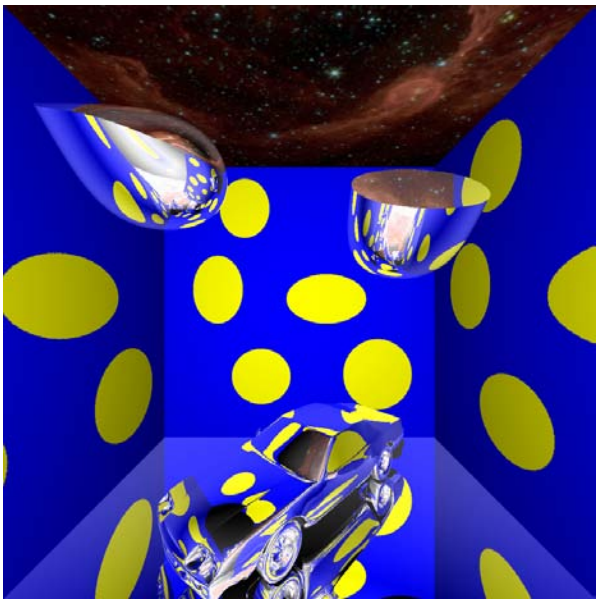


Abbildung 5.54: Beispielszene für Catmull-Clark-Unterteilungsflächen mit scharfen Kanten [Bod05]. Freies Automodell von Viewpoint Animation Engineering.

5.2.2.8 Ergebnisse

Der ShaoLin-Algorithmus berechnet nur die Teile des Unterteilungsflächenobjektes, die an der Beleuchtungsrechnung teilnehmen. Die kameraabgewandten und verdeckten Patches, die von keinem Strahl wie z.B. einem Schattenfühler angerührt werden, sind somit vom Raytracingprozess ausgeschlossen. Kritische Flächenteile, bei denen eine grobe Tesselierung sichtbar wird wie z.B. stark gekrümmte Flächenteile, Silhouettenbereiche und kameranahe Gebiete, werden zur Verbesserung der Bildqualität besonders fein tesseliert. Zu starke Verfeinerungen von Flächenteilen unter Pixelgröße werden verhindert. Ebenso wird auf eine ausreichende Verfeinerung geachtet, die beispielsweise für glatte Reflexionen wichtig ist. Das ShaoLin-Verfahren nutzt auf diese Weise die Rechenleistung und den Speicherplatz effizient zur Generierung eines Bildes in optimaler Qualität.

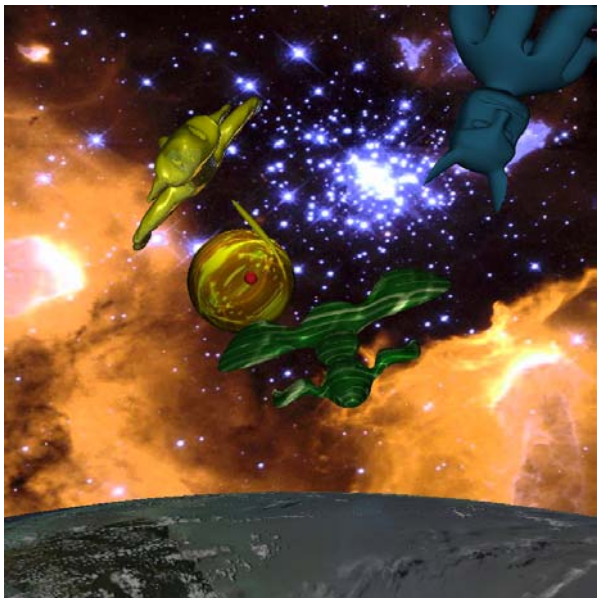


Abbildung 5.55: Beispielszene für Catmull-Clark-Unterteilungsflächen mit 706 Faces im Ausgangskontrollnetz [Bod05].

Abbildung 5.56: Beispielszene für Catmull-Clark-Unterteilungsflächen mit 12336 Faces im Ausgangskontrollnetz [Bod05].

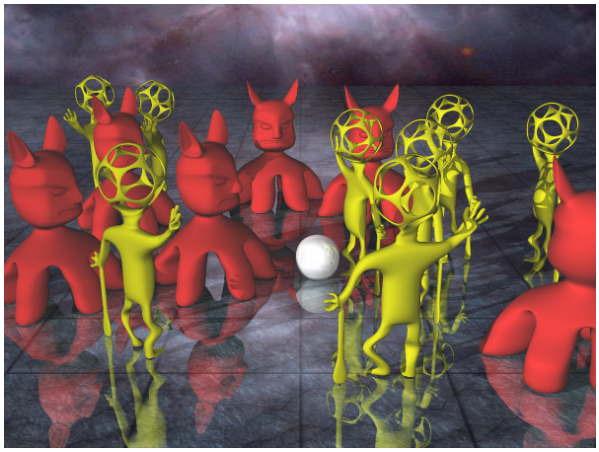


Abbildung 5.57: Beispielszene für ESubs mit uniformen Knotenintervallen [Jan05] und 5140 Faces im Ausgangskontrollnetz.

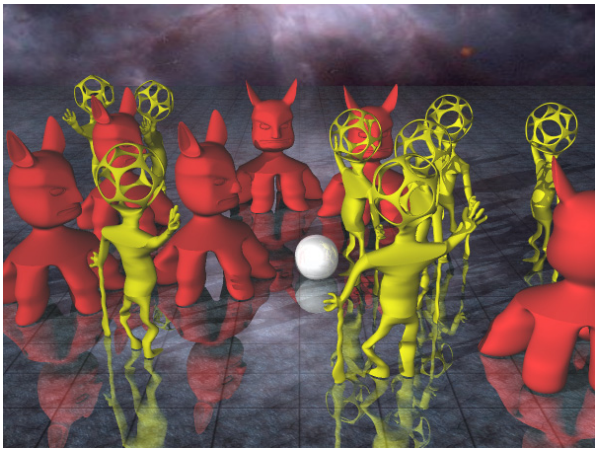


Abbildung 5.58: ESubs mit nicht-uniformen Knotenintervallen, nicht-konformen Faces und Special Features [Jan05].

5.2.2.8.1 Visuelle Ergebnisse

Um die adaptive Arbeitsweise des Algorithmus zu veranschaulichen, wurde eine Szene als Unterteilungstiefenkarte gerendert (siehe Abbildung 5.59). Die Farbe jedes Pixels gibt die Unterteilungstiefe des Schnittpunktes an. Die Schnittpunkte mit der Fläche resultieren aus den Sehstrahlen und den Schattenfühlern. Aus Übersichtsgründen wurde auf eine Einfärbung der Schnittpunkte der reflektierenden und transmittierenden Strahlen verzichtet. Abbildung 5.59 verdeutlicht den höheren Verfeinerungsgrad im Silhouettenbereich und in den stärker gekrümmten Flächenteilen des Objektes. Die übrigen inneren, flachen Objektteile sind in geringerer Unterteilungstiefe dargestellt. Auch bei den Schattenfühlern wird adaptiv verfeinert, die Schattensilhouette in Abbildung 5.59 ist deutlich feiner unterteilt als innen liegende Schattenbereiche.

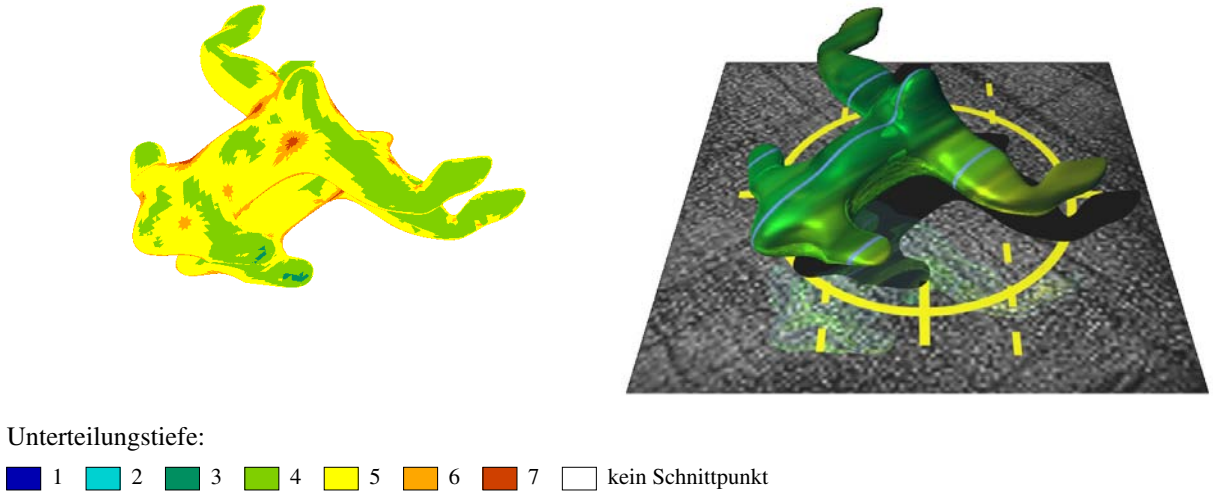


Abbildung 5.59: Unterteilungstiefenkarte (links) und Originalbild (rechts) einer Szene: Adaptiv verfeinerte Unterteilungsfläche durch Sehstrahlen und Schattenfühler. Modell von Matthias Richter.

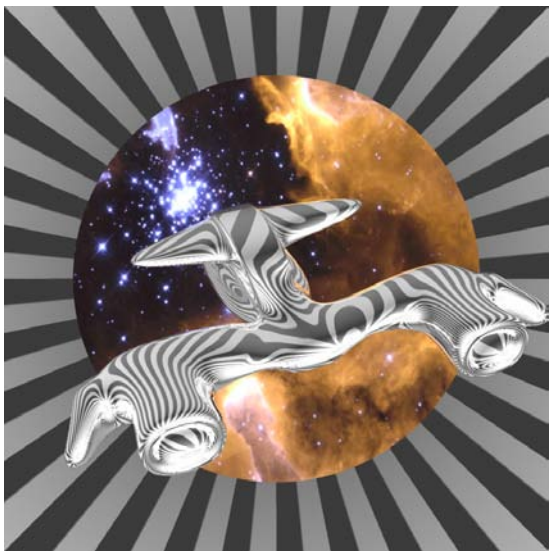


Abbildung 5.60: Vollkommen spiegelnde Unterteilungsfläche. Trotz adaptiver Verfeinerung sind die Reflexionen glatt. Modell von Matthias Richter.

Um die Glattheit der adaptiv tesselierten Fläche zu testen, wurde ein vollkommen spiegelndes Unterteilungsflächenobjekt in einem Streifenzylinder visualisiert (siehe Abbildung 5.60). Die reflektierten Streifen zeigen auf dem Unterteilungsflächenobjekt einen glatten Verlauf an, welches Rückschlüsse auf eine glatte Oberfläche zulässt. Special Features können ebenfalls mit dem ShaoLin-Algorithmus bearbeitet werden, wie z.B. in Abbildung 5.61 zu sehen.

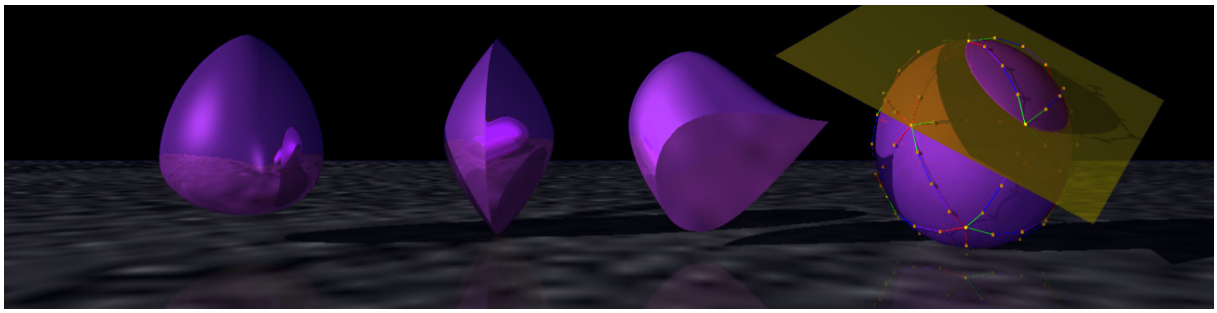


Abbildung 5.61: Beispiel für Unterteilungsflächen mit Special Features. Im Objekt rechts sind die Kontrollpolygone der Randkurven der Startpatches eingezeichnet. Für ein Startpatch ist die Ebene TP dargestellt (gelb markiert). Modell von Torsten Techmann.

5.2.2.8.2 Laufzeit und Speicherplatzverbrauch

Für die Messreihen dieses Abschnitts wurde ein Pentium IV mit Intel 845 Chipset, 1.7 GHz und 1 GB RAM genutzt. Das ausführbare Programm wurde mit dem Microsoft Visual C++ 6.0 Compiler unter Windows 2000 gebildet.

Als Testszenen dienten eine einfache Szene mit 268 Faces im Ausgangskontrollnetz (Frogship, siehe Abbildung 5.59, rechts), eine Szene mit 5756 Faces im Ausgangskontrollnetz (Col-Henge, siehe Abbildung 5.62) und eine größere Szene mit 10328 Faces im Ausgangskontrollnetz (Eggs'n'Bunnies, siehe Abbildung 5.63).

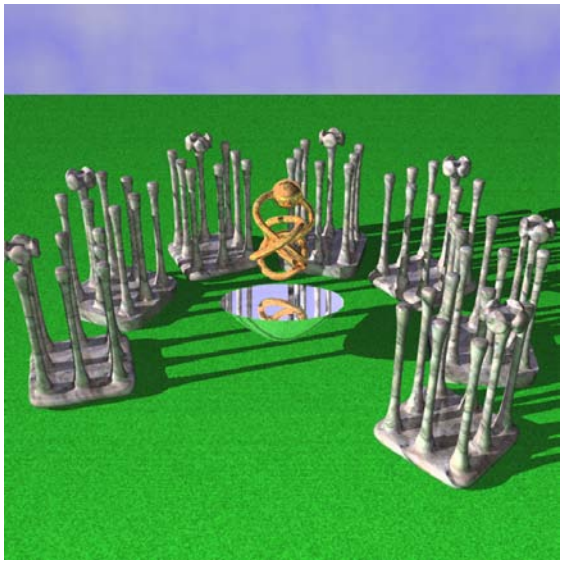


Abbildung 5.62: Testszene Col-Henge mit 5756 Faces im Ausgangskontrollnetz. Modell von Matthias Richter.

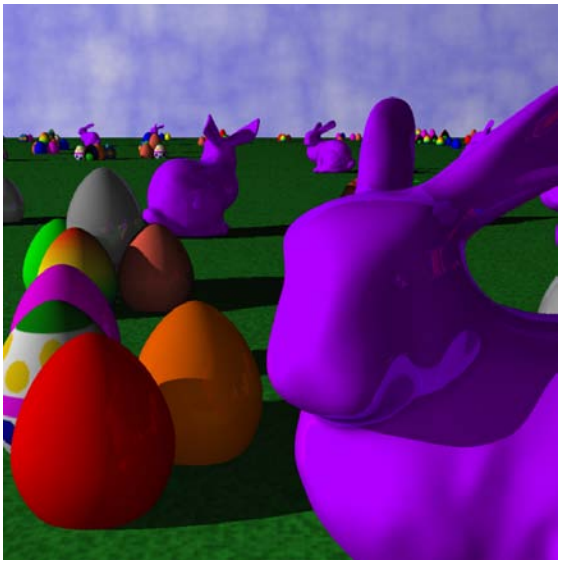


Abbildung 5.63: Testszene Eggs'n'Bunnies mit 10328 Faces im Ausgangskontrollnetz. Modell: Stanford Bunny.

Um das ShaoLin-Verfahren besser einordnen zu können, wurden die Testszenen mit zwei weiteren Methoden geraytraced. Bei dem ersten Algorithmus, einem uniformen Verfahren, wird die Unterteilungsfläche in einer festgelegten Tiefe tesseliert sobald das Bounding-Volume des Objektes von einem Strahl getroffen wird. Die zweite Methode ist ebenfalls ein adaptives Verfahren, welches auf Woodwards Ansatz basiert (siehe Abschnitt 5.2.2) und im Folgenden A3 (another adaptive approach) genannt wird. Der Algorithmus A3 verwendet die gleichen Verfeinerungskriterien wie das ShaoLin-Verfahren, womit eine bessere Vergleichsmöglichkeit gegeben ist. Die Parameter Kombiniertes ε, maxSize und minSize der adaptiven Methoden bieten dem Benutzer eine Vielzahl von Variationsmöglichkeiten in der Bildqualität. Eine Auswahl davon ist in den folgenden Tests verwendet worden. In den Tabellen 5.2 bis 5.4 ist für jede Testszene die Tiefenverteilung der Schnittpunkte bezüglich eines gegebenen ε Wertes aufgelistet. Die Abbildungen 5.64 bis 5.66 präsentieren die benötigte Laufzeit für jede Testszene mit den gegebenen Parametern und den unterschiedlichen Verfahren. Tabelle 5.1 zeigt den Speicherplatzbedarf für die jeweiligen Verfahren.

Bei kleinen Testszenen mit Objekten von geringer Faceanzahl spielen die Vorteile des adaptiven Verfahrens eine untergeordnete Rolle. Mit steigender Komplexität der Szene treten die Vorteile des adaptiven ShaoLin-Algorithmus in den Vordergrund: Durch die adaptive Vorgehensweise kann die Anzahl der berechneten Faces im Gegensatz zum uniformen Verfahren unter Beibehaltung der Bildqualität deutlich reduziert werden. Somit kann beim ShaoLin-Verfahren die Laufzeit verringert und Speicherplatz eingespart werden. Dieser Performancegewinn übersteigt die zusätzlichen Kosten, die beim adaptiven ShaoLin-Verfahren auftreten, bei weitem. Unter zusätzliche Kosten im Vergleich zur uniformen Tesselierung fallen beispielsweise die Auswertung der Verfeinerungskriterien.

Das exponentiellen Wachstum der Faceanzahl beim Unterteilen führt bei der uniformen Tesselierung zu einem hohen Bedarf an Speicherkapazität. Bei Raytracern, welche nicht auf das Rendern von großen Dreiecksmengen spezialisiert sind, führt dies zu Problemen in der Ausführung. Testszene Eggs'n'Bunnies konnte beispielsweise vom verwendeten Raytracer MRT in uniformer Tiefe 4 nicht mehr dargestellt werden.

Der weitere adaptive Ansatz A3 liefert auf Grund der gleichen Verfeinerungskriterien eine ähnliche adaptive Verfeinerung wie der ShaoLin-Algorithmus. Der Speicherplatzverbrauch ist somit adäquat zum ShaoLin-Verfahren. Der A3-Algorithmus ist jedoch deutlich langsamer als die ShaoLin-Methode, welches auf die Methodik des A3-Verfahrens mit seinen größeren Bounding-Volumes in der Ebene VP zurückzuführen ist.

	ShaoLin	uniform Tiefe 3	uniform Tiefe 4	uniform Tiefe 5	uniform Tiefe 6	uniform Tiefe 7
Frogship	83M	24M	48M	140M	495M	1635M
Col-Henge	491M	189M	690M	—	—	—
Eggs'n'Bunnies	1499M	1090M	—	—	—	—

Tabelle 5.1: Speicherplatzbedarf für das ShaoLin-Verfahren ($\varepsilon = 0.01$) und für den uniformen Ansatz.

komb. ε	Durchschnitts-tiefe	Schnittpunkte in Unterteilungstiefe						
		0–1	2	3	4	5	6	7
0.005	4.74	0	0	960	78606	118007	17546	3199
0.010	4.32	0	0	7260	143327	58230	8026	1413
0.020	4.10	0	0	24309	149340	37668	3876	443
0.040	3.98	0	60	38697	140984	31456	1635	170
0.080	3.94	0	668	41568	139241	29147	707	30
0.200	3.92	0	1163	42603	138283	28135	305	11
0.500	3.92	0	1234	42812	138087	27739	218	0

Tabelle 5.2: Anzahl der Schnittpunkte pro Unterteilungstiefe für die Testszene Frogship (minSize=2 Pixels, maxSize=12 Pixels).

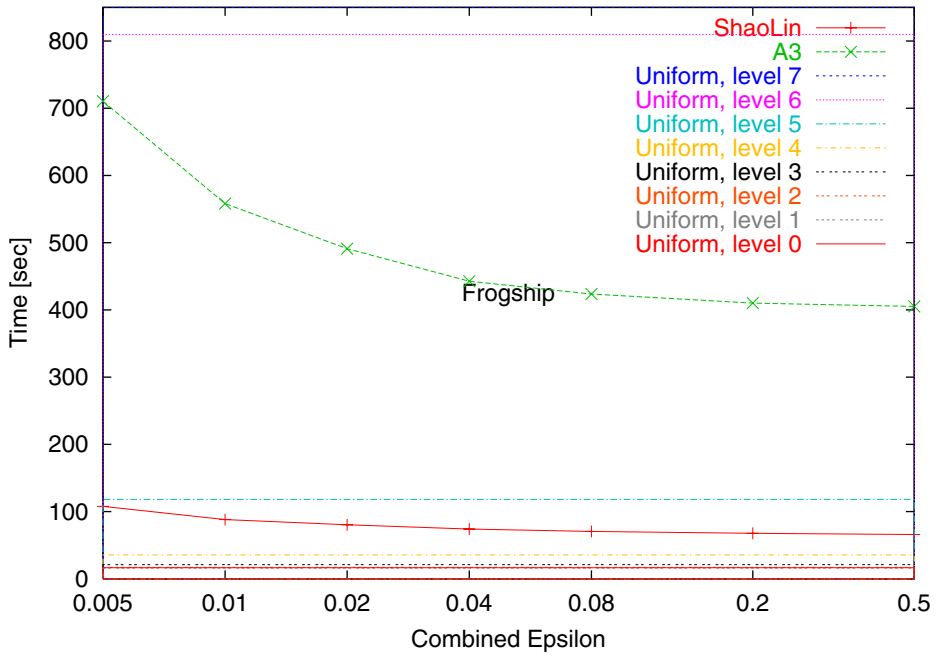


Abbildung 5.64: Laufzeitmessung für Testszene Frogship in Auflösung 800×800.

komb.	Durchschnitts-	Schnittpunkte in Unterteilungstiefe						
ε	tiefe	0–1	2	3	4	5	6	7
0.005	4.26	0	366	43581	406663	125442	32905	3434
0.010	3.89	0	3749	196371	335172	82529	23545	1366
0.020	3.57	0	26663	324712	210037	62414	18109	615
0.040	3.26	0	138635	298292	152784	53122	15835	338
0.080	3.06	0	232187	221692	129657	48624	15207	248
0.200	2.98	0	267544	186845	115271	47336	15055	204
0.500	2.96	0	274966	176705	112317	47033	15039	195

Tabelle 5.3: Anzahl der Schnittpunkte pro Unterteilungstiefe für die Testszene Col-Henge (minSize=2 Pixels, maxSize=12 Pixels).

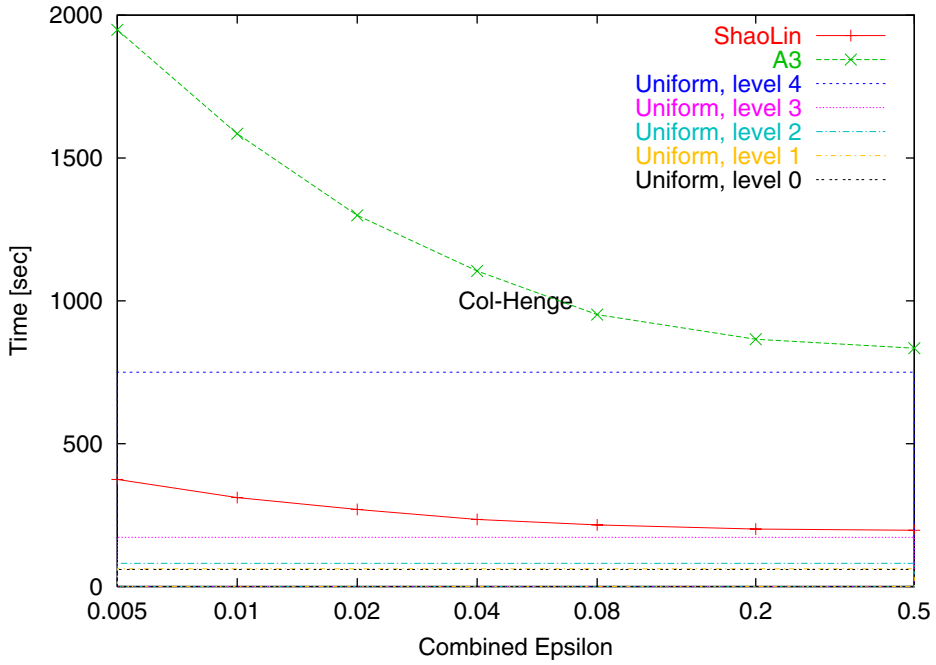


Abbildung 5.65: Laufzeitmessung für Testszene Col-Henge in Auflösung 800×800.

komb. ε	Durchschnitts-tiefe	Schnittpunkte in Unterteilungstiefe 0–1	2	3	4	5	6	7
0.005	4.48	0	5671	119254	262671	274859	103913	6074
0.010	4.32	0	27545	117212	299910	245126	84493	2089
0.020	4.24	0	43791	104466	322688	223068	76369	511
0.040	4.19	0	48299	121455	308671	216631	73181	196
0.080	4.16	0	49585	132244	296500	214256	71473	43
0.200	4.13	0	69865	117587	293285	212855	70949	9
0.500	4.12	0	74910	112192	292307	212612	70600	4

Tabelle 5.4: Anzahl der Schnittpunkte pro Unterteilungstiefe für die Testszene Eggs'n'Bunnies (minSize=2 Pixels, maxSize=20 Pixels).

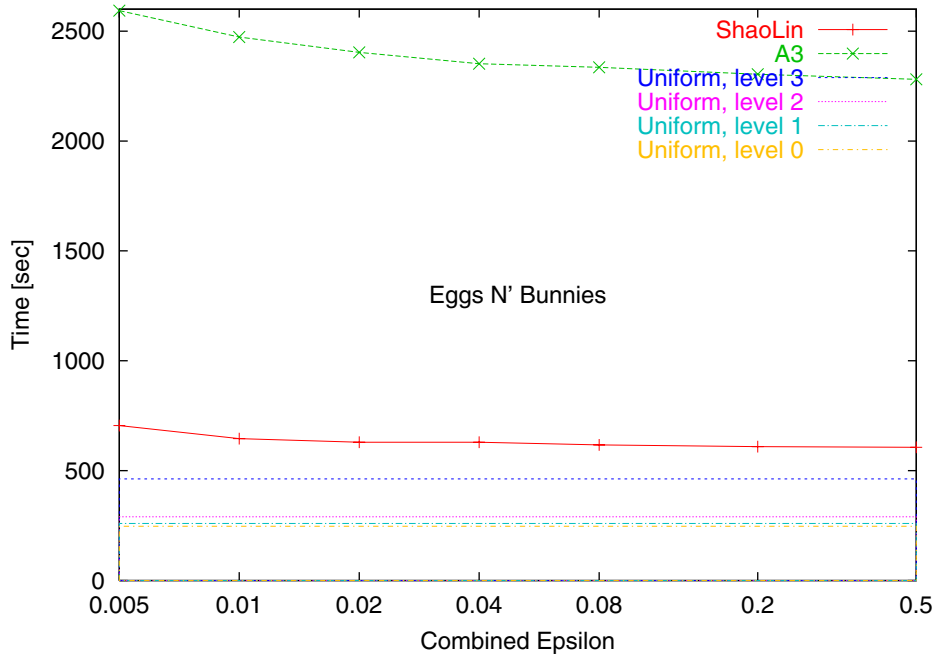
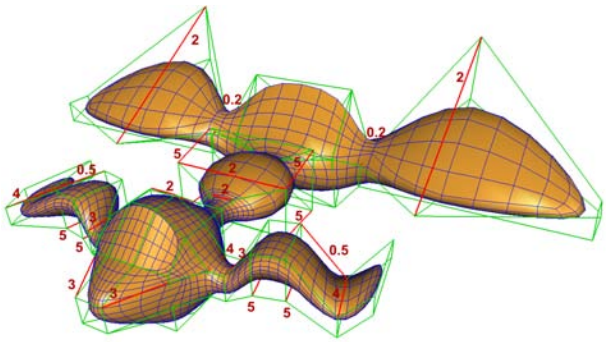


Abbildung 5.66: Laufzeitmessung für Testszene Eggs'n'Bunnies in Auflösung 800×800.

Kapitel 6

Zusammenfassung und Ausblick



6.1 Zusammenfassung

NURBS sind im Industriedesign die Standardflächen zum Modellieren von Freiformflächen. Mit den Vorteilen einer flexibleren Topologie und den interessanten Modellieroptionen durch Special Features sind Unterteilungsflächen jedoch zu einer attraktiven Alternative geworden.

In der vorliegenden Dissertation wurde eine neue, erweiterte Unterteilungsfläche (ESubs) entwickelt und implementiert, um die Vorteile von NURBS- und Catmull-Clark-Flächen nutzen zu können. ESubs verarbeiten eine beliebige zwei-mannigfaltige Topologie und besitzen ein Unterteilungsschema analog zu Catmull-Clark-Flächen. Das Kontrollnetz einer ESubs-Fläche enthält zusätzlich Knotenintervallwerte auf allen seinen Kanten. Da ESubs eine Verallgemeinerung von Catmull-Clark- und NURBS-Flächen sind, können sowohl Catmull-Clark- als auch bikubische NURBS-Flächen mit ESubs dargestellt werden. Bei uniformen Knotenintervallen liefert der ESubs-Regelsatz eine Catmull-Clark-Fläche. Unter Verwendung einer regulären Topologie im Viereckskontrollnetz mit gleichen Knotenintervallen auf gegenüber liegenden Kanten eines Faces (konkorme Faces) resultiert eine bikubische NURBS-Fläche. Die Knotenintervalle auf den Kanten des ESubs-Kontrollnetzes repräsentieren dann die beiden Knotenintervallvektoren der NURBS-Fläche. Liegen in einem Kontrollnetz mit beliebiger Topologie nur konforme Faces vor, so sind die ESubs stationär. Ohne Einschränkungen sind die ESubs-Flächen im Allgemeinen nicht-uniform und nicht-stationär.

Eine Modifikation der Knotenintervalle ist möglich, womit beispielsweise Special Features zu erzielen sind. Special Features werden mit Unterteilungsflächen herkömmlicherweise über eine Erweiterung des Regelsatzes realisiert. Dieses Verfahren wurde auf die ESubs übertragen und um nicht-uniforme Knotenintervalle erweitert. Somit bieten die ESubs eine breite Palette von Möglichkeiten an, Akzente und Features im Modell zu setzen.

Der Regelsatz der ESubs basiert auf den NURBS- und Catmull-Clark-Regeln. Um den Regelsatz einer neuen Unterteilungsfläche zu erhalten, werden üblicherweise zunächst die Unterteilungsregeln in Abhängigkeit der gewünschten Eigenschaften spezifiziert. Aus den Unterteilungsregeln wird die Unterteilungsmatrix S bestimmt, woraus mit Hilfe einer Eigenanalyse von S die Limitpunktregeln berechnet werden können. Für nicht-stationäre Unterteilungsschema kann dieses herkömmliche Verfahren jedoch nicht angewendet werden, da sich die Unterteilungsmatrix S bei jedem Unterteilungsschritt ändert. Daher wurde ein neues Verfahren entwickelt, um für nicht-stationäre Unterteilungsflächen den Regelsatz mit Limitpunktregeln zu bestimmen. Limitpunktregeln sind für eine Vielzahl von Anwendungen in der adaptiven Visualisierung sowie im CAD-Bereich unabdingbar. Durch die Verfügbarkeit von Limitpunktregeln ist ein sinnvoller Einsatz der Fläche in der Praxis erst möglich. ESubs sind die einzigen Unter-

teilungsflächen im Bereich Verallgemeinerung von Catmull-Clark- und bikubische NURBS-Flächen, die Limitpunktregeln anbieten.

Bei der Herleitung des ESubs-Regelsatz wird zwischen den Regeln für reguläre und irreguläre Topologien unterschieden. Im regulären Flächenbereich erhält jeder Vertex zwei lokale Knotenintervallvektoren, die aus den Knotenintervallen auf den Kanten des Kontrollnetzes bestimmt werden. Aus diesen lokalen Knotenintervallvektoren des Vertex lassen sich die lokalen Bézier-Kontrollpunkte rund um den Vertex analog zu den NURBS-Flächen berechnen. Die Bézier-Kontrollpunkte dienen als temporäre Variablen zur Berechnung der neuen Face- und Kantenpunkte. Der Limitpunkt eines Vertex wird mit Hilfe der NURBS-Limitpunktregeln aus der 1-Nachbarschaft des Vertex und seinen beiden Knotenintervallvektoren festgelegt. Der neue Vertexpunkt wird aus den umgebenen neuen Face- und Kantenpunkten sowie seinem Limitpunkt bestimmt. Somit wird sichergestellt, dass der neue Vertexpunkt gegen seinen festgelegten Limitpunkt konvergiert. Für die irregulären Vertices und ihre 1-Nachbarschaft werden die Catmull-Clark-Unterteilungsregeln benutzt. Als Limitpunktregel in den irregulären Vertices werden ebenfalls die Catmull-Clark-Limitpunktregeln verwendet. Damit liegt ein vollständiger Regelsatz vor, der die gewünschten Flächeneigenschaften erfüllt.

Bei Verwendung positiver Knotenintervalle ohne Special Features sind ESubs C^1-stetig an den irregulären Punkten und C^2-stetig im übrigen Bereich der Fläche mit Ausnahme des Randes der nichtkonformen Faces. Numerische Tests lassen jedoch mindestens G^1-Stetigkeit im Randbereich der nichtkonformen Faces erwarten.

Angelehnt an den Modellierungsprozess von NURBS- oder Catmull-Clark-Flächen können Modelle für ESubs auf gleiche Weise neu generiert werden. Weiterhin bieten ESubs die Möglichkeit, bestehende Catmull-Clark-Modelle nutzen zu können. Getrimmte NURBS-Modelle stehen ebenfalls nach einer Konvertierung mit gegebener Abweichungstoleranz als ESubs-Modell zur Verfügung. Die konvertierten Modelle können in einem ergänzenden Verarbeitungsschritt mit den erweiterten Optionen der ESubs-Flächen modifiziert werden. Mit den ESubs steht somit eine Fläche zur Verfügung, welche die Vorteile von NURBS- und Unterteilungsflächen vereint und in den Einsatzbereichen beider Flächentypen genutzt werden kann.

Der zweite Themenschwerpunkt der vorliegenden Arbeit behandelt die adaptive Visualisierung von Unterteilungsflächen. Dabei werden nur die Teile der Unterteilungsfläche in adäquater Tiefe berechnet und dargestellt, die an der Beleuchtungsrechnung teilnehmen und zur Verbesserung des Bildes beitragen. Dieser adaptive Ansatz wurde konsequent sowohl bei der interaktiven als auch bei der photorealistischen Visualisierung umgesetzt. Die entwickelten Verfahren liefern qualitativ hochwertige Bilder unter effizienter Nutzung der Rechenleistung und des Speichers, wobei bei der interaktiven Darstellung der Schwerpunkt auf einer hohen Framerate und bei der photorealistischen Visualisierung der Fokus auf einem hochwertigen Bild liegt.

Bei der interaktiven Darstellung erhält jedes Patch Bild für Bild eine geeignete Unterteilungstiefe zugeordnet. Der Klassifizierungsalgorithmus wertet dabei die Krümmung des Patches, die projizierte Größe und die Silhouettenzugehörigkeit des Patches aus. Da bei der interaktiven Visualisierung eine möglichst schnelle Bearbeitung Vorrang hat, wird eine effiziente Approximation der Entscheidungskriterien eingesetzt. Die patchweise Tesselierung des Ausgangskontrollnetzes wird mit Hilfe einer speziellen Datenstruktur (Slates) durchgeführt, welche das Ausgangskontrollnetz unverändert lässt. Die Slates besitzen eine statische Größe und sind unabhängig von der Größe des Ausgangskontrollnetzes. Die berechneten Limitpunkte können zwischengespeichert werden, um eine Beschleunigung zu erzielen. Eine Frameratenkontrolle ist durch eine dynamische Verschiebung der zugeordneten Unterteilungstiefen realisiert, die so genannte Simulator Sickness kann somit vermieden werden. Das Verfahren wurde für Dreiecks- und Viereckskontrollnetze entworfen und für Loop- und Catmull-Clark-Unterteilungsflächen implementiert. Das Verfahren zur interaktiven Darstellung von Unterteilungsflächen ist bestens geeignet für den Einsatz in einem Szenengraphsystem und wurde daher in das Szenengraphsystem OpenSG integriert.

Um Unterteilungsflächen photorealistisch darzustellen, wurden neue Methoden in das Softwaretool Modular Rendering Tool (MRT) des Instituts für ComputerGraphik der TU Braunschweig integriert. Zur Visualisierung von Unterteilungsflächen mittels Radiosity wurde das bestehende Verfahren im MRT um den Objekttyp Catmull-Clark-, ESubs- und Loop-Unterteilungsfläche erweitert. Die adaptive Verfeinerung resultiert aus der hierarchischen Radiosity-Berechnung in Abhängigkeit des Energieaustausches der Patches. Zum adaptiven Raytracen von Unterteilungsflächen wurde in dieser Arbeit ein neues Verfahren (ShaoLin-Algorithmus) entwickelt und implementiert. Schneidet ein zu testender Strahl den Hüllkörper eines Patches, so startet der ShaoLin-Algorithmus. Das Verfahren arbeitet hauptsächlich zweidimensional in einer Ebene, die durch die Eckpunkte des Patches definiert ist. Dabei wird der „Schattenwurf“ des Strahls in der Ebene verfolgt, welcher als Parallelprojektion entlang der Ebenennormale definiert ist. Wirft ein Strahl einen Schatten auf das Patch und ist der Strahl nahe genug am Patch, so ist ein Schnittpunkt möglich. Das Patch wird dann rekursiv entsprechend den Verfeinerungskriterien Krümmung, Silhouettenzugehörigkeit und Patchgröße unterteilt bis entweder ein Schnittpunkt gefunden wurde oder auszuschließen ist, dass ein Schnittpunkt existiert. Das Verfahren ist robust sowie effizient und kann beispielsweise auch für eine einfache Kollisionserkennung und für eine interaktive Auswahl eingesetzt werden.

6.2 Ausblick

Unterteilungsflächen sind hauptsächlich in Modellierungspaketen wie z.B. Maya, Cinema4D vertreten. Im Industriedesign-Bereich sind sie allerdings bislang nicht in Erscheinung getreten. Mit den ESubs stehen dem Modellierer nun Optionen zur Verfügung, die über die Möglichkeiten einer Catmull-Clark- und NURBS-Fläche hinausgehen und somit einen weitgefächerteren Einsatz erlauben. Komplexe Modelle können mit einem zusammenhängenden Kontrollnetz erzeugt werden, lokale Verfeinerungen sind ebenso möglich wie der Einsatz einer beliebigen zwei-mannigfaltigen Topologie. Ein Trimmen der Fläche ist zur Modellierung komplexer Objekte nicht notwendig, womit auch die komplette Problematik des Trimmvorganges entfällt. Zur Modellerzeugung stehen die Standard-Modelliervorgänge von NURBS- und Catmull-Clark-Flächen zur Verfügung: Beispielsweise kann eine ESubs-Fläche durch eine Rotation einer NURBS-Kurve gewonnen werden sowie durch diverse Kontrollnetzoperationen wie Extrude, Split-Face, Edge-Flip, die eine flexible Topologie verlangen. ESubs sind auch im Industriedesign vielseitig verwendbar: Zum Beispiel können nicht-uniforme Flächenstücke problemlos im Modell eingefügt werden. Individuelle Akzente sind im Modell durch eine Reihe von Special Features ebenfalls auf einfache Weise zu realisieren. Eine Integration der ESubs in komplexe Modellierungssysteme im CAD-Umfeld ist eine sinnvolle Nutzung und Erweiterung der vorliegenden Arbeit.

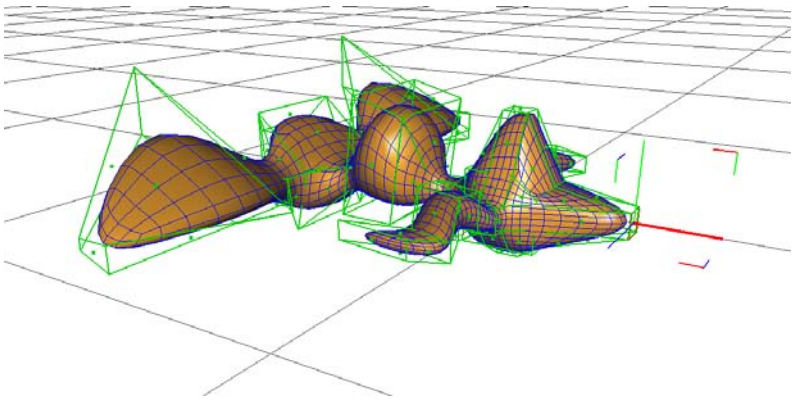


Abbildung 6.1: Modellierung mit ESubs.

NURBS-Modelle bestehen im Allgemeinen aus mehreren getrimmten NURBS-Patches, welche eine Zerlegung einer Zusammenhangskomponente bilden. Abbildung 6.2 zeigt ein solches Objekt, das aus 40 getrimmten Patches besteht. Die Handhabung eines solchen Objektes ist problematisch: Modellieroperationen, Texturierung und Tesselierung müssen beispielsweise geeignet auf alle einzelnen Teile des Modells übertragen werden. Weiterhin produziert die Tesselierung des Objektes im Trimmbereich keine gleichmäßigen Dreiecksnetze. ESubs besitzen die Eigenschaften bikubische NURBS-Flächen darzustellen und beliebige zwei-mannigfaltige Topologie verwenden zu können. Damit könnnte aus diesem Satz von getrimmten NURBS-Patches ein zusammenhängendes ESubs-Modell unter Einbeziehung einer gegebenen Abweichungstoleranz generiert werden. Ein solches zusammenhängendes ESubs-Modell vereinfacht die Modellierung und Texturierung deutlich. Die komplette Problematik des Trimmings entfällt und die Tesselierung der Fläche liefert somit ein gleichmäßigeres Dreiecksnetz.

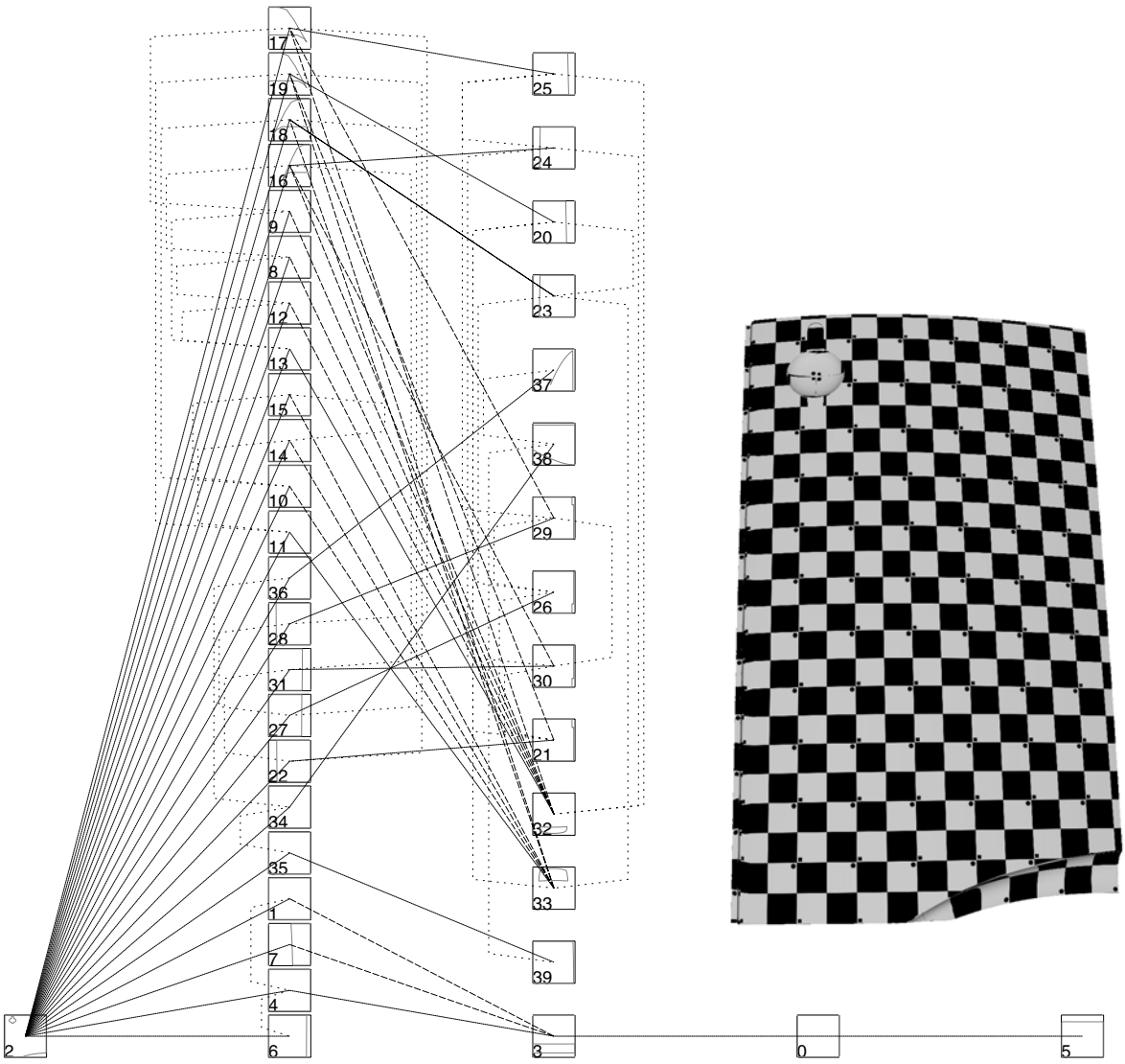


Abbildung 6.2: Texturiertes NURBS-Modell (rechts), bestehend aus 40 getrimmten Patches. Die 40 getrimmten Patches bilden eine Zerlegung einer Zusammenhangskomponente und sind über ihre Trimmkurven miteinander verbunden. Die Nachbarschaften der Patches und ihre Trimmkurven in ihrem Parametergebiet sind links dargestellt. Modell von Volkswagen AG, Nachbarschaftsgraph von Christoph Fünfzig.

Aufbauend auf die ESubs aus dieser Dissertation kann eine neue Kombifläche aus NURBS- und Unterteilungsflächen erarbeitet werden, die ihren Schwerpunkt von Unterteilungsflächen auf parametrische Flächen verlagert. Die günstigen Eigenschaften von NURBS- und Unterteilungsflächen könnten in der neuen Kombifläche zur Verfügung stehen und ebenso wie bei den ESubs ein breites Einsatzspektrum in Industrie und Forschung ermöglichen. Die ESubs orientieren sich in Aufbau und Herleitung an den Unterteilungsflächen. Bei der neuen Kombifläche sollten hingegen die parametrischen Flächen im Mittelpunkt stehen und die Konstruktion der Fläche vorgeben. Analog zu den NURBS-Flächen sollte eine patchweise Parametrisierung gegeben sein, die eine exakte Auswertung an beliebigen Parameterwerten ermöglicht. Die gewohnte Flexibilität in der Modellierung mit Unterteilungsflächen wäre auch bei dieser neuen Kombifläche wünschenswert: Der Anwender sollte auf eine beliebige Topologie und Special Features zurückgreifen können. Ebenso wie bei den ESubs könnten bei dieser neuen Kombifläche bikubische NURBS-Flächen identisch dargestellt werden sowie mit den Modellieroptionen der neuen Fläche weiter bearbeitet werden.

Eine vielversprechende Weiterentwicklung der vorliegenden Arbeit ist die Integration der ESubs in einen industriellen Fertigungsprozess. In einem zukunftsorientierten Ansatz könnte der komplette Modellierungs- und Vorverarbeitungsvorgang im Rechner ablaufen. Beispielsweise kann eine Autokonsole in einer Cave analog zum Modellieren mit einem echten Tonmodell erstellt werden: Über ein Trackingsystem sind die Bewegungen des Modellierers in der Cave am Freiformflächenmodell umzusetzen, so dass der Designer sein bearbeitetes Modell interaktiv formen kann. Mit den ESubs stehen dem Designer dabei Werkzeuge aus der NURBS- und Unterteilungsflächenmodellierung zur Verfügung. Arbeiten im Bereich Haptik verstärken die Realitätsnähe und erleichtern die Modellierung in einer virtuellen Umgebung. Durch den Einsatz der virtuellen Modellierung steht das Modell den übrigen Teilnehmern des Fertigungsprozesses unmittelbar zur Verfügung. Ein frühzeitiges Eingreifen und eine Korrektur des Modells durch alle Teilnehmer ist auf einfache Weise möglich. Es steht zudem eine Kommunikationsplattform zur Verfügung, die den sequentiellen Verarbeitungsprozess in einen parallelen teamorientierten Ansatz umwandelt. Durch den schnelleren Bearbeitungsprozess sind extreme Kosteneinsparungen zu erwarten, ebenso trägt die Wiederverwendung der virtuellen Modelle zur Kostensenkung bei.

Die entwickelten Konzepte der vorliegenden Arbeit bieten somit eine solide Basis für vielfältige Anwendungen im Bereich Modellierung und Visualisierung mit Unterteilungsflächen.

Anhang A

Ergänzungen zum Konvergenzbeweis

Um zu zeigen, dass Kontrollpunkt P_0^n für $n \to \infty$ gegen L konvergiert, ist Lemma 3 notwendig. In diesem Anhang wird Lemma 3 detailliert bewiesen und nochmal aufgeführt. Damit liegt der Konvergenzbeweis von Satz 3 vollständig vor.

Lemma 3. *Es sei P_0^0 ein Punkt des Ausgangskontrollnetzes M^0 und P_i^0, $i = 1,\ldots,8$ seine 1–Nachbarschaft. Punkt P_0^n bezeichnet den n-fach unterteilten Vertex P_0^0, erhalten mit dem ESubs-Regelsatz. Die 1–Nachbarschaft von P_0^n in Tiefe n erzeugt mit dem ESubs-Regelsatz sei P_i^n, $i = 1,\ldots,8$. Die Punkte, die mit den NURBS-Unterteilungsregeln aus P_i^n, $i = 0,\ldots,8$, $n \geq 0$, und den Knotenvektoren K_0^v, K_0^h von P_0^n generiert werden, seien R_i^{n+1}, $i = 0,\ldots,8$. D^n sei der Konvergenzradius von P_0^n aus Definition 8.*

Dann ist $|R_i^{n+1} - P_i^{n+1}|$, $i = 0,\ldots,8$, $n \geq 0$, beschränkt und es existiert für jedes $i = 0,\ldots,8$ eine Konstante H_i, unabhängig von der Unterteilungstiefe n, so dass gilt:

$$|R_i^{n+1} - P_i^{n+1}| \leq \tfrac{1}{2^n} \cdot H_i \cdot D^n$$

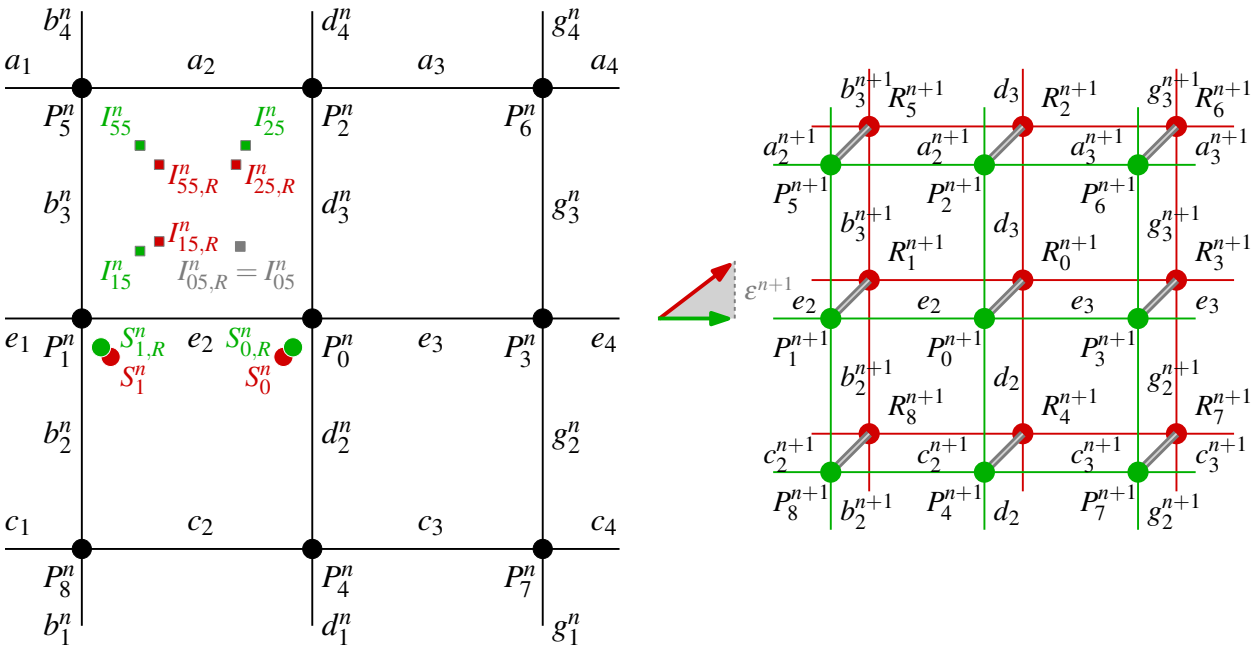


Abbildung A.1: Aus den Punkten P_i^n, $i = 0,\ldots,8$ sind mit Hilfe des ESubs-Regelsatzes die Punkte P_i^{n+1}, $i = 0,\ldots,8$ (grün dargestellt) generiert worden. Unter Verwendung der NURBS-Unterteilungsregeln mit den Knotenvektoren von P_0^n ergeben sich die Punkte R_i^{n+1}, $i = 0,\ldots,8$ (rot markiert).

Beweis Lemma 3:

Aus der Definition des ESubs-Regelsatzes folgt unmittelbar, dass in jeder Unterteilungstiefe n die Berechnung von L den gleichen Punkt ergibt. Ohne Beschränkung der Allgemeinheit sei L der Nullpunkt, damit gilt $D^n \geq |P_i^n|, i = 0,\ldots,8$.

Zum Beweis des Lemmas wird die Notation aus Abbildung A.1 verwendet: Die Punkte P_0^{n+1} und R_0^{n+1} sind die neuen Vertexpunkte; P_1^{n+1}, P_2^{n+1}, P_3^{n+1}, P_4^{n+1} und R_1^{n+1}, R_2^{n+1}, R_3^{n+1}, R_4^{n+1} bezeichnen die neuen Kantenpunkte; P_5^{n+1}, P_6^{n+1}, P_7^{n+1}, P_8^{n+1} und R_5^{n+1}, R_6^{n+1}, R_7^{n+1}, R_8^{n+1} sind die neuen Facepunkte. Die Abstandsabschätzung wird für den Vertexpunkt $|R_0^{n+1} - P_0^{n+1}|$ sowie exemplarisch für den Kantenpunkt $|R_1^{n+1} - P_1^{n+1}|$ und den Facepunkt $|R_5^{n+1} - P_5^{n+1}|$ durchgeführt. Die übrigen Abschätzungen für die Kantenpunkte $i = 2,3,4$ sowie die Facepunkte $i = 6,7,8$ laufen analog.

Abstandsabschätzung für Facepunkte:

Für den ESubs-Facepunkt P_5^{n+1} und für den Facepunkt R_5^{n+1}, erzeugt mit dem NURBS-Regelsatz, gilt (siehe Abbildung A.1):

$$\begin{aligned} P_5^{n+1} &= \frac{1}{4}\cdot(I_{05}^n + I_{15}^n + I_{55}^n + I_{25}^n) \\ R_5^{n+1} &= \frac{1}{4}\cdot(I_{05,R}^n + I_{15,R}^n + I_{55,R}^n + I_{25,R}^n) \end{aligned}$$

wobei I_{ab}^n die inneren lokalen Bézier-Kontrollpunkte der ESubs in Unterteilungstiefe n sind (siehe Abschnitt 3.3.1). Mit $I_{ab,R}^n$ werden die regulären inneren Bézier-Kontrollpunkte in Tiefe n bezeichnet, zu deren Berechnung die beiden Knotenvektoren von P_0^n zugrunde gelegt werden. Der Abstand zwischen dem NURBS-Facepunkt R_5^{n+1} und dem ESubs-Facepunkt P_5^{n+1} ist somit:

$$\begin{aligned} \left|R_5^{n+1} - P_5^{n+1}\right| &= \left|\frac{1}{4}\cdot(I_{05,R}^n + I_{15,R}^n + I_{55,R}^n + I_{25,R}^n) - \frac{1}{4}\cdot(I_{05}^n + I_{15}^n + I_{55}^n + I_{25}^n)\right| \\ &\leq \frac{1}{4}\cdot(|I_{05,R}^n - I_{05}^n| + |I_{15,R}^n - I_{15}^n| + |I_{55,R}^n - I_{55}^n| + |I_{25,R}^n - I_{25}^n|) \end{aligned} \tag{A.1}$$

Zur Abschätzung von $|R_5^{n+1} - P_5^{n+1}|$ werden die Abstände zwischen den inneren Bézier-Kontrollpunkten untersucht. Bei der Berechnung von $I_{05,R}^n$ und I_{05}^n werden die gleichen Knotenvektoren verwendet, daher gilt:

$$|I_{05,R}^n - I_{05}^n| = 0 \tag{A.2}$$

Für die Bestimmung von I_{15}^n werden die Knotenintervallvektoren von P_1^n benötigt,

$$K_1^v = \{b_2^n, b_2^n, b_3^n, b_3^n\}, \quad K_1^h = \{e_2, e_2, e_2, e_3\} \quad \text{mit} \quad b_j^n = d_j + \frac{1}{2^n}\cdot(b_j - d_j),\ j = 2,3$$

$I_{15,R}^n$ benutzt hingegen die Knotenvektoren von P_0^n. Daher ergibt sich für ihren Abstand:

$$|I_{15,R}^n - I_{15}^n| = \left|\frac{(e_2+e_3)\cdot(2\cdot d_3)\cdot P_1^n + (e_2+e_3)\cdot d_2\cdot P_5^n + e_2\cdot d_2\cdot P_2^n + e_2\cdot(2\cdot d_3)\cdot P_0^n}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)} - \right.$$

$$\left.\frac{(e_2+e_3)\cdot(2\cdot b_3^n)\cdot P_1^n + (e_2+e_3)\cdot b_2^n\cdot P_5^n + e_2\cdot b_2^n\cdot P_2^n + e_2\cdot(2\cdot b_3^n)\cdot P_0^n}{(2\cdot e_2+e_3)\cdot(b_2^n+2\cdot b_3^n)}\right|$$

Eine Umordnung nach $P_0^n, P_1^n, P_2^n, P_5^n$ liefert

$$
\begin{aligned}
&\leq \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot(b_2^n+2\cdot b_3^n)} \right| \cdot \\
&(|(e_2+e_3)\cdot 2\cdot(d_3\cdot(b_2^n+2\cdot b_3^n)-b_3^n\cdot(d_2+2\cdot d_3))|\cdot|P_1^n|+ \\
&|(e_2+e_3)\cdot(d_2\cdot(b_2^n+2\cdot b_3^n)-b_2^n\cdot(d_2+2\cdot d_3))|\cdot|P_5^n|+ \\
&|e_2\cdot(d_2\cdot(b_2^n+2\cdot b_3^n)-b_2^n\cdot(d_2+2\cdot d_3))|\cdot|P_2^n|+ \\
&|e_2\cdot 2\cdot(d_3\cdot(b_2^n+2\cdot b_3^n)-b_3^n\cdot(d_2+2\cdot d_3))|\cdot|P_0^n|) \\
&= \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot(b_2^n+2\cdot b_3^n)} \right| \cdot \\
&(|(e_2+e_3)\cdot 2\cdot(d_3\cdot b_2^n-b_3^n\cdot d_2)|\cdot|P_1^n|+|(e_2+e_3)\cdot 2\cdot(d_2\cdot b_3^n-b_2^n\cdot d_3)|\cdot|P_5^n|+ \\
&|e_2\cdot 2\cdot(d_2\cdot b_3^n-b_2^n\cdot d_3)|\cdot|P_2^n|+|e_2\cdot 2\cdot(d_3\cdot b_2^n-b_3^n\cdot d_2)|\cdot|P_0^n|)
\end{aligned}
$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n}\cdot(b_2-d_2)$ und $b_3^n = d_3 + \frac{1}{2^n}\cdot(b_3-d_3)$ folgt

$$
\begin{aligned}
&= \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot((d_2+\frac{1}{2^n}\cdot(b_2-d_2))+2\cdot(d_3+\frac{1}{2^n}\cdot(b_3-d_3)))} \right| \cdot \\
&(|(e_2+e_3)\cdot 2\cdot(d_3\cdot(d_2+\frac{1}{2^n}\cdot(b_2-d_2))-(d_3+\frac{1}{2^n}\cdot(b_3-d_3))\cdot d_2)|\cdot|P_1^n|+ \\
&|(e_2+e_3)\cdot 2\cdot(d_2\cdot(d_3+\frac{1}{2^n}\cdot(b_3-d_3))-(d_2+\frac{1}{2^n}\cdot(b_2-d_2))\cdot d_3)|\cdot|P_5^n|+ \\
&|e_2\cdot 2\cdot(d_2\cdot(d_3+\frac{1}{2^n}\cdot(b_3-d_3))-(d_2+\frac{1}{2^n}\cdot(b_2-d_2))\cdot d_3)|\cdot|P_2^n|+ \\
&|e_2\cdot 2\cdot(d_3\cdot(d_2+\frac{1}{2^n}\cdot(b_2-d_2))-(d_3+\frac{1}{2^n}\cdot(b_3-d_3))\cdot d_2)|\cdot|P_0^n|) \\
&= \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot((d_2+2\cdot d_3)+\frac{1}{2^n}\cdot((b_2-d_2)+2\cdot(b_3-d_3)))} \right| \cdot \\
&(|(e_2+e_3)\cdot 2\cdot(\frac{1}{2^n}\cdot(d_3\cdot b_2-d_2\cdot b_3))|\cdot|P_1^n|+ \\
&|(e_2+e_3)\cdot 2\cdot(\frac{1}{2^n}\cdot(d_2\cdot b_3-d_3\cdot b_2))|\cdot|P_5^n|+ \\
&|e_2\cdot 2\cdot(\frac{1}{2^n}\cdot(d_2\cdot b_3-d_3\cdot b_2))|\cdot|P_2^n|+|e_2\cdot 2\cdot(\frac{1}{2^n}\cdot(d_3\cdot b_2-d_2\cdot b_3))|\cdot|P_0^n|)
\end{aligned}
$$

Wegen $D^n \geq |P_i^n|, i = 0,\ldots,8$ gilt

$$\leq \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot((d_2+2\cdot d_3)+\frac{1}{2^n}\cdot((b_2-d_2)+2\cdot(b_3-d_3)))} \right| \cdot$$

$$(|(e_2+e_3)\cdot 2\cdot(\frac{1}{2^n}\cdot(d_3\cdot b_2-d_2\cdot b_3))|\cdot D^n+$$

$$|(e_2+e_3)\cdot 2\cdot(\frac{1}{2^n}\cdot(d_2\cdot b_3-d_3\cdot b_2))|\cdot D^n+$$

$$|e_2\cdot 2\cdot(\frac{1}{2^n}\cdot(d_2\cdot b_3-d_3\cdot b_2))|\cdot D^n+|e_2\cdot 2\cdot(\frac{1}{2^n}\cdot(d_3\cdot b_2-d_2\cdot b_3))|\cdot D^n)$$

$$= \frac{1}{2^n}\cdot D^n \cdot \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot((d_2+2\cdot d_3)+\frac{1}{2^n}\cdot((b_2-d_2)+2\cdot(b_3-d_3)))} \right| \cdot$$

$$(|(e_2+e_3)\cdot 2\cdot(d_3\cdot b_2-d_2\cdot b_3)|+|(e_2+e_3)\cdot 2\cdot(d_2\cdot b_3-d_3\cdot b_2)|+$$

$$|e_2\cdot 2\cdot(d_2\cdot b_3-d_3\cdot b_2)|+|e_2\cdot 2\cdot(d_3\cdot b_2-d_2\cdot b_3)|) \tag{A.3}$$

Da Term T_{15} entnommen aus Formel (A.3) mit

$$T_{15} = \left| \frac{1}{(2\cdot e_2+e_3)\cdot(d_2+2\cdot d_3)\cdot((d_2+2\cdot d_3)+\frac{1}{2^n}\cdot((b_2-d_2)+2\cdot(b_3-d_3)))} \right| \cdot$$

$$(|(e_2+e_3)\cdot 2\cdot(d_3\cdot b_2-d_2\cdot b_3)|+|(e_2+e_3)\cdot 2\cdot(d_2\cdot b_3-d_3\cdot b_2)|+$$

$$|e_2\cdot 2\cdot(d_2\cdot b_3-d_3\cdot b_2)|+|e_2\cdot 2\cdot(d_3\cdot b_2-d_2\cdot b_3)|)$$

beschränkt ist, existiert eine Konstante H_{15} unabhängig von Unterteilungstiefe n, so dass gilt

$$T_{15} \leq H_{15}$$

Damit resultiert für die Abstandsabschätzung der innerern Bézier-Kontrollpunkte $I_{15,R}^n$ und I_{15}^n:

$$|I_{15,R}^n - I_{15}^n| \leq \frac{1}{2^n}\cdot H_{15}\cdot D^n \tag{A.4}$$

Analog lassen sich für $|I_{55,R}^n - I_{55}^n|$ und $|I_{25,R}^n - I_{25}^n|$ die Konstanten H_{55} und H_{25} berechnen:

$$|I_{55,R}^n - I_{55}^n| \leq \frac{1}{2^n}\cdot H_{55}\cdot D^n \tag{A.5}$$

$$|I_{25,R}^n - I_{25}^n| \leq \frac{1}{2^n}\cdot H_{25}\cdot D^n \tag{A.6}$$

Schließlich ergibt sich für die Abstandsabschätzung zwischen dem NURBS-Facepunkt R_5^{n+1} und dem ESubs-Facepunkt P_5^{n+1} aus Formel (A.1):

$$|R_5^{n+1} - P_5^{n+1}| \leq \frac{1}{4}\cdot(|I_{05,R}^n - I_{05}^n| + |I_{15,R}^n - I_{15}^n| + |I_{55,R}^n - I_{55}^n| + |I_{25,R}^n - I_{25}^n|)$$

Formel (A.2), (A.4), (A.5), (A.6) eingesetzt liefert

$$\leq \frac{1}{4} \cdot (0 + \frac{1}{2^n} \cdot H_{15} \cdot D^n + \frac{1}{2^n} \cdot H_{55} \cdot D^n + \frac{1}{2^n} \cdot H_{25} \cdot D^n)$$

$$= \frac{1}{2^n} \cdot \frac{1}{4} \cdot (H_{15} + H_{55} + H_{25}) \cdot D^n$$

Aus $H_5 = \frac{1}{4} \cdot (H_{15} + H_{55} + H_{25})$ folgt schließlich

$$|R_5^{n+1} - P_5^{n+1}| \leq \frac{1}{2^n} \cdot H_5 \cdot D^n \tag{A.7}$$

Auf gleiche Art und Weise können die Abstandsabschätzungen zwischen den NURBS-Facepunkten R_6^{n+1}, R_7^{n+1}, R_8^{n+1} und den ESubs-Facepunkten P_6^{n+1}, P_7^{n+1}, P_8^{n+1} durchgeführt werden. Die entsprechenden Konstanten H_6, H_7, H_8 können ebenfalls mit derselben Vorgehensweise ermittelt werden. Somit erhält man als Abschätzung für die Facepunkte 6, 7 und 8:

$$|R_6^{n+1} - P_6^{n+1}| \leq \frac{1}{2^n} \cdot H_6 \cdot D^n \tag{A.8}$$

$$|R_7^{n+1} - P_7^{n+1}| \leq \frac{1}{2^n} \cdot H_7 \cdot D^n \tag{A.9}$$

$$|R_8^{n+1} - P_8^{n+1}| \leq \frac{1}{2^n} \cdot H_8 \cdot D^n \tag{A.10}$$

Abstandsabschätzung für Kantenpunkte:

Für den ESubs-Kantenpunkt P_1^{n+1} und für den Kantenpunkt R_1^{n+1}, generiert mit dem NURBS-Regelsatz, gilt (siehe Abbildung A.1):

$$P_1^{n+1} = \frac{1}{2 \cdot e_2 + e_3} \cdot ((\frac{e_2}{2} + e_3) \cdot S_1^n + (\frac{e_2}{2} + e_2) \cdot S_0^n))$$

$$R_1^{n+1} = \frac{1}{2 \cdot e_2 + e_3} \cdot ((\frac{e_2}{2} + e_3) \cdot S_{1,R}^n + (\frac{e_2}{2} + e_2) \cdot S_{0,R}^n))$$

wobei S_0^n, S_1^n Hilfspunkte zur Berechnung eines neuen ESubs-Kantenpunktes in Unterteilungstiefe n sind (siehe Abschnitt 3.3.1). Mit $S_{0,R}^n$, $S_{0,R}^n$ werden die regulären Hilfspunkte in Tiefe n bezeichnet, zu deren Berechnung die beiden Knotenvektoren von P_0^n zugrunde gelegt werden. Der Abstand zwischen dem NURBS-Kantenpunkt R_1^{n+1} und dem ESubs-Kantenpunkt P_1^{n+1} ist somit:

$$|R_1^{n+1} - P_1^{n+1}| = \left| \frac{1}{2 \cdot e_2 + e_3} \cdot ((\frac{e_2}{2} + e_3) \cdot S_{1,R}^n + (\frac{e_2}{2} + e_2) \cdot S_{0,R}^n)) - \right.$$

$$\left. \frac{1}{2 \cdot e_2 + e_3} \cdot ((\frac{e_2}{2} + e_3) \cdot S_1^n + (\frac{e_2}{2} + e_2) \cdot S_0^n)) \right|$$

$$= \left| \frac{\frac{e_2}{2} + e_3}{2 \cdot e_2 + e_3} \cdot (S_{1,R}^n - S_1^n) + \frac{\frac{e_2}{2} + e_2}{2 \cdot e_2 + e_3} \cdot (S_{0,R}^n - S_0^n) \right|$$

Zur Berechnung von $S_{0,R}$ und S_0 werden die gleichen Knotenvektoren verwendet, daher gilt $S_{0,R} = S_0$ und somit folgt

$$= \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot (S_{1,R} - S_1) \right|$$

mit Ausformulierung von $S_{1,R}$ und S_1

$$\begin{aligned} = \Bigg| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \Bigg(\frac{1}{2} \cdot \Bigg(\frac{1}{d_2+d_3} \cdot \Bigg(d_3 \cdot \frac{(d_2/2+d_3)\cdot P_8^n + (d_2/2+d_2)\cdot P_1^n}{2\cdot d_2+d_3} + \\ d_2 \cdot \frac{(d_3/2+d_3)\cdot P_1^n + (d_3/2+d_2)\cdot P_5^n}{2\cdot d_3+d_2} \Bigg) + P_1^n \Bigg) - \\ \frac{1}{2} \cdot \Bigg(\frac{1}{b_2^n+b_3^n} \cdot \Bigg(b_3^n \cdot \frac{(b_2^n/2+b_3^n)\cdot P_8^n + (b_2^n/2+b_2^n)\cdot P_1^n}{2\cdot b_2^n+b_3^n} + \\ b_2^n \cdot \frac{(b_3^n/2+b_3^n)\cdot P_1^n + (b_3^n/2+b_2^n)\cdot P_5^n}{2\cdot b_3^n+b_2^n} \Bigg) + P_1^n \Bigg) \Bigg) \Bigg| \end{aligned}$$

Eine Umordnung nach P_1^n, P_5^n, P_8^n liefert

$$\begin{aligned} = \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2} \right| \cdot \Bigg| \Bigg(\frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_3}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_3^n}{2\cdot b_2^n+b_3^n} \Bigg) \cdot P_8^n + \\ \Bigg(\frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_2}{2\cdot d_3+d_2} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_2^n}{2\cdot b_3^n+b_2^n} \Bigg) \cdot P_5^n + \\ \Bigg(\frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \\ \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n} \Bigg) \cdot P_1^n \Bigg| \\ \leq \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2} \right| \cdot \Bigg(\left| \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_3}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_3^n}{2\cdot b_2^n+b_3^n} \right| \cdot |P_8^n| + \\ \left| \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_2}{2\cdot d_3+d_2} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_2^n}{2\cdot b_3^n+b_2^n} \right| \cdot |P_5^n| + \\ \left| \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \right. \\ \left. \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n} \right| \cdot |P_1^n| \Bigg) \end{aligned} \tag{A.11}$$

Um $|R_1^{n+1} - P_1^{n+1}|$ abzuschätzen wird Formel (A.11) stückweise untersucht. Für

$$Teil_8 = \left| \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_3}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_3^n}{2\cdot b_2^n+b_3^n} \right| \cdot |P_8^n| \tag{A.12}$$

$$Teil_5 = \left| \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_2}{2\cdot d_3+d_2} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_2^n}{2\cdot b_3^n+b_2^n} \right| \cdot |P_5^n| \tag{A.13}$$

$$\begin{aligned} Teil_1 = \Bigg| & \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \\ & \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n} \Bigg| \cdot |P_1^n| \end{aligned} \tag{A.14}$$

wird die Existenz einer oberen Schranke überprüft. Die Ergebnisse führen zu einer oberen Schranke für die Abstände der Kantenpunkte $|R_1^{n+1} - P_1^{n+1}|$.

Abschätzung von $Teil_8$ aus Formel (A.11) und (A.12):
Für $Teil_8$ gilt

$$\begin{aligned} Teil_8 &= \left| \left(\frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_3}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_3^n}{2\cdot b_2^n+b_3^n}\right) \cdot P_8^n \right| \\ &= \left| \frac{d_3\cdot(d_2/2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_3^n+b_2^n) - b_3^n\cdot(b_2^n/2+b_3^n)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)}{(d_2+d_3)\cdot(2\cdot d_2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n)} \right| \cdot |P_8^n| \\ &= \left| \frac{Z_8}{N_8} \right| \cdot |P_8^n| \end{aligned} \tag{A.15}$$

Zur besseren Übersicht werden Zähler und Nenner von $Teil_8 = |\frac{Z_8}{N_8}| \cdot |P_8^n|$ getrennt untersucht. Für den Zähler Z_8 von $Teil_8$ aus Formel (A.15) ergibt sich:

$$Z_8 = d_3\cdot(d_2/2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n) - b_3^n\cdot(b_2^n/2+b_3^n)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n}\cdot(b_2-d_2)$ und $b_3^n = d_3 + \frac{1}{2^n}\cdot(b_3-d_3)$ folgt

$$\begin{aligned} &= d_3\cdot(d_2/2+d_3)\cdot\left((d_2+d_3)+\frac{1}{2^n}\cdot(b_2-d_2+b_3-d_3)\right)\cdot\left(2\cdot d_2+d_3+\frac{1}{2^n}\cdot(2\cdot(b_2-d_2)+b_3-d_3)\right) - \\ &\left(d_3+\frac{1}{2^n}\cdot(b_3-d_3)\right)\cdot\left(d_2/2+d_3+\frac{1}{2^n}\cdot\left(\frac{1}{2}\cdot(b_2-d_2)+b_3-d_3\right)\right)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3) \end{aligned}$$

Ausmultiplizieren und geeignetes Gruppieren liefert

$$
\begin{aligned}
&= d_3 \cdot (d_2/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) + \\
&\quad d_3 \cdot (d_2/2 + d_3) \cdot (d_2/2 + d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) + \\
&\quad d_3 \cdot (d_2/2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) + \\
&\quad d_3 \cdot (d_2/2 + d_3) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) - \\
&\quad d_3 \cdot (d_2/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) - \\
&\quad d_3 \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3) - \\
&\quad \frac{1}{2^n} \cdot (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (d_2/2 + d_3) - \\
&\quad \frac{1}{2^n} \cdot (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3) \\
&= \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2 + d_3) \cdot (d_2/2 + d_3) \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) + \\
&\qquad d_3 \cdot (d_2/2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) - \\
&\qquad d_3 \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3) - \\
&\qquad (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (d_2/2 + d_3) + \\
&\qquad \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) - \\
&\qquad\quad (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3))) \qquad \text{(A.16)}
\end{aligned}
$$

Z_8 ohne $\frac{1}{2^n}$, d.h. Formel (A.16) ohne $\frac{1}{2^n}$, wird im folgenden mit T_8 bezeichnet:

$$
\begin{aligned}
T_8(n) \quad = \quad & d_3 \cdot (d_2/2 + d_3) \cdot (d_2/2 + d_3) \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) + \\
& d_3 \cdot (d_2/2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) - \\
& d_3 \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3) - \\
& (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (d_2/2 + d_3) + \\
& \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3) \\
& \quad - (b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (\frac{1}{2} \cdot (b_2 - d_2) + b_3 - d_3))
\end{aligned}
$$

$T_8(n)$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt. Daher existiert für alle $n \in I\!N_0$ eine Konstante C_8^Z, für die gilt:

$$T_8(n) \leq C_8^Z$$

Somit ergibt sich als Abschätzung für den Zähler Z_8 von $Teil_8$:

$$Z_8 \quad \leq \quad \frac{1}{2^n} \cdot C_8^Z \tag{A.17}$$

Für den Nenner N_8 von $Teil_8$ aus Formel (A.15) ergibt sich:

$$N_8 = (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_2^n + b_3^n)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$= (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot ((d_2 + d_3) + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3))$$

$$\cdot ((2 \cdot d_2 + d_3) + \frac{1}{2^n} \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3))$$

Der Nenner N_8 von $Teil_8$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt, so dass eine Konstante C_8^N existiert, für die gilt:

$$N_8 \quad \geq \quad C_8^N$$

Damit ergibt sich für $Teil_8$ aus Formel (A.11) und (A.12):

$$Teil_8 = \left| (\frac{d_3}{d_2 + d_3} \cdot \frac{d_2/2 + d_3}{2 \cdot d_2 + d_3} - \frac{b_3^n}{b_2^n + b_3^n} \cdot \frac{b_2^n/2 + b_3^n}{2 \cdot b_2^n + b_3^n}) \right| \cdot |P_8^n|$$

$$\leq \frac{1}{2^n} \cdot \underbrace{\frac{C_8^Z}{C_8^N}}_{=C_8} \cdot |P_8^n|$$

$$= \frac{1}{2^n} \cdot C_8 \cdot |P_8^n|$$

Ebenso wird für $Teil_5$ verfahren:

Abschätzung von $Teil_5$ aus Formel (A.11) und (A.13):
Für $Teil_5$ gilt

$$Teil_5 = \left| (\frac{d_2}{d_2 + d_3} \cdot \frac{d_3/2 + d_2}{2 \cdot d_3 + d_2} - \frac{b_2^n}{b_2^n + b_3^n} \cdot \frac{b_3^n/2 + b_2^n}{2 \cdot b_3^n + b_2^n}) \cdot P_5^n \right|$$

$$= \left| \frac{d_2 \cdot (d_3/2 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n) - b_2^n \cdot (b_3^n/2 + b_2^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)}{(d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n)} \right| \cdot |P_5^n|$$

$$= \left| \frac{Z_5}{N_5} \right| \cdot |P_5^n| \tag{A.18}$$

Für den Zähler Z_5 von $Teil_5$ aus Formel (A.18) ergibt sich:

$$Z_5 = d_2 \cdot (d_3/2 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n) - b_2^n \cdot (b_3^n/2 + b_2^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$= d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3 + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3)) \cdot (2 \cdot d_3 + d_2 + \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2)) -$$

$$(d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)) \cdot (d_3/2 + d_2 + \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2)) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)$$

Ausmultiplizieren und geeignetes Gruppieren liefert

$$= d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) +$$

$$d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) +$$

$$d_2 \cdot (d_3/2 + d_2) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) +$$

$$d_2 \cdot (d_3/2 + d_2) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) -$$

$$d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) -$$

$$d_2 \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) -$$

$$\frac{1}{2^n} \cdot (b_2 - d_2) \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) -$$

$$\frac{1}{2^n} \cdot (b_2 - d_2) \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)$$

$$= \frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) +$$

$$d_2 \cdot (d_3/2 + d_2) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) -$$

$$d_2 \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) -$$

$$(b_2 - d_2) \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) +$$

$$\frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_2) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) -$$

$$(b_2 - d_2) \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2))) \qquad \text{(A.19)}$$

Z_5 ohne $\frac{1}{2^n}$, d.h. Formel (A.19) ohne $\frac{1}{2^n}$, wird im folgenden mit T_5 bezeichnet:

$$\begin{aligned} T_5(n) \quad = \quad & d_2 \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) + \\ & d_2 \cdot (d_3/2 + d_2) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) - \\ & d_2 \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\ & (b_2 - d_2) \cdot (d_3/2 + d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) + \\ & \frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_2) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\ & (b_2 - d_2) \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_2 - d_2) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)) \end{aligned}$$

$T_5(n)$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt. Daher existiert für alle $n \in I\!N_0$ eine Konstante C_5^Z, für die gilt:

$$T_5(n) \leq C_5^Z$$

Somit ergibt sich als Abschätzung für den Zähler Z_5 von $Teil_5$:

$$Z_5 \quad \leq \quad \frac{1}{2^n} \cdot C_5^Z$$

Für den Nenner N_5 von $Teil_5$ aus Formel (A.18) ergibt sich:

$$N_5 = (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$\begin{aligned} = & (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (d_2 + d_3 + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3)) \\ & \cdot (2 \cdot d_3 + d_2 + \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2)) \end{aligned}$$

Der Nenner von $Teil_5$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt, so dass eine Konstante C_5^N existiert, für die gilt:

$$N_5 \quad \geq \quad C_5^N$$

Damit ergibt sich schließlich für $Teil_5$ aus Formel (A.11) und (A.13):

$$\begin{aligned} Teil_5 \quad = \quad & \left| (\frac{d_2}{d_2 + d_3} \cdot \frac{d_3/2 + d_2}{2 \cdot d_3 + d_2} - \frac{b_2^n}{b_2^n + b_3^n} \cdot \frac{b_3^n/2 + b_2^n}{2 \cdot b_3^n + b_2^n}) \right| \cdot |P_5^n| \\ \leq \quad & \frac{1}{2^n} \cdot \underbrace{\frac{C_5^Z}{C_5^N}}_{=C_5} \cdot |P_5^n| \\ = \quad & \frac{1}{2^n} \cdot C_5 \cdot |P_5^n| \end{aligned}$$

Mit $Teil_1$ wird ebenso wie mit $Teil_8$ und $Teil_5$ verfahren:

Abschätzung von $Teil_1$ aus Formel (A.11) und (A.14):
Für $Teil_1$ gilt

$$
\begin{aligned}
Teil_1 &= \left| \left(\frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n}\right) \cdot P_1^n \right| \\
&= \left| \left(\frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n}\right) \right| \cdot |P_1^n| \\
&\leq \underbrace{\left| \left(\frac{d_3\cdot(d_2/2+d_2)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n) - b_3^n\cdot(b_2^n/2+b_2^n)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)}{(d_2+d_3)\cdot(2\cdot d_2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n)}\right) \right|}_{=S_A} \cdot |P_1^n| + \\
&\quad \underbrace{\left| \frac{d_2\cdot(d_3/2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_3^n+b_2^n) - b_2^n\cdot(b_3^n/2+b_3^n)\cdot(d_2+d_3)\cdot(2\cdot d_3+d_2)}{(d_2+d_3)\cdot(2\cdot d_3+d_2)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_3^n+b_2^n)} \right|}_{=S_B} \cdot |P_1^n| \\
&= S_A + S_B
\end{aligned} \tag{A.20}
$$

Beide Summanden S_A, S_B aus Formel (A.20) werden analog zu P_5^n und P_8^n analysiert, beginnend mit S_A:

$$
\begin{aligned}
S_A &= \left| \left(\frac{d_3\cdot(d_2/2+d_2)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n) - b_3^n\cdot(b_2^n/2+b_2^n)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)}{(d_2+d_3)\cdot(2\cdot d_2+d_3)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n)}\right) \right| \cdot |P_1^n| \\
&= \left| \frac{Z_A}{N_A} \right| \cdot |P_1^n|
\end{aligned} \tag{A.21}
$$

Für den Zähler Z_A von Summand S_A aus $Teil_1$ (Formel (A.21)) ergibt sich:

$$
Z_A = d_3\cdot(d_2/2+d_2)\cdot(b_2^n+b_3^n)\cdot(2\cdot b_2^n+b_3^n) - b_3^n\cdot(b_2^n/2+b_2^n)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)
$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n}\cdot(b_2-d_2)$ und $b_3^n = d_3 + \frac{1}{2^n}\cdot(b_3-d_3)$ folgt

$$
\begin{aligned}
&= d_3\cdot(d_2/2+d_2)\cdot\left(d_2+d_3+\frac{1}{2^n}\cdot(b_2-d_2+b_3-d_3)\right)\cdot\left(2\cdot d_2+d_3+\frac{1}{2^n}\cdot(2\cdot(b_2-d_2)+b_3-d_3)\right) - \\
&\quad \left(d_3+\frac{1}{2^n}\cdot(b_3-d_3)\right)\cdot\left(d_2/2+d_2+\frac{1}{2^n}\cdot((b_2-d_2)/2+b_2-d_2)\right)\cdot(d_2+d_3)\cdot(2\cdot d_2+d_3)
\end{aligned}
$$

Ausmultiplizieren und geeignetes Gruppieren liefert

$$
\begin{aligned}
&= d_3 \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) + \\
&\quad d_3 \cdot (d_2/2+d_2) \cdot \frac{1}{2^n} \cdot (b_2-d_2+b_3-d_3) \cdot (2 \cdot d_2+d_3) + \\
&\quad d_3 \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_2-d_2)+b_3-d_3) + \\
&\quad d_3 \cdot (d_2/2+d_2) \cdot \frac{1}{2^n} \cdot (b_2-d_2+b_3-d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_2-d_2)+b_3-d_3) - \\
&\quad d_3 \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) - \\
&\quad d_3 \cdot \frac{1}{2^n} \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) - \\
&\quad \frac{1}{2^n} \cdot (b_3-d_3) \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) - \\
&\quad \frac{1}{2^n} \cdot (b_3-d_3) \cdot \frac{1}{2^n} \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) \\
&= \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2+d_2) \cdot (b_2-d_2+b_3-d_3) \cdot (2 \cdot d_2+d_3) + \\
&\qquad d_3 \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot (b_2-d_2)+b_3-d_3) - \\
&\qquad d_3 \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) - \\
&\qquad \frac{1}{2^n} \cdot (b_3-d_3) \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) + \\
&\qquad \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2+d_2) \cdot (b_2-d_2+b_3-d_3) \cdot (2 \cdot (b_2-d_2)+b_3-d_3) - \\
&\qquad\quad (b_3-d_3) \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3))) \qquad \text{(A.22)}
\end{aligned}
$$

Z_A ohne $\frac{1}{2^n}$, d.h. Formel (A.22) ohne $\frac{1}{2^n}$, wird im folgenden mit T_A bezeichnet:

$$
\begin{aligned}
T_A(n) \quad = \quad & d_3 \cdot (d_2/2+d_2) \cdot (b_2-d_2+b_3-d_3) \cdot (2 \cdot d_2+d_3) + \\
& d_3 \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot (b_2-d_2)+b_3-d_3) - \\
& d_3 \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) - \\
& \frac{1}{2^n} \cdot (b_3-d_3) \cdot (d_2/2+d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3) + \\
& \frac{1}{2^n} \cdot (d_3 \cdot (d_2/2+d_2) \cdot (b_2-d_2+b_3-d_3) \cdot (2 \cdot (b_2-d_2)+b_3-d_3) - \\
& \quad (b_3-d_3) \cdot ((b_2-d_2)/2+b_2-d_2) \cdot (d_2+d_3) \cdot (2 \cdot d_2+d_3))
\end{aligned}
$$

$T_A(n)$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt. Daher existiert für alle $n \in I\!N_0$ eine Konstante C_A^Z, für die gilt:

$$T_A(n) \leq C_A^Z$$

Somit ergibt sich als Abschätzung für den Zähler von S_A:

$$Z_A \leq \frac{1}{2^n} \cdot C_A^Z$$

Für den Nenner N_A von Summand S_A aus $Teil_1$ (Formel (A.21)) ergibt sich:

$$N_A = (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_2^n + b_3^n)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$= (d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot ((d_2 + d_3) + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3))$$

$$\cdot ((2 \cdot d_2 + d_3) + \frac{1}{2^n} \cdot (2 \cdot (b_2 - d_2) + b_3 - d_3))$$

Der Nenner N_A von S_A ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt, so dass eine Konstante C_A^N existiert, für die gilt:

$$N_A \geq C_A^N$$

Damit ergibt sich folglich für den Summanden S_A aus $Teil_1$ von Formel (A.21):

$$S_A = \left| \left(\frac{d_3 \cdot (d_2/2 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_2^n + b_3^n) - b_3^n \cdot (b_2^n/2 + b_2^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_2 + d_3)}{(d_2 + d_3) \cdot (2 \cdot d_2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_2^n + b_3^n)} \right| \cdot |P_1^n|$$

$$\leq \frac{1}{2^n} \cdot \underbrace{\frac{C_A^Z}{C_A^N}}_{=C_A} \cdot |P_1^n|$$

$$= \frac{1}{2^n} \cdot C_A \cdot |P_1^n|$$

Der zweite Summand S_B von $Teil_1$ aus Formel (A.20) wird auf gleiche Weise wie S_A untersucht:

$$S_B = \left| \frac{d_2 \cdot (d_3/2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n) - b_2^n \cdot (b_3^n/2 + b_3^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)}{(d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n)} \right| \cdot |P_1^n|$$

$$= \left| \frac{Z_B}{N_B} \right| \cdot |P_1^n| \tag{A.23}$$

Für den Zähler Z_B von Summand S_B aus $Teil_1$ (Formel (A.23)) ergibt sich:

$$Z_B = d_2 \cdot (d_3/2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n) - b_2^n \cdot (b_3^n/2 + b_3^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$= d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3 + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3)) \cdot (2 \cdot d_3 + d_2 + \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2)) -$$

$$(d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)) \cdot (d_3/2 + d_3 + \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3)) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)$$

Ausmultiplizieren und geeignetes Gruppieren liefert

$$
\begin{aligned}
&= d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) + \\
&\quad d_2 \cdot (d_3/2 + d_3) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) + \\
&\quad d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) + \\
&\quad d_2 \cdot (d_3/2 + d_3) \cdot \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3) \cdot \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\
&\quad d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\
&\quad d_2 \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\
&\quad \frac{1}{2^n} \cdot (b_2 - d_2) \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\
&\quad \frac{1}{2^n} \cdot (b_2 - d_2) \cdot \frac{1}{2^n} \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \\
&= \frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) + \\
&\qquad d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\
&\qquad d_2 \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\
&\qquad (b_2 - d_2) \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) + \\
&\qquad \frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\
&\qquad\quad (b_2 - d_2) \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2))) \qquad \text{(A.24)}
\end{aligned}
$$

Z_B ohne $\frac{1}{2^n}$, d.h. Formel (A.24) ohne $\frac{1}{2^n}$, wird im folgenden mit T_B bezeichnet:

$$
\begin{aligned}
T_B(n) \quad = \quad & d_2 \cdot (d_3/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot d_3 + d_2) + \\
& d_2 \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\
& d_2 \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) - \\
& (b_2 - d_2) \cdot (d_3/2 + d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) + \\
& \frac{1}{2^n} \cdot (d_2 \cdot (d_3/2 + d_3) \cdot (b_2 - d_2 + b_3 - d_3) \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2) - \\
& \quad (b_2 - d_2) \cdot (\frac{1}{2} \cdot (b_3 - d_3) + b_3 - d_3) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2))
\end{aligned}
$$

$T_B(n)$ ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt. Daher existiert für alle $n \in I\!N_0$ eine Konstante C_B^Z, für die gilt:

$$T_B(n) \leq C_B^Z$$

Somit ergibt sich als Abschätzung für den Zähler von S_B:

$$Z_B \quad \leq \quad \frac{1}{2^n} \cdot C_B^Z$$

Für den Nenner N_B von S_B aus $Teil_1$ (Formel (A.21)) ergibt sich:

$$N_B = (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n)$$

Durch Einsetzen von $b_2^n = d_2 + \frac{1}{2^n} \cdot (b_2 - d_2)$ und $b_3^n = d_3 + \frac{1}{2^n} \cdot (b_3 - d_3)$ folgt

$$= (d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (d_2 + d_3 + \frac{1}{2^n} \cdot (b_2 - d_2 + b_3 - d_3))$$

$$\cdot (2 \cdot d_3 + d_2 + \frac{1}{2^n} \cdot (2 \cdot (b_3 - d_3) + b_2 - d_2))$$

Der Nenner von S_B ist für alle $n \in I\!N_0$ nach oben sowie nach unten beschränkt, so dass eine Konstante C_B^N existiert, für die gilt:

$$N_B \quad \geq \quad C_B^N$$

Damit folgt für den Summanden S_B aus $Teil_1$ von Formel (A.23):

$$S_B \quad = \quad \left| \frac{d_2 \cdot (d_3/2 + d_3) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n) - b_2^n \cdot (b_3^n/2 + b_3^n) \cdot (d_2 + d_3) \cdot (2 \cdot d_3 + d_2)}{(d_2 + d_3) \cdot (2 \cdot d_3 + d_2) \cdot (b_2^n + b_3^n) \cdot (2 \cdot b_3^n + b_2^n)} \right| \cdot |P_1^n|$$

$$\leq \quad \frac{1}{2^n} \cdot \underbrace{\frac{C_B^Z}{C_B^N}}_{=C_B} \cdot |P_1^n|$$

$$= \quad \frac{1}{2^n} \cdot C_B \cdot |P_1^n|$$

Damit ergibt sich für $Teil_1$ aus Formel (A.11) und (A.14):

$$Teil_1 = \left| \frac{d_3}{d_2 + d_3} \cdot \frac{d_2/2 + d_2}{2 \cdot d_2 + d_3} + \frac{d_2}{d_2 + d_3} \cdot \frac{d_3/2 + d_3}{2 \cdot d_3 + d_2} - \right.$$

$$\left. \frac{b_3^n}{b_2^n + b_3^n} \cdot \frac{b_2^n/2 + b_2^n}{2 \cdot b_2^n + b_3^n} - \frac{b_2^n}{b_2^n + b_3^n} \cdot \frac{b_3^n/2 + b_3^n}{2 \cdot b_3^n + b_2^n} \right| \cdot |P_1^n|$$

$$\leq S_A + S_B$$

$$\leq \frac{1}{2^n} \cdot C_A \cdot |P_1^n| + \frac{1}{2^n} \cdot C_B \cdot |P_1^n|$$

Mit $C_1 = C_A + C_B$ ergibt sich:

$$= \frac{1}{2^n} \cdot C_1 \cdot |P_1^n|$$

Für einen Kantenpunkt erhält man schließlich:

$$
\begin{aligned}
|R_1^{n+1} - P_1^{n+1}| \leq \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2} \right| \cdot \Bigg(& \left| \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_3}{2\cdot d_2+d_3} - \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_3^n}{2\cdot b_2^n+b_3^n} \right| \cdot |P_8^n| + \\
& \left| \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_2}{2\cdot d_3+d_2} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_2^n}{2\cdot b_3^n+b_2^n} \right| \cdot |P_5^n| + \\
& \left| \frac{d_3}{d_2+d_3} \cdot \frac{d_2/2+d_2}{2\cdot d_2+d_3} + \frac{d_2}{d_2+d_3} \cdot \frac{d_3/2+d_3}{2\cdot d_3+d_2} - \right. \\
& \left. \frac{b_3^n}{b_2^n+b_3^n} \cdot \frac{b_2^n/2+b_2^n}{2\cdot b_2^n+b_3^n} - \frac{b_2^n}{b_2^n+b_3^n} \cdot \frac{b_3^n/2+b_3^n}{2\cdot b_3^n+b_2^n} \right| \cdot |P_1^n| \Bigg) \\
\leq \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2} \right| \cdot & \left(\frac{1}{2^n} \cdot C_8 \cdot |P_8^n| + \frac{1}{2^n} \cdot C_5 \cdot |P_5^n| + \frac{1}{2^n} \cdot C_1 \cdot |P_1^n| \right) \\
\leq \left| \frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2} \right| & \cdot \frac{1}{2^n} \cdot (C_8+C_5+C_1) \cdot D^n
\end{aligned}
$$

Mit $H_1 = |\frac{\frac{e_2}{2}+e_3}{2\cdot e_2+e_3} \cdot \frac{1}{2}| \cdot (C_8+C_5+C_1)$ gilt:

$$|R_1^{n+1} - P_1^{n+1}| \leq \frac{1}{2^n} \cdot H_1 \cdot D^n$$

Die Abstandsabschätzungen der übrigen NURBS- und ESubs-Kantenpunkte $|R_i^{n+1} - P_i^{n+1}|$ für $i = 2,3,4$ (siehe Abbildung A.1) können analog durchgeführt werden. Auf gleiche Weise können die entsprechenden Konstanten H_i ermittelt werden, so dass gilt:

$$|R_i^{n+1} - P_i^{n+1}| \leq \frac{1}{2^n} \cdot H_i \cdot D^n, \qquad i = 1,\ldots,4$$

Abstandsabschätzung für Vertexpunkte

Aus der Definition des ESubs-Regelsatzes sowie dem Regelsatz der NURBS-Unterteilungsflächen folgt für den Punkt L

$$L = \sum_{i=0}^{8} \beta_i \cdot R_i^{n+1} = \sum_{i=0}^{8} \beta_i \cdot P_i^{n+1}$$

Der ESubs-Vertexpunkt P_0^{n+1} und der NURBS-Vertexpunkt R_0^{n+1} lässt sich daraus wie folgt berechnen:

$$
\begin{aligned}
R_0^{n+1} &= L - \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot R_i^{n+1} \\
P_0^{n+1} &= L - \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot P_i^{n+1}
\end{aligned}
$$

Der Abstand zwischen dem NURBS-Vertexpunkt R_0^{n+1} und dem ESubs-Vertexpunkt P_0^{n+1} ist somit:

$$|R_0^{n+1} - P_0^{n+1}| = \left| \left(L - \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot R_i^{n+1} \right) - \left(L - \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot P_i^{n+1} \right) \right|$$

$$|R_0^{n+1} - P_0^{n+1}| = \left| \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot (P_i^{n+1} - R_i^{n+1}) \right|$$

$$\leq \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot |R_i^{n+1} - P_i^{n+1}|$$

Aus der Abstandsabschätzung für Face- und Kantenpunkte folgt $|R_i^{n+1} - P_i^{n+1}| \leq \frac{1}{2^n} \cdot H_i \cdot D^n$ für $i = 1, \ldots, 8$ und somit

$$\leq \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot \frac{1}{2^n} \cdot H_i \cdot D^n$$

$$= (\frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot H_i) \cdot \frac{1}{2^n} \cdot D^n$$

mit $H_0 = \frac{1}{\beta_0} \cdot \sum_{i=1}^{8} \beta_i \cdot H_i$ ergibt sich

$$= \frac{1}{2^n} \cdot H_0 \cdot D^n$$

Somit ist $|R_i^{n+1} - P_i^{n+1}|$ für alle $i = 0, \ldots, 8$ beschränkt. Für jedes $i = 0, \ldots, 8$ existiert eine von der Unterteilungstiefe unabhängige Konstante H_i, so dass gilt

$$|R_i^{n+1} - P_i^{n+1}| \leq \frac{1}{2^n} \cdot H_i \cdot D^n, \qquad i = 0, \ldots, 8$$

■

Literaturverzeichnis

[Béz88] P. Bézier. Wie ein einfaches System entstand. In *Kurven und Flächen im Computer Aided Geometric Design*, pages 1–10. Vieweg, 1988. 1, 11

[BFK84] Wolfgang Boehm, Gerald Farin, and Jürgen Kahmann. A Survey of Curve and Surface Methods in CAGD. *Computer Aided Geometric Design*, (1):1–60, 1984. 1

[BFS96] Heinzgerd Bendels, Dieter Fellner, and Stephan Schäfer. Hierarchical Radiosity on Topological Data Structures. In *Proceedings of 3D Image Analysis and Synthesis'96*, pages 111–118, 1996. 100

[BKS00] S. Bischoff, L. Kobbelt, and H. P. Seidel. Towards Hardware Implementation of Loop Subdivision. In *Proceedings of Eurographics Workshop on Graphics Hardware*, pages 41–50, 2000. 74

[BM99] W. Boehm and A. Müller. On de Casteljau's Algorithm. *Computer Aided Geometric Design*, (16):587–605, 1999. 11

[BMZ04] Ioana Boier-Martin and Denis Zorin. Differentiable Parameterization of Catmull-Clark Subdivision Surfaces. In *Eurographics Symposium on Geometry Processing*, 2004. 103

[Bod05] Christian Bode. Photorealistische Visualisierung von Catmull-Clark-Unterteilungsflächen, 2005. Studienarbeit, Technische Universität Braunschweig. 102, 117, 118

[Boe77] W. Boehm. Über die Konstruktion von B-Spline-Kurven. *Computing*, (18):161–166, 1977. 11

[Boe80] W. Boehm. Inserting New Knots into B-Spline Curves. *Computer Aided Design*, (12):199–201, 1980. 11

[Bot] Mario Botsch. http://www.openmesh.org. OpenMesh Documentation. 75

[BS02] J. Bolz and P. Schröder. Rapid Evaluation of Catmull-Clark Subdivision Surfaces. In *Proceedings of Web3D'02*, pages 11–17, 2002. 74

[BSBK02] Mario Botsch, Stephan Steinberg, Stephan Bischoff, and Leif Kobbelt. OpenMesh - a Generic and Efficient Polygon Mesh Data Structure. In *Proceedings of OpenSG Symposium 2002*, 2002. 75

[CC78] E. Catmull and J. Clark. Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes. *Computer Aided Design*, (10):350–355, 1978. 1, 20, 28, 35

[CLR80] E. Cohen, T. Lyche, and R.F. Riesenfeld. Discrete B-Splines and Subdivision Techniques in Computer Aided Design and Computer Graphics. *Computer Graphics And Image Processing*, (14):87–111, 1980. 11

[CSS97] S. Campagna, P. Slusallek, and H.-P. Seidel. Ray Tracing of Spline Surfaces: Bézier Clipping, Chebyshev Boxing and Bounding Volume Hierarchy - A Critical Comparison with New Results. *The Visual Computer*, 13(1):265–282, 1997. 103

[dB72] C. de Boor. On Calculating with B-Splines. *J. Approx. Theory*, 6(1):50–62, 1972. 11

[dC63] P. de Casteljau. *Courbes et surfaces à pôle*. André Citroen Automobile SA, 1963. Microfiche P 4147-1. 11

[DS78] D. Doo and M. Sabin. Behaviour of Recursive Division Surfaces near Extraordinary Points. *Computer Aided Design*, (6):356–360, 1978. 1

[Far94] G. Farin. *Kurven und Flächen im Computer Aided Geometric Design*. Vieweg Verlag, 1994. 11, 14

[Fel92] W. D. Fellner. *Computergrafik*. BI Wissenschaftsverlag, 1992. 8

[Fel96] D. Fellner. Extensible Image Synthesis. In P. Wisskirchen, editor, *Object–Oriented and Mixed Programming Paradigms*, pages 7–21. Springer Verlag, 1996. 8, 100

[Fer64] J.C. Ferguson. Multivariable Curve Interpolation. *Journal of the ACM*, (2):221–228, 1964. 1

[FF04] Christoph Fünfzig and Dieter Fellner. Abschlussbericht Verbundprojekt OpenSG PLUS: Eine Open Source Echtzeit-Rendering-Bibliothek für VR- und AR-Anwendungen, June 2004. 75

[FHK02] G. Farin, J. Hoschek, and M. S. Kim. *Handbook of Computer Aided Geometric Design*. Elsevier, 2002. 1

[FS93] Thomas A. Funkhouser and Carlo H. Séquin. Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments. In *Proceedings of SIGGRAPH 93*, pages 247–254, 1993. 91

[GR74] W. Gordon and R.F. Riesenfeld. B-Spline Curves and Surfaces. In *Computer Aided Geometric Design*, pages 95–126. Academic Press, 1974. 1

[Hav02] S. Havemann. Interactive Rendering of Catmull-Clark Surfaces with Crease Edges. *The Visual Computer*, (18):286–298, 2002. 74

[HDD+94] H. Hoppe, T. DeRose, T. Duchamp, H. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. In *Proceedings of SIGGRAPH 1994*, pages 295–302, 1994. 1, 54

[HKB03] S. Hoffmann, H.-P. Krüger, and S. Buld. Vermeidung von Simulator-Sickness anhand eines Trainings zur Gewöhnung an die Fahrsimulatoren, 2003. Aus VDI-Berichte Nr. 1745. Simulation und Simulatoren - Mobilität virtuell gestalten. 8, 91

[HKD93] H. Halstead, M. Kass, and T. DeRose. Efficient, Fair Interpolation using Catmull–Clark Surfaces. In *Proceedings of SIGGRAPH 1993*, pages 35–44, 1993. 28, 31

[Hop97] H. Hoppe. View-Dependent Refinement of Progressive Meshes. In *Proceedings of SIGGRAPH 1997*, pages 189–198, 1997. 74

[Jan05] Rafael Janetzek. Ein hierarchisches Mesh für die adaptive Visualisierung von (erweiterten) Unterteilungsflächen, 2005. Diplomarbeit, Technische Universität Braunschweig. 100, 102, 117, 118

[KDS98] L. Kobbelt, K. Daubert, and H-P. Seidel. Ray Tracing of Subdivision Surfaces. In *Proceedings of 9th Eurographics Workshop on Rendering*, pages 69–80, 1998. 103

[Kob98] L. Kobbelt. Tight Bounding Volumes for Subdivision Surfaces. In *Proceedings of Pacific Graphics '98*, pages 17–26, 1998. 103

[LE97] D. Luebke and C. Erikson. View-Dependent Simplification of Arbitrary Polygonal Environments. In *Proceedings of SIGGRAPH 97*, pages 199–208, 1997. 74

[LG90] D. Lischinski and J. Gonczarowski. Improved Techniques for Ray Tracing Parametric Surfaces. *The Visual Computer*, 6(3):134–152, 1990. 103

[Loo87] C. Loop. Smooth Subdivision Surfaces Based on Triangles. Master's thesis, University of Utah, 1987. 1, 32

[MH00] K. Müller and S. Havemann. Subdivision Surface Tesselation on the Fly Using a Versatile Mesh Data Structure. In *Proceedings of Eurographics 2000*, volume 19, pages 151–159, 2000. 73, 74

[MRF06] K. Müller, L. Reusche, and D. Fellner. Extended Subdivision Surfaces: Building a Bridge between NURBS and Catmull-Clark Surfaces. *To appear in ACM Transactions on Graphics*, 2006. 35

[MTF03] K. Müller, T. Techmann, and D. Fellner. Adaptive Ray Tracing of Subdivision Surfaces. In *Proceedings of Eurographics 2003*, volume 22, pages 535–562, 2003. 73

[MU] Robert Nemiroff (MTU) and Jerry Bonnell (USRA). NASA Astronomy Picture of the Day Archive. http://antwrp.gsfc.nasa.gov/apod/archivepix.html. iii

[NSK90] T. Nishita, T. W. Sederberg, and M. Kakimoto. Ray Tracing Trimmed Rational Surface Patches. In *Proceedings of Computer Graphics and Interactive Techniques*, number 17, pages 337–345, 1990. 103

[Opea] OpenSG. http://www.opensg.org. 87

[Opeb] OpenSGPLUS. http://www.opensg.org/OpenSGPLUS/. 75

[PBP02] H. Prautzsch, W. Boehm, and M. Paluszny. *Bézier and B-Spline Techniques*. Springer Verlag, 2002. 16

[PS96] K. Pulli and M. Segal. Fast Rendering of Subdivision Surfaces. In *Proceedings of Eurographics Workshop on Rendering Techniques'96*, pages 61–70, 1996. 74, 79, 80, 95

[PT95] Les Piegl and Wayne Tiller. *The NURBS Book*. Springer-Verlag, 1995. 1, 4, 19, 20, 60, 63

[Ram87] L. Ramshaw. Blossoming: A Connect–the–Dot Approach to Splines. Technical report, Digital Systems Research Center, 1987. 11

[Reu03] Lars Reusche. Konvertierung von allgemeinen NURBS zu bikubischen NURBS mit Abweichungstoleranz zur weiteren Verwendung als erweiterte Unterteilungsflächen, 2003. Studienarbeit, Technische Universität Braunschweig. 63

[Reu05] Lars Reusche. Conversion of Trimmed NURBS Surfaces to Subdivision Surfaces. Master's thesis, 2005. Technische Universität Braunschweig. 2, 65

[RVB02] Dirk Reiners, Gerrit Voß and Johannes Behr. OpenSG - Basic Concepts. In *Proceedings of OpenSG Symposium 2002*, 2002. 8, 87

[Sab78] P. Sablonnière. Spline and Bézier Polygons Associated with a Polynomial Spline Curve. *Computer Aided Design*, (10):257–261, 1978. 11

[SAE93] Leon A. Shirman and Salim S. Abi-Ezzi. The Cone of Normals Technique for Fast Processing of Curved Patches. In *Proceedings of Eurographics 1993*, volume 12, pages 261–272, 1993. 76

[Sch97] Stephan Schäfer. Hierarchical Radiosity on Curved Surfaces. In *Proceedings of Eurographic Workshop on Rendering 1997*, pages 187–192, 1997. 100, 102

[Sch00] Stephan Schäfer. *Efficient Object-Based Hierarchical Radiosity Methods*. PhD thesis, 2000. Technical University Braunschweig. 100

[Set04] Volker Settgast. Subdivision Surfaces in OpenSG, 2004. Studienarbeit, Technische Universität Braunschweig. 75, 88, 89, 90, 92

[Sio] Anka Siol. Anka's 3D Welt. http://www.ankas-3dwelt.de/. iii

[SMFF03] V. Settgast, K. Müller, Ch. Fünfzig, and D. Fellner. Adaptive Tesselation of Subdivision Surfaces in OpenSG. In *Proceedings of OpenSG Symposium 2003*, pages 39–47, 2003. 73, 79

[SMFF04] V. Settgast, K. Müller, Ch. Fünfzig, and D. Fellner. Adaptive Tesselation of Subdivision Surfaces. *Computers & Graphics*, 28:73–78, 2004. 73

[Squ] Turbo Squid. 3D Models, Plugins, Textures, and more at Turbo Squid. http://www.turbosquid.com/. iii

[SSS98] Thomas W. Sederberg, David Sewell, and Malcolm Sabin. Non-Uniform Recursive Subdivision Surfaces. In *Proceedings of SIGGRAPH 1998 (July)*, pages 387–394, 1998. 3, 22, 35, 36, 52, 55

[Sta98] J. Stam. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Proceedings of SIGGRAPH 98*, pages 395–404, 1998. 74, 103

[Sta99] J. Stam. Evaluation of Loop Subdivision Surfaces. ACM SIGGRAPH 2000 Course Notes, 1999. 74, 103, 106

[SvSK03] A. Sovakar, A. von Studnitz, and L. Kobbelt. API Design for Adaptive Subdivision Schemes. In *Proceedings of OpenSG Symposium 2003*, pages 33–38, 2003. 74

[SZBN03] Thomas W. Sederberg, Jianmin Zheng, Almaz Bakenov, and Ahmad Nasri. T-Splines and T-NURCCs. In *Proceedings of SIGGRAPH 2003*, pages 477–484, 2003. 22, 35, 36

[TF04] Torsten Techmann and Dieter W. Fellner. Unterteilungsflächen: Datenstruktur und Regeln. Technical report, 2004. Technical report, Technische Universität Braunschweig. 100

[Tot85] D. T. Toth. On Ray Tracing Parametric Surfaces. In *Proceedings of SIGGRAPH 85*, number 12, pages 171–179, 1985. 103

[vdW93] B. L. van der Waerden. *Algebra I*. Springer Verlag, 1993. 14

[VW] Volkswagen AG. http://www.volkswagen.de. 71

[Woo89] C. Woodward. Ray Tracing Parametric Surfaces by Subdivision in Viewing Plane. In W. Strasser and H. P. Seidel, editors, *Theory and Practice of Geometric Modeling*. Springer Verlag, 1989. 103

[ZK02] D. Zorin and D. Kristjansson. Evaluation of Piecewise Smooth Subdivision Surfaces. *The Visual Computer*, 2002. 74

[ZS99] D. Zorin and P. Schröder. Subdivision for Modeling and Animation. ACM SIGGRAPH 1999 Course Notes, 1999. 1, 4, 7, 20, 21, 28, 32, 33, 34, 35, 45, 51

Eigene Veröffentlichungen

[1] A. Formella and K. Müller,
Bended Surfaces for Ray Tracing: A Fast View-Dependent Algorithm.
In *Proceedings of 3D Image Analysis and Synthesis'97*,
pages 19–26, 1997.

[2] K. Müller and S. Havemann,
Subdivision Surface Tesselation on the Fly Using a Versatile Mesh Data Structure,
In *Proceedings of Eurographics 2000*,
volume 19, pages 151–159, 2000.

[3] V. Settgast, K. Müller, Ch. Fünfzig, and D. Fellner,
Adaptive Tesselation of Subdivision Surfaces in OpenSG,
In *Proceedings of OpenSG Symposium 2003*,
pages 39–47, 2003.

[4] K. Müller, T. Techmann, and D. Fellner,
Adaptive Ray Tracing of Subdivision Surfaces,
In *Proceedings of Eurographics 2003*,
volume 22, pages 535–562, 2003.

[5] V. Settgast, K. Müller, Ch. Fünfzig, and D. Fellner,
Adaptive Tesselation of Subdivision Surfaces,
In *Computers & Graphics*,
pages 73–78, 2004.

[6] A. Formella and K. Müller,
A View Point Dependent Approach to Ray Trace Free-Form Surfaces,
In *Computer Graphics Forum*,
volume 23(2), pages 143–155, 2004.

[7] K. Müller, L. Reusche, and D. Fellner,
Extended Subdivision Surfaces: Building a Bridge between NURBS and Catmull-Clark Surfaces,
To appear in ACM Transactions on Graphics, 2006.