社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－ＳＬＤＭ－123（17）
2006／1／18

DVIによる超高速単方向リンクを用いた並列ボリュームレンダリング

岡村 大† 野田 祐介† 三輪 忍† 嶋田 創† 中島 康彦††
森 眞一郎† 富田 眞治†

† 京都大学大学院情報学研究科 〒606-8501 京都市左京区吉田本町
†† 京都大学大学院経済学研究科/科学技術振興機構さきがけ 〒606-8501 京都市左京区吉田本町
E-mail: †{okamura,noda,miwa,shimada,nakashim,moris,tomita}@lab3.kuis.kyoto-u.ac.jp

あらまし 近年の計算機処理能力の急速な向上の中で注目を浴びる可視化方法の一つとして，ボリュームレンダリングが挙げられる．本稿では，従来提案してきたVisAのボリュームデータ三重化の欠点を克服した大規模データの並列ボリュームレンダリングを行うシステムについて紹介する．ハードウェア実装向けのレイ・キャスティング法の工夫とDVIインタフェースを用いた超高速通信路により，高レスポンス，高フレームレートを実現する．
キーワード 可視化，ボリュームレンダリング，並列処理，FPGA，ハードウェアアクセラレータ

Dai OKAMURA†, Yusuke NODA†, Miwa SHINOBU†, Hajime SHIMADA†, Yasuhiko NAKASHIMA††, Shin-ichiro MORI†, and Shinji TOMITA†

† Graduate School of Informatics, Kyoto University, Kyoto-shi, 606-8501, Japan.
†† Graduate School of Economics, Kyoto University/PRESTO. JST Kyoto-shi, 606-8501, Japan.
E-mail: †{okamura,noda,miwa,shimada,nakashim,moris,tomita}@lab3.kuis.kyoto-u.ac.jp

Abstract This paper introduces a new Parallel Volume Rendering System which eliminates redundant. With optimized ray-casting method for hardware implementation and high speed communication path using DVI-Link, we achieve good responce and high frame-rate.
Key words Visualization, Volume Rendering, Parallel Processing, FPGA, Hardware Accelerator

1. はじめに

近年の計算機性能の急速な向上に伴い，大規模かつ高精度な数値シミュレーションへの期待が高まっている．なかでも，実時間の数値シミュレーションの可視化技術は，大規模な3次元データの処理を伴う医療などの分野において，現在研究が進められている．このような次世代のシミュレーション環境では，オペレータによるシミュレーション対象へのインタラクティブな操作に対応して，実時間でシミュレーションを行うとともに，即刻その結果を可視化などの手段により提示することが求められる．

我々は，個人あるいは小規模な組織単位で占有利用可能なPCクラスタを用いて，インタラクティブな数値シミュレーション及びその可視化を実時間処理する環境について研究を行っている．1台では実時間処理が不可能な大容量の数値データを分散処理し，再度集約し表示機器に出力するという一連の処理の性質上，高速な伝送路と実時間処理可能な可視化機構が不可欠である．

上記の可視化機構を実現可能なハードウェアアーキテクチャとして，我々はRevolver/C40[1],[2]やVisAを過去に提案し，VisAのプロトタイプとして処理の手順をそのままにボリュームデータの処理可能サイズを半分に制約したVisA Proカードを開発した[3],[4].

VisAアーキテクチャはボリュームレンダリングのひとつの有効な実装方法であるが，ボリュームデータを三重化して保持しているという欠点が存在する。

我々はこの欠点を取り除くためにVisAのアーキテクチャからは一旦離れ，ボリュームレンダリングのレイ・キャスティング法を工夫し，視点依存による三重化が不要な可視化機構を提案する．また，大規模データでのフレームレート向上のために，DVIインタフェースを単方向の高速通信路として利用し，並列分散処理による高速化についての検討も行った．

本稿では，その検討内容と現在の実装状況の報告を行う．以下では，2章でボリュームレンダリングの概要について説明し，3章で各種レイ・キャスティング法の紹介との今回ハードウェア化・並列化を行う上での検討，4章で我々が提案するDVIイ

ンタフェースの通信路として用いた並列処理の仕組みについて述べ，5章でまとめを行う。

2. ボリュームレンダリング処理の概要

ボリュームレンダリングとは，シミュレーション等により得られた3次元空間上の数値データを色Cと透明度tに対応づけ，3次元空間内部のデータの分布状況を可視化する．具体的には，視線上のボクセルの値を視点に近い順に $v_0, v_1, v_2, \cdots$ とするとピクセル値は次の式（畳み込み演算）で計算される．

$$C_k = \sum_{i=0}^{k}(1 - t(v_i)) \cdot c(v_i) \cdot \prod_{j=0}^{i-1} t(v_i) \qquad (1)$$

ここで $c(v_i), t(v_i)$ はそれぞれボクセル値 v_i を，色，透明度に変換して値であることを示している．さらに，式(1)は以下のような漸化式で表わされる．

$$C_k = C_{k-1} + (1 - t(v_k)) \cdot c(v_k) \cdot T_{k-1} \qquad (2)$$

$$T_k = t(v_k) \cdot T_{k-1} \qquad (3)$$

この畳み込み演算による合成は，隣接関係（特に視点からの距離における前後関係）を保持する限りには，部分合成が可能であるという特徴がある．すなわち，演算区間をいくつかの部分区間に分割し，それぞれの区間内での重畳計算を済ませ，その中間結果最後に畳み込み演算しても結果は同じになる。そこで，1) 複数のノードに分散して割当てた部分3次元空間（以下 サブボリュームと呼ぶ）に対して畳み込み演算を行い，2) 各ノードで得られた画像と画素毎の透明度を視点からの距離の順番に従って順次パイプライン的に合成する，という手法による並列化が可能である．

3. レイ・キャスティング法の工夫

ボリュームレンダリングの際，視線上に存在するボクセルの値を順に取り出す作業をレイ・キャスティングと言う．この処理は1フレーム分の描画で，ボリュームデータ全体を1度走査せねばならず，いかに早く全体を走査するかによってフレームレートが決定する．

ハードウェアによる実装を考えるとき，ボリュームデータを格納する場所は大容量で高速という条件からDRAMを選ぶことになる．近年DRAMは，一般的なプログラムの参照局所性に基づき，バースト転送を前提として高バンド幅を実現するという流れで高性能化が進んでいる．しかし，ボリュームレンダリングはその処理の性質上，ボクセルデータの時間的局所性に乏しく，空間的局所性についても視線に依存して低下する．このため，ボリュームレンダリングにおいてメモリに要求されるバンド幅を満足させるには，DRAMのバースト転送を念頭に置き，インタリーブやバンク切り替え，リフレッシュなどの処理によるデータレスポンスの低下を隠蔽できるレイ・キャスティング法を考える必要がある．

この章では，代表的なレイ・キャスティング法でありVisAでも用いているピクセル順レイ・キャスティング法と，ライン単位で保持されるCPUのキャッシュ特性に着目してこのレイ・キャスティング法の欠点を取り除いた，キューボイド順レイ・キャスティング法[5]を紹介する．そして，ハードウェア向けの実装方法として我々が提案するレイ・キャスティング手法を説明する．

3.1 ピクセル順レイ・キャスティング法

ピクセル順レイ・キャスティング法では，視線ベクトルを計算し，

（1） サンプリング

（2） ピクセル値更新

（3） 次サンプリング点の決定

をボリューム空間を外れるまで繰り返す，という処理を各ピクセルについて順次行っていく．2次元で4ブロックバースト転送と仮定し，前述の空間局所性を考えると，視線が左から右に向かう図1の左の場合，視線方向に沿って4つのボクセル値が順にサンプリングされ，キャッシュ・ヒット率は最大化される．一方視線が下から上に向かう図1の中央では，1回のバースト転送で得られるボリュームのフェッチ方向と視線が垂直に交差するため，フェッチした4つのデータのうち1つのボクセル値しか利用できない．中央の方法でライン・サイズが128ボクセルと仮定すると，左と同じフレームレートを求める場合128倍の転送速度が必要となり，メモリのアクセス速度の限界によってフレームレートが頭打ちになる．

このようにピクセル順レイ・キャスティング法はアルゴリズムが単純でハードウェア化は容易であるが高速化をはかる上での障壁が大きく，視線依存の速度低下が許容出来ない範囲に広がってしまう．

3.2 キューボイド順レイ・キャスティング法

ボリュームレンダリングでは，重畳計算処理の中断が可能であることに着目し，参照の空間的局所性を向上させる方法として，額田らが考案したキューボイド順のレイ・キャスティング法がある．

このレイ・キャスティング法では，あらかじめ視線ベクトルの計算を全ての視線について完了しておき，ボクセルをキャッシュ・メモリに蓄えられるだけのデータ量で空間分割し（個々の空間をキューボイドと呼ぶ），スクリーンに近いキューボイドから順に

（1） キューボイドを通る全ての視線に対し重畳計算を行う

（2） 中間値を各視線に対応するピクセルごとに保存する

（3） 次のキューボイドで登場する視線に対応した中間値を取り出し前データとして渡す

という処理を繰り返す．これによりバースト転送を活かしたメモリ・アクセスで蓄えられる値を全て利用し，視点依存の要求バンク幅の増大を抑えることが可能となる．図1右で処理順序を例示すると，1-1，1-2，...4-3，4-4の順となる．

額田らの論文[5]のItanium2プロセッサ上でのC++プログラムの処理速度のテストによれば，視点位置が最悪のパターンで6倍の速度向上し，最良のパターンでの中間値保存によるオーバーヘッドによる速度低下も1.2倍程度で済んでいる．

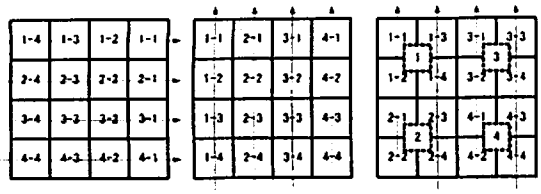

図 1　視線の方向とサンプリング点の処理順序

3.3　キューボイドの原理を応用したハードウェア実装向けレイ・キャスティング方

キューボイド順レイ・キャスティング法の中間値を一時保存する仕組みを利用することで，DRAM からのバースト転送に対しても同様のアプローチができる。

キャッシュの代わりにプリフェッチバッファを設け，適当な数のボリュームの集合（ここでは仮にブロックと呼ぶことにする）を 1 つの単位としてバースト転送によりバッファに取り込み，ブロックを通る視線上のボクセルを，キューボイドのレイ走査順序をまねて順に処理し，画面（最終画像出力用のパネル）のピクセル値に一時保存する形をとる．

CPU のキャッシュを前提としたキューボイド順レイ・キャスティング法の実装に比べ，プリフェッチバッファによるブロック単位のレイ・キャスティングでは，FPGA のリソースの範囲内で自由にバッファサイズを決定でき，入っているデータをより無駄なく使うことが出来る点で優れる．また，レンダリング処理の間に次の読み出しの初期化を行うことにより，DRAM のイニシャライズ時間の隠蔽と，バースト転送によって取り出されるデータのヒット率向上を同時に見込むことが出来る．これにより，必要バンド幅を抑えられるほか，DRAM のメモリ・アクセス回路の複雑化も避けられる．

4.　並列ボリュームレンダリングにおける通信

2 章で示したとおり，ボリュームレンダリングでは色情報 C と透明度 T を重畳し最終的に色情報を画面に表示する．並列化を考えたとき，重畳し終わったデータについては C を，終わっていないデータについては中間結果 (C,T) が伝搬される．各ノードには自分のボリュームの計算結果と前段のノードから伝搬された中間結果を合成する仕組みが必要となる．

既存の実装方法には，ソフトウェアによる実装では BSC や SLIC，専用ハードウェアによるクラスタにおける実装としては八分木の通信路を持った VG クラスタなどがある．これらはノード間の自由な通信路が確保されていることを前提としている．我々は単方向通信路によるリング的なクラスタでも，通信手法を工夫することで並列処理できる方法を提案する．

4.1　単方向リンクによるレンダリング実現の手法

中間結果を伝播する部分を拡張し隣接ノードに限定しないデータの伝播路を確保する．これにより，ボリューム同士は隣接するがノードの並びの関係から連続で処理できないボリューム空間についても，中間値の保存・伝搬を用いて一筆書きのボリューム空間の走査でレンダリングが可能となる（図 2）．

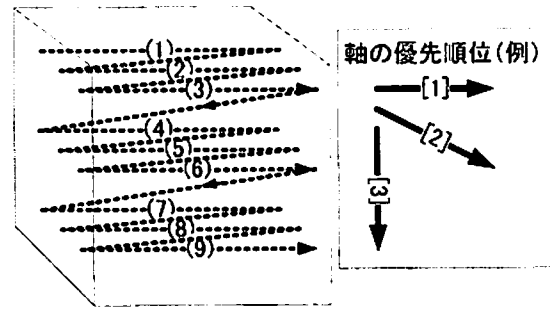

図 2　一筆書きによるボリューム空間の走査

ただし手法を単純に実装すると，ノード間の通信が増大した場合に中間合成結果のノード内への格納サイズ増加と合成手順の多段化によるレイテンシの増大が避けられない．そこで，筆者は超高速伝送路を活用した遅延線メモリ的な伝搬路を提案する．

4.2　DVI インタフェースによる通信

DVI インタフェースは本来液晶ディスプレイのデジタル接続用のインタフェースであるが，このインタフェースは液晶のピクセルクロックに同期して 1 ピクセル辺り 24bit(RGB 各 8bit) のデータを 1 フレーム分 × 毎秒 60 枚のデータを伝送できる通信路と見なすことが出来る．たとえば UXGA(1600×1200pixs) の表示では，

$$1600 \times 1200 \times 24 \times 60 = 2.76\text{Gbps}$$

という高速通信を行っている．提案する単方向リンクの実装に最適な安価な伝送路であり，また液晶への結果出力が容易という点も考慮し，ここでは DVI インタフェースによる具体的な通信方法を検討する．

4.2.1　ピクセル対応の通信方法

ピクセルの色情報を格納する 24bit をメモリのセル，フレーム中での位置 x-y をアドレス，図 3 のように数フレームを単位周期として単位周期内でバンク番号を割り振る．これにより DVI インタフェースによるディスプレーへの出力機構を 24bit 幅の FIFO の役割をする遅延線メモリ（書き換えはセル単位でパイプライン処理が可能）として扱う事ができる（図 4）．

4.2.2　時間多重化の検討

各ノードで通信に利用するバンクを決めて割り当て，自分のバンクの更新と担当外バンクの転送を行うことで，隣接しないノードへの伝播を実現する．また，バンクの利用割り当てに際しては中間データの処理が完了次第占有バンクの解放ができるようにする（図 5）．

これにより，透視投影の場合に想定されるノードごとの負荷のばらつきに対し，自ノードの処理がすんだノードが前後のノードの重畳をスケジューリングするなどの負荷分散での対応が可能となる．

4.2.3　空間多重化の検討

1 つのバンク内のフレーム空間に対しノードあたりの中間データ保存サイズは小さく，ノードごとのアクセス領域は独立

している．この性質を活かし重なり合わないサブボリュームを扱うノード同士は同一フレームの共用を行う（図6）．

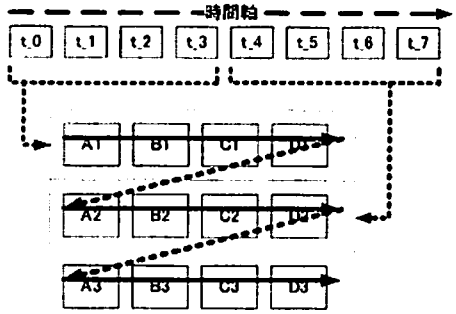

図3　時分割でのフレームの番号付け

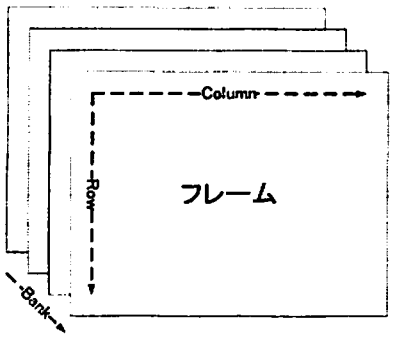

図4　ピクセルをメモリに対応づけた通信方法

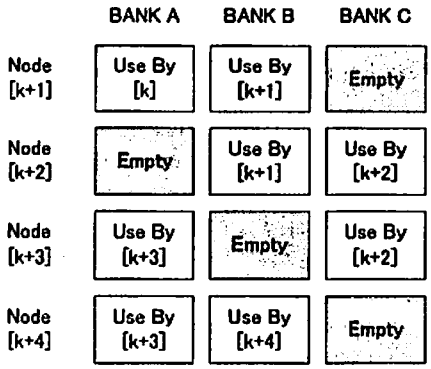

図5　バンクのノード間共有の概要

5. ま と め

レイ・キャスティング方法と並列化の通信方法を工夫することで，VisA のボリュームデータ三重化という欠点を克服したボリュームレンダリングのアーキテクチャを示した．

現在，VisA Pro カードに提案する単方向リンクを用いたボリュームレンダリング回路の評価回路を実装している．今後この回路を用いて，レイテンシやフレームレートの評価を行っていく．

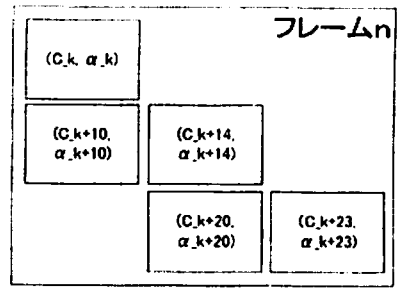

図6　フレームのノード間共有の概要

謝　辞

日頃より御討論いただく京都大学大学院情報学研究科富田研究室の諸氏に感謝します。本研究の一部は、日本学術振興会科学研究費補助金 基盤研究 S (課題番号 16100001)、21 世紀 COE プログラム (課題番号 14213201)、ならびに、文部科学省特定領域研究 S (課題番号 13224050) による。

文　献

[1] 對馬, 他, "ボリューム・レンダリング専用並列計算機 ReVolver のアーキテクチャ", 情報処理学会論文誌, 第 36 巻, 第 7 号, pp.1709-1718, 1995.

[2] 原瀬, 他, "ReVolver/C40 を用いた時系列ボリュームデータの実時間可視化," 情処研報 2002-ARC-142, pp.7-12, 2002.

[3] 生豊 公啓, "時変ボリュームデータの実時間可視化のための専用グラフィックスカード VisA の開発", 京都大学大学院情報学研究科 修士論文, 2003.

[4] 森, 他, 960MB/s の DVI-D 入出力リンクと DDR-SDRAM を 2 系統を持つ FPGA 搭載 PCI カード- 並列可視化処理への応用 -, 11 回 FPGA/PLD Design Conference 論文集, pp.31-34, Jan. 2004.

[5] 額田, 他, 参照局所性を最大化するボリューム・レンダリング・アルゴリズム, 情報処理学会論文誌：コンピューティングシステム, Vol. 44, No. SIG 11(ACS 3), pp.137-146(2003.08).