TECHNISCHE UNIVERSITÄT WIEN

Dissertation

Ein neuer Unterteilungsalgorithmus zur Schnittkurvenberechnung für parametrische Flächen

ausgeführt
zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

unter der Leitung von
em.Univ.Prof. Wilhelm Barth,
Institut 186 für Computergraphik und Algorithmen eingereicht
an der Technischen Universität Wien,
Fakultät für Technische Naturwissenschaften und Informatik,

von
Dipl.-Math. Katja Bühler,
Matrikelnummer 9727182,
Laudongasse 5/17,
A-1080 Wien, Österreich,
geboren am 22.10.1969 in Marktredwitz, Deutschland.

Wien, im Juni 2001.

Meinen Eltern.

Mein Dank gilt

Prof. Barth für die Betreuung und Unterstützung dieser Arbeit,

Prof. Purgathofer für die Übernahme des Koreferates,

meinen Kollegen für das angenehme Arbeitsklima und die fachliche und technische Unterstützung,

meiner ganzen Familie und allen meinen Freunden, für deren Geduld, Verständnis und die moralische Unterstützung.

Kurzfassung

Unterteilungsalgorithmen gehören zu den zuverlässigsten Algorithmen für den Schnitt zweier parametrischer Flächenpatches. Sie erfüllen vier der fünf Anforderungen an einen guten Schnittalgorithmus: Sie sind genau, zuverlässig, selbständig und auf alle Arten von parametrischen Flächen anwendbar - eine Kombination von Eigenschaften, die kaum ein anderer Schnittalgorithmus besitzt. Die hohe Laufzeit und ein großer Speicherverbrauch bei steigenden Anforderungen an die Genauigkeit der Ergebnisse, verhinderten allerdings weitgehend den praktischen Einsatz reiner Unterteilungsalgorithmen.

In dieser Arbeit werden mehrere Verbesserungen für Unterteilungsalgorithmen vorgeschlagen, deren zentrales Element eine neue Form von Hüllkörpern bildet: die *Linearen Intervallabschätzungen* (LIEs). Für sie werden, neben einer formalen Definition und Charakterisierung, zwei unterschiedliche sichere Berechnungsmethoden vorgestellt, die auf speziellen Taylor Modellen und Intervallarithmetik, bzw. auf der Ausnutzung der inneren Struktur von Affinformen und der damit verbundenen affinen Arithmetik basieren.

LIEs unterscheiden sich durch ihre parametrische Form essentiell von herkömmlichen kompakten Hüllkörpern. Sie erlauben eine völlig neue Behandlung der bei Unterteilungsalgorithmen auftretenden Teilprobleme, wie Schnittest, Parametergebietsreduzierung, Unterteilungsstrategie und der Verfeinerung der Ergebnisse. Für diese Teilprobleme werden neue Algorithmen entwickelt, die an die speziellen Eigenschaften der LIEs angepaßt sind und deren Gesamtheit einen neuen effizienten Unterteilungsalgorithmus bildet.

Schlagwörter: Parametrische Flächen; Schnitt; Schnittalgorithmus; Unterteilungsalgorithmus; Lineare Intervallabschätzung; LIE; Intervallarithmetik, affine Arithmetik; Taylor Modell; Hüllkörper.

Abstract

Subdivision algorithms are amongst the most reliable algorithms for the intersection of two parametric surface patches. They fit four of the five requirements on a good intersection algorithm: They are exact, reliable, independent and applicable to all kinds of parametric surfaces - a combination of properties that is hardly provided by any other type of algorithms to solve this problem. Though, the increasing consumption of memory and time in connection with higher requirements on the precision of the results inhibits the application of subdivision algorithms in practice up to now.

Several improvements for subdivision algorithms are proposed in this work. The center of these improvements is build by a new kind of bounding volumes: *Linear Interval Estimations* (LIEs). Besides of a formal definition and characterization of LIEs, two reliable methods for the computation of LIEs are introduced based on a new understanding of the use of affine arithmetic and a special application of Taylor Models and interval arithmetic.

LIEs differ in an essential way from conventional compact bounding volumes due to their parametric form. They allow a complete reconsideration of all occuring subproblems of subdivision algorithms, like the intersection test, the reduction of the parameter domains, subdivision strategies and the refinement of results. For all subproblems new algorithms adapted to the special characteristics of LIEs are proposed and combined to a new and efficient subdivision algorithm.

Keywords: parametric surfaces; intersection; subdivision algorithm; linear interval estimation; LIE; interval arithmetik, affine arithmetik; taylor models; bounding volumes.

Inhaltsverzeichnis

1

Einleitung

Seit Flächen zur Beschreibung von Objekten in Architektur, Design und Technik verwendet werden, versuchen Architekten, Künstler, Konstrukteure und Mathematiker Methoden zu entwickeln, mit denen Schnittkurven von Flächen dargestellt und berechnet werden können.

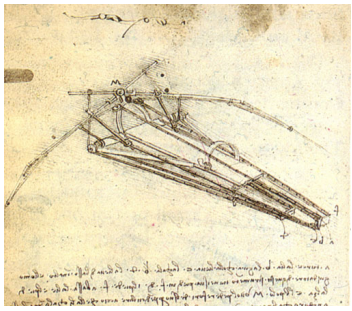
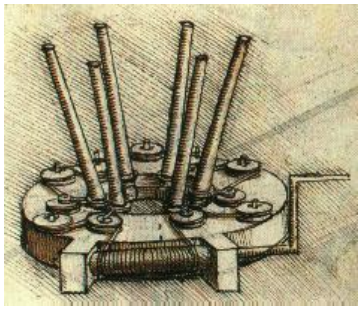
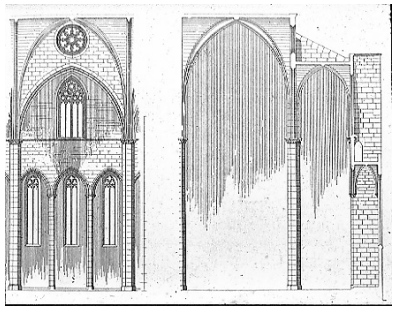




Abbildung 1.1: Zwei Konstruktionszeichnungen von Leonardo da Vinci (Flugmaschine (1488), Federkonstruktion (1498)) und Architekturzeichnung der Kirche Santa Maria del Bar, Barcelona (Baubeginn 1328)

1.1 Von der Bestimmung von Kegelschnitten...

Beschreibungen von Schnitten einfacher Flächen und Körper, wie Ebenen, Quader, Polyeder, Kugel, Kegel und Zylinder waren bereits Euklid (∼ 325–265 v. Chr.) und Aristaeus (∼ 370–300 v. Chr.) bekannt. Auf Appolonius von Perga (∼ 262–190 v. Chr.) ist der Begriff der Kegelschnitte zurückzuführen. Über diese schrieb er eine achtbändige Abhandlung, in die auch die Ergebnisse von Euklid und Aristaeus mit einflossen. Archimedes (287–212 v. Chr.) wendete diese Ergebnisse praktisch an: er beschrieb mit Hilfe von Ebene, Kugel, Zylinder und deren Schnitte Schwerpunkte bestimmter Konstruktionen.

Erst mit Hilfe der durch Descartes (1596–1650) begründeten analytischen Geometrie wurde es möglich, Flächen und deren Schnitte mathematisch exakt zu beschreiben. Während Descartes Philosoph und Physiker war, begann der Begründer der modernen Darstellenden Geometrie Gaspar Monge (1746–1818) seine Karriere als Konstrukteur und Zeichner. Er stellte Kurven und

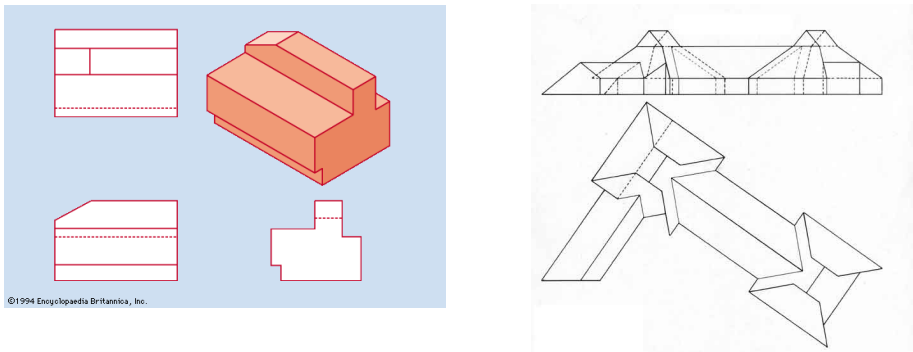


Abbildung 1.2: Neuere Konstruktionszeichnungen mit Hilfe Darstellender Geometrie: Bauteil und Dachkonstruktion

Flächen als Rißzeichnungen dar und entwickelte Regeln zur Konstruktion von Schnittkurven. Monge gilt zudem als Begründer der Differentialgeometrie, die später von Euler (1707–1783) und Gauss (1777-1855) weiterentwickelt wurde. Letzterer führte die Parameterdarstellung für Kurven und Flächen ein, die bis heute eine der wichtigsten Darstellungsformen ist.

Erst in den 70ger Jahren des 20. Jahrhunderts begannen Computer die Zeichentische der Konstrukteure abzulösen. Die klassische darstellenden Geometrie wurde durch die mathematische Beschreibung von Kurven und Flächen abgelöst, die konstruktive Schnittberechnung durch numerische oder iterative Ansätze zur Lösung des Problems. Eine Revolution in diesem Bereich stellten die von Bézier (Citroën) und DeCasteljau (Peugot) unabhängig entwickelten Bézierkurven und Flächen dar, die es erlaubten Flächen und Kurven mit Hilfe weniger Kontrollpunkte darzustellen, zu manipulieren und auf jedes beliebige System zu übertragen. Spezielle Eigenschaften der Bézierdarstellung ermöglichten auch die Entwicklung eines robusten iterativen Schnittalgorithmus. Die Notwendigkeit Flächen beliebiger Form mit günstigen aerodynamischen Eigenschaften darstellen zu können, trieb die Entwicklung weiter voran: Darstellungsformen für Spline-Kurven und -Flächen auf der Basis von Kontrollnetzen wurden in den nächsten Jahren von Mathematikern und Ingenieuren entwickelt, die größtenteils in den Entwicklungsabteilungen der Auto-, Flugzeug- und Schiffsindustrie arbeiteten.

In modernen CAD-Systemen spielen parametrische und implizite Flächendefinitionen neben Polygonnetzen und Unterteilungsflächen die wichtigste Rolle. Das Problem der robusten und schnellen Schnittberechnung zweier Flächen ist bis heute nur partiell gelöst: Werden nur begrenzt Einschränkung bezüglich zugelassener Flächentypen gemacht, erweist sich das Problem immer noch als sehr komplex. Exakte und vor allem schnelle Lösungen existieren nur für einige Spezialfälle. Die größer werdenden Anforderungen bezüglich Flexibilität im Design und bei der Kombination verschiedener Darstellungsmöglichkeiten von Flächen, lassen sich auch auf die Anforderungen an die Schnittalgorithmen übertragen.

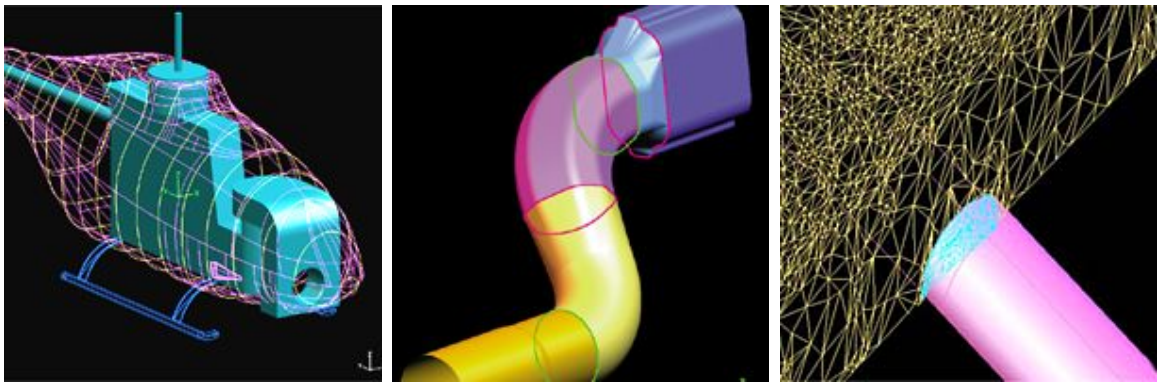


Abbildung 1.3: Beispiele für Konstruktionsmöglichkeiten durch die Verwendung unterschiedlicher Flächentypen in einem einzigen BRep-Modell. Die Bilder wurden den Webseiten der Tebis-AG (http://www.tebis.de/) entnommen.

1.2 ... zur Schnittberechnung in komplexen Flächenverbunden

Die am weitesten verbreiteten Methoden zum Entwurf komplexer Objekte in CAD/CAM-Systemen sind sogenannte Flächenverbunde, d.h. das Objekt wird aus mehreren Einzelflächen/Volumina zusammengesetzt. **Wichtigste Operation während des Designprozesses ist die Berechnung von Schnitten eines neu zugefügten Flächenstücks mit bereits vorhandenen Flächen und die exakte Bestimmung der neu entstandenen Topologie.**

Für Flächenverbunde existieren zwei bzw. drei verschiedene Konstruktions- und Darstellungsmöglichkeiten:

- **Constructive Solid Geometry (CSG)**: Ein CSG-Modell wird mit Hilfe von Mengenoperationen aus verschiedenen vorgegebene Primitiven zusammengesetzt. Dabei sind die Fläche und ihr Inneres gleichermaßen definiert.
- **Boundary Representation (BRep)**: Ein BRep-Modell beschreibt einen Körper nur durch seine orientierte Oberfläche, d.h. als Kombination von Eckpunkten, Kanten, Flächen und Orientierungen. Im topologischen Teil der Beschreibung werden Nachbarschaftsbeziehungen und Orientierungen der Elemente festgelegt, während in der geometrischen Beschreibung die Einbettung der Flächenelemente im Raum vorgenommen wird (z.B. die mathematische Beschreibung der Fläche). Flächen werden formal dabei so beschrieben, daß ihre Normalen nach außen zeigen. Praktisch wird die topologische Information eines BRep-Modells z.B. in Winged-Edge Darstellung ([Hoffmann '89]) oder als simple Adjazenzmatrix ([Krishnan et al. '97]) gespeichert.
- Eine Kombination der beiden Verfahren.

Beide Darstellungsarten haben jeweils Vor- und Nachteile. So ist ein CSG-Modell immer topologisch korrekt, während bei BRep-Modellen in jedem Konstruktionsschritt die topologische Korrektheit nachgeprüft werden muß. BRep-Modelle sind dafür leichter zu visualisiern als

CSG-Modelle. Für eine ausführliche Einführung in das Thema "Topologisch korrekte Boundary Represenations" siehe z.B. [Hoffmann '89] Kapitel 2.4.

Ein Algorithmus zum Schnitt zweier Körper in Boundary Representation wird in [Krishnan et al. '97] beschrieben. Dabei wird der Schnittberechnung der einzelnen Patches der größte Teil der benötigten Zeit zugeschrieben. Der Algorithmus wird in mehrere Teilschritte unterteilt:

1. Reduzierung der Anzahl der zu schneidenden Patches
   (a) Für alle Patches werden achsenparallele Einschließungen berechnet, die mit einem einfachen Sortieralgorithmus auf Schnitt untersucht werden.
   (b) Für Paare, die sich eventuell schneiden, werden engere Einschließungen berechnet und mittels linearer Programmierung bestimmt, ob eine separierende Ebene zwischen den beiden Volumen existiert.
2. Parallelisierung der Schnittprozedur für alle übriggebliebenen Patchpaare.
   (a) Schnittprozedur für ein einzelnes Patchpaar (in diesem Fall ein Verfolgungsalgorithmus mit Loop-Detection und Unterteilung des Parametergebietes)
   (b) Zusammensetzen der Schnittkurve des jeweiligen Patches.
3. Kombination der berechneten Schnittkurven der verschiedenen Patches.
4. Ergebnis ist eine stückweise Darstellung der Schnittkurve.

1.3 Ein neuer Schnittalgorithmus für parametrische Flächen

Unterteilungsalgorithmen gehören zu den zuverlässigsten Algorithmen für den Schnitt zweier parametrischer Flächenpatches. Sie erfüllen vier der fünf Anforderungen an einen guten Schnittalgorithmus: Sie sind genau, zuverlässig, selbständig und auf alle Arten von parametrischen Flächen anwendbar. Eine Kombination von Eigenschaften, die kaum ein anderer Schnittalgorithmus besitzt. Die hohe Laufzeit und ein großer Speicherverbrauch bei steigenden Anforderungen an die Genauigkeit der Ergebnisse, verhinderten allerdings weitgehend den praktischen Einsatz reiner Unterteilungsalgorithmen.

In dieser Arbeit werden mehrere Verbesserungen für Unterteilungsalgorithmen vorgeschlagen, deren zentrales Element eine neue Form von Hüllkörpern bildet: die *Linearen Intervallabschätzungen* (LIEs). LIEs unterscheiden sich durch ihre parametrische Form essentiell von herkömmlichen kompakten Hüllkörpern. Sie erlauben eine völlig neue Behandlung der bei Unterteilungsalgorithmen auftretenden Teilprobleme, wie Schnittest, Parametergebietsreduzierung, Unterteilungsstrategie und Verfeinerung der Ergebnisse.

Nach der Einführung grundlegender Begriffe im Zusammenhang mit parametrischen Flächen und einer analytisch/algebraischen Formulierung des Problems in Kapitel 2, wird in Kapitel 3

ein ausführlicher Überblick über bereits existierende Lösungen gegeben und die verschiedenen Ansätze werden diskutiert.

Die Betrachtungen in Kapitel 4 gelten der Analyse von Schwachpunkten von Unterteilungsalgorithmen und der Entwicklung von Konzepten zur Verbesserung, in deren Mittelpunkt die bereits oben erwähnten LIEs stehen.

LIEs werden im darauf folgenden Kapitel 5 definiert und charakterisiert: Basierend auf Intervallarithmetik, Taylor Modellen und affiner Arithmetik (siehe Anhang A), werden zwei verschiedene Berechnungsmethoden für LIEs entwickelt. Es wird gezeigt, daß LIEs zuverlässige und robuste Hüllkörper sind, die die Flächenstücke vergleichsweise eng einschließen, linear präzise und konvergent sind. Die lineare parametrische Struktur der LIEs erlaubt die Entwicklung eines einfachen Schnittest (Kapitel 6), der direkt von der Schnittsegmentberechnung für zwei Parallelogramme im Raum abgeleitet werden kann. Wichtigste Eigenschaft der LIEs ist ihre spezielle Parametrisierung, die in einem direkten Zusammenhang mit der Parametrisierung der eingeschlossenen Fläche steht. Sie ermöglicht die Übertragung der Ergebnisse des Schnitts zweier LIEs auf die von ihnen eingeschlossenen Flächen. Dadurch wird es möglich, den Schnittest für parametrische Flächen mit einer Parametergebietsreduzierung zu verbinden. Der Algorithmus ermöglicht auch die Berechnung von Intervallgeraden in den Parametergebieten und Intervallflächenkurven, die die eigentliche Schnittkurve einschließen. Unterteilungsstrategien und Abbruchkriterien des Unterteilungsalgorithmus werden in Kapitel 7 an die spezielle Struktur der LIEs angepaßt.

Es wurden verschiedene Versionen von Unterteilungsalgorithmen implementiert, die in Kapitel 8 beschrieben sind. In Kapitel 9 werden verschiedene Tests bezüglich Geschwindigkeit und Genauigkeit durchgeführt und bewertet. Die Arbeit endet mit einer Zusammenfassung und einem Ausblick auf mögliche Anwendungen, der neu entwickelten Hüllkörper auf andere Probleme.

Im Anhang A befindet sich ein Überblick über die wichtigsten Grundlagen der Intervallarithmetik, der Taylor Modelle und der affinen Arithmetik. Anhang B und C enthalten thematisch geordnete Bibliographien zu Schnittalgorithmen bzw. Arithmetiken.

1.4 Die wichtigsten Bezeichnungen und Vereinbarungen

Alle Betrachtungen diese Arbeit finden im zwei bzw. dreidimensionalen euklidischen Raum $\mathbb{E}^2(\mathbb{R})$ bzw. $\mathbb{E}^3(\mathbb{R})$ statt. Reelle Skalare werden mit kleinen Buchstaben ($\alpha \in \mathbb{R}$), reelle Punkte und Vektoren mit kleinen fetten Buchstaben ($\boldsymbol{v} \in \mathbb{R}^n$), reelle Matrizen mit großen fetten Buchstaben ($\boldsymbol{M} \in \mathbb{R}^{m \times n}$) bezeichnet. Die Menge der reellen kompakten Intervalle wird mit $\mathbb{IR}$ bezeichnet, der zugehörige Vektorraum mit $\mathbb{IR}^n$. Intervalle werden mit großen Buchstaben ($I \in \mathbb{IR}$), Intervallpunkte und -vektoren mit kleinen hohlen Buchstaben $\mathbb{v} \in \mathbb{IR}^n$ und Intervallmatrizen mit großen hohlen Buchstaben $\mathbb{A} \in \mathbb{IR}^{m \times n}$ bezeichnet.

2

Schnitt parametrischer Flächen: Die mathematische Formulierung des Problems

Parametrische Flächen gehören heute zu den wichtigsten Flächen im geometrischen Modellieren und bilden die Basis jeden CAD-Systems. Sie sind deshalb in ihrer allgemeinen Form Gegenstand aller weiteren Betrachtungen dieser Arbeit. Bézier- und NURBS-Flächen sind lediglich ein Spezialfall der allgemeinen parametrischen Flächen, es gelten also auch für sie alle in diesem und den folgenden Kapiteln getroffenen Aussagen.

2.1 Parametrische Flächen und Flächenkurven [Aumann '93]

Definition 2.1.1
Sei $G \subset \mathbb{R}^2$ ein Gebiet[1] *(das* Parametergebiet*) und*

$$\boldsymbol{f} : \begin{cases} G & \longrightarrow & \mathbb{E}^3 \\ (u,v) & \mapsto & \boldsymbol{f}(u,v) = (f_1(u,v), f_2(u,v), f_3(u,v))^T \end{cases}$$

eine C^r-Abbildung ($r \geq 1$). Dann heißt die Punktmenge

$$F := \{\boldsymbol{f}(u,v) | (u,v) \in G\}$$

*eine C^r-*Fläche *mit der* Parameterdarstellung $\boldsymbol{f}(u,v)$.

Ein Flächenpunkt $\boldsymbol{f}(u_0,v_0)$ heißt bezüglich $\boldsymbol{f}$ regulär*, wenn die Vektoren $\boldsymbol{f}_u(u_0,v_0) := \frac{\partial}{\partial u}\boldsymbol{f}(u_0,v_0)$ und $\boldsymbol{f}_v(u_0,v_0) = \frac{\partial}{\partial v}\boldsymbol{f}(u_0,v_0)$ linear unabhängig sind. Andernfalls heißt er singulär.*

Die Fläche F heißt bezüglich der Parameterdarstellung $\boldsymbol{f}$ regulär, wenn jeder Punkt $\boldsymbol{f}(u,v)$ regulär ist. $\boldsymbol{f}$ ist dann eine C^r- Immersion.

[1] Ein Gebiet ist eine offene und zusammenhängende Teilmenge eines topologischen Raumes

Definition 2.1.2
Eine reguläre Fläche heißt einfach, wenn $\boldsymbol{f}$ injektiv ist, also eine C^r-Einbettung.

Satz 2.1.1
Ist $F : \boldsymbol{f}(u,v), (u,v) \in G$ eine reguläre C^r-Fläche und

$$\boldsymbol{h} : \begin{cases} I \in I\!R & \longrightarrow & G \\ t & \mapsto & \boldsymbol{h}(t) := (u(t), v(t))^T \end{cases}$$

eine C^r-Abbildung $(r \geq 1)$ mit

$$\frac{d}{dt}\boldsymbol{h}(t) \neq \boldsymbol{o} \text{ für alle } t \in I$$

so ist die Flächenkurve

$$c : \boldsymbol{c}(t) := \boldsymbol{f}(\boldsymbol{h}(t)) = \boldsymbol{f}(u(t), v(t)),\ t \in I$$

von F eine reguläre Flächenkurve.

Spezielle Flächenkurven sind die *u-* bzw. *v-Linien* (auch: *Parameter-* oder *Isolinien*) von F. Die u-Linien erhält man für u variabel und $v = const$, die v-Linie analog. Die Parameterlinien besitzen die Tangentenvektoren $\boldsymbol{f}_u(u,v)$ bzw. $\boldsymbol{f}_v(u,v)$.

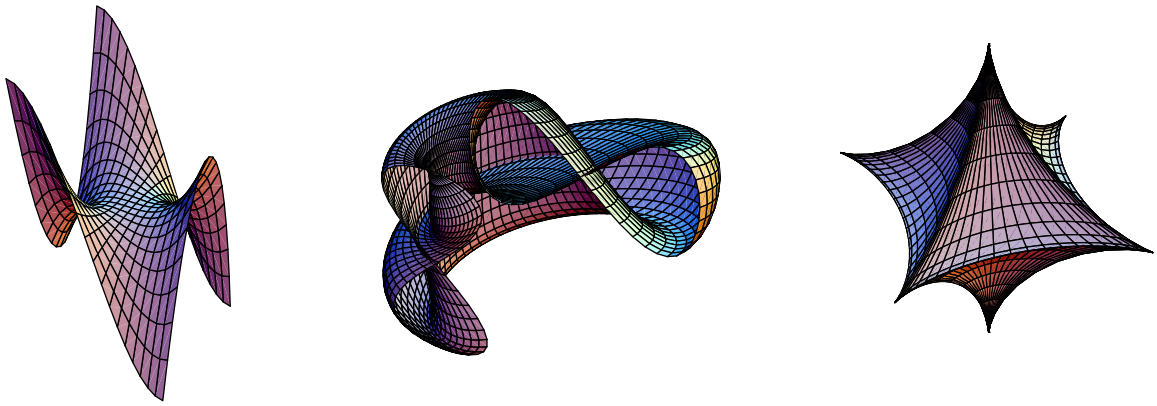


Abbildung 2.1: Einfache reguläre Fläche (Affensattel), Fläche mit Selbstdurchdringungen (Kleinsche Flasche) und singuläre Fläche (Astroidal). Alle Flächen sind mit ihren Isolinien dargestellt.

Der Begriff der *polynomialen/rationalen* Fläche stammt aus der algebraischen Geometrie: Als polynomial/rational werden solche Flächen (und auch Kurven) bezeichnet, für die eine polynomiale/rationale Parameterdarstellung existiert. Solche Flächen spielen in der Computergraphik eine spezielle Rolle, nur sie lassen sich in Bézier- oder NURBS-Form darstellen.

Eine parametrische Fläche heißt *algebraisch*, falls sie sich implizit in der folgenden Form darstellen läßt:

Definition 2.1.3
Die Gesamtheit aller Punkte $\mathbb{x}$ *des projektiven dreidimensionalen Raumes* $\mathcal{P}^3(C)$*, deren Koordinaten der Gleichung*

$$F_n(x,y,z,w) = \sum_{i+j+k+l=n} a_{ijkl} x^i\, y^j\, z^k\, w^l = 0$$

mit $a_{ijkl} \in \mathbb{R}$ *genügen, heißt algebraische Fläche vom Grad n.*

Alle Flächen, die sich rational parametrisieren lassen, sind gleichzeitig auch algebraische Flächen. Sie besitzen somit eine polynomiale implizite Form, was für einige Anwendungen im geometrischen Modellieren, aber auch für die Berechnung realistischer Darstellungen der Flächen von Bedeutung ist.

Vereinbarung:
Ab dem nächsten Kapitel werden parametrische Flächen meist als *Flächenstücke* oder *Patches* bezeichnet, da im Allgemeinen nicht Flächen als Ganzes untersucht werden, sondern nur kleine Teilstücke. Als Kurzform für eine Fläche $F : \boldsymbol{f}(u,v)$, $(u,v) \in \mathbb{I}$ wird auch die Bezeichnung $\boldsymbol{f}$ verwendet.

2.2 Analytische und algebraische Lösungsansätze für den Schnitt parametrischer Flächen

Gegeben seien zwei C^1-Flächenstücke F und G

$$F: \quad \boldsymbol{f}(u,v) = \begin{pmatrix} f_1(u,v) \\ f_2(u,v) \\ f_3(u,v) \end{pmatrix}, (u,v) \in I_u \times I_v =: \mathbb{I} \subset \mathbb{R}^2$$

$$G: \quad \boldsymbol{g}(s,t) = \begin{pmatrix} g_1(s,t) \\ g_2(s,t) \\ g_3(s,t) \end{pmatrix}, (s,t) \in J_s \times J_t =: \mathbb{J} \subset \mathbb{R}^2$$

Der Schnitt der beiden Flächen ist die Menge c aller gemeinsamen Punkte

$$c := \{\boldsymbol{x} \in \mathbb{R}^3 \mid \exists (u_0,v_0) \in \mathbb{I}, (s_0,t_0) \in \mathbb{J} : \boldsymbol{x} = \boldsymbol{f}(u_0,v_0) = \boldsymbol{g}(s_0,t_0)\}$$

d.h. die Lösungsmenge des unterbestimmten Gleichungssystems

$$\boldsymbol{f}(u,v) = \boldsymbol{g}(s,t),\ (u,v) \in \mathbb{I}, (s,t) \in \mathbb{J}$$

Liegt o.B.d.A. die Fläche F auch in impliziter Form $F: \ f(x,y,z,w) = 0$ vor, erhält man durch Substitution der homogenen Darstellung $\mathfrak{g}(s,t) = (g_1(s,t), g_2(s,t), g_3(s,t), g_0(s,t))^T$ eine implizite Gleichung der Parameterwerte der Schnittkurve in der s-t-Ebene:

$$f(g_1(s,t), g_2(s,t), g_3(s,t), g_0(s,t)) = 0$$

Liegen beide Flächen in impliziter Form vor $F : f(x, y, z, w) = 0$ und $G : g(x, y, z, w) = 0$, so kann c Lösungsmenge des Gleichungssystems

$$\begin{cases} f(x, y, z, w) = 0 \\ g(x, y, z, w) = 0 \end{cases}$$

bestimmt werden.

Sind beide Flächen algebraisch, so ist auch ihre Schnittkurve algebraisch.

Neben den klassischen Ansätzen existieren Beschreibungen des Schnittproblems über die orientierte Abstandsfunktion der beiden Flächen [Kriezis et al. '92], als singuläre Menge eines speziellen Matrixpolynoms [Krishnan, Manocha '97] und als Anfangswertproblem [Schramm '95].

2.3 Form der Schnittkurven

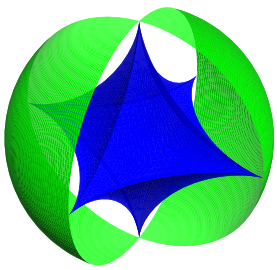
Die Lösungsmenge c kann folgende Formen annehmen:

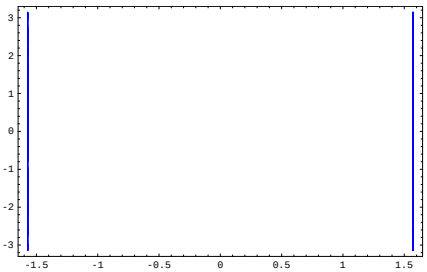
1. Es gibt keine Lösung - die beiden Flächen schneiden sich nicht.
2. Es gibt nur diskrete Lösungen, d.h. die beiden Flächen berühren sich in einzelnen Punkten.
3. Es gibt unendlich viele Lösungen und die beiden Flächen
   (a) haben eine gemeinsame Schnittkurve (einparametrige Lösung), die aus mehreren geschlossenen oder offenen Zweigen und singulären Punkten bestehen kann oder
   (b) überlappende sich teilweise, bzw. sind gleich (zweiparametrige Lösungsmenge).

Beispiele

Singuläre Punkte

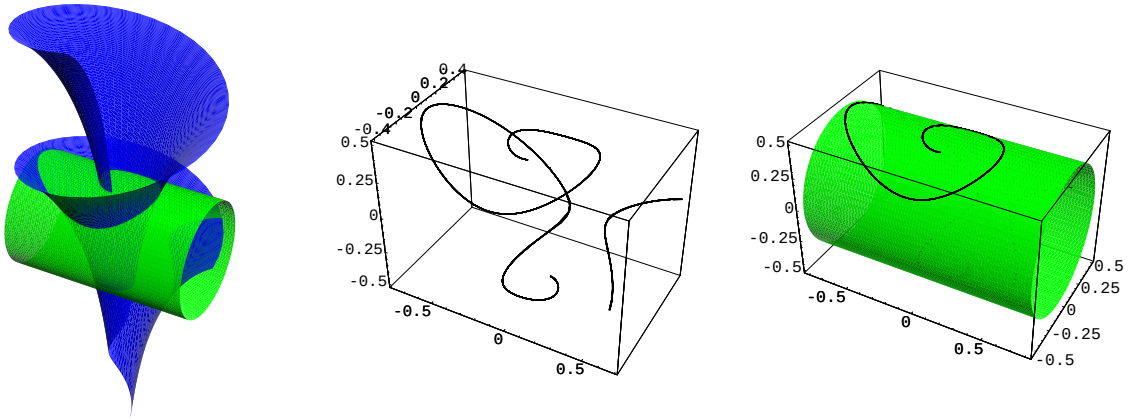
Schnitt der Einheitskugel mit dem Einheitsastroidal (Radius = 1). Das linke Bild zeigt den Astroidal und einen Teil der Kugel. Das rechte Bild zeigt die den sechs Schnittpunkten entsprechenden Parameterwerte im Parametergebiet der Kugel. Die beiden Geraden beschreiben die Parameterwerte des Schnittpunktes, der mit einem der Pole der Kugel zusammenfällt.





Zerfall in mehrere Zweige

Schnitt einer Dini-Fläche mit einem Zylinder. Die Schnittkurve zerfällt in zwei verschiedenen Zweige.



Überlappung von Flächenstücken

Überlappungen von Flächenpatches können in zwei Fällen auftreten:

- Wenn die Flächen stückweise definiert sind, also z.B. als Spline-Flächen (siehe Abbildung 2.2).
- Wenn beide Flächen C^∞-Flächen sind und beide Flächenstücke Teil der gleichen Fläche sind.

In [Hu et al. '97] werden einige Sätze zur Überlappung von Kurven- und Flächensegmenten aufgestellt. Prinzipiell gilt für C^∞ Kurven und Flächen:

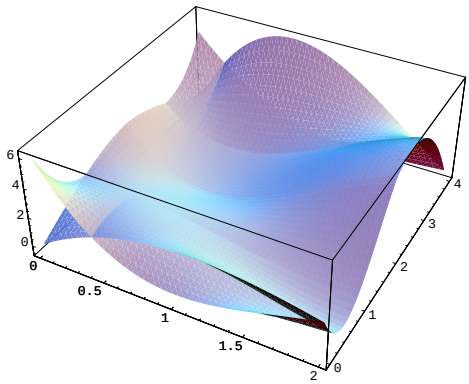


Abbildung 2.2: Zwei polynomiale C^1-Spline-Flächen, die sich teilweise überlappen.

Satz 2.3.1
Berühren sich zwei C^∞ Kurven oder Flächen entlang eines Kurven/Flächensegments, so sind sie gleich.

Daraus folgt direkt für C^∞ Kurven-/Flächensegmente

1. Überlappen sich zwei C^∞ Kurvensegmente c_1 und c_2 entlang eines Teilsegments, überdecken sie sich entweder überall oder der gemeinsame Teil endet an Begrenzungspunkten der Segmente.
2. Überlappen sich ein C^∞ Kurvensegmente c_1 und ein C^∞ Flächensegment C_2 entlang eines Teilsegments ohne Doppelpunkt, endet der gemeinsame Teil an Begrenzungspunkten der

Segmente.

3. Berühren sich zwei C^∞ Flächensegmente C_1 und ein C_2 entlang einer Kurve ohne Doppelpunkt, endet der gemeinsame Teil an Begrenzungspunkten der Segmente.

4. Überlappen sich zwei polynomiale Flächen, ist der sich überlappende Bereich von Rändern der beiden Flächensegmenten begrenzt.
   Zwei *allgemeine Flächen* (z.B. B-Splineflächen) können sich sehr wohl in einem inneren Teil der Fläche überlappen.

Die Kurvenberechnung kann abhängig vom gewählten Lösungsweg problematisch werden, wenn die Schnittkurve durch die spezielle Lage oder Form der Flächen eine besondere Form annimmt, z.B. wenn Überlappungen, Singularitäten oder Doppelpunkte auftreten. In so einem Fall spricht man auch von einem *schlecht konditionierten Problem.*

3

Existierende Lösungsansätze

In diesem Kapitel wird ein Überblick über bereits vorhandene Schnittalgorithmen für Flächen gegeben. Der Schwerpunkt liegt dabei auf der Behandlung des Problems für parametrische Flächen.

Beim Verschneiden parametrischer Flächen wird grundsätzlich zwischen geometrischem und parametrischem Verschneiden unterschieden [Fournier, Buchanan '94]:

- *geometrische Schnittalgorithmen* liefern als Ergebnis die Koordinaten der Schnittpunkte im Objektraum,
- *parametrische Schnittalgorithmen* die Parameterwerte der Schnittpunkte.

Parametrische Schnittalgorithmen haben in der Praxis die größere Bedeutung, da oft sowohl die Schnittpunkte im Objektraum als auch deren Parameterwerte benötigt werden, um später beispielsweise differentialgeometrische Eigenschaften (Tangenten, Normalen, Krümmungen) in den Punkten bzw. der Schnittkurve zu berechnen und das Wissen über die Parameterwerte eines Punktes immer auch das Wissen über seine Lage im Objektraum beinhaltet.

Die hier vorgestellten unterschiedlichen Lösungsansätze sind algebraischer, konstruktiver, diskreter, analytischer und iterativer Art und werden im letzten Abschnitt dieses Kapitels im Hinblick auf die in [Hoschek, Lasser '92] zusammengestellten Anforderungen an einen Schnittalgorithmus verglichen:

- **Genauigkeit** im numerischen Sinne,
- **Zuverlässigkeit** in dem Sinne, daß die Durchdringung vollständig und richtig ermittelt wird,
- **Schnelligkeit** im Hinblick auf interaktives Arbeiten,
- **Selbstständigkeit** in dem Sinne, daß die Durchdringung ohne interaktiven Eingriff des Benutzers ermittelt wird.

Auffällig ist, daß viele der existierenden Algorithmen sich auf bestimmte Flächentypen beschränken, z.B. Flächen mit polynomialer oder rationaler Parametrisierung oder noch spezieller auf Flächen in Bézier- oder NURBS-Darstellung. Nur wenige machen keinerlei Einschränkung bezüglich zulässiger Parametrisierungen (siehe ([Hohmeyer '91], [Huber '99], [Filip et al. '86], [Chang et al. '94], [Koparkar '91].
In Hoscheks Aufzählung fehlt ein Kriterium, das diese Tatsache mit einbezieht. Die Liste der Kriterien zur Bewertung von Schnittalgorithmen wird deshalb um folgende Forderung ergänzt:

- **Allgemeingültigkeit** in dem Sinne, daß es keinerlei Einschränkungen bezüglich der Parametrisierung und des Flächentyps geben soll.

3.1 Konstruktive Verfahren

Vor der Erfindung der Computer wurden Verfahren der Darstellenden Geometrie benutzt, um Schnitte von Flächen zeichnerisch zu bestimmen. In [Piegl '92] entwickelt Piegl Algorithmen, die diese Verfahren für CAD-Systeme nutzbar machen sollen. Die von ihm vorgestellten Verfahren beschränken sich allerdings nur auf sehr einfache Flächen, wie Quadriken, Dreh- und Regelflächen. Kern der vorgestellten Algorithmen ist die Dekomposition der Flächen in einfache ebene Kurven, die dann miteinander geschnitten werden. Für einfache Flächen können die Schnittkurven sehr stabil berechnet werden, da deren Konstruktion nicht auf numerischen Berechnungen basiert. Das gilt allerdings nur, wenn alle möglichen Sonderfälle genau implementiert werden und die Topologie des Schnittes vorher bekannt ist. Das vorgestellte Verfahren weist signifikante Nachteile auf: Um alle Sonderfälle abzufangen, muß für jedes mögliche Flächenpaar eine eigene Schnittroutine implementiert werden, da allgemeine Verfahren zur Schnittberechnung nicht existieren. Zudem muß der Benutzer die Topologie des Schnittes im Voraus bestimmen, das Programm ist also nicht selbständig.

3.2 Algebraische Lösungsverfahren

Das klassische algebraisches Lösungsverfahren und die dazu nötigen Techniken werden ausführlich in [Hoffmann '89] beschrieben. Vereinfacht läßt sich der Algorithmus folgendermaßen beschreiben: Eine der beiden (rationalen) Parameterdarstellungen wird mit Hilfe von Resultanten bzw. Gröbner Basen in eine implizite Darstellung der Fläche $f(x, y, z) = 0$ umgewandelt in die dann die Parameterdarstellung $\boldsymbol{g}(s, t)$ der zweiten Fläche eingesetzt wird. Ergebnis ist die implizite Gleichung der Schnittkurve im s-t-Parameterraum $f(g_1(s, t), g_2(s, t), g_3(s, t)) = 0$.
Obwohl der Algorithmus sehr stabil im Bezug auf Singularitäten und andere Sonderfälle reagiert, hat sich dieser Ansatz in der Praxis für Flächen höheren Grades (d.h. höher als vom Grad zwei) als nicht praktikabel erwiesen. Hoffmann ([Hoffmann '89]) nennt dafür folgende Gründe:

1. Die Bestimmung der impliziten Darstellung der einen Fläche verlangt umfangreiche symbolische Berechnungen.

2. Die Substitution der Parameterform der zweiten Fläche in die implizite Darstellung der ersten ist in der Praxis aufgrund möglicher Rundungsfehler nicht grundsätzlich stabil.

3. Die algebraische Darstellung der Schnittkurve hat im allgemeinen einen sehr hohen algebraischen Grad. (Im Falle des Schnittes zweier bikubischer Flächenstücke ist die Schnittkurve vom Grad ≤ 324)

3.3 Diskretisierungsmethoden

Im Bereich der Diskretisierungsmethoden werden zwei Methoden zur Schnittberechnung parametrischer Flächen unterschieden: die Gittermethode und die Parameterwertdiskretisierung. Die erste Methode diskretisiert beide Flächen zunächst durch Netze von Flächenpunkten, die dann stückweise linear bzw. bilinear approximiert werden. Die so vereinfachten Flächen werden geschnitten und liefern eine stückweise lineare bzw. quadratische Approximation der Schnittkurve. Für eine detaillierte Diskussion sei hier auf [Hoschek, Lasser '92] verwiesen.

Ein klassischer auf Parameterwertdiskretisierung basierender Algorithmus wird von [Wang '92] beschrieben und setzt sich aus folgenden Schritten zusammen:

1. Bestimmung einzelner Punkte auf der Schnittkurve durch Schneiden von Isolinien der einen Fläche mit der zweiten Fläche, bzw. durch Berechnung von Approximationen der Schnittpunkte und anschließender iterativer Verfeinerung der Ergebnisses mit Newtoniteration.

2. Sortieren der Punkte nach Zusammenhangskomponenten, wobei die Richtungen der Tangenten der Schnittkurve mit in Betracht gezogen werden. Treten Unsicherheiten auf, werden weitere Schnittpunkte berechnet.

3. Approximation der Punkte mit niedriggradigen polynomialen Kurven innerhalb einer vorgegebenen Toleranz.

4. Verifikation des Ergebnisses durch eine approximative Berechnung der Distanz der beiden Flächen an unsicheren Stellen.

3.4 Verfolgungsalgorithmen

Verfolgungsalgorithmen bestimmen, ausgehend von einem bereits bekannten Schnittpunkt der beiden Flächen, eine Folge von Punkten eines Zweiges der Schnittkurve, indem sie lokale differentialgeometrische Eigenschaften wie Tangentialebenen und Normalenvektoren der Flächen betrachten.

Die meisten Verfolgungsalgorithmen bestehen aus drei Phasen

1. Bestimmung von Startpunkten.

2. Verfolgung der einzelnen Zweige der Schnittkurve.

3. Sortierung der Ergebnisse, Parametrisierung der Schnittkurvenzweige.

Bei der konkreten Umsetzung eines Verfolgungsalgorithmus in die Praxis treten mehrere Probleme auf, für deren Lösung sehr unterschiedliche Ansätze existieren:

- Die Bestimmung der Startpunkte: Es muß eine sichere Methode gefunden werden, mit der für *jeden* Zweig der Schnittkurve *genau ein* Startpunkt bestimmt werden kann.
- Die Bestimmung signifikanter Punkte (siehe unten) und geschlossener Zweige der Schnittkurve, um eine korrekte Verfolgung der Kurve und die Terminierung des Algorithmus zu gewährleisten.
- Die korrekte Wahl der Schrittweite in der Verfolgungsphase, da eine zu groß gewählte Schrittweite zu falschen Verbindungen einzelner Zweige führen kann, eine zu dicht gewählte Schrittweite den Algorithmus ineffizient macht.

Unter dem Begriff *signifikant* (bzw. *kritisch*) werden folgende Punkte der Schnittkurve zusammengefaßt [Patrikalakis '93]:

- Singuläre Punkte - Sie besitzen keine bestimmte Tangente, dadurch kann die Richtung für eine weitere Verfolgung der Schnittkurve nicht mehr bestimmt werden.
- Extremalpunkte bezüglich der Ränder des Parametergebiets ("Turning Points") - Sie markieren optimale Punkte zur Unterteilung des Parametergebiets, um geschlossene Zweige der Kurve aufzubrechen.
- Punkte, die auf dem Rand eines der beiden Patches liegen - Sie markieren Start- und Endpunkte von Kurvenzweigen.

3.4.1 Bestimmung von Startpunkten

Methoden, die einzelne Startpunkte bestimmen

Viele Schnittalgorithmen kombinieren Unterteilungsmethoden mit einem Verfolgungsalgorithmus: Im ersten Schritt werden Startpunkte mit Hilfe eines Unterteilungsalgorithmus bestimmt und erst im zweiten Schritt die einzelnen Zweige verfolgt. Solche hybriden Algorithmen haben bessere Chancen, wirklich alle Teile der Schnittkurven zu entdecken. Beispiele sind in [Bürger, Schaback '93], [Barnhill, Kersey '90] und [Koparkar '91] beschrieben.

In [Barnhill et al. '87] wird zur Bestimmung von Startpunkten folgender Algorithmus vorgeschlagen: Das Parametergebiet wird zunächst adaptiv in rechteckige Gebiete unterteilt bis ein bestimmter Feinheitsgrad erreicht ist. Im letzten Schritt werden die rechteckigen Gebiete in Dreiecke unterteilt, um eine stückweise lineare Approximation der Fläche zu erhalten. Auf ähnliche Weise werden einige Isolinien der zweiten Fläche stückweise linear approximiert. Um eine Approximation eines Schnittpunktes zu erhalten, muß jetzt lediglich eine univariante lineare Approximation einer Isolinie mit der bivarianten Approximation der Fläche geschnitten werden. Um einen tatsächlichen Schnittpunkt der beiden Flächen $\boldsymbol{f}(u,v)$ und $\boldsymbol{g}(s,t)$ zu erhalten, wird mit der im vorangegangenen Schritt gewonnenen Approximation als Startpunkt eine Newton-Raphson Iteration des nicht linearen Systems

$$\boldsymbol{f}(u,v) - \boldsymbol{g}(s,t) = 0$$

durchgeführt (u sei o.B.d.A. fest). Konvergiert die Iteration nicht, werden die Approximationen der Fläche und der Isolinie verfeinert und eine neue Approximation des Schnittpunktes berechnet.

Abdel-Malek und Yeh ([Abdel-Malek, Yeh '96], [Abdel-Malek, Yeh '97]) schlagen ein weiteres iteratives Verfahren vor. Ihr Algorithmus basiert auf einer speziellen Formulierung des Problems, die die Schnittgleichung und die Parametergebietsbeschreibungen in einem einzigen Vektor $\mathbf{\Phi}$ vereint. Die Startpunkte werden mit Hilfe einer iterativer Optimierung bzw. mit einer Iteration, deren Iterationsmatrix gerade die Moore-Penrose Pseudo-Inverse der Jacobi-Matrix von $\mathbf{\Phi}$ ist, ermittelt. Das Verfahren ist allerdings lokal und garantiert nicht, daß alle Zweige der Schnittkurve erfaßt werden.
[Müllenheim '91] legt über die Parametergebiete ein Punktgitter, das rekursiv und adaptiv verfeinert wird. Die zu den Parameterwerten gehörenden Punkte der beiden Flächen werden paarweise auf ihren Abstand hin untersucht, optimale Paare selektiert und als Basis für eine weitere Verfeinerung der Gitter herangezogen.

Bestimmung aller Zweige einer Schnittkurve

Ein schwer zu lösendes Problem bei Verfolgungsalgorithmen ist die sichere Bestimmung aller Zweige einer Schnittkurve: Der Verfolgungsalgorithmus braucht zur vollständigen Berechnung des Schnittes jeweils einen Startpunkt pro Zweig und er terminiert nur dann, wenn diese Schnittkurve keine geschlossene Schleife bildet.

Eine bereits im letzten Abschnitt angesprochene Lösung des Problems ist die Bestimmung von Startpunkten mit Hilfe von Unterteilungsalgorithmen. Ein reiner Ansatz dieser Art kann allerdings ebenfalls nicht gewährleisten, daß tatsächlich alle Zweige der Kurve erfaßt werden, da das stark vom Feinheitsgrad der Unterteilung abhängt.

Unter dem Begriff *loop detection algorithms* werden Algorithmen zusammengefaßt, die Startpunkte berechnen, indem sie zunächst alle geschlossenen Zweige und kritische Punkte der Kurve bestimmen und auf Grundlage der gewonnenen Informationen das Parametergebiet unterteilen. Grundlage für die Annahme, daß mit diesem Verfahren tatsächlich Startpunkte für alle Zweige der Schnittkurve berechnet werden können, ist folgende Aussage:

Schneiden sich zwei Flächenstücke nicht in einer geschlossenen Kurve, müssen alle Schnittkurven den Rand mindestens eines Patches schneiden. ([Sinah et al. '85])

D.h. unterteilt man die Parametergebiete so, daß nur Teilpatches entstehen deren Schnittkurven keine geschlossenen Zweige besitzen, lassen sich leicht Startpunkte für alle Zweige der Schnittkurve berechnen: Man schneidet einfach die Begrenzungskurven der Patches mit dem jeweils anderen Patch.

Sinha et al. und Sederberg et al. entwickelten in drei Artikeln ein sehr einfaches Kriterium für die Bestimmung geschlossener Zweige der Schnittkurve, das auf der Annahme beruht, daß ein

geschlossener Zweig nur dann entstehen kann, wenn beide Flächenpatches mindestens ein Paar paralleler bzw. kollinearer Normalen besitzt:

Zwei C^1-Flächen schneiden sich in einer geschlossenen Kurve, wenn

- es einen Normalenvektor auf der einen Fläche gibt, der senkrecht zum einem Tangentialvektor der anderen Fläche ist. ([Sederberg, Meyers '88])
- es einen Normalenvektor auf einer der Flächen gibt, der parallel zu einem Normalenvektor der anderen Fläche ist. (Parallel Normal Criterion, [Sinah et al. '85])
- kein Innenprodukt zweier Normalenvektoren der einen und der anderen Fläche Null wird, d.h. wenn eine Linie existiert, die beide Flächen senkrecht schneidet. (Collinear Normal Criterion, [Sederberg et al. '89])

Während die ersten beiden Kriterien relativ aufwendig zu berechnen sind, wird in [Sederberg et al. '89] ein einfaches und schnelles Verfahren vorgestellt, das auf der Auswertung der Normalengleichungen mit Intervallarithmetik basiert.

[Kriezis et al. '92] beschreiben die Schnittkurve zweier B-Spline Flächenstücke $\boldsymbol{q}(s,t)$ und $\boldsymbol{r}(u,v)$ als Nullstellenmenge der orientierten Abstandsfunktion

$$\phi(u,v) = \boldsymbol{n} \cdot (\boldsymbol{r}(u,v) - \boldsymbol{Q}(\boldsymbol{r}(u,v)))$$

wobei $\boldsymbol{Q}(\boldsymbol{r}(u,v))$ die Orthogonalprojektion des Punktes $\boldsymbol{r}(u,v)$ auf die Fläche $\boldsymbol{q}(s,t)$ bezeichnet und $\boldsymbol{n}$ der Normalenvektor von $\boldsymbol{q}(s,t)$ in $\boldsymbol{Q}(\boldsymbol{r}(u,v))$.

Geschlossene Teilkurven entsprechen einer geschlossenen Niveaulinie von ϕ im Parametergebiet, z.B. $\phi(u,v) = c = 0$. Ist das von einer solchen Kurve eingeschlossene Gebiet einfach zusammenhängend, enthält es ein Extremum von ϕ und somit einen kritischen Punkt, d.h. einen Punkt im u-v-Parametergebiet, in dem $\phi(u,v)$ stationär ist, also $\nabla\phi = \mathbf{0}$ oder $\phi_u = \phi_v = 0$. Diese Punkte geben Auskunft über singuläre Punkte und geschlossene Teilkurven der eigentlichen Schnittkurve. Zur Berechnung der kritischen Punkte wird von den Autoren die Rotationszahl des Vektorfelds $\nabla\phi$ entlang einer geschlossenen Kurve im u-v-Parametergebiet verwendet.

Das Problem ist, daß die oben genannte Bedingung nur notwendig, aber nicht hinreichend für die Bestimmung geschlossener Teilkurven ist. Durch das Verfahren ist nicht gewährleistet, daß das Parametergebiet korrekt aufgeteilt wird. In [Patrikalakis '93] weist Patrikalakis darauf hin, daß die Methode, die in [Sederberg, Meyers '88] beschriebene und in [Sederberg et al. '89] verbesserte Methode sicherer ist.

Ein ähnliches, aber einfacheres Verfahren wird in [Hohmeyer '91] vorgestellt. Das auf die Eigenschaften der Gaußabbildung einer Fläche aufbaut und folgenden Satz als Grundlage besitzt
Seien $\boldsymbol{s}_1$ und $\boldsymbol{s}_2$ zwei C^1 Flächen, deren Gaußabbildungen in $N_1 \subset \mathbb{R}^3$ bzw. $N_2 \subset R^3$ enthalten seien. Existieren Vektoren $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{R}^3$, so daß

$$\begin{array}{ll} \boldsymbol{v}_1 \cdot \boldsymbol{n}_1 > 0 & \boldsymbol{v}_1 \cdot \boldsymbol{n}_2 < 0 \\ \boldsymbol{v}_2 \cdot \boldsymbol{n}_1 > 0 & \boldsymbol{v}_2 \cdot \boldsymbol{n}_2 > 0 \end{array}$$

für alle $\boldsymbol{n}_1 \in N_1$ und $\boldsymbol{n}_2 \in N_2$, dann ist die Schnittkurve eine Kurve, ein isolierter Punkt oder eine Menge von Kurven und isolierten Punkten. Weiter gilt, daß alle isolierten Punkte der Schnittkurve auf dem Rand der Flächenstücke liegen, die Schnittkurve keine singulären Punkte besitzt und keine geschlossenen Zweige existieren.

3.4.2 Verfolgung der Schnittkurve

Um ausgehend von einem Startpunkt weitere Punkte der Schnittkurve zu bestimmen, existieren eine Vielzahl von Algorithmen mit differentialgeometrischen und analytischen Ansätzen. Den meisten der hier vorgestellten Algorithmen ist ein Schritt zur Bestimmung von geschlossenen Teilschnittkurven vorangegangen, wie er im vorigen Abschnitt besprochen wurde.

Ein geometrisch motiviertes Verfahren wird von Barnhill et al. in [Barnhill et al. '87] und [Barnhill, Kersey '90] vorgestellt: Sie berechnen im Startpunkt die Tangente der Schnittkurve, indem sie die Schnittgerade der beiden Tangentialebenen berechnen und bestimmen in Richtung dieser Geraden eine Approximation für einen weiteren Schnittpunkt, die mit Hilfe einer Newton-Raphson Iteration verfeinert wird.

Ebenfalls geometrisch konstruieren Wu et al. ([Wu, Andrade '99]) und Stoyanov ([Stoyanov '92]) einen Folgepunkt. Eine erste Näherung des nächsten Punktes der Schnittkurve wird als Punkt auf einer geometrisch konstruierten Approximation des Schmiegkreises des vorangegangenen Kurvenpunktes berechnet. Stoyanov berechnet dagegen eine parabolische Approximation des nächsten Kurvenpunktes.

Mit Hilfe einer Taylorentwicklung der Schnittgleichung um den Startpunkt wird in [Montaudouin et al. '86] und [Bajaj et al. '88] ein weiterer Schnittpunkt der Kurve berechnet.

[Hohmeyer '91] stellt mit Hilfe der im Loop-Detection-Schritt berechneten Vektoren $\boldsymbol{v}_1$ und $\boldsymbol{v}_2$ ein Gleichungssystem auf, das mit einer einfachen Newton-Iteration, die die vorangegangene Lösung als Startpunkt besitzt, gelöst wird und so Punkt für Punkt der Schnittkurve berechnet.

[Markot, Magedson '91] präsentieren einen Algorithmus, der eine bereits existierende stückweise lineare Approximation mit Hilfe einer Newton-Raphson-Iteration weiter verfeinert. Zusätzlich wird beschrieben, wie für in den bereits berechneten sicheren Kurvenpunkten Ableitungen erster und zweiter Ordnung berechnet werden, die dann für eine stückweise Approximation dritter Ordnung der Schnittkurve verwendet werden.

Neuere Algorithmen benutzen in der Verfolgungsphase oft *Homotopie- oder Einbettungsmethoden*, die ausgehend von einem Startpunkt mit Hilfe von Differentialgleichungssystemen weitere Schnittpunkte berechnen.

In [Kriezis et al. '92] wird das Schnittproblem als Anfangswertproblem einer gewöhnlichen Differentialgleichung formuliert: Die Schnittkurve kann sowohl als Nullstellenmenge der im letzten

Abschnitt vorgestellten orientierten Abstandsfunktion ϕ beschrieben werden, als auch als diejenige Flächenkurve, die mit der Orthogonalprojektion auf die andere Fläche identisch ist. Aufgrund dieser beiden Beschreibungen wird ein System von Tensorproduktdifferentialgleichungen aufgestellt, deren Lösung ausgehend von einem Anfangswert numerisch verfolgt wird. (Der Autor benutzte für die Integration die Extrapolationsmethode von Adams.)
Schramm ([Schramm '95]) wendet ein Intervall-Prädiktor-Korrektor-Verfahren zur Lösung der Schnittgleichung

$$\boldsymbol{F}(u,v,s,t) := \boldsymbol{f}(u,v) - \boldsymbol{g}(s,t) = 0$$

auf das System

$$\frac{d}{d\alpha}\boldsymbol{F}(u(\alpha),v(\alpha),s(\alpha),t(\alpha)) = \frac{d}{d(u,v,s,t)}\boldsymbol{F}\left(\frac{d}{d\alpha}u, \frac{d}{d\alpha}v, \frac{d}{d\alpha}s, \frac{d}{d\alpha}t\right)$$

an, während [Abdel-Malek, Yeh '96] ein ebensolches Verfahren auf ein spezielles Gleichungssystem anwendet, das auf der bereits im letzten Abschnitt im Zusammenhang mit einem Loop-Detection-Algorithmus vorgestellten Moore-Penrose-Pseudoinversen basiert.
Nach Bestimmung der Schnittopologie, speziell der Wendepunkte und der Anfangs- und Endpunkte einzelner Zweige mit Hilfe von Differentialgleichungssystemen, verfolgen [Grandine, Klein IV '97] die Schnittkurve indem sie ein Spline-Kollokations-Verfahren auf ein spezielles Zwei-Punkte-Randwertproblem für algebraische Gleichungen vom Index zwei anwenden.

3.5 Unterteilungsalgorithmen

Unterteilungsalgorithmen basieren auf dem Prinzip *Teile und Herrsche*: Das Problem wird in kleinere Teilprobleme aufgeteilt, die leichter zu lösen sind. Die Ergebnisse der Teilprobleme werden später zur Lösung des gesamten Problems zusammengefügt. Zur Lösung des Schnittproblems setzt sich ein solcher Algorithmus wie folgt zusammen:

- *Teile*: Die beiden Flächenstücke werden einem groben Schnittest unterzogen. Fällt dieser negativ aus, schneiden sich die Flächen nicht, sonst wird weiter unterteilt bis ein bestimmtes Abbruchkriterium erfüllt ist. Ergebnis dieses Schrittes ist eine Menge von Teilparametergebieten der beiden Patches, die die Parameterwerte der Schnittkurve einschließen. Diese Teilergebnisse werden eventuell durch einen *Nachbearbeitungsschritt* verfeinert.
- *Herrsche*: Sortieren der Teilergebnisse in aufeinanderfolgende Teilstücke der Schnittkurve.

Eine zentrale Stellung nimmt im ersten Teil des Algorithmus der Schnittest ein.

Im Allgemeinen werden dazu die zu schneidenden Flächenstücke in einfache Körper eingeschlossen, die sich wesentlich leichter einem Schnittest unterziehen lassen, als die eigentlichen Flächenstücke.

Klassische **Einschließungskörper** sind achsenparallele Quader (siehe [Gleicher, Kass '92], [de Figueiredo '96], [Filip et al. '86], [Gopalsamy et al. '91], [Chang et al. '94]), die sich zwar

leicht berechnen und schneiden lassen, aber die Fläche nur sehr ungenau approximieren. Orientierte Quader oder Parallelepipede sind aufwendiger zu berechnen und schwerer zu schneiden, passen sich aber der Form und der Lage des Flächenstücks sehr viel besser an (siehe [Huber, Barth '99], [Bürger, Schaback '93]). Die Anzahl der nötigen Unterteilungen sinkt dadurch drastisch.

Für polynomiale und rationale Bézier- und Spline-Flächen können mit Hilfe der Konvexe-Hülle-Eigenschaft der Kontrollpunkte relativ einfach (homogene) Hüllkörper berechnet werden: siehe z.B. [Bürger, Schaback '93] und [Yamada, Yamaguchi '96] für homogene Hüllkörper. [Goldman, DeRose '86] stellen eine Methode vor mit der auch Hüllkörper für rationale Flächen berechnet werden können, die die Konvexe-Hülle-Eigenschaft nicht erfüllen. Eine spezielle Stellung nehmen die – ebenfalls auf polynomiale Flächen beschränkten – Tchebycheff-Boxen ein [Fournier, Buchanan '94], die aus der Entwicklung der Parameterdarstellung nach Tchebycheff Polynomen gewonnen werden. Auf Basis dieser Darstellung können einschließende Kugeln, Parallelepipede und achsenparallele Einschließungsquader berechnet werden. [Gopalsamy et al. '91] berechnen achsenparallele Boxen für polynomiale Flächen mit Hilfe diskreter Abtastpunkte der Fläche und einer oberen Schranke der dazu gehörenden Lagrangeschen Interpolationspolynomen. Huber beschreibt in [Huber '99] die Berechnung von Parallelepipeden auf Basis des Mittelwertsatzes für vektorwertige Funktionen, während in [Filip et al. '86] achsenparallele Boxen für Dreiecks- und Tensorproduktflächen mit Hilfe der zweiten Ableitung berechnet werden.

Für eine genauere Diskussion der verschiedenen Einschließungskörper, deren Berechnung und die damit verbundenen Schnittalgorithmen sei an dieser Stelle auf Abschnitt 5.7 verwiesen.

Die **Reduzierung des Parametergebiets** auf relevante Abschnitte ist eine weitere Strategie, um die Anzahl der zu untersuchenden Teilparametergebiete zu reduzieren. In [Huber '99] wird eine Methode eingeführt, die nach jedem positiv verlaufenem Schnittest die entsprechenden Parametergebiete auf relevante Teilgebiete zurechtschneidet. [Manocha, Krishnan '97] schlagen eine algebraische Methode vor, um die Parametergebiete bei einem Schnittalgorithmus für eine Kurve und eine Fläche das Parametergebiet zu reduzieren.

Auch bei der **Art der Unterteilung** werden je nach Algorithmus verschiedene Strategien vorgeschlagen: Unterteilung des Parametergebiets in der Mitte, in vier gleiche Teile ([Koparkar '91],[Gleicher, Kass '92],[de Figueiredo '96]), abwechselnd [Huber, Barth '99] in der Mitte der beiden Hauptrichtungen des Parametergebiets oder je nach Form des Patches entlang der einen oder anderen Hauptrichtung [Koparkar '91].

Einen weiteren wichtigen Stellenwert nimmt das **Abbruchkriterium** für die Unterteilung ein, dessen Wahl davon abhängt, ob der Unterteilung noch ein weiterer Schritt folgt oder nicht:
In einem reinen Unterteilungsalgorithmus ist die Größe der berechneten Teilpatches ausschlaggebend für die Genauigkeit der Ergebnisse. So wird z.B. in [Huber '99] der Durchmesser des Schnittes der beiden einschließenden Parallelepipede als Abbruchkriterium vorgeschlagen und mit dem Abbruchkriterium "Größe des Teilpatches" verglichen (siehe auch [Huber '98]).

Die meisten Unterteilungsalgorithmen verfeinern die **Ergebnisse** in einem weiteren Schritt.
Im einfachsten Fall werden die berechneten Teilpatches linear approximiert und die Schnittgeraden dieser Approximationen berechnet (siehe z.B. [Filip et al. '86],[Houghton et al. '85]). Als Abbruchkriterium für die Unterteilung gilt in diesem Fall eine geringe Krümmung, um einen möglichst kleinen Approximationsfehler bei der Linearisierung zu erzielen.
In [Bürger, Schaback '93] wird ein Unterteilungsalgorithmus mit einem Verfolgungsalgorithmus verbunden: Sind die Flächenstücke klein genug, d.h. treten keine Degenerationen der Schnittkurve und keine geschlossenen Zweige der Schnittkurve mehr auf, werden mit einem Verfolgungsalgorithmus Näherungen für Schnittpunkte berechnet.

Um die **Geschwindigkeit** zu erhöhen, werden in [Chang et al. '94] ein parallelisierter Unterteilungsalgorithmus und in [Bürger, Schaback '93] ein paralleler hybrider Algorithmus vorgeschlagen.

3.6 Andere Methoden

[Sederberg, Nishita '91] schlagen in ihrem Artikel einen Algorithmus vor, der direkt eine Hermite-Approximation der Schnittkurve liefert. Der Vorteil liegt auf der Hand: man erhält sofort eine komplette Parametrisierung der Schnittkurve. Allerdings muß auch hier gewährleistet sein, daß die Schnittkurve keine geschlossenen Zweige besitzt.

3.7 Schnittalgorithmen mit verifizierten Ergebnissen

Nicht bei allen oben genannten Algorithmen kann mit Sicherheit gesagt werden, daß sämtliche Teile der Schnittkurve gefunden und richtig erfaßt wurden: Werden die Einschließungskörper bei Unterteilungsalgorithmen nur angenähert ([Houghton et al. '85]) oder nur Linearisierungen der Patches auf Schnitte untersucht ([Barnhill et al. '87]), können Teile der Schnittkurve übersehen werden. Das Gleiche gilt für Algorithmen, die lediglich mit Gleitpunktarithmetik arbeiten, wie das beispielsweise bei fast allen Verfolgungsalgorithmen der Fall ist: Aufgrund von Rundungs- und Approximationsfehlern kann nie eine sichere Aussage bezüglich der Schnittkurve getroffen werden.

Es existieren nur wenige Arbeiten, die das Problem der vollständigen und richtigen Erfassung des Schnittes mit Hilfe von Intervall- bzw. affiner Arithmetik (siehe Anhang A) zu lösen versuchen. Dazu gehören die Arbeiten von [Gleicher, Kass '92] und [de Figueiredo '96], die mit Hilfe von Intervall- bzw. affiner Arithmetik achsenparallele Einschließungsquader berechnen die für den Schnittest in einem Unterteilungsalgorithmus verwendet werden. [Huber, Barth '99] verwenden Intervallarithmetik zur Berechnung einschließender Parallelepipede.

Ein weiteres sicheres Verfahren ist das in [Schramm '95] entwickelte Intervall-Prädiktor-Korrektor Verfahren. In [Sederberg et al. '89] wird Intervallrechnung zur sicheren Bestimmung

aller Zweige der Schnittkurve verwendet.

Ein Verfolgungsalgorithmus für den Schnitt zweier Intervallflächenstücke wird in [Hu et al. '97] entwickelt. Wandelt man zwei gewöhnliche Flächenstücke in Intervallflächen um, kann man mit diesem Algorithmus eine Intervalleinschließung der Schnittkurve berechnen.

[Keyser et al. '96] verwenden exakte Arithmetik um Trimmkurven bei der Umwandlung von Solid- in B-Rep-Modelle zu berechnen.

3.8 Vergleich bestehender Methoden

Unter den in diesem Kapitel vorgestellten Methoden haben sich Unterteilungs- und Verfolgungsmethoden als praktikable Lösungen für das Problem herausgestellt. Konstruktive Methoden sind nur in Spezialfällen anwendbar, Diskretisierungsmethoden zu ungenau und algebraische Methoden zu aufwendig.

Vergleicht man die beschriebenen Verfolgungs- und Unterteilungsalgorithmen und Algorithmen, die beide Verfahren kombinieren, bezüglich der am Beginn des Kapitels beschriebenen Kriterien Genauigkeit, Zuverlässigkeit, Schnelligkeit, Selbstständigkeit und Allgemeingültigkeit, so schneiden die hybriden Algorithmen am besten ab.

Reine Unterteilungsalgorithmen können beliebig genaue Ergebnisse liefern und arbeiten auch bei schlecht konditionierten Problemen zuverlässig, unabhängig und je nach Art der verwendeten Hüllkörper auch allgemeingültig. Ein Problem stellt allerdings, bei wachsender Genauigkeit, die hohe Rechenzeit dar.

Verfolgungsalgorithmen sind bei gleicher Genauigkeit einerseits schneller als Unterteilungsalgorithmen, andererseits aber auch wesentlich instabiler, da sie bei der Bestimmung geschlossener Teilkurven und der eigentlichen Verfolgung der Kurve auf bestimmte Differenzierbarkeitsbedingungen der beiden Flächen angewiesen sind.
So ist beispielsweise der in [Grandine, Klein IV '97] vorgestellte Algorithmus auf "Flächen, für die grundsätzlich die Möglichkeit besteht, auf robuste Art und Weise die auftretenden nichtlinearen (Differential-)Gleichungssysteme zu lösen.". Konkret umgesetzt wurde der Algorithmus für Tensorprodukt-Spline-Flächen, für die diese Voraussetzung erfüllt ist. Andere Verfolgungsalgorithmen beschränken sich auf reguläre parametrische Flächen ([Barnhill, Kersey '90], [Abdel-Malek, Yeh '96], [Wu, Andrade '99]) oder Tensorprodukt-NURBS-Flächen [Kriezis et al. '92]. Die von Sederberg et al. vorgeschlagenen loop detection Algorithmen [Sederberg, Meyers '88] und [Sederberg et al. '89] sind auf Bézier-Tensorprodukt bzw. C^1-Flächen beschränkt. Der Fall, daß sich zwei Flächenpatches überlappen können, wird bei fast alle Verfolgungsalgorithmen ausgeschlossen bzw. bedarf einer Sonderbehandlung.

Nach Bürger/Schaback [Bürger, Schaback '93] sind **hybride Algorithmen**, d.h. ein divide-and-conquer-Ansatz kombiniert mit einem Verfolgungsalgorithmus, die besseren Schnittalgorithmen,

da sie die Robustheit der Unterteilungsalgorithmen mit der Genauigkeit und Geschwindigkeit der Verfolgungsalgorithmen kombinieren. Teilprobleme, die vom Verfolgungsalgorithmus nicht sauber gelöst werden können, werden einfach weiter unterteilt.

4

Das Konzept des allgemeinen Unterteilungsalgorithmus und Ansätze für mögliche Verbesserungen

Dieses Kapitel soll einen Überblick über einige grundlegende Überlegungen geben: Im ersten Abschnitt werden die Gemeinsamkeiten der in Kapitel 3 Abschnitt 3.5 vorgestellten Unterteilungsalgorithmen zusammengefaßt und die grundsätzlichen Vorteile des Konzeptes erörtert, die die Motivation zur Verbesserung bestehender Algorithmen darstellen.
In einem weiteren Abschnitt werden die Nachteile existierender Algorithmen und mögliche Verbesserungen diskutiert.

4.1 Der Allgemeine Unterteilungsalgorithmus

Die im letzten Kapitel vorgestellten Unterteilungsalgorithmen unterscheiden sich zwar in der Lösung einzelner Teilprobleme, wie Art und Berechnung der Einschließungskörper, Schnitt derselben und Unterteilungsmethoden, arbeiten aber alle nach dem gleichen Konzept:

Input: Zwei Flächenpatches A und B.
Output: Eine Liste mit den Teilparametergebieten beider Flächen, die sicher alle Parameterwerte enthalten, die den Schnittpunkten der Flächen entsprechen.

Schnitt(Patch A, Patch B) {
  Berechne Hüllkörper $V(A)$ und $V(B)$
  Falls (Schnittest von $V(A)$ und $V(B)$ negativ)
    Abbrechen;
  Reduzierung des Parametergebietes;
  Falls (Abbruchkriterium erfüllt ist) {
    Schreibe Ergebnisse;

```
    Abbrechen;
  }
  Unterteile A in m Teilpatches A_i, i=1,...,m
  Unterteile B in n Teilpatches B_j, j=1,...,n
  Für i = 1, ..., m
    Für j = 1, ..., n
      Schnitt(A_i,B_j);
}
```

Bemerkung:
Die Reduzierung des Parametergebietes ist nicht Teil aller Unterteilungsalgorithmen, er wird aber in diese Arbeit mit einbezogen, da sich mit diesem Zusatzschritt die Ergebnisse oft stark verbessern lassen.

Unterteilungsalgorithmen sind im Sinne der in Kapitel 3 genannten Qualitätskriterien

- **Genau**: Ergebnisse können je nach Schärfe des Abbruchkriteriums beliebig genau berechnet werden.
- **Zuverlässig**: Unter der Voraussetzung, daß zuverlässige Einschließungskörper und Schnittests verwendet werden, werden grundsätzlich im Rahmen der festgelegten Genauigkeit alle Zweige der Schnittkurve bestimmt und auch Überlappungen korrekt festgestellt.
- **Selbständig**: Die Berechnung des Schnittes funktioniert ohne interaktiven Eingriff des Benutzers auch in Sonderfällen.
- **Allgemeingültig**: Der Algorithmus ist grundsätzlich für alle Arten parametrischer Flächen geeignet.

Obige Kriterien können je nach Implementierung mehr oder weniger erfüllt sein. So ist bei Unterteilungsalgorithmen, die

- im letzten Schritt eine Linearisierung durchführen (siehe z.B. [Filip et al. '86])
- unsichere Einschließungskörper verwenden (siehe z.B. [Houghton et al. '85])
- Schnittests/-berechnungen mit unsicherer Arithmetik durchführen

nicht mehr gewährleistet, daß tatsächlich alle Teile der Schnittkurve bestimmt werden. Die Allgemeingültigkeit des Algorithmus ist abhängig von der Art der Berechnung der verwendeten Hüllkörper: Hüllkörper, die beispielsweise auf Basis von Kontrollpunkten berechnet werden, sind nur für rationale Flächen geeignet.([Yamada, Yamaguchi '96],[Goldman, DeRose '86])

Hauptkritikpunkt an allen bisher veröffentlichten Unterteilungsalgorithmen ist die mangelnde Geschwindigkeit:
Algorithmen, die die einfach zu berechnenden achsenparallelen Boxen als Hüllkörper verwenden, zeichnen sich durch eine hohe Anzahl an Unterteilungen aus, während Algorithmen mit komplizierteren Hüllkörpern wiederum mehr Zeit für Berechnung und Schnitt derselben benötigen.

4.2 Mögliche Ansätze zur Verbesserung

Möchte man bestehende Unterteilungsalgorithmen verbessern, stehen zwei Punkte im Zentrum der Betrachtungen: **Die Gewährleistung der Zuverlässigkeit und die Erhöhung der Geschwindigkeit der Berechnungen**.

Um die Zuverlässigkeit des Algorithmus zu garantieren, sollten die Berechnung der Hüllkörper, der Schnittest und eventuelle Verfeinerungsschritte so durchgeführt werden, daß die Ergebnisse exakt sind, bzw. sichere Einschließungen der Ergebnisse liefern.

Die Kosten für die Berechnung eines orientierten Hüllkörpers, der die Fläche eng umschließt, muß gegen die dadurch eingesparte Anzahl der Unterteilungen aufgewogen werden: Nur wenn der Aufwand für Berechnung und Schnitt der neuen Hüllkörper im Vergleich zu einfacheren Hüllkörpern nicht zu groß ist, wird sich der Algorithmus signifikant verbessern.

Ähnliches gilt für die Reduzierung des Parametergebietes: auch hier müssen Aufwand und Nutzen verglichen werden. Eine effektive Parametergebietreduzierung kann die Anzahl der zu unterscheidenden Teilparametergebiete und damit die Rechenzeit erheblich reduzieren.

Das Abbruchkriterium sollte eine klare Aussage über die Qualität der Berechnung machen und einfach zu berechnen sein, da es in jedem Schritt überprüft werden muß.

Ein gutes Unterteilungsschema muß aus dem gleichen Grund einfach gehalten sein und die Unterteilung selbst muß sicher sein, d.h. es muß eventuell so gerundet werden, daß sich die beiden Teilstücke am Rand überlappen.

Es ist klar, daß die Art der Hüllkörper und die Methode für den Schnittest in direktem Zusammenhang stehen und es wäre durchaus sinnvoll, auch das Abbruchkriterium direkt mit bestimmten Eigenschaften der beiden Hüllkörper zu verbinden.

Die Anforderungen an die einzelnen Teilschritte des Algorithmus lassen sich folgendermaßen zusammenfassen:

- **Hüllkörper**: einfach zu berechnen, zuverlässig, möglichst eng einschließend.
- **Schnittest für Hüllkörper**: schnell, zuverlässig.
- **Abschneiden der Parametergebiete**: effektiv bezüglich der Reduzierung, schnell, zuverlässig.
- **Abbruchkriterium**: einfach zu berechnen, sinnvoll.
- **Art der Unterteilung**: einfach, dem Algorithmus angepaßt, zuverlässig.

4.3 Lineare Intervallabschätzungen als zentrales Element im neuen Algorithmus

Ursprünglicher Ansatz für den hier neu entwickelten Algorithmus war die Idee, die in [Huber, Barth '99] als Hüllkörper verwendeten Parallelepipede durch den Einbezug der zweiten Ableitung in die Fehlerabschätzung weiter zu verbessern, um dadurch engere Einschließungen zu bekommen. Es zeigte sich, daß die Einschließungen mit dieser Methode tatsächlich enger werden, das Problem einen effizienten Schnittest zu erhalten, wurde damit allerdings nicht gelöst.

Erst die Einführung von Linearen Intervallabschätzungen als Hüllkörper hat zum gewünschten Ergebnis geführt:

- Sie sind schneller zu berechnen als entsprechende Parallelepipede, da keine Eckpunkte und Kanten berechnet werden müssen (Siehe Kapitel 5).
- Ihre Struktur erlaubt einen einfachen Schnittest, der im allgemeinen als Beiprodukt das Parametergebiet reduziert. D.h. Schnittest und Reduzierung des Parametergebietes sind ein einziger Schritt. (Siehe Kapitel 6)
- Das Endergebnis sind einschließende achsenparallele Parameterteilgebiete und/oder eng einschließende Intervallinien im Parametergebiet. (Siehe Kapitel 6)
- Die extrem effiziente Parametergebietreduzierung ermöglicht eine neue Art der Unterteilungsreihenfolge, die die Effizienz des Algorithmus weiter steigert. (Siehe Kapitel 7)
- Ebenfalls aus der Struktur der LIEs ergibt sich fast direkt ein einfaches Abbruchkriterium, das sich an der "Flachheit" der Fläche orientiert.(Siehe Kapitel 7)

In den folgenden Kapiteln, werden alle oben genannten neuen Elemente eingeführt.

Der neue Algorithmus hat später etwa folgende Form:

Input: Ein Paar AB bestehend aus den Flächenpatches A und B.
Output: Je eine Liste mit Teilparametergebieten und Streifen in den Parametergebieten, die sicher alle Parameterwerte der Schnittpunkte enthalten.

Schneide(Flächenpaar AB) {
  Berechne LIEs L(A) und L(B)
  Schnittest und Parametergebietreduzierung, $AB \rightarrow \tilde{A}\tilde{B}$;
  Falls (Schnittest negativ)
    Abbruch;
  Falls (Abbruchkriterium erfüllt) {
    Berechne LIE s $L(\tilde{A})$ und $L(\tilde{B})$;
    Schreibe Ergebnisse;
    Abbruch;
  }

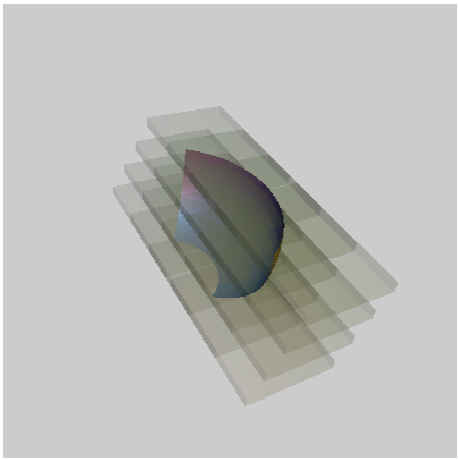
Adaptive Unterteilung der Parametergebiete;
Rekursive Aufrufe von **Schneide** für die zwei neuen Flächenpaare,
die aus der Unterteilung hervorgegangen sind;
}

Die endgültige Version des Algorithmus verwendet sowohl LIEs als auch achsenparallele Boxen als Hüllkörper und wird in Kapitel 9 Abschnitt 9.6 beschrieben.

5

Die Einschließungskörper: Lineare Intervallabschätzungen

In diesem Kapitel werden Lineare Intervallabschätzungen als Hüllkörper für parametrische Flächen eingeführt. Es werden zwei verschiedene Methoden zur Berechnung von LIEs vorgestellt, die auf der Berechnung eines Taylor Modells [Berz, Hofstätter '98] für das Flächenpatch, bzw. auf der Ausnutzung der speziellen inneren Struktur der affinen Arithmetik [Stolfi, de Figueiredo '97] basieren. (Einen kurzer Überblick über die wichtigsten Definitionen, Sätze und Methoden im Zusammenhang mit Intervallarithmetik, Taylor Modellen und affiner Arithmetik wird in Anhang A gegeben.)



Weiter werden spezielle Eigenschaften von LIEs untersucht und Form und Existenz von LIEs bei Auftreten bestimmter Sonderfälle betrachtet. Am Ende des Kapitels steht sowohl ein Vergleich der vorgestellten Berechnungsmethoden für LIEs, als auch der Vergleich von LIEs mit anderen Hüllkörpern.

5.1 Allgemeine Definition

Definition 5.1.1
Das Intervallebenenstück

$$\mathbb{L}(u,v) = \mathbb{p} + u^* \boldsymbol{y}_1 + v^* \boldsymbol{y}_2, \quad (u^*, v^*) \in I_u^* \times I_v^* = \mathbb{I}^* \in I\mathbb{R}^2 \tag{5.1}$$

mit $\mathbb{p} \in I\mathbb{R}^3$, $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathbb{R}^3$ *heißt lineare Intervallabschätzung (LIE) des parametrischen Flächenpatches*

$$\boldsymbol{f}(u,v),\ (u,v) \in \mathbb{I} \in I\mathbb{R}^2$$

falls eine bijektive und stetig differenzierbare Abbildung

$$\phi : \left\{ \begin{array}{lcl} \mathbb{I} & \rightarrow & \mathbb{I}^* \\ (u,v) & \mapsto & \phi(u,v) := (u^*(u), v^*(v)) \end{array} \right.$$

existiert, so daß für alle $(u,v) \in \mathbb{I}$ *gilt*

$$\boldsymbol{f}(u,v) \in \mathbb{L}(\phi(u,v)) = \mathbb{L}(u^*(u), v^*(v)).$$

Zu jedem Punkt des Flächenpatches existiert also ein Intervallpunkt der linearen Intervallabschätzung, der den Flächenpunkt enthält.

Sind $rad(I_u^*) \neq 0$ und $rad(I_v^*) \neq 0$ kann man durch die zulässige Parametertransformation

$$\phi^{-1} : \left\{ \begin{array}{lcl} \mathbb{I}^* & \rightarrow & \mathbb{I} \\ (u^*, v^*) & \mapsto & (u,v) := \phi^{-1}(u^*, v^*) = \left(\begin{array}{l} \frac{u_2 - u_1}{u_2^* - u_1^*}\ u^* + \frac{u_1 u_2^* - u_2 u_1^*}{u_2^* - u_1^*} \\ \frac{v_2 - v_1}{v_2^* - v_1^*}\ v^* + \frac{v_1 u_2^* - v_2 u_1^*}{v_2^* - v_1^*} \end{array} \right) \end{array} \right.$$

erreichen, daß sich die Parametrisierungen der Fläche und der LIE entsprechen, d.h. es gilt:

$$\boldsymbol{f}(u,v) \in \mathbb{L}(u,v), \text{ für alle } (u,v) \in \mathbb{I} \tag{5.2}$$

5.2 Berechnung linearer Intervallabschätzungen als Taylor-Modell 2.Grades

Definition 5.2.1 ([Berz, Hofstätter '98])
Sei $\boldsymbol{f} : \mathbb{I} \mapsto I\!R^3$ $n+1$*-mal stetig differenzierbar auf* $\mathbb{I} \in I\!I\!R^2$ *und sei* $\boldsymbol{T}_{u_0,v_0}(u,v)$ *das bivariante Taylorpolynom n.-ten Grades von* $\boldsymbol{f}$ *um* (u_0, v_0) *dann heißt*

- *ein Intervallvektor* $\mathbb{J} \in I\!I\!R^2$ *für den gilt*

  $$\forall (u,v) \in \mathbb{I} : \boldsymbol{f}(u,v) - \boldsymbol{T}_{u_0,v_0}(u,v) \in \mathbb{J}$$

  eine Restgliedabschätzung n.-ter Ordnung von $\boldsymbol{f}$ auf $\mathbb{I}$.
- *das Paar* $(\boldsymbol{T}_{u_0,v_0}, \mathbb{J})$ Taylor-Modell n.-ter Ordnung von $\boldsymbol{f}$
- *die Menge aller Restgliedabschätzungen* Restgliedfamilie

Satz 5.2.1
Sei

$$(\boldsymbol{T}_{u_0,v_0}, \mathbb{J})$$

ein Taylor-Modell 1.Ordnung des Flächenpatches $\boldsymbol{f}(u,v), (u,v) \in \mathbb{I}$ *dann ist*

$$\mathbb{L}(u,v) = \boldsymbol{T}_{u_0,v_0}(u,v) + \mathbb{J} \tag{5.3}$$

eine lineare Intervallabschätzung von $\boldsymbol{f}$

Beweis:
Sei $\boldsymbol{T}_{u_0,v_0}$ das Taylorpolynom 1.Grades um $(u_0, v_0) \in \mathbb{I}$ und

$$\boldsymbol{R}(u,v,\xi_u,\xi_v) := \frac{1}{2}((u-u_0)^2\boldsymbol{f}_{uu}(\xi_u,\xi_v)+2\,(u-u_0)(v-v_0)\boldsymbol{f}_{uv}(\xi_u,\xi_v)+(v-v_0)^2\boldsymbol{f}_{vv}(\xi_u,\xi_v)) \quad (5.4)$$

das zugehörige Lagrange Restglied.

Nach dem Satz von Taylor gilt dann: Für alle $(u,v) \in \mathbb{I}$, $(\xi_u,\xi_v) \in [u,u_0] \times [v,v_0]$, so daß gilt

$$\boldsymbol{f}(u,v) = \boldsymbol{T}_{u_0,v_0}(u,v) + \boldsymbol{R}_{u_0,v_0}(u,v,\xi_u,\xi_v)$$

Sei $\mathbb{R}$ eine Wertebereichseinschließung von $\boldsymbol{R}_{u_0,v_0}(u,v,\xi_u,\xi_v)$, dann gilt

$$\boldsymbol{f}(u,v) \in \boldsymbol{T}_{u_0,v_0}(u,v) + \mathbb{R} \quad \text{für alle } (u,v) \in \mathbb{I}.$$

Durch einfache Umformung des Taylor-Modells $(\boldsymbol{T}_{u_0,v_0}, \mathbb{R})$ erhält man die Darstellung

$$\mathbb{L}(u,v) = \mathbb{r} + u\boldsymbol{f}_u(u_0,v_0) + v\boldsymbol{f}_v(u_0,v_0)$$

mit $\mathbb{r} := \mathbb{R} + \boldsymbol{f}(u_0,v_0) - u_0\boldsymbol{f}_u(u_0,v_0) - v_0\boldsymbol{f}_v(u_0,v_0)$ □

Eine (grobe) Intervallabschätzung $\mathbb{J}$ des Langrangen Restgliedes (5.4) in allen vier Variablen erhält man nach Satz 1.1.1 (Einschließungseigenschaft) durch einfache Intervallauswertung der Restgliedformel in allen vier Variablen

$$\mathbb{R} := \boldsymbol{R}(I_u, I_v, I_u, I_v).$$

Bessere Wertebereichsabschätzungen für das Restglied lassen sich mit etwas Mehraufwand beispielsweise mit Hilfe von zentrierten Formen (siehe z.B. [Neumeier '90], *centered forms*) berechnen.

5.3 Berechnung mit Hilfe affiner Arithmetik

Satz 5.3.1
Sei $\boldsymbol{f}(u,v)$, $(u,v) \in \mathbb{I} := I_u \times I_v$ *mit* $rad(I_u), rad(I_v) > 0$ *ein stetiges Flächenstück im* $\mathbb{R}^3$,

$$\hat{u} := u_0 + u_1\epsilon_u \quad \textit{und} \quad \hat{v} := v_0 + v_1\epsilon_v; \qquad \epsilon_u, \epsilon_v \in [-1,1]$$

die zu I_u *und* I_v *gehörenden Affinformen und*

$$\boldsymbol{f}(\hat{u},\hat{v}) = \hat{\boldsymbol{f}}(\epsilon_u,\epsilon_v,\epsilon_1,...,\epsilon_n) = \boldsymbol{f}^0 + \boldsymbol{f}^u\epsilon_u + \boldsymbol{f}^v\epsilon_v + \sum_{i=1}^{n}\boldsymbol{f}^i\epsilon_i; \quad \epsilon_u,\epsilon_v,\epsilon_i \in [-1,1],\ i = 1,...,n$$

die Auswertung von $\boldsymbol{f}$ *mit* $\hat{u}$ *und* $\hat{v}$.

Dann ist

$$\mathbb{L}(\epsilon_u, \epsilon_v) := \boldsymbol{f}^0 + \boldsymbol{f}^u \epsilon_u + \boldsymbol{f}^v \epsilon_v + \mathbb{Q}; \quad \epsilon_u, \epsilon_v \in [-1, 1], \tag{5.5}$$

mit

$$\mathbb{Q} := \left[-\sum_{i=1}^{n} |\boldsymbol{f}^i|, \sum_{i=1}^{n} |\boldsymbol{f}^i| \right]$$

eine lineare Intervallabschätzung von $\boldsymbol{f}$*, wobei* $|\boldsymbol{f}^i| := (|f_1^i|, |f_2^i|, |f_3^i|)^T,\ i = 1, ..., n.$

Beweis:
Aus der Definition der Affinformen und der für sie definierten Operationen (siehe Anhang A Abschnitt 1.2) folgt:

Für alle $(u, v) \in \mathbb{I}$ existieren $\epsilon_u^0, \epsilon_v^0, \epsilon_i^0 \in [-1, 1], i = 1, ..., n$ so, daß

$$\boldsymbol{f}(u, v) = \hat{\boldsymbol{f}}(\epsilon_u^0, \epsilon_v^0, \epsilon_1^0, ..., \epsilon_n^0)$$

Wegen der Einschließungseigenschaft der Intervallrechnung (Anhang A, Satz 1.1.1) gilt damit dann auch:
Für alle $(u, v) \in \mathbb{I}$ existieren $\epsilon_u^0, \epsilon_v^0 \in [-1, 1]$ so, daß

$$\boldsymbol{f}(u, v) \in \hat{\boldsymbol{f}}(\epsilon_u^0, \epsilon_v^0, [-1, 1], ..., [-1, 1]) = \boldsymbol{f}^0 + \boldsymbol{f}^u \epsilon_u + \boldsymbol{f}^v \epsilon_v + \mathbb{Q}$$

Durch Zusammenfassung von $\boldsymbol{f}^0$ und $\mathbb{Q}$ zu $\mathfrak{q} := \boldsymbol{f}^0 + \mathbb{Q}$ erhält man eine Darstellung der Gleichung (5.5) in der Form (5.1):

$$\mathbb{L}(\epsilon_u, \epsilon_v) := \mathfrak{q} + \boldsymbol{f}^u \epsilon_u + \boldsymbol{f}^v \epsilon_v; \quad \epsilon_u, \epsilon_v \in [-1, 1] \tag{5.6}$$

Die Zuordnung zwischen den Intervallen I_u und I_v und den Affinformen $\hat{u}, \hat{v}$ ist eineindeutig, falls I_u und I_v keine Punktintervalle sind. In diesem Fall läßt sich die Beziehung zwischen den Parametern u und v und den Fehlersymbolen ϵ_u und ϵ_v mit Hilfe der bijektiven Abbildungen

$$\phi : \begin{cases} [-1, 1] \times [-1, 1] & \rightarrow & \mathbb{I} \\ (\epsilon_u, \epsilon_v) & \mapsto & (u, v) := \phi(\epsilon_u, \epsilon_v) = (rad(I_u)\ \epsilon_u + mid(I_u),\ rad(I_v)\ \epsilon_v + mid(I_v)) \end{cases}$$

bzw.

$$\phi^{-1} : \begin{cases} \mathbb{I} & \rightarrow & [-1, 1] \times [-1, 1] \\ (u, v) & \mapsto & (\epsilon_u, \epsilon_v) := \phi^{-1}(u, v) = (\frac{1}{rad(I_u)}(u - mid(I_u), \frac{1}{rad(I_v)}(v - mid(I_v)) \end{cases}$$

darstellen. Damit existiert eine bijektive Zuordnung zwischen den Punkten des Flächenstücks und den Intervallpunkten der Intervallebene (5.6).

Durch Umparametrisierung von $\mathbb{L}$ mit ϕ erhält man eine Darstellung der LIE in der Form (5.2):

$$\mathbb{L}(u, v) = \mathfrak{q} + \boldsymbol{f}^u u + \boldsymbol{f}^v v; \quad (u, v) \in \mathbb{I}$$

□

5.4 Eigenschaften Linearer Intervallabschätzungen

5.4.1 Existenz und Zuverlässigkeit

Man kann eine LIE mit der Taylor-Modell-Methode genau dann berechnen, wenn die Voraussetzungen des Satzes von Taylor erfüllt sind, d.h. wenn das Flächenstück über seinem Parametergebiet zweimal stetig differenzierbar ist. Für die Methoden der affinen Arithmetik, wird lediglich die Stetigkeit des Patches vorausgesetzt.

Gelten die oben genannten Voraussetzungen, garantiert die Verwendung der Intervallarithmetik mit gerichteter Rundung, bzw. affine Arithmetik, daß das Flächenpatch tatsächlich innerhalb der berechneten LIE liegt.

5.4.2 Lineare Präzision

Satz 5.4.1
Lineare Intervallabschätzungen eines Ebenenstücks, die als Taylor-Modell oder mit affiner Arithmetik berechnet wurden, sind unter Ausschluß von Rundungsfehlern mit dem Ebenenstück identisch.

Beweis:
Sei $\boldsymbol{f}$ eine parametrisches Ebenenstück im $\mathbb{E}^3$

$$\boldsymbol{f}(s,t) = \boldsymbol{p} + s\boldsymbol{v}_1 + s\boldsymbol{v}_2; \qquad (s,t) \in \mathbb{I}$$

Berechnet man eine LIE von $\boldsymbol{f}$ mit Hilfe eines Taylor-Modells um $(s_0,t_0) \in \mathbb{I}$, so erhält man aufgrund der Tatsache, daß alle 2. Ableitungen verschwinden, das folgende Ergebnis:

$$\begin{aligned} \mathbb{L}_{\text{Taylor}}(s,t) &= \boldsymbol{f}(s_0,t_0) + (s-s_0)\boldsymbol{v}_1 + (t-t_0)\boldsymbol{v}_2 \\ &= \boldsymbol{p} + s\boldsymbol{v}_1 + s\boldsymbol{v}_2; \qquad (s,t) \in \mathbb{I} \end{aligned}$$

Es gilt also $\mathbb{L}_{\text{Taylor}}(s,t) \equiv \boldsymbol{f}(s,t)$.

Analoges gilt für die Auswertung von $\boldsymbol{f}$ mit Affinformen: Da nur affine Operationen vorkommen, treten keine Approximationsfehler auf. □

5.4.3 Konvergenz

Satz 5.4.2
Sei $\mathbb{I}_1 \supset \mathbb{I}_2 \supset \ldots$ eine Folge von Teilparametergebieten mit punktförmigem Grenzwert $\mathbb{I} = ([u_0,u_0],[v_0,v_0]) = (u_0,v_0)$. Sei

$$\mathbb{L}_k(u,v) = \mathbb{p} + u\,\boldsymbol{y}_1 + v\,\boldsymbol{y}_2, \quad (u,v) \in I_u \times I_v = \mathbb{I}_k \subset \mathbb{R}^2$$

eine lineare Intervallabschätzung des Flächenstücks $\boldsymbol{f}(u,v)$ $(u,v) \in \mathbb{I}_k$ die mit der Taylor-Modell Methode oder mit affiner Arithmetik berechnet wurde. Dann gilt

$$\mathbb{L}_k(u,v),\ (u,v) \in \mathbb{I}_k \quad \xrightarrow{k\to\infty} \quad \boldsymbol{f}(u_0,v_0)$$

Beweis:
Für Taylor-Modell basierte LIEs folgt die Behauptung aus der Voraussetzung an das Flächenpatch: Aus der Kompaktheit des Definitionsbereichs und der zweifachen stetige Differenzierbarkeit von $\boldsymbol{f}$ auf $\mathbb{I}$ folgt die Beschränktheit der ersten und zweiten Ableitungen auf $\mathbb{I}$ und damit die Konvergenz.

Für LIEs, die auf der Berechnung mit affiner Arithmetik basieren gilt:
Sei $\mathbb{I}_k = I_u^k \times I_v^k$. Weiter seien $\hat{u}_k = u_0^k + \epsilon_u^k u_1^k,\ \epsilon_u^k \in [-1,1]$ und $\hat{v}_k = v_0^k + \epsilon_v^k v_1^k,\ \epsilon_v^k \in [-1,1]$ die zu I_u^k bzw. I_v^k gehörenden Affinformen. Dann gilt

$$\hat{u}_k \xrightarrow{k\to\infty} \hat{u} = u_0^\infty = u_0 \qquad \text{und} \qquad \hat{v}_k \xrightarrow{k\to\infty} \hat{v} = v_0^\infty = v_0$$

Daraus folgt

$$\hat{\boldsymbol{f}}_k = \boldsymbol{f}(\hat{u}_k, \hat{v}_k) \xrightarrow{k\to\infty} \boldsymbol{f}(u_0, v_0)$$

□

5.5 LIEs von Flächen mit Singularitäten, Selbstdurchdringungen und/oder starker Krümmung

5.5.1 Der Einfluß von Singularitäten

Eine LIE $\mathbb{L}(u,v) = \mathbb{p} + u\,\boldsymbol{y}_1 + v\,\boldsymbol{y}_2, \quad (u,v) \in I_u \times I_v = \mathbb{I} \subset \mathbb{R}^2$ ist nicht degeneriert, wenn

1. $\boldsymbol{n} = \boldsymbol{y}_1 \times \boldsymbol{y}_2 \neq 0$
2. $rad(\mathbb{p}) < \infty$

Der erste Punkt ist für eine LIE, die als Taylor-Modell berechnet wurde, genau dann erfüllt, wenn $\boldsymbol{f}_u \times \boldsymbol{f}_v \neq \boldsymbol{0}$ ist, d.h. wenn die Parametrisierung der Fläche im Entwicklungspunkt regulär ist. Die Erfüllung des zweiten Punktes hängt von der Art der Berechnung der Intervalleinschließung des Restgliedes ab. Eine beschränkte Intervalleinschließung existiert, wenn die zweiten Ableitungen über dem Parametergebiet beschränkt sind.

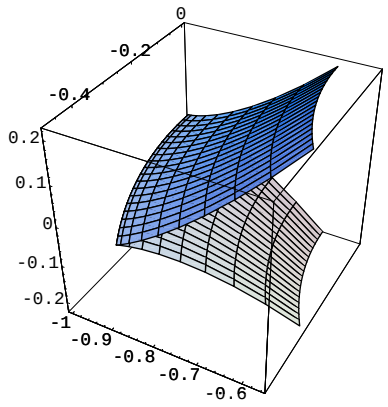
Singuläre Punkte des Flächenpatches haben auf die Berechnung der LIE mit Taylor-Modellen keinen Einfluß, solange die Fläche im Entwicklungspunkt des Taylor-Modells regulär und die Ableitung in der Singularität stetig ist.

Eine allgemeine Aussage darüber zu treffen, wann ein mit affiner Arithmetik berechnete LIE degeneriert ist, ist schwierig, da die Bestimmung der beiden Vektoren $\boldsymbol{f}^u$ und $\boldsymbol{f}^v$ in keinem Zusammenhang mit den differentialgeometrischen Eigenschaften der Fläche stehen. Es besteht also keine Gefahr, daß bei Existenz singulärer Punkte Degenerationen auftreten können, wie folgendes Beispiel zeigt.

Beispiel:

Gegeben sei das Flächenstück

$$\boldsymbol{f}(\alpha,t) = \begin{pmatrix} \cos\alpha(t^2-1) \\ \sin\alpha(t^2-1) \\ t^3 \end{pmatrix}$$



mit $(\alpha,t) \in [0,0.5] \times [-0.6,0.6]$. Da $\frac{\partial}{\partial t}\boldsymbol{f}(\alpha,0) = \mathbf{0}$ für alle $\alpha \in [0,0.5]$, degeneriert die Taylor-Modell basierte LIE (Abbildung 5.1) zu einer einparametrigen linearen Menge von Intervallboxen wenn der Mittelpunkt des Parametergebiets $(\alpha_0,t_0) = (0.25,0)$ als Entwicklungspunkt gewählt wird:

$$\mathbb{L}_{Taylor}(\alpha,t) := \begin{pmatrix} [-1.11706,-0.553217] \\ [-0.185176,0.362399] \\ [-0.648,0.648] \end{pmatrix} + \begin{pmatrix} 0.247404 \\ -0.968912 \\ 0 \end{pmatrix} \alpha; \qquad \alpha \in [0,0.5]$$

Eine mit Hilfe affiner Arithmetik berechnete LIE (Abbildung 5.2) degeneriert in diesem Fall nicht:

$$\mathbb{L}_{AA}(\epsilon_\alpha,\epsilon_t) := \begin{pmatrix} [-0.980539,-0.583779] \\ [-0.292927,-0.106509] \\ [-0.108,0.108] \end{pmatrix} + \begin{pmatrix} 0.0506125 \\ -0.198215 \\ 0 \end{pmatrix} \epsilon_\alpha + \begin{pmatrix} 0 \\ 0 \\ 0.108 \end{pmatrix} \epsilon_t$$

mit $(\epsilon_\alpha,\epsilon_t) \in [-1,1]^2$.

Ein Vergleich der Durchmesser der Intervallvektoren von $\mathbb{L}_{Taylor}$ und $\mathbb{L}_{AA}$ dieses Beispiels bzw. der Abbildungen 5.1 und 5.2 macht die bessere Approximationseigenschaft der auf affiner Arithmetik basierenden LIEs deutlich (siehe Abschnitt 5.6).

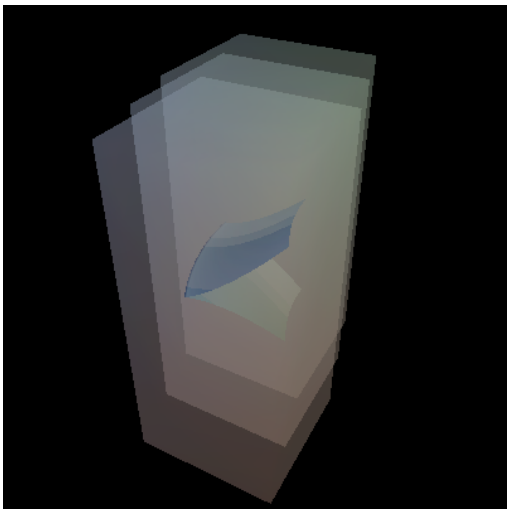


Abbildung 5.1: Taylor-Modell basierte degenerierte LIE

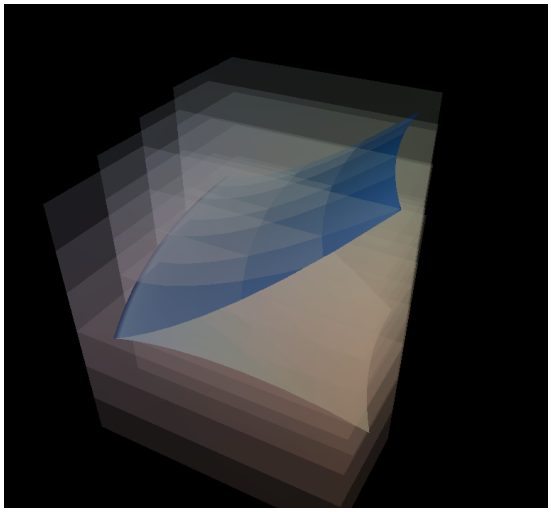


Abbildung 5.2: LIE basierend auf affiner Arithmetik

5.5.2 Der Einfluß von Doppelpunkten

Ist ein Flächenpatch $\boldsymbol{f} = \{\boldsymbol{p} \,|\, \exists (u,v) \in \mathbb{I} : \boldsymbol{p} = \boldsymbol{f}(u,v)\}$ nicht einfach, d.h. existieren Patchpunkte mit $\boldsymbol{p} = \boldsymbol{f}(u_1, v_1) = \boldsymbol{f}(u_2, v_2)$, ist die Zuordnung

$$\psi : \begin{cases} \mathbb{L}_f & \rightarrow & \boldsymbol{f} \\ \mathbb{p} & \mapsto & \boldsymbol{p} \end{cases}$$

von Patchpunkten und Intervallboxen der LIE $\mathbb{L}_f = \{\mathbb{p} \,|\, \exists (u,v) \in \mathbb{I} : \mathbb{p} = \mathbb{L}(u,v)\}$ in Objektraum nicht mehr injektiv. Es existieren also mehrere Intervallboxen, die ein und dem selben Punkt zugeordnet sind. Da LIEs eine lineare Struktur besitzen hat das zur Folge, daß sich die Intervallboxen stark vergrößern.
Obwohl in so einen Fall eine LIE definiert ist und mit beiden Methoden berechnet werden kann, ist aber klar, daß diese aufgrund ihrer linearen Struktur die Fläche nicht besonders gut approximieren kann.

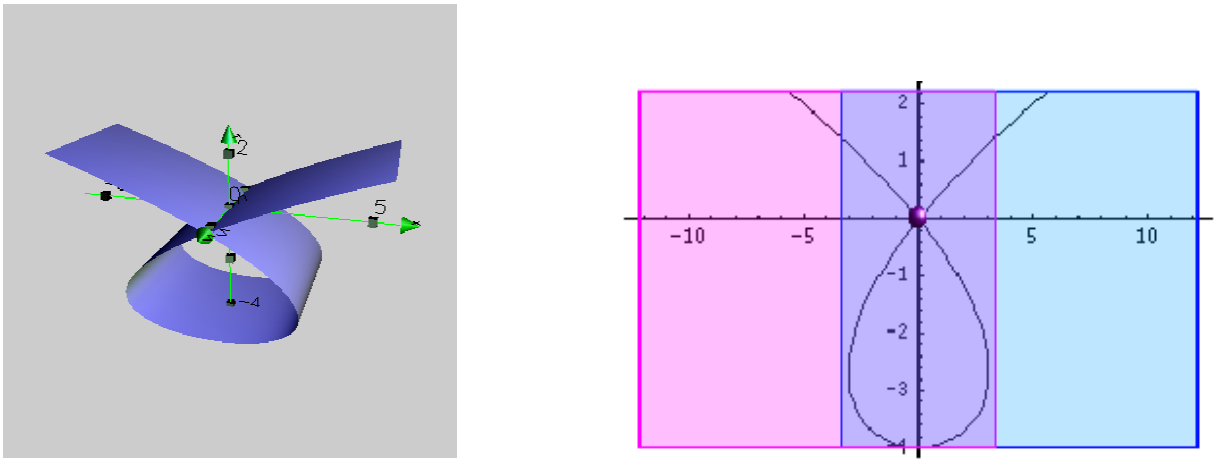


Abbildung 5.3: Die parametrische Kurve $\boldsymbol{c}(t) = (-4t+t^3, -4+t^2)^T,\ t \in [-2.5, 2.5]$ stelle für $s = const$ einen Querschnitt des Flächenstücks $\boldsymbol{f}(s,t) = (-4t+t^3, -4+t^2, s)^T,\ (s,t) \in [-2.5, 2.5]^2$ dar. $\boldsymbol{c}$ besitzt einen Doppelpunkt für $t = -2$ und $t = 2$. Das rote und das blaue Rechteck sind die zu den Punkten $\boldsymbol{c}(-2) = \boldsymbol{c}(2)$ gehörenden einschließenden Intervallboxen.(Mit affiner Arithmetik berechnet.)

5.5.3 Flächen mit starken Krümmungen

Den letzten Abschnitt kann man allgemeiner formulieren. Grunsätzlich gilt:

- Je größer die Krümmung und die Ausdehnung der Fläche ist, desto größer werden die Intervallpunkte der LIE.
- Läßt sich die Fläche zudem nicht injektiv auf die LIE projizieren, entsteht die gleiche Situation, wie bei den Doppelpunkten: Die Intervallpunkte der LIE vergrößern sich aufgrund der linearen Struktur der LIE extrem. Dadurch wird die Fläche in den meisten Fällen stark überschätzt.

5.6 Vergleich der Berechnungsmethoden

Experimentell kann beobachtet werden, daß Taylor-LIEs großer Flächenstücke im Vergleich zu affinen LIEs oft wesentlich schlechter approximieren. Dieser Effekt wird aber bei kleiner und damit meistens auch flacher werdenden Flächenstücken geringer.

Die Ursache hierfür liegt darin, daß affine Arithmetik zur Approximation nicht affiner Funktionen die Tchebycheff- oder Min-Max-Approximation [Stolfi, de Figueiredo '97] verwendet. Die dadurch gewährleistete gute Approximation überträgt sich bei der Auswertung der Parameterform zum Teil auf die Position der resultierenden linearen Intervallabschätzung. Die Lage einer durch affine Arithmetik berechnete LIE ist meistens sehr viel günstiger, als die einer Taylor-LIE. Letztere orientiert sich an der Lage einer der Tangentialebenen des Patches. Ist der Entwicklungspunkt der Taylorapproximation ungünstig gewählt, kann das zu großen Überschätzungen führen.

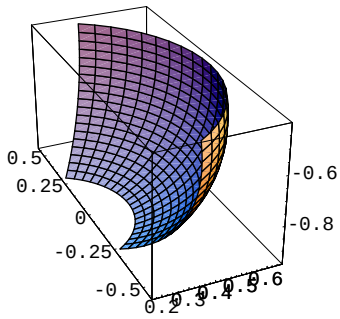
In Abschnitt 5.5 wurde bereits gezeigt, daß LIEs die auf affiner Arithmetik basieren, robuster auf eventuell auftretende Singularitäten des Flächenstücks reagieren.

Der Berechnungsaufwand ist für affine Arithmetik basierte LIEs höher als für solche, die mit Hilfe von Taylor-Modellen berechnet wurden. Grund dafür ist neben der im Vergleich zur Intervallarithmetik grundsätzlich aufwendigeren affinen Arithmetik ([Stolfi, de Figueiredo '97]) auch deren ineffiziente Implementierung[1]

Beispiel

Dieses Beispiel soll die besseren Approximationseigenschaften der LIEs demonstrieren, die mit affiner Arithmetik berechnet wurden. Berechnet wurden LIEs für die Fläche

$$\boldsymbol{f} = \begin{pmatrix} \cos(u)\,\sin(v)\,(1-\cos(v))/2 \\ \sin(u)\,\sin(v)\,(1-\cos(v))/2 \\ \cos(v) \end{pmatrix}$$



mit $(u, v) \in [-1, 1] \times [2, 2.8]$.

Die berechneten LIEs haben folgende Form

$$\mathbb{L}_{Taylor}(u,v) = \begin{pmatrix} [0.672582, 1.86574] \\ [-0.861035, 0.861035] \\ [0.883718, 0.959096] \end{pmatrix} + u \begin{pmatrix} 0 \\ 0.586773 \\ 0 \end{pmatrix} + v \begin{pmatrix} -0.412446 \\ 0 \\ -0.675463 \end{pmatrix}$$

[1] Bis heute liegt leider noch keine vollständige effiziente Implementierung der affinen Arithmetik vor. Für die Berechnungen, die dieser Arbeit zugrunde liegen, wurde die objektorientierte Implementierung von [Iwaarden, Stolfi '97] verwendet, da die Implementierung von [Stolfi '93] nicht vollständig ist.

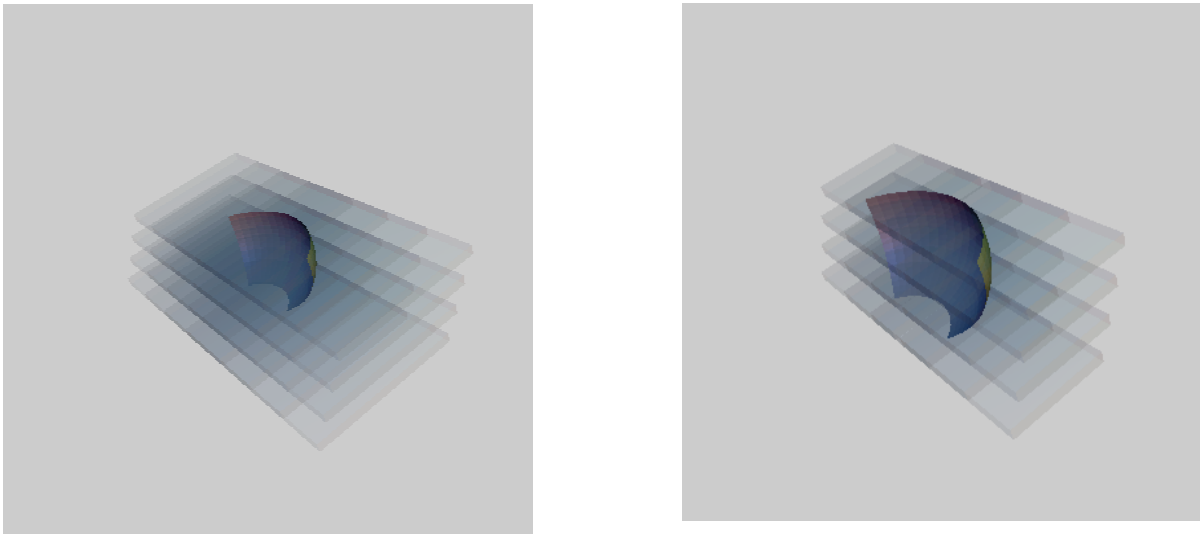


Abbildung 5.4: Taylor Model basierte LIE (links) und die engere LIE berechnet mit affiner Arithmetik (rechts) aus dem Beispiel in Abschnitt 5.6.

und

$$\mathbb{L}_{AA}(u,v) = \begin{pmatrix} [0.887621, 1.47281] \\ [-0.468701, 0.468701] \\ [0.86636, 0.93478] \end{pmatrix} + u \begin{pmatrix} 0 \\ 0.510245 \\ 0 \end{pmatrix} + v \begin{pmatrix} -0.313925 \\ 0 \\ -0.670358 \end{pmatrix}$$

Abbildung 5.4 zeigt die beiden LIEs im Vergleich.

5.7 Vergleich von LIE´s mit anderen Einschließungskörpern

5.7.1 Kompakte Kontra parametrische Hüllkörper

Die klassische Einschließungskörper für parametrische Flächen, wie achsenparallele Quader, orientierte Quader, Parallelepipede, Kugeln und Ellipsoide, sind kompakte Primitive die alle ein Merkmal gemeinsam haben: Sie sind Hüllkörper des gesamten Flächenstücks unabhängig von der Lage einzelner Punkte des Patches. Es existiert also keine Abhängigkeit zwischen den Punkten des Hüllkörpers und den Flächenpunkten. Hierin liegt der größte und bemerkenswerteste Unterschied zwischen einem kompakten Hüllkörper und einer linearen Intervallabschätzung. Anders als bei herkömmlichen Hüllkörpern ist jeder Intervallpunkt der LIE Hüllkörper eines ihm eindeutig zugeordneten Punktes des Patches. LIEs liegen, genau wie das Flächenstück, in parametrisierter Form vor und es besteht eine direkte Verbindung zwischen der Parametrisierung des Patches und der Parametrisierung der LIE. Im nächsten Kapitel wird der Nutzen dieses Konzeptes deutlich: Es wird gezeigt, daß sich die Ergebnisse des Schnittalgorithmus für zwei LIEs direkt auf die beiden eingeschlossenen Flächenstücke übertragen lassen.

5.7.2 LIEs und achsenparallele Einschließungen

Achsenparallele Einschließungen für Flächenpatches sind die wohl am weitesten verbreiteten Einschließungskörper, da sie sehr einfach zu berechnen sind und auch einen einfachen Schnittest erlauben. Eine sichere Möglichkeit zur Bestimmung einer achsenparallelen Einschließung $\mathbb{f} \in \mathbb{IR}^3$ ist die direkte Auswertung der Parameterdarstellung $\boldsymbol{f}(u,v)$ mit den Parameterintervallen I_u und I_v [Gleicher, Kass '92]

$$\mathbb{f} := \boldsymbol{f}(I_u, I_v)$$

oder deren Affinformen $\hat{u}$ und $\hat{v}$ ([de Figueiredo '96])

$$\mathbb{f} := [\ \boldsymbol{f}^0 - \sum_{i=1}^{n} |\boldsymbol{f}^i|,\ \boldsymbol{f}^0 + \sum_{i=1}^{n} |\boldsymbol{f}^i|\ ]$$

wobei $\boldsymbol{f}(\hat{u},\hat{v}) = \hat{\boldsymbol{f}}(\epsilon_1, ..., \epsilon_n) = \boldsymbol{f}^0 + \sum_{i=1}^{n} \epsilon_i \boldsymbol{f}^i$.

Der Schnittest für die zwei Einschließungen $\mathbb{f}$ und $\mathbb{g}$ bestimmt sich aus dem Schnitt der beiden Intervallvektoren: Ist $\mathbb{f} \cap \mathbb{g} = \emptyset$, schneiden sich auch die beiden Flächenstücke nicht.

In [Filip et al. '86] werden mit Hilfe einer Abschätzung der zweiten Ableitung achsenparallele Quader für beliebige Flächen bezüglich dreieckiger und quadratischer Parametergebiete berechnet. Dafür wird eine ebene Approximation der Fläche aus den Eckpunkten des Patches berechnet und eine Abschätzung des Fehlers mit Hilfe der zweiten Ableitung. Die dadurch erzielten Einschließungen sind besser als bei einer direkten Auswertung mit Intervallarithmetik.
Andere Algorithmen beschränken sich meistens auf polynomiale oder rationale Flächen. Der in [Gopalsamy et al. '91] entwickelte Algorithmus zur Berechnung achsenparalleler Quader ist auf polynomiale Kurven und Flächen beschränkt. Mit Hilfe der Berechnung spezieller Flächenpunkte und den zum Polynom gehörenden Lagrange Polynomen kann eine Konstante berechnet werden, die eine Abschätzung erlaubt, die in 96% der Fälle eine bessere Einschließung liefert, als Methoden, die achsenparallele Boxen auf Basis von Kontrollpolygonen berechnen.

LIEs sind im Vergleich zu den oben genannten Berechnungsmethoden für achsenparallele Einschließungen aufwendiger zu bestimmen. Durch ihre orientierte Lage sind sie aber im Allgemeinen enger als achsenparallele Einschließungen.

Es gibt allerdings auch Ausnahmen: Sei beispielsweise die einzuschließende Fläche $\boldsymbol{f}$ rational und besitze außerdem Selbstdurchdringungen. In Abschnitt 5.5.2 wurde bereits gezeigt, daß die LIEs von Flächen mit Selbstdurchdringungen stark übeschätzt sind. Da $\boldsymbol{f}$ rational ist, kann die Fläche mit Hilfe eines (Bézier-) Kontrollnetzes dargestellt werden, dessen achsenparallele Hülle auch Hülle von $\boldsymbol{f}$ ist und durchaus sehr viel enger sein kann als die LIE.

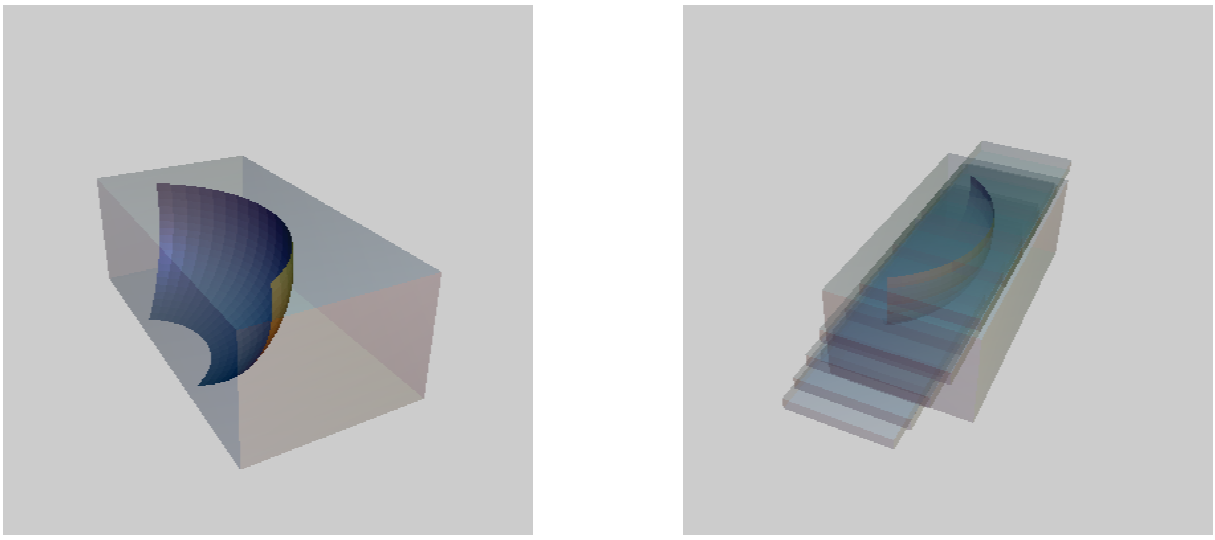


Abbildung 5.5: Das linke Bild zeigt die mit Intervallaritmetik berechnete achsenparallele Bounding Box der Fläche aus dem Beispiel in Abschnitt 5.6. Das rechte Bild zeigt Box und Fläche in Kombination mit der LIE die mit affiner Arithmetik berechnet wurde.

5.7.3 LIEs und Parallelepipede

Die Berechnung von Parallelepipeden ist wesentlich aufwendiger, als die Bestimmung achsenparalleler Einschließungen. Allerdings erhält man durch ihre dem Flächenstück angepaßte Lage auch hier wesentlich engere Abschätzungen, die zu einer Reduzierung der Anzahl der Unterteilungen führt. In [Bürger, Schaback '93] wird der Einsatz von achsenparallelen Boxen mit der Verwendung von Parallelepipeden als Hüllkörper in Unterteilungsalgorithmen für rationale Flächen verglichen. Anhand einer Komplexitätsananlyse und mittels Experimenten wird festgestellt, daß sich bei steigender Genauigkeit die Verwendung von Parallelepipeden bzw. orientierten Bounding Boxen in jedem Fall auszahlt.

In der Literatur finden sich unterschiedliche Ansätze, parallelepipedförmige Einschließungen für Flächenstücke zu berechnen. Wie bereits in Kapitel 3 erwähnt, können für polynomiale und rationale Flächen auch orientierte Einschließungskörper als konvexe Hülle von Kontrollpunkten berechnet werden. Solche Einschließungen können unter Umständen enger sein, als eine LIE.

Für allgemeine parametrische Flächen werden einschließende Parallelepipede oft mit Hilfe der ersten oder zweiten Ableitung berechnet. So beispielsweise in [Huber '99]. Grundlage für die Berechnung einer Einschließung des Flächenpatches $\boldsymbol{f}(u,v), (u,v) \in \mathbb{I} = I_u \times I_v = [\underline{u}, \overline{u}] \times [\underline{v}, \overline{v}]$ bildet der Mittelwertsatz für bivariante vektorwertige Funktionen (der dem Satz von Taylor für eine Taylorentwicklung vom Grad Null entspricht).
Aus diesem Satz folgt erstens für ein $(u_0, v_0) \in \mathbb{I}$

$$\boldsymbol{f}(u,v) \in \mathfrak{q}(u,v) = \boldsymbol{f}(u_0,v_0) + (u-u_0)\boldsymbol{f}_u(I_u) + (v-v_0)\boldsymbol{f}_v(I_v), \qquad \forall (u,v) \in \mathbb{I}$$

und zweitens die Behauptung, daß die konvexe Hülle der vier Quader $\mathfrak{q}_1 = \mathfrak{q}(\underline{u}, \underline{v})$, $\mathfrak{q}_2 = \mathfrak{q}(\underline{u}, \overline{v})$,

$\mathfrak{q}_3 = \mathfrak{q}(\overline{u}, \underline{v})$, $\mathfrak{q}_4 = \mathfrak{q}(\overline{u}, \overline{v})$ eine konvexe Hülle von $\boldsymbol{f}$ ist.

Der Algorithmus von Huber hat folgende Form:

1. Berechne die engste achsenparallele Einschließung $\mathbb{b}_1$ von $\mathfrak{q}_1, \mathfrak{q}_2, \mathfrak{q}_3, \mathfrak{q}_4$.

2. Berechne die Spannvektoren $\boldsymbol{y}_1, \boldsymbol{y}_2$ für die Trägerebene des Epipeds,

$$\boldsymbol{y}_1 := \frac{1}{4\, rad(I_u)}(\boldsymbol{f}_3 - \boldsymbol{f}_1 + \boldsymbol{f}_4 - \boldsymbol{f}_2)$$

$$\boldsymbol{y}_2 := \frac{1}{4\, rad(I_v)}(\boldsymbol{f}_2 - \boldsymbol{f}_1 + \boldsymbol{f}_4 - \boldsymbol{f}_3)$$

   wenn $\boldsymbol{f}_i, i = 1, ..., 3$ die vier Eckpunkte des Patches bezeichnen.

3. Berechne die drei Normalenvektoren der Seitenebenen

$$\boldsymbol{n}_1 := \boldsymbol{y}_1 \times \boldsymbol{y}_2 \qquad \boldsymbol{n}_2 := \boldsymbol{n}_1 \times \boldsymbol{y}_2 \qquad \boldsymbol{n}_3 := \boldsymbol{n}_1 \times \boldsymbol{y}_1$$

4. Berechne die Ebenengleichungen $\underline{\epsilon}_i : \boldsymbol{n}_i\boldsymbol{x} = \underline{\lambda}_i$ und $\overline{\epsilon}_i : \boldsymbol{n}_i\boldsymbol{x} = \overline{\lambda}_i$, $i = 1, 2, 3$ so, daß die Eckquader $\mathfrak{q}_1, \mathfrak{q}_2, \mathfrak{q}_3, \mathfrak{q}_4$ von den Ebenen umschlossen werden.

Durch die sechs einschließenden Ebenen ist das gesuchte Epiped bereits vollständig definiert. Um später allerdings einen effizienten Schnittest durchzuführen, müssen auch die Kanten unde Ecken des Epipeds bestimmt werden:

5. Berechne den Schnittpunkt $\boldsymbol{a}$ der Ebenen $\underline{\epsilon}_i, i = 1, 2, 3$.

6. Berechne die Kantenvektoren des Epipeds als Richtungsvektoren der Schnittgeraden von jeweils zwei Ebenen der $\underline{\epsilon}_i, i = 1, 2, 3$.

Vergleicht man diesen vielstufigen Algorithmus mit den Algorithmen zur Berechnung von LIEs kann folgendes festgehalten werden:

- Da Taylor-Modell basierte LIEs auf einer Taylorentwicklung vom Grad 1 beruhen, folgt schon aus der Definition, daß eine solche LIE eine engere Einschließung liefert, als die von Huber berechneten Parallelepipede. In Abschnitt 5.7 wurde diskutiert, daß LIEs, die mit affiner Arithmetik berechnet wurden im Allgemeinen noch engere Einschließungen liefern. (Siehe Abbildung 5.6)

- Für Taylor-Modell basierte LIEs müssen erste und zweite Ableitungen berechnet werden, für die Huberschen Parallelepipede nur Ableitungen erste Ordnung, dafür aber zusätzlich die Schritte 2 – 6 des oben beschriebenen Algorithmus. Ein direkter Vergleich ist nicht möglich, da der Aufwand zur Berechnung der zweiten Ableitungen von der Parameterdarstellung selbst abhängt. (Für Flächenpatches mit einfachen Parameterdarstellungen ist die Berechnung von LIEs sicher weniger aufwendig, als die eines Parallelepipeds.)

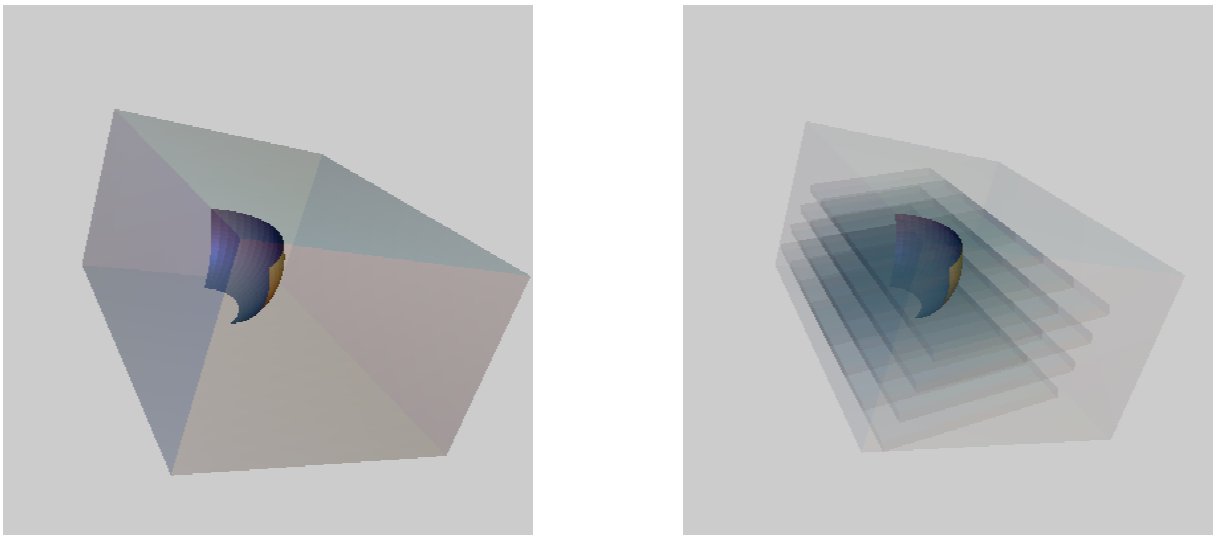


Abbildung 5.6: Die Bilder zeigen das Parallelepiped von Huber (links) für die Fläche des Beispiels aus Abschnitt 5.6 und die zugehörende Taylor-Modell-LIE (rechts)

6

Der Schnittest und die Reduzierung des Parametergebietes

LIEs sind parametrische Intervallebenenstücke, also Parallelogramme im $\mathbb{R}^3$, die statt aus gewöhnlichen Punkten aus einer Menge von Intervallpunkten bestehen. In diesem Kapitel wird deshalb zunächst ein Algorithmus entwickelt, der mit Hilfe von Intervallarithmetik das Schnittgeradensegment zweier gewöhnlicher Parallelogramme im Raum berechnet. Die Parallelogramme werden als Stücke parametrischer Ebenen dargestellt, der Schnitt als parametrisches Geradenstück. Der Algorithmus berechnet neben einer parametrischen Darstellung des Schnittgeradenstücks auch die Parameterteilgebiete, die das Schnittsegment enthalten. Es wird außerdem gezeigt, daß die Anwendung der Intervallarithmetik in diesem Algorithmus nicht zu Überschätzungen führt, sondern ein bis auf Rundungsfehler exaktes Ergebnis liefert.

Der für gewöhnliche Parallelogramme entwickelte Algorithmus wird dann auf den Schnitt zweier LIEs übertragen. Ergebnis sind drei Algorithmen, die eine relativ enge achsenparallele Einschließung des Schnittes, einen schmalen einschließenden Streifen in Form einer Intervallgeraden bzw. in Form zweier paralleler Geradensegmente berechnen.

Da die Parametrisierung der LIEs mit der Parametrisierung der Flächenpatches übereinstimmt, sind die Einschließungen des Schnittes zweier LIEs auch gleichzeitig Einschließungen der Parameterwerte einer möglichen Schnittkurve der von den LIEs eingeschlossenen Patches. Durch Einsetzen der gewonnenen Einschließungen der Parameterwerte in die entsprechende Flächengleichung erhält man Einschließungen der Schnittkurve im Objektraum.

6.1 Der Schnitt zweier Parallelogramme im $\mathbb{R}^3$

Gegeben seien zwei Parallelogramme $\boldsymbol{e}_1$ und $\boldsymbol{e}_2$ im $\mathbb{R}^3$ als Teilstücke nicht paralleler parametrischer Ebenen:

$$\boldsymbol{e}_1(u,v) = \boldsymbol{p} + u\boldsymbol{y}_1 + v\boldsymbol{y}_2; \qquad (u,v) \in I_u \times I_v = \mathbb{I} \tag{6.1}$$

$$\boldsymbol{e}_2(s,t) = \boldsymbol{q} + s\boldsymbol{w}_1 + t\boldsymbol{w}_2; \qquad (s,t) \in J_s \times J_t = \mathbb{J} \tag{6.2}$$

Gesucht wird, falls vorhanden, die genaue Schnittstrecke der beiden Parallelogramme in allen vier möglichen Parametrisierungen, also nach s, t, u und v. Oder anders gesprochen: Gesucht werden die Parametergebiete $\tilde{\mathbb{I}} \subseteq \mathbb{I}$ und $\tilde{\mathbb{J}} \subseteq \mathbb{J}$, die genau die Parameterwerte s,t,u und v der Schnittstrecke enthalten.

Durch Gleichsetzen von (6.1) und (6.2) erhält man das unterbestimmte lineare Gleichungssystem,

$$(\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad -\boldsymbol{w}_1 \quad -\boldsymbol{w}_2) \begin{pmatrix} u \\ v \\ s \\ t \end{pmatrix} = \boldsymbol{r} \tag{6.3}$$

mit $\boldsymbol{r} := \boldsymbol{q} - \boldsymbol{p}$, aus dem die Schnittgerade der beiden Trägerebenen $\boldsymbol{e}_1(u,v), u, v \in \mathbb{R}$ und $\boldsymbol{e}_2(s,t), s, t \in \mathbb{R}$ berechnet werden kann.

Sind jeweils drei der Vektoren $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{w}_1, \boldsymbol{w}_2$ linear unabhängig, lassen sich durch Umformung und Anwendung der Cramerschen Regel auf das System (6.3) folgende Gleichungen ableiten:

$$\begin{aligned}
s_1(u) &= \tfrac{1}{\alpha}(\kappa + \beta\, u) & \qquad u_1(s) &= \tfrac{1}{\beta}(-\kappa + \alpha s) \\
s_2(v) &= \tfrac{1}{\gamma}(\lambda - \beta\, v) & \qquad u_2(t) &= \tfrac{1}{\delta}(\mu - \alpha t) \\
t_1(u) &= \tfrac{1}{\alpha}(\mu - \delta\, u) & \qquad v_1(s) &= \tfrac{1}{\beta}(\lambda - \gamma s) \\
t_2(v) &= \tfrac{1}{\gamma}(\nu + \delta\, v) & \qquad v_2(t) &= \tfrac{1}{\delta}(-\nu + \gamma t)
\end{aligned}$$

mit

$$\begin{aligned}
\alpha &:= |\boldsymbol{y}_2 \quad -\boldsymbol{w}_1 \quad -\boldsymbol{w}_2| & \qquad \kappa &:= |\boldsymbol{y}_2 \quad \boldsymbol{r} \quad -\boldsymbol{w}_2| \\
\beta &:= |\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad -\boldsymbol{w}_2| & \qquad \lambda &:= |\boldsymbol{y}_1 \quad \boldsymbol{r} \quad -\boldsymbol{w}_2| \\
\gamma &:= |\boldsymbol{y}_1 \quad -\boldsymbol{w}_1 \quad -\boldsymbol{w}_2| & \qquad \mu &:= |\boldsymbol{y}_2 \quad -\boldsymbol{w}_1 \quad \boldsymbol{r}| \\
\delta &:= |\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad -\boldsymbol{w}_1| & \qquad \nu &:= |\boldsymbol{y}_1 \quad -\boldsymbol{w}_1 \quad \boldsymbol{r}|
\end{aligned}$$

die sich zu vier parametrischen Gleichungen der Schnittgeraden in den Parametergebieten der Parallelogramme zusammenfassen lassen:

$$g_1 : \ \boldsymbol{g}_1(u) := \begin{pmatrix} s_1(u) \\ t_1(u) \end{pmatrix} = \frac{1}{\alpha}\left(\begin{pmatrix} \kappa \\ \mu \end{pmatrix} + u \begin{pmatrix} \beta \\ -\delta \end{pmatrix}\right); \ u \in I_u \tag{6.4}$$

$$g_2 : \ \boldsymbol{g}_2(v) := \begin{pmatrix} s_2(v) \\ t_2(v) \end{pmatrix} = \frac{1}{\gamma}\left(\begin{pmatrix} \lambda \\ \nu \end{pmatrix} + v \begin{pmatrix} -\beta \\ \delta \end{pmatrix}\right); \ v \in I_v \tag{6.5}$$

und

$$h_1 : \ \boldsymbol{h}_1(s) := \begin{pmatrix} u_1(s) \\ v_1(s) \end{pmatrix} = \frac{1}{\beta}\left(\begin{pmatrix} -\kappa \\ \lambda \end{pmatrix} + s \begin{pmatrix} \alpha \\ -\gamma \end{pmatrix}\right); \ s \in J_s \tag{6.6}$$

$$h_2 : \ \boldsymbol{h}_2(t) := \begin{pmatrix} u_2(t) \\ v_2(t) \end{pmatrix} = \frac{1}{\delta}\left(\begin{pmatrix} \mu \\ -\nu \end{pmatrix} + t \begin{pmatrix} -\alpha \\ \gamma \end{pmatrix}\right); \ t \in J_t \tag{6.7}$$

Diese vier Geradensegmente enthalten jeweils die Schnittstrecke der beiden Parallelogramme, können diese aber je nach Lage der Parallelogramme überschätzen.

6.1.1 Geometrische Lösung

Wertet man $\boldsymbol{g}_1(u)$ für $u_{min} := \min(I_u)$ und $u_{max} := \max(I_u)$ aus, erhält man die Parameterwerte $\boldsymbol{g}^1_{min} := (s^1_{min}, t^1_{min})^T, \boldsymbol{g}^1_{max} := (s^1_{max}, t^1_{max})^T$ der Schnittpunkte der das Parallelogramm $\boldsymbol{e}_1$ begrenzenden v-Linien mit der Trägerebene des Parallelogramms $\boldsymbol{e}_2$.

Analog erhält man aus den Gleichungen $\boldsymbol{g}_2(v)$ für $v_{min} := \min(I_v)$ und $v_{max} := \max(I_v)$ die Parameterwerte $\boldsymbol{g}^2_{min} := (s^2_{min}, t^2_{min})^T, \boldsymbol{g}^2_{max} := (s^2_{max}, t^2_{max})^T$ der Schnittpunkte der das Parallelogramm $\boldsymbol{e}_1$ begrenzenden u-Linien mit der Trägerebene des Parallelogramms $\boldsymbol{e}_2$.

Durch $\overline{\boldsymbol{g}^1_{min}\boldsymbol{g}^1_{max}}$ und $\overline{\boldsymbol{g}^2_{min}\boldsymbol{g}^2_{max}}$ sind zwei Geradensegmente definiert, die das Schnittsegment sicher enthalten.

Berechnet man auf analoge Weise die Schnittpunkte der Begrenzungsgeraden von von $\boldsymbol{h}_1$ und $\boldsymbol{h}_2$ mit $\boldsymbol{e}_1$ und damit die Geradensegmente $\overline{\boldsymbol{h}^1_{min}\boldsymbol{h}^1_{max}}$ und $\overline{\boldsymbol{h}^2_{min}\boldsymbol{h}^2_{max}}$, ist die tatsächliche Schnittstrecke (falls vorhanden) der Durchschnitt der vier berechneten Geradensegmente. (Siehe Abbildung 6.1.)

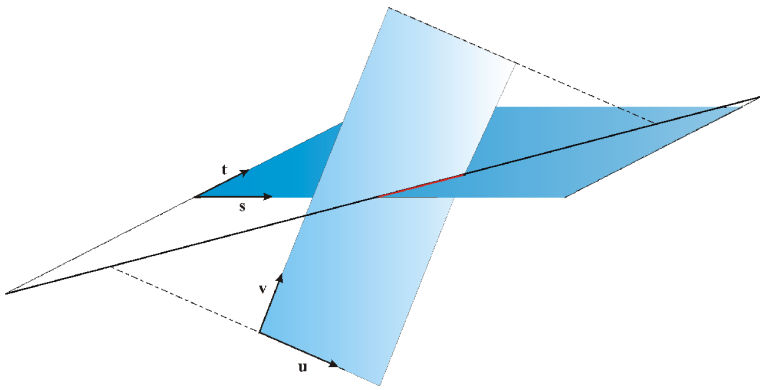


Abbildung 6.1: Schnittstrecke zweier Parallelogramme (rot): Es müssen alle Durchstoßpunkte der Begrenzungsgeraden der Parallelogramme jeweils mit der Trägerebene des jeweils anderen Parallelogramms geschnitten werden.

6.1.2 Berechnung des Schnittsegments mit Hilfe von Intervallarithmetik

Behauptung 1
Sei $I \subset \mathbb{R}$ ein beliebiges reelles Intervall. Dann ist $\boldsymbol{g}_1(I)$ gleich dem Wertebereich $W(\boldsymbol{g}_1, I)$ von $\boldsymbol{g}_1(u),\ u \in I$. Analoges gilt für $\boldsymbol{g}_2(v), \boldsymbol{h}_1(s)$ und $\boldsymbol{h}_2(t)$.

Beweis:
Da $\boldsymbol{g}_1(u), \boldsymbol{g}_2(v), \boldsymbol{h}_1(s)$ und $\boldsymbol{h}_2(t)$ linear sind und der jeweilige Parameter nur einmal im

Funktionsterm vorkommt, folgt die Behauptung direkt aus der Anwendung von Satz 1.1.2 (Anhang A) auf die Funktionen der einzelnen Komponenten. □

Berechnet man also die Wertebereiche der Funktionen $\boldsymbol{g}_1(u)$ und $\boldsymbol{g}_2(v)$ jeweils direkt mittels Intervallarithmetik, erhält man aufgrund Behauptung 1 das gleiche Ergebnis wie oben:
Es gilt $\min(\boldsymbol{g}_1(I_u)) \equiv \boldsymbol{g}^1_{min}$ und $\min(\boldsymbol{g}_2(I_v)) \equiv \boldsymbol{g}^2_{min}$.

Behauptung 2
Der Intervallvektor

$$\tilde{\mathbb{J}} := \boldsymbol{g}_1(I_u) \cap \boldsymbol{g}_2(I_v) \cap \mathbb{J}$$

ist für $\tilde{\mathbb{J}} \neq \emptyset$ eine (überschätzte) Einschließung für die Parameterwerte des Schnittes des Parallelogramms $\boldsymbol{e}_1$ mit der Trägerebenen des Parallelogramms $\boldsymbol{e}_2$.

Beweis:
Sei $\hat{\mathbb{J}} := \boldsymbol{g}_1(I_u) \cap \boldsymbol{g}_2(I_v)$ und $\tilde{\mathbb{J}} := \hat{\mathbb{J}} \cap \mathbb{J}$

1. Ist $\hat{\mathbb{J}} \not\subseteq \mathbb{J}$ aber $\tilde{\mathbb{J}} \neq \emptyset$, so bildet das Gebiet $\tilde{\mathbb{J}}$ eine im allgemeinen überschätzte Einschließung des Parametergebietes der Schnittstrecke bezüglich s und t.

   Denn: In diesem Fall liegen die beiden inneren ($\boldsymbol{e}_2(\min(\hat{\mathbb{J}}))$ und $\boldsymbol{e}_2(\max(\hat{\mathbb{J}}))$) der berechneten vier Schnittpunkte der u- und v-Linien des Parallelogramms $\boldsymbol{e}_2$ teilweise oder ganz außerhalb des Parallelogramms $\boldsymbol{e}_1$.

   Da $\tilde{\mathbb{J}} \neq \emptyset$, ist die eigentliche Schnittstrecke eine Teilstrecke der berechneten Strecke und $\hat{\mathbb{J}}$ ist eine Überschätzung des dazugehörigen Parametergebietes.

   Ist die Schnittstrecke nicht parallel zu einer der Begrenzungslinien des Parametergebiets, so ist auch $\tilde{\mathbb{J}}$ eine Überschätzung des Parametergebiets der eigentlichen Schnittstrecke, da die Schnittpunkte der Strecke mit den Seiten des Parametergebiets in den meisten Fällen nicht auf den Eckpunkten des Schnittes liegen. (Siehe Abbildung 6.2.)

2. Falls $\hat{\mathbb{J}} \subseteq \mathbb{J}$, so bildet $\hat{\mathbb{J}} =: \tilde{\mathbb{J}}$ bereits die exakte Einschließung des Parametergebietes, in dem die Schnittstrecke liegt.

   Denn: In diesem Fall liegen die inneren beiden ($\boldsymbol{e}_2(\min(\hat{\mathbb{J}}))$ und $\boldsymbol{e}_2(\max(\hat{\mathbb{J}}))$) der berechneten vier Schnittpunkte der u- und v-Linien des Parallelogramms $\boldsymbol{e}_2$ innerhalb des Parallelogramms $\boldsymbol{e}_1$ und sind somit die exakten Begrenzungspunkte der Schnittstrecke.

3. Ist das Intervall leer, schneiden sich die beiden Parallelogramme nicht.

Behauptung 3
Sei $\tilde{\mathbb{J}} \neq \emptyset$. Der Intervallvektor

$$\tilde{\mathbb{I}} := \boldsymbol{h}_1(\tilde{J}_s) \cap \boldsymbol{h}_2(\tilde{J}_t)$$

ist die genaue Einschließung der Parameterwerte des Schnittes des Palallelogramms $\boldsymbol{e}_2$ mit der Trägerebenen des Parallelogramms $\boldsymbol{e}_1$.

Beweis:
Berechnet man $\tilde{\mathbb{I}} := \boldsymbol{h}_1(\tilde{J}_s) \cap \boldsymbol{h}_2(\tilde{J}_t)$, so gilt $\tilde{\mathbb{I}} \subseteq \mathbb{I}$:

$$\begin{aligned}\tilde{\mathbb{I}} &= \boldsymbol{h}_1(\tilde{J}_s) \cap \boldsymbol{h}_2(\tilde{J}_t) \\ &= \boldsymbol{h}_1(s_1(I_u) \cap s_2(I_v) \cap J_t) \cap \boldsymbol{h}_2(t_1(I_u) \cap t_2(I_v) \cap J_s)\end{aligned}$$

$\boldsymbol{h}_1, \boldsymbol{h}_2$ injektiv, da linear

$$= \boldsymbol{h}_1(s_1(I_u)) \cap \boldsymbol{h}_1(s_2(I_v)) \cap \boldsymbol{h}_1(J_s) \cap \boldsymbol{h}_2(t_1(I_u)) \cap \boldsymbol{h}_2(t_2(I_v)) \cap \boldsymbol{h}_2(J_s)$$

$\boldsymbol{h}_1 \circ s_1 = id$ und $\boldsymbol{h}_2 \circ t_1 = id$

$$\begin{aligned}&= \mathbb{I} \cap \boldsymbol{h}_1(s_2(I_v)) \cap \boldsymbol{h}_1(J_s) \cap \mathbb{I} \cap \boldsymbol{h}_2(t_2(I_v)) \cap \boldsymbol{h}_2(J_s) \\ &\subseteq \mathbb{I}\end{aligned}$$

$\tilde{\mathbb{I}}$ bildet folglich, analog zum Beweis Behauptung 2, Fall 2), die genauen Einschließung des u-v-Parametergebiets in dem die Schnittstrecke liegt. (Siehe Abbildung 6.3).

Behauptung 4
Sei $\tilde{\mathbb{I}} \neq \emptyset$. Der Intervallvektor

$$\tilde{\tilde{\mathbb{J}}} := \boldsymbol{g}_1(\tilde{I}_u) \cap \boldsymbol{g}_2(\tilde{I}_v)$$

ist die genaue Einschließung Parameterwerte des Schnittes des Palallelogramms $\boldsymbol{e}_1$ mit der Trägerebenen des Parallelogramms $\boldsymbol{e}_2$.

Beweis analog zu Behauptung 3. (Siehe auch Abbildung 6.4).

Die Behauptungen 2 – 4 werden zu folgendem Satz zusammengefaßt:

Satz 6.1.1
Sind jeweils drei der Vektoren $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{w}_1, \boldsymbol{w}_2$ linear unabhängig, so ist die Schnittstrecke der beiden Parallelogramme $\boldsymbol{e}_1$ und $\boldsymbol{e}_2$ durch jede der vier Gleichungen $\boldsymbol{g}_1(u),\ u \in \tilde{I}_u$, $\boldsymbol{g}_2(v),\ v \in \tilde{I}_v$, $\boldsymbol{h}_1(s)\ s \in \tilde{\tilde{J}}_s$ und $\boldsymbol{h}_2(t)\ t \in \tilde{\tilde{J}}_t$ gegeben, wobei

$$\tilde{\mathbb{J}} := \boldsymbol{g}_1(I_u) \cap \boldsymbol{g}_2(I_v) \cap \mathbb{J}$$

$$\tilde{\mathbb{I}} := \boldsymbol{h}_1(\tilde{J}_s) \cap \boldsymbol{h}_2(\tilde{J}_t)$$

$$\tilde{\tilde{\mathbb{J}}} := \boldsymbol{g}_1(\tilde{I}_u) \cap \boldsymbol{g}_2(\tilde{I}_v)$$

Ist mindestens eines der Parametergebiete $\tilde{\mathbb{J}}, \tilde{\mathbb{I}}$ oder $\tilde{\tilde{\mathbb{J}}}$ leer, so schneiden sich die Parallelogramme nicht.

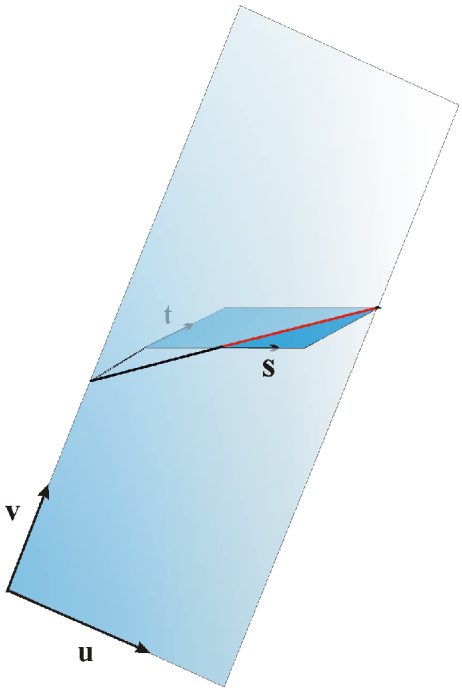


Abbildung 6.2: Die Parallelogramme aus Abbildung 6.1 nach dem ersten Reduktionsschritt,...

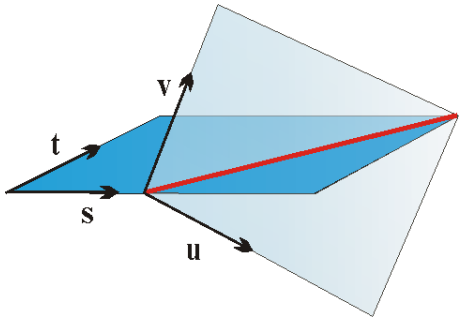


Abbildung 6.3: ... nach dem zweiten...

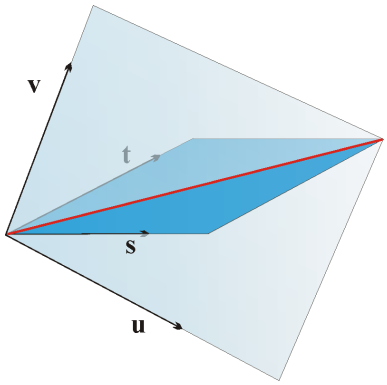


Abbildung 6.4: ... und nach dem dritten Schritt.

6.1.3 Sonderfälle

Sonderfälle treten auf, falls eine oder mehrere der Determinanten α, β, γ und δ verschwinden. Geometrisch interpretiert heißt das, wenn

1. eine Kante des einen Parallelogramms parallel zum anderen Parallelogramm
   (eine der Determinanten verschwindet)
   oder sogar parallel zu einer seiner Kanten ist.
   (jeweils zwei verschwindende Determinanten: $\gamma = \delta = 0$,$\gamma = \beta = 0$, $\alpha = \beta = 0$ oder $\alpha = \delta = 0$)

2. $\boldsymbol{e}_1$ und $\boldsymbol{e}_2$ parallel sind, aber nicht in der gleichen Ebene liegen ($\alpha = \beta = \gamma = \delta = 0$ und mindestens zwei der Determinanten $\kappa, \lambda, \mu, \nu \neq 0$).

3. $\boldsymbol{e}_1$ und $\boldsymbol{e}_2$ in der gleichen Ebene liegen. ($\alpha = \beta = \gamma = \delta = \kappa = \lambda = \mu = \nu = 0$).

Fall 1 kann wie folgt behandelt werden:
Für

- $\alpha = 0$ sei $\boldsymbol{g}_1(u) := \mathbb{J}$

- $\gamma = 0$ sei $\boldsymbol{g}_2(v) := \mathbb{J}$
- $\beta = 0$ sei $\boldsymbol{h}_1(s) := \mathbb{I}$
- $\delta = 0$ sei $\boldsymbol{h}_2(t) := \mathbb{I}$

Satz 6.1.1 gilt auch hier: $\tilde{\mathbb{I}}$ und $\tilde{\tilde{\mathbb{J}}}$ sind die optimalen Einschließungen der Lösung.

Im Fall 2 gilt: die beiden Parallelogramme schneiden sich nicht.

Im Fall 3 muß durch Schnitt der Begrenzungsgeraden der beiden Parallelogramme das Parametergebiet festgestellt werden, in dem sich die beiden Parallelogramme schneiden.

6.2 Der Schnitt zweier LIEs

Gegeben seien zwei LIEs im $\mathbb{R}^3$

$$\mathbb{L}_1(u,v) = \mathbb{p} + u\boldsymbol{y}_1 + v\boldsymbol{y}_2; \qquad (u,v) \in I_u \times I_v = \mathbb{I} \tag{6.8}$$

$$\mathbb{L}_2(s,t) = \mathbb{q} + s\boldsymbol{w}_1 + t\boldsymbol{w}_2; \qquad (s,t) \in J_s \times J_t = \mathbb{J} \tag{6.9}$$

wobei jeweils drei der Vektoren $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{w}_1, \boldsymbol{w}_2$ linear unabhängig seien. (Die anderen Fälle werden als Sonderfälle betrachtet.)

Gesucht wird, falls vorhanden, eine möglichst enge Intervallschnittstrecke der beiden LIEs in allen vier möglichen Parametrisierungen, also nach s, t, u und v. Oder anders gesprochen: Gesucht werden die Parametergebiete $\tilde{\mathbb{I}} \subseteq \mathbb{I}$ und $\tilde{\mathbb{J}} \subseteq \mathbb{J}$, die die Parameterwerte s,t,u und v der Intervallschnittstrecke möglichst eng einschließen.

Durch Gleichsetzen von (6.8) und (6.9) erhält man das unterbestimmte lineare Gleichungssystem,

$$(\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad -\boldsymbol{w}_1 \quad -\boldsymbol{w}_2) \begin{pmatrix} u \\ v \\ s \\ t \end{pmatrix} = \boldsymbol{r}; \quad \boldsymbol{r} \in \mathbb{r} := \mathbb{q} - \mathbb{p}; \tag{6.10}$$

aus dem die Intervallschnittgerade der beiden LIEs $\mathbb{L}_1(u,v),\ (u,v) \in \mathbb{I}$ und $\mathbb{L}_2(s,t),\ (s,t) \in \mathbb{J}$ berechnet werden kann.

6.2.1 Die direkte Methode

Eine direkte Methode zur Lösung des Problems ist die Folgende: Löst man alle möglichen regulären 3×3 Teilsysteme des Systems (6.10), erhält man für u, v, s und t jeweils einschließende Intervalle, deren Durchschnitt im Idealfall eine enge Einschließung der Ergebnismenge für den jeweiligen Parameter bildet.

6.2.2 Der "Parallelogramm-Algorithmus" für LIEs

Sind jeweils drei der Vektoren $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{w}_1, \boldsymbol{w}_2$ linear unabhängig, kann prinzipiell auch genau so vorgegangen werden wie in Abschnitt 6.1.
Die Gleichungen für s,t,u und v sind äquivalent zu den dort aufgestellten Gleichungen bis auf die Bestimmung von κ, λ, μ und ν, die in diesem Falle durch die Werte K, L, M und N ersetzt werden:

$$\begin{array}{lll} K := & |\boldsymbol{y}_2 & \mathbb{r} \quad -\boldsymbol{w}_2| \\ L := & |\boldsymbol{y}_1 & \mathbb{r} \quad -\boldsymbol{w}_2| \\ M := & |\boldsymbol{y}_2 & -\boldsymbol{w}_1 \quad \mathbb{r}| \\ N := & |\boldsymbol{y}_1 & -\boldsymbol{w}_1 \quad \mathbb{r}| \end{array}$$

Die Gleichungen (6.4)–(6.7) werden damit mit

$$\begin{array}{llll} S_1(u) = & \frac{1}{\alpha}(K + \beta\, u) & U_1(s) = & \frac{1}{\beta}(-K + \alpha\, s) \\ S_2(v) = & \frac{1}{\gamma}(L - \beta\, v) & U_2(t) = & \frac{1}{\delta}(M - \alpha\, t) \\ T_1(u) = & \frac{1}{\alpha}(M - \delta\, u) & V_1(s) = & \frac{1}{\beta}(L - \gamma\, s) \\ T_2(v) = & \frac{1}{\gamma}(N + \delta\, v) & V_2(t) = & \frac{1}{\delta}(-N + \gamma\, t) \end{array}$$

zu

$$\mathbb{g}_1(u) := \begin{pmatrix} S_1(u) \\ T_1(u) \end{pmatrix} = \frac{1}{\alpha}\left(\begin{pmatrix} K \\ M \end{pmatrix} + u \begin{pmatrix} \beta \\ -\delta \end{pmatrix}\right);\ u \in I_u \tag{6.11}$$

$$\mathbb{g}_2(v) := \begin{pmatrix} S_2(v) \\ T_2(v) \end{pmatrix} = \frac{1}{\gamma}\left(\begin{pmatrix} L \\ N \end{pmatrix} + v \begin{pmatrix} -\beta \\ \delta \end{pmatrix}\right);\ v \in I_v \tag{6.12}$$

und

$$\mathbb{h}_1(s) := \begin{pmatrix} U_1(s) \\ V_1(s) \end{pmatrix} = \frac{1}{\beta}\left(\begin{pmatrix} -K \\ L \end{pmatrix} + s \begin{pmatrix} \alpha \\ -\gamma \end{pmatrix}\right);\ s \in J_s \tag{6.13}$$

$$\mathbb{h}_2(t) := \begin{pmatrix} U_2(t) \\ V_2(t) \end{pmatrix} = \frac{1}{\delta}\left(\begin{pmatrix} M \\ -N \end{pmatrix} + t \begin{pmatrix} -\alpha \\ \gamma \end{pmatrix}\right);\ t \in J_t \tag{6.14}$$

Bemerkung:
Aus Angang A, Satz 1.1.5 folgt, daß $\mathbb{g}_i$ und $\mathbb{h}_j$, $i, j = 1, 2$ für definierte Werte von u, v, s und t jeweils eine optimale achsenparallele Einschließung der Lösungsmenge des zugehörigen Gleichungssystems angeben, falls die Determinanten K, L, M und N berechnet werden, indem man sie zuerst nach dem Komponenten des Intervallvektors $\mathbb{r}$ entwickelt.

Somit gilt auch für LIEs

Behauptung 5
Der Intervallvektor

$$\tilde{\mathbb{J}} = \mathbb{g}_1(I_u) \cap \mathbb{g}_2(I_v) \cap \mathbb{J}$$

ist für $\tilde{\mathbb{J}} \neq \emptyset$ eine Einschließung für die Parameterwerte des Schnittes von $\mathbb{L}_1$ mit der Intervallträgerebene von $\mathbb{L}_2$.

Für LIEs gilt allerdings nicht die Aussage, daß die in Behauptung 3 und 4 berechneten Intervalle $\tilde{\mathbb{I}}, \tilde{\tilde{\mathbb{J}}}$ die optimalen Einschließungen sind, da sich zum Einen durch den iterativen Charakter der Berechnung Überschätzungen ergeben und zum Anderen nur achsenparallele Einschließungen berechnet werden: Jeder Intervallpunkt des berechneten Schnittsegments schließt eine gewisse Menge von Intervallpunkten der LIEs ein, die vom Schnitt betroffen sind.

Um die Überschätzungen zu reduzieren, werden die berechneten Einschließungen in jedem Schritt mit vorangegangenen Einschließungen bzw. den ursprünglichen Parametergebieten geschnitten.

Die Behauptungen 3 und 4 werden deshalb wie folgt abgeändert:

Behauptung 6
Sei $\tilde{\mathbb{J}} \neq \emptyset$. Der Intervallvektor

$$\tilde{\mathbb{I}} = \mathbb{h}_1(\tilde{J}_s) \cap \mathbb{h}_2(\tilde{J}_t) \cap \mathbb{I}$$

ist eine Einschließung der Parameterwerte des Schnittes von $\mathbb{L}_2$ mit der Intervallträgerebene von $\mathbb{L}_1$.

Behauptung 7
Sei $\tilde{\mathbb{I}} \neq \emptyset$. Der Intervallvektor

$$\tilde{\tilde{\mathbb{J}}} = \mathbb{g}_1(\tilde{I}_u) \cap \mathbb{g}_2(\tilde{I}_v) \cap \tilde{\mathbb{J}}$$

ist eine Einschließung der Parameterwerte des Schnittes von $\mathbb{L}_1$ mit der Intervallträgerebene von $\mathbb{L}_2$.

Die Behauptungen 5 – 7 werden zu folgendem Satz zusammengefaßt:

Satz 6.2.1
Seien $\mathbb{L}_1$ und $\mathbb{L}_2$ die durch die Gleichungen 6.8 und 6.9 definierten LIEs und $\mathbb{g}_1, \mathbb{g}_2, \mathbb{h}_1, \mathbb{h}_2$ die Intervallgeraden mit den Gleichungen 6.11 – 6.14. Sind jeweils drei der Vektoren $\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{w}_1, \boldsymbol{w}_2$ linear unabhängig, so ist eine Einschließung der Intervallschnittstrecke von $\mathbb{L}_1$ und $\mathbb{L}_2$ durch jede der vier Gleichungen $\mathbb{g}_1(u),\ u \in \tilde{I}_u$, $\mathbb{g}_2(v),\ v \in \tilde{I}_v$, $\mathbb{h}_1(s)\ s \in \tilde{\tilde{J}}_s$ und $\mathbb{h}_2(t)\ t \in \tilde{\tilde{J}}_t$ gegeben, wobei

$$\tilde{\mathbb{J}} := \mathbb{g}_1(I_u) \cap \mathbb{g}_2(I_v) \cap \mathbb{J}$$

$$\tilde{\mathbb{I}} := \mathbb{h}_1(\tilde{J}_s) \cap \mathbb{h}_2(\tilde{J}_t) \cap \mathbb{I}$$

$$\tilde{\tilde{\mathbb{J}}} := \mathbb{g}_1(\tilde{I}_u) \cap \mathbb{g}_2(\tilde{I}_v) \cap \tilde{\mathbb{J}}$$

Ist mindestens eines der Parametergebiete $\tilde{\mathbb{J}}, \tilde{\mathbb{I}}$ oder $\tilde{\tilde{\mathbb{J}}}$ leer, so schneiden sich die LIEs nicht.

6.2.3 Sonderfälle

Die in Abschnitt 6.1.3 behandelten Sonderfälle gelten in ähnlicher Weise auch für LIEs.

- Fall 1 ($\gamma = \delta = 0$,$\gamma = \beta = 0$, $\alpha = \beta = 0$ oder $\alpha = \delta = 0$) wird genauso behandelt wie bei Parallelogrammen.
- Fall 2 ($\alpha = \beta = \gamma = \delta = 0$ und mindestens zwei der Determinanten $\kappa, \lambda, \mu, \nu \neq 0$) und Fall 3 ($\alpha = \beta = \gamma = \delta = \kappa = \lambda = \mu = \nu = 0$) werden gemeinsam behandelt: Ist $\alpha = \beta = \gamma = \delta = 0$ werden achsenparallele Einschließungsquader berechnet (z.B. durch direkte Auswertung der Flächenbeschreibungen mit Intervallen oder affiner Arithmetik) mit denen ein einfacher Schnittest vorgenommen wird.

6.3 Eigenschaften der Intervallschnittstrecken $\mathbb{g}_1, \mathbb{g}_2, \mathbb{h}_1$ und $\mathbb{h}_2$

Seien $\mathbb{g}_1(u), \mathbb{g}_2(v), \mathbb{h}_1(s)$ und $\mathbb{h}_2(t)$ die in Satz 6.2.1 beschriebenen Intervallstrecken.

6.3.1 Die Mittelpunktsgerade

Satz 6.3.1
$\mathbb{g}_1$ und $\mathbb{g}_2$ besitzen die gleiche Mittelpunktsgerade. Analoges gilt für die Geraden $\mathbb{h}_1$ und $\mathbb{h}_2$.

Beweis (für $\mathbb{g}_1$ und $\mathbb{g}_2$):
Seien $\boldsymbol{a} := \frac{1}{\alpha}(k_m, m_m)^T, \boldsymbol{b} := \frac{1}{\gamma}(l_m, n_m)^T$ und $\boldsymbol{z} := (\beta, -\delta)^T$ mit

$$\begin{aligned} k_m &:= mid(\mathbb{r})(\boldsymbol{y}_2 \times \boldsymbol{w}_2) & m_m &:= -mid(\mathbb{r})(\boldsymbol{y}_2 \times \boldsymbol{w}_1) \\ l_m &:= mid(\mathbb{r})(\boldsymbol{y}_1 \times \boldsymbol{w}_2) & n_m &:= -mid(\mathbb{r})(\boldsymbol{y}_1 \times \boldsymbol{w}_1). \end{aligned}$$

Dann sind

$$\boldsymbol{g}_1^m(u) := mid(\mathbb{g}_1(u)) = \boldsymbol{a} + \tfrac{1}{\alpha}\, u\, \boldsymbol{z} \tag{6.15}$$
$$\boldsymbol{g}_2^m(v) := mid(\mathbb{g}_2(v)) = \boldsymbol{b} - \tfrac{1}{\gamma}\, v\, \boldsymbol{z} \tag{6.16}$$

die Mittelpunktsgeraden von $\mathbb{g}_1$ und $\mathbb{g}_2$ und es gilt

$$\boldsymbol{g}_1^m(\frac{\Delta}{\gamma}) \equiv \boldsymbol{g}_2^m(0) = \boldsymbol{b}$$

mit $\Delta := mid(\mathbb{r})\,(\boldsymbol{w}_1 \times \boldsymbol{w}_1)$. Der Punkt $\boldsymbol{g}_2^m(0)$ liegt also auf der Geraden $\boldsymbol{g}_1^m$.

Da die Richtungsvektoren $\frac{1}{\alpha}\boldsymbol{z}$ und $-\frac{1}{\gamma}\boldsymbol{z}$ der beiden Geraden ebenfalls linear abhängig sind folgt, daß die Gleichungen 6.15 und 6.16 die gleiche Gerade beschreiben.

Für $\mathbb{h}_1$ und $\mathbb{h}_2$ kann analog argumentiert werden. □

Bemerkung:
Obiger Satz macht nur eine Aussage über die Mittelpunktsgerade als Ganzes, d.h. die zwei Segmente $\boldsymbol{g}_1^m(u), u \in \tilde{I}_u$ und $\boldsymbol{g}_2^m(v), v \in \tilde{I}_v$ gehören zwar zur gleichen Gerade, sind aber nicht unbedingt gleich lang.

6.3.2 Einschließende Geraden

Die vier in Satz 6.2.1 berechneten Intervallstrecken können jeweils auch durch zwei gewöhnliche parallele Strecken dargestellt werden (siehe Abbildung 6.5).

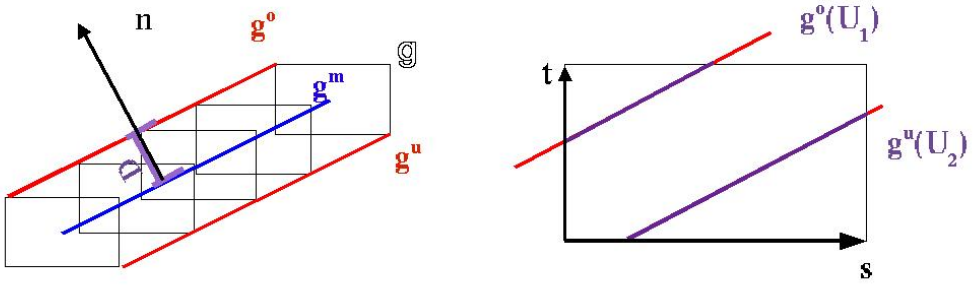


Abbildung 6.5: Berechnung der beiden einschließenden Geraden $\boldsymbol{g}_u$ und $\boldsymbol{g}_o$ der Intervallgeraden $\mathfrak{g}$. Das rechte Bild zeigt das Anpassen der Länge der berechneten Geradensegmente an das eigentliche Parametergebiet.

Das Problem läßt sich leicht allgemein formulieren:

Sei

$$\mathfrak{g}(u) := \mathfrak{a} + u\,\boldsymbol{b}, \quad u \in I \subset \mathbb{R} \tag{6.17}$$

eine beliebige Intervallgerade mit $\mathfrak{a} = (A_1, A_2)^T \in \mathbb{IR}^2$, $\boldsymbol{b} = (b_1, b_2)^T \neq (0,0)^T \in \mathbb{R}^2$.

Um die beiden einschließenden Geraden für $\mathfrak{g}$ zu berechnen, muß die Mittelpunktsgerade

$$\mathfrak{g}^m(u) := \boldsymbol{a} + u\,\boldsymbol{b}, \quad u \in I \subset \mathbb{R} \tag{6.18}$$

mit $\boldsymbol{a} := mid(\mathfrak{a})$ so in positive und negative Richtung ihrer Normalen $\boldsymbol{n} := \frac{1}{\|\boldsymbol{b}\|}(-b_2, b_1)^T$ verschoben werden, daß die Intervallgerade zwischen den beiden Geraden liegt.

Der Abstand eines Punktes $\boldsymbol{x}$ von der Geraden $\boldsymbol{g}^m$ ist gegeben durch $d(\boldsymbol{x}) := |\,\boldsymbol{n}(\boldsymbol{a} - \boldsymbol{x})\,|$.

Aufgrund der Punktsymmetrie der Eckpunkte der Intervallgeradenpunkte von $\mathfrak{g}$ zum entsprechenden Punkt der Mittelpunktskurve reicht es, den Abstand von zwei ihrer vier Eckpunkte zu $\boldsymbol{g}^m$ zu berechnen. Das Maximum dieser beiden Abstände

$$D := max\{d(\begin{pmatrix} min(A_1) \\ min(A_2) \end{pmatrix}), d(\begin{pmatrix} min(A_1) \\ max(A_2)) \end{pmatrix}\} \tag{6.19}$$

ist dann der Betrag des Abstands der beiden einschließenden Geraden von der Mittelpunktsgeraden.

Damit sind die Darstellungen für die beiden Geraden festgelegt:

$$\boldsymbol{g}^o(u) := \boldsymbol{a}^o + u\,\boldsymbol{b}; \quad u \in U \tag{6.20}$$

$$\boldsymbol{g}^u(u) := \boldsymbol{a}^u + u\,\boldsymbol{b}; \quad u \in U \tag{6.21}$$

mit $\boldsymbol{a}^o := \boldsymbol{a} + D\,\boldsymbol{n}$ und $\boldsymbol{a}^u := \boldsymbol{a} - D\,\boldsymbol{n}$.

Die tatsächliche Länge der Intervallgeraden ist größer als die der Mittelpunktkurve, da die Ausdehnung der Intervallpunkte am Anfang und Ende der Geraden mitgerechnet werden muß. Diese kann nicht größer sein als der Durchmesser der Intervallpunkte der Geraden $\|2\,rad(\mathbb{g})\|$. Ersetzt man das Parameterintervall U durch das Intervall

$$\tilde{U} := U \cap [-c, c]$$

mit $c := \frac{\|2\,rad(\mathbb{g})\|}{\|\boldsymbol{b}\|}$, bildet das durch die zwei Geraden $\boldsymbol{g}^o(u), u \in \tilde{U}$ und $\boldsymbol{g}^u(u), u \in \tilde{U}$ definierte Parallelogramm ein sichere Einschließung des Schnittsegments.

Sollen die Geraden $\boldsymbol{g}^o$ und $\boldsymbol{g}^u$ zudem in dem rechteckigen Gebiet $\mathbb{G} := G_1 \times G_2 \in \mathbb{IR}^2$ liegen, muß für beide Geraden das Parameterintervall U entsprechend angepaßt werden. Die beiden neuen Parameterintervalle $\in U^o$ und $\in U^u$ ergeben sich aus dem Schnitt der beiden Geraden mit $\mathbb{G}$:

$$U^o := \frac{1}{b_1}(G_1 - a_1^o) \cap \frac{1}{b_2}(G_2 - a_2^o) \cap \tilde{U} \tag{6.22}$$

$$U^u := \frac{1}{b_1}(G_1 - a_1^u) \cap \frac{1}{b_2}(G_2 - a_2^u) \cap \tilde{U} \tag{6.23}$$

für $\boldsymbol{b} \neq (0,0)^T$. Ist $b_1 = 0$ oder $b_2 = 0$ werden die entsprechenden Terme in den Gleichungen 6.22 und 6.23 durch $\tilde{U}$ ersetzt.

Die Ergebnisse werde in dem folgenden Satz zusammengefaßt, wobei die gleichen Bezeichnungen wie oben gewählt wurden.

Satz 6.3.2
Sei $\mathbb{g}(u), u \in U \subset \mathbb{R}$ eine Intervallgerade des $\mathbb{R}^2$, wie in Gleichung 6.17 definiert. Dann sind durch $\boldsymbol{g}^o(u), u \in U^o$ und $\boldsymbol{g}^u(u), u \in U^u$ zwei Geraden definiert, die $\mathbb{g}$ auf dem Gebiet $\mathbb{G} \subset \mathbb{R}^2$ einschließen.

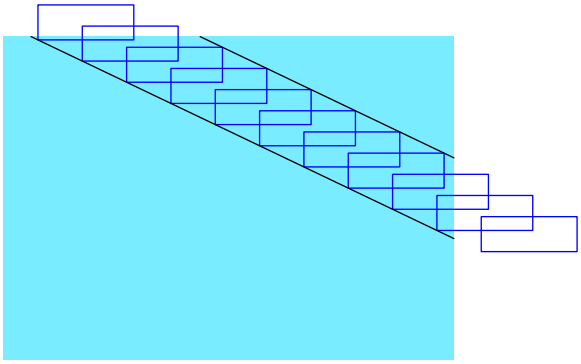


Abbildung 6.6: Die Intervallgerade $\mathbb{g}$ mit seinen beiden einschließenden Geraden im Parametergebiet $\mathbb{G}$.

Satz 6.3.2 ist die Basis für einen Algorithmus, der in Abschnitt 6.3.4 beschriebenen wird; mit ihm können Flächenkurven berechnet werden, die die Schnittkurve zweier Flächenstücke einschließen.

6.3.3 Einschließungseigenschaft

Abbildung 6.7 zeigt an Hand eines Beispiels, daß die Intervallgeraden $\mathbb{g}_1$ und $\mathbb{g}_2$ nicht unbedingt gleich breit sein müssen. Da beide Intervallgeraden laut Definition eine Einschließung des Schnittes der beiden LIEs $\mathbb{L}_1$ und $\mathbb{L}_2$ darstellen, und nach Satz 6.3.1 die gleiche Mittelpunktsgerade besitzen, liegt $\mathbb{L}_1 \cap \mathbb{L}_2$ sicher innerhalb der schmaleren Intervallgeraden.

Seien $\mathbb{g}_1, \mathbb{g}_2$ die in Satz 6.2.1 beschriebenen Intervallstrecken und $D(\mathbb{g}_i), i = 1, 2$ der in Gleichung 6.19 definierte Abstand der jeweiligen einschließenden Geraden von der Mittelpunktskurve.

Satz 6.3.3
Ist $D(\mathbb{g}_i) < D(\mathbb{g}_j);\ i \neq j;\ i, j \in \{1, 2\}$, *so gilt* $\mathbb{g}_i(\mathbb{R}) \subset \mathbb{g}_j(\mathbb{R})$ *und* $\mathbb{g}_i$ *ist eine engere Einschließung des Schnittes von* $\mathbb{L}_1$ *und* $\mathbb{L}_2$ *als* $\mathbb{g}_j$.

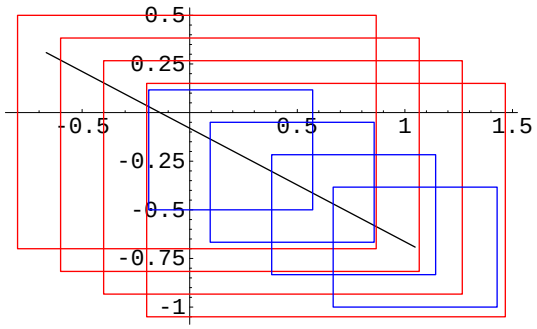


Abbildung 6.7: Illustration der Sätze 6.3.1 und 6.3.3: Das Bild zeigt eine diskrete Darstellung der Intervallgeraden $\mathbb{g}_1(u)$ (blau) und $\mathbb{g}_2(v)$ (rot) für $\boldsymbol{y}_1 := (0.2, 0.4, 0)^T$, $\boldsymbol{y}_2 := (1, 0.2, 0.1)^T$, $\boldsymbol{w}1 := (0.4, 0.5, 0.6)^T$, $\boldsymbol{w}_2 := (0, 0, 1)^T$ und $\mathbb{r} := ([0.1, 0.2], [-0.2, 0.1], [0, 0.2])^T$. Die gemeinsame Mittelpunktsgerade ist schwarz eingezeichnet. Die Gerade $\mathbb{g}_1$ hat den Durchmesser 0.916565 und die Gerade $\mathbb{g}_2$ den Durchmesser 1.87632.

6.3.4 Algorithmen

Datenstrukturen:

Line(2DVector Point, 2DVector Vect, Interval Param)
// Line(a, b, U) definiert die ebene parametrische Gerade $\boldsymbol{a} + u\,\boldsymbol{b},\ u \in U$

LinePair(Line g1, Line g2)
// Definiert ein Paar von Geraden $(g1, g2)$.

IntervalLine(2DIntervalVector IntPoint, 2DVector Vect, Interval Param)
// IntervalLine(a, b, U) definiert die ebene parametrische Gerade $\mathbb{a} + u\,\boldsymbol{b},\ u \in U$.

Algorithmus: Distance

Berechnet den Abstand D (6.19) der beiden einschließenden Geraden der Intervallgeraden $\mathbb{g}$ (6.17) von deren Mittelpunktsgeraden (6.18).

Input: Intervallgerade g.
Output: Distanz D.

```
double Distance (IntervalLine g)
{
  if (g.Vect = (0,0))       // g degeneriert
    D = 1/2 * norm(diam(g.IntPoint));
  else
    Compute D with formula (6.19);
  return D;
}
```

Algorithmus: EnclosingLines

Berechnet die Einschließenden Geraden $\boldsymbol{g}^o$ (6.20) und $\boldsymbol{g}^u$ (6.21) der Geraden $\mathfrak{g}$ (6.17) in dem Gebiet $\mathbb{G}$.

Input: Einzuschließende Intervallgerade g und Gebiet G.
Output: Paar einschließender Geraden.

```
void EnclosingLines (IntervalLine g, 2DIntervallVector G)
{
  if (g.Vect = (0,0))       // g degeneriert
   return;
  else {
    D = Distance(g);

    Compute vectors a_u, a_o like in equation (6.21) and (6.20);
    Compute parameter domains U^u and U^o like in equation (6.23) and (6.22);

    IntervalLine g1(a_u, b, U^u);
    IntervalLine g2(a_o , b, U^o);

    return LinePair(g1,g2);
  }
}
```

Algorithmus: ChooseThighterLine

Liefert die schmalere zweier Intervallgeraden $\mathfrak{g}_1$ und $\mathfrak{g}_2$.

Input: Intervallgeraden $\mathfrak{g}_1$ und $\mathfrak{g}_2$.
Output: Schmalere der beiden Geraden.

```
IntervalLine ChooseThighterLine (IntervalLine g1, IntervalLine g2)
{
  D1 = Distance(g1);
  D2 = Distance(g2);
  if (D1 < D2)
    return g1;
  else
    return g2;
}
```

6.4 Schnittest und Parametergebietsreduzierung für Flächenpatches

Sollen zwei Flächenpatches auf einen möglichen Schnitt untersucht werden, kann das mit Hilfe des Schnittestes für die beiden die Patches einschließenden LIEs getan werden. Schneiden sich die beiden LIEs nicht, d.h. ist mindestens eines der in Satz 6.2.1 beschriebenen Teilparametergebiete $\tilde{\mathbb{I}}$ und $\tilde{\tilde{\mathbb{J}}}$ leer, können sich auch die beiden Patches nicht schneiden. Da die Parametrisierungen der LIEs identisch sind mit den Parametrisierungen der Flächenpatches (Siehe Definition 5.1.1) schließen die im vorigen Abschnitt berechneten Teilparametergebiete $\tilde{\mathbb{I}}$ und $\tilde{\tilde{\mathbb{J}}}$ nicht nur die Parameterwerte des Schnittsegments der beiden LIEs, sondern auch diejenigen der Schnittkurve der beiden Patches ein.(Siehe Abbildung 6.8 und 6.9.)
Die Anwendung des Schnittestes für LIEs als Schnittest für Flächenpatches und als Methode zur Parametergebietsreduzierung wird in folgendem Satz zusammengefaßt, der eine direkte Folgerung aus Definition 5.1.1 und Satz 6.2.1 ist.

Satz 6.4.1
Seien $\boldsymbol{f}_1(u,v),(u,v) \in \mathbb{I}$ *und* $\boldsymbol{f}_2(s,t),(s,t) \in \mathbb{J}$ *zwei parametrische Flächenstücke des* $\mathbb{R}^3$ *und* $\mathbb{L}_1(u,v),(u,v) \in \mathbb{I}$ *und* $\mathbb{L}_2(s,t),(s,t) \in \mathbb{J}$ *die zugehörigen LIEs. Genügen* $\mathbb{L}_1$ *und* $\mathbb{L}_2$ *den Voraussetzungen des Satzes 6.2.1 und sind* $\tilde{\mathbb{I}}$ *und* $\tilde{\tilde{\mathbb{J}}}$ *die beim Schnittest von* $\mathbb{L}_1$ *und* $\mathbb{L}_2$ *berechneten Teilparametergebiete, so gilt:*

- *Ist* $\tilde{\mathbb{I}}$ *und/oder* $\tilde{\tilde{\mathbb{J}}}$ *leer, so schneiden sich* $\boldsymbol{f}_1$ *und* $\boldsymbol{f}_2$ *nicht.*
- *Sind* $\tilde{\mathbb{I}} \neq \emptyset$ *und* $\tilde{\tilde{\mathbb{J}}} \neq \emptyset$*, dann liegen die Parameterwerte der Schnittkurve der beiden Patches in den Teilparametergebieten* $\tilde{\mathbb{I}} \subseteq \mathbb{I}$ *und* $\tilde{\tilde{\mathbb{J}}} \subseteq \mathbb{J}$ *und die eigentliche Schnittkurve in den Teilpatches* $\boldsymbol{f}_1(u,v),(u,v) \in \tilde{\mathbb{I}}$ *und* $\boldsymbol{f}_2(s,t),(s,t) \in \tilde{\tilde{\mathbb{J}}}$.

Die Parametergebietsreduzierung ist also ein Nebenprodukt des Schnittestes für die beiden Flächenpatches.

Algorithmus: IntersectionTest

Input:
Das Flächenpaar AB mit den zugehörigen LIEs LIE(A) und LIE(B).

Output:
Explizit: TRUE, falls Schnittest erfolgreich war; FALSE, falls nicht.
Implizit: Falls Schnittest erfolgreich: die zurechtgeschnittenen Parametergebiete von AB; geändertes Intersection-Flag des Paares.

Boolean **IntersectionTest**(Flächenpaar AB mit LIE(A), LIE(B)){
  Berechne $\alpha, \beta, \gamma, \delta, K, L, M, N$
  If $(\alpha = \beta = \gamma = \delta = 0)$ // Sonderfall 1: LIE´s parallel oder gleich
    If Achsenparallele Bounding Boxen sich schneiden
      AB.inters = true;
      Return true; // Die Flächen schneiden sich eventuell.
    else
      AB.inters = false;
      Return false; // Die Flächen schneiden sich nicht.
  Berechne $\tilde{\mathbb{J}}$; // siehe Behauptung 5 und Sonderfallbehandlung 2+3
  If $(\tilde{\mathbb{J}} = \emptyset)$
    AB.inters = false;
    Return false; // Die Flächen schneiden sich nicht.
  Berechne $\tilde{\mathbb{I}}$; // siehe Behauptung 6 und Sonderfallbehandlung 2+3
  If $(\tilde{\mathbb{I}} = \emptyset)$
    AB.inters = false;
    Return false; // Die Flächen schneiden sich nicht.
  Berechne $\tilde{\tilde{\mathbb{J}}}$; // siehe Behauptung 7 und Sonderfallbehandlung 2+3
  If $(\tilde{\tilde{\mathbb{J}}} = \emptyset)$
    AB.inters = false;
    Return false; // Die Flächen schneiden sich nicht.
  A.domain = $\tilde{\mathbb{I}}$ //$\tilde{\mathbb{I}}$ definiert Parametergebiet des reduzierten Patches A
  B.domain = $\tilde{\tilde{\mathbb{J}}}$ //$\tilde{\tilde{\mathbb{J}}}$ definiert Parametergebiet des reduzierten Patches B
  Return true; //Die Flächen schneiden sich eventuell.
}

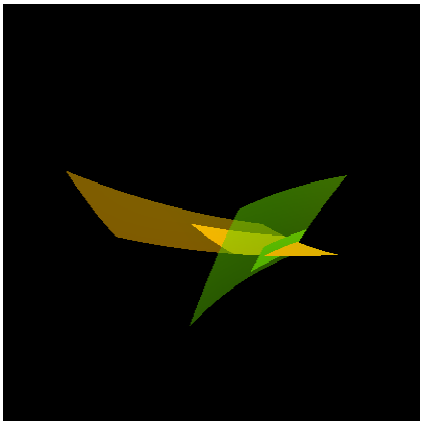


Abbildung 6.8: Zwei Flächenpatches vor und nach dem Schnittest.

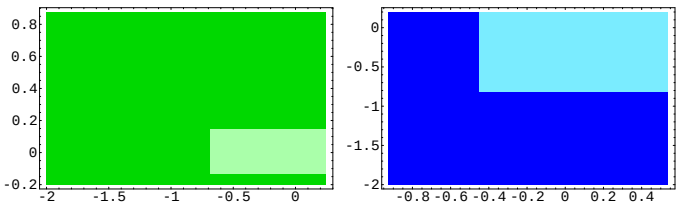


Abbildung 6.9: Die zu den Flächenpatches in Figur 6.8 gehörenden Parametergebiete. Die hellen Flächen sind die jeweiligen Parametergebiete nach dem Schnittest.

6.5 Orientierte Einschließungen der Schnittkurve in den Parametergebieten und im Objektraum

Im letzten Abschnitt wurde ein Algorithmus für einen Schnittest zweier Flächenstücke $\boldsymbol{f}_1(u,v),(u,v) \in \tilde{\mathbb{I}}$ und $\boldsymbol{f}_2(s,t),(s,t) \in \tilde{\tilde{\mathbb{J}}}$ angegeben, der gleichzeitig die Patches im Idealfall so reduziert, daß nur noch relevante Teile der Patches übrigbleiben. Die in Satz 6.4.1 beschriebenen Teilparametergebiete, die eine mögliche Schnittkurve sicher enthalten, sind achsenparallel und stellen damit in den meisten Fällen eine relative starke Überschätzung dar.

Einschließung der Schnittkurve im Parameterraum mit Intervallgeraden

Eine bessere Einschließung der Schnittkurve in den Parametergebieten bilden die in Satz 6.2.1 definierten Intervallstrecken $\mathbb{g}_1(u), u \in \tilde{I_u}$, $\mathbb{g}_2(v), v \in \tilde{I_v}$, $\mathbb{h}_1(s), s \in \tilde{\tilde{J}}_s$, und $\mathbb{h}_2(t), t \in \tilde{\tilde{J}}_t$. Aus Definition 5.1.1 folgt, daß sie nicht nur eine Einschließung des Schnittes der LIEs von $\boldsymbol{f}_1$ und $\boldsymbol{f}_2$ darstellen, sondern auch der Schnittkurve der beiden Flächenpatches $\boldsymbol{f}_1$ und $\boldsymbol{f}_2$.

Nach Satz 6.3.3 reicht es aus, in jedem Parametergebiet die jeweils schmalere Intervallstrecke auszuwählen, da sie den Durchschnitt der beiden Geraden darstellt. Im folgenden werden mit $\mathbb{g}(\tau), \tau \in \tilde{I_\tau}$ und $\mathbb{h}(\nu), \nu \in \tilde{\tilde{J}}_\nu$ die jeweils schmaleren Geraden aus $\{\mathbb{g}_1, \mathbb{g}_2\}$ und $\{\mathbb{h}_1, \mathbb{h}_2\}$ bezeichnet. Abbildung 6.10 zeigt $\mathbb{g}$ und $\mathbb{h}$ für ein konkretes Beispiel.

Sind $\{\boldsymbol{g}^u, \boldsymbol{g}^o\}$ bzw. $\{\boldsymbol{h}^u, \boldsymbol{h}^o\}$ die nach Abschnitt 6.3.2 bestimmten einschließenden Geradenpaare von $\mathbb{g}$ und $\mathbb{h}$, so bilden folglich auch diese eine Einschließung der Schnittkurve. (Siehe Abbildung 6.11.)

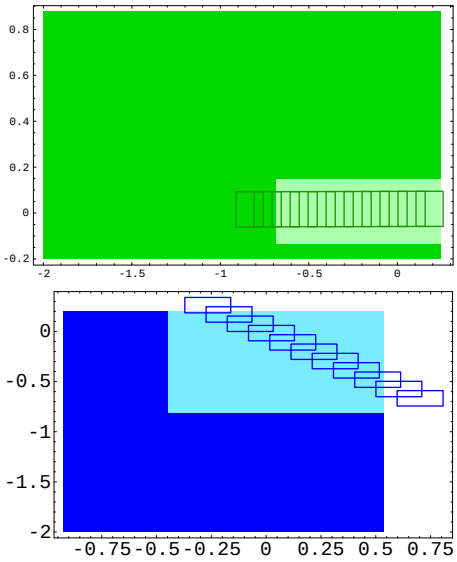


Abbildung 6.10: Die Parametergebiete aus Abbildung 6.9 mit einschließenden Intervallgeraden

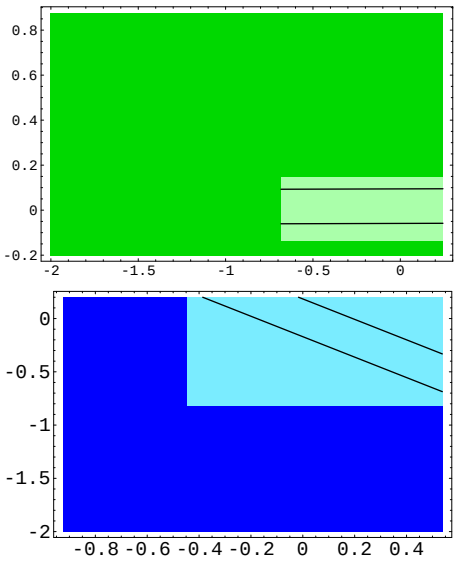


Abbildung 6.11: Die einschließenden Geradenpaaren der Intervallgeraden aus Abbildung 6.10

Einschließung der Schnittkurve im Objektraum mit Intervallflächenkurven

Setzt man die Formeln von $\mathbb{g}$ und $\mathbb{h}$ in die Parameterdarstellung der zugehörigen (Teil-) Flächenstücks $\boldsymbol{f}_1(u,v),(u,v) \in \tilde{\mathbb{I}}$ bzw. $\boldsymbol{f}_2(s,t),(s,t) \in \tilde{\tilde{\mathbb{J}}}$ ein, erhält man die zwei (unterschiedliche) Intervallflächenkurven,

$$\mathbb{c}_1(\tau) := \boldsymbol{f}_2(\mathbb{g}(\tau)), \quad \tau \in \tilde{I}_\tau \tag{6.24}$$

$$\mathbb{c}_2(\nu) := \boldsymbol{f}_1(\mathbb{h}(\nu)), \quad \nu \in \tilde{\tilde{J}}\nu \tag{6.25}$$

die jeweils die Schnittkurve räumlich umschließen. (Siehe Abbildung 6.12.)

Allerdings ist zu beachten, daß bei der Auswertung der Kurvengleichungen 6.24 und 6.25 Überschätzungen auftreten können.

Eine engere Einschließung im Objektraum kann gewonnen werden, indem man $\mathbb{c}_1$ und $\mathbb{c}_2$ miteinander schneidet: Da die tatsächliche Schnittkurve in beiden Intervallgeraden liegt, muß sie auch in deren Durchschnitt liegen.

Eine Einschließung der Schnittkurve in Form zweier Flächenkurven erhält man, wenn man die einschließenden Geradenpaare $\{\boldsymbol{g}^u, \boldsymbol{g}^o\}$ und $\{\boldsymbol{h}^u, \boldsymbol{h}^o\}$ jeweils in die zugehörige Flächengleichung einsetzt:

$$\boldsymbol{c}_1^u(\tau) := \boldsymbol{f}_2^u(\mathbb{g}(\tau)), \quad \tau \in \tilde{I}_\tau \tag{6.26}$$

$$\boldsymbol{c}_1^o(\tau) := \boldsymbol{f}_2^o(\mathbb{g}(\tau)), \quad \tau \in \tilde{I}_\tau \tag{6.27}$$

$$\boldsymbol{c}_2^u(\nu) := \boldsymbol{f}_1^u(\mathbb{h}(\nu)), \quad \nu \in \tilde{\tilde{J}}_\nu \tag{6.28}$$

$$\boldsymbol{c}_2^o(\nu) := \boldsymbol{f}_1^o(\mathbb{h}(\nu)), \quad \nu \in \tilde{\tilde{J}}_\nu \tag{6.29}$$

Damit werden einerseits die Überschätzungen der Kurven $\mathbb{c}_i, i = 1,2$ umgangen, andererseits erhält man dadurch aber nur "lokale" Einschließungen, die fest mit dem jeweiligen Patch verbunden sind. (Siehe Abbildung 6.13.)

Zusammenfassung

Satz 6.5.1
Seien $\boldsymbol{f}_1(u,v),(u,v) \in \mathbb{I}$ und $\boldsymbol{f}_2(s,t),(s,t) \in \mathbb{J}$ zwei parametrische Flächenstücke des $\mathbb{R}^3$. Weiter seien $\mathbb{g}(\tau),\tau \in \tilde{I}_\tau$ und $\mathbb{h}(\nu),\nu \in \tilde{\tilde{J}}_\nu$ die jeweils schmalere Intervallgerade aus den in Satz 6.2.1 definierten Paaren von Intervallschnittstrecken $\{\mathbb{g}_1,\mathbb{g}_2\}$ und $\{\mathbb{h}_1,\mathbb{h}_2\}$ der zu $\boldsymbol{f}_1$ und $\boldsymbol{f}_2$ gehörenden LIEs. $\{\boldsymbol{g}^u,\boldsymbol{g}^o\}$ und $\{\boldsymbol{h}^u,\boldsymbol{h}^o\}$ seien die einschließenden Geradenpaare von $\mathbb{g}$ und $\mathbb{h}$. $\mathbb{c}_i, i = 1,2$ und $\boldsymbol{c}_i^j, i = 1,2, j = u,o$ seien definiert wie in den Gleichungen (6.24)–(6.29),
Dann gilt:

- *Die Intervallgerade $\mathbb{g}$ und das Geradenpaar $\{\boldsymbol{g}^u,\boldsymbol{g}^o\}$ sind jeweils Einschließungen den der Schnittkurve entsprechenden Parameterwerten in $\mathbb{J}$. Die Intervallgerade $\mathbb{h}$ und das Geradenpaar $\{\boldsymbol{h}^u,\boldsymbol{h}^o\}$ sind jeweils Einschließungen den der Schnittkurve entsprechenden Parameterwerten $\boldsymbol{s}$ in $\mathbb{I}$.*

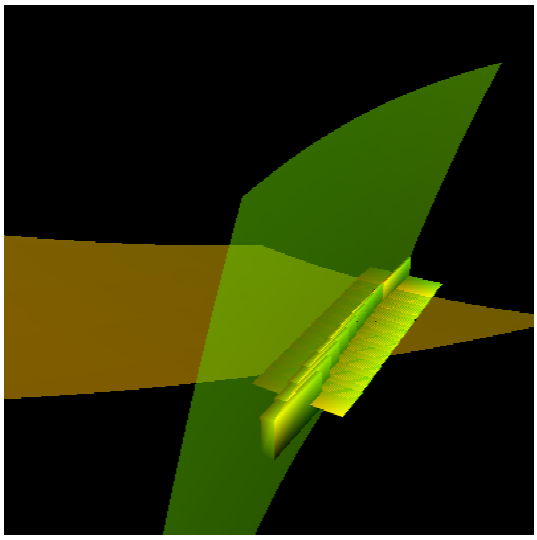


Abbildung 6.12: Die beiden Flächenpatches mit den den Intervallgeradensegmenten aus Abbildung 6.10 entsprechenden Intervallflächenkurven

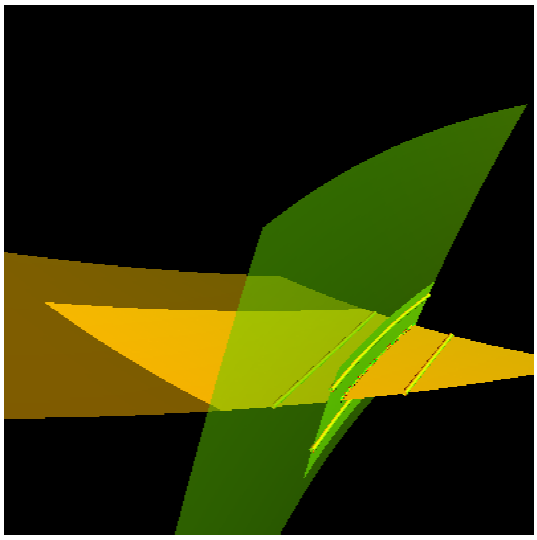


Abbildung 6.13: Die den Geradenpaaren in Abbildung 6.11 entsprechenden einschließenden Flächenkurven und die tatsächliche Schnittkurve

- *Die Intervallflächenkurven* $\mathbb{c}_1$, $\mathbb{c}_2$ *und* $\mathbb{c}_1 \cap \mathbb{c}_2$ *bilden jeweils eine Einschließung der Schnittkurve im Objektraum.*
- *Die Flächenkurvenpaare* $\{\boldsymbol{c}_1^u, \boldsymbol{c}_1^o\}$ *und* $\{\boldsymbol{c}_2^u, \boldsymbol{c}_2^o\}$ *bilden auf der Fläche* $\boldsymbol{f}_2$ *bzw.* $\boldsymbol{f}_1$ *eine Einschließung der Schnittkurve.*

6.6 Vergleich von Schnittests für LIEs mit Schnittests für andere Einschließungskörper

Grundsätzlich ist festzustellen, daß es keinen Schnittest für herkömmlichen Einschließungskörper wie achsenparallele und orientierte Bounding Boxen, Parallelepipede oder konvexe Polyeder gibt, der den Schnittest mit einer Reduzierung des Parametergebietes kombiniert. Ein direkter Vergleich der Algorithmen ist also nicht möglich, da die Parametergebietsreduzierung einen sehr großen Einfluß auf die Gesamtlaufzeit des Schnittalgorithmus hat.

Betrachtet man den Schnittest unabhängig von einer möglichen Parametergebietsreduzierung, sind achsenparallele Einschließungen sicher die Hüllkörper, die am einfachsten zu behandeln sind. Werden nämlich beide als Intervallvektoren $\mathbb{a}, \mathbb{b}$ dargestellt, erhält man den Schnitt beider Boxen einfach als Durchschnitt der Intervallboxen $\mathbb{a} \cap \mathbb{b}$.

Für orientierte Bounding Boxen sind in der Literatur grundsätzlich vier verschiedene Ansätze für einen Schnittest zu finden, die sich problemlos auf den Schnitt zweier Parallelepipede übertragen lassen:

1. Direkter Ansatz: Berechnung der Durchstoßpunkte der Kanten der einen Box bezüglich der Seitenflächen der anderen Box. Schneidet keine der Kanten des einen Epipeds eine Seitenfläche des anderen, so sind die Epipede disjunkt. Sonderfall: Die Epipede liegen ineinander. Im schlechtesten Fall müssen je nach Algorithmus $12 * 12$ bzw. $6 * 12$ Durchstoßpunkte berechnet werden. (Siehe z.B. [Huber '99] bzw. [Gottschalk et al. '96].)

2. Lösung eines linearen Optimierungsproblems. Das Problem wird als Schnitt von 12 Halbebenen formuliert. Eine Beschreibung des Algorithmus ist in [Preparata, Shamos '85] zu finden.

3. Bestimmung der beiden am nächsten beieinander liegenden Punkte der beiden Epipede mit Hilfe eines iterativen Algorithmus (closest feature algorithms). In [Bergen '99] wird hierfür der Gilbert-Johnsen-Keerthi Algorithmus verwendet und gezeigt, daß dieser schneller ist als der Lin-Canny-Algorithmus [Lin, Canny '91].

4. Berechnung einer separierenden Achse ([Gottschalk et al. '96], [Huber '99]). Der Algorithmus basiert auf der Tatsache, daß zwei sich nicht schneidende konvexe Polytope im $\mathbb{E}^3$ immer durch eine Ebene getrennt werden können, die parallel zu je einer der Seitenflächen oder zu je einer der Kanten der beiden Polytope ist. Für Parallelepipede ergeben sich daraus 15 mögliche separierende Achsen. Projiziert man die beiden Parallelepipede auf diese Achsen und schneiden sich die beiden Projektionen auf einer der Achsen nicht, so schneiden sich auch die beiden Parallelepipede nicht.

In [Gottschalk et al. '96] werden die vier Algorithmen diskutiert und getestet, wobei in Experimenten Algorithmus 4 mehr als 30 mal schneller Ergebnisse lieferte, als Algorithmus 2 und mehr als 10 mal schneller war als Algorithmus 3.
Ein Grund hierfür ist sicher, daß der Algorithmus in den meisten Fällen nicht alle 15 möglichen Achsen testen muß, da es reicht eine separierende Achse zu finden. Der Algorithmus kann also abbrechen, sobald eine solche gefunden wurde.

Der Schnittalgorithmus für LIEs verhält sich ähnlich: ist eines der berechneten Teilparametergebiete leer, bricht der Algorithmus ab, da sich die beiden LIEs in diesem Fall sicher nicht schneiden. Vergleicht man direkt die Anzahl der Operationen des Algorithmus 4 mit den für einen LIE-Schnittest nötigen Schritte ergibt sich folgende Tabelle (die Werte für Algorithmus 4 sind [Gottschalk et al. '96] entnommen):

Operation	**Algorithmus 4**	**LIE Schnittest**
Addition / Subtraktion	60	8 (double) + 12(interval)
Multiplikation	81	12 (double) + 16 (interval)
Division	/	4
Betrag	24	/
Intervallschnitte	/	6
Vergleiche	15	3 (interval)

Alle Werte verstehen sich unter der Voraussetzung, daß keine Sonderfälle auftreten. Die Berechnung der acht Determinanten im LIE Schnittest wurden so optimiert, daß Mehrfachbe-

rechnungen soweit wie möglich vermieden werden.

Um beide Algorithmen besser vergleichen zu können, werden die Intervalloperationen folgendermaßen in gewöhnliche Gleitpunktoperationen umgewandelt:
Eine Intervalladdition /-Subtraktion entspricht 2 gewöhnlichen Additionen/Subtraktionen, eine Intervallmultiplikation 4 gewöhnlichen Multiplikationen und 6 Vergleichen. Der Schnitt zweier Intervalle entspricht vier gewöhnlichen Vergleichen, die drei Intervallvergleiche sind ein Test, ob die Intervalle leer sind oder nicht und entsprechen einem gewöhnlichen Vergleich (Siehe Anhang A). Damit kann obige Tabelle folgendermaßen angepaßt werden:

Operation	**Algorithmus 4**	**LIE Schnittest**
Addition / Subtraktion	60	32
Multiplikation	81	76
Division	/	4
Betrag	24	/
Vergleiche	15	27

Zusammenfassend kann also Folgendes festgehalten werden:

- Ein Schnittest für LIEs ist nicht aufwendiger als ein guter Schnittest für Parallelepipede.
- Der Schnittest für LIEs reduziert im Gegensatz zu dem Algorithmen für einschließende Parallelepipede das Parametergebiet der eingeschlossenen Fläche, das bei Unterteilungsalgorithmen die Grundlage für weitere Berechnungen bildet. Die Rechenzeit für den gesamten Algorithmus wird dadurch erheblich reduziert.

7

Unterteilungsstrategien und Abbruchkriterien

7.1 Unterteilungsstrategien

In der Literatur finden sich unterschiedliche Vorgehensweisen, wie im Unterteilungsalgorithmus die einzelnen Flächenstücke für den nächsten Rekursionsschritt unterteilt werden. In den meisten Fällen wird das Parametergebiet gleichmäßig in zwei ([Huber '99]), vier ([de Figueiredo '96], [Gleicher, Kass '92], [Filip et al. '86]) oder n ([Bürger, Schaback '93]) Teilgebiete geteilt. Einige wenige Autoren schlagen eine adaptive Strategie vor, die sich nach der Topologie des (Teil-)Patches oder des Parametergebiets richtet ([Koparkar '91]).

Koparkar diskutiert in [Koparkar '91] die verschiedenen Unterteilungsstrategien mit dem Hintergrund, daß die Flachheit der beiden Patches als Abbruchkriterium herangezogen wird:

- Es ist nicht immer nötig, beide Patches zu unterteilen: Ist beispielsweise eines der Patches eben, bringt eine weitere Unterteilung keine Vereinfachung des Patches mit sich.
- Ein Patch muß nicht unbedingt entlang beider Parameter unterteilt werden, falls die Schnittgeradensegmente linearer Approximationen der Patches als Endergebnis ausgegeben werden, und nicht die berechneten Teilparametergebiete. Ist das Patch beispielsweise in einem Parameter linear (wie z.B. bei einem Zylinder) wird das Patch durch Unterteilung entlang dieses Parameters nicht flacher.
- Die Unterteilung im Mittelpunkt des Parametergebiets erhält die Symmetrie in der Größe der Teilparametergebiete, die allerding nicht unbedingt einer Symmetrie in der Größe der beiden Teilflächen entspricht.

Auf Basis dieser Überlegungen entwickelt Koparkar einen relativ komplexen Unterteilungsmechanismus, der Linearität und Ausdehnung der Flächenstücke in den Entscheidungsprozeß, wie und wo unterteilt werden soll, mit einbezieht.

Bei der Verwendung von LIEs als Hüllkörper werden bei jedem Schnittest und der dadurch durchgeführten Parametergebietsreduzierung, Größe und Form beider Patches der Topologie der Schnittkurve angepaßt. Der Durchmesser der Intervallpunkte der LIEs gibt schnell und zuverlässig Auskunft über die Flachheit des Flächenpatches.

Bezieht man diese Eigenschaft und das spezielle Verhalten der LIEs während des Schnittests mit in den Entwurf eines sinnvollen Unterteilungsschemas ein, können sehr viel bessere Ergebnisse bezüglich der Geschwindigkeit erzielt werden, als bei einer uniformen Unterteilung in vier oder zwei Teilflächen.

Die **doppelt-adaptive Unterteilung** für LIE basierte Unterteilungsalgorithmen funktioniert nach folgendem Schema:

1. *Unterteile Fläche A in zwei Teilpatches, wenn die Intervallpunkte der einschließenden LIE einen größeren Durchmesser haben, als die der Fläche B. Sonst unterteile Fläche B.*

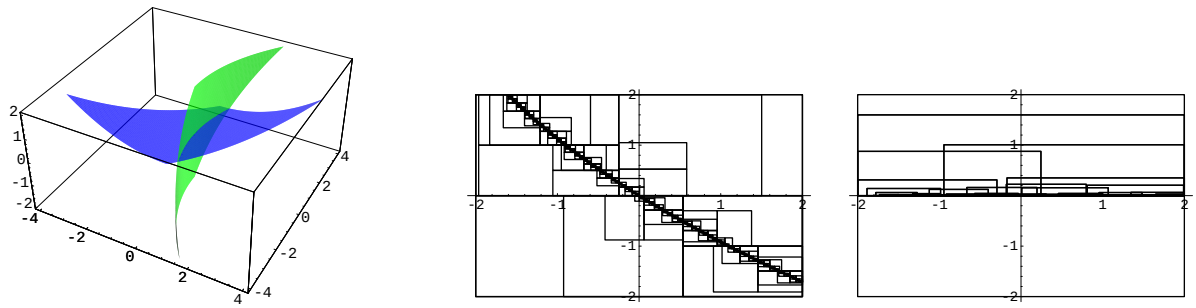
2. *Unterteile stets in der Mitte der längeren Seite des Parametergebiets.*

Diese Art der Unterteilung, kombiniert mit der Parametergebietsreduzierung, ist sehr effektiv, da sie sich sehr gut an die Form der Schnittkurve anpaßt. Tests zeigen eine Geschwindigkeitssteigerung bis zu 100% im Vergleich zu einer Unterteilung des Parametergebiets der jeweils flacheren Fläche in vier gleiche Teile. (Siehe Kapitel 9)

Bei einfacheren Flächenstücken kann es durchaus vorkommen, daß nur eine der beiden Flächen unterteilt wird und sich die Teilpatches der anderen Fläche nur durch die Parametergebietsreduzierung ergeben, wie das folgende Beispiel zeigt.

Beispiel

In diesem Beispiel wird nur das Parametergebiet der blauen Fläche (links) tatsächlich unterteilt (146 mal), während die Unterteilung des Parametergebietes der grünen Fläche nur durch die beim Schnittest der LIEs vorgenommene Parametergebietsreduzierung zustande kommt. (Eine weitere Diskussion des Beispiels findet sich in Kapitel 9).



7.2 Abbruchkriterien

Grundsätzlich muß zwischen Abbruchkriterien im Objekt- und im Parameterraum unterschieden werden. Abbruchkriterien im Objektraum betrachten Eigenschaften der Flächenpatches selbst, wie z.B. Größe und Krümmung der Patches ([Huber '99],[Bürger, Schaback '93], [Gleicher, Kass '92],[Filip et al. '86]) oder der Qualität einer Einschließung der Schnittkurve im Objektraum [Huber '99]. Abbruchkriterien im Parameterraum sind oft einfacher zu berechnen als Abbruchkriterien im Objektraum. Sie stellen eine Einschließung möglicher Parameterwerte der Schnittkurve dar, sagen aber wenig über die Qualität der Einschließung im Objektraum aus.

Abbruchkriterien im Parameterraum

- **Flächeninhalt/Größe der Teilparametergebiete:** Leicht zu berechnen ([Huber '99]), gibt aber im Prinzip keine Auskunft über Größe und Form der zugehörigen Teilflächen. In [Huber '99] wird solange unterteilt, bis die Teilparametergebiete relativ zum ursprünglichen Parametergebiet als Punkte betrachtet werden können, die dann als Ergebnis ausgegeben werden.

- **Breite der einschließenden Intervallgeraden:** Ein Abbruchkriterium, das nur im Zusammenhang mit LIEs angewandt werden kann. Hier wird eine obere Schranke für den in Kapitel 6 Abschnitt 6.3.2 Gleichung 6.19 definierten Radius der Intervallgeraden angegeben. Diese Abbruchkriterium gibt nicht nur Auskunft über die Qualität der Einschließung im Parameterraum wieder, sondern spiegelt indirekt auch die Flachheit und Lage der beiden geschnitten LIEs wieder: Je flacher die beiden LIEs sind und je senkrechter sie aufeinander stehen, umso schmaler ist die Intervallgerade.

Abbruchkriterien im Objektraum

Die Wahl des Abbruchkriteriums bei einem Unterteilungsalgorithmus hängt auch stark davon ab, ob der Unterteilung ein Verfeinerungsschritt folgen soll und wenn ja, welcher Art dieser sein wird.

- **Krümmungen der Flächen:** Um nach der Unterteilung eine lineare Approximation der Kurve zu berechnen, müssen beide Patches linear approximiert werden, d.h. je "flacher" beide Patches sind, desto besser ist auch ihre lineare Approximation. Ein sinnvolles Abbruchkriterium für die Unterteilung ist in diesem Fall eine obere Schranke für die Abweichung des Flächenpatches von einer approximierenden Ebene, die auf unterschiedliche Arten bestimmt werden kann.

  - In [Filip et al. '86] wird ein Kriterium zur Beurteilung der Flachheit der Teilflächen über die Berechnung globaler oberer Schranken der ersten bzw. zweiten Ableitungen und die Anzahl der Unterteilungen entwickelt.
  - Werden orientierte Hüllkörper verwendet, gibt die Dicke der Hüllkörper Auskunft über die Krümmung der Fläche ([Huber '99],[Bürger, Schaback '93]).

  - Die Komplanarität der Eckpunkte ([Koparkar '91]), Abweichungen von Tangenten ([Houghton et al. '85]) oder Normalen ([Barnhill, Kersey '90]) werden ebenfalls als Kriterium für die Flachheit der Fläche verwendet .
  - Werden LIEs als Hüllkörper verwendet, bietet sich zur Beurteilung der Flachheit der Fläche eine Untersuchung der Intervallpunkte der einschließenden LIEs an.

- **Ausdehnung im Objektraum:** In [Gleicher, Kass '92] wird eine obere Schranke für den Durchmesser der achsenparallelen Bounding Box als Abbruchkriterium verwendet. Eine Verfeinerung der Ergebnisse wird bei diesem Algorithmus nicht vorgenommen.

- **Existenz geschlossener Zweige der Schnittkurve:** Bei hybriden Algorithmen, die einen Unterteilungsalgorithmus mit einem Verfolgungsalgorithmus kombinieren, wird so lange unterteilt, bis keine geschlossenen Zweige bei Schnitten der Teilpatches auftreten. Algorithmen zur Entscheidung, ob geschlossenen Kurven existieren, wurden ausführlich in Kapitel 3 vorgestellt.

8

Implementierung des Schnittalgorithmus

8.1 Die implementierten Varianten des Schnittalgorithmus

In Kapitel 4 Abschnitt 4.1 ist die allgemeine Form des Unterteilungsalgorithmus vorgestellt worden, für dessen Teilschritte in den vorangegangenen Kapiteln verschiedene neue Strategien entwickelt wurden:

- Berechnung von Hüllkörpern (Kapitel 5)
- Schnittest und Parametergebietsreduzierung (Kapitel 6)
- Unterteilungsstrategie (Kapitel 7)
- Abbruchkriterium (Kapitel 7)

Für Testzwecke wurden neben den neuen Methoden auch herkömmliche achsenparallele Bounding Boxen, gewöhnliche adaptive und uniforme Unterteilungsstrategien, die ILSS-Schnittmethode, ein hybrider Ansatz und unterschiedliche Abbruchkriterien in das System integriert.

Liste realisierter Teischritte:

Hüllkörper [BOUND]:

BOX Intervallbox $\mathbb{b}$. Berechnet durch direkte Auswertung der Parameterdarstellung $\boldsymbol{f}(u,v), (u,v) \in I_u \times I_v)$ mit den Parameterintervallen: $\mathbb{b} := \boldsymbol{f}(I_u, I_v)$.

TMLIE Lineare Intervallabschätzung als Taylor Modell. Siehe Kapitel 5.

AALIE Lineare Intervallabschätzung auf Basis affiner Arithmetik. Siehe Kapitel 5.

BOXTMLIE Verwendung einer Kombination von achsenparallelen Bounding Boxen und TMLIEs in einem hybriden Algorithmus.

BOXAALIE Wie bei BOXTMLIE aber mit AALIEs.

Schnittest [TEST]:

BOX Schnittest für achsenparallele Intervallboxen. Der Algorithmus wurde als Schnitt zweier Intervallvektoren implementiert.

ILSS Schnittest für LIEs basierend auf dem *Interval Linear System Solver* (ILSS) [Knüppel '93b] mit kombinierter Parametergebietsreduzierung. Dieser Algorithmus berechnet den Schnitt zweier LIEs nach dem in Kapitel 6 Abschnitt 6.2.1 vorgestellten Verfahren.

LIE Algorithmus **IntersectionTest** aus Kapitel 6 Abschnitt 6.4 mit kombinierter Parametergebietsreduzierung

HYBRID Hybrider Schnittest. Nur wenn sich die achsenparallelen Hüllen schneiden, wird der Algorithmus **IntersectionTest** ausgeführt.

Abbruchkriterium [TERM]:

BOX Größerer Durchmesser der beiden einhüllenden achsenparallelen Boxen.

LIE Größerer Durchmesser der Intervallpunkte der beiden LIEs.

DOMAIN Größerer Durchmesser der beiden Parametergebiete.

LINE Größerer Radius der Schnittgeraden der LIEs: Es wird zunächst solange unterteilt, bis der größerer Durchmesser der Intervallpunkte der beiden LIEs die Schranke erreicht hat. Dann erst wird in jedem Schritt der Radius der breiteren Schnittgerade als Abbruchkriterium betrachtet. Dazu wurden die drei Algorithmen aus Kapitel 6 Abschnitt 6.3.4 verwendet.

Ergebnis [RESULT]:

DOMAIN Die rechteckigen Parametergebiete sich eventuell schneidender Patches.

LINE Die den Schnitt einschließenden Geraden in den Parametergebieten und die dazu gehörenden rechteckigen Parametergebiete.

Unterteilung [SUBDIV]:

BOX Das Parametergebiet des Patches, dessen achsenparallele Box den größeren Durchmesser besitzt, wird uniform in vier Teilgebiete unterteilt.

UNI Die Parametergebiete der Patches werden abwechselnd jeweils in vier gleichgroße Teilgebiete unterteilt.

ADAPT Das Patch, dessen LIE Intervallpunkte den größeren Durchmesser besitzen, wird uniform in vier Teilpatches unterteilt.

DOUBLE Das Parametergebiet des Patches, dessen LIE Intervallpunkte den größeren Durchmesser besitzen, wird in der Mitte der längeren Seite des Parametergebiets in zwei Teile unterteilt. Siehe *doppelt adaptive Unterteilung* in Kapitel 7 Abschnitt 7.1.

Mögliche Kombinationen:

	TEST				**TERM**			
BOUND	BOX	ILSS	LIE	HYBRID	BOX	LIE	DOMAIN	LINE
BOX	X/O				X	O		
TMLIE/AALIE		X	X	X		X	X	X
BOXTMLIE				X		X	X	X
BOXAALIE				X		X	X	X

	RESULT		**SUBDIV**			
BOUND	DOMAIN	LINE	BOX	UNI	ADAPT	DOUBLE
BOX	X	O	X / O			
TMLIE/AALIE	X	X		X	X	X
BOXTMLIE	X	X		X	X	X
BOXAALIE	X	X		X	X	X

Gibt es mehrere Kreuze in einer Kategorie, darf immer nur eines davon ausgewählt werden. Die Algorithmen für LIEs sind beliebig kombinierbar, während bei der Verwendung achsenparalleler Boxen die Algorithmen so ausgewählt werden müssen, daß nur Kreuze oder nur Kreise vorkommen.

8.2 Technische Details

Alle Algorithmen wurden in C++[1] implementiert. Die Profil/BIAS-**Intervallbibliothek** von Olaf Knüppel ([Knüppel '93a], [Knüppel '93b], [Knüppel '93c]) stellte die Basis für die nötige Intervallarithmetik. Das Paket war in seiner ursprünglichen Form mit den verwendeten C/C++ Bibliotheken nicht kompatibel und wurde entsprechend angepaßt.Profil/BIAS stellt neben der Grundarithmetik auch Datentypen und Operationen für (Intervall-)Vektoren und Matrizen zur Verfügung, sowie eine Implementierung des Interval System Solvers (ILSS) und Funktionen zur Berechnung von Ableitungen erster und zweiter Ordnung mittels automatischer Differenzierung. Als Basis für die **affine Arithmetik** wurde das Paket von van Iwaarden [Iwaarden, Stolfi '97] verwendet. Zur vorhandenen Implementierung wurden Mechanismen hinzugefügt, die die Fehlersymbole aufzeichnen, die direkt mit den Affinformen der Eingangsparameter einer Berechnung

[1]Daten: glibc6.0, gcc Version 2.95.2 unter Suse LINUX 7.2. PC-Daten: 800 Mhz Athlon Prozessor, 512 MB RAM

zusammenhängen. Die Erweiterung macht eine direkte Berechnung der LIEs möglich. Zusätzlich wurde eine Klasse implementiert, die die nötigen Operationen für vektorielle Affinformen zur Verfügung stellt.

Datenstrukturen

Zentrale Datenstruktur sind Paare von Flächenpatches: Jede Fläche trägt Informationen zum Flächentyp, dem Parametergebiet, den Daten der zugehörigen LIE und bei Bedarf auch einer achsenparallelen Hüllbox. Ein Flächenpaar besteht aus zwei Flächen und der Information, ob sich die Flächen eventuell schneiden oder nicht.

Die jeweiligen Flächenpaare werden in jedem Rekursionsschritt des LIE-Schnittalgorithmus beim Schnittest aneinander angepaßt. Das heißt, auch wenn nur eine der Flächen im vorangegangenen Schritt unterteilt wurde, wird das gemeinsame zweite Patch beim Schnittest im Idealfall so reduziert, daß möglichst viele der Teilgebiete wegfallen, die sicher keinen Schnitt enthalten.

Nach den Schnittests ergeben sich also aus ein und demselben Flächenstück zwei völlig unterschiedliche Flächen, die an ihren jeweiligen "Partner" angepaßt wurden. Der Aufbau einer separaten Baumstruktur um Mehrfachberechnungen zu verhindern (siehe z.B. [Huber '99]), wird dadurch überflüssig. Zum einen können durch die Anpassung in jedem Schritt die Daten für andere Teilpatches nicht mehr verwendet werden, zum anderen wird durch die Rekursion automatisch ein Baum von Flächenpaaren auf- und abgebaut, der jeweils nur die Informationen enthält, die für den aktuellen und die nachfolgenden Iterationsschritte benötigt werden (siehe Figur 8.1).

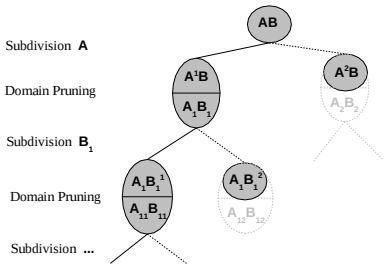


Abbildung 8.1: Beispiel für die Aufteilung und Reduzierung der Flächen während des Schnittalgorithmus.

Testprogramme und Benutzeroberfläche

Das Programm selbst existiert in zwei Versionen, die sich nur in der Berechnung der LIEs unterscheiden: Eine Version verwendet Taylor-Modell basierte LIEs, die andere Version LIEs, die mit Hilfe affiner Arithmetik berechnet wurden.

Für elf Testflächen wurden Parameterdarstellungen (bei der Taylor-Modell-Version inklusive Ableitungen) direkt in das System integriert, die beliebig kombinierbar und deren Parametergebiete frei definierbar sind. Die Ergebnisse der Berechnungen werden in Dateien abgelegt.

Es wurde eine graphische Benutzeroberfläche[2] implementiert, die für beide Programmversionen verwendet wird und in der Abbildung 8.2 auszugsweise dargestellt wird.

[2] Verwendet wurde hierfür das *Qt-Toolkit* Version 2.2 (http://www.trolltech.com/)

Visualisierung der Ergebnisse

Die Visualisierung der Ergebnisse wurde mit Hilfe speziell programmierter Tools mit Mathematica[3] und MathGL3D[4] durchgeführt.

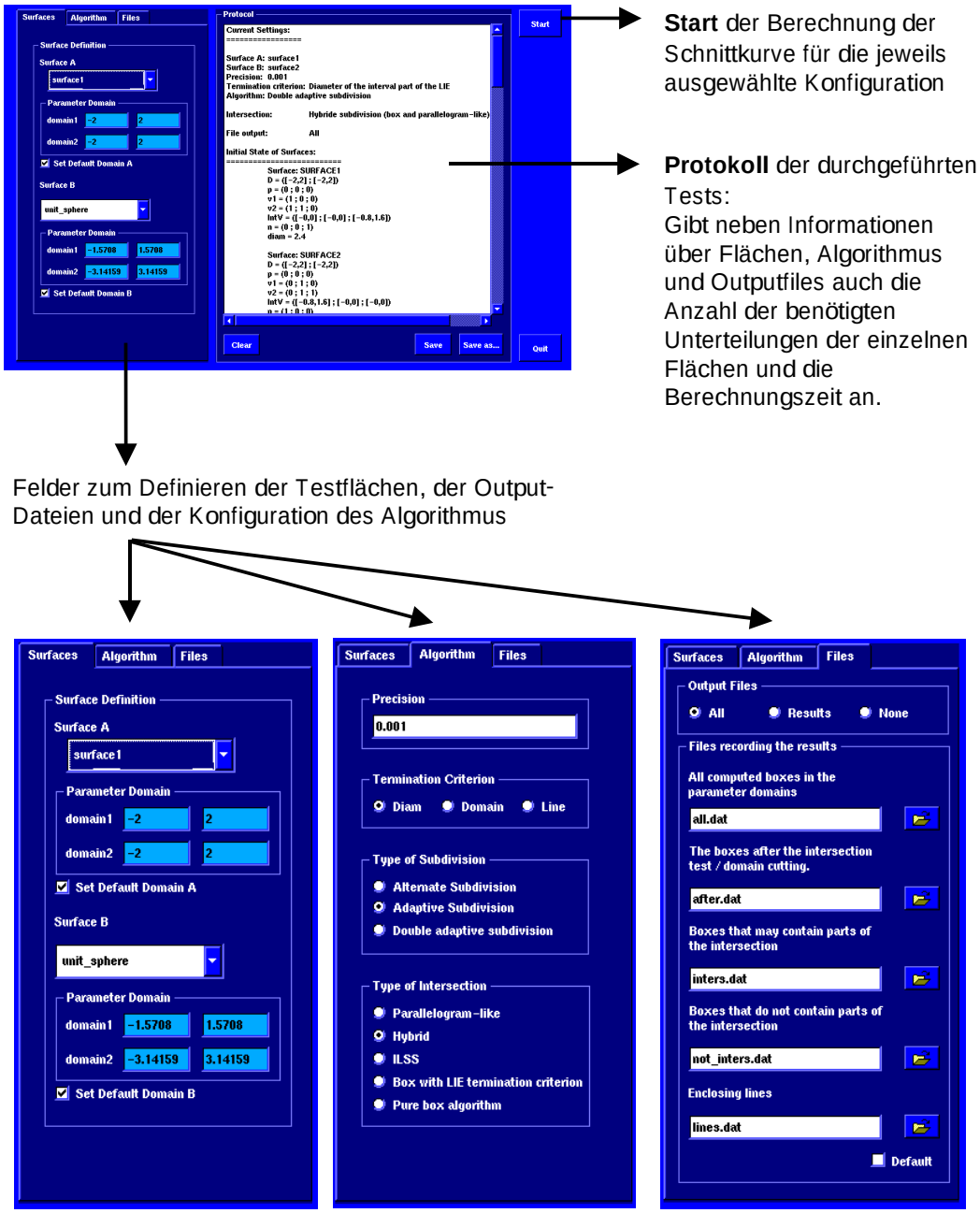


Abbildung 8.2: User Interface des Testprogrammes

[3] http://www.wolfram.com
[4] http://phong.informatik.uni-leipzig.de/ kuska/mview3d.html/

9

Tests, Ergebnisse und die beste Variante des Schnittalgorithmus

In diesem Kapitel werden anhand einiger Beispiele die Verwendung der verschiedenen Hüllkörper, Schnittalgorithmen, Unterteilungsstrategien und Abbruchkriterien miteinander verglichen.

Folgende Fragestellungen wurden im Rahmen der Testreihe untersucht:

- Wie verhält sich der Schnittalgorithmus bei der Verwendung der verschiedenen Hüllkörper und Schnittests?
- Welchen Einfluß hat die Unterteilungsstrategie auf die Laufzeit?
- Wie hoch ist die Qualität der Ergebnisse für verschiedenen Abbruchkriterien?

Für die nötigen Tests wurde die in Kapitel 8 beschriebene Implementierung verwendet. Die angegebenen absoluten Zeiten beziehen sich auf einen PC mit 800 MHz Athlon Prozessor und 512 MB RAM. Die Zeitangaben sind nur als Richtwerte zu sehen, da

- während der Tests größere Laufzeitschwankungen beobachtet wurden und
- die verwendeten Bibliotheken (siehe auch Bemerkung in Anhang A Abschnitt 1.2.5) und die eigenen Implementierungen nur teilweise optimiert wurden. Letzteres gilt vor allem für die Algorithmen, die mit Kombinationen von LIE und BOX Hüllkörpern arbeiten.

9.1 Vergleich verschiedener Hüllkörper und Schnittests

In diesem Abschnitt wird in drei Tests die Verwendung unterschiedlicher Hüllkörper und Schnittests diskutiert. Unterteilungsstrategie und Abbruchkriterium bleiben für fast alle verwendete Hüllkörper gleich. Die geforderte Genauigkeit der Ergebnisse wurde relativ hoch angesetzt. Die getesteten Algorithmen haben folgende Konfigurationen:

Algorithmus	BOUND	TEST	TERM	RESULT	SUBDIV
TaylorLie	TMLIE	LIE	LIE=0.0001	LINE	DOUBLE
AffineLie	AALIE	LIE	LIE=0.0001	LINE	DOUBLE
TaylorHybrid	BOXTMLIE	HYBRID	LIE=0.0001	LINE	DOUBLE
AffinHybrid	BOXAALIE	HYBRID	LIE=0.0001	LINE	DOUBLE
TaylorILSS	TMLIE	ILSS	LIE=0.0001	LINE	DOUBLE
Box	BOX	BOX	LIE=0.0001	LINE	DOUBLE
PureBox	BOX	BOX	DOMAIN=0.0001	DOMAIN	ADAPT

Test 1: Zwei Quadrikenpatches

Testflächen:

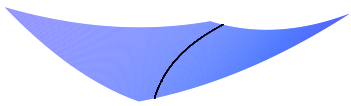
$$F: \ \boldsymbol{f}(u,v) = \begin{pmatrix} u+v \\ v \\ 0.1(u+v)^2 \end{pmatrix}; \ (u,v) \in [-2,2]^2$$

$$G: \ \boldsymbol{g}(s,t) = \begin{pmatrix} 0.1(s+t)^2 \\ s+t \\ t \end{pmatrix}; \ (s,t) \in [-2,2]^2$$

Die Abbildung rechts oben zeigt die Flächen F (grün) und G (blau). Das Bild unten neben der Tabelle zeigt die berechnete Schnittkurve auf einem der Patches.

Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorLie	428	0	0.15	0.06
TaylorHybrid	428	0	0.14	0.06
TaylorILSS	420	4	0.23	0.09
AffineLie	328	0	0.33	0.13
AffineHybrid	328	0	0.33	0.13
Box *)	12028	11068	1.9	0.77
PureBox	29128	24652	2.48	1.0



*) Die hier angegebenen Ergebnisse weichen von den Ergebnissen in [Bühler '01b], [Bühler '01c] und [Bühler, Barth '01] stark ab. Dieses Phänomen ist auf einen Rundungsfehler in den an sich identischen Implementierungen zurückzuführen, der sich lediglich dadurch ergibt, daß in der Version der drei Artikel die Intervallabschätzung des Restglieds als Durchmesser genommen wird, bei diesem Test aber der Durchmesser des Intervallpunktes der LIE. Prinzipiell sollte dieser Durchmesser gleich sein, da es sich

beim Intervallpunkt nur um die "verschobene" Version der Intervallabschätzung des Restgliedes handelt. Die durch den Rundungsfehler verursachte Differenz hat aber zur Folge, daß der Algorithmus eine völlig andere (und zufälligerweise effizientere) Unterteilungsreihenfolge auswählt.

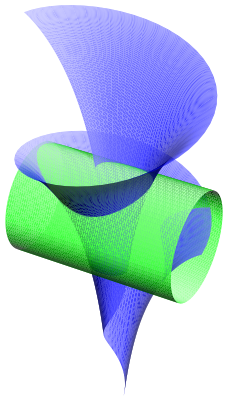
Test 2: Dini-Fläche und Zylinder

Testflächen:

- **Dini-Fläche** (blau):

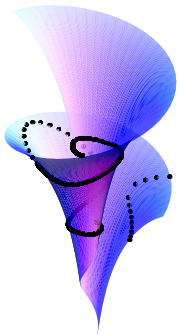
$$F: \; \boldsymbol{f}(u,v) := \begin{pmatrix} \cos(u)\sin(v) \\ \sin(u)\sin(v) \\ \cos(v) + \log(\tan(\frac{v}{2})) + 0.2\,u \end{pmatrix};$$

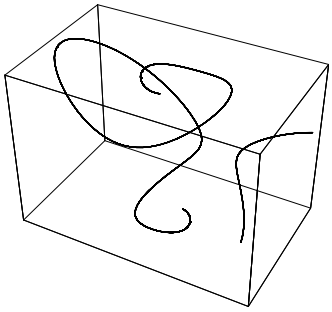
mit $(u,v) \in [0, 3.5\,\pi] \times [0.1, 1.0]$.

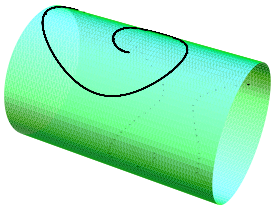


- **Zylinder** (grün):

$$G: \; \boldsymbol{g}(s,t) := \begin{pmatrix} s \\ 0.5\,\cos(t) \\ 0.5\,\sin(t) \end{pmatrix}; \; (s,t) \in [-0.8, 0.8] \times [0, 2\,\pi]$$

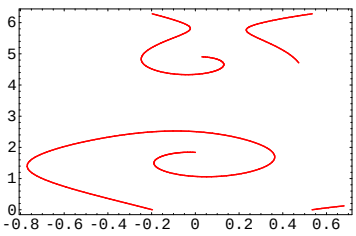






Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorLie	1736	322	0.92	0.07
TaylorHybrid	1712	290	0.99	0.08
TaylorILSS	1836	352	1.37	0.11
AffineLie	1488	140	3.02	0.24
AffineHybrid	1484	138	3.16	0.25
Box	45624	50542	12.19	0.97
PureBox	88652	12986	12.5	1.0



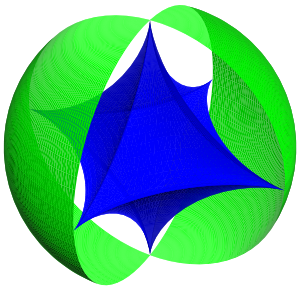
Test 3: Flächen und Schnittkurven mit Singularitäten

Testflächen:

- **Astroidal** (blau):

$$F: \ \boldsymbol{f}(u,v) := \begin{pmatrix} \cos^3(u)\cos^3(v) \\ \sin^3(u)\cos^3(v) \\ \sin^3(v) \end{pmatrix};$$

mit $(u,v) \in [-\pi/2, \pi/2] \times [\pi, \pi]$.



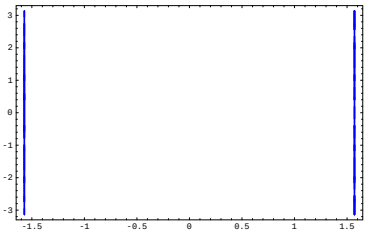
- **Einheitskugel** (grün):

$$G: \ \boldsymbol{g}(s,t) := \begin{pmatrix} \cos(s)\cos(t) \\ \cos(s)\sin(t) \\ \sin(s) \end{pmatrix};$$

mit $(s,t) \in [-\pi/2, \pi/2] \times [\pi, \pi]$.

Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorLie	1982	4592	2.24	0.29
TaylorHybrid	1182	4384	1.52	0.2
TaylorILSS	1686	4688	3.17	0.42
AffineLie	3014	2240	7.85	1.03
AffineHybrid	2510	2192	7.62	1.0
Box *)	118742	119976	53.16	/
PureBox *)	41412	26944	4.58	/



*) Bei beiden Algorithmen mußte die Genauigkeit reduziert werden, da mit der ursprünglichen Genauigkeit die Rechenzeit jenseits der Grenze von drei Minuten lag. Die Zahlen des Box-Algorithmus wurden für eine Genauigkeit LIE = 0.001 berechnet, die des PureBox-Algorithmus für die Genauigkeit DOMAIN = 0.01.

Bewertung:

- Der Unterteilungsalgorithmus ist bei Verwendung der affinen Arithmetik zur Berechnung der LIEs langsamer, als bei der Verwendung der Taylor-Modell-Methode, obwohl weniger Unterteilungen notwendig sind. Der Zeitverbrauch der Algorithmen, die die affine Arithmetik zur Berechnung der LIEs verwenden, steigt bei ansteigender Komplexität des Problems im Vergleich zur Taylor-Modell-Methode.

- Der hybride Ansatz bringt in den beiden Tests mit nichtsingulären Schnittkurven nur Ver-

besserungen im Hinblick auf die Anzahl der Unterteilungen. Die Gesamtlaufzeit verbessert sich aber nicht. In Test 3 ist der hybride Ansatz vor allem im Hinblick auf die Taylor Modell basierten LIEs effektiver, als der reine LIE Algorithmus.

- Der ILSS-Schnittalgorithmus ist in allen drei Tests langsamer, als der Schnittalgorithmus, der in Kapitel 6 entwickelt wurde.
- Die Algorithmen, die ausschließlich mit achsenparallelen Hüllboxen arbeiten, benötigen je nach Komplexität des Problems, mindestens die 15-fache Zeit um vergleichbare Ergebnisse zu erzielen. Treten Singularitäten auf, wie im dritten Test, steigt die Anzahl der benötigten Unterteilungen sprunghaft an.

9.2 Vergleich der Unterteilungsstrategien

Diese Testreihe untersucht den Einfluß der verschiedenen Unterteilungsstrategien auf das Laufzeitverhalten des Algorithmus. Untersucht wurden, mit LIEs als Hüllkörpern und dem Durchmesser der einschließenden Geraden als Abbruchkriterium, die alternierende uniforme Unterteilung in vier Teilpatches (UNI), die adaptive Unterteilung in vier Teilpatches (ADAPT) und die doppelt adaptive Unterteilung (DOUBLE).

Folgende Algorithmen wurden getestet:

Algorithmus	BOUND	TEST	TERM	RESULT	SUBDIV
TaylorUni	TMLIE	LIE	LINE=0.001	LINE	UNI
TaylorAdapt	TMLIE	LIE	LINE=0.001	LINE	ADAPT
TaylorDouble	TMLIE	LIE	LINE=0.001	LINE	DOUBLE

Test 4: Quadrikenpatches

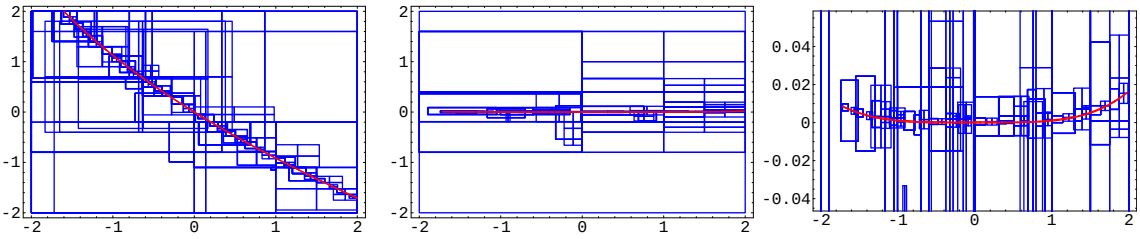
Testflächen:

Die beiden Flächen sind definiert wie in Test 1 des vorigen Abschnittes.

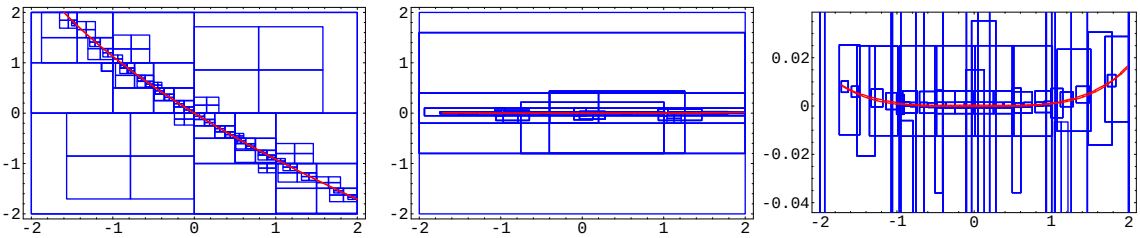
Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorUni	184	200	0.09	1.0
TaylorAdapt	236	4	0.08	0.9
TaylorDouble	146	0	0.06	0.67

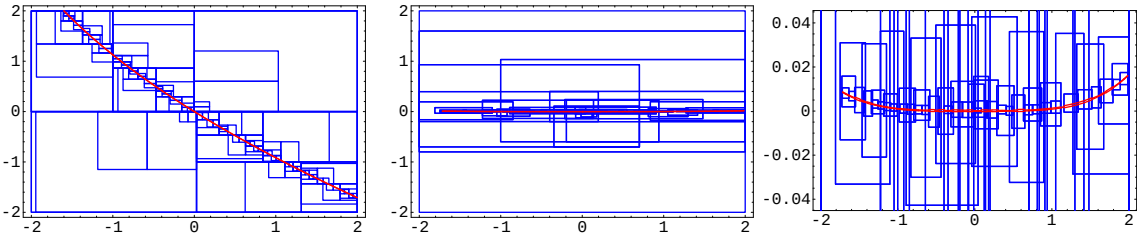
Bild 9.1 zeigt die untersuchten Teilparametergebiete der Patches für die verschiedenen Algorithmen.



(a) TaylorUni



(b) TaylorAdapt



(c) TaylorDouble

Abbildung 9.1: Ergebnisse der Algorithmen aus Test 4: Die Bilder zeigen die Unterteilung (blau) und die einschließenden parallelen Linien (rot) der Schnittkurve in den Parametergebieten beider der Flächen für den jeweiligen Algorithmus. Links: Parametergebiet der Fläche F, Mitte: Parametergebiet der Fläche G, Rechts: Vergrößerung des mittleren Bildes.

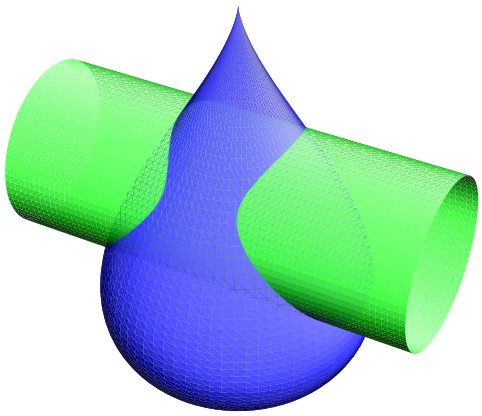
Test 5: Tropfen und Zylinder

Testflächen:

- **Tropfen** (blau):

$$F: \boldsymbol{f}(u,v) := \begin{pmatrix} cos(u)\, sin(v)\, \frac{1-cos(v)}{2} \\ sin(u)\, sin(v)\, \frac{1-cos(v)}{2} \\ cos(v) \end{pmatrix}$$

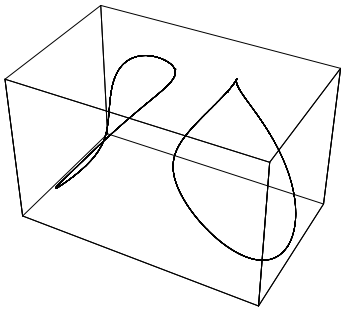
mit $(u,v) \in [-\pi,\pi] \times [0,\pi]$.

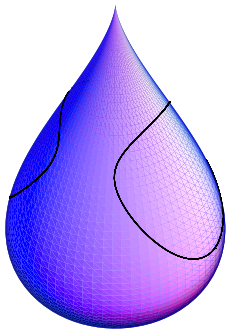


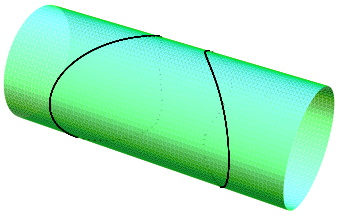
- **Zylinder** (grün):

$$G: \boldsymbol{g}(s,t) := \begin{pmatrix} s \\ 0.4\, \sin(t) \\ 0.4\, \cos(t) \end{pmatrix};$$

mit $(s,t) \in [-1,1] \times [0, 2\,\pi]$.





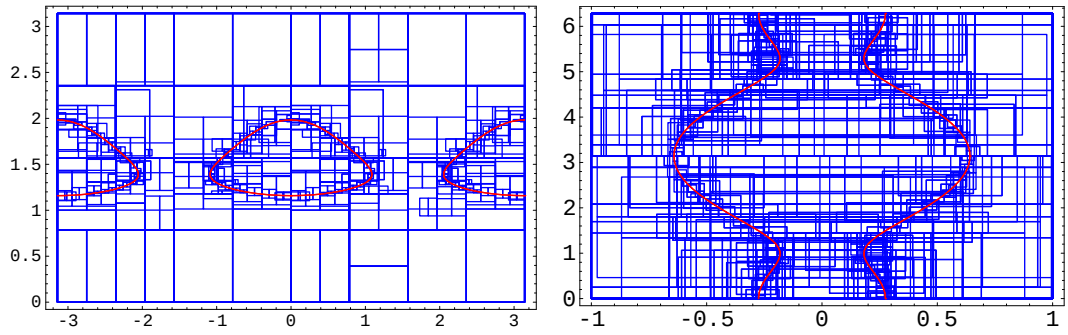


Ergebnisse:

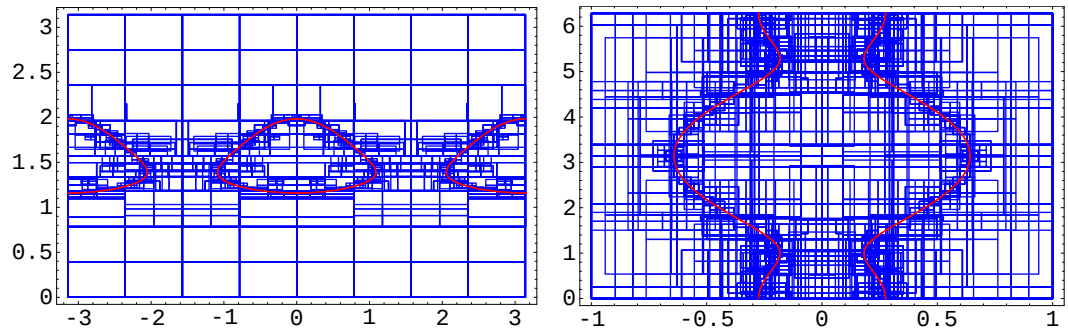
	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorAdapt	1332	640	0.57	1.0
TaylorDouble	604	176	0.26	0.46

Bild 9.2 zeigt die untersuchten Teilparametergebiete der Patches für die verschiedenen Algorithmen.

Bemerkung:
Die uniforme Unterteilung führt bei diesem Beispiel nicht zu einem annehmbaren Ergebnis. Grund hierfür ist, daß durch die Parametergebietsreduzierung die Parametergebiete ungleichmäßig verkleinert werden. Die uniforme Unterteilung wählt für die nächste Unterteilungen



(a) TaylorDouble



(b) TaylorAdapt

Abbildung 9.2: Ergebnisse der Algorithmen aus Test 5: Die Bilder zeigen die Unterteilung (blau) und die einschließenden parallelen Linien (rot) der Schnittkurve in den Parametergebieten beider Flächen für den jeweiligen Algorithmus.

aber immer abwechselnd eines der beiden Parametergebiete, egal wie klein es bereits ist. Da der Zylinder eine wesentlich einfachere Fläche darstellt, als der Tropfen, werden dessen Parametergebiete sehr viel schneller reduziert und konvergiere gegen einzelne Punkte für die keine LIE mehr berechnet werden kann und somit auch keine Parametergebietsreduzierung mehr stattfindet.

Bewertung:

- Die doppelt adaptive Unterteilung ist vor allem bei komplexeren Beispielen wesentlich schneller als eine einfach adaptive Unterteilung. (Siehe Test 5)
- Die alternierende Unterteilung, die keinerlei topologische Eigenschaften der Flächenstücke beachtet bringt nur für einfache Beispiele akzeptable Laufzeiten. (Test 4 + 5)

9.3 Vergleich verschiedener Abbruchkriterien - Qualität der Ergebnisse

In diesem Abschnitt werden die drei Abbruchkriterien für den Algorithmus untersucht: Flachheit des Patches, Durchmesser des Parametergebietes und Durchmesser des Intervallschnittsegements. Dabei geht es nicht nur um die Geschwindigkeit, mit der die Ergebnisse berechnet werden können, sondern auch um die Qualität der Ergebnisse.

Die Testalgorithmen haben folgende Konfiguration:

Algorithmus	BOUND	TEST	TERM	RESULT	SUBDIV
TaylorDiam	TMLIE	LIE	LIE	LINE	DOUBLE
TaylorDomain	TMLIE	LIE	DOMAIN	LINE	DOUBLE
TaylorLine	TMLIE	LIE	LINE	LINE	DOUBLE
TaylorHybridLine	BOXTMLIE	HYBRID	LINE	LINE	DOUBLE
AffineLine	AALIE	LIE	LINE	LINE	DOUBLE
AffineHybridLine	BOXAALIE	HYBRID	LINE	LINE	DOUBLE
BoxDiam	BOX	BOX	LIE	LINE	DOUBLE
PureBox	BOX	BOX	DOMAIN	DOMAIN	ADAPT

Test 6: Vergleich verschiedener Abbruchkriterien

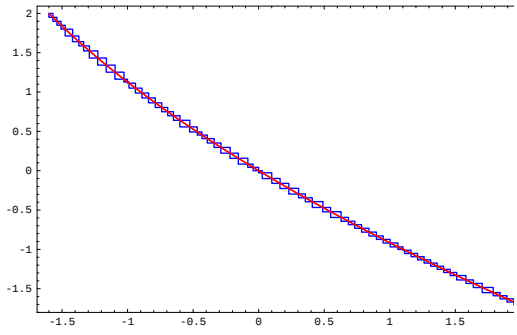
Testflächen:

Die beiden Flächen sind definiert wie in Test 1 des vorigen Abschnittes. Die Abbruchkriterien wurden so gewählt, daß visuell ähnlicher Ergebnisse entstehen.

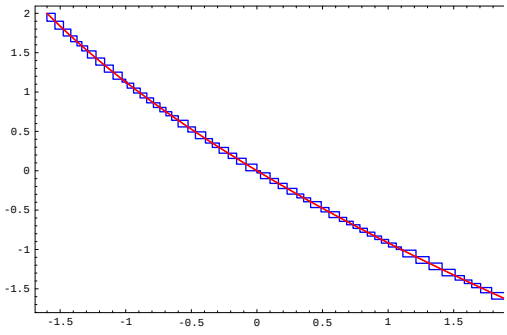
Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorDiam LIE = 0.001	146	0	0.05	0.0625
TaylorDomain DOMAIN = 0.01	126	0	0.06	0.075
TaylorLine LINE = 0.001	146	0	0.05	0.0625
AffineLine LINE = 0.001	118	0	0.12	0.15
BoxDiam LIE = 0.001	4570	4518	0.8	1.0
PureBox DOMAIN = 10^{-5}	105372	88020	10.67	13.34

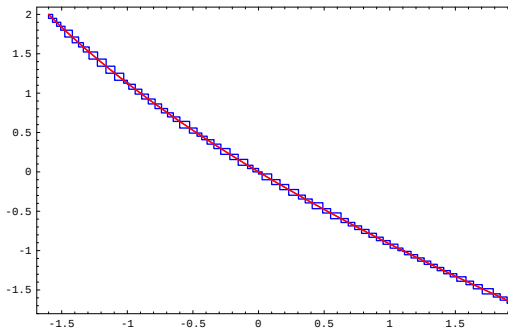
Die Bilder 9.3 (a) – (f) zeigen die Ergebnisse der verschiedenen Algorithmen. Die blauen Quadrate sind die einschließenden Parametergebiete, die roten Linien Paare einschließender Geraden. (Die Geraden liegen so dicht zusammen, daß sie nicht zu unterscheiden sind.)



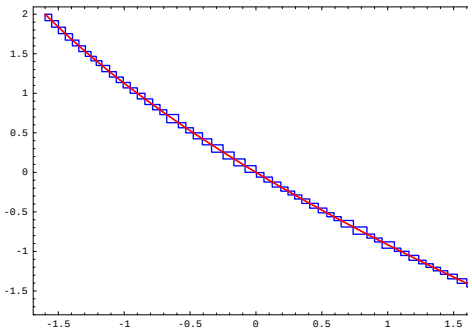
(a) TaylorDiam



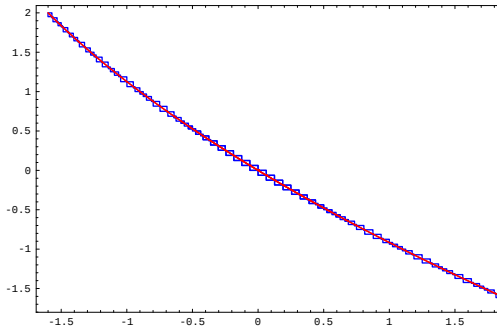
(b) TaylorDomain



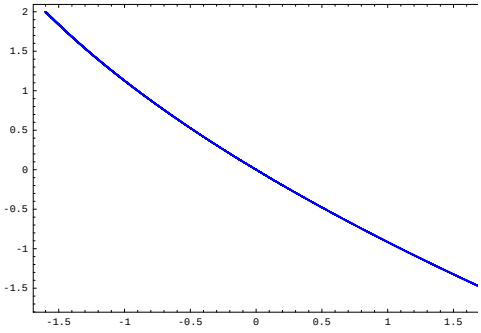
(c) TaylorLine



(d) AffineLine



(e) BoxDiam



(f) PureBox

Abbildung 9.3: Ergebnisse der getesteten Algorithmen aus Test 6: Die Bilder zeigen die einschließenden achsenparallelen Teilgebiete (blau) und die einschließenden parallelen Linien (rot) der Schnittkurve im Parametergebiet der Flächen F.

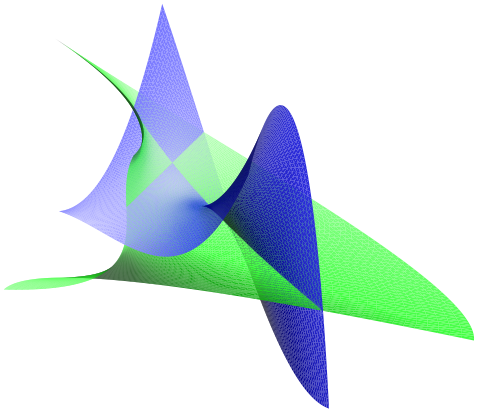
Test 7: Qualität des Ergebnisses bei auftretenden Singularitäten

Testflächen:

- **Affensattel 1** (blau):

$$F: \boldsymbol{f}(u,v) := \begin{pmatrix} u \\ v \\ u^3 - 3v^2u \end{pmatrix};$$

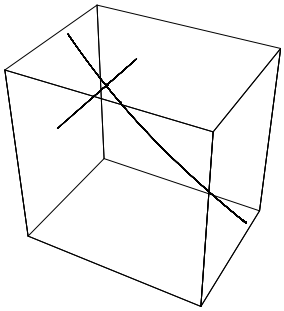
mit $(u,v) \in [-0.3,1] \times [-0.3,1]$.

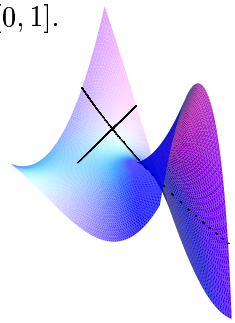


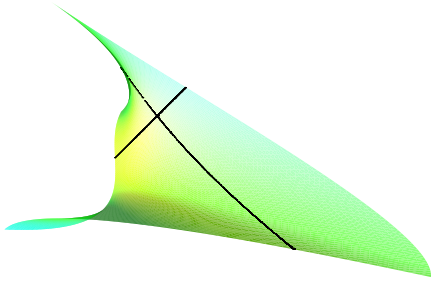
- **Affensattel 2** (grün):

$$G: \boldsymbol{g}(s,t) := \begin{pmatrix} s^3 - 3t^2s \\ t \\ s \end{pmatrix};$$

mit $(s,t) \in [-1,0.4] \times [0,1]$.



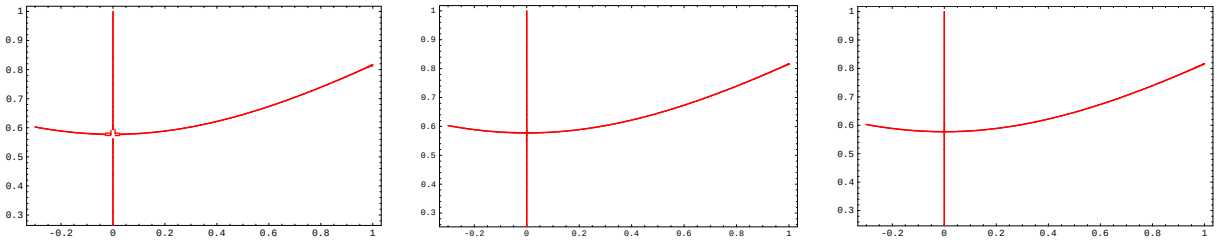




Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorDiam LIE = 0.001	246	250	0.15	0.34
TaylorHybridLine LINE = 0.001	264	250	0.16	0.36
TaylorLine LINE = 0.001	288	276	0.2	0.45
AffineHybridLine LINE = 0.001	234	266	0.37	0.84
AffineLine LINE = 0.001	244	278	0.44	1.0

Abbildung 9.4 zeigt, wie unterschiedlich die Ergebnisse im singulären (Doppel-)Punkt der Schnittkurve sind.



(a) TaylorDiam (b) TaylorLine (c) AffineLine

Abbildung 9.4: Die Bilder zeigen die einschließenden Geraden, die Ergebnis der in den Bildunterschriften genannten Algorithmen aus Test 7 sind.

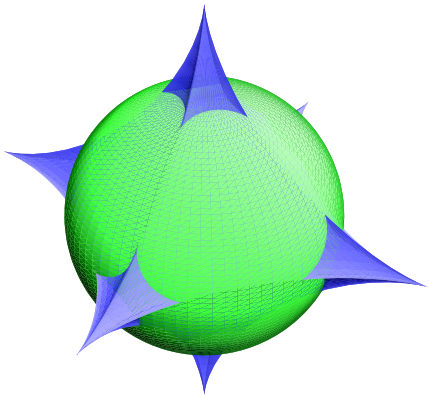
Test 8: Vergleich der Ergebnisse bei kleiner werdender oberen Schranke für das Abbruchkriterium

Testflächen:

- **Astroidal** (blau):

$$F: \ \boldsymbol{f}(u,v) := \begin{pmatrix} 1.2\ \cos^3(u)\cos^3(v) \\ 1.2\ \sin^3(u)\cos^3(v) \\ 1.2\ \sin^3(v) \end{pmatrix};$$

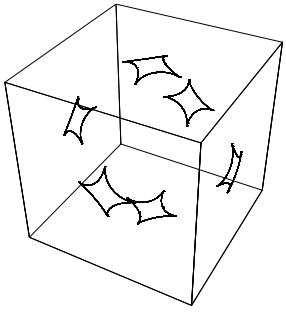
mit $(u,v) \in [-\pi/2, \pi/2] \times [\pi, \pi]$.

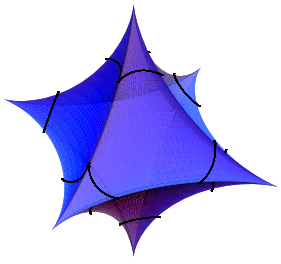


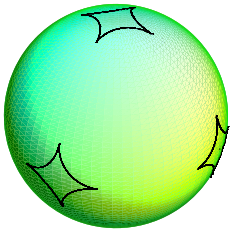
- **Einheitskugel** (grün):

$$G: \ \boldsymbol{g}(s,t) := \begin{pmatrix} \cos(s)\cos(t) \\ \cos(s)\sin(t) \\ \sin(s) \end{pmatrix};$$

mit $(s,t) \in [-\pi/2, \pi/2] \times [\pi, \pi]$.







Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorHybridLine LINE = 0.1	774	936	0.67	0.04
TaylorLine LINE = 0.1	1470	1728	1.05	0.07
AffineHybrideLine LINE = 0.1	878	544	1.62	0.11
AffineLine LINE = 0.1	1390	848	2.62	0.18
TaylorHybridLine LINE = 0.01	1398	1320	1.15	0.08
TaylorLine LINE = 0.01	2494	2336	1.43	0.1
AffineHybrideLine LINE = 0.01	1638	768	2.82	0.19
AffineLine LINE = 0.01	2590	1136	4.22	0.28
TaylorHybridLine LINE = 0.001	2150	1864	1.84	0.12
TaylorLine LINE = 0.001	3294	2944	2.1	0.14
AffineHybrideLine LINE = 0.001	2814	1200	5.26	0.35
AffineLine LINE = 0.001	3950	1648	7.19	0.48
TaylorHybridLine LINE = 0.0001	4614	2632	3.9	0.26
TaylorLine LINE = 0.0001	5854	3712	4.42	0.3
AffineHybrideLine LINE = 0.0001	6486	1776	12.37	0.83
AffineLine LINE = 0.0001	7646	2224	14.92	1.0

Die Abbildungen 9.5 und 9.6 zeigen die Ergebnisse der getesteten Algorithmen in den jeweiligen Parametergebieten.

Bewertung

- Bei den einfachen Flächen aus Test 6 werden für LIEs ähnliche Ergebnisse bezüglich Anzahl der Unterteilungen und Qualität der Ergebnisse für alle drei untersuchten Abbruchkriterien erreicht. (Siehe Abbildung 9.2.)
- Um Ergebnisse ähnlicher Qualität mit achsenparallelen Hüllquadern zu erreichen, müssen im Falle des PureBox-Algorithmus über 1500 mal mehr Unterteilungen vorgenommen werden wofür ca. 500% mehr Zeit benötigt wird. Im Falle des BoxDiam-Algorithmus sind immer noch 72 mal mehr Unterteilungen und 13.3 % mehr Zeit nötig. (Test 6)
- Das LINE Abbruchkriterium benötigt zwar etwas mehr Zeit, liefert aber besseren Einschließungen durch parallele Geradensegmente, da nach Bedarf weiter unterteilt wird, bis eine bestimmter Abstand der Linie unterschritten wird. Abbildung 9.4 (Test 7) zeigt den Unterschied des LINE Abbruchkriteriums im Vergleich zum LIE Kriterium: Die beiden Testflächen Berühren sich in einem Punkt. Dadurch entsteht ein singulärer Punkt der Schnittkurve, der von den beiden Line-Algorithmen sehr viel besser approximiert wird, als von dem Algorithmus mit LIE-Abbruchkriterium.
- Die Abbildung 9.5 und 9.6 in Test 8 zeigen, daß die berechneten einschließenden Linien unter Umständen bei zu groben Wahl der Abbruchschranke keine sinnvolle Ergebnisse liefern. Das gilt vor allem dann, wenn die Schnittkurve Singularitäten aufweist. Andererseits

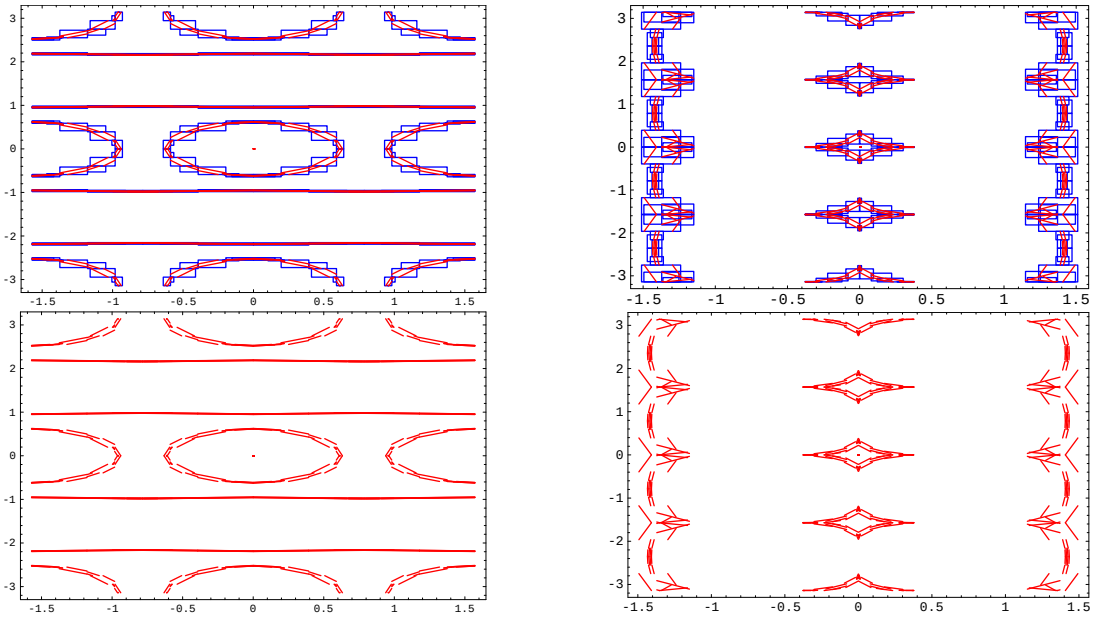


Abbildung 9.5: Ergebnisse des TaylorLine Algorithmus aus Test 8 für einen Durchmesser der einschließenden Intervallinien von 0.1. Die beiden linken Bilder zeigen die Einschließungen der Schnittkurven im Parametergebiet des Astroidals, die rechten Einschließungen im Parametergebiet der Kugel. Die oberen Bilder zeigen jeweils eine Kombination der achsenparallelen Einschließungen mit den einschließenden Geraden, die unteren nur die einschließenden Geraden.

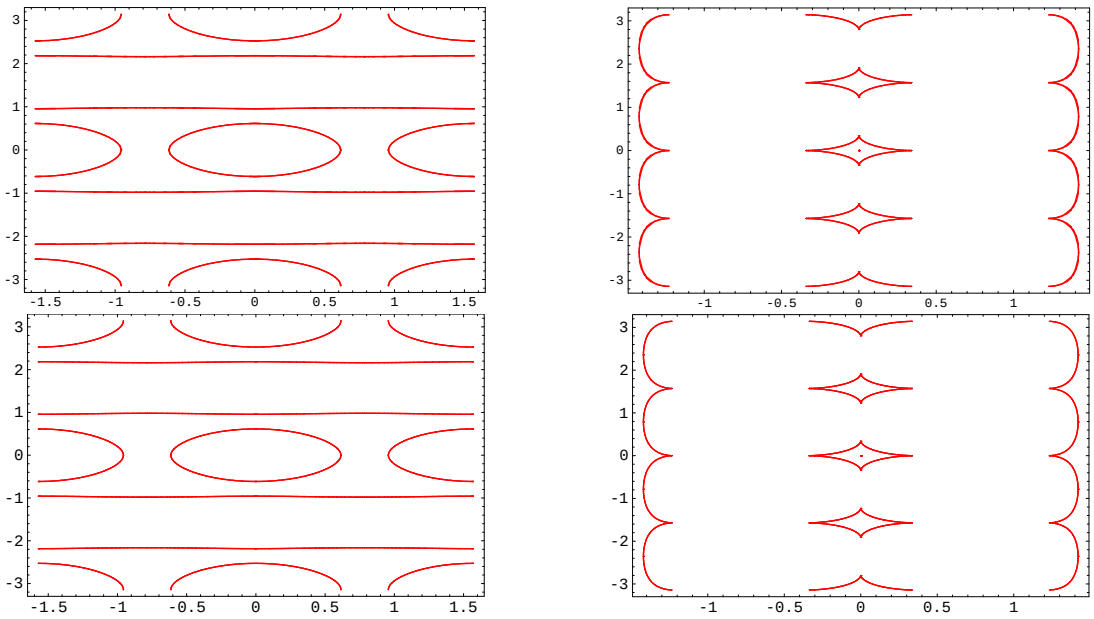


Abbildung 9.6: Die oberen Bilder zeigen die Linien-Einschließungen aus Test 8 für LINE = 0.01, die unteren für LINE = 0.001

liefert der Algorithmus schon bei einer vergleichsweise geringen Genauigkeit sehr genaue Einschließungen des Ergebnisses.

- Test 7 und Test 8 zeigen deutlich die Überlegenheit des hybriden Ansatzes bezüglich der Geschwindigkeit, falls Schnittkurven mit Singularitäten auftreten. Test 8 macht allerdings auch deutlich, daß bei steigender Genauigkeit der Abstand zwischen hybridem und einem reinen LIE Algorithmus sinkt.

9.4 Vergleich der Ergebnisse mit dem hybriden Epipedalgorithmus aus [Huber '99] und [Huber '98]

Ein direkter Vergleich der beiden Algorithmen ist schwer, da unterschiedliche Implementierungen und Abbruchkriterien verwendet wurden, die einen direkten Zeitvergleich unmöglich machen.

Test 9: Epiped/LIES

Testkonfiguration:

Algorithmus	BOUND	TEST	TERM	RESULT	SUBDIV
TaylorHybrid	BOXTMLIE	HYBRID	DOMAIN	LINE	DOUBLE

Als zweiter Algorithmus wurde der hybride Epipedalgorithmus aus [Huber '99] herangezogen. Das Abbruchkriterium für die Unterteilung der beiden Patches ist relativ zum ursprünglichen Flächeninhalt des jeweiligen Parametergebietes angegeben. Zu Vergleichszwecken wurde in diesem Test das kleinere Schranke verwendet.

Ergebnisse:

- **Testflächen:** Quadrikenpatches aus Test 1:

	Unterteilungen		
Algorithmus	F	G	gesamt
TaylorHybrid DOMAIN = 0.0016	278	0	278
Huber DOMAIN = 0.0016	747	591	1065

- **Testflächen:** Tropfen und Zylinder aus Test 5:

	Unterteilungen		
Algorithmus	F	G	gesamt
TaylorHybrid DOMAIN = 0.0002	1120	362	1482
Huber DOMAIN = 0.0002	2211	2469	4680

Bewertung:

Auch wenn kein direkter Zeitvergleich möglich ist, macht die Anzahl der benötigten Unterteilungen deutlich, daß LIEs die Flächen sehr viel enger einschließen und/oder der Schnittest sehr

viel effizienter funktioniert, als in Hubers Algorithmus.

Zu beachten ist, daß die tatsächlichen Einschließungen im Parametergebiet des TaylorHybrid Algorithmus wesentlich genauer sind, da ja zusätzlich noch die einschließenden Intervallinien berechnet werden.

Eine theoretische Diskussion zum Vergleich der Berechnung des Schnittest von LIEs im Vergleich zu Parallelepipeden findet man in Kapitel 5 Abschnitt 5.7.3 und Kapitel 6 Abschnitt 6.6.

9.5 Probleme

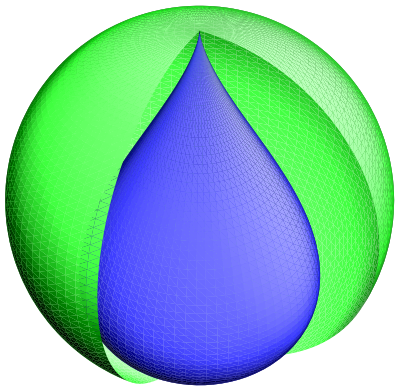
Ein sprunghafter Anstieg von Unterteilungen tritt auf, wenn eine der Flächen eine Singularität besitzt, in der beide Ableitungen verschwinden. In diesem Fall wird ein großer Teil der LIEs als Hüllkörper durch gewöhnliche achsenparallele Boxen ersetzt und dadurch fällt natürlich auch die Parametergebietsreduzierung weg:

Test 10: Tropfen und Einheitkugel

- **Tropfen** (blau):

$$F: \ \boldsymbol{f}(u,v) := \begin{pmatrix} cos(u)\, sin(v)\, \frac{1-cos(v)}{2} \\ sin(u)\, sin(v)\, \frac{1-cos(v)}{2} \\ cos(v) \end{pmatrix}$$

  mit $(u,v) \in [-\pi,\pi] \times [0,\pi]$.



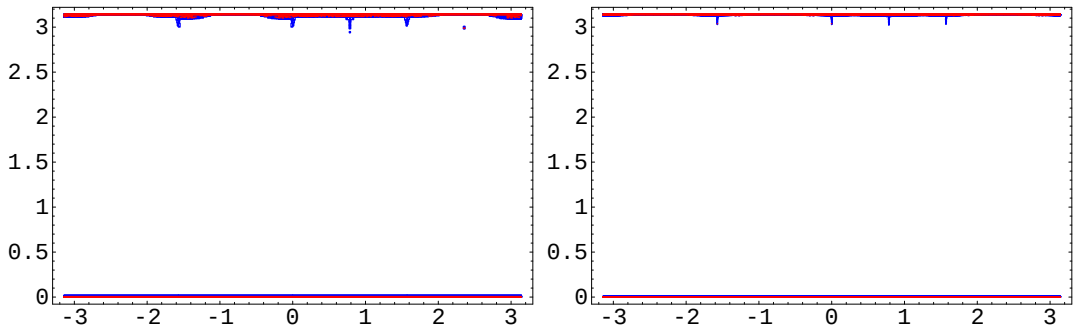
- **Einheitskugel** (grün):

$$G: \ \boldsymbol{g}(s,t) := \begin{pmatrix} \cos(s)\cos(t) \\ \cos(s)\sin(t) \\ \sin(s) \end{pmatrix};$$

  mit $(s,t) \in [-\pi/2,\pi/2] \times [\pi,\pi]$.

Testkonfiguration

Algorithmus	BOUND	TEST	TERM	RESULT	SUBDIV
TaylorLie	TMLIE	LIE	LIE	LINE	DOUBLE
TaylorHybrid	BOXTMLIE	HYBRID	LIE	LINE	DOUBLE
AffineLie	AALIE	LIE	LIE	LINE	DOUBLE
AffineHybrid	BOXAALIE	HYBRID	LIE	LINE	DOUBLE



(a) AffineHybrid 0.001

(b) AffineHybrid 0.0001

Abbildung 9.7: Ergebnisse der Algorithmen aus Test 10: Die Bilder zeigen die Unterteilung (blau) und die einschließenden parallelen Linien (rot) der Schnittkurve im Parametergebiet der Fläche F für den jeweiligen Algorithmus.

Ergebnisse:

	Unterteilungen		Zeit	
Algorithmus	F	G	absolut (sek.)	relativ
TaylorHybrid LIE = 0.001	4260	4920	5.17	0.06
TaylorLIE LIE = 0.001	6132	6410	6.37	0.08
AffineHybrid LIE = 0.001	3640	4008	9.32	0.11
AffineLIE LIE = 0.001	5046	5162	12.8	0.15
TaylorHybrid LIE = 0.0001	24142	24966	24.95	0.3
TaylorLIE LIE = 0.0001	35800	34910	33.91	0.41
AffineHybrid LIE = 0.0001	21774	20866	54.41	0.66
AffineLIE LIE = 0.0001	31612	29868	83.06	1.0

Abbildung 9.7 zeigt die Ergebnisse der Algorithmen, die LIEs basierend auf affiner Arithmetik im hybriden Algorithmus verwendet haben. Taylor Modell basierte LIEs liefern visuell ähnliche Ergebnisse. Das LINE Abbruchkriterium versagt in diesem Test.

9.6 Zusammenfassung der Testergebnisse zu einem neuen Schnittalgorithmus

Die Tests in diesem Kapitel haben gezeigt, daß im allgemeinen der hybride doppelt adaptive Ansatz, d.h. LIEs kombiniert mit achsenparallelen Boxen und der doppelt adaptiven Unterteilung, sowohl bezüglich Zeitaufwand als auch bezüglich der Anzahl der Unterteilungen die besten Ergebnisse liefert (siehe Tests 3 und 6–10). Sind die Flächen und Schnittkurven einfach, können die Ergebnisse des reinen LIE-Ansatzes auch gleich oder etwas besser sein (siehe Test 1 und 2) als die eines hybriden Ansatzes. Auffällig ist, daß die Algorithmen, die LIEs verwenden, die mit affiner Arithmetik berechnet wurden, in den meisten Fällen 2-3 mal langsamer sind, als solche, die die

Taylor Modell Methode verwenden, obwohl ca. 20-25% weniger Unterteilungen benötigt wurden.

Die Algorithmen, in denen nur achsenparallele Bounding Boxen verwendet, bzw. nur eine gewöhnliche adaptive bzw. uniform Unterteilung angewandt wurden, liefern in den durchgeführten Tests sehr viel schlechtere Ergebnisse (Tests 1–3 und 6).

Das Abbruchkriterium, das sich an der Breite der einschließenden Intervallinien orientiert, liefert gleichmäßigere Einschließungen, als das Abbruchkriterium, das sich nur an der Dicke der Intervallpunkte der LIEs orientiert (Tests 6–8). Letzteres reagiert aber robuster in Bezug auf spezielle Singularitäten (Siehe Test 10).

Der bereits in Kapitel 4 skizzierte Algorithmus wird als Ergebnis dieser Testreihe noch in seiner hybriden Form für die Kombination

BOUND	TEST	TERM	RESULT	SUBDIV
BOXTMLIE oder BOXAALIE	HYBRID	LIE oder LINE	LINE	DOUBLE

wiedergegeben:

Input: Ein Paar AB bestehend aus den Flächenpatches A und B, den dazugehörigen einschließenden LIEs L(A),L(B) und den achsenparallelen Boxen Box(A),Box(B).

Output: Je eine Liste mit Teilparametergebieten und Streifen in den Parametergebieten, die sicher alle Parameterwerte der Schnittpunkte enthalten.

Schneide(Flächenpaar AB){
  Falls (Box(A) $\cap$ Box(B) $= \emptyset$)
    Abbruch;
  LIE-Schnittest und Parametergebietsreduzierung: $AB \rightarrow \tilde{A}\tilde{B}$;
  Falls (LIE-Schnitttest negativ)
    Abbruch;
  Falls (*Abbruchkriterium* erfüllt) {
    LIE-Schnittest und Parametergebietsreduzierung: $\tilde{A}\tilde{B} \rightarrow \tilde{\tilde{A}}\tilde{\tilde{B}}$;
    Falls (LIE-Schnitttest negativ)
      Abbruch;
    sonst{
      Berechne einschließenden Linien;
      Schreibe Ergebnisse;
      Abbruch;
    }
  Doppelt Adaptive Unterteilung der Parametergebiete;
  Rekursive Aufrufe von **Schneide** für die zwei neuen Flächenpaare,
  die aus der Unterteilung hervorgegangen sind;
}

10

Rück- und Ausblick

Ziel der Arbeit war es, einen schnellen, robusten und zuverlässigen Unterteilungsalgorithmus für den Schnitt beliebige parametrische Flächen zu entwickeln, der eine Einschließung der Schnittkurve sowohl in den Parametergebieten der beiden Patches, als auch im Objektraum als Ergebnis liefern sollte.

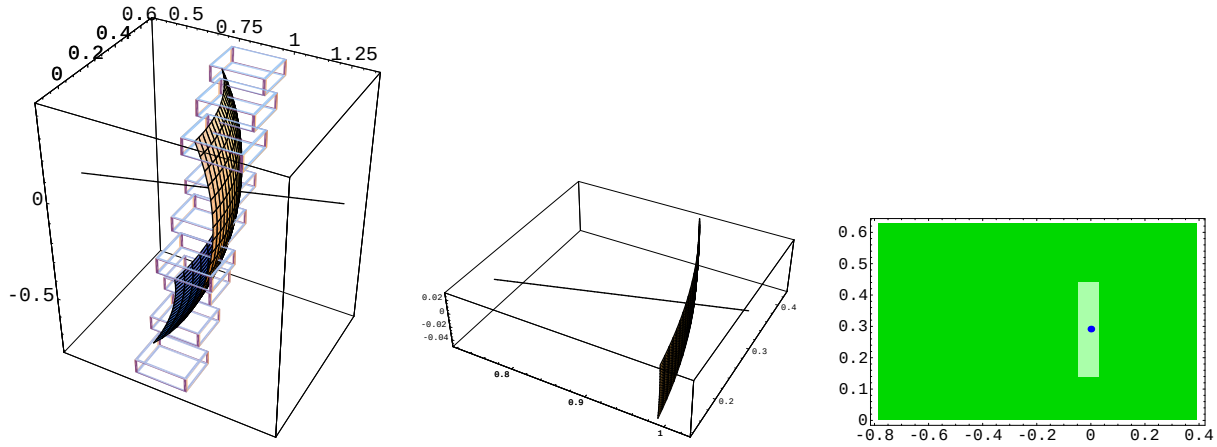
Zu diesem Zweck wurden Linearen Intervallabschätzungen als eine neue Art von Hüllkörpern eingeführt. Die vorgeschlagenen Berechnungsmethoden für LIEs garantieren eine robuste Berechnung und die sichere Einschließung des Flächenstückes, da beide auf der Basis zuverlässiger Arithmetiken entwickelt wurden. Es wurde gezeigt, daß LIEs die Flächen vergleichweise eng einschließen und implizit Auskunft über die Flachheit der Patches geben.

Die parametrische Form der LIEs erlaubt es, bei einem Schnittest nicht nur eine Aussage darüber zu treffen, ob und wo sich zwei Flächen im Objektraum schneiden, sondern auch über die Lage möglicherweise betroffener Parameterwerte. Mit dieser Eigenschaft stellen Lineare Intervallabschätzungen eine völlig neue Art von Hüllkörpern dar, die sich essentiell von den üblichen kompakten Hüllkörpern unterscheidet, die bei Schnittoperationen nur Auskunft über die Lage des Schnittest im Objekt-, aber nicht im Parameterraum geben können.

Für alle wichtigen Einzeloperationen eines allgemeinen Unterteilungsalgorithmus wurden neue und effiziente Algorithmen entwickelt, die den speziellen Eigenschaften der LIEs angepaßt und nach einer Testreihe zu einem neuen und vergleichsweise schnellen Unterteilungsalgorithmus zusammengefügt wurden.

Die positiven Ergebnisse dieser Arbeit legen nahe, das Konzept des parametrischen linearen Hüllkörpers auch auf andere parametrische Objekte zu übertragen und vorliegende Unterteilungsalgorithmen für eine Vielzahl von Schnittalgorithmen neu zu überdenken.

In [Bühler '01b] wurde die Idee der LIEs für parametrische Flächen auf allgemeine parametrische Objekte übertragen. Im gleichen Artikel finden sich auch Ansätze für Schnittalgorithmen für alle möglichen Kombinationen von Strahl, Kurven und Flächenschnitten. Abbildung 10.1 zeigt ein Beispiel für die Parametergebietsreduzierung bei einem Schnittest parametrische



(a) Flächenpatch mit Strahl

(b) Reduziertes Flächenpatch nach dem Schnittest

(c) Reduzierung im Parametergebiet. Der Punkt markiert den eigentlichen Schnittpunkt.

Abbildung 10.1: Beispiel für die Parametergebietsreduzierung bei einem Strahl/Flächenschnittest.

Fläche/Strahl, wenn LIEs als Hüllkörper für die Fläche verwendet werden.

Eine weitere mögliche Verallgemeinerung von LIEs bezieht sich nur auf Taylor Modell basierte LIEs. Die Verwendung von Taylor Modellen zweiter Ordnung als Hüllkörper stellen noch wesentlich engeren Einschließungen für parametrische Kurven und Flächen dar als LIEs. Ob sich der Mehraufwand für die Berechnung und einen Schnittest lohnt muß allerdings je nach Anwendung entschieden werden.

LIEs für parametrische Flächen stellen eine Linearisierung der jeweiligen Parameterform bezüglich eines bestimmten Parametergebietes dar. Das gleiche Prinzip kann auch auf implizite Kurven und Flächen übertragen werden. In [Bühler '01a] werden implizite LIEs für implizite Kurven und Flächen definiert: Die Lineare Intervallabschätzung einer impliziten Kurve über einen bestimmtest Gebiet ist beispielsweise eine implizite Intervallgerade der Form $ax + by \in I$, falls das Gebiet einen Teil der Kurve enthält.

Anhang A

Verifiziertes Rechnen

1.1 Intervallarithmetik

In der Beschreibung zum Buch "Interval Methods for Systems of Equations" ([Neumeier '90]) werden Intervalle folgendermaßen beschrieben: "An interval is the natural way of specifying a number that is specified only within a certain tolerances.". Ausgangspunkt für die Verwendung der Intervallrechnung in der Numerischen Mathematik war ursprünglich der Wunsch, die durch die begrenzte Genauigkeit der Maschinenarithmetik verursachten Rundungsfehler zu erfassen und eine enge Fehlerabschätzung zu erhalten ([Alefeld, Herzberger '74]). Die moderne Intervallanalysis beschäftigt sich unter anderem mit Methoden zur Bestimmung von Lösungen linearer und nichtlinearer Gleichungssysteme vor dem Hintegrund unsicherer Daten, der Berechnung von Wertebereichen von Funktionen und deren Ableitungen und der Verifizierung numerischer Berechnungen.

In den folgenden Abschnitten wird ein kurzer Überblick über Grundlagen der Intervallarithmetik und -analysis gegeben, die für die folgenden Betrachtungen relevant sind. Für eine ausführliche Einführung in das Thema sei an dieser Stelle auf die Bücher [Alefeld, Herzberger '74] und [Neumeier '90] verwiesen, die die Basis für die folgenden Abschnitte bildeten.

1.1.1 Definitionen und Bezeichnungen

Ein *abgeschlossenes reelles Intervall I* ist definiert als

$$[a, b] := \{x \in \mathbb{R} \mid a \leq x \leq b\}$$

Die Menge aller abgeschlossenen reellen Intervalle wird im Folgenden mit $\mathbb{IR}$ bezeichnet.

Sei $I = [a, b] \in \mathbb{IR}$, dann bezeichnet

$\inf(I)$	$:= a$	das *Infimum* von I
$\sup(I)$	$:= b$	das *Supremum* von I
$rad(I)$	$:= \frac{(b-a)}{2}$	den *Radius* von I
$mid(I)$	$:= \frac{(a+b)}{2}$	den *Mittelpunkt* von I
$\|I\|$	$:= \max\{a, b\}$	den *Betrag* von I.

Ein Intervall I heißt *dünn*, wenn $\inf(I) = \sup(I)$.

Sei M eine nichtleere beschränkte Teilmenge aus $\mathbb{R}$, dann wird mit

$$\Box M := [\inf\{M\}, \sup\{M\}]$$

die *Hülle* von M bezeichnet.

1.1.2 Relationen

Zwei Intervalle I und J heißen gleich ($I = J$), wenn zwischen ihnen mengentheoretische Gleichheit besteht. Die Relation "=" ist reflexiv, transitiv und symmetrisch.

Die Relationen $<, \leq, >, \geq$ sind nur für Intervalle definiert, die sich höchstens einen Randpunkt gemeinsam haben.

1.1.3 Arithmetische Operationen

Ist $* \in \{+, -, \cdot, /\}$, dann gilt für $I, J \in \mathbb{IR}$

$$I * J = \{z = x * y \mid x \in I, y \in J\}$$

Explizit ergeben sich für $I = [a, b], J = [c, d]$ folgende Rechenregeln:

$I + J$	$:= [a + c, b + d]$
$I - J$	$:= [a - d, b - c]$
$I \cdot J$	$:= [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}]$
$I \ / \ J$	$:= [a, b] \cdot [\frac{1}{d}, \frac{1}{c}]$

Für die Division wird hierbei vorausgesetzt, daß $0 \notin J$.

1.1.4 Vektoren und Matrizen

Für *Intervallvektoren* ($\mathbb{x}, \mathbb{y}, \mathbb{I}, \mathbb{J}, ... \in \mathbb{IR}^n$) und *Intervallmatrizen* ($\mathbb{A}, \mathbb{B}, \mathbb{M}, ... \in \mathbb{IR}^{m \times n}$) gelten alle obigen Aussagen komponentenweise.

1.1.5 Intervallauswertung arithmetischer Ausdrücke

Die Rechenregeln für die intervallmäßige Auswertung stetiger Grundfunktionen wie Betrag, Wurzel, Quadrat, Sinus, Cosinus, die Exponentialfuntion und der Logarithmus ergeben sich direkt aus der Betrachtung des (bekannten) Wertebereichs und ihren Monotonieeigenschaften:
Sei ϕ eine der oben genannten Funktionen und $I \in \mathbb{IR}$, so gilt

$$\phi(I) := \square\{\phi(x) \mid x \in I\} = \{\phi(x) \mid x \in I\}$$

Satz 1.1.1 (Einschließungseigenschaft [Alefeld, Herzberger '74])
Sei $f(u_1, ..., u_n)$ eine stetige Funktion in $u_i \in U_i \in \mathbb{IR}$. Dann gilt für alle $(u_1, ..., u_n) \in \prod_{i=1}^{n} U_i$ $f(u_1, ..., u_n) \in f(U_1, ..., U_n)$.

Satz 1.1.2 ([Alefeld, Herzberger '74])
Gegeben sei ein reelles Polynom p der reellen Veränderlichen x. Dieses sei dargestellt durch den Ausdruck

$$p(x) = (...((a_m x + a_{m-1})^{n_{m-1}} + a_m - 2)^{n_{m-2}} + ...a_1)^{n_1} + a_0$$

mit $n_\nu \geq 2,\ 1 \leq \nu \leq m-1$.

Dann gilt bei der Auswertung der auftretenden Potenzen als

$$X^k := [\min_{x \in X} x^k, \max_{x \in X} x^k]$$

stets $W(p, X) = p(X)$.

Für allgemeine arithmetische Ausdrücke gilt:

Satz 1.1.3 ([Neumeier '90])
Sei $f(\boldsymbol{x}); \boldsymbol{x} := (x_1, ..., x_n) \in \mathbb{I} \in \mathbb{IR}^n$ ein arithmetischer Ausdruck, in dem jedes $x_i, i = 1, ..., n$ nur einmal vorkommt. Dann gilt

$$f(\mathbb{J}) = \square\{f(x) \mid x \in \mathbb{J}\} \text{ für alle } \mathbb{J} \subseteq \mathbb{I}$$

Satz 1.1.3 wird in [Stolfi, de Figueiredo '97] auch *Fundamentalinvariante der Wertebereichsanalye* genannt.

Die direkte Intervallauswertung arithmetischer Ausdrücke führt nur in wenigen Spezialfällen zu einer optimalen Einschließung des Wertebereichs. Ein großes Problem bilden bei der Auswertung auftretende Überschätzungen. In Fällen, in denen eine Variable mehrmals im Ausdruck auftritt, hängt die Qualität der Einschließung schon oft entscheidend von der Art ab, in welcher Form der Ausdruck geschrieben wird, da die Intervallarithmetik keine Abhängigkeiten der Variablen berücksichtigt. Betrachtet man beispielsweise die Funktion $f(x) = x - x; x \in [-1, 1]$, so gilt $W_f = \{0\}$; die intervallmäßige Auswertung von f überschätzt das tatsächliche Ergebnis sehr

grob: $f([-1,1]) = [-1,1] - [-1,1] = [-2,2]$.

Einige Methoden, um engere Einschließungen für Wertebereiche einer Funktion $f(x), x \in I$ zu bekommen, werden in [Neumeier '90], Kapitel 2, basierend auf Lipschitzkonstanten, dem Mittelwertsatz $f(x) = f(mid(I)) + f'(x)(x - mid(x))$, *slopes* oder allgemeiner auf den sogenannten *centered forms* $f(x) = f(\tilde{z}) + s(x - \tilde{z})$ mit Zentrum $\tilde{z}$ und Steigung s, beschrieben.

1.1.6 Lineare Intervallgleichungssysteme

Eine *lineare Intervallgleichung* mit der Koeffizientenmatrix $\mathbb{A} \in \mathbb{IR}^{m \times n}$ und der rechten Seite $\mathbb{b} \in \mathbb{IR}^m$ ist als die Familie linearer Gleichungen

$$A\boldsymbol{b} = \boldsymbol{x}; \qquad A \in \mathbb{A}, \boldsymbol{b} \in \mathbb{b} \tag{1.1}$$

definiert (siehe z.B. [Neumeier '90]). Die Menge

$$\Sigma(\mathbb{A}, \mathbb{b}) := \{\boldsymbol{x} \in \mathbb{R}^n \mid A\boldsymbol{x} = \boldsymbol{b} \text{ für ein } A \in \mathbb{A} \text{ und } \boldsymbol{b} \in \mathbb{b}\} \tag{1.2}$$

heißt *Lösungsmenge* von (1.1).

Ist A regulär, so ist $\Sigma(\mathbb{A}, \mathbb{b})$ beschränkt und die Hülle

$$\mathbb{A}^H\mathbb{b} := \Box\Sigma(\mathbb{A}, \mathbb{b}) \tag{1.3}$$

ist definiert.

Satz 1.1.4 ([Neumeier '90])
Ist $\mathbb{A}$ regulär und dünn, so ist die Hülle der Lösungsmenge mit der Lösungsmenge identisch:

$$\mathbb{A}^H\mathbb{b} = \mathbb{A}^{-1}\mathbb{b}$$

Aus Satz 1.1.4 und

$$A^{-1} = \frac{1}{|A|} \begin{pmatrix} |\boldsymbol{e}_1\ \boldsymbol{a}_2\ \boldsymbol{a}_3| & |\boldsymbol{e}_2\ \boldsymbol{a}_2\ \boldsymbol{a}_3| & |\boldsymbol{e}_3\ \boldsymbol{a}_2\ \boldsymbol{a}_3| \\ |\boldsymbol{a}_1\ \boldsymbol{e}_1\ \boldsymbol{a}_3| & |\boldsymbol{a}_1\ \boldsymbol{e}_2\ \boldsymbol{a}_3| & |\boldsymbol{a}_1\ \boldsymbol{e}_3\ \boldsymbol{a}_3| \\ |\boldsymbol{a}_1\ \boldsymbol{a}_2\ \boldsymbol{e}_1| & |\boldsymbol{a}_1\ \boldsymbol{a}_2\ \boldsymbol{e}_2| & |\boldsymbol{a}_1\ \boldsymbol{a}_2\ \boldsymbol{e}_3| \end{pmatrix}$$

folgt unmittelbar

Satz 1.1.5
Sei $A := (\boldsymbol{a}_i) \in \mathbb{R}^{3\times3}$; $\boldsymbol{a}_i \in \mathbb{R}^3$ regulär und $\mathbb{b} \in \mathbb{IR}^3$. Dann ist die Lösungsmenge des Gleichungssystems (1.1) gerade

$$\mathbb{A}^H\mathbb{b} = \mathbb{A}^{-1}\mathbb{b} = \frac{1}{|A|} \begin{pmatrix} |\mathbb{b}\ \boldsymbol{a}_2\ \boldsymbol{a}_3| \\ |\boldsymbol{a}_1\ \mathbb{b}\ \boldsymbol{a}_3| \\ |\boldsymbol{a}_1\ \boldsymbol{a}_2\ \mathbb{b}| \end{pmatrix}$$

falls die Determinanten $|\mathbb{b}\ \boldsymbol{a}_2\ \boldsymbol{a}_3|$, $|\boldsymbol{a}_1\ \mathbb{b}\ \boldsymbol{a}_3|$ und $|\boldsymbol{a}_1\ \boldsymbol{a}_2\ \mathbb{b}|$ jeweils so berechnet werden, daß sie nach den Komponenten des Vektors $\mathbb{b}$ entwickelt werden.

1.1.7 Implementierungen

- XSC-Sprachen: Intervallbibliotheken der Universität Karsruhe, die in Fortran, C und C++ erhältlich sind
- BIAS/Profil: C++ Intervallbibliothek der TU Hamburg-Harburg
- ...

1.2 Affine Arithmetik [Andrade et al. '94]

Um die durch die Intervallarithmetik verursachten Überschätzungen zu reduzieren, wurde die 1994 in [Andrade et al. '94] erstmals vorgestellte affine Arithmetik entwickelt. Affine Arithmetik liefert, genau wie die Intervallarithmetik, automatisch Approximation- und Rundungsfehler für jede berechnete Größe, beachtet aber zusätzlich noch die Abhängigkeiten der Eingabegrößen. Affine Arithmetik ist wesentlich komplexer als Intervallarithmetik und somit auch teurer in Bezug auf die Rechenzeit. Je nach Anwendung kann dies aber durch die höhere Genauigkeit der Ergebnisse ausgeglichen werden.

Die folgende Definition der Affinform weicht von den in [Andrade et al. '94] und [Stolfi, de Figueiredo '97] gegebene Definitionen ab.

1.2.1 Definitionen und Bezeichnungen

Eine *Affinform* ist definiert als

$$\hat{x} = \hat{x}(\epsilon_1, ..., \epsilon_n) := x_0, +x_1\epsilon_1 + x_2\epsilon_2 + ... + x_n\epsilon_n; \quad \epsilon_i \in [-1, 1]$$

wobei die x_i bekannte reelle Koeffizienten darstellen und die ϵ_i Variablen sind, die für unabhängige Fehler- oder Unsicherheitsquelle stehen, die ihren Ursprung sowohl in Inputdaten, als auch in Rundungs- und Approximationsfehlern haben können. x_0 heißt *Zentrum* der Affinform. Die $x_i;\ i = 1, ..., n$ heißen *partielle Abweichungen*, die $\epsilon_i;\ i = 1, ..., n$ *Fehlersymbole* der Affinform.

Kurzschreibweise: $\hat{x} = (x_0, ..., x_n)$.

Der Wertebereich einer Affinform repräsentiert also ähnlich einem Intervall einen unsicheren Wert x, allerdings mit dem Unterschied, daß die Unsicherheiten, von denen der Wert abhängt sehr viel genauer aufgeschlüsselt sind, als das bei einem Intervall der Fall ist.

Im Folgenden wird der Begriff *Affinform* mit dem Wertebereich der oben definierten Affinform gleichgesetzt.

1.2.2 Konvertierung Intervallarithmetik/Affine Arithmetik

Sei $\hat{x} := x_0, +x_1\epsilon_1 + x_2\epsilon_2 + ... + x_n\epsilon_n$ eine Affinform. Dann gilt für alle $x \in \hat{x}$

$$x \in X := [x_0 - \sum_{i=1}^{n} |x_i|,\ x_0 + \sum_{i=1}^{n} |x_i|]$$

X ist das kleinste Intervall, daß alle möglichen Werte von x enthält.

Umgekehrt gilt: Sei $Y \in \mathbb{IR}$. Dann ist

$$\hat{y} = y_0 + y_k \epsilon_k$$

mit $y_0 := mid(Y)$; $y_k := rad(Y)$ die Y repräsentierende Affinform.

1.2.3 Affine Operationen

Seien $\hat{x} = (x_0, ..., x_n)$ und $\hat{y} = (y_0, ..., y_n)$ zwei Affinformen bezüglich der gleichen Fehlersymbole $\epsilon_0, ..., \epsilon_n$ und sei $\alpha \in \mathbb{R}$. Dann gelte

1. $\hat{x} \pm \hat{y} := (x_0 \pm y_0, ..., x_n \pm y_n)$
2. $\alpha\hat{x} := (\alpha\, x_0, ..., \alpha\, x_n)$
3. $\hat{x} + \alpha := (x_0 \pm \alpha, x_1, ..., x_n)$

1.2.4 Nicht affine Operationen

Allgemeine Vorgehensweise:

1. Unterteile die Operation (falls möglich) in einen affinen Teil und in einen nicht affinen Teil.
2. Berechne den affinen Teil wie im vorigen Abschnitt beschrieben.
3. Berechne eine affine (Best-)Approximation (z.B. eine Tschebycheffapproximation) für den nicht-affinen Teil, multipliziere den Approximationsfehler mit einem neuen Fehlersymbol und addiere ihn zur Affinform.

Da eine tatsächliche Tschebycheffapproximation aufgrund ihrer Geometrie bei der Umwandlung der berechneten Affinform in ein Intervall teilweise unerwünschte Überschätzungen liefert, wird in [Stolfi, de Figueiredo '97] empfohlen, die affine Approximation mittels einer Min-Range-Approximation zu berechnen.

Beispiele für die Berechnung der Affinformen mit eine Min-Range Approximation

Seien $\hat{x} = (x_0, ..., x_n)$ und $\hat{y} = (y_0, ..., y_n)$ zwei Affinformen bezüglich der gleichen Fehlersymbole $\epsilon_0, ..., \epsilon_n$ und sei $\alpha \in \mathbb{R}$.

1. **Multiplikation:**

$$\hat{x} * \hat{y} = (x_0 y_0, x_0 y_1 + y_0 x_1, ..., x_0 y_n + y_0 x_n, (\sum_{i=0}^{n} |x_i|)(\sum_{i=0}^{n} |y_i|))$$

2. **Quadratwurzel:**

$$\sqrt{\hat{x}} = (\alpha x_0 + \beta, \alpha x_1, ..., \alpha x_n, \delta)$$

mit

$$\alpha := \frac{1}{\sqrt{b} + \sqrt{a}}; \quad \beta := \frac{\sqrt{b} + \sqrt{a}}{8} + \frac{1}{2}\frac{\sqrt{a}\sqrt{b}}{\sqrt{b} + \sqrt{a}}; \quad \delta := \frac{1}{8}\frac{(\sqrt{b} - \sqrt{a})^2}{\sqrt{b} + \sqrt{a}}$$

Die Behandlung von Rundungsfehlern

Für Rundungsfehler wird ein neues Fehlersymbol eingeführt, zu dessen zugehöriger partieller Abweichung die oberen Schranken aller auftretender Rundungsfehler addiert werden.

1.2.5 Implementierungen

- Affine Arithmetic Package in C von de Figueiredo. Hier fehlt allerdings die Implementierung der Grundfunktionen (Sinus, Cosinus, Tangens,...)
- C++ Bibliothek von Ronald van Iwaarden, die auf Grund der verwendeten statischen Datenstrukturen ineffizient ist.

1.3 Taylormodelle, [Berz, Hofstätter '98]

Taylormodelle sind ein weiterer Versuch, die durch reine Intervallarithmetik verursachten Überschätzungen zu reduzieren. Dazu übertrug Berz das seit langem angewandte Verfahren, Ableitungen einer Funktion effizient und beliebig genau über deren Taylorentwicklung zu bestimmen, auf die Auswertung komplizierterer Ausdrücke. Es zeigt sich, daß die Überschätzungen, die bei einer direkten Intervallauswertung auftreten, wesentlich reduziert werden können, wenn der Ausdruck als Taylorpolynom beliebigen Grades dargestellt und ausgewertet wird.

1.3.1 Definitionen und Grundlagen

Grundlage für die Entwicklung der Theorie der Taylormodelle bildet der Satz von Taylor:

Satz 1.3.1 (z.B. [Harro Heuser '88a])
Sei $f \in C^{(n)}(I)$, $I = (x_0 - \alpha, x_0 + \alpha) \in I\mathbb{R}$ mit existierender $n+1$. Ableitung auf I. Dann existiert für alle $x \in I$ ein $\xi \in I$, so daß

$$f(x) = T(x) + R_n(x)$$

wobei

$$T_n(x) := \sum_{\nu=0}^{n} \frac{f^{(\nu)}(x_0)}{\nu!}(x - x_0)^{\nu}$$

das n-te Taylorpolynom von f an der Stelle x_0 und

$$R_n(x) := \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1}$$

Lagranges Restglied *der Taylorformel heißt.*

Für den multivarianten und speziell den bivarianten Fall siehe z.B. [Harro Heuser '88b].

Sei also $f \in C^{n+1}(D_f)$; $D_f \subset \mathbb{R}^m$ und $D_f \subseteq B \in I\mathbb{R}^n$. Desweiteren bezeichne $T_n(x)$ das n-te Taylorpolynom von f um den Punkt $\boldsymbol{x}_0 \in B$. Dann heißt:

- das Intervall I mit

  $$\forall \boldsymbol{x} \in B : f(\boldsymbol{x}) - T_n(\boldsymbol{x}) \in I$$

  Restgliedabschätzung n-ter Ordnung von f auf B.

- das Paar (T_n, I) *Taylormodell n. Ordnung von f*

- die Menge aller Restgliedabschätzungen *Restgliedschar*

Da die $n+1$. Ableitung als stetig vorausgesetzt wurde und B kompakt ist, ist die Restgliedabschätzung immer beschränkt und es existiert eine optimale Abschätzung, denn:
Sei T_n das n.-te Taylorpolynom von f und $g := f - T_n$. Dann ist g stetig und besitzt zwei Extrema für x_l und x_u. und es ist offensichtlich, daß $I := [g(x_l), g(x_u)]$ ein Element der Restgliedschar ist, das in jeder anderen Restgliedabschätzung enthalten ist. I wird *optimale Restgliedabschätzung* genannt.

1.3.2 Arithmetische Ausdrücke

Ein Taylormodell für einen komplexeren Ausdruck kann sukzessive aus Taylormodellen für Grundfunktionen abgeleitet werden. Dazu entwickelte Berz eine Art Taylorarithmetik, die wie im Falle der Intervallrechnung auf Mengenoperationen basiert.

Seien im Folgenden (T_f, I_f) und (T_g, I_g) Taylormodelle n.-ter Ordnung der Funktionen f und g die jeweils über $B \in \mathbb{IR}^n$ definiert seien.

Addition/Subtraktion:

$$(T_{f\pm g}, I_{f\pm g}) := (T_f \pm T_g, I_f \pm I_g)$$

Multiplikation:

Für jedes $\boldsymbol{x} \in B$ existieren $e_f \in I_f$ und $e_g \in I_g$ so, daß $f(\boldsymbol{x}) = T_f(\boldsymbol{x}) + e_f$ und $f(\boldsymbol{x}) = T_g(\boldsymbol{x}) + e_g$. Dann ist die Multiplikation der beiden Taylormodelle gegeben durch

$$(T_{fg}, I_{fg}) := (T_f T_g - \overline{T}_{fg}, \Box E(\boldsymbol{x}, e_f, e_g))$$

wobei $\overline{T}_{fg}$ die Terme des Produktes $T_f T_g$ enthält, deren Grad höher als n ist. Weiter sei $E(\boldsymbol{x}, e_f, e_g) := \overline{T}_{fg} + T_f e_g + T_g e_f + e_f e_g$ und $\Box E(\boldsymbol{x}, e_f, e_g)$ bezeichne die Hülle von E auf $B \times I_f \times I_g$.
Die Qualität der Berechnung hängt stark von der Qualität der Einschließung des Polynoms E ab. Scharfe Einschließungen, z.B. mittels einer Entwicklung des Polynoms nach Tschebycheff- oder Bézierpolynonem und der Einschließung der Tschebycheff-/Bézierpunkte zu berechnen, ist kostspielig. Eine Auswertung mittels Hornerschema ist die schnellste Möglichkeit, macht aber das Ergebnis unscharf.

1.3.3 Grundfunktionen

Zur Berechnung einer der Grundfunktionen Sinus, Cosinus, Exp, usw. mit einem Taylormodell als Argument existiert keine allgemeine Strategie. In den meisten Fällen kann aber folgendermaßen vorgegangen werden:

Sei (T_f, I_f) ein Taylormodell n.-ter Ordnung für f über B und sei g eine der Grundfunktionen. Gesucht ist das Taylormodell für die Funktion $f \circ g$.

Ansatz:

1. $c := T_f(\boldsymbol{x}_0), \qquad \tilde{T}_f(\boldsymbol{x}) := T_f(\boldsymbol{x}) - c$
2. Verwende eine Additionstheorem für g (z.B. $exp(a+b) = exp(a)exp(b)$) um die Berechnung von c und $\tilde{T}_f(\boldsymbol{x}) + I_f$ zu trennen.
3. Berechne den konstanten Teil korrekt oder eine enge Einschließung desselben.
4. Berechne das Taylormodell für den nicht-konstanten Teil der Funktion $h(\tilde{T}_f(\boldsymbol{x}) + I_f)$. Das Restglied wird mit Standardintervallarithmetik ausgewertet.

Anhang B

Bibliographie Schnittalgorithmen

2.1 Verfolgungsalgorithmen

2.1.1 Startpunkte

Gregor Müllenheim (1991). On Determining Start Points for a Surface/Surface Intersection Algorithm. *Computer Aided Geometric Design*, 8(5):401–408

Karim Abdel-Malek, Harn-Jou Yeh (1997). On the determination of starting points for parametric surface intersections. *Computer Aided Design*, 29(1):21–35

2.1.2 Loop detection

P. Sinah, E. Klassen, K.K. Wang (1985). Exploiting topological and geometric properties for selective subdivision. In *Proc. ACM Symposium on Computational Geometry*, S. 39–45, 1985

Thomas W. Sederberg, Ray J. Meyers (1988). Loop detection in surface patch intersections. *Computer Aided Geometric Design*, 5(2):161–171

T.W. Sederberg, H. Christiansen, S. Katz (1989). Improved test for closed loops in surface intersections. *Computer Aided Design*, 21(8):505–508

Michael E. Hohmeyer (1991). A Surface Intersection Algorithm Based on Loop Detection. In **Jaroslaw Rossignac, Joshua Turner** (Hrsg.) (1991), *SMA '91: Proceedings of the First Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin, Texas, June 5-7*, S. 197–208. ACM Press, 1991

2.1.3 Verfolgung

Geometrische Ansätze

Tz.E. Stoyanov (1992). Marching Along Surface/Surface Intersection Curves with an Adaptive Step Length. *Computer Aided Geometric Design*, 9(6):485–490

Shin-Ting Wu, Lenimar N. Andrade (1999). Marching along a regular surface/surface intersection with circular steps. *Computer Aided Geometric Design*, 16:249–268

Taylorapproximation

Yves de Montaudouin, Wayne Tiller, Havard Vold (1986). Applications of power series in computational geometry. *Computer-Aided Design*, 18(10):514–524

Chandrajit L. Bajaj, Chris M. Hoffmann, Robert E. Lynch, John E. H. Hopcroft (1988). Tracing surface intersections. *Computer Aided Geometric Design*, 5(4):285–307

Sonstige

C. Asteasu, A. Orbegozo (1991). Parametric piecewise surfaces intersection. *Computers and Graphics*, 15(1):9–13

2.1.4 Parametrisierung der Schnittkurve

Robert Markot, Robert Magedson (1991). Procedural method for evaluating the intersection curves of two parametric surfaces. *Computer Aided Design*, 23(6):395–404

Chandrajit L. Bajaj, Guoliang Xu (1994). NURBS approximation of surface/surface intersection curves. *Adv. Comput. Math*, 2(1):1–21

2.1.5 Komplette Algorithmen

Robert E. Barnhill, Gerald Farin, Michael Jordan, Bruce R. Piper (1987). Surface/surface intersection. *Computer Aided Geometric Design*, 4(1–2):3–16

George A. Kriezis, Nicholas M. Patrikalakis, Franz-Erich Wolter (1992). Topological and differential equation methods for surface intersections. *Computer-Aided Design, 1992*, 24:41–55

2.1.6 Konvergenz von Verfolgungsalgorithmen

Gregor Müllenheim (1990). Convergence of a surface/surface intersection algorithm. *Computer Aided Geometric Design*, 7(5):415–423

2.2 Einbettungsmethoden

George A. Kriezis, Nicholas M. Patrikalakis, Franz-Erich Wolter (1992). Topological and differential equation methods for surface intersections. *Computer-Aided Design, 1992*, 24:41–55

Nicholas M. Patrikalakis (1993). Surface-to-Surface Intersections. *IEEE Computer Graphics and Applications*, 13(1):89–95

Nicholas M. Patrikalakis (1992). Interrogation of surface intersections. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 161–185. SIAM, Philadelphia, 1992

Peter Schramm (1995). *Sichere Verschneidung von Kurven und Flächen im CAGD.* Dissertation, Universität Karlsruhe, Fakultät für Mathematik, 1995

Karim Abdel-Malek, Harn-Jou Yeh (1996). Determining intersection curves between surfaces of two solids. *Computer Aided Design*, 28(6-7):539–549

Thomas A. Grandine, Frederick W. Klein IV (1997). A new approach to the surface intersection problem. *Computer Aided Geometric Design*, 14(2):111–134

2.3 Unterteilungsalgorithmen

2.3.1 Achsenparallele Bounding Boxen

E. G. Houghton, R. F. Emnett, J. D. Factor, Ch. L. Sabharwal (1985). Implementation of a Divide-and-Conquer Method for Intersection of Parametric Surfaces. *Computer Aided Geometric Design*, 2:173–183

Daniel Filip, Robert Magedson, Robert Markot (1986). Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design*, 3(4):295–311

Michael Gleicher, Michael Kass (1992). An interval refinement technique for surface intersection. In *Proceedings of the Graphics Interface '92, Vancouver, British Columbia, 11–15 May 1992*, S. 242–249, Toronto, Ont., Canada, 1992. Canadian Information Processing Society

Long Chyr Chang, Wolfgang W. Bein, Edward Angel (1994). Surface intersection using parallelism. *Computer Aided Geometric Design*, 11(1):39–69

Luiz Henrique de Figueiredo (1996). Surface Intersection Using Affine Arithmetic. In **Wayne A. Davis, Richard Bartels** (Hrsg.) (1996), *Proceedings of the Graphics Interface '96, Toronto, Ontario, 22-24 May*, S. 168–175. Canadian Information Processing Society, Canadian Human-Computer Communications Society, 1996

Atsushi Yamada, Fujio Yamaguchi (1996). Homogeneous bounding boxes as tools for intersection algorithms of rational beézier curves and surfaces. *The Visual Computer*, 12:202–214

2.3.2 Orientierte Bounding Boxen und Parallelepipede

Ronald N. Goldman, Tony D. DeRose (1986). Recursive subdivision without the convex hull property. *Computer Aided Geometric Design*, 3(4):247–265

Robert E. Barnhill, Scott Kersey (1990). A marching method for parametric surface/surface intersection. *Computer Aided Geometric Design*, 7(1-4):257–280

Heiko Bürger, Robert Schaback (1993). A parallel multistage method for surface/surface intersection. *Computer Aided Geometric Design*, 10(3):277–292

Ernst Huber (1999). *Ein Einschließungsalgorithmus zur sicheren Berechnung der vollständigen Berechnung der Schnittkurve zweier Flächen.* Dissertation, TU Wien, 1999

Ernst Huber, Wilhelm Barth (1999). Surface-to-Surface intersection with complete and guaranteed results. In **T. Csendes** (Hrsg.) (1999), *Developments in Reliable Computing*, S. 185–198. Kluwer Academic Publishers, 1999

Bézier Clipping:
Thomas W. Sederberg, Tomoyuki Nishita (1991). Geometric Hermite Approximation of Surface Patch Intersection Curves. *Computer Aided Geometric Design*, 8(2):97–114

2.3.3 Reduzierung des Parametergebietes

Ernst Huber (1999). *Ein Einschließungsalgorithmus zur sicheren Berechnung der vollständigen Berechnung der Schnittkurve zweier Flächen.* Dissertation, TU Wien, 1999
nachsehen! - auch andere Artikel?
Ernst Huber (1998). Intersecting General Parametric Surfaces Using Bounding Volumes. In *Tenth Canadian Conference on Computational Geometry - CCCG '98, Vancouver, Canada, August 15 – 18*, S. 52–54, 1998

Ernst Huber, Wilhelm Barth (1999). Surface-to-Surface intersection with complete and guaranteed results. In **T. Csendes** (Hrsg.) (1999), *Developments in Reliable Computing*, S. 185–198. Kluwer Academic Publishers, 1999

2.3.4 Konvergenz

Long Chyr Chang, Wolfgang W. Bein, Edward Angel (1994). Surface intersection using parallelism. *Computer Aided Geometric Design*, 11(1):39–69

2.3.5 Nur Bounding Boxen

S. Gopalsamy, D. Khandekar, S. P. Mudur (1991). A New Method of Evaluating Compact Geometric Bounds for Use in Subdivision Algorithms. *Computer Aided Geometric Design*, 8(5):337–356

Stefan Gottschalk, Ming Lin, Dinesh Manocha (1996). OBB-Tree: A Hierarchical Structure for Rapid Interference Detection. In **Holly Rushmeier** (Hrsg.) (1996), *Proceedings of Siggraph '96, New Orleans, Louisiana, August 4–9*, S. 171–180. ACM Siggraph, Addison Wesley, 1996

Gino van den Bergen (1999). A Fast and Robust GJK Implementation for Collision Detection of Convex Objects. *Journal of Graphics Tools*, 4(2):7–25

2.3.6 Algorithmen die Unterteilungsalgorithmen für Kurven und Flächen verwenden

Ronald N. Goldman, Tony D. DeRose (1986). Recursive subdivision without the convex hull property. *Computer Aided Geometric Design*, 3(4):247–265

2.4 Algebraische Methoden

C. Asteasu (1988). Intersection of Arbitrary Surfaces. *Computer Aided Design*, 20(9):533–538

Thomas Garrity, Joe Warren (1989). On computing the intersection of a pair of algebraic surfaces. *Computer Aided Geometric Design*, 6:137–153

Christoph M. Hoffmann (1990). A dimensionality paradigm for surface interrogations. *Computer Aided Geometric Design*, 7(6):517–532

Dinesh Manocha, John F. Canny (1991). A New Approach to Surface Intersection. *Int. J. Computational Geometry and Applications*, 1(4):491–516

2.5 Diskretisierungsmethoden

K.Y. Wang (1992). Parametric surface intersections. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 187–204. SIAM, Philadelphia, 1992

John K. Johnston (1993). A new intersection algorithm for cyclides and swept surfaces using circle decomposition. *Computer Aided Geometric Design*, 10:1–24

2.6 Konstruktive Methoden

Les Piegl (1992). Constructive geometry approach to surface-surface intersection. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 137–159. SIAM, Philadelphia, 1992

2.7 Hybride Methoden

2.7.1 Algebraische und Verfolgungsmethoden

Shankar Krishnan, Dinesh Manocha (1997). An Efficient Surface Intersection Algorithm Based on Lower-Dimensional Formulation. *ACM Transactions on Graphics*, 16(1):74–106

2.7.2 Unterteilungs- und Verfolgungsalgorithmen

Robert E. Barnhill, Scott Kersey (1990). A marching method for parametric surface/surface intersection. *Computer Aided Geometric Design*, 7(1-4):257–280

Michael E. Hohmeyer (1991). A Surface Intersection Algorithm Based on Loop Detection. In **Jaroslaw Rossignac, Joshua Turner** (Hrsg.) (1991), *SMA '91: Proceedings of the First Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin, Texas, June 5-7*, S. 197–208. ACM Press, 1991

Heiko Bürger, Robert Schaback (1993). A parallel multistage method for surface/surface intersection. *Computer Aided Geometric Design*, 10(3):277–292

Pramod Koparkar (1991). Surface intersection by switching from recursive subdivision to iterative refinement. *The Visual Computer*, 8:47–63

2.7.3 Subdivision und Hermite Approximation

Thomas W. Sederberg, Tomoyuki Nishita (1991). Geometric Hermite Approximation of Surface Patch Intersection Curves. *Computer Aided Geometric Design*, 8(2):97–114

2.8 Schnittalgorithmen, die Interval- bzw. affine Arithmetik verwenden

2.8.1 Artikel, die Teilprobleme mit Intervallarithmetik lösen

Loop detection

T.W. Sederberg, H. Christiansen, S. Katz (1989). Improved test for closed loops in surface intersections. *Computer Aided Design*, 21(8):505–508

Thomas W. Sederberg, Tomoyuki Nishita (1991). Geometric Hermite Approximation of Surface Patch Intersection Curves. *Computer Aided Geometric Design*, 8(2):97–114

Verfolgung impliziter Kurven

John M. Snyder (1992). Interval Analysis For Computer Graphics. *Computer Graphics*, 26(2):121–130

2.8.2 Komplette Schnittalgorithmen die auf verifiziertem Rechnen beruhen

Michael Gleicher, Michael Kass (1992). An interval refinement technique for surface intersection. In *Proceedings of the Graphics Interface '92, Vancouver, British Columbia, 11–15 May 1992*, S. 242–249, Toronto, Ont., Canada, 1992. Canadian Information Processing Society

Luiz Henrique de Figueiredo (1996). Surface Intersection Using Affine Arithmetic. In **Wayne A. Davis, Richard Bartels** (Hrsg.) (1996), *Proceedings of the Graphics Interface '96, Toronto, Ontario, 22-24 May*, S. 168–175. Canadian Information Processing Society, Canadian Human-Computer Communications Society, 1996

Peter Schramm (1995). *Sichere Verschneidung von Kurven und Flächen im CAGD.* Dissertation, Universität Karlsruhe, Fakultät für Mathematik, 1995

Chun-Yi Hu, Takashi Maekawa, Nicholas M. Patrikalakis, Xiuzi Ye (1997). Robust interval algorithm for surface intersections. *Computer Aided Design*, 29(9):617–627

Ernst Huber, Wilhelm Barth (1999). Surface-to-Surface intersection with complete and guaranteed results. In **T. Csendes** (Hrsg.) (1999), *Developments in Reliable Computing*, S. 185–198. Kluwer Academic Publishers, 1999

2.9 Überblicke

Nicholas M. Patrikalakis (1992). Interrogation of surface intersections. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 161–185. SIAM, Philadelphia, 1992

Josef Hoschek, Dieter Lasser (1992). *Grundlagen der geometrischen Datenverarbeitung.* B.G. Teubner Stuttgart, 2 Auflage

Gerald Farin (1992). An SSI Bibliography. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 205–207. SIAM, Philadelphia, 1992

Nicholas M. Patrikalakis (1993). Surface-to-Surface Intersections. *IEEE Computer Graphics and Applications*, 13(1):89–95

Anhang C

Bibliographie Arithmetiken

3.1 Intervallarithmetik

3.1.1 Grundlagen

Götz Alefeld, Jürgen Herzberger (1974). *Einführung in die Intervallrechnung.* Nummer 12 in Reihe Informatik. BI Wissenschaftsverlag

Arnold Neumeier (1990). *Interval Methods for Systems of Equations*, Band 37 von *Encyclopedia of Mathematics and its Applications.* Cambridge University Press

3.1.2 Anwendungen

Computergraphik

Chun Yi Hu, Nicholas M. Patrikalakis, Xiuz Ye (1996a). Robust interval solid modelling Part 1: representations. *Computer Aided Design*, 28(10):807–817

Chun Yi Hu, Nicholas M. Patrikalakis, Xiuz Ye (1996b). Robust interval solid modelling Part 2: boundary evaluation. *Computer Aided Design*, 28(10):819–830

Thomas W. Sederberg, Rida T. Farouki (1992). Approximation by interval Bezier curves. *IEEE Computer Graphics and Applications*, 12(5):87–95

Tom Duff (1992). Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry. *Computer Graphics (Proceedings of Siggraph '92, Chicago, Illinois, July 26–31)*, 26(2):131–138

A. Bowyer, J. Berchtold, D. Eisenthal, I. Voiculescu, K. Wise (2000). Interval Methods in Geometric Modeling. In **Ralph Martin, Wenping Wang** (Hrsg.) (2000), *Proceedings of the Conference on Geometric Modeling and Processing (GMP-00), Los Alamitos, April 10–12,*

S. 321–327. IEEE, 2000

John M. Snyder, Adam R. Woodbury, Kurt Fleischer, Bena Currin, Alan H. Barr (1993). Interval methods for multi-point collisions between time-dependent curved surfaces. *Computer Graphics*, 27(Annual Conference Series):321–334

Nilo Stolte, Arie Kaufman (1998). Parallel Spatial Enumeration of Implicit Surfaces using Interval Arithmetic for Octree Generation and and its Direct Visualization. In *Proceedings of Implicit Surfaces'98, Seattle, June 15–16*, S. 81–88, 1998

Irina Voiculescu, Jakob Berchtold, Adrian Bowyer, Ralph R. Martin, Qijiang Zhang (2000). Interval and Affine Arithmetic for Surface Location of Power- and Bernstein-Form Polynomials. In **Roberto Cipolla, Ralph Martin** (Hrsg.) (2000), *The Mathematics of Surfaces IX*, S. 410–423, London, Berlin, Heidelberg, 2000. Springer

John M. Snyder (1992). Interval Analysis For Computer Graphics. *Computer Graphics*, 26(2):121–130

3.2 Affine Arithmetik

3.2.1 Grundlagen

Marcus Andrade, João Comba, Jorge Stolfi (1994). Affine Arithmetic. Submittet to Interval 94, 1994

Jorge Stolfi, Luiz Enrique de Figueiredo (1997). *Self-Validated Numerical Methods and Applications.* unpublished

Roland van Iwaarden, Jorge Stolfi (1997). Affine Arithmetic Sofware. 1997

3.2.2 Anwendungen

Optimierung

Luiz de Figueiredo, Ronald van Iwaarden, Jorge Stolfi (1997). Fast Interval Branch and Bound Methodes for Unconstrained Global Optimization with Affine Arithmetic. Submitted to SIAM Journal of Optimization, March 1997

Computergraphik

Luiz Henrique de Figueiredo, Jorge Stolfi (1996). Adaptive enumeration of implicit surfaces with affine arithmetic. *Computer Graphics Forum*, 15(5):287–296

Luiz Henrique de Figueiredo (1996). Surface Intersection Using Affine Arithmetic. In **Wayne A. Davis, Richard Bartels** (Hrsg.) (1996), *Proceedings of the Graphics Interface '96, Toronto, Ontario, 22-24 May*, S. 168–175. Canadian Information Processing Society, Canadian Human-Computer Communications Society, 1996

Affonso de Cusatis Junior, Luiz Henrique de Figueiredo, Marcello Gattas (1999). Interval methods for raycasting implicit surfaces with affine arithmetic. In *Proceedings of XII SIBGRPHI*, S. 1–7, 1999

Wolfgang Heidrich, Philip Slusallek, Hans-Peter Seidel (1998). Sampling procedural shaders using affine arithmetic. *ACM Transactions on Graphics*, 17(3):158–176

Wolfgang Heidrich, Hans-Peter Seidel (1998). Ray-tracing Procedural Displacement Shaders. In **Wayne Davis, Kellogg Booth, Alain Fourier** (Hrsg.) (1998), *Proceedings of the Graphics Interface '98, San Francisco, Juni 18–20*, S. 8–16. Morgan Kaufmann Publishers, 1998

Irina Voiculescu, Jakob Berchtold, Adrian Bowyer, Ralph R. Martin, Qijiang Zhang (2000). Interval and Affine Arithmetic for Surface Location of Power- and Bernstein-Form Polynomials. In **Roberto Cipolla, Ralph Martin** (Hrsg.) (2000), *The Mathematics of Surfaces IX*, S. 410–423, London, Berlin, Heidelberg, 2000. Springer

Literaturverzeichnis

Karim Abdel-Malek, Harn-Jou Yeh (1996). Determining intersection curves between surfaces of two solids. *Computer Aided Design*, 28(6-7):539–549.

Karim Abdel-Malek, Harn-Jou Yeh (1997). On the determination of starting points for parametric surface intersections. *Computer Aided Design*, 29(1):21–35.

Götz Alefeld, Jürgen Herzberger (1974). *Einführung in die Intervallrechnung.* Nummer 12 in Reihe Informatik. BI Wissenschaftsverlag.

Marcus Andrade, João Comba, Jorge Stolfi (1994). Affine Arithmetic. Submittet to Interval 94, 1994.

C. Asteasu, A. Orbegozo (1991). Parametric piecewise surfaces intersection. *Computers and Graphics*, 15(1):9–13.

C. Asteasu (1988). Intersection of Arbitrary Surfaces. *Computer Aided Design*, 20(9):533–538.

Günther Aumann (1993). Differentialgeometrie. Vorlesungsskript, Sommersemester 1993.

Chandrajit L. Bajaj, Chris M. Hoffmann, Robert E. Lynch, John E. H. Hopcroft (1988). Tracing surface intersections. *Computer Aided Geometric Design*, 5(4):285–307.

Chandrajit L. Bajaj, Guoliang Xu (1994). NURBS approximation of surface/surface intersection curves. *Adv. Comput. Math*, 2(1):1–21.

Robert E. Barnhill, Gerald Farin, Michael Jordan, Bruce R. Piper (1987). Surface/surface intersection. *Computer Aided Geometric Design*, 4(1–2):3–16.

Robert E. Barnhill, Scott Kersey (1990). A marching method for parametric surface/surface intersection. *Computer Aided Geometric Design*, 7(1-4):257–280.

Gino van den Bergen (1999). A Fast and Robust GJK Implementation for Collision Detection of Convex Objects. *Journal of Graphics Tools*, 4(2):7–25.

Martin Berz, Georg Hofstätter (1998). Computation and Application of Taylor Polynomials with Interval Remainder Bounds. *Reliable Computing*, 4:83–97.

Katja Bühler (2001). Über den Schnitt zweier Linearer Intervallabschätzungen. Technischer Bericht TR-186-2-01-07, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Karlsplatz 13/186/2, A-1040 Vienna, Austria, 2001.

A. Bowyer, J. Berchtold, D. Eisenthal, I. Voiculescu, K. Wise (2000). Interval Methods in Geometric Modeling. In **Ralph Martin, Wenping Wang** (Hrsg.) (2000), *Proceedings of the Conference on Geometric Modeling and Processing (GMP-00), Los Alamitos, April 10–12*, S. 321–327. IEEE, 2000.

Katja Bühler, Wilhelm Barth (2000). Intersecting Parametric Surfaces Using Special Arithmetics. In *Extended Abstract, Proceedings of the Alhambra 2000: A Joint Mathematical European-Arabis Conference, Granada, Spain, July 3–7*, 2000.

Katja Bühler, Wilhelm Barth (2001). A new intersection algorithm for parametric surfaces based on linear interval estimations. In *Scientific Computing, Validated Numerics, Interval Methods*. Kluwer Academic Publishers, 2001.

Katja Bühler (1995). Rationale algebraische Kurven auf Dupinschen Zykliden. Diplomarbeit, Universität Karlsruhe, 1995.

Katja Bühler (2001a). Linear Interval Estimations for Implicit Curves and Surfaces. Extended Abstract. Accepted at the Workshop *Uncertainty in Geometric Computations*, The University of Sheffield 5th - 6th July, 2001.

Katja Bühler (2001b). Linear Interval Estimations for Parametric Objects - Theory and Applications. *Computer Graphics Forum*, 20(3).

Katja Bühler (2001c). Taylor Models and Affine Arithmetics - Towards a More Sophisticated Use of Reliable Methods in Computer Graphics. In *Proceedings of the Seventeenth Spring Conference on Computer Graphics (SCCG), Budmerice, Slovakia, April 25–28*, Bratislava, Slovakia, April 2001. Comenius University Press.

Heiko Bürger, Robert Schaback (1993). A parallel multistage method for surface/surface intersection. *Computer Aided Geometric Design*, 10(3):277–292.

Long Chyr Chang, Wolfgang W. Bein, Edward Angel (1994). Surface intersection using parallelism. *Computer Aided Geometric Design*, 11(1):39–69.

Affonso de Cusatis Junior, Luiz Henrique de Figueiredo, Marcello Gattas (1999). Interval methods for raycasting implicit surfaces with affine arithmetic. In *Proceedings of XII SIBGRPHI*, S. 1–7, 1999.

Luiz Henrique de Figueiredo, Jorge Stolfi (1996). Adaptive enumeration of implicit surfaces with affine arithmetic. *Computer Graphics Forum*, 15(5):287–296.

Luiz Henrique de Figueiredo (1996). Surface Intersection Using Affine Arithmetic. In **Wayne A. Davis, Richard Bartels** (Hrsg.) (1996), *Proceedings of the Graphics Interface '96, Toronto, Ontario, 22-24 May*, S. 168–175. Canadian Information Processing Society, Canadian Human-Computer Communications Society, 1996.

Tom Duff (1992). Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry. *Computer Graphics (Proceedings of Siggraph '92, Chicago, Illinois, July 26–31)*, 26(2):131–138.

Gerald Farin (1992). An SSI Bibliography. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 205–207. SIAM, Philadelphia, 1992.

Luiz de Figueiredo, Ronald van Iwaarden, Jorge Stolfi (1997). Fast Interval Branch and Bound Methodes for Unconstrained Global Optimization with Affine Arithmetic. Submitted to SIAM Journal of Optimization, March 1997.

Daniel Filip, Robert Magedson, Robert Markot (1986). Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design*, 3(4):295–311.

Aain Fournier, John Buchanan (1994). Chebyshev Polynomials for Boxing and Intersections of Parametric Curves and Surfaces. *Computer Graphics Forum*, 13(3):C/127–C/142.

Thomas Garrity, Joe Warren (1989). On computing the intersection of a pair of algebraic surfaces. *Computer Aided Geometric Design*, 6:137–153.

Michael Gleicher, Michael Kass (1992). An interval refinement technique for surface intersection. In *Proceedings of the Graphics Interface '92, Vancouver, British Columbia, 11–15 May 1992*, S. 242–249, Toronto, Ont., Canada, 1992. Canadian Information Processing Society.

Ronald N. Goldman, Tony D. DeRose (1986). Recursive subdivision without the convex hull property. *Computer Aided Geometric Design*, 3(4):247–265.

S. Gopalsamy, D. Khandekar, S. P. Mudur (1991). A New Method of Evaluating Compact Geometric Bounds for Use in Subdivision Algorithms. *Computer Aided Geometric Design*, 8(5):337–356.

Stefan Gottschalk, Ming Lin, Dinesh Manocha (1996). OBB-Tree: A Hierarchical Structure for Rapid Interference Detection. In **Holly Rushmeier** (Hrsg.) (1996), *Proceedings of Siggraph '96, New Orleans, Louisiana, August 4–9*, S. 171–180. ACM Siggraph, Addison Wesley, 1996.

Thomas A. Grandine, Frederick W. Klein IV (1997). A new approach to the surface intersection problem. *Computer Aided Geometric Design*, 14(2):111–134.

Harro Heuser (1988a). *Lehrbuch der Analysis, Teil 1.* Teubner Verlag, Stuttgart, 5 Auflage.

Harro Heuser (1988b). *Lehrbuch der Analysis, Teil 2.* Teubner Verlag, Stuttgart, 4 Auflage.

Wolfgang Heidrich, Philip Slusallek, Hans-Peter Seidel (1998). Sampling procedural shaders using affine arithmetic. *ACM Transactions on Graphics*, 17(3):158–176.

Wolfgang Heidrich, Hans-Peter Seidel (1998). Ray-tracing Procedural Displacement Shaders. In **Wayne Davis, Kellogg Booth, Alain Fourier** (Hrsg.) (1998), *Proceedings of the Graphics Interface '98, San Francisco, Juni 18–20*, S. 8–16. Morgan Kaufmann Publishers, 1998.

Christoph Martin Hoffmann (1989). *Geometric and Solid Modeling.* Morgan Kaufmann.

Christoph M. Hoffmann (1990). A dimensionality paradigm for surface interrogations. *Computer Aided Geometric Design*, 7(6):517–532.

Michael E. Hohmeyer (1991). A Surface Intersection Algorithm Based on Loop Detection. In **Jaroslaw Rossignac, Joshua Turner** (Hrsg.) (1991), *SMA '91: Proceedings of the First Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin, Texas, June 5-7*, S. 197–208. ACM Press, 1991.

Josef Hoschek, Dieter Lasser (1992). *Grundlagen der geometrischen Datenverarbeitung.* B.G. Teubner Stuttgart, 2 Auflage.

E. G. Houghton, R. F. Emnett, J. D. Factor, Ch. L. Sabharwal (1985). Implementation of a Divide-and-Conquer Method for Intersection of Parametric Surfaces. *Computer Aided Geometric Design*, 2:173–183.

Chun Yi Hu, Nicholas M. Patrikalakis, Xiuz Ye (1996a). Robust interval solid modelling Part 1: representations. *Computer Aided Design*, 28(10):807–817.

Chun Yi Hu, Nicholas M. Patrikalakis, Xiuz Ye (1996b). Robust interval solid modelling Part 2: boundary evaluation. *Computer Aided Design*, 28(10):819–830.

Chun-Yi Hu, Takashi Maekawa, Nicholas M. Patrikalakis, Xiuzi Ye (1997). Robust interval algorithm for surface intersections. *Computer Aided Design*, 29(9):617–627.

Ernst Huber, Wilhelm Barth (1999). Surface-to-Surface intersection with complete and guaranteed results. In **T. Csendes** (Hrsg.) (1999), *Developments in Reliable Computing*, S. 185–198. Kluwer Academic Publishers, 1999.

Ernst Huber (1998). Intersecting General Parametric Surfaces Using Bounding Volumes. In *Tenth Canadian Conference on Computational Geometry - CCCG '98, Vancouver, Canada, August 15 – 18*, S. 52–54, 1998.

Ernst Huber (1999). *Ein Einschließungsalgorithmus zur sicheren Berechnung der vollständigen Berechnung der Schnittkurve zweier Flächen.* Dissertation, TU Wien, 1999.

Roland van Iwaarden, Jorge Stolfi (1997). Affine Arithmetic Sofware. 1997.

John K. Johnston (1993). A new intersection algorithm for cyclides and swept surfaces using circle decomposition. *Computer Aided Geometric Design*, 10:1–24.

John Keyser, Shankar Krishnan, Dinesh Manocha (1996). Efficient B-rep Generation of Low-Degree Sculptured Solids Using Exact Arithmetic. Technischer Bericht TR96-040, Department of Computer Science, University of North Carolina - Chapel Hill, Oktober 22 1996.

Wolfgang Knüppel (1993a). BIAS - Basic Interval Arithmetic Subroutines. Technischer Bericht 93.3, Technische Universität Hamburg Harburg, Technische Informatik II, 1993.

Wolfgang Knüppel (1993b). PROFIL - Programmer' s Runtime Optimized Fast Interval Library. Technischer Bericht 93.4, Technische Universität Hamburg Harburg, Technische Informatik II, 1993.

Wolfgang Knüppel (1993c). PROFIL/BIAS - Extensions. Technischer Bericht 93.5, Technische Universität Hamburg Harburg, Technische Informatik II, 1993.

Pramod Koparkar (1991). Surface intersection by switching from recursive subdivision to iterative refinement. *The Visual Computer*, 8:47–63.

George A. Kriezis, Nicholas M. Patrikalakis, Franz-Erich Wolter (1992). Topological and differential equation methods for surface intersections. *Computer-Aided Design, 1992*, 24:41–55.

S. Krishnan, M. Gopi, D. Manocha, M. Mine (1997). Interactive Boundary Computation of Boolean Combinations of Sculptured Solids. *Computer Graphics Forum*, 16(3):67–78.

Shankar Krishnan, Dinesh Manocha (1997). An Efficient Surface Intersection Algorithm Based on Lower-Dimensional Formulation. *ACM Transactions on Graphics*, 16(1):74–106.

Ming C. Lin, John F. Canny (1991). A Fast Algorithm for Incremental Distance Calculation. In *IEEE International Conference on Robotics and Automation*, S. 1008–1014, April 1991.

Dinesh Manocha, John F. Canny (1991). A New Approach to Surface Intersection. *Int. J. Computational Geometry and Applications*, 1(4):491–516.

Dinesh Manocha, Shankar Krishnan (1997). Algebraic pruning: a fast technique for curve and surface intersection. *Computer Aided Geometric Design*, 14(9):823–845.

Robert Markot, Robert Magedson (1991). Procedural method for evaluating the intersection curves of two parametric surfaces. *Computer Aided Design*, 23(6):395–404.

Gregor Müllenheim (1990). Convergence of a surface/surface intersection algorithm. *Computer Aided Geometric Design*, 7(5):415–423.

Gregor Müllenheim (1991). On Determining Start Points for a Surface/Surface Intersection Algorithm. *Computer Aided Geometric Design*, 8(5):401–408.

Yves de Montaudouin, Wayne Tiller, Havard Vold (1986). Applications of power series in computational geometry. *Computer-Aided Design*, 18(10):514–524.

Arnold Neumeier (1990). *Interval Methods for Systems of Equations*, Band 37 von *Encyclopedia of Mathematics and its Applications*. Cambridge University Press.

Marco Paluszny, Katja Bühler (1998). Canal Surfaces and Inversive Geometry. In *Mathematical Methodes for Curves and Surfaces II*, S. 367–374. L.L. Schumaker, 1998.

Nicholas M. Patrikalakis (1992). Interrogation of surface intersections. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 161–185. SIAM, Philadelphia, 1992.

Nicholas M. Patrikalakis (1993). Surface-to-Surface Intersections. *IEEE Computer Graphics and Applications*, 13(1):89–95.

Les Piegl (1992). Constructive geometry approach to surface-surface intersection. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 137–159. SIAM, Philadelphia, 1992.

Franco P. Preparata, Michael Ian Shamos (1985). *Computational Geometry: An Introduction.* Texts and Monographs in Computer Science. Springer-Verlag, Berlin, Germany.

Peter Schramm (1995). *Sichere Verschneidung von Kurven und Flächen im CAGD.* Dissertation, Universität Karlsruhe, Fakultät für Mathematik, 1995.

T.W. Sederberg, H. Christiansen, S. Katz (1989). Improved test for closed loops in surface intersections. *Computer Aided Design*, 21(8):505–508.

Thomas W. Sederberg, Rida T. Farouki (1992). Approximation by interval Bezier curves. *IEEE Computer Graphics and Applications*, 12(5):87–95.

Thomas W. Sederberg, Ray J. Meyers (1988). Loop detection in surface patch intersections. *Computer Aided Geometric Design*, 5(2):161–171.

Thomas W. Sederberg, Tomoyuki Nishita (1991). Geometric Hermite Approximation of Surface Patch Intersection Curves. *Computer Aided Geometric Design*, 8(2):97–114.

P. Sinah, E. Klassen, K.K. Wang (1985). Exploiting topological and geometric properties for selective subdivision. In *Proc. ACM Symposium on Computational Geometry*, S. 39–45, 1985.

John M. Snyder, Adam R. Woodbury, Kurt Fleischer, Bena Currin, Alan H. Barr (1993). Interval methods for multi-point collisions between time-dependent curved surfaces. *Computer Graphics*, 27(Annual Conference Series):321–334.

John M. Snyder (1992). Interval Analysis For Computer Graphics. *Computer Graphics*, 26(2):121–130.

Jorge Stolfi, Luiz Enrique de Figueiredo (1997). *Self-Validated Numerical Methods and Applications.* unpublished.

Jorge Stolfi (1993). Affine Arithmetic Sofware. http://www.dcc.unicamp.br/ stolfi/EXPORT/ software/c/Index.html♯libaa, 1993.

Nilo Stolte, Arie Kaufman (1998). Parallel Spatial Enumeration of Implicit Surfaces using Interval Arithmetic for Octree Generation and and its Direct Visualization. In *Proceedings of Implicit Surfaces'98, Seattle, June 15–16*, S. 81–88, 1998.

Tz.E. Stoyanov (1992). Marching Along Surface/Surface Intersection Curves with an Adaptive Step Length. *Computer Aided Geometric Design*, 9(6):485–490.

Irina Voiculescu, Jakob Berchtold, Adrian Bowyer, Ralph R. Martin, Qijiang Zhang (2000). Interval and Affine Arithmetic for Surface Location of Power- and Bernstein-Form Polynomials. In **Roberto Cipolla, Ralph Martin** (Hrsg.) (2000), *The Mathematics of Surfaces IX*, S. 410–423, London, Berlin, Heidelberg, 2000. Springer.

K.Y. Wang (1992). Parametric surface intersections. In **R. E. Barnhill** (Hrsg.) (1992), *Geometry Processing for Design and Manufacturing*, S. 187–204. SIAM, Philadelphia, 1992.

Shin-Ting Wu, Lenimar N. Andrade (1999). Marching along a regular surface/surface intersection with circular steps. *Computer Aided Geometric Design*, 16:249–268.

Atsushi Yamada, Fujio Yamaguchi (1996). Homogeneous bounding boxes as tools for intersection algorithms of rational beézier curves and surfaces. *The Visual Computer*, 12:202–214.

Curriculum Vitae

Name:	Dipl.-Math. Katja Bühler
Geburtstag:	22.10.1969 in Marktredwitz, Deutschland.
Nationalität:	Deutsch
Adresse:	Laudongasse 5/17, A-1080 Wien, Österreich
EMail:	katja@cg.tuwien.ac.at

Ausbildung:

April 1989	Abitur, Gymnasium Neureut, Karlsruhe, Deutschland
Oktober 1989 - August 1996	Studium der reinen Mathematik mit Nebenfach Informatik an der Universität Fidericiana zu Karlsruhe. Spezialisierung in Geometrie, Numerik und Computergraphik.
1994-1996	Studium "Angewandte Kulturwissenschaften" als Begleitstudium an der Universität Karlsruhe.
August 1996	Diplom in Mathematik.
Seit April 1998	Doktoratsstudium der Technischen Wissenschaften an der Technischen Universität Wien, Österreich.
Seit Oktober 1999	Studentin des "Interdisziplinären Lehrgangs für Höhere Lateinamerika-Studien" am österreichischen Lateinamerika Institut, Wien.

Beruf:

Juli 1990 - Mai 1992	Wissenschaftliche Hilfskraft am Institut für Werkzeugmaschinen und Betriebstechnik, Universität Karlsruhe.
Mai 1995 - Januar 1997	Wissenschaftliche Hilfskraft am Institut für Angewandte Mathematik (Prof. Kulisch), Universität Karlsruhe.
Dezember 1997	Verleihung eines individuellen Forschungsstipendiums für Graduierte durch den Deutschen Akademischen Austauschdienst (DAAD) für einen einjährigen Forschungsaufenthalt in Venezuela.
März 1997 - Februar 1998	Forschungsaufenthalt im Centro de Computación Gráfica y Geometría Applicada, Universidad Central de Venezuela, Caracas, Venezuela.
Seit April 1998	Universitätsassistentin am Institut für Computergraphik und Algorithmen, Technische Universität Wien.

Publikationen:

Katja Bühler (1995). Rationale algebraische Kurven auf Dupinschen Zykliden. Diplomarbeit, Universität Karlsruhe, 1995

Marco Paluszny, Katja Bühler (1998). Canal Surfaces and Inversive Geometry. In *Mathematical Methodes for Curves and Surfaces II*, S. 367–374. L.L. Schumaker, 1998

Katja Bühler, Wilhelm Barth (2000). Intersecting Parametric Surfaces Using Special Arithmetics. In *Extended Abstract, Proceedings of the Alhambra 2000: A Joint Mathematical European-Arabis Conference, Granada, Spain, July 3–7*, 2000

Katja Bühler, Wilhelm Barth (2001). A new intersection algorithm for parametric surfaces based on linear interval estimations. In *Scientific Computing, Validated Numerics, Interval Methods*. Kluwer Academic Publishers, 2001

Katja Bühler (2001). Über den Schnitt zweier Linearer Intervallabschätzungen. Technischer Bericht TR-186-2-01-07, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Karlsplatz 13/186/2, A-1040 Vienna, Austria, 2001

Katja Bühler (2001c). Taylor Models and Affine Arithmetics - Towards a More Sophisticated Use of Reliable Methods in Computer Graphics. In *Proceedings of the Seventeenth Spring Conference on Computer Graphics (SCCG), Budmerice, Slovakia, April 25–28*, Bratislava, Slovakia, April 2001. Comenius University Press

Katja Bühler (2001b). Linear Interval Estimations for Parametric Objects - Theory and Applications. *Computer Graphics Forum*, 20(3)

Katja Bühler (2001a). Linear Interval Estimations for Implicit Curves and Surfaces. Extended Abstract. Accepted at the Workshop *Uncertainty in Geometric Computations*, The University of Sheffield 5th - 6th July, 2001