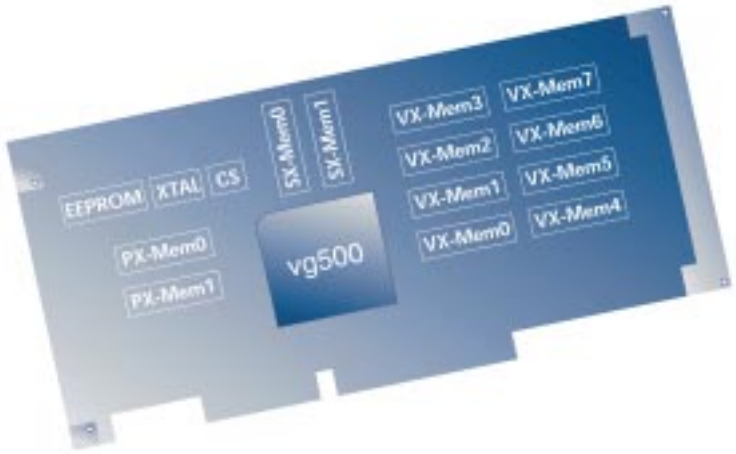
Der Mitsubishi VolumePro Real-Time Volume Rendering Beschleuniger

The VoxelBlaster Real-Time Ray-Casting System

Hanspeter Pfister et al.
Mitsubishi Electric Research Lab



Vortrag im Rahmen des Computer Graphik Seminars
im Sommersemester 1999
ETH Zürich

Stephan Würmlin

ABSTRACT

Real-time volume rendering is an enabling technology for medical applications including diagnosis, surgical training, and surgical simulation. This paper describes the Mitsubishi VolumePro Board, the world's first high-quality real-time volume rendering system for PC class computers. VolumePro implements object-space ray-casting with parallel slice-by-slice processing similar to shear-warp rendering. The discussion of the architecture focuses on the rendering pipeline, the memory organization, and sectioning. The paper describes several features of VolumePro, such as supersampling, supervolumes, and crop and cut planes. The system renders about 533 million interpolated, Phong illuminated, composited samples per second. This is sufficient to render a 256^3 volume at 30 frames per second.

1 Einleitung

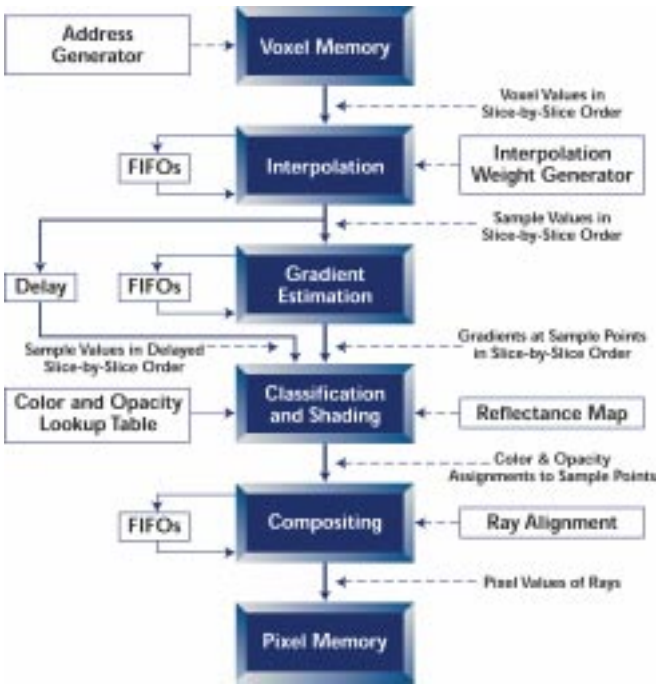
Die Visualisierung von wissenschaftlichen, technischen und biomedizinischen Daten ist ein stetig wachsender Bereich der Computer Graphik. In vielen Fällen sind die Objekte volumetrisch. Typisch werden diese Daten als dreidimensionales Gitter von Volumen Elementen, sogenannten Voxels, repräsentiert. Beispiele von volumetrischen Daten sind medizinische 3D-Daten (CT, MRI), simulierte Datensätze aus der Flüssigkeitsdynamik oder berechnete Modelle der finiten Elemente.
Bei diesem Volumenrendering wird ein Objekt nicht mit einer Liste von Oberflächenpolygonen oder Spline Patchs repräsentiert, sondern mit einem Volumen von gesampelten Datenpunkten. Volumen erlauben es uns, sowohl das Innere von Objekten als auch die Oberfläche darzustellen.
Während Volumenrendering eine populäre Visualisierungstechnik ist, limitiert doch die Absenz von interaktiven Bilderraten die Verbreitung. Um ein Bild zu rendern, braucht es normalerweise mehrere Sekunden. Um diese Limitierung aufzuheben, wurde das VolumePro System entworfen.
Das VolumePro System basiert auf der Cube-4 Volumenrendering Architektur [PK96], die an der State University of New York at Stony Brook entwickelt wurde. Mitsubishi Electric lizenzierte die Cube-4 Technologie und entwickelte die Enhanced Memory Cube-4 (EM-Cube) Architektur [OP97]. Das VolumePro System ist eine weiterentwickelte, kommerzielle Version von EM-Cube [ME] [MERL].
VolumePro ist ein Hardware System für Real-Time Volumenrendering auf Windows NT Computern. VolumePro rendert 500 Millionen interpolierte, Phong-beleuchtete Samples pro Sekunde. Dies Leistung reicht, um Datensätze von bis zu 16 Millionen Voxels (zum Beispiel 256^3) bei 30 Bildern pro Sekunde zu rendern.
Ein VolumePro System besteht aus einer VolumePro PCI Karte, einer normalen 3D Graphikkarte und Software. Die VolumePro Karte besitzt bis zu 128 MByte Volumenspeicher und einen vg500 Rendering Chip. Das Volume Library Interface (VLI) ist eine Sammlung von C++ Objekten, welche die Programmierungsschnittstelle (API) des VolumePro System beinhalten.

2 Ray-Casting - Der Rendering Algorithmus

VolumePro implementiert eine erweiterte Version des Ray-Casting Algorithmus, das Template-based Ray-Casting [Pf97]. Ray-Casting ist einer der meist gebrauchten "image-order" Volumenrendering Algorithmen. Ein "image-order" Algorithmus iteriert über alle Pixels des Ausgabebildes und bestimmt den Beitrag der Voxel zu jedem Pixel. Beim Ray-Casting werden Strahlen vom Betrachterpunkt in das Volumen geworfen. Dabei werden die Beiträge der Voxel entlang jedem Strahl dazu verwendet, um die Pixel-Farbe zu bestimmen. Bei "object-order" Algorithmen wird über die Volumendaten iteriert und dabei die Beiträge jedes Voxels zu den Pixels bestimmt. Ray-Casting offeriert hohe Bildqualität und ist einfach parallelisierbar.
Es wird eine Ray-Casting Technik verwendet, die, ähnlich wie Shear-Warp Rendering [LL94], die Datentraversierung folgendermassen durchführt: Anstatt die Strahlen in den Bildbereich zu werfen, werden die Strahlen von jedem Pixel auf der Basisebene in den Datensatz geworfen. Die Basisebene ist die Oberfläche der Volumendaten, welche am meisten parallel und am nähesten zur Bildebene liegt. Das resultierende Basisebenen-Bild wird dann mittels 3D Transformation und 2D Bildresampling auf die Bildebene projiziert. Dieser Bild-Warp wird auf der normalen 3D Graphikkarte und nicht auf der VolumePro Karte durchgeführt. Der Vorteil dieser Shear-Warp Technik ist, dass Voxels in Voxelebenen (sogenannten Slices) gelesen und abgearbeitet werden können, welche parallel zu dieser Basisebene liegen. Innerhalb eines Slice werden die Voxels, immer eine Scanline von Voxels zusammen, von oben nach unten aus dem Speicher gelesen.

Dies führt zu regulärem, "object-order" Datenzugriff. Zusätzlich erlaubt diese Technik Parallelismus durch Replikation der Rendering Pipelines, welche dann auf mehreren Voxels in einer Voxel Scanline zur gleichen Zeit arbeiten.

3 Beschreibung der Rendering Pipeline



Eine Charakteristik von VolumePro ist, dass jeder Voxel nur einmal pro Bild aus dem Speicher gelesen wird. So müssen Voxels durch die Pipeline rezirkulieren, um genau dann für Berechnungen da zu sein, wenn sie benötigt werden. Diese Rezirkulation ist das Herz der VolumePro Architektur.
VolumePro kann durch den hohen Grad des Pipelining in jedem Taktzyklus einen neuen Voxel akzeptieren. VolumePro besitzt auch keine Pipeline Verzögerungen. Dies bedeutet, dass VolumePro ein Volumendatensatz in der gleichen Geschwindigkeit rendern kann, wie die Voxel gelesen werden.
Nun folgt eine Beschreibung der Pipeline von VolumePro. Sie besteht im wesentlichen aus folgenden Elementen: Interpolationseinheit, Gradientenschätzungseinheit, Klassifikations- und Schattierungseinheit und Kompositionseinheit. Diese werden in den Kapiteln 3.1 bis 3.4 kurz vorgestellt. Gleichzeitig wird auch der Datenfluss durch die Pipeline beschrieben.

3.1 Interpolation

Voxels werden aus dem Voxelmemory gelesen und der Interpolationseinheit in der Reihenfolge Slice-by-Slice, Beam-by-Beam präsentiert. Ein Slice ist eine Voxelebene, welche parallel zur Basisebene ist. Ein Beam ist eine Scanline von Voxeln, welche parallel zu einer Hauptachse des Datensatzes ist. Dabei wird ein Voxel pro Takt gelesen.
Die Aufgabe der Interpolationseinheit ist die Konvertierung des Streams von Voxelwerten in einen Stream von Samplewerten.
Jeder Samplewert wird dabei von seinen acht nächsten Nachbarvoxels (nearest neighboring voxels) mittels trilinearer Interpolation bestimmt. Dabei muss nicht nur der aktuelle Wert verfügbar sein, sondern auch alle anderen Voxel der trilinearen Nachbarschaft. Dabei werden die Eingabevoxels und andere Werte im Voxel Slice FIFO, Voxel Beam FIFO und dem Voxel Shift Register gespeichert. Die trilineare Interpolation benötigt auch einen Satz von Gewichten. Diese werden von einem Weight Generator erzeugt.

3.2 Gradientenschätzung

Die Ausgabe der Interpolationseinheit ist ein Stream von Samplewerten. Dieser Stream wird der Gradientenschätzungseinheit präsentiert, um die x-, y- und z-Komponenten des Gradienten für jedes Sample zu schätzen. Dies geschieht mittels Zentraldifferenzen Gra-

dienten Filters.
Im VolumePro System werden die z-Gradienten von den Voxelwerten geschätzt und dann interpoliert, um die z-Gradienten der Samplepunkte zu erhalten. Die x- und y-Gradienten werden von interpolierten Samplepunkten bestimmt.
Der Gradient eines bestimmten Samplepunktes hängt nicht nur von Daten des aktuellen Slices ab, sondern auch von Daten des vorherigen und nächsten Slice. Somit operiert die Gradientenschätzungseinheit immer ein Voxel, ein Beam und ein Slice hinter der Interpolationseinheit.
Es müssen zwei volle Slices, zwei volle Beams und zwei Extrasamples gebuffert werden. Diese werden im Sample Slice FIFO, Sample Beam FIFO und im Sample Shift Register gespeichert.

3.3 Klassifizierung und Schattierung

Die Werte aus der Gradientenschätzung müssen nun mit den Samplewerten gepaart werden, welche dann der Klassifizierungs- und Schattierungseinheit präsentiert werden. Die Samplewerte kommen dabei direkt von der Interpolationseinheit. Da die Gradientenschätzungen viel später kommen als die Gradienten, müssen die Samplewerte verzögert werden.
Die Klassifizierung (Zuweisung der RGBα Werte) für jeden Samplepunkt ist ein Table Lookup des 12-bit Samplewertes in die 4096×36-bit Klassifizierungs Lookup Table. Diese Tabelle wird vorgerechnet und in das VolumePro System geladen. Die Ausgabe dieses Table Lookup ist ein 36-bit RGBα Wert.
Die VolumePro Rendering Pipeline implementiert das Phong-Beleuchtungsmodell für jeden Samplepunkt. Die Beleuchtung wird dabei berechnet, indem man die Beiträge der diffusen, spekularen und emissiven Beiträge aufaddiert. Sowohl für die diffusen wie auch für die spekularen Beiträge werden die Beleuchtungswerte über ein Table Lookop in einer Reflectionmap erhalten. Die Implementierung dieser Map unterstützt eine unlimitierte Anzahl von Lichtquellen. Da der Reflektionsvektor in Hardware berechnet wird, muss die Reflectionsmap Tabelle bei einem Wechsel des Betrachterpunktes nicht erneut geladen werden. Nur wenn die Anzahl oder die Position der Lichtquellen sich verändern, muss die Tabelle neu geladen werden.

3.4 Komposition

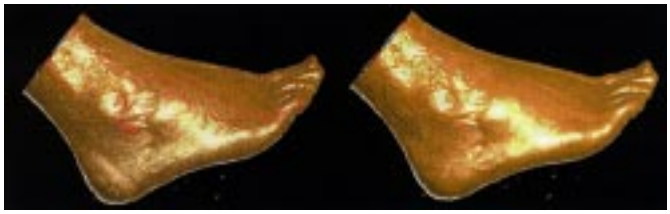
Die Ausgabe der Klassifikations- und Schattierungseinheit ist ein Stream von Farb- und Transparenzwerten für Samplewerte. Die Kompositionseinheit führt diese Samplewerte nun in die Pixelwerte der Strahlen zusammen. Da die Samples nicht in der Reihenfolge der Strahlen präsentiert werden, müssen die Pixelwerte jedes Strahles nun ein Samplepunkt nach dem anderern zusammengefügt werden. Dann müssen sie gebuffert werden, bis die Farb- und Transparenzwerte des nächsten Samplepunktes im Stream da ist.
Das Ray Slice FIFO, das Ray Beam FIFO und das Ray Shift Register speichern diese Werte der partiell akumulierten Strahlen.
Schliesslich, nach der Akumulierung der Farb- und Transparenzwerte aller Samplepunkte eines Strahles, wird der resultierende Pixelwert des Strahles vom VolumePro System ausgegeben. Pixelwerte der Basisfläche werden in den Pixelspeicher geschrieben und dann in die normale 3D Graphikkarte transferiert für den letzten Bild-Warp.

4 Spezielle Features

Die VolumePro Architektur unterstützt spezielle Features. Diese sind Supersampling, Supervolumes, partielle Volumen Updates, Cutplanes, Cropping, 3D Linien- und 3D

Ebenen-Cursors. Diese werden nur kurz beschrieben, ohne einen kompletten Überblick zu gewähren.

4.1 Supersampling



Die Supersampling-Technik verbessert die Qualität des gerenderten Bildes. Die Samplepunkte sind im Volumendatensatz in höherer räumlicher Auflösung definiert als im Voxelraum. Supersampling in x-Richtung würde mehr Samples pro Beam und in y-Richtung mehr Beams pro Slice bedeuten. In z-Richtung bedeutet es mehr Sampleslices pro Volumen. Das VolumePro System unterstützt Supersampling in Hardware nur in z-Richtung. Der Einfluss des Supersampling auf die Rendering Pipeline ist minimal. Die Renderingrate leidet jedoch stark darunter.

4.2 Supervolumes

Das VolumePro System kann, wegen limitierter On-Chip Buffergrösse der Slice-FIFO's, nur Volumendatensätze bis zu 256^3 Voxels rendern. Grössere Datensätze sind jedoch möglich. Dabei muss die Software den Volumendatensätze zuerst in Blöcke partitionieren. Jeder Block wird getrennt gerendert, und die resultierenden Bilder werden in Software zusammengefügt. Das gesamte Handling der nötigen Operationen wird von der VLI-Software durchgeführt. Es gibt keine Limitierung der Supervolumengrösse.

4.3 Subvolumes und partielle Volumen Updates

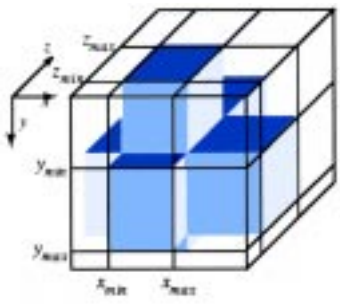
Der VolumePro Memorycontroller kann programmiert werden, um Subvolumen vom Hostspeicher zum Voxelspeicher und umgekehrt zu transferieren. Subvolumen können dann zwischen Bildern aktualisiert werden. Dies ermöglicht dynamische und partielle Aktualisierungen der Daten, um vierdimensionale Animationseffekte zu erhalten.
Ausserdem ermöglicht VolumePro, Sektionen von grösseren Supervolumen stückweise zu laden, um dem Benutzer zu ermöglichen, effektiv durch einen Teil des Supervolumens zu gehen, ohne immer das ganze Interessensgebiet neu zu laden.

4.4 Cropping und Cut Planes

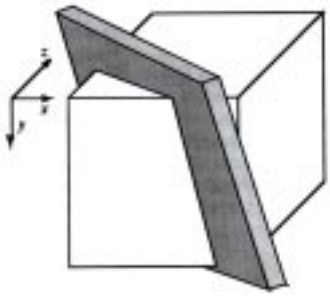
VolumePro unterstützt zwei Features, um einen Volumendatensatz zu clippen. Diese machen es möglich Slices, Crosssections oder Teile des Volumens zu visualisieren.

4.4.1 Cropping

Cropping (Schnitte) schneiden einfach vom Volumen Stücke ab, entlang Ebenen parallel zu den Oberflächen. Im Bild sieht man die drei Scheiben, die parallel zu jeder der drei Achsen des Volumendatensatzes verlaufen. Diese können auch kombiniert werden. Intersektionen, Vereinigungen und Inverse können dann dazu benutzt werden, Regionen der Sichtbarkeit zu definieren.



4.4.2 Cut Planes

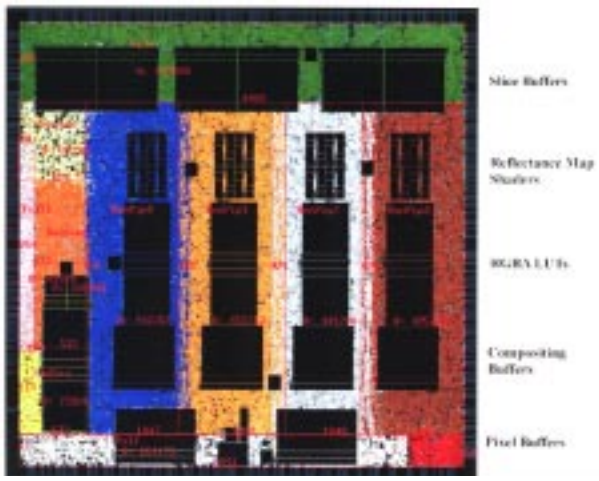


Cut Planes (Schnittebenen) sind planare Slices (Scheiben) einer gewissen Dicke durch den Volumendatensatz mit einem bestimmten Winkel. Samples sind nur sichtbar, wenn sie zwischen den parallelen Oberflächen und der Schnittebene liegen, andernfalls wird die Transparenz auf Null gesetzt. Alternativ können Samples auch nur sichtbar gemacht werden, wenn sie ausserhalb der Schnittebene liegen. VolumePro unterstützt nur eine Schnittebene.

4.5 3D Cursor

VolumePro besitzt ein Feature, welches einen in Hardware generierten, mit Software kontrollierbaren 3D Cursor in den Volumendatensatz einfügt. Der 3D Cursor erlaubt es dem Benutzer, räumliche Abhängigkeiten in einem Volumen zu erforschen und identifizieren. Die Cursors werden mit Alphablending in das Volumen eingeblendet. VolumePro unterstützt einen Cross-Hair und einen Cross-Plane Cursor.

5 Die vg500 Chip Architektur



Die Rendering Engine des VolumePro Systems ist der vg500 Chip (dies steht für Volume Graphics at 500 million samples per second!) mit vier parallel rendernden Pipelines. Es ist ein Application Specific Integrated Circuit (ASIC) mit etwa 3.2 Millionen Transistoren und 2 MBits on-chip SRAM. Der vg500 Chip wird in 0.35 micron Technologie fabriziert und läuft mit einer Taktrate von 133 MHz.

5.1 Parallele Pipelines

Um einen Datensatz von 256^3 Voxels bei 30 Bildern pro Sekunde zu rendern, muss das VolumePro System $256 \times 256 \times 256 \times 30$ Voxels pro Sekunde lesen können. Das sind etwa 503 Millionen Voxels pro Sekunde. VolumePro operiert mit einer Taktfrequenz von 133 MHz und kann in jedem Taktzyklus einen neuen Voxel lesen. Mit vier Pipelines erreicht VolumePro somit $4 \times 133 \times 10^6$ oder 532 Millionen Voxels pro Sekunde. Dies reicht für die angestrebten 30 Bilder pro Sekunde.
Die vier Pipelines operieren auf angrenzenden Voxels oder Samplewerten in x-Richtung.

5.2 Voxelspeicher Organisation

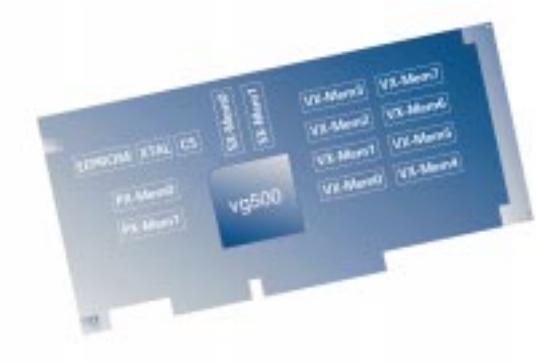
Der vg500 Chip besitzt vier 16-bit Speicherschnittstellen zum Voxelspeicher. Eine typische Voxelspeicherkonfiguration besteht aus vier SDRAM Modulen. vg500 benutzt 64-megabit SDRAMs, welche Burst Mode Zugriff mit 133 MHz besitzen. Im Maximum können sechzehn SDRAMs benutzt werden, welche eine maximale Voxelspeicherung von 128 Megabyte zulassen.

Voxels werden im Speicher so angeordnet, dass immer acht oder mehr aufeinanderfolgende Voxel gelesen werden können. Dabei werden die Voxels in sogenannten Miniblocks angeordnet. Ein Miniblock ist dabei ein 2x2x2 Array von in einem Speichermodul linear gespeicherten Voxels.
Miniblocks sind so auf Speichermodule verteilt, dass vier anliegende Miniblocks, in jeder Richtung, immer in verschiedenen Modulen gespeichert sind. Damit werden Speicherzugriffskonflikte verhindert.
Miniblocks in einem Speichermodul sind noch so angeordnet, dass anliegende Miniblocks nie im gleichen Speicherblock eines SDRAM Chips (Memory Bank) gespeichert sind.
Diese Speichermechanismen werden Skewing genannt.
Der vg500 Chip enthält die notwendige Hardware, um die Voxels in Miniblocks anzuordnen und um die obengenannten Skew- und Deskew-Mechanismen durchzuführen.

5.3 Sektionierung

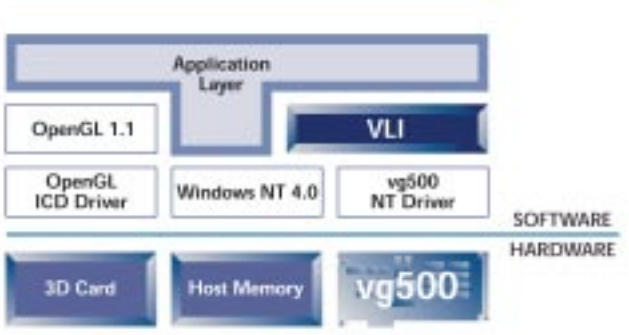
Um die Grösse des On-Chip Speichers der verschiedenen FIFO's für heutige Halbleiterprozesse in gewissen Limiten zu halten, wird ein Volumendatensatz in Sektionen zerlegt. Dabei werden gewisse Zwischenwerte zwischen der Abarbeitung der Sektionen in den externen Speicher ausgelagert. Im VolumePro System wird dieses Sektionierung nur in x-Richtung durchgeführt. Dabei ist jede Sektion 32 Voxels gross.

6 Die VolumePro PCI Karte



Die VolumePro PCI Karte besteht aus einem einzigen Board mit einer 32-bit 66 MHz PCI Schnittstelle. Das Board enthält einen vg500 Volumerendering ASIC, zwanzig 64 Mbit SDRAM mit 16-bittigen Datenpfaden, ein serielles EEPROM und eine Taktgenerierungsschaltung.
Der vg500 ASIC ist direkt mit dem PCI Bus des Systems verbunden. Zugriff auf die internen Register und die Off-Chip Speicher wird mittels der PCI Schnittstelle durchgeführt. Die Peak Burst Datenrate liegt bei 264 MByte pro Sekunde.
Die meisten Register sind write-only und sind memory-mapped über ihre eigenen PCI Basisadressregister. Volume-, Pixel- und Sectionmemory sind read-/writeable und direkt memory-mapped in den PCI Adressraum.
VolumePro unterstützt viele Standard 16-bit und 32-bit Pixelformate, und ist auch befähigt, on-the-fly Konversionen zwischen den Formaten durchzuführen.
Die Standardgrösse des Voxelspeichers ist 128 MByte, welche in vier Gruppen mit je vier SDRAM's organisiert sind. Zwei 64 Mbit SDRAM's sind der Sectionspeicher und ebenfalls zwei 64 Mbit SDRAM's sind der Pixelspeicher, welcher das gerenderte Basisebenen-Bild enthält. Sie können bis zu sechzehn Basisebenen enthalten, jede mit 512x512 32-bit Pixeln. Dies erlaubt Doublebuffering von mehreren Basisebenen auf der PCI Karte, pipelined Retrievel und Warping der Bilder. Die Basisebenen-Pixel werden über den PCI Bus zur normalen 3D Graphikkarte transferiert. Diese übernimmt dann den letzten Bildwarp und die Anzeige.
Das VolumePro System ist äusserst skalierbar. Zum Beispiel können verschiedene PCI Karten zu einem Hochgeschwindigkeitsnetzwerk zusammengeschlossen werden. Alternativ können mehrere vg500 ASIC's zu einem multi-processing Rendering Board zusammengeschlossen werden.

7 Das Volume Library Interface - VLI



Die VLI API ist ein Satz von C++ Klassen, welche den vollumfänglichen Zugriff auf die Features des vg500 Chips ermöglichen. VLI ersetzt keine vorhandenen API's. VLI arbeitet mit der 3D Graphik Library, wie zum Beispiel der OpenGL, zusammen, um das Rendern der Volumen und die Darstellung des Resultates in einer 3D Szene zu ermöglichen.

Die VLI Klassen können folgendermassen gruppiert werden:

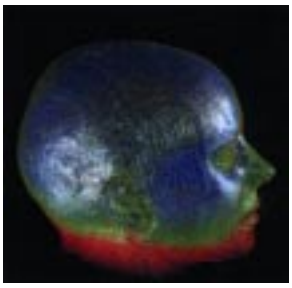
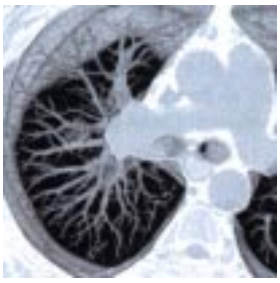
- **Volume Data Handling**
  Die VBIVolume-Klasse managt die Speicherung der Voxeldaten, das Voxeldatenformat und Volumentransformationen.
- **Rendering Element Creation**
  Diese Klassen erzeugen und spezifizieren verschiedene Rendering-Elemente:
  VBILookupTable, VBICamera, VBICrop, VBICutPlane, VBILight, VBICursor.
- **Rendering Context**
  Die Klasse VBIContext ist ein Container-Objekt, welches alle Attribute enthält, die für das Rendern des Volumen benötigt werden.

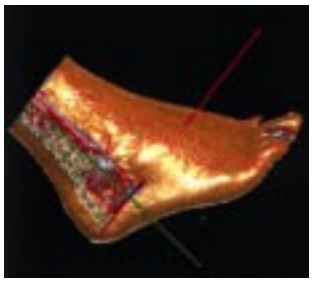
Die VLI berechnet automatisch die Reflectance-Maps, richtet die Alpha-Correction ein und managt Supervolumes, Supersampling und partielle Updates der Volumendaten.

8 Schlussbemerkungen

VolumePro ist das erste Single-Chip Real-Time Volumenrendering System. Die Leistung des Systems – 500 Millionen trilinear interpolierte, Phong-beleuchtete Samples pro Sekunde – ist enorm. Ein Reality Engine 2 System mit mehreren Textureinheiten erreicht mit vier Rastermanagern ca. 160 Millionen unschattierte, trilinear interpolierte Samples pro Sekunde [Pf99]. Auch wenn eine Onyx2 Infinite Reality sicher bessere Werte als die Reality Engine, auch mit Schattierung, erreichen kann, können diese “Texture Hardware” Volumenrendering Systeme immer noch keine hochqualitativen Real-Time Renderings durchführen.
Die wichtigen Beiträge des Papers sind die Reduzierung der erforderlichen Bandbreite und Inter-Chip Kommunikation mittels Block Skew Memory und die Reduzierung des on-chip Speichers mittels Sektionierung.
Sollte die VolumePro Karte wirklich diese hohen Bildraten erreichen, ist sie ein Muss für die Visualisierung volumetrischer Daten.
Die Limiten des VolumePro Systems sind die nicht vorhandenen Möglichkeiten der beliebigen Vermischung von Volumen und Polygonen, sowie die fehlende Unterstützung der perspektivischen Projektion.
Die vorhandenen Techniken des VolumePro Systems kombiniert mit der Ähnlichkeit des Texturmappings mit dem Volumenrendering werden sicher dazu führen, dass in der Zukunft Textursysteme alle Elemente der Volumenrendering Pipeline in Hardware implementieren.

Zu guter Letzt noch diverse mit VolumePro bei 30 Bildern pro Sekunde gerenderte Bilder .



Referenzen

[Pf99] H. Pfister. *Architectures for Real-Time Volume Rendering*. In Future Generation Computer Systems, Vol. 15, April 1999.

[OP97] R. Osborne, H. Pfister, H. Lauer, N. McKenzie, S. Gibson, W. Hiatt und T. Ohkami. *EM-Cube: An Architecture for Low-Cost Real-Time Volume Rendering*. In Proceedings of the SIGGRAPH/Eurographics Workshop on Graphics Hardware, August 1997.

[Pf97] H. Pfister. *Architectures for Real-Time Volume Rendering*. Ph.D. Dissertation, State University of New York at Stony Brook, January 1997.

[PK96] H. Pfister und A. Kaufman. *Cube-4 - A Scalable Architecture for Real-Time Volume Rendering*. In 1996 ACM/IEEE Symposium on Volume Visualization, October 1996.

[PKC94] H. Pfister, A. Kaufman und T. Chiueh. *Cube-3 - A Real-Time Architecture for High-Resolution Volume Visualization*. In 1994 ACM/IEEE Symposium on Volume Visualization, October 1994.

[LL94] P. Lacroute und M. Levoy. *Fast Volume Rendering using a Shear-Warp Factorization of the Viewing Transform*. In Computer Graphics, Proceedings of SIGGRAPH 94, July 1994.

[ME] Mitsubishi Electric, *Volume Graphics Business Group*.
Homepage unter http://www.3dvolumegraphics.com.

[MERL] Mitsubishi Electric, *MERL - Mitsubishi Electric Research Lab*.
Homepage unter http://www.merl.com.