# A Volumetric Method for building Complex Models from Range Images

Brian Curless, Marc Levoy
(präsentiert an der SIGGRAPH ’96)

**Fachseminar Graphische Datenverarbeitung**, bei Prof. Dr. M. Gross
von **Oscar Chinellato**, 17. Juni 1997

## 1 Einführung

In den vergangenen Jahren hat die Laser-Scan-Technologie grosse Verbesserungen erfahren, die dazu führten, Daten schneller und genauer gewinnen zu können. Vor allem in den Bereichen der Medizin, des Reverse-Engineering und nicht zuletzt in der Film-Branche ist die Nachfrage nach solchen Geräten gestiegen.

Die Funktionsweise dieser Apparaturen ist bei allen Ausführungen sehr ähnlich : Ein Objekt, welches sich auf einer Unterlage befindet, wird mittels eines Distanzmessers abgetastet, und man gewinnt so eine endliche Anzahl fehlerbehafteter Messungen des Gegenstandes. Die Art und Weise, wie sich die Unterlage bezüglich des Scanners bewegt, unterscheidet sich zwar, und so natürlich auch die Art der Daten, die gewonnen wird, die Tatsache jedoch, dass die meisten Volumen sich nicht vollständig durch einen Scan abtasten lassen, bleibt.

Dies bedingt, dass man Methoden entwickelt, die es erlauben mehrere, solcher Messungen (Range-Scans) zu einer einzigen Darstellung der Objektoberfläche zusammenzuführen. Dabei sollten aber gewissen Kriterien berücksichtigt werden :

- **Vollständige Ausnützung der Scan-Daten** Durch mehrfaches Scannen kann eine gewisse Redundanz auftreten. Diese sollte man ausnutzen, um die Approximationsgenauigkeit des Objektes so hoch wie möglich zu halten.

- **Fähigkeit, Löcher in der Oberfläche zu stopfen** Sind trotz mehrfachem Scannen des Objektes noch Stellen auf dem Gegenstand vorhanden, die nicht abgetastet wurden, können diese durch gewisse Heuristiken beseitigt werden. Die nachbearbeiteten Modelle sollten „wasserdicht“ sein und trotzdem noch eine gewisse „Ästhetik“ besitzen (darauf wird in Kapitel 4 genauer eingegangen).

- **Inkrementelles Update** Es soll nach jedem Scan-Durchgang die Möglichkeit bestehen, die bis dahin kumulierten Daten zu visualisieren. Somit kann die für den nächsten Scan beste Orientierung gewählt werden. Weiterhin bietet sich so die Möglichkeit einer eventuellen Parallelisierung,

- **Darstellung von Unsicherheiten** Erhaltene Messwerte sind fehlerbehaftet. Dieser Nachteil kann jedoch klein gehalten werden, wenn bekannt ist, wie sich die Fehler manifestieren.

- **Robustheit** Ausreisser oder gewisse Einschränkungen in der Scan-Technologie sollten den Algorithmus nicht in die Knie zwingen. Er sollte in der Lage sein, Spezialfälle wie z.B. selbstüberlappende Bereiche und Löcher in Oberflächen „intelligent“ zu behandeln.

- **Zeit- und Speichereffizienz** Komplexe Objekte bedürfen vieler Scans und somit fallen viele Daten an. Diese sollten in einer speicherfreundlichen Art und Weise verwaltet werden, welche die Laufzeit klein hält.

Doch werfen wir nun einen Blick auf ...

## 2 Verwandte Arbeiten und Verfahren

Da schon seit geraumer Zeit auf dem Gebiet der Modellierung von Objekten aus Range-Daten geforscht wird, ist es nicht verwunderlich, dass eine Vielzahl von Methoden entwickelt wurde, um diese Aufgabe zu lösen.

Im Wesentlichen lassen sich die Algorithmen in zwei Kategorien aufteilen : Die einen rekonstruieren die Oberfläche aus unorganisierten Punkten, die anderen rekonstruieren die Oberfläche durch Ausnützen der Struktur, die in den Daten steckt. Diese zwei Richtungen unterscheiden sich wiederum dadurch, dass sie die Oberfläche durch parametrisierte Flächen resp. durch eine implizite Funktion beschreiben.

Jene, die auf unorganisierten Punkten beruhen, besitzen den Vorteil, dass sie keine Annahme über die Struktur des Objektes treffen müssen. Somit sind sie fähig, Oberflächen zu rekonstruieren, von denen man keine weitere Informationen besitzt, was sogar dazu führen kann, dass diese Art von Algorithmus in einigen Fällen die einzige Lösung darstellt.

Ein Beispiel der Rekonstruktion parametrisierter Flächen bietet sich in der Delaunay-Triangulierung (Boissonnat, 1984). Methoden die auf impliziten Funktionen beruhen, erzeugen z.B. vorzeichenbehaftete Distanzfunktionen und extrahieren nach gewissen Messungen eine Isofläche (Hoppe et al., 1992).

Zwar sind die Algorithmen, die auf unorganisierten Punkten funktionieren, praktisch überall anwendbar, sie machen jedoch nicht Gebrauch von wichtigen Informationen, wie z.B. Oberflächennormalen und Vertrauensintervallen. Das hat zur Folge, dass diese in Bereichen, wo die Oberfläche glatt ist, und sich anständig verhält gut, in anderen Bereichen aber, wo grosse Steigungen und Krümmungen oder systematische Messfehler auftreten, schlecht abschneiden.

Betrachtet man die Algorithmen, die die Struktur der Daten ausnutzen, so kristallisieren sich auch hier zwei Grundmethoden heraus. Die eine berechnet parametrisierte Flächen, wie z.B. Polygonnetze die verschmolzen werden (Soucy und Laurendau, 1995), Polygonnetze, welche affin transformiert und geclippt werden, mit dem Ziel, den quadratischen Fehler zu minimieren (Turk und Levoy, 1994). Diese Verfahren verhalten sich zwar besser als die, die auf unorganisierten Punkten arbeiten, versagen aber kläglich in Bereichen mit grosser Krümmung.

Die Restlichen definieren die Oberfläche durch eine implizite Funktion. Diese Methoden unterteilen sich erneut in zwei Gruppen : Jene, die die Voxels durch diskrete Zustände klassifizieren oder jene, welche die Voxels als Samples einer kontinuierlichen Funktion betrachten.

Zu den „diskreten“ Vertretern ist zu sagen, dass sie nur in der Theorie untersucht, in der Praxis jedoch nie angewandt wurden. Des weiteren ist es nicht möglich, ohne eine kontinuierliche Funktion das Fehlerintervall abzuschätzen.

Die letzte Gattung, in die auch das Verfahren von Curless und Levoy fällt, erzeugt eine implizite Funktion resp. diskretisiert diese durch Samples, die in Voxels abgespeichert werden. Es sind zwar einige andere Versuche getätigt worden, z.B. von Grosso, Succi, Hilton und anderen (1988, 1990, 1996); ein Hauptproblem jedoch war, dass sie keine Aussagen über die Richtung des Fehlers, über die Effizienz und über die „Lochfüllfähigkeit“ ihrer Methoden machten.

## 3 Volumenbasierte Rekonstruktion

Wie oben erwähnt, ist der hier besprochene Algorithmus bestrebt, eine implizite Funktion zur Darstellung der Oberfläche zu berechnen.

Als erstes legt man um das zu scannende Objekt einen virtuellen Würfel, der in kleine, gleichgrosse Voxels aufgeteilt wird. In den Ecken jedes Voxels steht der Wert einer Funktion. Mit Hilfe dieser Samples ist es schlussendlich möglich, eine Isofläche zu extrahieren. Doch dazu später.

Diese Funktion, die hier diskretisiert wird, ist eine gewichtete, vorzeichenbehaftete Distanzfunktion, welche sich aus den Distanzen, von Punkten aus, die in der Nähe der geschätzten Oberfläche liegen, bis hin zu dieser selbst, zusammensetzt. Die Niveaufläche, auf der die Funktion den Wert Null annimmt, stellt die optimale „Kurve“ im Sinne der kleinsten, gewichteten Abstandsquadraten dar. Die Herleitung der Verknüpfungsvorschrift resp. der Beweis der Optimalität soll nun hier dargestellt werden.

Um die ohnehin schon schwierige Herleitung etwas zu vereinfachen, werden einige Annahmen getroffen, welche am Schluss wieder fallen gelassen werden. Des weiteren wird die Herleitung für den zweidimensionalen Fall gezeigt, und erst am Schluss wird die dritte Dimension hinzugenommen.

Die Hauptannahmen seien die folgenden : Jeder Scan liefert eine Distanzfunktion $d_i$ für eine bestimmte Scanrichtung $v_i$ , und nicht etwa einen Datensatz, mit Hilfe dessen man diese diskretisieren könnte. Mit jeder Distanzfunktion $d_i$ ist eine Gewichtsfunktion $w_i$ verbunden.

Das Problem, dass es nun zu lösen gilt, ist folgendes : Durch n-maliges Scannen eines Objektes aus der jeweiligen Richtung $v_i$ erhält man jeweils eine Distanzfunktion $d_i$ (i=1,2,...,n) im Koordinatensystem $[v_i,w_i]$, wobei die Distanz in Richtung $v_i$ gemessen wird. Gesucht ist jene Funktion *y=f(x)* im Koordinatensystem [x,y], welche das gewichtete Abstandsquadrat minimiert. Geht man das ganze von der mathematischen Seite an, so kommt man auf die folgende Formulierung :

Die Scanrichtung $v_i$ (2. Hauptachse), resp. $u_i$ (1. Hauptachse) sind wie folgt definiert :

$$v_i = \begin{bmatrix} v_{x,i} \\ v_{y,i} \end{bmatrix}, u_i = \begin{bmatrix} v_{y,i} \\ -v_{x,i} \end{bmatrix} \qquad \|v_i\| = 1, \|u_i\| = 1$$

Um Punkte aus dem [x,y]-Koordinatensystem in das $[u_i,v_i]$-Koordinatensystem zu überführen, kann man folgende Abbildung anwenden :

$$\begin{bmatrix} v_{y,i} & -v_{x,i} \\ v_{x,i} & v_{y,i} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$$

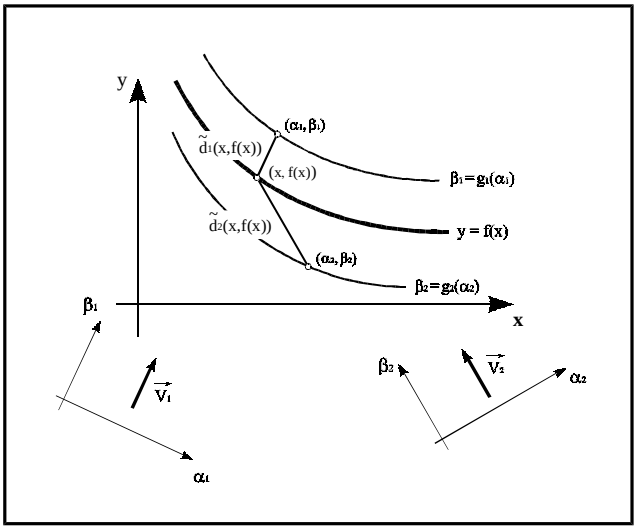


**Bild 1.** Berechnen der optimalen Funktion *y=f(x)*

Weiterhin seien die Distanzfunktionen $d_i(\alpha,\beta)$ zu den „gescannten“ Kurven, sowie die Gewichtsfunktionen $w_i(\alpha,\beta)$ im jeweiligen Koordinatensystem $[u_i,v_i]$ bekannt. Die Funktionen $f_i$ stellen die Funktion *y=f(x)* im Koordinatensystem $[u_i,v_i]$ dar. Zu $d_i(\alpha,\beta)$ ist zu sagen, dass es die Distanz vom Punkt $[\alpha,\beta]$ zu der Kurve $g_i(\alpha)$ in Richtung $v_i$ angibt (siehe Bild 1). Das Ziel ist es, den folgenden Ausdruck zu minimieren :

$$D = \sum_{i=1}^{n} \int_{-\infty}^{\infty} d_i(\alpha, f_i(\alpha))^2 \, w_i(\alpha, f_i(\alpha)) d\alpha$$

Um die Funktion *y=f(x)* ins Spiel zu bringen und um alle Integrale auf dem selben Bereich resp. im selben Koordinatensystem auswerten zu können, werden die Variablen gemäss obiger Abbildung transformiert. Dies führt zu neuen Distanz- und Gewichtsfunktionen und zu einem neuen Integrationsbereich ...

$$\tilde{d}_i(x,y) = d_i(x \cdot v_{y,i} - y \cdot v_{x,i}, x \cdot v_{x,i} + y \cdot v_{y,i})$$
$$\tilde{w}_i(x,y) = w_i(x \cdot v_{y,i} - y \cdot v_{x,i}, x \cdot v_{x,i} + y \cdot v_{y,i})$$

$$\Rightarrow \quad D(y) = \sum_{i=1}^{n} \int_{-\infty}^{\infty} \tilde{d}_i(x,y)^2 \, \tilde{w}_i(x,y) \cdot \vec{v}_i \circ \left[ -\frac{\partial y}{\partial x}, 1 \right] dx$$

Nun ist es Zeit, vom zweidimensionalen Fall in den Dreidimensionalen überzugehen. Die daraus resultierende Formel lautet :

$$D(z) = \iint \sum_{i=1}^{n} \tilde{d}_i(x,y,z)^2 \, \tilde{w}_i(x,y,z) \cdot \vec{v}_i \circ \left[ -\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y}, 1 \right] dxdy \quad \textbf{(1)}$$

Jene Funktion *z=f(x,y),* die den Ausdruck (1) minimiert, beschreibt also die Fläche, die im Sinne der gewichteten Abstandsquadrate optimal liegt. Um solche Minima zu suchen, bedient man sich der Variationsrechnung. Mit Hilfe dieses Werkzeuges lässt sich die folgende Klasse von Problemen lösen :

$$I = \iiint \sum_{i=1}^{n} h_i(x, y, z, \frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}) dxdydz \quad \textbf{(2)}$$

Es sei das Integral (2) zu minimeren, d.h. gesucht ist die Funktion *z=f(x,y),* welche den Ausdruck minimiert. Aus dem Gebiet der Variationsrechnung ist die Gleichung von Euler-Lagrange bekannt, welche folgende Aussage über die optimale Funktion macht :

$$\sum_{i=1}^{n} \left[ \frac{\partial h_i}{\partial z} - \frac{\partial}{\partial x} \frac{\partial h_i}{\partial \left( \frac{\partial z}{\partial x} \right)} - \frac{\partial}{\partial y} \frac{\partial h_i}{\partial \left( \frac{\partial z}{\partial y} \right)} \right] = 0 \quad \textbf{(3)}$$

Vergleicht man Die Gleichungen (1) und (2) so liegt es nahe, einige neue Definitionen einzuführen, um Euler-Lagrange anwenden zu können :

$$h_i(x, y, z, \frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}) = \sum_{i=1}^{n} e_i(x,y,z) \cdot \vec{v}_i \circ \left[ -\frac{\partial z}{\partial x}, -\frac{\partial z}{\partial y}, 1 \right]$$

$$e_i(x,y,z) = \tilde{d}_i(x,y,z(x,y))^2 \, \tilde{w}_i(x,y,z(x,y))$$

Wendet man nun Formel (3) auf die neuen Definitionen an, so resultiert nach ziemlich langer und sehr komplizierter Rechnung (Beweis in [2]) :

$$\Rightarrow \quad \sum_{i=1}^{n} v_i \cdot \nabla e_i(x,y,z) = \sum_{i=1}^{n} v_i \cdot \nabla (\tilde{d}_i(x,y,z)^2 \, \tilde{w}_i(x,y,z)) = 0 \quad \textbf{(4)}$$

Der Ausdruck (4) stellt nichts anderes als die Richtungsableitung von $e_i(x,y,z)$ in Richtung $v_i$ dar. Berechnet man diese Ableitung so ergibt sich folgendes :

$$\sum_{i=1}^{n} v_i \cdot \nabla e_i(x,y,z) = \sum_{i=1}^{n} \left[ 2\|v_i\| \cdot \tilde{d}_i(x,y,z) \tilde{w}_i(x,y,z) \frac{\partial}{\partial vi} \tilde{d}_i(x,y,z) + \|v_i\| \cdot \tilde{d}_i(x,y,z)^2 \frac{\partial}{\partial vi} \tilde{w}_i(x,y,z) \right]$$

Wenn man die Differentialfaktoren besser betrachtet, so sieht man :

$$\frac{\partial}{\partial vi}\tilde{d}_i(x,y,z)=1, \qquad \frac{\partial}{\partial vi}\tilde{w}_i(x,y,z)=0$$

Der erste Ausdruck beschreibt die Richtungsableitung der vorzeichenbehafteten Distanzfunktion. Da die Distanz linear ist, ergibt sich eben diese Ableitung. Die Gewichtsfunktion wird jeweils absichtlich so gewählt, dass sie nur entlang der $u_i$-Achse ändert, d.h. der Funktionswert bleibt in $v_i$-Richtung konstant. Deshalb verschwindet die Ableitung.

Nach all diesen Umformungen lässt sich also aussagen, dass für alle Punkte auf der optimalen Fläche, d.h. *z=f(x,y)* folgender Zusammenhang gilt :

$$\Rightarrow \qquad F(x,y,z)=\sum_{i=1}^{n} w_i(x,y,z)\,d_i(x,y,z)=0 \qquad \textbf{(5)}$$

Die Relation (6) beschreibt die am Anfang des Kapitels erwähnte implizite Funktion. Alle Punkte auf der optimalen Oberfläche ergeben den Nullwert. Alle anderen Punkte ergeben einen davon verschiedenen Wert.

Aus dieser Formel lässt sich auch noch eine Vorschrift aufstellen, die es dann ermöglicht, das Verfahren inkrementell zu benützen ...

$$D_n(x)=\frac{\sum_{i=1}^{n} wi(x)di(x)}{\sum_{i=1}^{n} wi(x)}, W_n(x)=\sum_{i=1}^{n} w_i(x)$$

$$\Rightarrow \quad \boxed{D_{n+1}(x)=\frac{W_n(x)D_n(x)+w_{n+1}(x)d_{n+1}(x)}{W_n(x)+w_{n+1}(x)}, \quad W_{n+1}(x)=W_n(x)+w_{n+1}(x) \quad \textbf{(6)}}$$

Diese Vorschriften werden benötigt, um das Volumenupdate auszuführen. Sie berechnen für jeden VoxelEckPunkt den Funktionswert gemäss (5), basierend auf dem vorhergehenden Wert.

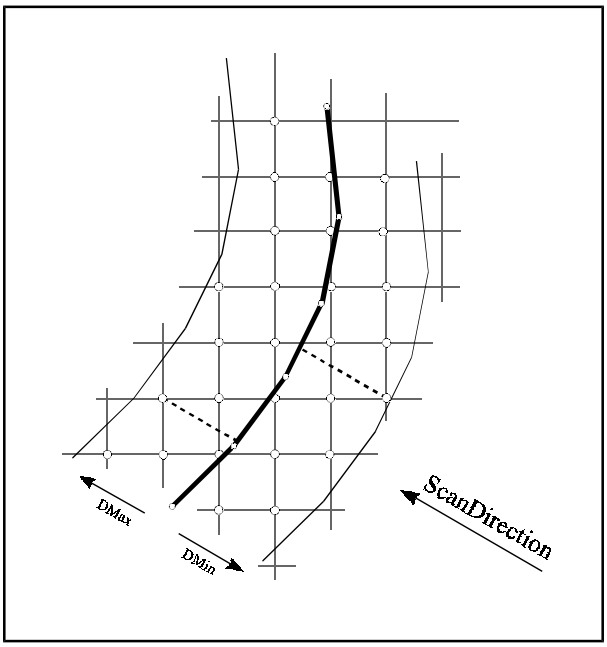


**Bild 2.** Distanzfunktion durch Interpolation Update-Bereich zw. [DMin..DMax]

An dieser Stelle ist es Zeit, die restlichen Annahmen fallen zu lassen. Da man beim Scannen anstelle einer Distanzfunktion nur eine endliche Anzahl Punkte einer ungefähren Oberfläche erhält, bildet man sich eine Distanzfunktion, indem man aus den gewonnenen Daten ein Polygonmesh konstruiert. Ist nun ein beliebiger Punkt gegeben, so kann man seine Distanz zur fehlerbehafteten Oberfläche mittels bilinearer Interpolation berechnen.

Zumal der Scanner eine gewisse Genauigkeit einzuhalten vermag, schränkt sich der Bereich, der Punkte resp. VoxelEckPunkte enthält, die in der Nähe der Tatsächlichen Oberfläche liegen, ein. Üblicherweise betrachtet man nur jene Voxels die maximal um die Hälfte der Fehlerintervalllänge (DMin resp. DMax) von der gescannten Oberfläche entfernt sind. (Bild 2)

Nun zu den Gewichten. Wird während des Scanvorgangs festgestellt, dass die erhaltenen Werte grosse Fehler beinhalten könnten, so werden diese mit einem kleinen Gewicht versehen. Fehler treten meistens dort auf, wo der Gradient stark ändert, der Schnitt des Scanner-Strahls mit dem Objekt schleifend ist usw. . Die Gewichte werden analog zu den Distanzen bilinear interpoliert.

Mit Hilfe der oben hergeleiteten Bedingung (5) und der konstruierten Distanz- resp. Gewichtsfunktionen lässt sich jetzt ein Algorithmus zur Rekonstruktion der Objektoberfläche aufschreiben :

0 Setze VoxelEckGewichte = 0 und VoxelEckDistanzen = 0

1 Scanne Objekt aus Richtung $v_i \rightarrow$ Punktemenge P, Konvertiere Menge P in Dreiecksmesh D

2 Betrachte alle Voxels in der Nähe des Meshes :

- Bestimme Distanz des VoxelEckPunktes zum Mesh (bilin. Interpol.)
- Bestimme Gewicht des VoxelEckPunktes (bilin. Interpol.)
- Updaten des vorhandenen Distanzwertes mittels Formeln (6)

3 Isoflächenextraktion durch MarchingCubes

4 Ist Modell zu grob, so wiederhole mit neuer Richtung $v_{i+1}$ von Schritt 1 an

## 4 Füllen von Löchern

Der Algorithmus, der im vorhergehenden Kapitel besprochen wurde, hat zum Ziel, gescannte Bereiche des Gegenstandes zu einer Oberfläche zu rekombinieren. Bereiche, die nicht abgetastet wurden, erscheinen demzufolge als Löcher. Obwohl dieses Resultat eine genaue Umsetzung der gewonnenen Daten darstellt, ist es wünschenswert, schlussendlich ein „wasserdichtes“, d.h. ohne Löcher behaftetes, Modell zu generieren. Für einige Anwendungen ist dies eine nötige Vorbedingung, wie z.B. für die Stereolithographie.
Eine Möglichkeit besteht darin, die Löcher in den Meshes, die aus den einzelnen Range-Scans gewonnen werden, zu stopfen. Dieser Ansatz erweist sich als günstig, wenn die fraglichen Bereiche fast eben sind. Da Löcher jedoch meist durch Unebenheiten entstehen, versagen diese Methoden.

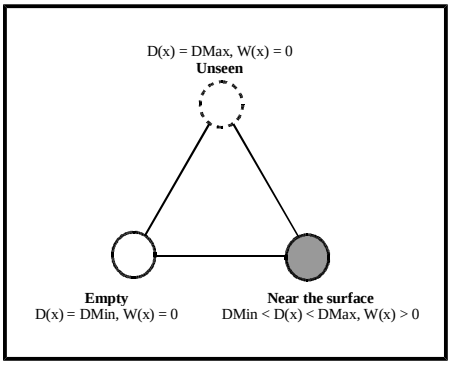


**Bild 3.** Zustandskodierung der Voxels

In dem hier vorgestellten Verfahren hingegen operiert man auf dem Volumen, welches ohnehin mehr Information enthält als ein Mesh. Die Hauptidee ist die folgende : Alle VoxelEckPunkte befinden sich in einem von drei möglichen Zuständen, nämlich „Unseen“, „Empty“ oder „Near the surface“. Löcher in der Oberfläche können sich ausschliesslich dort befinden, wo Übergange zwischen „Unseen“ und „Empty“ auftreten. Und genau diese Bereiche werden mit Polygonmeshes gestopft. In der tatsächlichen Implementation des Algorithmus werden die Zustände gemäss Bild 3 kodiert.

Um die Voxel, die im „Empty“-Zustand sein sollten zu finden, benützt man ein Verfahren namens „Space-Carving“. Dieses Verfahren verfolgt, ausgehend vom Durchstosspunkt des Scan-Strahles mit dem Gegenstand, den Strahl in Richtung $v_i$ zurück bis hin zum Sensor, und alle Voxels die unterwegs getroffen werden, gelangen in den Zustand „Empty“. Wegen dieser Klassifizierung ergibt sich eine Erweiterung des in Kapitel 3 aufgeführten Algorithmus :

0 Setze VoxelEckGewichte = 0, VoxelEckDistanzen = 0 und **Voxel = „Unseen“**

1 Scanne Objekt aus Richtung $v_i \rightarrow$ Punktemenge P, Konvertiere Menge P in Dreiecksmesh

2 Betrachte alle Voxels in der Nähe des Meshes :

- Bestimme Distanz des VoxelEckPunktes zum Mesh (bilin. Interpol.)
- Bestimme Gewicht des VoxelEckPunktes (bilin. Interpol.)
- Updaten des vorhandenen Distanzwertes mittels Formeln (6)
- **Ändern des Voxelzustandes in „Near the Surface“**
- **Verfolge Scan-Strahl zurück und markiere durchlaufeneVoxels mit „Empty“ $\rightarrow$ Space-Carving**

3 Isoflächenextraktion durch MarchingCubes, **Extraktion der Lochfüller mittels Betrachtung der „Unseen“-„Empty“-Übergänge**

4 Ist Modell zu grob, so wiederhole mit neuer Richtung $v_{i+1}$ von Schritt 1 an

Der Vorteil dieser Repräsentation ist der, dass nach wie vor dieselbe Isoflächenexraktionsmethode verwendet werden kann, mit dem Zusatz, dass „Lochfüller“ erkannt werden (siehe Bild 3). Somit ist man in der Lage, alle Löcher zu stopfen und diese „Lochfüller“ von der regulären Oberfläche zu unterscheiden.

Dies erweist sich insofern als vorteilhaft, als dass aufgrund von Aliasing-Artefakten bei den unstetigen Übergängen zwischen „Unseen“ und „Empty“ ein Filtering (Analytic Filtering, Supersampling and Averaging down usw.) an den Rändern jener Flächen vorgenommen werden kann. Diese selektive Filterung erlaubt es, jene Teile der Oberfläche, die genau bekannt sind, unberührt zu lassen.

Noch einige Bemerkungen zum Space-Carving. Space-Carving erlaubt es, Information über den freien Raum zu gewinnen, die es dann ermöglicht, auf intelligente Art Löcher zu finden und zu füllen. Doch unsere Methode ist nur in der Lage, von einem Objekt zurück zu „carven“, d.h. es braucht für ein Carving immer einen Schnittpunkt mit dem Gegenstand. Es gibt jedoch Fälle, in denen noch mehr Carving erwünscht wäre, denkt man z.B. an eine Röhre, durch deren Zentrum man sehen würde.

Um auch in solchen Fällen einen „Schnittpunkt“ mit dem Scan-Strahl zu erzeugen, positioniert man eine sogenannte Back-Drop-Plane hinter das Objekt (ausserhalb des Volumens), welche bei Löchern, die das Objekt vollständig durchdringen, trotzdem einen Schnittpunkt liefert, der dann auch dort Carving ermöglicht.

## 5 Implementation

**Hardware** Die Beispiele, die in Kapitel 6 folgen werden, wurden mit Hilfe eines 3030 Cyberware-Laser-Scanners und einer Cyberware MS Motion-Platform erzeugt. Die Plattform wurde relativ zum Laser verschoben, wobei stets die Schnittlinie mit der Scanebene registriert wurde. Um die Messqualität zu erhöhen, wurde das Verfahren der „Spacetime analysis“ benutzt [3]. Die Vorteile dieses Verfahrens sind weniger verrauschte Range-Daten, grössere Stabilität sogar bei Änderungen im Reflexionsverhalten der Oberfläche und weniger Ausreisser in der Nähe von Objektunstetigkeiten.

Ein weiterer Vorteil besteht darin, dass die Messunsicherheit bei Anwendung der „Spacetime analysis“-Methode in Richtung des Sensors zeigt. Dies hat für die mathematische Herleitung der Formeln in Kapitel 3 enorme Wichtigkeit, für eine genaue Erklärung schaue man in [2] nach.

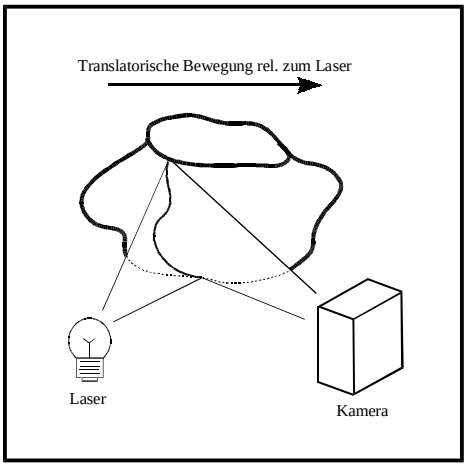


**Bild 4.** Schema der Scan-Apparatur

**Software** Um detaillierte, komplexe Modelle zu kreieren, benötigt man ungeheuer viel Daten, die in ein hochauflösendes Voxel-Grid eingefügt werden müssen. Es ist deshalb nicht ungewöhnlich, von einem komplizierten Objekt bis zu 70 Scans à je 10 Millionen Punkte aufzunehmen und diese in ein Voxel-Grid mit 150 Millionen Voxels abzubilden. Daher ist es offensichtlich nötig, auf Zeit- und Speichereffizienz zu achten.

Ein erster Schritt in diese Richtung besteht darin, auf das Voxel-Grid ein Run-Length-Encoding (RLE) anzuwenden. Es werden Läufe der Art „Empty“, „Unseen“ und „Varying“ generiert, wobei die variierenden Daten in unkodierter Form abgespeichert werden. Die daraus resultierende Struktur weist nach dem „Encoding“ noch typischerweise einen Zehntel bis sogar einen Zwanzigstel der ursprünglichen Grösse auf. Tatsächlich ist es sogar so, dass der Speicher, der durch ein vollständiges Voxel-Grid belegt wird, weniger ist als der, der durch das schlussendlich extrahierte Mesh verbraucht wird.

Das Volumenupdate kann im Grunde genommen als inverses Volumerendering betrachtet werden. Anstatt von einem Volumen zu lesen und auf eine Ebene zu schreiben, liest man von einer Fläche (Range-Image, Mesh) und schreibt in ein Volumen (Voxel-Grid). Deshalb ist es möglich, eine aus dem Volume-Rendering bewährte Technik zu verwenden; um hohe Performanz zu erreichen, durchquert man das Volumen und das Range-Image in Scanline-Order. Üblicherweise sind die Scanlines des Volumens und jene des Range-Scans windschief zueinander, was zu einer Einbusse in Punkto Effizienz führen würde. Darum wird auf das Range-Image ein Resampling angewandt, mit dem Zweck, die Scanline-Richtungen zu harmonisieren. Dieses Resamplen ist im Prinzip eine Art Projektion der Messpunkte des Range-Image auf eine Ebene, die parallel zur Hauptachse des Würfels verläuft (siehe Bild 4).

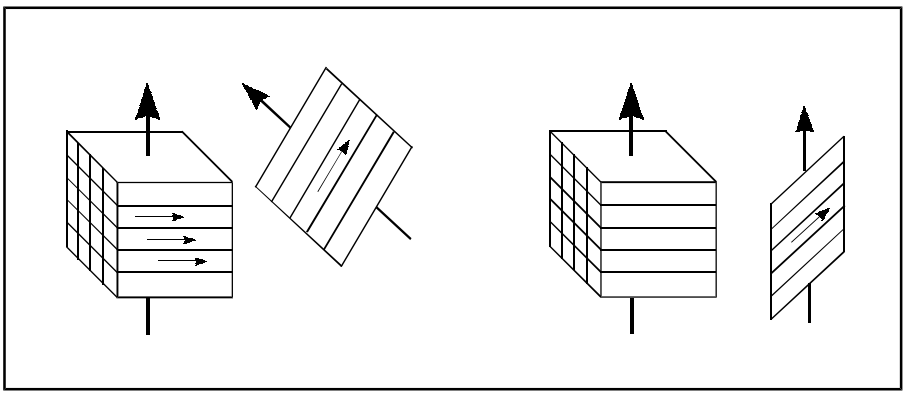

**Bild 5.** Resampling des Range-Image

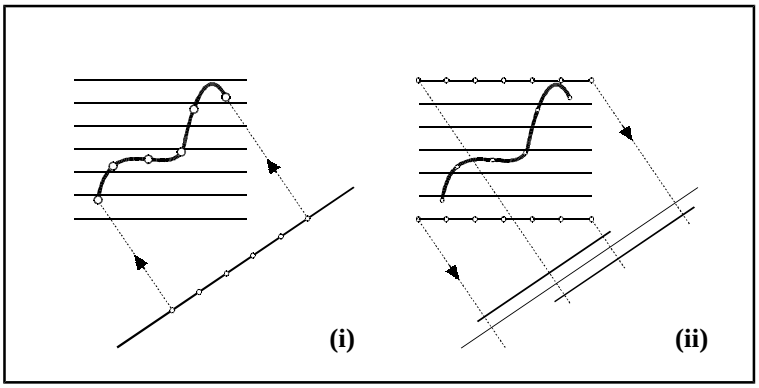


**Bild 6.** Mapping der Voxel auf das Range-Image

Da die Struktur, wie bereits erwähnt, RLE-kodiert ist, wäre es zu aufwendig, von der Range-Image Daten zu lesen und diese ins „richtige“ Voxel zu stecken (Mappen des Range-Image auf das Voxel-Grid, Bild 6.i). Den Aufwand, den man mit Dekodieren, richtiges Voxel in der RLE Struktur suchen, reinschreiben und erneut kodieren, betreiben müsste, wäre prohibitiv gross. Der Flaschenhals dieser Methode besteht in der Suche in der RLE-Struktur. In umgekehrter Richtung hingegen ist es um einiges einfacher. Man „mappt“ eine Voxel-Scanline auf die neue Range-Image, schaut wo diese zu liegen kommt und überprüft ob der Voxel überhaupt betroffen ist. Somit kann die RLE-Struktur linear durchlaufen werden und nur an den nötigen stellen ist ein Aufsplitten derselben nötig. (Bild 6.ii)

Da man a priori nicht weiss, welche Voxels in der Nähe der Objektoberfläche liegen, wäre es gut, eine Datenstruktur mitzuführen, die eine ungefähre Aussage darüber machen könnte. Im Extremfall könnte man so ganze Voxel-Scanlines beim Update überspringen. Diese Struktur wird hier als binärer Baum implementiert, in dem die maximalen resp. minimalen Tiefen benachbarter Scanlines eines Range-Image gespeichert werden. Beim Durchlaufen einer Voxel-Scanline kann so entschieden werden, welches Intervall derselben, wenn überhaupt, durch die Daten beeinflusst wird. Mit Hilfe dieses binären Baumes wird der Algorithmus etwa um den Faktor 15 beschleunigt.

Zum Schluss soll aus dem Volumen jene Isofläche extrahiert werden, die den Schwellwert Null hat. Da, wie in Kapitel 3, Gleichung (6), bewiesen wurde, dass nur eine optimal Fläche den Funktionswert Null hat, wird also genau diese extrahiert. Für diesen Schritt verwendet man den Marching-Cubes Algorithmus mit einer Look-Up-table, um die zweideutigen Fälle zu eliminieren. Um auch hier nochmals auf Effizienz zu schauen, operiert man nur auf Voxels, die in der RLE mit „Varying“ markiert sind, oder jene, die Übergänge zwischen „Empty“ und „Unseen“ aufweisen.

## 6 Resultate, Beispiele

Die hier aufgeführten Beispiele wurden so ausgewählt, dass man anhand dieser, die Robustheit des Algorithmus, seine „Lochfüllfähigkeit“ und seine Detailerhaltunig demonstrieren kann.

Als erstes soll die Robustheit aufgezeigt werden. Zu diesem Zweck wurde ein sehr kritisches Objekt ausgesucht. Der Spiralbohrer stellt insofern eine Herausforderung dar, als dass er, egal wo ein Querschnitt gewählt wird, immer scharfe Kanten aufweist. Zum Vergleich wurde die Methode von [4] angewandt. Gewisse Eigenschaften derselben lassen den Algorithmus völlig versagen, wohingegen die volumenbasierte Rekonstruktion ziemlich erfolgreich abschneidet.

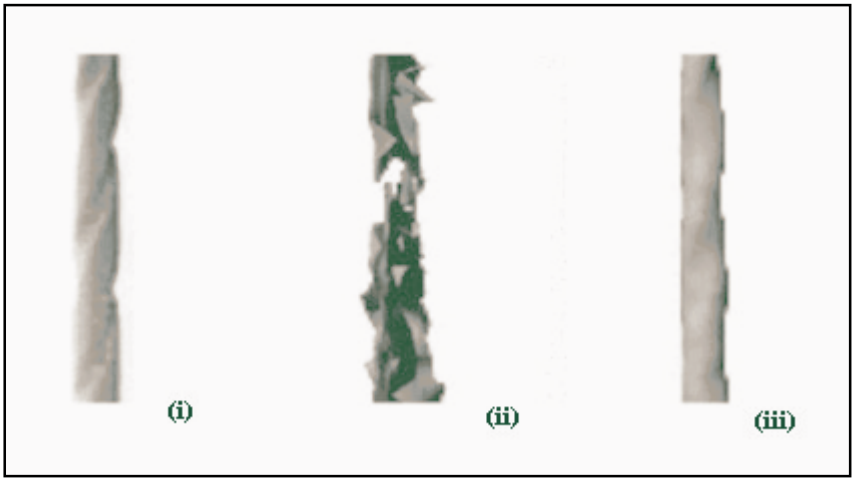


**Bild 7.** (i) : Fotografie des Spiralbohrers, (ii) : Rekonstruktion des Gegenstandes mit [4], (iii) : Wiederaufbau mittels volumenbasierter Rekonstruktion

Dass dieses Verfahren so gut Löcher zu füllen vermag, verdankt es dem Space-Carving. Wie auf dem nächsten Bild zu sehen, erweist sich der Einsatz einer Back-Drop-Plane als sehr vorteilhaft. Trotzdem ist es nicht ratsam immer eine solche Plane einzusetzen, weil die Range-Images dann dichter werden und so die Laufzeit zunimmt.

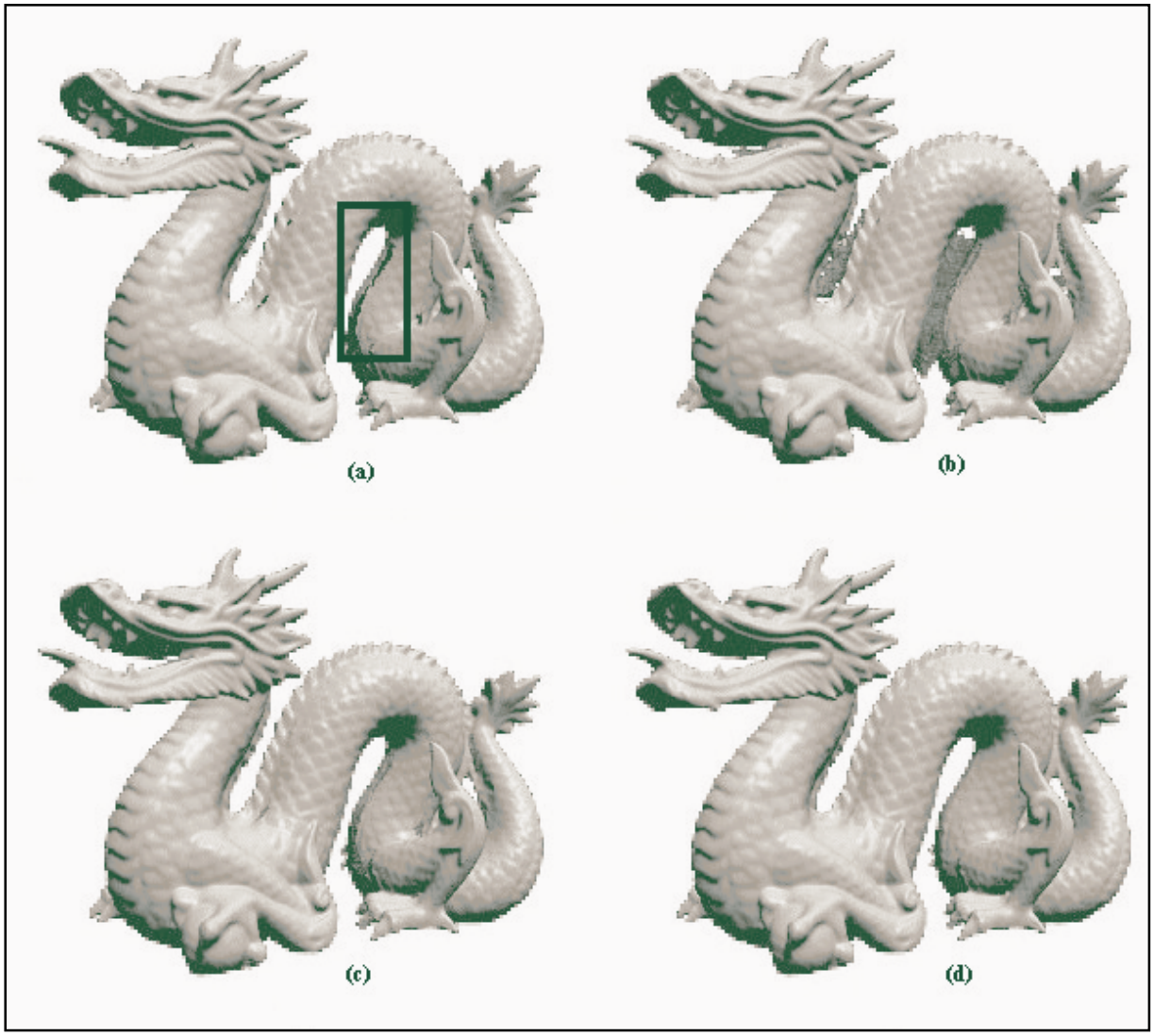


**Bild 8.** (a) Rekonstruktion mit 61 Range-Scans. Der Umrahmte bereich ist in Bild 9 vergrössert dargestellt. Kein Space-Carving, kein Löcherfüllen. Löcher sind die dunklen Stellen im Rahmen.
(b) Identische Daten wie in (a), diesmal mit Space-Carving und Hole-Filling.
(c) 10 Range-Images mehr als in (b) und eine Back-Drop-Plane wurde gesetzt.
(d) Nachfilterung von (c). Da nur bei den Löchern gefiltert wird, bleibt die Detailstufe im Rest des Modells erhalten.

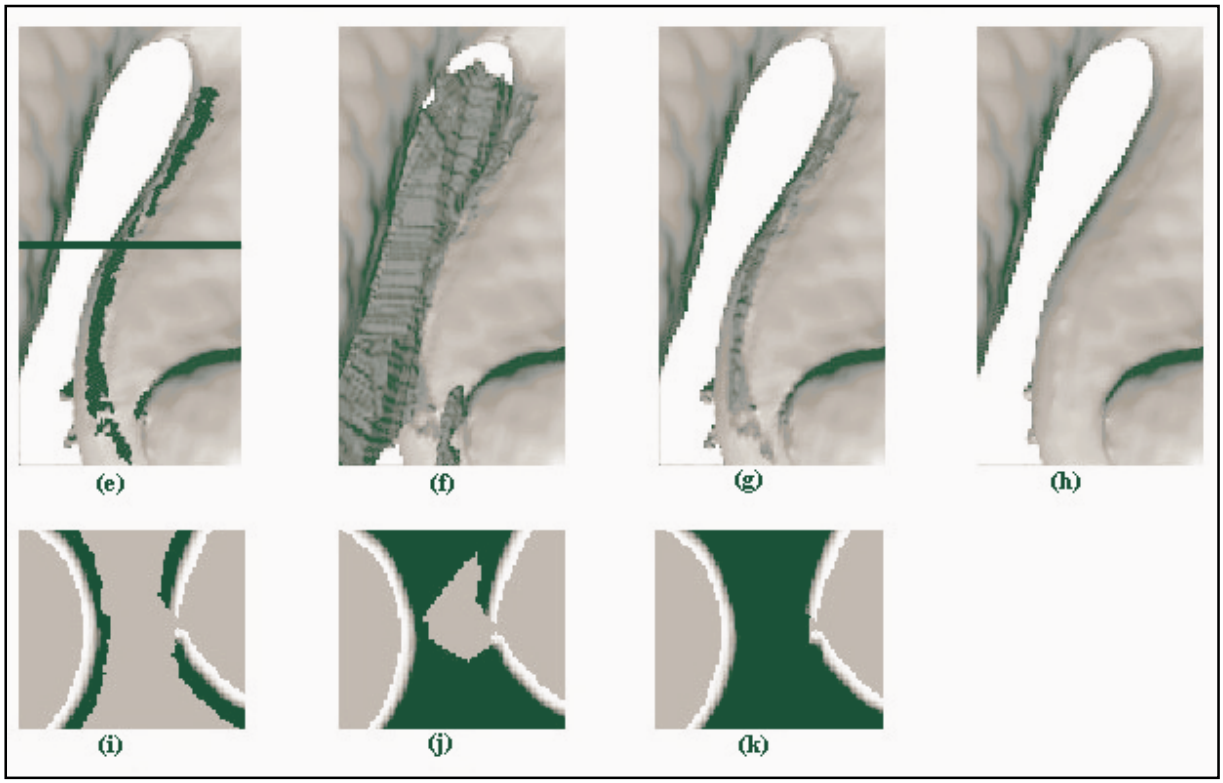


**Bild 9.** (e) Vergrösserung des in (a) markierten Bereiches
(f) Vergrösserung von (b), Space-Carving und Hole-Filling wurde zwar eingesetzt, trotzdem ist noch zuviel „Material“ vorhanden.
(g) Unter Zuhilfenahme einer Back-Drop-Plane werden die leeren Bereiche auch als solche erkannt.
(h) Nach einem Filtering werden die Kanten, die zwischen Lochfüllern und effektivem Objekt entstehen, weitgehend geglättet.
(i) Ein Querschnitt des Stückes in (e), Querschnittsebene in (e) markiert.
(j) Obwohl Carving angewandt wurde, ist noch überschüssiges Material zu erkennen (siehe auch (f))
(k) Nach dem Einsatz der Back-Drop-Plane ist das Objekt praktisch einwandfrei Rekonstruiert.

Die Voxelgrössen belaufen sich auf 250-300µm; typische Auflösungen von Scanner liegen im Bereich von 100µm. Wie man später nachrechnen konnte, liegt der gerichtete, mittlere quadratische Fehler der Isofläche zu den Scans bei etwa 100µm, und da dies gerade ungefähr der Scanner-Resolution entspricht, deutet dies auf einen optimalen Algorithmus hin.

## 7 Diskussion und zukünftige Forschungsrichtungen

Werfen wir zuerst einen Blick auf den Algorithmus. Die Kernidee, die dem hier vorgestellten Verfahren zu Grunde liegt, ist zwar nicht neu, die Kombination der einzelnen Schritte und die „Lochfüllmethode“ jedoch schon. Das Resultat, welches man erhält, besitzt einige wichtige und wünschenswerte Eigenschaften (siehe Kapitel 1), unter anderem sind die Modelle „wasserdicht“, die Methode unterstützt inkrementelles Update, sie ist nicht empfindlich, was Ausreisser und Unstetigkeiten anbelangt, und sie ist fähig, durchgehende Löcher mittels Space-Carving korrekt darzustellen.

Durch das Run-Length-Encoding auf das Voxel-Grid, und dem nach dem Resampling scanlineorientierten Traversieren der Struktur wird die Effizienz soweit gesteigert, dass es möglich wird, in relativ kurzer Zeit eine beträchtliche Anzahl Range-Scans zu erfassen und miteinzubeziehen. Es wurden schon Experimente mit bis zu 70 Range-Images durchgeführt, welche in ein hochauflösendes Voxel-Grid integriert wurden, was schlussendlich zu einem Mesh von einigen Millionen Polygonen führte. Die Tatsache, dass die resultierenden

Rekonstruktionen frei von Löchern sind, ist nicht zu unterschätzen, denn erst dadurch wird es möglich, einige Verfahren anzuwenden.

Trotzdem ist das Verfahren nicht perfekt. Sowohl der Algorithmus als auch die Scan-Technologie bringen einige Eigenschaften mit sich, welche gewisse Einschränkungen aufzwingen. Die von der Softwareseite her bekannten Probleme entstehen zum Beispiel in Bereichen, wo Flächen scharfe Kanten bilden, jedoch nicht alle Flächen hinreichend gescannt wurden. Weitere Probleme treten in Bereichen auf, wo das Objekt zu dünn ist. Wie in Kapitel 3 erklärt, ist es nicht möglich, im Vorhinein zu wissen, welche Punkte auf der optimalen Fläche liegen. Deshalb stellt man ein Intervall auf, in welchem die potentiellen Kandidaten liegen. Ist dieses jedoch grösser als das Objekt dick, so entstehen Fehler bei der Betrachtung der impliziten Funktion. Somit ist eine Mindestdicke des Objektes vorgeschrieben.

Doch auch die Hardware zwingt das Verfahren zu gewissen Restriktionen. Eine erste ist die, dass der Scanner nur die „äussere“ Oberfläche abtasten kann. Bereiche innerhalb von Löchern bleiben ungesehen. Weiterhin ist es durchaus möglich, dass sehr komplexe Gebilde eine grosse Zahl an Range-Scans benötigen. Dann kommt hinzu, dass die vom Laser ausgesandten Strahlen wieder auf einen Sensor treffen müssen. Dies bedingt, dass sowohl der Laser wie auch der Sensor einer gewissen Bewegungsfreiheit bedürfen, welche aber nicht bei allen Objekten gegeben ist. Zum Schluss stellt sich auch noch die Optik quer. Normalerweise wird Licht auf das Objekt geschossen und reflektiert, von einem Sensor registriert und evaluiert. Einerseits reflektieren zu stark glänzende Oberflächen das Licht auf unvorhersehbare Art, andererseits absorbieren zu dunkle Objekte zu viel Licht und zu helle Reflektieren das ambiente Licht zu stark.

Das Ganze lässt sich dahingehend verbessern, als dass man das Space-Carving schneller implementieren könnte. Es sollte überdies möglich sein, den Algorithmus auf einem Parallelrechner laufen zu lassen, dies dank der inkrementellen Eigenschaft der impliziten Funktion. Weiterhin kann man noch agressiveres Carving betreiben, z.B. bei all jenen Messungen, bei denen kein Licht zum Sensor zurückkommt. Es sollte aber bei Verbesserung der Scan-Technologie möglich sein, in nicht all zu ferner Zukunft grössere Objekte zu bearbeiten, unter anderem ganze Grundstücke und Gebäude.

## Literaturverzeichnis


[1] „A Volumetric Method for Building Complex models from Range Images“, Brian Curless and Marc Levoy
In *Proceedings of SIGGRAPH ’96*, pages 303-312, ACM Press

[2] „Better optical triangulation and volumetric reconstruction of complex models from range images“,
Brian Curless, PhD thesis, Stanford University

[3] „Better optical triangulation through spacetime analysis“, BrianCurless and Marc Levoy
In *Proceedings of IEEE International Conference on Computer Vision*, pages 987-994

[4] „Zippered polygon meshes from range images.“, G.Turk and Marc Levoy
In *Proceedings of SIGGRAPH ’94*, pages 311-318, ACM Press