

# Implizites Modellieren und Co-Visualisierung mittels Tetraedergittern

Tobias Frank

tobias.frank@nmbgmbh.de

**Abstract:** Tetraedergitter gewinnen immer mehr an Bedeutung für technische und naturwissenschaftliche Anwendungen. Die vorliegende Arbeit stellt neue Algorithmen für das Erstellen von komplexen impliziten Modellen – basierend auf Tetraedergittern – und deren effiziente Visualisierung vor. Zum *Modellieren* wurde ein neuartiger Algorithmus zur Oberflächenrekonstruktion von komplexen 3D-Flächen aus schlecht konditionierten Daten, wie sie oft in naturwissenschaftlichen Anwendungen auftreten, entwickelt. Dabei kann die Fläche begrenzt sein und Unstetigkeiten aufweisen. Des Weiteren wurde im Rahmen dieser Arbeit eine innovative Methode zur intuitiven Echtzeit-Manipulation von implizit definierten Objekten erarbeitet. Die Manipulation erfolgt interaktiv direkt am 3D-Volumenmodell. Die *Visualisierung* besteht aus einer generischen Architektur zur Informationsgewinnung und Interpretation mittels Co-Visualisierung diverser Attribute und Booleschen Operatoren der Constructive Solid Geometry mit konstanter Komplexität.

## 1 Einleitung

In den letzten Jahren hat die Bedeutung der Geoinformatik stetig zugenommen. Nicht nur im Bereich Raumplanung, Logistik oder Location Based Services sondern auch als Instrument zur optimalen Förderung von Bodenschätzen mittels hochauflösender 3D-Darstellung von Lagerstätten. Im Bereich der 3D-Geomodellierung werden überwiegend triangulierten Flächen, Kartesische Gitter bzw. regelmäßige Hexaedergitter benutzt. Diese Methoden haben Beschränkungen in der volumetrischen Darstellung oder führen zu enorm großen Modellen, die wenn überhaupt mit Clustern bearbeitet werden können. Die in dieser Arbeit entwickelte Lösung basiert auf unstrukturierten Tetraedergittern zusammen mit impliziten Funktionen. Damit wurde eine Erhöhung der geometrischen Auflösung bei gleichzeitiger Reduzierung der Rechenzeiten erreicht. Die vorliegende Arbeit umfasst eine komplette Prozesskette zum Erstellen, Manipulieren und Interpretieren von komplexen geologischen Modellen. Ausgehend von schlecht konditionierten Punktwolken werden Grenzflächen rekonstruiert, die zur Modellbeschreibung dienen. Diese können interaktiv manipuliert werden, um somit Ambiguitäten aufzulösen und Expertenwissen des Benutzers einfließen zu lassen. Schließlich werden die Modelle für eine fundierte Interpretation mit verschiedenen Attributen visualisiert. Die entwickelten Technologien sind keineswegs auf geologische Anwendungen beschränkt, sondern können problemlos in klassischen Anwendungen im CAD (Computer Aided Design) und CAE (Computer Aided Engineering) Umfeld angewandt werden

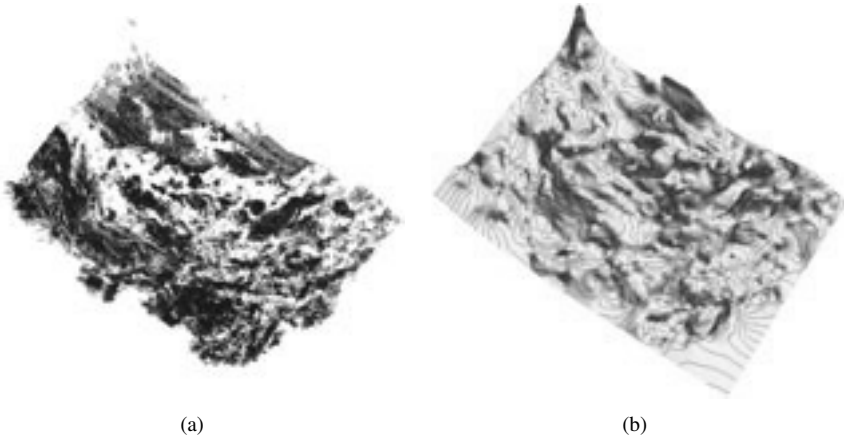


Abbildung 1: (a) 377384 Punkte, die aus einem seismischen Signal extrahiert wurden. Diese Punktwolke zeigt mehrere große Lücken, und steile Flanken. Zudem ist der Datensatz stark verrauscht. (b) Rekonstruktion von (a) mit 85855 Dreiecken. Die implizite Funktion wurde auf einen Tetraedergitter mit 95073 Elementen (15625 Knoten) interpoliert. Dieses Gitter entstand aus einem initialen Gitter mit 4841 Elementen nach dreizehn Iterationen lokaler Delaunay Verfeinerung. Die Gesamtberechnungsdauer der Rekonstruktion betrug etwa eine halbe Stunde auf normaler PC Hardware. Daten mit freundlicher Genehmigung von Shell.

## 2 Oberflächenrekonstruktion aus schlecht konditionierten Punktmengen

Zum Modellieren geologischer Objekte wurde ein neuartiger Algorithmus zur Oberflächenrekonstruktion entwickelt. Oberflächenrekonstruktion ist ein sehr intensiv untersuchtes Problem der Computational Geometry, im Bereich CAD und der Geomodellierung. Jedoch beschränken sich die meisten Arbeiten auf Modelle, die mittels 3D-Laserscanner unter Laborbedingungen gewonnen werden. Die in dieser Arbeit verwendeten Punktmengen sind jedoch sehr schlecht konditioniert und zeichnen sich dadurch aus, dass sie mit sehr großem Rauschen belastet sind. Es treten Regionen mit geringen oder keinen Daten auf und die zu rekonstruierenden Flächen sind topologisch komplex, indem sie Diskontinuitäten aufweisen (Abb. 2) oder begrenzt sind. Die Eingangsdaten bestehen aus  $n$  Punkten  $p_i \in \mathbb{R}^3$  mit  $i \in 1, \dots, n$ . Diese Punkte sind unorganisierte Tripel aus  $x, y, z$ -Werten ohne weitere Information bezüglich ihrer Topologie (Abb. 3(a)). Die Herausforderung der Oberflächenrekonstruktion besteht darin, eine Fläche  $S$  zu generieren, welche die Punktwolke interpoliert.

### 2.1 Algorithmus

Der neuentwickelte Algorithmus generiert eine stückweise lineare Rekonstruktion der Punktmenge. Dazu wird die Domäne der Punktmenge in ein unstrukturiertes Tetraeder-

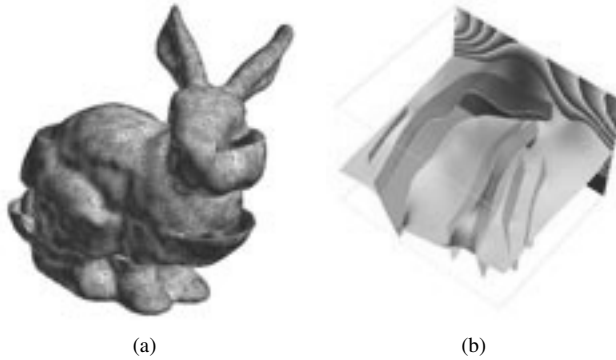


Abbildung 2: (a) Stanford Bunny mit künstlicher Diskontinuität, rekonstruiert aus 8171 Punkten. (b) geologische Grenzfläche. Die Diskontinuität entsteht durch geologische Störungen. Der Querschnitt zeigt die implizite Funktion aus welcher der Horizont rekonstruiert wurde. Daten mit freundlicher Genehmigung von Earth Decision.

gitter eingebettet. Dieses Tetraedergitter zeichnet sich durch eine adaptive Größe seiner Elemente aus. In Regionen der Eingangspunkte finden sich kleine Tetraeder und in Regionen abseits der Eingangspunkte große Tetraeder. Im Vergleich zu regelmäßigen Kartesischen Gittern wird somit eine hohe numerische Auflösung in den entscheidenden Regionen erreicht, ohne die Elementzahl übermäßig ansteigen zu lassen. Reicht die numerische Auflösung des initialen Gitters nicht aus, so kann dieses iterativ, lokal verfeinert werden. Auf diesem Gitter wird eine implizite Funktion  $\varphi(x, y, z)$  interpoliert (Abb. 3(b)). Dabei sind die Eingangsdaten Boundary Constraints mit dem Wert  $\varphi(x, y, z) = 0$ . Zusätzlich werden Inequality Constraints mit  $\varphi(x, y, z) > 0$  auf der einen Seite von  $S$  und  $\varphi(x, y, z) < 0$  auf der anderen Seite von  $S$  definiert. Verglichen mit Constraints von [TO02], die einen konstanten Wert mit invertierten Vorzeichen für das Innere und das Äußere einer geschlossenen Fläche definieren, haben die verwendeten Inequality Constraints einen weitaus geringeren Einfluss auf die Größe des Gradienten der impliziten Funktion. Daraus folgt, dass die Position der Constraints einen weitaus geringeren Einfluss auf das Interpolationsergebnis hat und nur sehr wenige dieser Constraints benötigt werden. Abschließend wird ein Roughness Constraint definiert, der den Gradienten von  $\varphi(x, y, z)$  und somit direkt die Rauigkeit der rekonstruierten Fläche kontrolliert. Diskontinuitäten in der zu rekonstruierenden Fläche werden durch Diskontinuitäten im Graphen des Tetraedergitters realisiert, wodurch eine Unstetigkeit in  $\varphi(x, y, z)$  entsteht. Die Interpolation von  $\varphi(x, y, z)$  erfolgt mittels Discrete Smooth Interpolation (DSI) (siehe A). Die herausragende Eigenschaft dieser Methode ist, dass die Komplexität nicht primär von der Anzahl der Punkte in den Eingangsdaten, sondern von der Anzahl der Knoten der Tetraedergitter abhängt. Für die betrachteten Fälle lag die Anzahl der Knoten im Tetraedergitter immer weit unter der Anzahl der Punkte der Eingangsdaten. Im Vergleich dazu leiden die Methoden, die auf den Radialen Basis Funktionen (RBF) [SPOK95, TO99] basieren, an einer höheren Komplexität im Bezug auf die Größe der Punktmenge, die rekonstruiert werden soll. Im gesamten Algorithmus müssen zu keinem Zeitpunkt Normalenvektoren auf der Punktmenge bestimmt werden. Eine Berechnung der Normalenvektoren ist vor allem bei

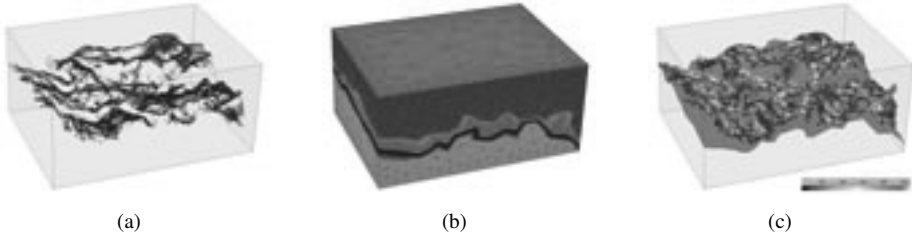


Abbildung 3: (a) 377384 Punkte extrahiert aus einem seismischen Signal. (b) initiales Tetraedergitter mit 4841 Elementen. (c) rekonstruierte Fläche aus (a) mittels (b). Eingangsdaten sind mit einem Fehlerkriterium gezeichnet, welches den Interpolationsfehler definiert. Daten mit freundlicher Genehmigung von Shell.

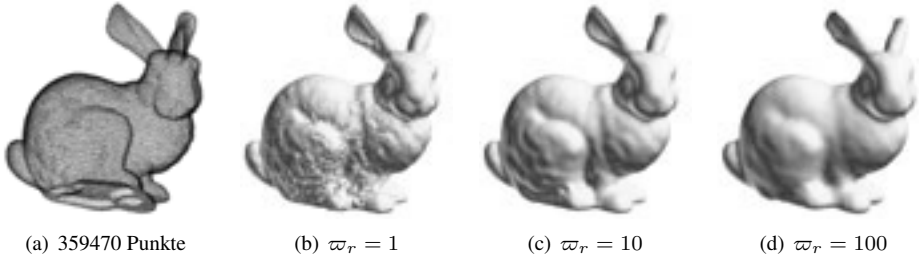


Abbildung 4: (a) Punktemenge mit Gausschen Rauschen. (b)-(d) Rekonstruktionen von (a) mit unterschiedlichen relativen Gewicht der Roughness Constraints ( $\varpi_r$ ).

Rauschen, bei geringer Dichte oder in Regionen mit starker Krümmung der Eingangsdaten sehr fehleranfällig oder unmöglich. Stellt sich heraus, dass die Rekonstruktion von den Eingangsdaten abweicht (Abb. 3(c)), kann die numerische Auflösung des Tetraedergitters durch lokale Verfeinerung erhöht werden und eine anschließende Reinterpolation führt zu einer exakteren Lösung. Dieses iterative Vorgehen bietet sich vor allem für komplexe Strukturen an. Die Triangulation der Isofläche erfolgt mit dem sog. Marching Tetrahedrons Algorithmus (Spezialisierung des Marching Cubes [LC87] Algorithmus), und anschließender Optimierung der Dreiecke der diskretisierten Fläche.

## 2.2 Rauschen

Eine große Schwierigkeit bei der Rekonstruktion von geologischen Flächen ist das starke Rauschen der Eingangsdaten. Dabei ist es nicht wünschenswert, dass alle Eingangsdaten respektiert werden, was zu einer rauen Fläche führen würde. DSI behandelt Boundary und Roughness Constraints im Sinne der kleinsten Quadrate. Werden diese beiden Klassen von Constraints gewichtet (Gleichung 9), so kann mittels DSI die Rauigkeit der rekonstruierten Fläche kontrolliert werden. Je höher das Gewicht des Roughness Constraints im Vergleich zu den Boundary Constraints ist, desto glatter wird die rekonstruierte Fläche.

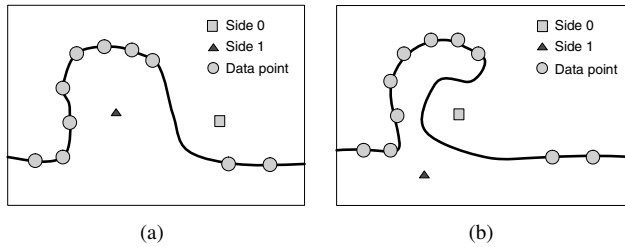


Abbildung 5: (a) Lücken in den Eingangsdaten werden möglichst glatt interpoliert. (b) Das manuelle Verschieben von Inequality oder Boundary Constraints führt zu einer Verformung der Rekonstruktion.

Das relative Gewicht der Roughness Constraints bestimmt also die Steifigkeit der Fläche. Abb. 4(a) zeigt eine Punktwolke des Stanford Bunnys bestehend aus 359470 Punkten mit Gausschen Rauschen. Die Rekonstruktion wurde mit verschiedenen relativen Gewicht der Roughness Constraints durchgeführt. Bei einem geringen relativen Gewicht der Roughness Constraints erhält man eine Fläche die möglichst allen Eingangspunkten folgt. Dies führt bei starken Rauschen zu Artefakten (Abb. 4(b)). Wählt man das relative Gewicht der Roughness Constraints zu hoch so verliert die Rekonstruktion an Details (Abb. 4(d)). Ein ideales relatives Gewicht der Roughness Constraints erhält Details ohne unerwünschte Artefakte zu erzeugen (Abb. 4(c)).

### 3 Manipulieren von impliziten Modellen

Die schlechte Konditionierung von wissenschaftlichen Daten führt oft zu topologischen Ambiguitäten, welche nur manuell behoben werden können. Ebenso ist es notwendig bei lückenhaften Daten das Expertenwissen eines Anwenders einfließen zu lassen, indem er das implizite Modell interaktiv manipuliert. Die entwickelte Methode beruht darauf, Kontrollpunkte, die das implizite Modell beeinflussen, zu verschieben und die implizite Funktion  $\varphi(x, y, z)$  zu reinterpolieren. Dieses Verfahren ähnelt dem von [TO02], basiert jedoch auf einer völlig anderen mathematischen Methode, nämlich DSI, das auch zur Oberflächenrekonstruktion benutzt wurde. Abb. 5 zeigt schematisch die Kontur einer Rekonstruktion. Die Eingangsdaten haben eine Lücke, welche durch den vorgestellten Algorithmus in der Weise gefüllt wird, so dass die globale Rauigkeit minimiert wird (Abb. 5(a)). Dies muss jedoch nicht mit der Realität übereinstimmen. Der Anwender kann nun sein Wissen oder Informationen aus anderen Datenquellen in der Form einbringen, dass er das Modell verformt, indem er bestehende Inequality oder Boundary Constraints verschiebt (Abb. 5(b)). Im Folgenden werden die Erweiterungen beschrieben, mit denen die Manipulationen interaktiv in Echtzeit durchgeführt werden können. Das implizite Ausgangsmodell, das durch die Funktion  $\varphi$  beschrieben wird, kann durch Rekonstruktion oder andere Methoden interpoliert worden sein. Dabei ist anzumerken, dass die Menge der Constraints, die ein Modell definieren, weitaus größer ist als die Menge der Constraints, die dazu benutzt werden das Modell zu verformen. Die Grundlage von Real-Time MxDSI

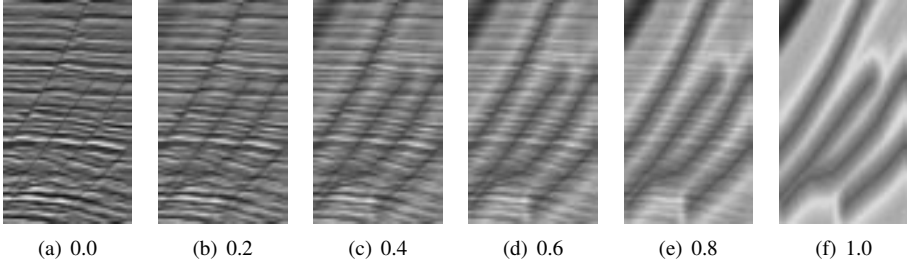


Abbildung 6: Kombination von zwei Attributen: seismisches Signal (links) und implizite Distanzfunktion (rechts) mittels benutzerdefinierter Fragmentshadern. Zahlenwerte beschreiben das Gewicht der Distanzfunktion für die Blendingfunktion.

(RtMxDSI) ist eine Dekomposition der Menge der Constraints  $\mathcal{C}$  in zwei Teilmengen. Die Base-Constraints  $\mathcal{C}_B$  beschreiben die statischen Eingangsdaten und die Float-Constraints  $\mathcal{C}_F$  stellen dynamische Constraints zur Verformung des Modells dar. Es gilt:  $\mathcal{C} = \mathcal{C}_B \cup \mathcal{C}_F$ , somit können wir für die Dekomposition der Matrix  $\mathbf{A}$  und dem Vektor  $\mathbf{b}$  schreiben:

$$\mathbf{A} = \mathbf{A}_B + \mathbf{A}_F \quad (1)$$

$$\mathbf{b} = \mathbf{b}_B + \mathbf{b}_F \quad (2)$$

Zusammen mit Gleichung 11 ergibt sich:

$$\left| \begin{array}{l} \mathbf{A}_B = \sum_{c \in \mathcal{C}_B^s} A_c \cdot A_c^t \\ \mathbf{A}_F = \sum_{c \in \mathcal{C}_F^s} A_c \cdot A_c^t \end{array} \right. \quad \text{und} \quad \left| \begin{array}{l} \mathbf{b}_B = \sum_{c \in \mathcal{C}_B^s} b_c \cdot A_c \\ \mathbf{b}_F = \sum_{c \in \mathcal{C}_F^s} b_c \cdot A_c \end{array} \right. \quad (3)$$

Damit erhält man für das Minimierungsproblem aus Gleichung 12:

$$LS(\varphi) = \frac{1}{2} \cdot \varphi^t \cdot (\mathbf{A}_B + \mathbf{A}_F) \cdot \varphi - (\mathbf{b}_B + \mathbf{b}_F)^t \cdot \varphi \quad (4)$$

Die Implementierung dieser Erweiterung beruht auf zwei linearen Gleichungssystemen, dem Base-System und dem Float-System. Die Lösung des Base-Systems ist das initiale implizite Modell. Ändert sich der Zustand des Float-Systems dadurch, dass ein Float-Constraint hinzugefügt, entfernt oder verschoben wurde, so werden beide Systeme zu einem gemeinsamen linearen Gleichungssystem nach Gleichung 4 vereint. Änderungen der Float-Constraints betreffen nur das Float-System. Das Vereinen der beiden Systeme und die Reinterpolation von  $\varphi(x, y, z)$  – mittels Konjugierter Gradienten Methode – erfolgt sehr performant, da die neue Lösung  $\varphi^*$  sehr nah bei der initialen Lösung  $\varphi$  liegt.

## 4 Co-Visualisierung

In den letzten Abschnitten wurde sowohl das Erstellen von impliziten Modellen aus schlecht konditionierten Punktdaten, als auch das Manipulieren dieser Modelle dargestellt.

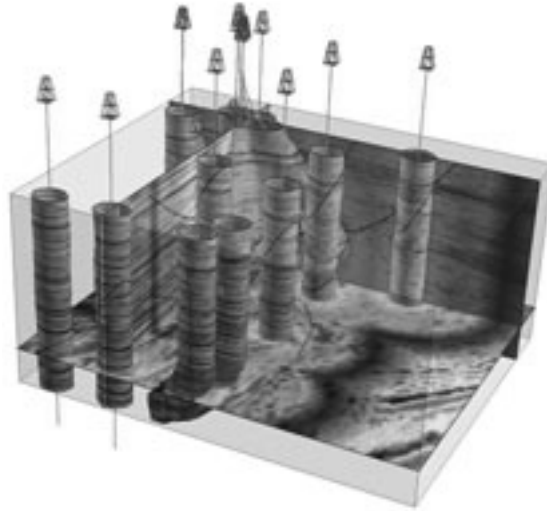


Abbildung 7: Tetraedergitter, das mittels Querschnitten und Isoflächen einer impliziten Distanzfunktion visualisiert wurde. Diverse geologische Attribute wurden co-visualisiert. Daten mit freundlicher Genehmigung von Earth Decision.

Als letzter Schritt in einer Prozesskette zum Erstellen und Charakterisieren geologischer Modelle mittels Tetraedergittern, sind Werkzeuge zur effektiven Interpretation notwendig. Grundlagen der Co-Visualisierung sind Multitexturing, prozedurale Texturen und die Kombination mehrerer impliziter Modelle mittels Operatoren der Constructiv Solid Geometry (CSG) und die Kombination aus diesen. Dabei spielte die Programmierbarkeit moderner Graphikprozessoren (GPU) – vor allem der Fragmentshader und Textureinheiten – eine entscheidende Rolle. Abb. 6 zeigt eine Serie von Ausschnitten eines geologischen Modells, in der zwei Attribute (physikalisches Signal und implizite Distanzfunktion), mit Hilfe einer als Fragmentshader implementierten Blendingfunktion, kontinuierlich kombiniert sind. Abb. 7 zeigt ein Tetraedermodell, das mit Querschnitten und Iso-Flächen einer impliziten Distanzfunktion visualisiert wurde. Die Flächen sind mit einer impliziten Distanzfunktion, einem physikalischen Signal und den Konturen von Störungsflächen co-visualisiert. Nicht nur die Visualisierung verschiedener Attribute ist für eine Interpretation notwendig, sondern auch die Implementierung von bedingten Abfragen. Diese Abfragen können mittels Booleschen CSG Operatoren formuliert werden (Abb. 8). Die implizite Modellbeschreibung eignet sich hervorragend für diese Art der Berechnungen. Im Rahmen dieser Arbeit wurden Boolesche CSG Operatoren als prozedurale Texturen realisiert, die auf Isoflächen der impliziten Modelle angewandt werden. Die Berechnung Boolescher CSG Operationen findet dabei komplett auf der GPU statt und die Komplexität dieser Methode hängt nur von der Bildschirmauflösung ab und ist somit konstant im Bezug auf die Modellgröße.

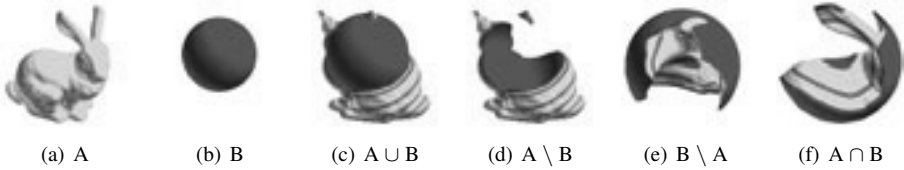


Abbildung 8: Boolesche CSG Operatoren auf zwei implizite Modelle (A und B) angewandt.

## 5 Zusammenfassung

In der vorliegenden Arbeit wurden Methoden und Algorithmen für das Erstellen und Manipulieren von impliziten Modellen auf Tetraedergittern sowie Co-Visualisierungstechniken zur effektiven Interpretation dieser Modelle entwickelt. Damit wird eine Prozesskette in der Geomodellierung von der Modellerzeugung bis hin zur Interpretation und Manipulation abgedeckt. Die Modellerzeugung beinhaltet einen neuartigen Algorithmus zur Oberflächenrekonstruktion – mittels impliziten Funktionen – aus schlecht konditionierten und unorganisierten Punktwolken. Diese impliziten Modelle können interaktiv manipuliert werden, wodurch der Benutzer intuitiv sein Expertenwissen und Informationen aus weiteren Quellen einfließen lassen kann. Schließlich werden diese auf Tetraedergittern definierten impliziten Modelle, mit Hilfe von programmierbaren Grafikprozessoren, in Echtzeit visualisiert. Ebenso wurden hocheffiziente bedingte Abfragen mittels CSG Operatoren mit konstanter Komplexität entwickelt und bilden die Grundlage für eine stichhaltige Interpretation der erstellten Modelle.

## Danksagung

Dieses Forschungsprojekt wurde im Rahmen des „gOcad Research Project“ durchgeführt. Den Firmen und akademischen Sponsoren der „gOcad Research Group“ gilt der Dank für die Finanzierung dieser Arbeit.

## A Matrixformulierung von DSI für Tetraedergitter

Dieser Anhang gibt eine kleine Einführung in die Discrete Smooth Interpolation (DSI) Methode [Mal02] in einer angepassten Matrixformulierung für die Interpolation einer impliziten Funktion auf den Knoten eines Tetraedergitters [MTMH05]. Der Einfachheit halber beschränkten sich die Ausführungen auf skalare Funktionen.  $\mathcal{M}(\Omega, N, \varphi, \mathcal{C})$  beschreibt ein diskretes Modell im 3D-Euklidischen Raum. Eine endliche Menge  $\Omega$  von  $n$  verbundenen Knoten definieren ein Tetraedergitter, das als Graph betrachtet werden kann. Dieser Graph definiert einen Nachbarschaftsoperator  $N$  auf  $\Omega$ . Die Nachbarschaft  $N(\alpha)$  eines beliebigen Knoten  $\alpha \in \Omega$  ist eine Untermenge von Knoten von  $\Omega$ , welche direkt



mit  $\alpha$  verbunden sind, einschließlich  $\alpha$  selbst. Eine Funktion  $\varphi$  auf  $\Omega$  ist folgendermaßen definiert:

$$\varphi = [ \varphi(1), \dots, \varphi(\alpha), \dots, \varphi(n) ]^t \quad (5)$$

Die letzte Komponente  $\mathcal{C}$  im zugrunde liegendem linearen diskreten Modell ist eine Menge an linearen Constraints, die von  $\varphi$  erfüllt werden müssen:

$$A_c^t \cdot \varphi \bowtie b_c \quad \forall c \in \mathcal{C} \quad (6)$$

$A_c$  ist eine Spaltenmatrix von der gleichen Größe wie  $\varphi$ ,  $b_c$  ein gegebener Koeffizient und  $\bowtie$  einer der Operatoren  $\{\simeq, =, \geq\}$ . Der Binäroperator  $\bowtie$  aus Gleichung 6 beschreibt die möglichen Constraintklassen: Soft-Constraints  $\mathcal{C}^s$  wie  $\simeq$  und Hard-Constraints  $\mathcal{C}^h$  ( $=, \geq$ ). Die DSI Methode ist ein Interpolator, der die Funktion  $\varphi$  in der Weise approximiert, dass die Hard-Constraints  $\mathcal{C}^h$  streng und die Soft-Constraints  $\mathcal{C}^s$  im Sinne der kleinsten Quadrate respektiert werden. Zusätzlich wird immer eine Untermenge an Soft-Constraints  $\mathcal{C}^{ss}$ , die ein Rauigkeitskriterium definieren, benötigt. Das Rauigkeitskriterium kontrolliert den Gradienten von  $\varphi$  von einem Knoten  $\alpha$  zu seinen Nachbarn  $\beta \in N(\alpha)$ . In dieser Ausführung werden die Hard-Constraints  $\mathcal{C}^h$  vernachlässigt, so dass die Menge der DSI-Constraints  $\mathcal{C}$  aus  $k$  Soft-Constraints  $\mathcal{C} = \mathcal{C}^s = \{c_1, \dots, c_k\}$  besteht, welche durch  $k$  lineare Gleichungen definiert sind.

$$\left| \begin{array}{l} A_{c_1}^t \cdot \varphi \simeq b_{c_1} \\ \vdots \\ A_{c_k}^t \cdot \varphi \simeq b_{c_k} \end{array} \right. \quad (7)$$

Der DSI Interpolator minimiert den durch die Verletzung der Soft-Constraints bedingten Interpolationsfehler im Sinne der kleinsten Quadrate. Dieses Minimierungsproblem (LS) kann wie folgt formuliert werden:

$$LS(\varphi) = \sum_{c \in \mathcal{C}^s} \|A_c^t \cdot \varphi - b_c\|^2 \quad (8)$$

Für die Priorisierung der einzelnen Soft-Constraints werden Gewichte eingeführt:

$$\forall c \in \mathcal{C}^s : \left| \begin{array}{l} A_c \leftarrow \frac{\varpi_c}{\|A_c\|} \cdot A_c \\ b_c \leftarrow \frac{\varpi_c}{\|A_c\|} \cdot b_c \end{array} \right. \quad (9)$$

Die Normalisierung führt dazu, dass jeder Constraint  $c \in \mathcal{C}^s$  ein relatives Gewicht  $\varpi_c^2$  bezüglich der übrigen Constraints  $\mathcal{C}^s$  besitzt. Weiterhin gilt:

$$\|A_c^t \cdot \varphi - b_c\|^2 = \varphi^t \cdot (A_c \cdot A_c^t) \cdot \varphi - 2 \cdot (b_c \cdot A_c^t) \cdot \varphi + b_c^2 \quad (10)$$

Definiert man die quadratische, symmetrische Matrix  $\mathbf{A}$  und den Vektor  $\mathbf{b}$ :

$$\left| \begin{array}{l} \mathbf{A} = \sum_{c \in \mathcal{C}^s} A_c \cdot A_c^t \\ \mathbf{b} = \sum_{c \in \mathcal{C}^s} b_c \cdot A_c \end{array} \right. \quad (11)$$

so ergibt sich für das Minimierungsproblem LS (8) aus Gleichung (10) und (11) :

$$LS(\varphi) = \frac{1}{2} \cdot \varphi^t \cdot \mathbf{A} \cdot \varphi - \mathbf{b}^t \cdot \varphi \quad (12)$$

Diese quadratische Form wird mittels Konjugierten Gradienten [HS52] minimiert.

## Literatur

- [HS52] M. R. Hestenes und E. Stiefel. Methods of Conjugate Gradients for Solving Linear Systems. *J. Res. Nat. Bur. Stand.*, 49:409–436, 1952.
- [LC87] W.E. Lorensen und H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, Seiten 163–169, New York, NY, USA, 1987. ACM Press.
- [Mal02] J.-L. Mallet. *Geomodeling*. Oxford University Press, New York, 2002.
- [MTMH05] P. Muron, A.-L. Tertois, J.-L. Mallet und J. Hovadik. An Efficient and Extensible Interpolation Framework Based on The Matrix Formulation of the Discrete Smooth Interpolation. In *25th Gocad Meeting Proceedings*, Nancy, France, 2005.
- [SPOK95] V. V. Savchenko, A. A. Pasko, O. G. Okunev und T. L. Kunii. Function Representation of Solids Reconstructed from Scattered Surface Points and Contours. *Computer Graphics Forum*, 14(4):181–188, 1995.
- [TO99] G. Turk und J. F. O’Brien. Shape transformation using variational implicit functions. In *ACM SIGGRAPH 1999, Computer Graphics Proceedings*, Seiten 335–342, New York, NY, USA, 1999. ACM Press.
- [TO02] G. Turk und J. F. O’Brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855–873, 2002.

**Tobias Frank** wurde 1977 in Weiden i. d. Opf. geboren. Nach dem Abitur im Jahre 1997 studierte er Umwelt-Engineering an der Technischen Universität Bergakademie Freiberg. Im Hauptstudium spezialisierte er sich auf Kommunikationstechnologien und Informationsdienste. Während des Studiums arbeitete er an diversen Projekten im Bereich telefonbasierter Spracherkennung und VoIP (unter anderem für die IBM Deutschland GmbH). 2003 beendete er als Dipl.-Ing. sein Studium mit der Diplomarbeit „Implementation of a 3D-XML application server“. Direkt im Anschluss promovierte er (cotutelle de these) am INPL (Institut National Polytechnique de Lorraine) in Nancy (Frankreich) in Zusammenarbeit mit dem Lehrstuhl „Geoscience Mathematics and Informatics“ der Technischen Universität Bergakademie Freiberg. 2006 schloß er seine Promotion zum Dr. rer. nat. mit der Arbeit „Advanced Visualization and Modeling of Tetrahedral Meshes“ mit summa cum laude ab. Seitdem ist er als Senior-Softwareingenieur und Projektleiter im Bereich Computer Aided Engineering bei der Neuen Materialien Bayreuth GmbH beschäftigt. Im November 2006 wurde er für seine interdisziplinäre Arbeit und praxisrelevanten Forschungsergebnisse mit dem SAX-IT Nikolaus-Joachim-Lehmann Preis ausgezeichnet.