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

對馬雄次[†,☆] 明石英也[†,☆] 金喜都[†]
森眞一郎[†] 中島浩[†] 富田眞治[†]

ボリュームレンダリングは，CT-スキャナ，MRI などから得られる人体の断面から，立体的な画像を生成する手法として非常に重要な位置を占めている手法である．また，この手法は，科学技術計算の結果生じる3次元データの解析手段としての可視化手法としても注目を浴びている．しかし，ボリュームレンダリングには膨大な計算資源を必要とするため，高速化が困難であった．特に科学技術計算の結果の可視化には，遠近法による画像生成と，半透明なボリュームデータの取り扱いが必要なため，高速化はさらに困難である．そこで，本稿では，科学技術計算の結果の高速な可視化を目的としたボリュームレンダリング専用並列計算機 ReVolver のアーキテクチャについて提案を行う．ReVolver では，ボリュームデータを3重化して格納することにより，従来の専用並列計算機で問題であった，ボクセルアクセス時のバンクコンフリクトを無くすことを最大の特徴としている．この特徴により，半透明なボリュームの遠近法を用いた描画を高速に行うことができる．さらに，ReVolver では，ボリュームデータを用いたレイトレーシングをサポートすることで，従来のサーフェスデータでは表現の困難であった境界の曖昧な対象を表現する能力も持つ．

A Parallel Computer Architecture for Volume Rendering : ReVolver

YUJI TSUSHIMA,[†] HIDEYA AKASHI,[†,☆] XIDU JIN,[†]
SHIN-ICHIRO MORI,[†] HIROSHI NAKASHIMA [†]
and SHINJI TOMITA [†]

Volume Rendering is very important technique to generate 3 D image of medical data and to analyze results of scientific computations. However, Volume Rendering needs so much computing power that it is difficult to build real-time Volume Rendering machine. Moreover, in the case of scientific visualization, it becomes more difficult to realize such machine, because scientific visualization requires to process perspective projection and to deal with translucent volume data. This paper describes the architecture of real-time Volume Rendering machine : ReVolver. In order to provide conflict free memory access along any ray, though previously proposed machines are suffered from bank conflict problem, ReVolver solves this problem by having triple volume data. With this feature, ReVolver produces images at high speed by perspective projection using translucent volume data. Furthermore by supporting ray tracing with volume data, ReVolver can visualize obscure boundary objects which are hardly expressed by surface model.

1. はじめに

ボリュームレンダリングは従来医療分野においてMRI 等からの断面画像から立体的な画像を生成するのに使用されていた．さらに，今日では計算機環境の発達により膨大な量の科学技術計算の結果が生成されており，計算結果の解析手法として，ボリュームレンダリングによる可視化が注目されている．

しかし，ボリュームレンダリングには膨大な計算資源を必要とするので，汎用の並列計算機上での並列アルゴリズムや専用の並列計算機が開発されてきた．しかし，多くの専用並列計算機は医療画像を扱うことが主な目的であり，平行投影法により不透明なボリュームデータを用いて画像生成を行ってきた．

これに対して，本稿で提案するボリュームレンダリング専用並列計算機 ReVolver では，科学技術計算の結果の可視化を行うことも目的としている．このため，遠近法に基づく画像生成と，半透明なボリュームの表示が必要条件となる．

これらを満足させるために，ReVolver では単純化したサンプリング方法とそれに合ったメモリ構成を採用することにより，任意の視線に関して，バンクコン

† 京都大学工学部
Faculty of Engineering, Kyoto University
* 現在，日立製作所
Presently with Hitachi, Ltd.

フリクトなしでデータを読み出せる構成をとることで対処している．さらに，ReVolver では，ボリュームレンダリングにレイトレーシング法の原理を応用し，雲のように境界が曖昧で，従来のサーフェスデータでは表現の困難な対象の表示[10),11)]を行う機構を持つ．

なお，本稿では，ボリュームデータは，3 次元空間を等間隔でサンプリングした各格子点ごとにデータの属性値（色，透明度，温度等）が与えられるデータ構造とし，各格子点の値をボクセル値と呼ぶ．

以下，2 章では既存のボリュームレンダリング専用並列計算機における並列処理方式を分類し，3 章では，われわれの開発している ReVolver での要件と設計方針について述べる．4 章では，ReVolver の構成について述べ，5 章では，レイトレーシングをサポートする際の問題点とその対処法について述べる．さらに 6 章では，既存の専用並列計算機と汎用の並列計算機を用いた手法との比較を行い，最後に 7 章でまとめと今後の課題について述べる．

2. ボリュームレンダリングでの並列処理方式

（1） 後方投影アルゴリズムの並列化：視点からスクリーン上の各ピクセルへ順に視線を生成し，視線と交差するボクセルからピクセルの輝度（以下ではピクセル値）の計算を並列化する手法．

・ピクセル並列処理：スクリーンを分割し，サブスクリーンごとにプロセッサ（以下，PE）を割り当てる方式．各 PE では後方投影アルゴリズムによりピクセルの輝度を求めていく．PARCUMII[4)] に見られる（**図 1** 右上参照）．

・ボクセル並列処理：後方投影アルゴリズムにおいて，各視線に対するボクセルの処理を並列化する．CUBE[5)] に見られる（図 1 左上参照）．

（2） 前方投影アルゴリズムの並列化：ボリュームを構成する各ボクセルからスクリーン上の各ピクセルに向かって投影することにより，各ピクセルの値を求めるアルゴリズムの並列化．

・ボリューム並列：ボリューム空間を分割して，各サブボリュームの処理に PE を割り当てる方式．各 PE は，サブボリュームをスクリーンに投影し，最終的に各 PE の画像をマージすることにより画像を得る．GODPA[3)]，SCOPE[9)]，3 DP[4),7)] に見られる（図 1 下参照）．

3. 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

3.1 アーキテクチャの要件

ReVolver の目的は，既存の専用並列計算機と異なり，科学技術計算の結果の可視化を目標としているため，以下の要件を満たさなければならない．

（1） 2 種類のボリュームデータのサポート

・離散モデル：単位立方体の中心にボクセル値が存在し，ボクセル内の任意の点のボクセル値を同一とみなすモデル．

・連続モデル：単位立方体の頂点にボクセル値が与

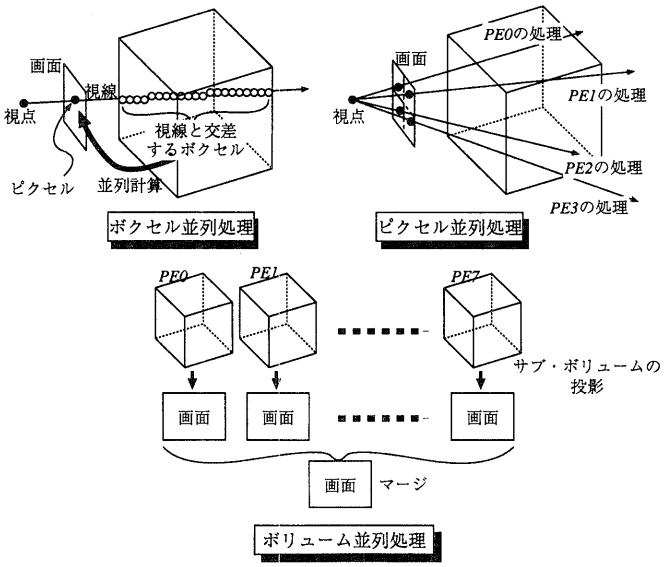


図 1 並列処理の方式
Fig. 1 Parallel processing methods.

えられ，ボクセル内の点の値は線形補間により求めるモデル．

（2） 半透明なボリュームの表示：内部の様子を可視化できるようにするために必要．

（3） 遠近法のサポート：科学技術計算では，地震波の解析[8]などのように大きな空間を扱うものがある．このような場合には，遠近法での表示が有効となる．

（4） リアルタイム表示：ボリュームデータの解析・理解の援助として，視点の移動による再描画や断面の表示をリアルタイムに行うのは重要である．

（5） レイトレーシング法のサポート：従来のサーフェスデータでは表現の困難な，境界の曖昧な対象を可視化するために，レイトレーシング法をボリュームデータを用いて行う．このため，視線とボリュームの交差判定機構と，反射光・屈折光（以下，高次光と呼ぶ）を扱う機構を設ける．

3.2 設計方針

前節での要件を満たすために ReVolver では，以下のような設計方針をとる．

（1） 後方投影アルゴリズムの採用：半透明なボリューム表示を行うため，レイと交差するボクセル間の順序を保存しなければならない．前方投影アルゴリズムでは，ボクセル間の前後関係を判定するのが困難であり不向きであるので，ボクセル間の順序付が容易な後方投影アルゴリズムを採用する．

（2） ボクセル並列処理による高速化：同時に複数の視線を処理するピクセル並列処理では，ボリュームメモリへのアクセス競合が起こる可能性が高くなり高速化しにくい．そこで，ボクセル並列処理を採用し，木構造の並列パイプライン処理により，1ピクセルをパイプラインピッチで計算を行い高速化を図る．

（3） 任意の視線に対してコンフリクトフリーなメモリ構造：ボクセル並列処理で効率よく計算を行うには，レイと交差するボクセルを効率よくアクセスできることが最も重要である．そこで，サンプリング方法を単純化し，任意の視線に対して交差するボクセルを同時に取り出すことのできるメモリ構造を実現する．さらに，レイトレーシングでは視線の方向が任意となるため，このメモリ構造はレイトレーシングのサポートのためにも必要となる．

4. ReVolver の構成

4.1 全体構成

3.2節で述べた方針に基づき設計されたボリュームレンダリング専用並列計算機 ReVolver の概要を図2に示す．以下では各部について述べる．

（1） SCU (System Control Unit)：システム全体を制御し，ホストマシンからのコマンド/ボリュームデータ/各種パラメータを各部に伝達する．

（2） 視線生成ステージ：SCU から送られてくる視点/スクリーンの情報から各ピクセルを通るレイを生成し，ボクセル値生成ステージに送る．レイトレーシングを行う時は，シェーディングステージからの情報に基づき視点を退避させ，高次光をボクセル値生成ステージに送る．

（3） ボクセル値生成ステージ：ボクセル値生成ステージは以下の2つの部分から構成されている．

（a） VMU (Volume Memory Unit)：視線生成ステージより送られてくるレイの情報に基づいて，レイと交差するすべてのボクセル値を並列に読み出す．詳しい構成は4.2.2項で述べる．

（b） c/α-LUT (Look Up Table)：VMU から読み出されたボクセル値から，色・透明度・屈折率を生成する．また，レイトレーシング法のための交差判定も行う．構成は，4.2.3項で述べる．

（4） ピクセル値計算ステージ：SCU からの計算モードにしたがって，c/α-LUT で生成された色・透明度から，ピクセル値を計算する．

計算モードが通常のボリュームレンダリングの時，ピクセルを通るレイと交差するボクセルを視点から近い順に $v_0, v_1, \cdots, v_{n-1}$ とすると，以下に表される式(1)を評価し，そのピクセルの値を求める．

$$P=\sum_{i=0}^{n-1} K_i \prod_{j=0}^{i-1} \alpha(v_j) \qquad (1)$$

$c(v_i)$，$\alpha(v_i)$ は，ボクセル値 v_i に対する色，透明度であり，K_i は，$c(v_i)(1-\alpha(v_i))$ を表している．一方，計算モードがレイトレーシングの時，通常のピクセル値計算に加えて，レイと表面の交差位置の特定を行い，交差する場合には，交差直前のボクセルまで式(1)を評価した値をピクセル値とする．ピクセル値計算ステージの構成については4.3節で，また交差判定については5.1節で述べる．

（5） シェーディングステージ：ピクセル値計算ステージから送られて来るピクセル値からシェーディングを施したスクリーンを構築する．通常のボリュームレンダリングでは depth gradient shading を行う．レイトレーシングでは，レイと表面の交差する点での法線計算，ならびに高次光の方向の計算を行い，シェーディングは行わない．

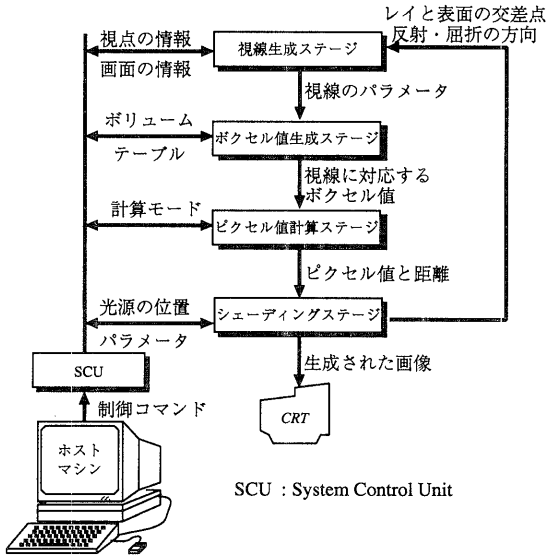


図2 ReVolver の全体構成
Fig. 2 Structure of ReVolver.

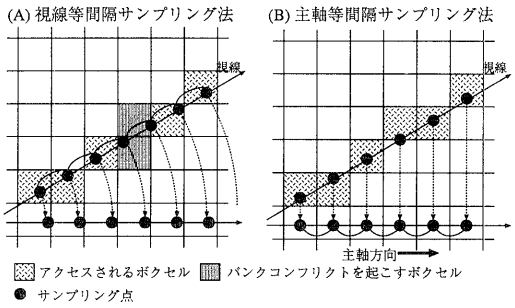


図3 サンプリング方法
Fig. 3 Sampling methods.

4.2 ボクセル値生成ステージの構成

4.2.1 サンプリング方法の単純化

ボリュームデータは，x, y, z 軸方向にそれぞれ n 分割された立方体である．このボリュームデータをある軸方向に垂直な平面で分割した場合，従来のサンプリング方法では，レイの方向にかかわらず，レイの方向に対して等間隔にサンプリング（以下，視線等間隔サンプリング法）を行っているため，図3(A)のようにバンクコンフリクトを起こす場合がある．

ReVolver では，レイベクトルの要素の絶対値が最大となる座標軸（以下，主軸）の方向に等間隔でボクセルのサンプリングを行う主軸等間隔サンプリング法を採用する（図3(B)参照）．この場合，任意の方向のレイに関して，サンプリングされるボクセルの座標の主軸成分がすべて異なるため，バンクコンフリクトを生じない．このことを利用し，任意の方向のレイに対してバンクコンフリクトせずにアクセスできるメモリ構造を4.2.2項で示す．

4.2.2 VMU の構成

ボリューム空間を主軸と垂直な n 枚の平面に分割し各々を別のバンクに格納すれば，ある方向のレイの主軸方向にサンプリングすればバンクコンフリクトをなくすことが可能となる．そこで，任意の方向のレイでは主軸として x, y, z 軸が可能であるので，図4のように，ボリューム空間をそれぞれの軸に対して垂直な平面群に分割し，各平面群ごとに順に $1, 2, \cdots, n$ と番号を付け，それぞれの平面群での i 番目の平面上のボクセルを，i 番目の VMU ノードの対応するメモリバンクに格納する（図4参照）．

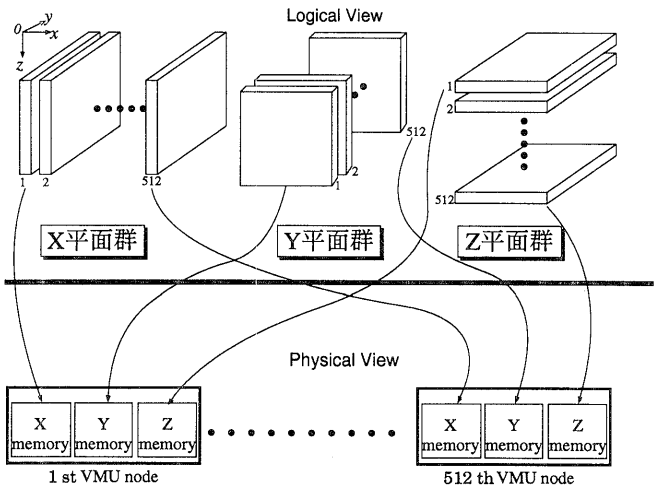


図4 VMU ノードの構成
Fig. 4 Structure of VMU-node.

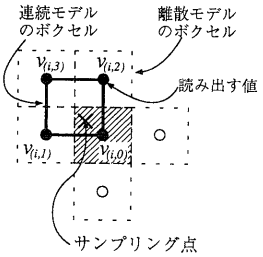


図5 VMU からの読み出し方法
Fig. 5 VMU access patterns.

すなわち，ボリューム内の座標が (p, q, r) のボクセルが格納されるのは，p 番目の VMU ノードの X バンク，q 番目の VMU ノードの Y バンク，r 番目の VMU ノードの Z バンクになる．この方法では，ボリュームデータは3重化されていることになるが，任意の主軸に対して垂直な平面上のボクセルが異なるメモリバンクに割り付けられるので，任意のレイに対してバンクコンフリクトしないことを保証できる．

VMU ノード内のメモリバンクへのアクセスは，レンダリングの方法に応じて以下の3通りの方法で行う．

（1） 離散モデルのボクセルへのアクセス：レイと交差するボクセルを読み出す．読み出すボクセルは図5では，斜線部のボクセルとなる．

（2） 連続モデルのボクセルへのアクセス：レイと交差するボクセルの側面にある4ボクセルを読み出す．図5の黒丸で示したボクセルとなる．

（3） レイトレーシングでのアクセス：レイが交差するボクセルを中心として，上下左右のボクセルを読み出す．図5では，$v_{(i,0)}, v_{(i,2)}, v_{(i,3)}$ と白丸の5ボクセルとなる．これは，視点を主軸以外の軸方向に平行移動した点から元のレイと平行なレイをキャストして交差するボクセルを読み出しているのと等価である．この平行レイを用いて，レイトレーシングにおける法線ベクトルの算出を行う．

4.2.3 c/α-LUT の構成

VMU から読み出されたボクセル値から式(1)に必要な色・透明度，レイトレーシングに必要な屈折率をパイプライン的に求める．この色・透明度・屈折率の情報はあらかじめ作成されており，テーブルに格納されている．レンダリングの際はテーブル参照によりこれらの値の生成を行う．このテーブルへの参照方法は，目的に応じて以下の3通り用意する．

（1） ボクセル値から直接求める：VMU ノードから読みだされたボクセル値をインデックスとしてテーブル参照を行う．通常のボリュームレンダリングにおいて用いる．

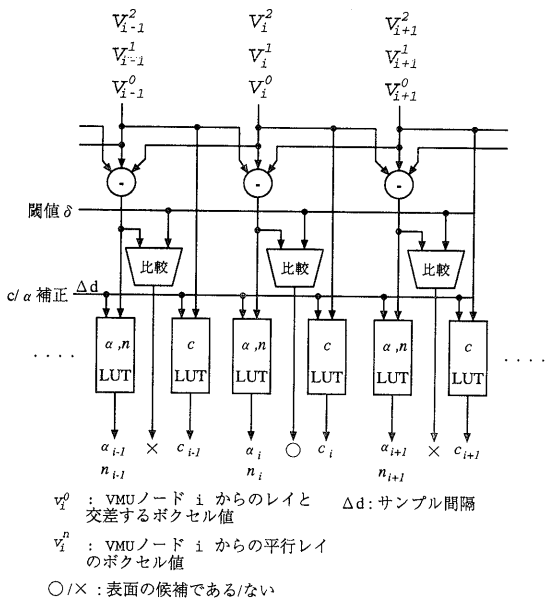


図6 c/α-LUT の構成
Fig. 6 Structure of c/α-LUT.

（2） 隣接するボクセル値の差から求める：隣接するボクセル値の差をインデックスとしてテーブル参照を行う．この方法は，ボクセル値の変化の大きな箇所を強調するために用いられる．

（3） 連続モデルの補間を行い補間した値から求める：連続モデルでのレンダリングに使用する．連続モデルでは，tri-liner 補間法[6]により値を求める．主軸を Z 軸，サンプル点の座標の小数部を (x_i, y_i)，補正に必要な頂点のボクセル値を $v_{(i,0)}, v_{(i,1)}, v_{(i,2)}, v_{(i,3)}$ とする(図5参照)．この時，ReVolver では式(2)で表される値をインデックスとしてテーブル参照を行う．

$$v_i=\{(1-x_i)*(1-y_i)-2x_i*y_i\}*v_{(i,0)} + x_i*(1-y_i)*v_{(i,1)} + (1-x_i)*y_i*v_{(i,2)}+x_i*y_i*v_{(i,3)} \quad (2)$$

従来の視線等間隔サンプリング法では，この補間にボクセルの8つの頂点の値が必要であるが，ReVolver での主軸等間隔サンプリング法では，ボクセルの一辺と同じ長さでサンプリングするので，ボクセルの側面に位置する4つのボクセル値でよい．したがって，メモリアクセスが少なく，VMU ノード間のデータの授受がないので，高速に処理できる．

また，c/α-LUT では，隣接ボクセル間の差を閾値 δ と比較し，レイトレーシングでの交差判定に必要な表面の候補を選び出す．これらの機能を持つ c/α-LUT

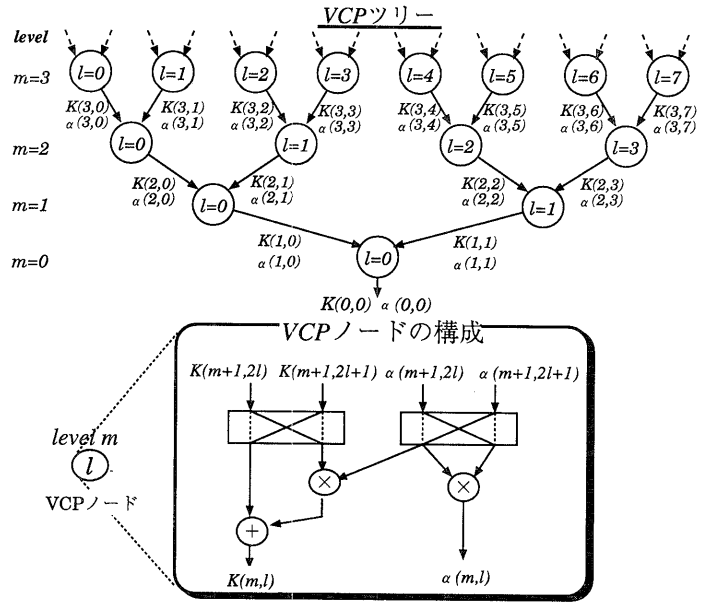


図7 ピクセル値計算ステージの構成
Fig. 7 Structure of pixel calculation stage.

の構成を図6に示す.

4.3 ピクセル値計算ステージの構成

ピクセル値計算ステージでは,c/α-LUT の出力から式(1)にしたがってピクセル値を計算する.いま,以下のように $K(m, l)$, $\alpha(m, l)$ を定義する.

$$K(m,l)=\sum_{i=\frac{nl}{2^m}}^{\frac{n(l+1)}{2^m}-1} K_i \prod_{j=\frac{nl}{2^m}}^{i-1} \alpha(v_j)$$

$$\alpha(m,l)=\prod_{i=\frac{nl}{2^m}}^{\frac{n(l+1)}{2^m}-1} \alpha(v_i) \qquad (3)$$

ただし, $m=0, 1, \cdots, \log_2(n-1)$, $l=0, 1, \cdots, 2m-1$ である.この時, $k(m, l)$, $\alpha(m, l)$ は以下のように表される.

$$K(m,l)=K(m-1,2l)+\alpha(m+1,2l)K(m+1,2l+1)$$

$$\alpha(m,l)=\alpha(m+1,2l)\alpha(m+1,2l+1) \qquad (4)$$

式(1)の P, K_i, $\alpha(v_i)$ はそれぞれ,式(3)を用いると $K(0,0)$, $K(\log_2 n, i)$, $\alpha(\log_2 n, i)$ と表される.したがって,ピクセル値 P を計算するには,式(4)を再帰的に実行し,$K(0,0)$ を求めればよいことが分かる.このため,式(4)の1回分の計算を行うVCPノードを用意する.このVCPノードを図7のようにツリー状に並べ,同じ深さのノードは並列動作させ,深さ方向にパイプライン処理を行う.このパイプラインをc/α-LUT からの色・透明度の出力サイクルと同じ時間で動作させることにより,ボクセルの読み出し1回分の時間でピクセル値の計算ができることになる.

すなわち,ReVolver では,このボクセルの読み出し時間がマシンサイクルとなっている.

4.4 ボリュームレンダリングの性能

ボリュームの大きさを 512^3,スクリーンの大きさを 512^2,ボリュームメモリを構成するDRAMのアクセスタイムを160 nsとする.離散モデルでのレンダリングでは,各VMUノードから1つのデータを読み出す時間でVCPツリーが動作するので,描画速度としては, $512^2 \times 160$ ns=42 ms/frame (~24画面/秒) となる.一方,連続モデルのレンダリングでは,各VMUから図5で黒丸をつけた4つのデータを読み出す時間でVCPツリーが動作する.データの読み出し時間は,縦の2つが連続アドレスであるため,DRAMのFast Page Modeでのアクセスを行う.ここでDRAMのサイクルタイムを80 nsとすると,4つの値にアクセスするのに $(160+80)\times 2=480$ ns かかる.このため,描画速度としては, $512^2 \times 480$ ns=125 ms/frame (~8画面/秒) となる.

5. レイトレーシングのサポート機構

レイトレーシングでは,レイが表面と交差すると,高次光を生成する.これをレイがボリューム空間の外に出るまで繰り返す.このため,レイの方向は任意となるが,任意の方向のレイに対してバンクコンフリクトせずにデータを読み出せるというReVolverの特長

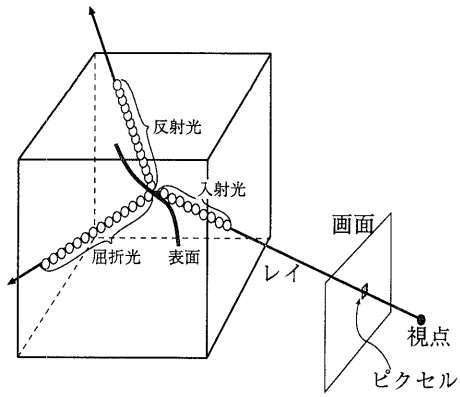


図8 ボリュームデータでのレイトレーシング
Fig. 8 Ray tracing with volume data.

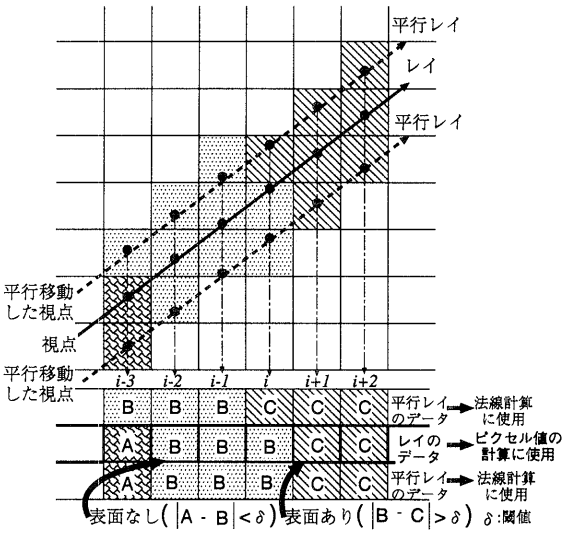


図9 レイと表面の交差判定
Fig. 9 Cross point determination.

を生かして，各レイを1マシンサイクルで処理することにより高速化を図る．ここで，図8にボリュームデータを用いた時のレイトレーシングの様子を示す．図8では入射光，反射光，屈折光のがそれぞれ1マシンサイクルで処理されるため，このピクセル値の計算には合計3マシンサイクルを要することとなる．以下では，ボリュームデータを用いてレイトレーシングを行う際に問題となる点を挙げ，その解決方法について述べる．

5.1 ボリュームデータによる交差判定

レイトレーシングでは，レイを追跡し表面との交差判定を行うが，ボリュームデータでは表面を決定しにくいという問題がある．そこで，ReVolver ではレイと交差するボクセル群の中で，隣接するボクセル値の差が閾値 δ 以上であり，その中で視点から最も近い場所に表面があると考える．この閾値δは，画像生成前にパラメータとして c/α-LUT に設定される．図9では，$i-2$，$i+1$ 番目でそれぞれ隣接するボクセル値と異なるが，$i-2$ では隣接ボクセル値間の差が閾値δ未満であり，$i+1$ では閾値δより大きいので，$i+1$ を表面とすることになる．この処理は c/α-LUT で隣接ボクセル間での差と閾値を比較し，表面となる位置の候補を選び，VCP ツリーを用いて視点から最も近い候補を選ぶことにより行われる．これは，ピクセル値計算と同時に行うことができるので，オーバヘッドとはならない．

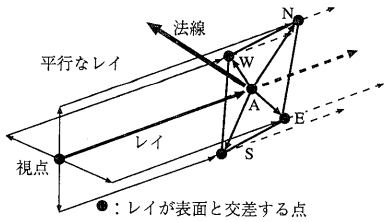


図10 法線計算
Fig. 10 Calculation of normal vector.

5.2 ボリュームデータによる法線計算

法線は高次光の方向を計算するのに必要である．従来の方法では，ボリュームを用いて法線計算をするのに，法線を求める点の周辺のボクセルに存在濃度を設定し法線の計算を行っていた[12]．しかし，この方法では，メモリバンクに対するアクセスが不均質となり，高速化を図れない．そこで，ReVolver では法線の計算をするのに，図9で示した平行レイを用い，平行レイとレイが表面と交差する位置から法線計算を行う．平行レイによりアクセスするボクセルは元のレイと同じメモリバンクに対して起きることから，メモリアクセスが均質となる．また平行レイに対する処理は交差判定のみであり，ピクセル値を求める処理と同時に行えるため，オーバヘッドにはならない．以下では，法線の計算方法について述べる．

平行レイとレイの表面との交差位置と法線の関係を図10に示す．図10での W，N，E，S は平行レイの交点であり，元のレイの交点を A とする．これらを用いると，法線は以下のようにして求められる．

$$\vec{N}=(\overrightarrow{AN}\times\overrightarrow{AW}+\overrightarrow{AW}\times\overrightarrow{AS}+\overrightarrow{AS}\times\overrightarrow{AE}+\overrightarrow{AE}\times\overrightarrow{AN})/4$$

（ただし，$\vec{A}\times\vec{B}$ は $\vec{A}$，$\vec{B}$ の外積を表す）

この法線計算については，平行レイとレイが表面と交差する点 W, N, E, S, A を求める処理は c/α-LUT とピクセル値計算ステージにより行い，法線ならびに高次光の方向計算はシェーディングステージを用いて

行う．

5.3 レイトレーシングでの性能

レイトレーシングの場合，レイの交差回数により性能が大きく左右されるので，最大3回の交差があると仮定して性能予測を行う．この場合，処理すべきレイの数は，通常に比べて最大で $(2^0+2^1+2^2+2^3)$ 倍となる．また，各VMUノードのメモリバンクでアクセスするデータは，図5で黒丸と白丸を付したボクセルとなるため，サンプリング点のボクセルの上下のボクセルはFast Page Modeを使用できるが，左右のボクセルはランダムアクセスとなる．したがって，5つの値にアクセスするのに $(160\times3+80\times2)$ ns $=640$ ns かかり，VCPツリーはこの時間で動作する．よって，描画時間は最大で $512^2\times15\times640$ ns $=2520$ ms となる．

6. 関連研究

6.1 専用ハードウェアの例

既存の並列計算機の概要について**表1**にまとめる．これらの並列計算機の中で，CUBEはReVolverと同様に後方投影アルゴリズムをボクセル並列により処理を行う計算機であるので，以下ではCUBEとの比較を行う．

CUBEでは，視線方向を $\pm x$，$\pm y$，$\pm z$ の6通りに制限し，これらの方向からの平行投影のみしか行わない．n をボリューム空間の1辺として，(p, q, r) のボクセルを以下の式(5)で定められる番号のメモリバンクに格納する．したがって，視線が軸と平行な場合は n ボクセルを同時に読み出すことができる．

$$k=(p+q+r) \bmod n \qquad (5)$$

しかし，視線が軸と平行にならない場合，ボリュームメモリの内容を回転してからでないと画像生成が行えない．そのため表1に示すように，投影は62 ms (～16 frames/sec) であるのに対し，回転を伴うと，573 ms (～1.7 frames/sec) 程度に性能低下する．これに対して，ReVolverでは視線の方向は任意であり，平行投影だけでなく遠近法を用いても描画性能は，離散モデル時で42 msと一定であるため，インタラクティブな使用でも問題ない．

6.2 汎用の並列計算機上でのボリュームレイトレーシング

ボリュームデータを用いたレイトレーシングは，現在のところ汎用の並列計算機を用いてソフトにより行われており，アーキテクチャでサポートしているものはない．そこで，汎用の並列計算機上でソフトによるボリュームレイトレーシングの例として参考文献2)で示されている方法との比較を行う．

参考文献2)で示されている方法は，ボリューム並列処理によって，ボリュームデータを用いたレイトレーシングを行うものである．つまり，ボリューム空間を幾つかのサブボリュームに分割し，サブボリュームごとにプロセッサが画像生成を行った後，これらの画像をマージして，最終的な画像を得る．このアルゴリズムを用いて，CM-5上で比較的密なボリュームデータをレンダリングするのに要した時間を**表2**に示す[2]．

しかし，ReVolverでは最大3回の反射屈折を繰り返す場合でも最大2.52 sec/frameと512台構成のCM-5を上回っており，レイトレーシング法でも高速であることが分かる．

7. おわりに

本稿では，ボリュームレンダリングを高速に実行する専用並列計算機ReVolverのアーキテクチャの提案を行い，ボクセル値生成ステージ，ピクセル値計算ステージの構成を示した．

ReVolverでは，任意の方向の視線と交差するボク

表2 CM-5でのレンダリング時間
Table 2 Rendering time on CM-5.

プロセッサ数	64	128	256	512
時間 (sec)	24.430	12.697	6.3434	3.1878

表1 既存のボリュームレンダリング専用並列計算機の概要
Table 1 Abstract of existing computers for Volume Rendering.

名称	アルゴリズム	並列処理方式	ボリューム	半透明表示/遠近法	速度
GODPA	前方投影	ボリューム並列	256^3	×/×	20～30画像/秒
SCOPE	後方投影	ボリューム並列	—	−/○	—
3 DP[4]	前方投影	ボリューム並列	$64^3\times256$	×/○	10画面/秒
PARCUM II	後方投影	ピクセル並列†	512^3	×/×	—
CUBE	後方投影	ボクセル並列	512^3	△/△††	投影62 ms/回転511 ms
ReVolver	後方投影	ボクセル並列	512^3	○/○	投影42 ms

† 4^3 voxel単位で読み書きを行う　†† リアルタイムでは不可

セルをバンクコンフリクトなしで読み出すことのできるメモリ構造を実現し，このメモリ構造に見合う性能を出すために，ピクセル値計算部では演算器をツリー状に配置し，パイプライン動作させることによりメモリサイクルでピクセル値を計算することとした．これにより，約 42 ms/frame 程度の性能が期待できる．

さらに，ReVolver では高品位な表示を行う機構として，(1)連続モデルによる表示，(2)レイトレーシングによる画像生成，を行う．連続モデルでの表示については，主軸等間隔サンプリング法に基づく線形補間の方法を示した．また，レイトレーシングでは，レイと表面の交差判定をピクセル値計算と同時に行えることと，視線の平行移動によって表面の法線ベクトルを求める方法を示した．この連続モデルでの描画性能は約 125 ms/frame で，レイトレーシングでは，約 2520 ms/frame 程度の性能が予測される．

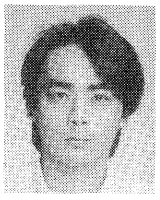
今後の課題としては，各部について詳細な検討を行い，プロトタイプマシン[13]の作成を行う予定である．

謝辞 日頃御討論いただく京都大学工学部情報工学教室富田研究室の諸氏に感謝いたします．本研究の一部は文部省科学研究費（一般研究(A)課題番号 06402057「3 次元メモリによる実時間可視化機構を内蔵した柔構造スーパコンピュータの構成方式」）による．

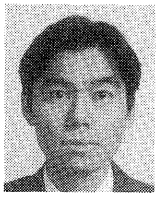
参 考 文 献

1) 明石英也：ボリュームレンダリング向き並列計算機のアーキテクチャ，京都大学工学部情報工学科 修士論文 (1993).
2) Ma, K.-L., Painter, J. S., Hansen, C. D. and Krogh, M. F.: A Data Distributed, Parallel Algorithm for Ray-Traced Volume Rendering, *Parallel Rendering Symposium,* pp. 15 - 22 (1993).
3) Goldwasser, S. M.: A Generalized Object Display Processor Architecture, *IEEE Computer Graphics & Applications,* Vol. 4, No. 10, pp. 43-55 (1984).
4) Jackel, D. and Strasser, W.: Reconstructing Solids from Tomographic Scans—The PARCUM II System, *Advances in Computer Graphics Hardware II,* pp. 209-227 (1988).
5) Kaufman, A. and Bakalash, R.: Memory and Processing Architecture for 3D Based Imagery, *IEEE Computer Graphics & Applications,* Vol. 8, No. 6, pp. 10-23 (1988).
6) Levoy, M.: Display of Surfaces from Volume Data, *IEEE Computer Graphics & Applications,* Vol. 8, No. 5, pp. 29-37 (1988).
7) Ohashi, T., Uchiki, T. and Tokoro, M.: A Three-Dimensional Shaded Display Method for Voxel-Based Representation, *Eurographics 85,* pp. 221-232 (1988).
8) Sabella, P.: A Rendering Algorithm for Visualizing 3D Scalar Fields, *Computer Graphics,* Vol. 22, No. 4, pp. 51-58 (1988).
9) 内木哲也，所真理雄：3 次元メモリを用いた立体図形表示機構—SCOPE—，電子通信学会論文誌，Vol. J 68-D, No. 4, pp. 741-748 (1985).
10) Blinn, J. F.: Light Reflection Functions for Simulation of Clouds and Dusty Surfaces, *Computer Graphics,* Vol. 16, No. 3, pp. 21-29 (1982).
11) Kajiya, J. T. and von Herzen, B. P.: Ray Tracing Volume Densities, *Computer Graphics,* Vol. 18, No. 3, pp. 165-175 (1984).
12) 田山典男，清水則明，漆間文俊：可変歩幅 DDA による 3 次元メモリ空間での高速なボクセル追跡，情報処理学会論文誌，Vol. 32, No. 5, pp. 581-589 (1991).
13) 對馬雄次，金 喜都ほか：ボリュームレンダリング専用並列計算機 ReVolver/C 40 のアーキテクチャ，情報処理学会研究会報告 94-ARC-107, pp. 137-144 (1994).

(平成 6 年 9 月 19 日受付)
(平成 7 年 1 月 12 日採録)



對馬 雄次（学生会員）
昭和 43 年生．平成 5 年京都大学工学部情報工学科卒業．平成 7 年同大学院修士課程修了．現在，(株)日立製作所に勤務．

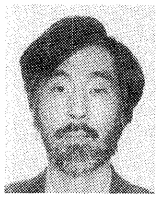


明石 英也（正会員）
1968 年生．1991 年京都大学工学部情報工学科卒業．1993 年同大学院修士課程修了．同年 4 月より(株)日立製作所中央研究所に勤務．並列計算機の研究開発に従事．

金 喜都
1958 年生．1982 年ハルピン科学技術大学計算機工学科卒業．1988 年ハルピン工業大学計算機工学専攻修士課程修了．1993 年京都大学大学院工学研究科情報工学専攻博士後期課程入学．現在，グラフィックス専用マシンの研究に従事．

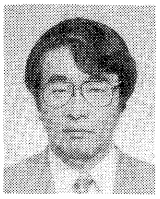
森　眞一郎（正会員）

1963年生．1987年熊本大学工学部電子工学科卒業．1989年九州大学大学院総合理工学研究科情報システム学専攻修士課程修了．1992年同大学院博士後期課程単位取得退学．同年京都大学工学部情報工学教室助手，現在に至る．並列処理，計算機アーキテクチャの研究に従事．IEEE-CS，ACM各会員．



中島　　浩（正会員）

昭和31年生．昭和56年京都大学大学院工学研究科情報工学専攻修士課程修了．同年三菱電機(株)入社．推論マシンの研究開発に従事．平成4年より京都大学工学部助教授．並列計算機のアーキテクチャ，プログラミング言語の実行方式に関する研究に従事．工学博士．昭和63年元岡賞，平成5年坂井記念特別賞受賞．



富田　眞治（正会員）

1945年生．1968年京都大学工学部電子工学科卒業．1973年同大学院博士課程修了．工学博士．同年京都大学工学部情報工学教室助手．1978年同助教授．1986年九州大学大学院総合理工学研究科教授，1991年京都大学工学部情報工学科教授，現在に至る．計算機アーキテクチャ，並列処理システムなどに興味を持つ．著書「並列計算機構成論」「計算機システム工学」「並列処理マシン」など．電子情報通信学会，IEEE，ACM各会員．