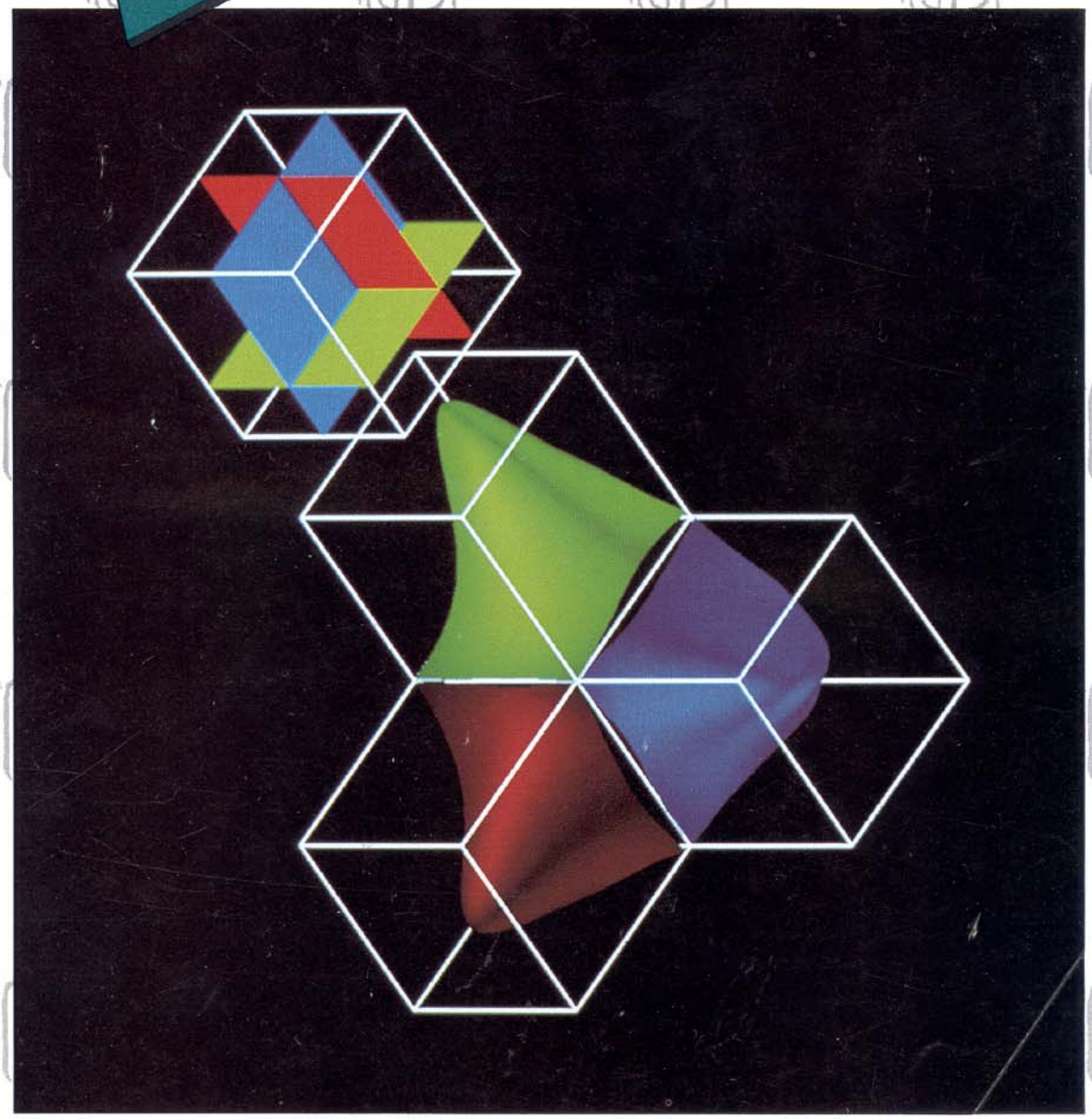
CAD und Computergraphik



13. Jahrgang (1990) • Nummer 5 • Februar 1991

Erscheinungsort Wien • Verlagspostamt 1040 Wien • P.b.b. • S 60.- • DM 10.-
ISSN 1011-646X • Imprimés à taxe réduite • Taxe perçue • Postgebühr bar bezahlt
Bureau de poste A-1040 Vienne • Autriche

Visualisierungstechniken zur Darstellung dreidimensionaler Datenmengen

Bernd Hamann

In diesem Beitrag sollen unterschiedliche Darstellungsarten dreidimensionaler Datenmengen erläutert werden. Es wird vorausgesetzt, daß Punkte mit skalaren (oder auch vektorwertigen) Funktionswerten gegeben sind, die in einer verständlichen Weise auf einem Bildschirm darzustellen sind. Die beschriebenen Verfahren gehören somit in den Bereich des sogenannten ***Volume Renderings*** *oder* ***Volume Visualization****. Anwendungen für diese Methoden finden sich im Bereich der Meteorologie (Temperatur-, Druckdaten), der medizinischen Diagnostik (Computer-Tomographie CT), Magnetische Resonanz-Tomographie MRI), der Physik und der Mathematik (Funktionen dreier Veränderlicher). Die Verfahren sollen möglichst einfach zu implementieren sein und ein gutes Verständnis der zu untersuchenden Daten bieten. Weiterhin müssen die Algorithmen schnell sein, so daß eine Echtzeitverarbeitung möglich ist.*

1. Problemstellung und Klassifizierung der Methoden

Es wird davon ausgegangen, daß nur zwei Arten dreidimensionaler Datenmengen zu visualisieren sind: entweder sind die Daten von der Form

$$\{(\mathbf{x}_i, f_i) \mid \mathbf{x}_i \in \mathbf{R}^3, f_i \in \mathbf{R}, i = 1...n\} \tag{1}$$

unregelmäßig im Raum verteilt oder sie sind in einer regelmäßigen Struktur der Form

$$\{(\mathbf{x}_i, f_i) = (x_i, y_j, z_k, f_{i,j,k}) \mid \mathbf{x}_i \in \mathbf{R}^3, f_i \in \mathbf{R}, \\ i = 0...n_x, j = 0...n_y, k = 0...n_z\} \tag{2}$$

angeordnet. Diese Daten können sowohl originale Meßdaten als auch Daten sein, die erst durch ein approximierendes oder interpolierendes Verfahren gewonnen worden sind. Sind ausreichend viele Meßdaten gegeben, um eine sehr feine Auflösung auf dem Bildschirm zu gewähren, so sollten die Daten direkt visualisiert werden. Ist die Anzahl der Datenpunkte aber zu gering, so müssen zunächst mathematische Methoden zur Approximation/Interpolation der Daten bestimmt und die resultierende Funktion an geeigneten Punkten im Raum und in ausreichender Auflösung ausgewertet werden. Sind Daten in unstrukturierter Form nach (1) gegeben, so sollten in jedem Fall zunächst eine approximierende Funktion ermittelt werden, um nach deren Auswertung eine Struktur nach (2) zu erhalten. Diese Form der Datenaufbereitung ermöglicht eine effizientere Anwendung der in den folgenden Kapiteln vorgeschlagenen Verfahren.

Auf die Methode zur Approximation/Interpolation soll hier nicht eingegangen werden. Gute Überblicke geben: [ALFE89], [BARN85], [BOEH84] und [FRAN90]. Methoden zur Qualitätsanalyse solcher Verfahren werden in [RATH88] erörtert. Hat man sich für ein bestimmtes Verfahren entschieden, so wird die sich ergebenen Funktion f nur auf dem Gebiet

$$D = \{\mathbf{x} = (x,y,z)^T \mid \\ x \in [x_{min}, x_{max}], y \in [y_{min}, y_{max}], z \in [z_{min}, z_{max}]\} \tag{3}$$

betrachtet und an $(n_x + 1)(n_y + 1)(n_z + 1)$ Gitterpunkten ausgewertet, so daß die Datenmenge

$$\{(x_i, y_j, z_k, f(x_i, y_j, z_k)) \mid i = 0...n_x, j = 0...n_y, k = 0...n_z\} \quad (4)$$

dargestellt werden muß. Hierbei wird eine Partition wie folgt angenommen:

$$x_i = x_{min} + i \cdot dx, \quad dx = \frac{x_{max} - x_{min}}{n_x}, \quad i = 0...n_x,$$
$$y_i = y_{min} + j \cdot dy, \quad dy = \frac{y_{max} - y_{min}}{n_y}, \quad j = 0...n_y, \quad (5)$$
$$z_k = z_{min} + k \cdot dz, \quad dz = \frac{z_{max} - z_{min}}{n_z}, \quad k = 0...n_z.$$

Die Schreibweise in (1) und (2) macht deutlich, daß für bestimmte Raumpunkte Funktionswerte festgeschrieben sind. Im folgenden werden ausschließlich Verfahren erörtert, die nur skalare, keine vektorwertigen Funktionswerte zulassen. Sollen vektorwertige Daten analysiert werden, so müssen die Methoden entweder komponentenweise angewendet oder grundsätzlich andere Techniken benutzt werden.

Eine sehr rechenintensive Darstellungsmethode ist die Strahlverfolgung. Vereinfacht dargestellt setzt dieses Verfahren nur die Definition eines Augpunktes, einer Position und Orientierung des Datenvolumens und eine durch Position des Datenvolumens und Augpunkt bestimmte Strahlrichtung voraus. Für jeden Bildpunkt auf dem Sichtschirm wird nun ein Strahl ausgesandt, der in das Datenvolumen bis zu einer bestimmten Tiefe eindringen kann (je nach Transparenz-Eigenschaften). Entlang dieses Strahles werden in festgelegter Diskretisierung die getroffenen skalaren Werte im Datenvolumen aufsummiert. Diese Summe legt zugleich die Farbe der zugehörigen Bildschirmposition fest. Techniken der Strahlverfolgung zur Visualisierung der hier gegebenen Volumendaten werden zum Beispiel in den Arbeiten [KAJI84], [LEVO88] und [SABE88] behandelt.

Besonders im Bereich der medizinischen Diagnostik haben sich solche Verfahren bereits zu einem großen Teil durchgesetzt, da sie im Gegensatz zu herkömmlichen zweidimensionalen Schichtaufnahmen von Körperteilen die räumliche Struktur eines Objekts unmittelbar erkennen lassen. Wenn diese Methoden vom algorithmischen Standpunkt auch einfach erscheinen, so erfordern sie doch einen sehr hohen Rechenaufwand, so daß eine rasche Darstellung der dreidimensionalen Daten kaum möglich ist. In [FOLE90] wird beschrieben, wie eine erhebliche Beschleunigung solcher Verfahren erfolgen kann. Die Idee beruht hier darauf, eine feste Anzahl an Bildern für verschiedene Orientierungen (verschiedene Augpunkte) unter Benutzung von Strahlverfolgung zu berechnen und diese

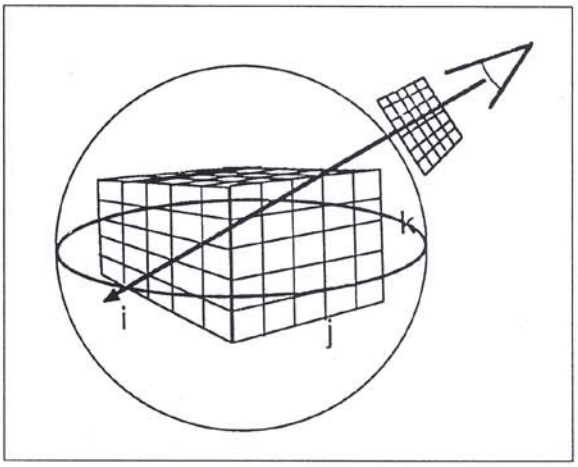


Abb. 1: Strahlverfolgung in Datenvolumina

Bilder anschließend zu *interpolieren*. Wenn ein einzelnes Bild aus $m \cdot n$ Bildpunkten besteht, so werden für jeden einzelnen Bildpunkt (i, j) Splines ermittelt, die die Farbwerte für diesen Punkt in der berechneten Bildsequenz interpolieren. Es ist daher erforderlich, $m \cdot n$ Splines zu ermitteln.

Die *Abbildungen 2* und *3* sind unter Anwendung der Algorithmen von Levoy [LEVO88] bzw. Sabella [SABE88] erstellt worden. Die in *Abbildung 2* dargestellte Datenmenge basiert auf 68 • 64 • 64 Dichtewerten (Computertomographie, Datenstruktur (2)), in *Abbildung 3* sind Partikeldichten im Raum simuliert worden.

Es sollen hier unkomplizierte Algorithmen beschrieben werden, die aussagekräftige Ergebnisse in Echtzeit liefern. Zuerst wird die *Quadermethode* beschrieben, die darauf basiert, das gegebene Datenvolumen D (3) in Untervolumina zu zerlegen und diese dann als eine Menge kleiner Quader darzustellen. Diese Technik kann noch dadurch verbessert werden, daß all jene Untervolumina Transparenzeigenschaften zugeschrieben werden, wodurch alle Quader durchscheinend werden. Dieses wird in der *transparenten Quadermethode* erläutert. Die *Schnittmethode* beruht darauf, das Datenvolumen (3) mit Ebenen zu schneiden, so daß man Polygonflächen als Schnitte erhält, die dann gemäß den dort vorgefundenen skalaren Daten und der zugeordneten Farbtafel coloriert werden. Die *Zerlegungstechnik* stellt eine Funktion dreier Veränderlicher als eine Menge bivariater Funktionen dar, wobei jeweils eine der drei Variablen festgehalten bleibt. Verfahren, die mehr auf mathematischen und CAGD-Methoden beruhen, werden unter *Kontur- und CAGD-Methoden* diskutiert. Dabei wird zunächst eine Datenreduktion angestrebt, eine erste

auf Dreiecken beruhende Approximation einer Kontur bestimmt, diese Triangulierung dann verbessert und schließlich eine normalenstetige Fläche als glatte Kontur-Approximation berechnet. Die Berechnung von Konturen und Triangulierungen im Raum wird zum Beispiel in [CHOI88] und [LORE87] betrachtet. In [ZUCK81] wird eine Methode zur Gradienten-Approximation angegeben. Dreieckspflaster werden unter anderem ausführlich in [FARI86], [FARI88], [HAGE89], [HAMA90], [HOSC89] und [NIEL87] erörtert.

2. Die Quadermethode

Diese Methode setzt als Eingabe Daten der Form (3) voraus. Die Idee besteht hier darin, das gegebene Gesamtvolumen (Quader) in kleinere Untervolumina einzuteilen, zwischen denen jeweils ein gewisser Freiraum gelassen wird, so daß man in das Gesamtvolumen "hineinschauen" kann. Zunächst müssen die Anzahl darzustellender Quader und die Größe des Freiraumes zwischen je zwei Quadern bestimmt werden. Die Parameter q_x, q_y und q_z sind die Anzahl Quader in x-, y- und z-Richtung. Die Größe des Freiraumes zwischen den Quadern wird bestimmt durch das Längenverhältnis von Freiraum und Quader: Freiraumlänge / Quaderlänge. Dieses Verhältnis ist für jede Koordinate festzulegen. Die Symbole für diese Verhältnisse sind α_x, α_y und α_z. Die absoluten Ausmaße der Quader sind damit durch die Verhältnisse bestimmt. Die linke vordere untere Ecke eines Quaders $Q_{i,j,k}$ ist gegeben durch das Tripel $(x_i, x_j, x_k)^T$:

$$x_i = x_{min} + i\,(1+\alpha_x)\Delta x,\ \Delta x = \frac{x_{max}-x_{min}}{q_x+\alpha_x(q_x-1)},\ i = 0...(q_x-1),$$

$$y_j = y_{min} + j\,(1+\alpha_y)\Delta y,\ \Delta y = \frac{y_{max}-y_{min}}{q_y+\alpha_y(q_y-1)},\ j = 0...(q_y-1), \quad (6)$$

$$z_k = z_{min} + k(1+\alpha_z)\Delta z,\ \Delta z = \frac{z_{max}-z_{min}}{q_z+\alpha_z(q_z-1)},\ k = 0...(q_z-1).$$

Die Werte Δx, Δy und Δz geben also die Ausmaße der Quader an. Die betrachtete Funktion wird nun an den acht Eckpunkten aller Quader ausgewertet; minimaler und maximaler auftretender Funktionswert werden zugleich gespeichert. Jede der sichtbaren Seitenflächen eines Quaders wird Gouraud-schattiert, d.h. es werden die Farben für die vier Eckpunkte einer Seitenfläche ermittelt, anschließend werden sie bilinear interpoliert. Hierbei wird eine lineare Abbildung zwischen auftretenden Funktionswerten und verwendeten Farben vorgenommen, somit der minimale Funktionswert auf die Farbe mit minimalem Index, der maximale Funktionswert auf die Farbe mit maximalem Index abgebildet. Die Darstellung und Rotation der insgesamt $q_x q_y q_z$ Quader ist in Echtzeit möglich. Die räumliche Gitterstruktur dieser Visualisierungstechnik kann dadurch noch verbessert werden, daß Linien zwischen den Schwerpunkten benachbarter Quader gezogen werden (in x-, y- und z- Richtung).

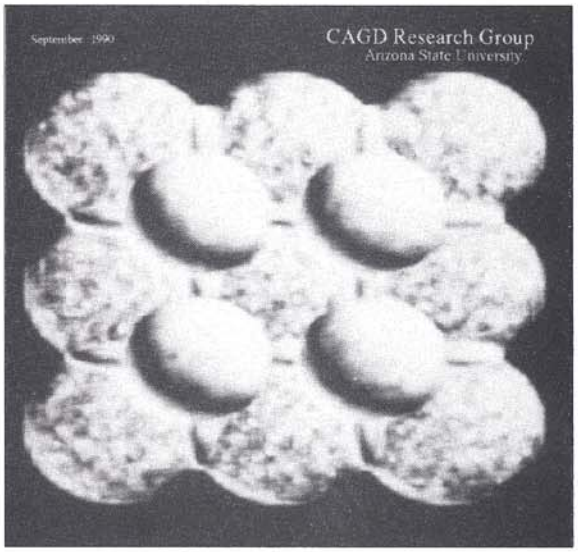


Abb. 3: Strahlverfolgung nach Sabella

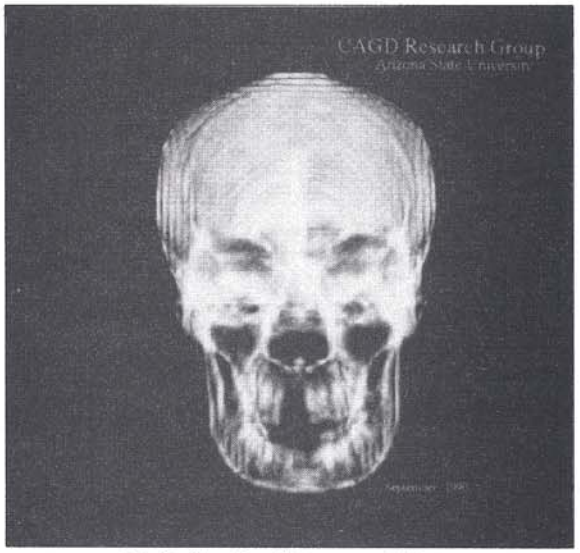


Abb. 2: Strahlverfolgung nach Levoy

Die hier beschriebene Methode kann auch auf Volumina angewendet werden, die durch eine Abbildung der Form

$$(x,y,z)^T = (x(u,v,w),\ y(u,v,w),\ z(u,v,w))^T,$$
$$u\in[u_{min},u_{max}],\ v\in[v_{min},v_{max}],\ w\in[w_{min},w_{max}], \quad (7)$$

entstanden sind. Die Quader sind hier Quader im uvw-Raum, die nicht unbedingt auf Quader im xyz-Raum abgebildet

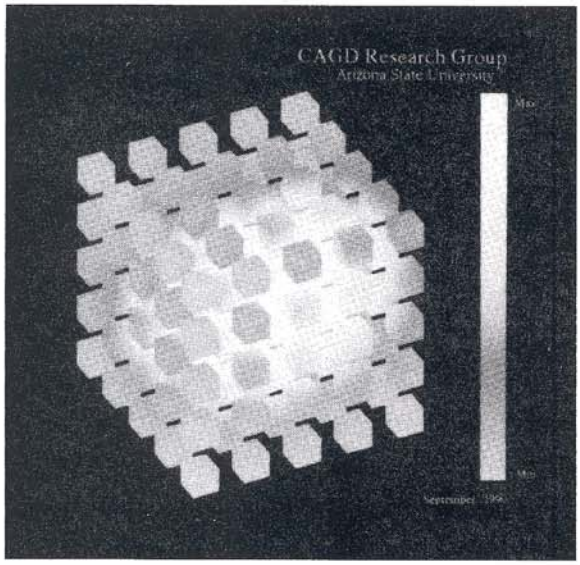


Abb. 4: Quadermethode mit
$q_x = q_y = q_z = 5$ **und** $\alpha_x = \alpha_y = \alpha_z = 1$

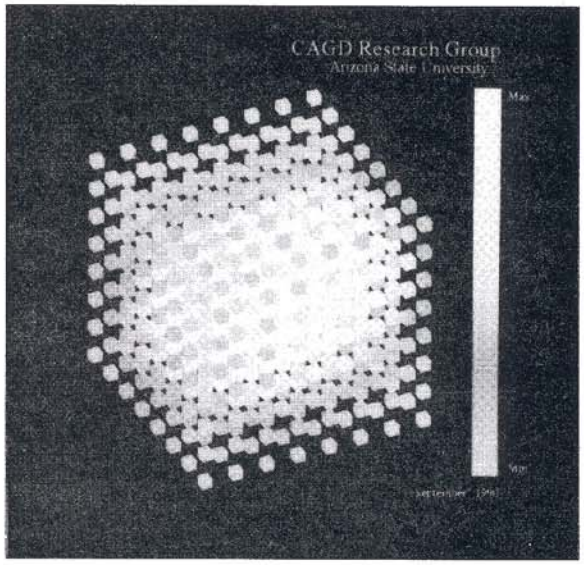


Abb. 5: Quadermethode mit
$q_x = q_y = q_z = 8$ **und** $\alpha_x = \alpha_y = \alpha_z = 2$

werden müssen. Auf diese Weise lassen sich z.B. Funktionen repräsentieren, die auf einem Kugelvolumen definiert sind.

In den *Abbildungen 4* und *5* wird das Verfahren mit verschiedenen Auflösungsparametern gezeigt. Die hierbei verwendete Funktion dreier Veränderlicher lautet

$$f(x, y, z) = 15 \cdot (e^{-\frac{5}{1000}((x-10)^2+(y-10)^2+(z-10)^2)} + e^{-\frac{25}{10000}((x-15)^2+(y-20)^2+(z-20)^2)} + e^{-\frac{5}{1000}((x-25)^2+(y-25)^2+(z-25)^2)}).$$

Hierbei ist $x,y,z \in [0,39]$. *Abbildung 6* stellt die Funktion $f(x,y,z) = \cos(2 \cdot \sqrt{x^2+y^2+z^2})$ auf einem Teil eines (Einheits-) Kugelvolumens dar. Die Zerlegung führt somit nicht auf Quader. Das Volumen selbst ist bestimmt durch die Abbildung

$$(x,y,z)^T = (u \cos v,\ u \sin v \cos w,\ u \sin v \sin w)^T.$$

Hierbei gilt $u \in [0, 1]$, $v \in [0, \pi]$ und $w \in [0, \frac{3}{2}\pi]$.

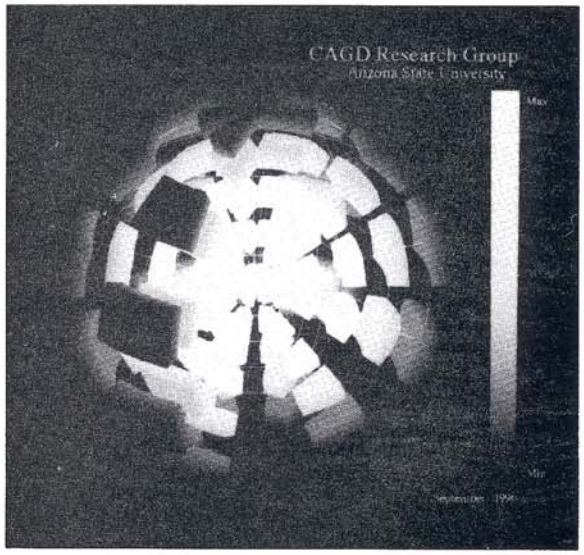


Abb. 6: Zerlegung eines Kugelvolumens

3. Die transparente Quadermethode

Die jetzt beschriebene Technik macht Gebrauch vom sogenannten *Alpha-Blending*, einem Verfahren das von einigen Graphik-Rechnern von Hardwareseite unterstützt wird und durchscheinende Polygonflächen zuläßt. Im folgenden wird der Begriff *Polygon* auch benutzt werden, wenn die durch das (ebene) Polygon begrenzte Fläche gemeint ist. Die darzustellenden Polygone (eben, geschlossen) müssen dazu gemäß Abstand zur Bildebene von hinten nach vorne in sortierter Reihenfolge vorliegen. Ein Parameter legt den Grad der Transparenz aller Polygone fest. Ist c_{alt} die alte Farbe für Bildschirmpunkte (i, j), c_{poly} die Farbe des eben darzustellenden Polygons an Position (i, j) und t der global verwendete Transparenz-Parameter, so erhält man die neue Farbe an (i, j) durch lineare Interpolation:

$c_{neu} = (1-t)c_{poly} + tc_{alt}$, wobei $t \in [0,1]$ gilt.
Gilt $t = 0$, so wird damit Transparenz unterdrückt.

Abb. 7: Transparente Quadermethode mit $n_x = n_y = n_z = 9$ und $t = 0.5$

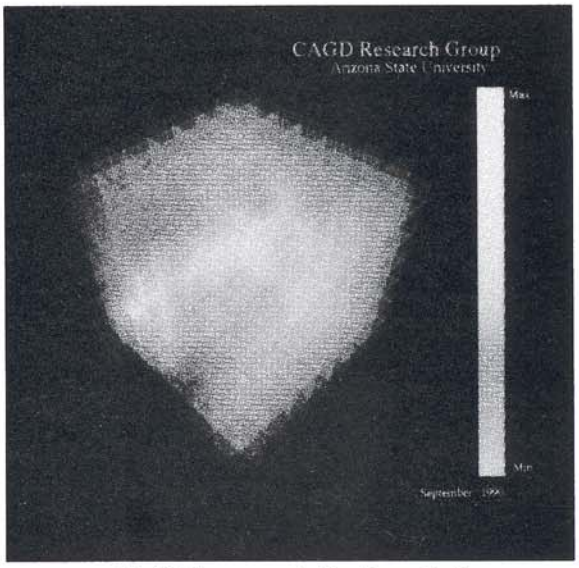


Abb. 8: Transparente Quadermethode mit $n_x = n_y = n_z = 9$ und $t = 0.95$

Die Idee des Freiraumes nach Verfahren 1 kann wegen der Transparenz der Polygone aufgegeben werden. Sind n_x, n_y und n_z die spezifizierten Auflösungsparameter, so wird das Datenvolumen nach (3) in n_x $(n_y - 1)(n_z - 1)$ Rechtecke parallel zur yz-Ebene, $(n_x - 1)$ n_y $(n_z - 1)$ Rechtecke parallel zur xz-Eben und $(n_x - 1)(n_y - 1)n_z$ Rechtecke parallel zur xy-Ebene eingeteilt. Somit sind ungefähr 3- $n_x n_y n_z$ Rechtecke darzustellen. Der Sortieraufwand für diese Anzahl Rechtecke ist bei feiner Auflösung erheblich. Daher ist Echtzeitverarbeitung nur für variierende Transparenz t bei konstanter Orientierung des Datenvolumens (3) möglich. Eine Veränderung der Orientierung zieht eine Neusortierung der Polygone nach sich, so daß interaktives Arbeiten nicht mehr gewährleistet ist.

In den *Abbildungen 7* und *8* wird eine Gas-Konzentration mit verschiedenen Transparenzeigenschaften gezeigt. Zur Art der Daten sei hier auf [LONG89] verwiesen.

4. Die Schnittmethode

Bei der Schnittmethode wird das Datenvolumen der Form (3) mit Ebenen geschnitten und die Daten auf den sich ergebenden Schnittpolygonflächen farblich dargestellt. Der Einfachheit halber werden hier Ebenen parallel zur xy-, xz- und yz-Ebene benutzt. Somit ergeben sich als Schnitte Rechtecke, auf denen die trivariate Funktion in festgelegter Diskretisierung ausgewertet wird. Alle damit entstehenden Rechtecke werden nun Gouraud-schattiert ausgegeben. Werden als Auflösungsparameter n_x, n_y und n_z festgelegt, so erhält man eine Darstellung der Funktion $f(x,y,z)$ als Restriktion von f auf drei Rechtecke, also $f(x_i,y,z)$, $y \in [y_{min},y_{max}]$, $z \in [z_{min},z_{max}]$, $i = 0...(n_x-1)$, $f(x,y_j,z)$, $x \in [x_{min},x_{max}]$, $z \in [z_{min},z_{max}]$, $j = 0...(n_y-1)$, und $f(x,y,z_k)$, $x \in [x_{min},x_{max}]$, $y \in [y_{min},y_{max}]$, $k = 0...(n_z-1)$. Die Unterteilung der drei Definitionsintervalle von f geschieht dazu wieder äquidistant. Hierbei können interaktiv entweder einer der drei Indizes i, j oder k oder aber Orientierung der gezeigten Rechtecke modifiziert werden. Es ist natürlich möglich, eine Funktion erheblich feiner als in den *Abbildungen 9* und *10* aufzulösen, wenn dadurch ein besseres Verständnis gegeben ist.

In den *Abbildungen 9* und *10* wird noch einmal dieselbe Gas-Konzentration wie in den *Abbildungen 7* und *8* gezeigt. Eine Änderung der Farbskalen kann dazu beitragen, daß bestimmte Eigenschaften der Daten besser erkannt werden können.

5. Die Zerlegungstechnik

Soll eine Funktion zweier Veränderlicher veranschaulicht werden, so werden die Funktionswerte gewöhnlich über der xy-Ebene und dort über dem jeweiligen Definitionsgebiet aufgetragen. Die gleiche Technik wird hier verwendet, indem ganze Mengen jeweils bivariater Funktionen der Formen $f_i(y, z) = f(x_i, y, z)$, $i = 0...(n_x - 1)$, $f_j(x, z) = f(x, y_j, z)$, $j = 0...(n_y - 1)$, und $f_k(x, y) = f(x, y, z_k)$,

Abb. 9: Schnittmethode mit $n_x = 80$, $n_y = 130$, $n_z = 20$

Abb. 10: Schnittmethode mit $n_x = 80$, $n_y = 130$, $n_z = 20$, einfarbige Farbskala

$k = 0...(n_z - 1)$ betrachtet werden. Sind die drei festzuhaltenden Variablen x_i, y_j und z_k bestimmt, so werden die drei korrespondierenden bivariaten Funktionen $f_i(y,z)$, $f_j(x,z)$ und $f_k(x,y)$ als Flächen über drei geeignete Seitenflächen des Datenvolumens nach (3) dargestellt. Geeignet heißt hier, daß je eine Seitenfläche parallel zur xy-, xz- und yz-Ebene benutzt wird. Diese die Funktion repräsentierenden Flächen bestehen dabei aus Dreiecken, die die exakte Fläche linear approximieren und bei der Auswertung der

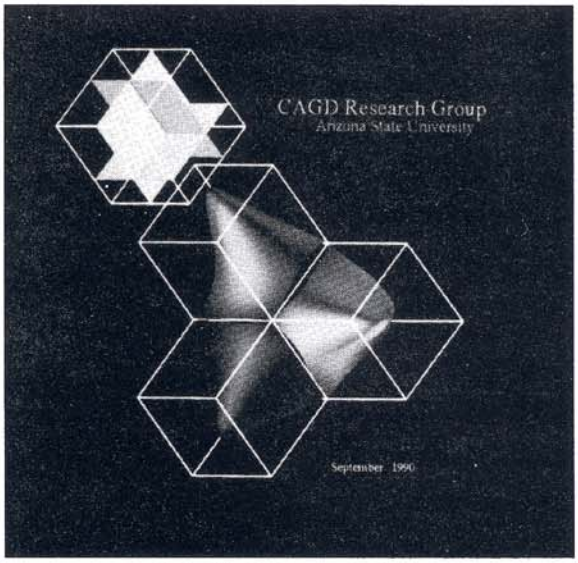


Abb. 11: Zerlegungstechnik

Funktion auf einem rechtwinkeligen Gitter entstehen. Schattierungen sind damit ebenfalls möglich. Da die entstehenden Graphen außerhalb des Definitionsvolumens verlaufen sollen, ist darauf zu achten, daß alle auftretenden Funktionswerte linear auf ein nicht-negatives Intervall transformiert werden. Dieses garantiert, daß die Graphen ausschließlich nach außen, über dem Datenvolumen aufgetragen werden können. Gleichzeitig wird in einem weiteren Quader die relative Lage der drei Ebenen $x = x_i$, $y = y_j$ und $z = z_k$, zueinander angezeigt.

In *Abbildung 11* wird die trivariate Funktion aus Kapitel 2 unter Anwendung der Zerlegungstechnik dargestellt.

6. Kontur- und CAGD-Methoden

Eine übliche CAGD-Vorgehensweise zum Verständnis diskreter Datenmengen im Raum besteht darin, eine an den Datenpunkten interpolierende Funktion zu berechnen und diese anschließend auf einem regelmäßigen Gitter auszuwerten und danach Konturen dieser Funktion als Flächen darzustellen. Es bietet sich aber noch eine andere Technik an: Man kann zu den Daten der Form (1) oder (2) zunächst eine stückweise lineare Approximation einer Kontur $f(x,y,z) = const$ in Form einer Menge von Dreiecken berechnen und diese dazu verwenden, eine glatte Fläche zu konstruieren, die die berechneten Eckpunkte all dieser Dreiecke interpoliert. Dieses Vorgehen scheint sich vor allem im medizinischen Bereich zur Visualisierung von CT- und MRI-Daten durchzusetzen. Dieses Verfahren läßt sich im wesentlichen in folgende Schritte einteilen.

6.1 Datenglättung

Berücksichtigt man die Tatsache, daß die gegebenen Daten mit Meßfehlern behaftet sind, so sollte als erster Schritt eine Datenglättung erfolgen. Ist über die Art der Datengewinnung und damit der Meßfehler nichts bekannt so bieten sich zwei Methoden an. Sind die Daten in Form (2) gegeben, so lassen sich je 27 Funktionswerte $f_{i,j,k}$ an den Datenpunkten $(x_{i+I}, y_{j+J}, z_{k+K})^T$, $I, J, K \in \{-1, 0, 1\}$, als Bézierordinaten $b_{r,s,t}$ eines Polynomes der Form

$$f(\bar{x},\bar{y},\bar{z}) = \sum_{t=0}^{2}\sum_{s=0}^{2}\sum_{r=0}^{2} b_{r,s,t} \cdot B_r^2(\bar{x})\, B_s^2(\bar{y})\, B_t^2(\bar{z}) \qquad (8)$$

deuten, wobei $B_r^2(\bar{x})$, $B_s^2(\bar{y})$ und $B_t^2(\bar{z})$ Bernsteinpolynome vom Grad 2 sind und $\bar{x}, \bar{y}, \bar{z} \in [0, 1]$ gilt (lokale Parameter). Wertet man die Summe nun für (0.5, 0.5, 0.5) aus, so erhält man den Funktionswert, durch den $f_{i,j,k}$ ersetzt wird. Die variationsreduzierende Eigenschaft der Bernsteinpolynome erklärt die glättende Wirkung dieser Technik.

Sind die Daten in Form (1) gegeben und ist zusätzlich eine Triangulierung im Raum gegeben (in Form einer Menge von Tetraedern), so läßt sich für jeden Datenpunkt $\mathbf{x}_i$ eine lokal approximierende Funktion f zu der Menge der Daten

$$\{(\mathbf{x}_i, f_i)\} \cup$$
$$\{(\mathbf{x}_j, f_j) \mid \overline{\mathbf{x}_i\mathbf{x}_j} \textit{ ist eine Kante in der Triangulierung}\}$$

in der Form

$$f(x,y,z) = \sum_{\substack{r,s,t\geq 0 \\ r+s+t\leq 2}} c_{r,s,t}\, x^r y^s z^t \qquad (9)$$

unter Anwendung der Methode der kleinsten Quadrate angeben. Auswertung von f an $\mathbf{x}_i$ ergibt den modifizierten Wert für den alten Funktionswert f_i.

6.2 Datenreduktion

Für diesen Schritt wird davon ausgegangen, daß die gegebene Punktmenge im Raum trianguliert vorliegt (Tetraeder). Daten der Form (1) werden derart trianguliert, daß man eine Delaunay-Triangulierung erhält, jeder einzelne *Würfel* impliziert in der Datenform (2) mit den Eckpunkten $(x_{i+I}, y_{j+J}, z_{k+K})^T$, $I, J, K \in \{0, 1\}$, kann in sechs Tetraeder unterteilt werden, um ebenso eine Delaunay-Triangulierung des gesamten Datenvolumens zu erhalten. In [LEMÉ89] wird ein Algorithmus angegeben, der für bivariate Daten der Form $\{(\mathbf{x}_j, f_j) \mid \mathbf{x}_j \in \mathbf{R}^2, f_i \in \mathbf{R}, i = 1...n\}$ (mit bekannter Delaunay-Triangulierung) iterativ Datenpunkte $\mathbf{x}_i$ entfernt, wenn deren Einfluß auf eine approximierende Funktion f vernachlässigbar klein ist. Der Einfluß eines Punktes $\mathbf{x}_i$ wird dabei wie folgt gemessen: Ist X_i die Menge aller Punkte, die mit $\mathbf{x}_i$ eine Kante in der gegebenen Triangulierung bilden, so bestimmen die Punkte in X_i ein (nicht unbedingt konvexes) geschlossenes Polygon P_i, so daß der Punkt $\mathbf{x}_i$ im Inneren dieses Polygons liegt. Über dem Gebiet begrenzt durch das Polygon P_i werden nun zwei Splines konstruiert. Der eine berücksichtigt (x_i, f_i) und alle Daten verbunden mit dem Randpolygon P_i, der andere berücksichtigt nur P_i und die Funktionswerte dort. Unterscheiden sich die beiden resultierenden Splines um weniger als eine vorgegebene Toleranz ε, so wird (x_i, f_i) von den Daten entfernt. Dieses Verfahren läßt sich leicht auf den trivariaten Fall übertragen. Hierbei ist P_i ein (nicht unbedingt konvexes) Polyeder im Raum und die beiden zu vergleichenden Splines werden über dem Gebiet im Raum verglichen, das durch das Polyeder P_i begrenzt ist. Eine ausführliche Darstellung dieses Verfahrens für den bivariaten Fall findet man in [LEMÉ89].

6.3 Bestimmung einer Kontur-Approximation

Liegt das gesamte Datenvolumen in Form einer Menge von Tetraedern vor, so wird eine Kontur $f(x,y,z) = const = c$ wie folgt approximiert. Sind f_a und f_b die Funktionswerte an den Endpunkten einer Tetraederkante $\overline{\mathbf{x}_a\mathbf{x}_b}$ und liegt c zwischen f_a und f_b, d.h. $c = (1-t)f_a + t f_b$, $t \in (0,1)$, so wird der Punkt $(1-t)\mathbf{x}_a + t\mathbf{x}_b$ als Approximation eines Punktes auf der Kontur betrachtet. Auf diese Weise schneidet eine Kontur einen Tetraeder entweder in drei oder vier Punkten längs der Kanten. Im ersten Fall bestimmen die drei Punkte ein Dreieck, im zweiten Fall muß das (ebene) Polygon mit vier Eckpunkten in zwei Dreiecke unterteilt werden. Die so gewonnenen Dreiecke werden nun als Approximation der Kontur(-Fläche) $f(x,y,z) = c$ betrachtet. Für den Fall, daß eine Kontur genau durch einen oder mehrere Eckpunkte dieser Tetraeder verläuft, ist eine Sonderbehandlung erforderlich. Dieses Problem läßt sich beheben, indem man nicht erlaubt, eine Kontur zu bestimmen, die mit einem Funktionswert an einem Datenpunkt übereinstimmt.

Sind die Daten in Form (2) gegeben und hat man sich dafür entschieden, diese Struktur beizubehalten (keine Datenreduktion), so gewinnt man eine Approximation der Kontur z.B. nach [LORE87]. Durch die zugrundeliegende Gitterstruktur der Datenpunkte ergeben sich 2^8 Möglichkeiten wie eine Kontur einen Datenwürfel mir den Eckpunkten $(x_{i+I}, y_{j+J}, z_{k+K})^T, I, J, K \in \{0, 1\}$ schneiden kann. Diese 256 Fälle erklären sich damit, daß acht Eckpunkte vorliegen und an jedem ein Funktionswert vorliegen kann, der entweder kleiner oder größer als die gesuchte Kontur $f(x, y, z) = c$ ist. Zunächst werden approximativ *Konturpunkte* auf den Würfelkanten durch lineare Interpolation der Funktionswerte längs der Kanten berechnet. Anschließend werden diese Punkte dazu benutzt, geschlossene *Konturpolygone* zu konstruieren, deren Kanten auf den Seitenflächen eines Würfels verlaufen. Diese Polygone

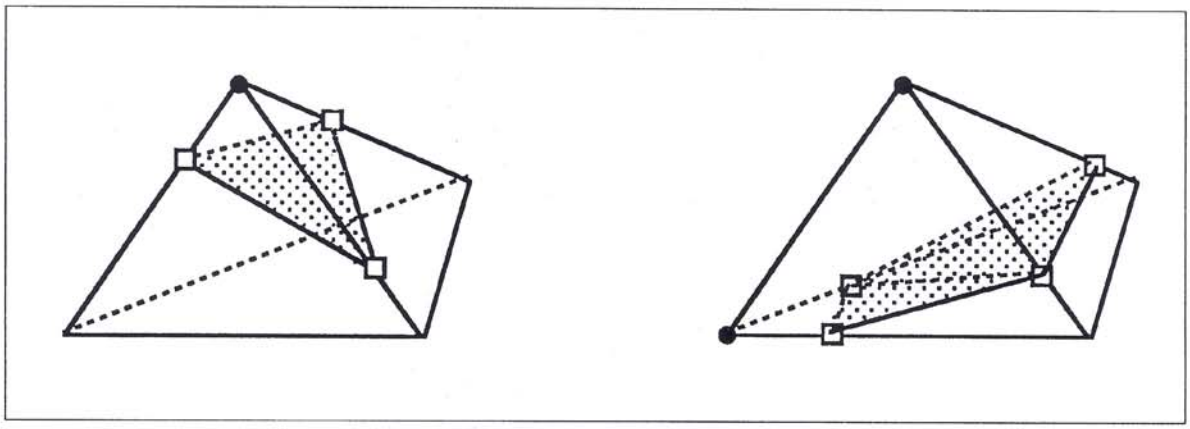


Abb. 12: Konturdreiecke in einem Tetraeder

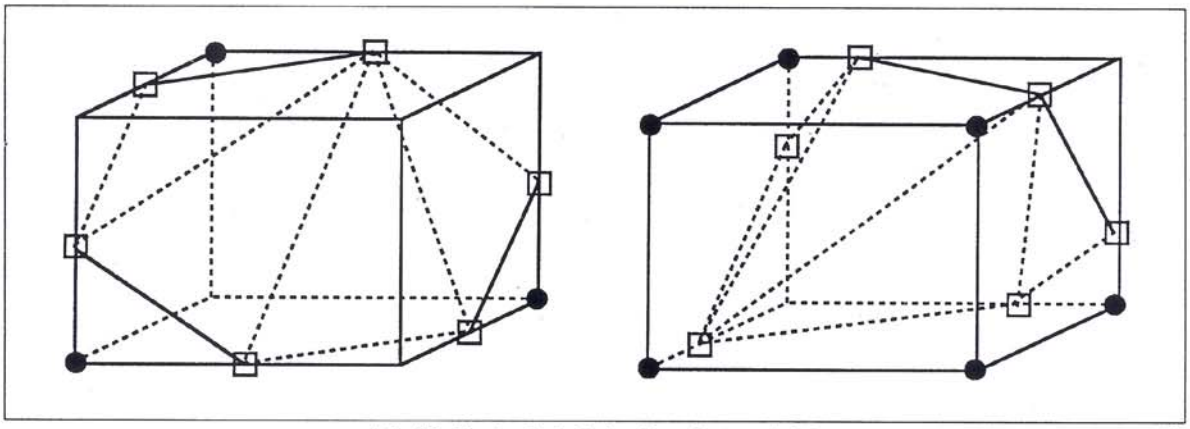


Abb. 13: Konturdreiecke in einem Datenwürfel

müssen nun geeignet trianguliert werden, um eine stückweise ebene Approximation der Kontur innerhalb eines Würfels zu gewinnen. In jedem Fall erhält man eine Menge von Polyedern (Kontur kann in mehrere nicht-zusammenhängende Teile zerfallen), die nur aus Dreiecken bestehen. Ein Algorithmus zur Verbesserung einer solchen Triangulierung im Raum findet sich z.B. in [CHOI88]. Als Qualitätsmaß der Triangulierung werden dort die Winkel zwischen Normalen benachbarter Dreiecke verwendet. Diese Winkel sollen dabei möglichst klein sein. Sind von vornherein Daten in sehr hoher Auflösung gegeben, so reicht es meistens aus, die jetzt vorliegenden Dreiecke direkt schattiert auf dem Bildschirm auszugeben. In *Abbildung 12* werden die zwei möglichen Fälle illustriert, wie eine Kontur innerhalb eines Tetraeders verläuft, in *Abbildung 13* werden zwei Beispiele gegeben, wie eine Kontur Datenwürfel schneiden kann und die resultierenden Konturpolygone innerhalb der Datenwürfel trianguliert werden können.

6.4 Gewinnung topologischer Informationen

Im folgenden soll beschrieben werden, wie effektiv die Nachbardreiecke eines Dreiecks berechnet werden und die Komponente der gesamten Kontur bestimmt werden kann, zu der ein bestimmtes Dreieck gehört. Es wird hier davon ausgegangen, daß in einer Liste jeder einzelne Konturpunkt in Form seiner x-, y- und z-Koordinaten gespeichert ist (geometrische Information), in einer zweiten Liste je Konturdreieck die drei Indizes jener drei Konturpunkte, die ein Dreieck bilden (Indizes bezüglich der ersten Liste). Zwei Dreiecke gelten als Nachbarn, wenn sie genau zwei Punkte gemeinsam haben. Effektiv werden die maximal drei Nachbardreiecke eines Dreiecks bestimmt, indem die Eckpunkte-Indizes je zweier Dreiecke in der zweiten Liste verglichen werden.

Da eine Kontur $f(x,y,z,) = const$ in mehrere nicht-zusammenhängende Komponenten zerfallen kann, muß für jedes Konturdreieck bestimmt werden, zu welcher Komponente

es gehört. Dies ist notwendig, wenn die einzelnen Komponenten in den folgenden Modellierschritten getrennt behandelt werden sollen. Die Zugehörigkeit zweier Dreiecke zu derselben Komponente ist wie folgt definiert. Zwei Dreiecke D_i und D_j gehören derselben Komponenete an, wenn Dreiecke D_{k_1}, D_{k_2}, ..., D_{k_m} existieren, so daß die Dreiecke D_i und D_{k_1}, D_{k_1} und D_{k_2}, ..., $D_{k_{m-1}}$ und D_{k_m} und D_{k_1} und D_j paarweise Nachbardreiecke sind. Effektiv wird die Komponente, zu der ein Dreieck gehört, iterativ bestimmt. Das erste Dreieck D_1 in der Liste aller Dreiecke wird der ersten Komponente zugewiesen, iterativ wird nun für jedes weitere Dreieck D_i festgestellt, ob eines seiner Nachbardreiecke bereits zu einer bestimmten Komponente gehört, wenn keines der Nachbardreiecke einer Komponente zugewiesen ist, so wird für D_i eine neue Komponente eingeführt, wenn wenigstens ein Nachbardreieck bereits einer Komponente zugewiesen ist, so wird unter all den Komponenten, denen die Nachbardreiecke von D_i angehören, eine ausgewählt, der D_i zugewiesen wird (existieren jetzt unter den Nachbardreiecken von D_i Dreiecke, die einer anderen Komponente als der von D_i zugewiesen sind, so muß sichergestellt werden, daß alle Dreiecke, die einer dieser anderen Komponenten angehören, derselben Komponente wie D_i zugewiesen werden).

6.5 Berechnung einer glatten Fläche

Die berechneten *Konturdreiecke* können jetzt dazu benutzt werden, eine normalstetige Fläche zu konstruieren, die durch die drei Eckpunkte jedes Dreiecks geht und außerdem vorgeschriebene Normalen an diesen Eckpunkten besitzt. Diese Normalen können z.B. durch geeignete Differenzen-Schemata als Gradienten der zugrundeliegenden trivariaten Funktionen abgeschätzt werden. Manche der Verfahren, die eine solche glatte Fläche ermitteln, erfordern zusätzlich topologische Information, wie z.B. die Kenntnis der Nachbardreiecke eines Dreiecks. Als Verfahren zur Bestimmung einer glatten Fläche werden z.B. [HAMA89] und [NIEL87] vorgeschlagen. Dort wird jede einzelne Dreiecksfläche als eine Konvexkombination der Form

$$\mathbf{x}(u_1, u_2, u_3) = \sum_{i=1}^{3} w_i(u_1, u_2, u_3)\, \mathbf{x}_i(u_1, u_2, u_3) \quad (10)$$

dargestellt. Hierbei ist (u_1, u_2, u_3) das Tripel baryzentrischer Koordinaten eines Punktes in einem Dreieck im Raum (ein Dreieck ist selbst das Parametergebiet für die zugehörige Dreiecksfläche), w_i, $i = 1, 2, 3$, sind Gewichtsfunktionen mit den Eigenschaften $w_i \geq 0$ und $\Sigma w_i = 1$. Die drei Flächenanteile $\mathbf{x}_i$ interpolieren je zu allen drei Randkurven und je zu den Normalen längs einer der drei Randkurven der endgültigen Dreiecksfläche. Die so erhaltene normalstetige Fläche wird wiederum durch Dreiecke

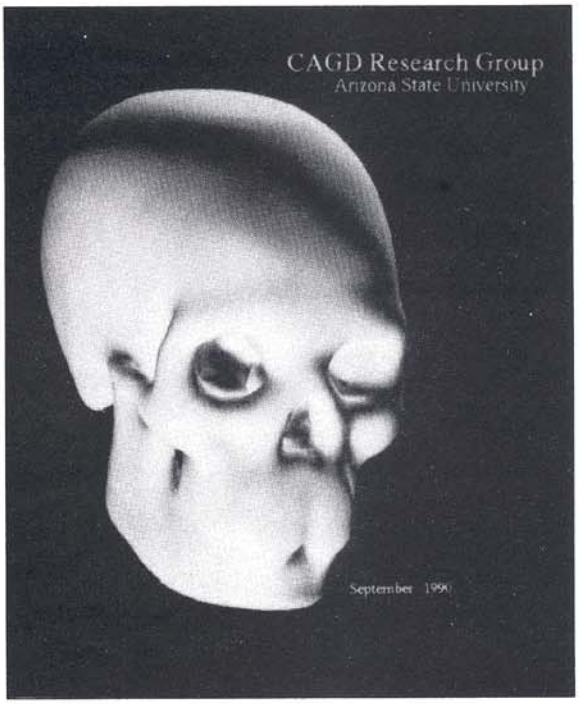


Abb. 14: Konturmethode für CT-Daten

approximiert, die dann auf dem Bildschirm Gouraud-schattiert ausgegeben werden. Es muß hier jedoch angemerkt werden, daß dieses sehr rechenintensive Verfahren nicht immer angebracht ist, da häufig schon die erste lineare Approximation einer Kontur (nach Schritt *6.3*) ausreichend hohe Auflösung besitzt. In *Abbildung 14* wird die Oberfläche eines menschlichen Schädels als Kontur einer trivariaten Funktion dargestellt, die als eine diskrete Datenmenge von 68•64•64 Dichtewerten einer Computertomographie in Form (2) gegeben ist. Die Daten sind zunächst nach *6.1* geglättet worden, Datenreduktion ist nicht erfolgt. Das Bild ist das Ergebnis nach Schritt *6.3* zur Approximation einer Kontur. Die lineare Approximation besteht aus etwa 30.000 Konturpunkten und 60.000 Konturdreiecken (Gouraud-schattiert), die nach einem Verfahren basierend auf [LORE87] bestimmt worden sind.

7. Schlußbemerkung

Alle in dieser Veröffentlichung besprochenen Verfahren sind vom Autor auf einem Silicon Graphics Rechner vom Typ 4D/220 GTX implementiert worden. Diese Forschung wurde durch den Vertrag zwischen dem U.S. Department of Energy und der Arizona State University unter Nummer DE-FG02-87ER25041 und durch Forschungsmittel der

NATO unter Nummer RG 0097/88 ermöglicht. Herzlicher Dank gilt allen Mitgliedern der Computer Aided Geometric Design - Forschungsgruppe im Computer Science Department der Arizona State University, hier speziell Prof.Dr. Gregory M. Nielson. Besonderer Dank gilt auch Herrn Wayne Woodland, der die Photographien der gezeigten Abbildungen angefertigt hat.

8. Literatur

[ALFE89] P. Alfeld: Scattered Data Interpolation in Three or More Variables. In: T. Lyche, L. Schumaker (eds.): Mathematical Methods in Computer Aided Geometric Design. In: Academic Press, (1989), 1-33.

[BARN85] R.E. Barnhill: Surfaces in Computer Aided Geometric Design: A Survey with New Results. In: Computer Aided Geometric Design, Vol.2, No.1-3, (1985), 1-17.

[BLOO90] B.B. Bloomquist: Contouring Trivariate Surfaces. Master Thesis, Arizona State University, Computer Science Department (1990).

[BOEH84] W. Boehm, G. Farin, J. Kahmann: A Survey of Curve and Surface Methods in CAGD. In: Computer Aided Geometric Design, Vol.1, (1984), 1-60.

[BROW82] D.H. Ballard, C.M. Brown: Computer Vision, Prentice Hall, (1982).

[CHOI88] B.K. Choi, H.Y. Shin, Y.I. Yoon, J.W. Lee: Triangultaion of Scattered Data in 3D Space. In: CAD, Vol.20, No.5, (1988), 239-248.

[DREB88] R.A. Drebin, L.Carpenter, P. Hanrahan: Volume Rendering. In: Computer Graphics, Vol.22, No.4, (1988), 65-74.

[FARI86] G. Farin: Triangular Berstein-Bézier Patches. In: Computer Aided Geometric Design, Vol.3, No.2, (1986), 83-127.

[FARI88] G. Farin: Curves and Surfaces for Computer Aided Geometric Design. Academic Press, (1988).

[FOLY90] T.A. Foley, D.A. Lane, G.M. Nielson: Towards Animating Ray-Traced Volume Visualization. Erscheint in: Visualization and Computer Animation Journal (1990).

[FRAN90] R. Franke, G.M. Nielson: Scattered Data Interpolation and Applications: A Tutorial and Survey. In: H. Hagen, D. Roller (eds.): Geometric Modelling: Methods and Their Applications. Springer (1990).

[FU87] K.S. Fu, R.C. Gonzalez, C.S.G. Lee: Robotics, McGraw-Hill (1987).

[FUCH89] H. Fuchs, M. Levoy, S.M. Pizer: Interactive Visualization of 3D Medical Data. In: Computer, Vol.22, No.8, (1989), 46-51.

[HAGE89] H. Hagen, H. Pottmann: Curvature Continuous Triangular Interpolants. In: T. Lyche, L.L. Schumaker (eds.): Mathematical Methods in Computer Aided Geometric Design, Academic Press, (1989), 373-384.

[HAMA90] B. Hamann, G. Farin, G.M. Nielson: A Parametric Triangular Patch Based on Degree Elevated Conics. SIAM-Konferenz Geometric Design, November 1989. In: G. Farin: Rational Curves and Surfaces. Erscheint (1990).

[HOSC89] J. Hoschek, D. Lasser: Grundlagen der Geometrischen Datenverarbeitung. B.G. Teubner, Stuttgart (1989).

[KAJI84] J. Kajiya, B. von Herzen: Ray-Tracing Volume Densities. In: Computer Graphics, Vol.18, No.3, (1984), 165-173.

[LEMÉ89] A.J.Y. Le Méhauté, Y. Lafranche: A Knot Removal Strategy for Scattered Data in $\mathbb{R}^2$. In: T. Lyche, L.L. Schumaker (eds.): Mathematical Methods in Computer Aided Geometric Design. In: Academic Press, (1989), 419-426.

[LEVO88] M. Levoy: Display of Surfaces from Volume Data. In: IEEE Computer Graphics and Applications, Vol.8, No.3, (1988), 29-37.

[LEVO90] M. Levoy: Volume Rendering. In: IEEE Computer Graphics and Applications, Vol.10, No.2, (1990), 33-40.

[LONG89] M.B. Long, K. Lyons, J.K. Lam: Aquisition and Representation of 2D and 3D Data from Turbulent Flows and Flames. In: Computer, Vol.22, No.8, (1989), 39-45.

[LORE87] W.E. Lorensen, H.E. Cline: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In: Computer Graphics, Vol.21, No.4, (1987), 163-169.

[NEY90] D.R: Ney, E.K. Fishman, D. Magid, R.A. Drebin: Volumetric Rendering. In: IEEE Computer Graphics and Applications, Vol.10, No.2, (1990), 24-32.

[NIEL87] G.M. Nielson: A Transfinite, Visually Continuous, Triangular Interpolant. In: G. Farin (ed.): Geometric Modeling: Algorithms and New Trends. In: SIAM Publications, Philadelphia, (1987), 235-246.

[NIEL90] G.M. Nielson, B. Hamann: Techniques for the Interactive Visualization of Volumetric Data. In: Proceedings of the IEEE Conference Visualization '90, San Francisco, Kalifornien, USA, Oktober 1990.

[PETE87] C. S. Petersen, B.R. Piper, A.J. Worsey: Adaptive Contouring of a Triavariate Interpolant. In: G. Farin (ed.): Geometric Modeling: Algorithms and New Trends. In: SIAM Publications, Philadelphia, (1987), 385-395.

[RATH88] W. Rath: Computerunterstützte Darstellung von Hyperflächen des $\mathbf{R}^4$ und deren Anwendungsmöglichkeiten im CAGD. In: CAD & Computergraphic, Jg.11, Nr.4, (1988), 111-117.

[SABE88] P. Sabella: A Rendering Algorithm for Visualizing 3D Scalar Fields. In: Computer Graphics, Vol.22, No.4, (1988), 51-55.

[TIED90] U. Tiede, K.H. Höhne, M. Bomans, A. Pommert, M. Riemer, G. Wiebecke: Surface Rendering. In: IEEE Computer Graphics and Applications, Vol.10, No.2, (1990), 41-53.

[ZUCK81] S.W. Zucker, R.A. Hummel: A Three-Dimensional Edge Operator. In IEEE Transactions on Pattern Recognition and Machine Intelligence, Vol. PAMI-3, No.3, (1981), 324-331.

Autor

Dipl.Inform. Bernd Hamann
Computer Science Department
Arizona State University
Tempe, AZ 85287-5406
U.S.A.