GPU-basierte Smart Visibility Techniken für die Planung von Tumor-Operationen

Christian Tietjen, Christoph Kubisch, Stefan Hiller, Bernhard Preim

Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg
tietjen@isg.cs.uni-magdeburg.de

Kurzfassung. Bei der Planung von Tumoroperationen ist die Einschätzung von Abständen und Infiltrationen zu vitalen Strukturen wichtig. Im Bereich der medizinischen Visualisierung wurden hierfür bereits zahlreiche Techniken entwickelt, die unter dem Begriff *Smart Visibility* zusammengefasst werden. Zu diesen zählen *Ghost Views* und *Section Views*. In diesem Beitrag wird eine GPU-basierte Realisierung dieser Techniken für polygonale Daten vorgestellt. Die Parametrisierung der Techniken erfolgt automatisch, um einen klinischen Einsatz ermöglichen zu können.

1 Einleitung

Um die Lage von Tumoren in ihrer Umgebung beurteilen zu können, müssen vor allem Abstände und Infiltrationen in gesundes Gewebe korrekt eingeschätzt werden. Die Darstellung der direkten Umgebung des Tumors ist dabei entscheidend. Optimalerweise müssten alle Strukturen völlig opak gezeichnet werden, um Form und räumliche Lagebeziehungen einschätzen zu können. Problematisch sind dabei die Verdeckungen, die in 3D-Visualisierungen zwangsweise auftreten. Die transparente Darstellung umgebender Strukturen ist oft nicht hilfreich, da je nach Transparenzgrad der Tumor oder die umliegende Struktur nicht ausreichend sichtbar ist. Daher wurden Visualisierungstechniken entwickelt, die die Sichtbarkeit garantieren, ohne auf Transparenz angewiesen zu sein. Diese Techniken werden unter dem Begriff *Smart Visibility* zusammengefasst [1]. Im folgenden Beitrag wird eine GPU-basierte Realisierung zweier Techniken aus diesem Bereich für polygonale Daten vorgestellt, mithilfe derer die Sichtbarkeit von verdeckten Strukturen sichergestellt werden kann. Außerdem wird eine automatische Parametrisierung dieser Verfahren vorgestellt, sodass diese Techniken ohne Mehraufwand für den Nutzer verwendet werden können.

Von Zhou et al. wurde vorgeschlagen, einen kugelförmigen Bereich frei in einem direkten Volumen-Rendering (DVR) zu bewegen, um einen beliebigen Bereich hervorzuheben [2]. Die Sichtbarkeit wird hierbei nicht sichergestellt. Bei dem so genannten *Ghost View* wird die Sichtbarkeit eines ausgewählten Tumors garantiert, in dem Strukturen ausgeblendet werden, die direkt vor dem Tumor liegen. Alternativ kann die Sichtbarkeit einer segmentierten Struktur in einem DVR auch mithilfe einer *Maximum Importance Projection* gewährleistet werden [3]. Dabei werden verdeckende Strukturen kegelförmig (konisch) weggeschnitten,

um die Tiefe des Schnittes und die umliegenden Strukturen sichtbar zu machen. Dies wird im weiteren Verlauf der Arbeit als *Section View* bezeichnet.

Für polygonale Oberflächennetze wurden ebenfalls *Smart Visibility*-Techniken entwickelt [4,5]. Das Problem bei der Generierung von *Section Views* in polygonalen Szenen ist die Tatsache, dass die Dreiecksnetze nur die Oberfläche einer Struktur repräsentieren. Wenn diese Strukturen nicht parallel zur Sichtrichtung aufgeschnitten werden, ist keine Geometrie im Inneren zu sehen – die Strukturen sind „leer“. Die bisher untersuchten Systeme haben dieses Problem unter Verwendung von *Constructive Solid Geometry* (CSG) behoben [6]. Im vorliegenden Beitrag wird ein Verfahren vorgestellt, das für diesen speziellen Anwendungsfall optimiert wurde und so wesentlich bessere Laufzeiten aufweist und leichter zu implementieren ist.

2 Material und Methoden

Zunächst wird der *Ghost View* vorgestellt. Die Umsetzung der Technik auf der GPU ist dabei trivial, der Fokus in diesem Abschnitt liegt auf der automatischen Parametrisierung des Schnittvolumens. Die Berechnung des Schnittvolumens ist auch für den im zweiten Teil vorgestellten *Section View* von Bedeutung.

Voraussetzung für die Verfahren ist eine Segmentierung aller relevanten Strukturen zur Extraktion der polygonalen Oberflächen.

2.1 Ghost View

Beim *Ghost View* wird die Sichtbarkeit eines Tumors gewährleistet, indem Strukturen ausgeblendet werden, die zwischen dem Tumor und dem Betrachter liegen. Dabei wird ein Zylinder als Schnittvolumen verwendet, dessen Achse immer zum Betrachter ausgerichtet ist. Um die Schnittkante hervorzuheben, wird sie mit einer farblichen Umrandung versehen. Zusätzlich hebt sich der Schnitt durch seine gleichmäßige Form hervor, da die dargestellten anatomischen Modelle in der Regel eine unregelmäßige und unebene Form besitzen. Die Silhouette der ausgeblendeten Strukturen kann weiterhin angezeigt werden, um die Begrenzung von geschnittenen Strukturen zu betonen.

Zunächst werden der Mittelpunkt und der Radius des Tumors berechnet, der freigestellt – also sichtbar gemacht – werden soll. Geometrie innerhalb des so entstandenen Kreises wird ausgeblendet. Die vertikale Position des Zylinders richtet sich nach der Tiefe des hervorzuhebenden Tumors, so dass dahinter liegende Strukturen sichtbar bleiben.

Durch die Verwendung eines Fragment-Shaders können die betroffenen Fragmente einfach modifiziert werden. Bei einem Zylinder mit dem Mittelpunkt (x_M, y_M), Radius r_M und maximaler Tiefe z_M sowie einem Fragment mit der Position (x_F, y_F, z_F) gilt:

$$\sqrt{(x_M - x_F)^2 + (y_M - y_F)^2} - r_M \begin{cases} < 0 - \epsilon, \Rightarrow \text{Fragment ausblenden} \\ = 0 \pm \epsilon, \Rightarrow \text{farbliche Umrandung} \\ > 0 + \epsilon, \Rightarrow \text{Fragment normal behandeln} \end{cases}$$

Durch ϵ wird dabei die Breite der farblichen Umrandung angegeben. Zur Bestimmung des Zylinders müssen die Eckpunkte des Tumors zunächst auf den Bildschirm projiziert werden, um anschließend den Mittelpunkt der 2D-Punktmenge bestimmen zu können. Die Größe des kleinsten umschließenden Kreises wird auf der CPU bestimmt [7]. Der am weitesten hinten liegende projizierte Punkt wird dabei als z_M für die vertikale Platzierung des Zylinders verwendet.

Zur Beschleunigung der Berechnung wird die dreidimensionale konvexe Hülle des Tumors ermittelt, da nur diese Punkte zur weiteren Verarbeitung interessant sind [8]. Durch diese einmalige Berechnung kann die Zahl der zu projizierenden Punkte stark reduziert werden.

2.2 Section View

Beim *Section View* wird ein vorn abgerundeter Kegel als Schnittgeometrie verwendet, um zu entscheiden, welches Fragment ausgeblendet werden soll und welches nicht. Dadurch werden die Schnittflächen zwischen dem Kegel und den Strukturen sichtbar. Durch die gleichmäßige Form des Kegels hebt sich der Schnitt gut hervor, sodass die Manipulation der Szene deutlich wird. Die Abrundung verhindert, dass die Spitze des Kegels als Feature in den Daten missverstanden wird. Der Kegel wird dabei genauso positioniert wie der Zylinder des *Ghost Views*. Die abgerundete Spitze wird an z_M ausgerichtet. An der Stelle, an der er von der Halbkugel zum normalen Kegel übergeht, beträgt der Radius des Kegels r_M.

Der grobe Ablauf des Algorithmus ist wie folgt: Zunächst muss bestimmt werden, welche Strukturen von dem Kegel geschnitten werden. Bei dem Schnitt mit dem Kegel können drei Fälle auftreten:

1. die Struktur liegt komplett außerhalb des Kegels,
2. die Struktur liegt komplett innerhalb des Kegels oder
3. die Struktur wird vom Kegel geschnitten.

Genau genommen muss von allen geschnittenen Strukturen diejenige bestimmt werden, dessen Vorderseite am nächsten zur Schnittfläche liegt. Dieser Test wird pro Fragment durchgeführt. Die Fragmente, die den Test bestanden haben, werden anschließend so manipuliert, dass der Farbwert mit der Normalen des Schnittkegels berechnet wird. Die vorhandene Geometrie wird also genutzt, um einen Eindruck einer CSG-Berechnung zu simulieren.

Zunächst wird einmalig die Schnittgeometrie in eine Textur mit Größe des Ausgabebildes gerendert. Die RGB-Werte speichern hierbei die Richtung der Normalen der Schnittgeometrie, in unserem Fall des Kegels. Der α-Kanal speichert die Tiefe der Schnittgeometrie.

Die benötigten Fragmente werden über eine Kombination aus Stencil- und Depthtest ermittelt. Getestet werden ausschließlich die Fragmente, die innerhalb der Schnittgeometrie liegen. Der Stencilbuffer wird so parametrisiert, dass er beim Rendern einer Vorderseite inkrementiert, bzw. beim Rendern einer Rückseite dekrementiert wird. Wenn der Stencilbuffer einen positiven Wert aufweist,

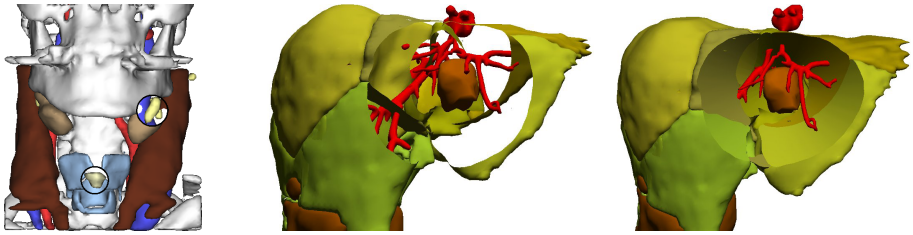
wird der Depthbuffer an dieser Stelle auf die Tiefe der Vorderseite gesetzt. An dieser Position wird die Struktur vom Kegel angeschnitten, da sie hier nicht durch eine Rückseite geschlossen wird. Dieser Test wird pro Struktur durchgeführt, da sonst nicht differenziert werden kann, ob eine Struktur komplett innerhalb des Kegels liegt, bzw. welche Struktur vom Kegel geschnitten wird.

Der Stencilbuffer wird nach dem Rendern jeder Struktur wieder gelöscht. Der Depthbuffer wird jedoch erhalten, da nur so bestimmt werden kann, welche Vorderseite am nächsten an der Schnittgeometrie liegt. Der Depthtest wird dazu im Gegensatz zum normalen Rendering invertiert (*Lower Equal*), d.h. es wird immer der z-Wert der weiter entfernten Struktur gespeichert.

Durch den Aufbau der Stencil- und Depthtest-Kombination wird der Depthbuffer nur an den Positionen verändert, an denen die Strukturen vom Kegel geschnitten werden. Die Beleuchtung der Fragmente an diesen Positionen muss manipuliert werden, um den Schnitt mit dem Kegel zu simulieren. An allen anderen Positionen wird die Szene in einem zweiten Rendering-Schritt normal gerendert. Das Rendering wird dabei auf Fragmente eingeschränkt, die außerhalb des Kegels liegen. Dadurch werden alle Strukturen verworfen, die komplett innerhalb des Schnittvolumens liegen.

3 Ergebnisse

Beide Verfahren wurden auf einer NVIDIA GeForce 9600 GT Grafikkarte getestet. Bei der Laufzeituntersuchung des *Ghost Views* konnten keine nennenswerten Unterschiede in der Frame-Rate im Gegensatz zum Rendering ohne *Ghost View* festgestellt werden, der *Section View* konnte mit 500fps gerendert werden.



(a) *Ghost View* (b) *Section View*

Abb. 1. (a) Mithilfe des *Ghost View* werden ein Tumor und ein vergrößerter Lymphknoten hervorgehoben, welche sonst verdeckt wären. (b) Darstellung einer Leber mit den portalvenösen Versorgungsgebieten sowie der Portalvene. Links sind die „leeren“ Strukturen nach dem Schnitt mit dem Kegel zu sehen. Rechts wird die CSG-Simulation angewendet. Die Szene beinhaltet ca. 70000 Dreiecke und wird inkl. *Section View* mit ca. 500fps gerendert.

4 Diskussion

Die Einsatzmöglichkeiten dieser Techniken liegen vor allem in der Therapieplanung. Für ausgewählte Fragestellungen, z.B. operative Entfernung von Halstumoren, ist eine Evaluierung geplant, um zu untersuchen, ob die Zielgruppe die räumlichen Verhältnisse schneller und genauer einschätzen kann.

Der *Ghost View* eignet sich sehr gut zur Hervorhebung eines oder mehrerer Tumoren, da nur eine kleine Region um den Tumor verändert wird. Bei dem *Ghost View* ist leider nicht zu erkennen, wie weit hinten der Tumor liegt, da *alle* davor liegenden Strukturen entfernt werden. Der Abstand zu direkt angrenzenden Strukturen kann ebenfalls nicht eingeschätzt werden, da diese ebenfalls entfernt werden.

Um diesen Nachteil zu beheben, wurde die Technik zu einem *Section View* erweitert. Durch den *Section View* kann die Umgebung eines Tumors sehr gut exploriert werden. Voraussetzung für den beschriebenen Ansatz ist, dass die Oberfläche geschlossen ist, da sonst nicht bestimmt werden kann, ob sich die jeweilige Struktur mit dem Kegel schneidet. Zurzeit ist es nicht möglich, den *Section View* mit transparenten Oberflächen zu nutzen, da sonst für jede einzelne Struktur die Schnittinformation gespeichert werden muss. Eine Kombination mit einem *Depth Peeling*-Ansatz ist daher geplant [9]. Da der Stenciltest für jede Struktur einzeln durchgeführt werden muss, wären auch einzelne Schnittgeometrien pro Struktur möglich, um die verdeckende Geometrie zwiebelartig entfernen zu können.

Mit Ausnahme der Ermittelung der Zylinder- bzw. Kegelparameter werden alle auszuführenden Berechnungen direkt auf der Grafikkarte ausgeführt. Die Techniken sind relativ einfach zu implementieren und haben ein sehr gutes Laufzeitverhalten.

Literaturverzeichnis

1. Viola I, Gröller ME. Smart Visibility in Visualization. In: Proc. of EG Workshop on Computational Aesthetics in Graphics, Visualization and Imaging; 2005. .
2. Zhou J, Hinz M, Tönnies KD. Hybrid Focal Region-Based Volume Rendering of Medical Data. In: Bildverarbeitung für die Medizin. Springer; 2002. p. 113–116.
3. Viola I, Kanitsar A, Gröller ME. Importance-Driven Volume Rendering. In: IEEE Visualization. IEEE Computer Society Press; 2004. p. 139–145.
4. Diepstraten J, Weiskopf D, Ertl T. Interactive Cutaway Illustrations. Computer Graphics Forum. 2003;22(3):523–532.
5. Li W, Ritter L, Agrawala M, Curless B, Salesin D. Interactive Cutaway Illustrations of Complex 3D Models. In: ACM SIGGRAPH 2007 papers. ACM Press; 2007. p. 31.
6. Kirsch F, Döllner J. OpenCSG: A Library for Image-Based CSG Rendering. In: Proc. of the USENIX Annual Technical Conference. USENIX; 2005. p. 129–140.
7. Welzl E. Smallest Enclosing Disks (Balls and Ellipsoids). In: New Results and New Trends in Computer Science. vol. 555 of LNCS. Springer; 1991. p. 359–370.
8. Preparata FP, Shamos MI. Computational Geometry: An Introduction. Springer; 1985.
9. Everitt C. Interactive Order-Independent Transparency. NVIDIA; 2001.