視角制限ピクセル並列処理によるボリューム・レンダリング向きの超高速専用計算機のアーキテクチャ

金 喜都[†,☆] 對馬雄次[†,☆☆] 中山明則[†,☆☆☆]
森眞一郎[†] 中島浩[†] 富田眞治[†]

本稿では，512^3 ボクセル集合に対する半透明ボリューム表示，遠近法による投影，超高速実時間画像生成を目的とした，ボリュームレンダリング専用並列計算機のアーキテクチャを提案する．本計算機では，妥当な角度内に視角を制限し，3次元ボリューム空間を視覚化することによって，表示図形歪みを抑えるとともに，安価に3次元画像メモリを構成できる特徴を有している．ボリューム空間は，メモリバンク群に対応する512枚のスライス群に写像され，スクリーンの1行分のピクセルを通る512本の視線がパイプライン的にボリューム空間に並列アクセスできる．512個のプロセッサは1次元に接続され，マクロパイプライン的に動作する．パイプラインサイクル ΔT は80 nsであり，1秒間に48枚の動画が出力できる．

Super-high Speed Volumc Rendering Architecture Based on Pixel Parallelism with the Restrained Visual Angle

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

This paper presents a super-high speed ray casting system for volume rendering of 512^3 volumetric data. In order to realize real-time image generation from a translucent volume by the parallel or perspective projection, this system restricts the view angle within a certain range such that no distortion will be appeared on the generated image. In this system, volume data is divided into 512 slices and the slices are stored in the conflict free memory of 512 banks associated with differenent slices each other. Thus, the memory has an ability of simultaneous memory accesses for 512 rays which go though 512 pixels along a scan line. In the system, one dimensional processor array with 512 processors corresponding to 512 banks of the memory operate in a macro pipeline fashion. Provided with the pipeline cycle of 80 ns, this system processes 512 rays every 512 cycles, resulting 48 frames generation per second.

1. はじめに

医療画像や科学技術計算結果の可視化による解析手法として，ボリュームレンダリングが注目を集めている．サーフェスレンダリングと異なって，ボリュームレンダリングの目的は，通常は見ることのできないボリュームオブジェクトの内部を探索し，従来は解析が困難であった複雑な構造や振舞いを理解するための視覚的な洞察手法を提供することにある．また，それはテクスチャマッピングなどきめの細かい映像表現が容易に行いうる特徴も有している．

しかし，ボリュームレンダリングは3次元情報を3次元配列として直接扱うため，膨大な計算資源を必要とし，高速な画像生成が困難であった．このため，並列処理が必要不可欠となる．

本稿では，512^3 ボクセル集合に対する半透明なボリューム表示，遠近法による投影，視点移動にともなう高速な画像生成，時間軸に沿う連続的な現象表示のための動画作成を目的とした，ピクセル並列レイキャスティングが超高速に行える専用並列計算機のアーキテクチャを提案する．

高速ボリュームレンダリングでは視線に沿ったボクセルへの並列（パイプライン）アクセスが本質的に

† 京都大学工学研究科
Faculty of Engineering, Kyoto University
☆ 現在，ウッドランド株式会社
Presently with Woodland, Ltd.
☆☆ 現在，株式会社日立製作所
Presently with Hitachi, Ltd.
☆☆☆ 現在，NTTデータ通信株式会社
Presently with NTT DATA Corporation

重要であり，3次元構造のコンフリクトフリーなメモリを必要とする．先に我々は主軸等間隔サンプリングに基づいた3重構造メモリによるReVolver/C40を提案した．この方式はコンフリクトフリーなメモリを構成するのに3次元メモリを3重化する必要があり，メモリ容量上問題があった．本稿での方式は，視角の上限を実用上問題とならない範囲に制限することで，1重構造のコンフリクトフリーなメモリを構成できる特徴がある．この方式を視角制限ピクセル並列法（Pixel Parallelism with the Restrained Visual Angle: PPRVA）と呼ぶことにする．本稿では，PPRVAの原理およびマクロパイプラインを用いたシステム構成法について述べる．

2. ボリュームレンダリング法

本章では以下の章の理解に必要となるボリュームレンダリングに関する基本的な事項について簡単に説明する．

2.1 ボリューム空間とスクリーン空間

ボリュームレンダリングとは，ボクセルと呼ばれる単位立方体集合の格子構造で構成されている3次元の空間内の内部情報を，2次元のスクリーン空間に投影することにより，オブジェクトの内部構造も可視化の対象となるように，3次元空間を可視化する手法である．この3次元空間をボリューム空間と呼ぶ．

3次元格子状のボリューム空間は n^3 ボクセルを持つものとする．それを3次元直交座標系 O_{-LAB} 上のL，A，B軸の $[0, n)$ 半開区間上に定義し，各ボクセルの8つの頂点を3次元座標系の整数座標点上に置く．

また，対応したスクリーン空間は n^2 ピクセルを持つものとする．

2.2 レイキャスティング法とサンプリング手法

レイキャスティング（ray casting）法はボリューム空間の中身を覗き，半透明なボリューム表示を行う手法である．アルゴリズムとしては，視点からスクリーンの各ピクセルに対して視線を発生し，その視線に沿って不透明度が1になるまで光を吸収するボリューム内を前進する．そして，最終的にはそれまでの累積カラー値を求めてそのピクセルの色とする．

視線上のボクセル値のサンプリングには2つの方式がある．

(i) 主軸等間隔サンプリング

主軸等間隔サンプリング法では，**図1**に示すように視線上のボクセルのサンプリングを，当該視線の方向ベクトルの成分の絶対値が最大となる座標軸（以後主軸と呼ぶ）上で，等間隔（ボクセル間の距離）で行うようにする．すなわち，あるピクセルを通る視線 Ray_i $(i = 1 \sim n^2)$ 上のボクセル値のサンプリング間隔 $|\overrightarrow{\Delta D_i}|$ は次の式(1)によって決まる．

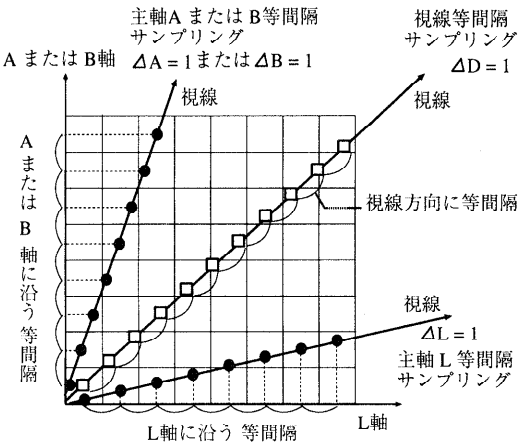


図1 主軸等間隔と視線等間隔のサンプリングの比較
Fig. 1 Comparison of the sampling scheme.

$$|\overrightarrow{\Delta D_i}| = \sqrt{\Delta L_i{}^2 + \Delta A_i{}^2 + \Delta B_i{}^2} \qquad (1)$$

ただし，$0 \le |\Delta L_i|, |\Delta A_i|, |\Delta B_i| \le 1$ かつ

$\exists \Delta V \in \{|\Delta L_i|, |\Delta A_i|, |\Delta B_i|\}$, $\Delta V = 1$.

ここで，ΔV は主軸サンプリング間隔でもあり，ボクセル（Voxel）の稜線の長さでもある．

ゆえに，$1 \le |\overrightarrow{\Delta D_i}| \le \sqrt{3}$

(ii) 視線等間隔サンプリング

一方，視線等間隔サンプリング法では，視線方向に沿って等間隔にサンプリングを行うので，$\Delta D_i = 1$ である．

本システムでは，主軸等間隔サンプリングを採用した．それは次の理由による．

(1) コンフリクトフリーなメモリ構成が容易であること

後述のように，ボリューム空間をメモリバンク群に対応するスライス群に写像することで，ボクセル並列アクセスによるバンクコンフリクトをなくすメモリ構成を容易に達成できる．

(2) パイプライン構成が容易であること

メモリより最大n個のボクセル値が出力され，nステージのパイプラインで処理できる．視線等間隔サンプリングではパイプラインステージ数は長くなり，$\sqrt{3} \times n$ となる．

(3) ピクセル値補正は容易であること

主軸等間隔サンプリング法は，軸と平行でない視線が通常の視線等間隔サンプリング法に比べると，その

サンプリング間隔は最大で $\sqrt{3}$ 倍となるため，視線ごとにピクセル値の補正が必要となる．しかし，この補正は容易に行いうる．

2.3 ピクセル値の算出方式

主軸等間隔サンプリングに基づくレイキャスティング法によるピクセル値Pの計算式は，視線が通るボクセル v_i を，視点から近い順に $i = 0, 1, \ldots, n-1$ とするとき，以下のように表現される．

$$P = \sum_{i=0}^{n-1} K(v_i) \prod_{j=0}^{i-1} \alpha(v_j) \qquad (2)$$

$c(v_i)$，$\alpha(v_i)$ は，ボクセル値 v_i に対応する色，透明度であり，$K(v_i)$ は，$c(v_i)(1-\alpha(v_i))$ を表している．ただし，$K(v_i) = c(v_i)$ とする場合もある．上の算式 (2) より

$$\begin{cases} \alpha_m = \prod_{i=0}^{m} \alpha(v_i) \\ K_m = \sum_{i=0}^{m} K(v_i) \times \alpha_{m-1} \end{cases}$$

とおくと，式 (2) は，$\alpha_{-1} = 1$，$K_{-1} = 0$ を初期値として

$$\begin{cases} \alpha_j = \alpha_{j-1} \times \alpha(v_j) \\ K_j = K_{j-1} + K(v_j) \times \alpha_{j-1} \end{cases} \qquad (3)$$
$$(j = 0 \sim n-1)$$

と漸化式で書き下せる．また，最終結果の K_{n-1} はピクセル値Pである．

したがって，視線 Ray_i $(i = 1 \sim n^2)$ のピクセル値は，2.2 節の式 (1) により，ボリューム空間への入射点座標 (L_{in}, A_{in}, B_{in}) から視線の方向に追跡し，$j = 0, 1, \ldots, n-1$ の順によって，点 $(L_{in} + j \times \Delta L, A_{in} + j \times \Delta A, B_{in} + j \times \Delta B)$ のボクセル値 v_j に式 (3) を繰り返して行うことで算出される．

2.4 ピクセル値の補正方法

式 (3) に使う補正後の $K'(v_j)$，$\alpha'(v_j)$ は，$c'(v_j) = c(v_j) \times \alpha(v_j)^{\Delta D - 1}$，$\alpha'(v_j) = \alpha(v_j)^{\Delta D}$ で近似計算をすると，PEへの負担が大きいので，スピードが落ちることになる．したがって，補正を高速に行うため，その代わりに，ΔD $(1 \le \Delta D \le \sqrt{3})$ の値によって，K と α の LUT（Look Up Table）を拡張する方法で求めることにする．

α の拡張 LUT テーブルの構成は**表 1** のようになっている．

表 1 では，$\Delta D = 1.d_{-1}d_{-2}\cdots d_{-m}$，$(\overbrace{1.00\cdots0}^{m+1} \le \Delta D \le \overbrace{1.11\cdots1}^{m+1}, m \ge 2)$ と仮定すると，$(\Delta D - 1) \times 2^m$ を α の拡張 LUT の下位桁アドレス（テーブルの行インデックス），$v_i \times 2^{m+1}$ を上位桁アドレス（テーブルの列インデックス）として，補正後の $\alpha'(v_j)$ を高速に求めることができる．K の拡張 LUT テーブルの構成も同様である．

表 1 ΔD と Voxel 値による α の拡張 Look Up Table
Table 1 Extpanded α Look Up Table based on ΔD and voxel values.

$d_{-1}d_{-2}\cdots d_{-m+1}d_{-m}$	Voxel 値 0	1	⋯	255
0 0 ⋯ 0 0	$\alpha_{1,1}$	$\alpha_{1,2}$	⋯	$\alpha_{1,256}$
0 0 ⋯ 0 1	$\alpha_{2,1}$	$\alpha_{2,2}$	⋯	$\alpha_{2,256}$
0 0 ⋯ 1 0	$\alpha_{3,1}$	$\alpha_{3,2}$	⋯	$\alpha_{3,256}$
0 0 ⋯ 1 1	$\alpha_{4,1}$	$\alpha_{4,2}$	⋯	$\alpha_{4,255}$
⋮	⋮	⋮	⋮	⋮
1 1 ⋯ 1 1	$\alpha_{2^m,1}$	$\alpha_{2^m,2}$	⋯	$\alpha_{2^m,256}$

3. 視角制限ピクセル並列処理

本システムでは視点から見たボリューム空間への視角の大きさに制限が設けられている．この視角制限によって，ボリューム空間への並列アクセスを可能とするコンフリクトフリーな3次元メモリを安価に構築できる．本章では視角制限ピクセル並列処理の原理と諸性質について述べ，3次元メモリの構成を示す．

3.1 視角制限の妥当性

視点が立方体ボリューム空間の対角頂点の延長線にある場合（視角が最大になる場合），本計算機のピクセル並列処理で生成された画像の視角は本計算機のメモリ構成によって 54° **準広角**という最小視角以内に制限される．

このように視角を制限することによって，後述のようにコンフリクトフリーな3次元メモリを構成できる．視角制限が有効であるのは，通常の科学技術計算結果の可視化において広角表示が必要とされないためである．すなわち，広角的な視角による平面スクリーンへの映像には次に述べる2つの欠点があり，可視化の目的に反するからである．

(1) 広角映像の欠点1

広角的な視角で，平面スクリーンの各ピクセルに対してレイキャスティングをすると，ボリュームの中央部を通過する視線の数が広角になるほど少なくなる．逆に，ボリュームの周辺空間を通る視線の数が広角になるほど多くなる．したがって，**ボリューム中央部の情報欠落**の現象が生じる．す

なわち，最も見たい中央部の情報が欠落する．

(2) 広角映像の欠点2

広角的な視角で生成された**歪みの映像**は，人間が対象の直接見えない内部構造をはっきり見分ける（認識する）ことを妨げる．

平面スクリーン映像の視覚特性は

- 標準視角 40° まででは，変形しにくい
- 視角 23° では，肉眼視覚に近くきわめて自然な視覚効果表現が得られる

といわれているため，物理的なメモリ構成から導入された 54° 視角制限は通常の可視化を考えるうえで十分大きな角度となっている．視角制限が 54° となる理由については 3.3.1 項で述べる．

3.2 ボリューム空間と観察空間

本節では以降の視角制限ピクセル並列処理に必要となる諸用語を定義しておこう．

3.2.1 ボリューム空間からメモリ空間への写像

ボリューム空間を，**図2** のように，L 軸の整数座標点 $1,2,\ldots,n-1$ に沿って，L 軸に垂直な n 個のスライス L_i $(i=0\sim n-1)$ に薄く切り，L 軸上の**スライス群**とする．L 軸上の i $(i=0\sim n-1)$ 番目スライス L_i の厚さは，該当軸上の半閉区間 $[i,i+1)$ に対応する．ここで，各スライスに対しては L 軸に垂直な面は**スライスの断面**，L 軸に平行な面は**スライスの側面**，スライス群に対しては両側にある L 軸に垂直な表面は**スライス群の底面**，あと 4 つの表面は**スライス群の側面**，底面上にない稜は**側稜**と名付ける．

物理的な実装では，ボリューム空間はスライス群に対応したメモリバンク群に格納される．

3.2.2 スライス群を囲んだ視点空間の区分

視点からスクリーンを通してスライス群空間を見る場合，スライス群の両底面からの投影がスクリーン上で重なる場合，その視点空間を**スライス群の底面観察空間**，それ以外の視点空間を**スライス群の側面観察空間**と呼ぶ．

図3 に示すようにスライス群の底面観察空間はスライス群の中心を通る 4 つの対角線の延長線で限られた，スライス群の底面を含む上下 2 つの陰影囲いの四角錐空間である．それ以外の空間はスライス群の側面観察空間である．

明らかに，視点がスライス群の側面観察空間にある場合，スライス群の両底面を通る視線はない．

3.2.3 視点の移動

3 次元空間でボリューム周りに視点を移動することは，**図4** のように静止の視点座標系 O_{-XYZ} 空間では，ボリューム空間内に固定されたボディ座標系 $O_{-L_0A_0B_0}$ を回転・平行移動して表現される．

(1) **ボディ座標系の定義**：図4 の (a) に示すように，ボリューム空間の中心にボディ座標系の原点を置き，ボディ座標系の L_0，A_0，B_0 軸を各々 L，A，B 軸に平行かつ同方向に設定する．

(2) **ボディ座標系の視点座標系への配置**：最初の位置として，ボディ座標系の原点を視点座標系の $(0,0,Z_0)$ に置き，L_0，A_0，B_0 軸が各々 X，Y，Z 軸に平行・同方向に設置される．

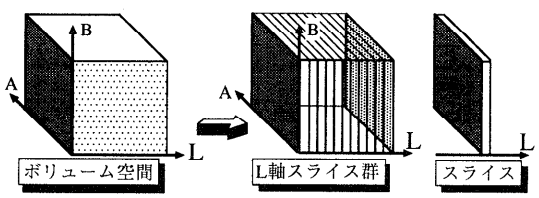


図2 L 軸に沿ったボリューム空間の分割
Fig. 2 Volume space slicing on L axis.

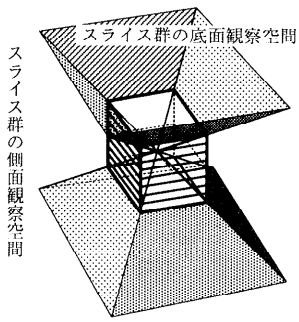


図3 視点空間の区分
Fig. 3 Dividing space based on the position of a viewpoint.

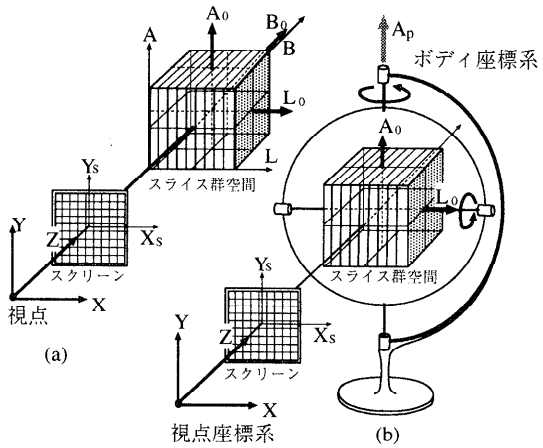


図4 視点座標系および視点座標系でのボディ座標系の回転
Fig. 4 The rotation of the body coordinate system in the viewpoint coodinate system.

(3) **ボディ座標系の回転移動**：ボリューム空間の中に固定されたボディ座標系がそれ自身の原点回りに任意に回転することは各々 L_0, A_0, B_0 軸回りの回転より合成される．しかし，座標変換の複雑さを減少し，視線の計算を超高速に行うため，図4の(b)に示すように，ボリューム空間が $\mathbf{A_P}$, $\mathbf{L_0}$ 2軸回りだけの回転を行うことにする．ただし，A_P 軸は視点座標系の中に，$(0,0,Z_0)$ を経て，Y軸に平行なものである．

(4) **ボディ座標系の平行移動**：視点座標系のZ軸に沿って，ボディ座標系を平行移動してよい．

(5) **ボディ座標系の回転特徴**：図4の(b)の回転方式から見ると，L_0 軸がZ軸に重畳していない場合，

(a) スライス群の各断面に垂直な L_0 軸のスクリーンへの投影がスクリーン座標系の X_S 軸に重なる．

(b) スライス群の各断面に垂直，L_0 軸に平行な4側稜のスクリーンへの正射影が X_S 軸に平行である．

3.3 視角制限ピクセル並列処理の諸性質

3.3.1 視角制限ピクセル並列処理の前提条件

視角制限ピクセル並列処理には次項で述べる諸性質がある．この方式では，以下の点を前提条件としている．

前提条件1 ボリューム空間およびその表面ボクセルについて：

(1) 全ボリューム空間がスクリーン領域の内に一度に投影可能なものとする．すなわち，全ボリューム空間が1画面ですべて表示可能である．

(2) ボリューム表面の各ボクセルがスクリーンへ投影される際には，その水平投影幅はスクリーンのピクセル幅を超えないようにする．これは54°準広角を超えない視角で保証できる．

ここで，視角制限が54°になっている理由を述べる．**図5**はボリュームの対角頂点の延長線上にある視点からボリュームを見た場合を示している．図5の(a)，(c)に示すように，視角が54°までの場合には，各スライスの側面に入射される視線の数はたかだか1個である．すなわち，図5の(a)，(c)に示す視線の入射間隔 $\Delta\lambda$（隣接入射点のスライスの断面に垂直な方向上の間隔）はスライスの厚さより大きい．図5の(a)ではスライス番号1，2，3，5，8にそれぞれ視線①，②，③，④，⑤が1つずつ入射している．図5の(b)に示すように，視角が54°を超えると，視点に近いスライスの側面には多数の視線が入射される．図5の(b)では，第1スライスに2本の視線（①と②）が入射している．視角が54°を超えるにつれてスライス番号の若い順にメモリコンフリクトが生じてくる（スライス群の底面への入射の場合について後述する）．

このように各スライスをメモリバンクに対応された3次元メモリにおいてコンフリクトフリーを実現する条件が視角54°となる．

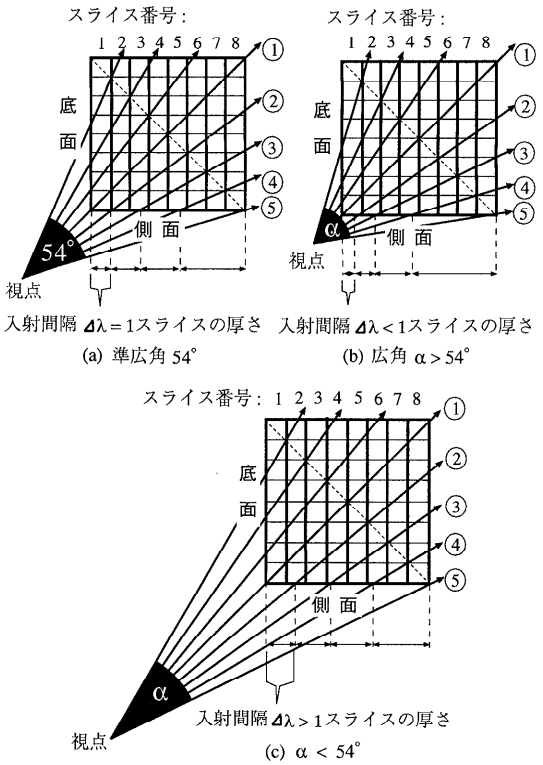


図5 視角制限の妥当性
Fig. 5 The reasonableness of restrained visual angle.

前提条件2 全スクリーン分の視線のスライス群への入射方法：視線群ずつ一度に投入する．ここで視線群とは，スクリーンの1行分に対応する視線の集まりである．

前提条件3 視線群のスライス群への入射方法：

(1) スライス群の**側面**に至る複数本の視線はそれらが交差した各スライスに**同時**に入射させる；

(2) スライス群の**底面**に至る複数本の視線は1つずつ**順番**に底面より入射させる．すなわ

ち，先行の視線が1つスライスを通過するごとに後続の視線を入射させる．

前提条件4 視線がボクセルを通過する時間間隔 ΔT: 前提条件3でスライス群に入射された各視線はΔTごとに次々にその視線上のボクセルを処理する．

3.3.2 視角制限ピクセル並列処理の諸性質

上記を前提とすると，視角制限ピクセル並列処理は以下の性質を満たす．

性質1 異なる視線が同一スライスの側面に入射することはない．

性質2 異なる視線の処理が同時に同一スライスでなされることはない．

性質3 視点がスライス群の側面観察空間にあるならば，視線群は n×ΔT 時間でスライス群を通過する．したがって，n×ΔT ごとに新しい視線群がスライス群に投入できる．

性質4 視点がスライス群の底面観察空間にあるならば，各視線群がスライス群を通るレイテンシ時間は 2×n×ΔT を超えない．このレイテンシは n×ΔT の視線群入射時間とスライス群での通過時間 n×ΔT に分割でき，パイプライン化できる．したがって，n×ΔT ごとに新しい視線群がスライス群に投入できる．

3.3.3 視角制限ピクセル並列処理の諸性質の証明

3.3.3.1 性質1について

これは先に述べた視角制限の理由から明白である．

3.3.3.2 性質2について

図6に示すように，スライス側面に入射された視線は同期してスライス群を通過する．入射地点で異なるスライスに入射されるので，平行視線，放射視線とも同一時刻に同一スライスに2つ以上の視線が存在することはない．底面に入射される視線は図6(b)のように先行視線が1つのスライスを通過するたびに後続の視線が投入される．図6(b)では，視線①がスライス番号1で3回処理された後，スライス番号2に移る．このとき視線②がスライス番号1に入射される．したがって，同一時刻には同一スライスに2つの視線は存在しない．

3.3.3.3 性質3について

3.2.2項により，スライス群の側面観察空間の視点位置による視線群は，スライス群の両底面を通ることはない．

このとき，視線群はすべてがスライス群の側面に入射する場合もあるし，スライス群の側面と底面に入射する場合もある．したがって，図6に示すように実スライス群の底面の外側に若干の仮想スライス（仮想スライスのボクセルに対応する透明度を1，色を0とする）を加え，スライス群の底面に至る視線を，仮想スライスの側面ごとに入射させるようにする．この拡張されたスライス群を**拡張スライス群**と呼ぶ．底面に入射される視線を仮想スライスの側面に入射されるものと見なせば，視線群は n×ΔT でスライス群を通過することが分かる．

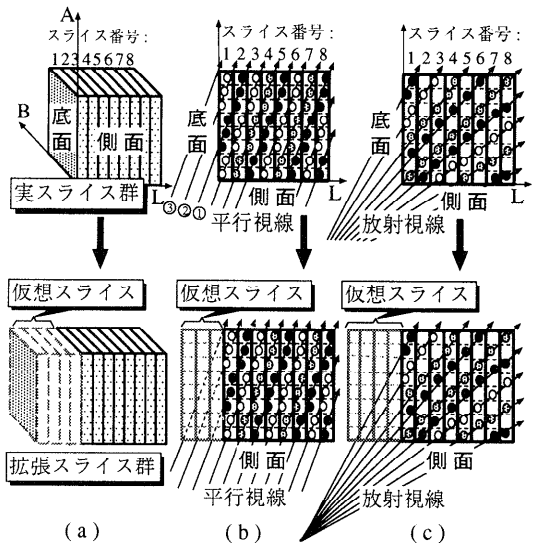


図6 底面に至る視線を仮想スライスに同時入射
Fig. 6 Simultaneous incidences of rays by introducing the virtual slice.

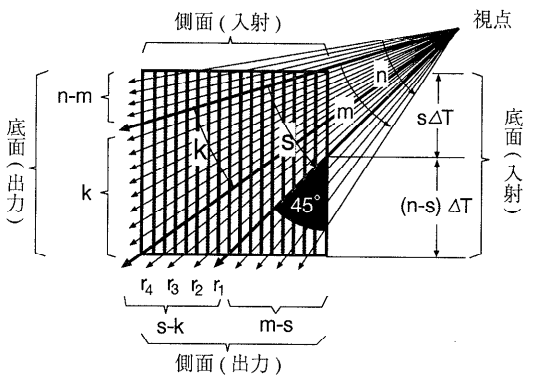


図7 視点がスライス群の底面観察空間にある場合
Fig. 7 The incidences of rays when the viewpoint is in the base observation space.

3.3.3.4 性質4について

底面観察空間の視点位置による視線群は2底面を通過するものがある．このとき，視線群の半分以上はスライス群の底面に入射される．視線群がスライス群を通る様子を考えよう．仮に**図7**に示すように，

(1) 視線群の m 本はスライス群の底面に入射される．このうち，
  (a) s 本の主軸が各断面に垂直である．s 本の中に，
    (i) k 本はスライス群の両底面を貫く
    (ii) s−k 本はスライス群の側面を出る
  (b) m−s 本の主軸がスライス群の断面に平行である
(2) n−m 本はスライス群の側面に入射されるとする．

(a) 実スライス群の側面へ至る n−m 本の視線は同時に入射され，それらの主軸が各断面に垂直となるので（$\Delta L = 1$），必ず $n \times \Delta T$ までにスライス群の底面を出る．

(b) 次に，スライス群の両底面を通過する k 本の視線は，$\Delta L = 1$ なので，ΔT に 1 本ずつ入射できる．$k \times \Delta T$ 後に入射完了し，以後 $n \times \Delta T$ 時間必要であるので全体で，$(n+k) \times \Delta T$ までにスライス群の底面を出る．

(c) 側面から出力する s−k 本の視線も $\Delta L = 1$ なので，$(s-k) \times \Delta T$ で入射完了し，また先行の視線（図の r_4, r_3, r_2, r_1 の順）がスライス群を通過する時間間隔は後続の視線より少なくとも ΔT だけ多いので，後続の視線が先行の視線よりスライス群を遅く出ることはない．すなわち，入射底面に近いもの（図 7 の r_1, r_2, r_3, r_4 の順）から先に出る．したがって，これらの視線は $(n+k) \times \Delta T$ までにスライス群を出る（r_4 が出力される時間）．

(d) 上記の (a) と（(b), (c)）は同時に処理される（(b) と (c) は逐次に処理）ので，$s \times \Delta T$ で $s+(n-m)$ 本の視線が入射されたことになる．残った m−s 本は主軸がスライス群の断面に平行であり，その通過距離が図 7 のように $(n-s)$ であるので，前述の 3.3.3.2 項と 3.3.3.3 項の場合と同じように $(n-s) \times \Delta T$ までにスライス群を通過する．この時間は図 7 の r_1 が出力される時間と同一となる．

以上より，底面観察空間の場合には，視線群は最大で $n \times \Delta T$ で入射完了し，さらに，その後，$n \times \Delta T$ でスライス群を通過する．

3.4 バンクコンフリクトフリーなメモリ構成

3.2.1 項よりスライス群はボリューム空間を格納しておくメモリバンク群である．したがって，3.3.2 項の**性質 1, 2, 3, 4** によって，視角制限ピクセル並列処理を行う各視線群に属する相異なる 2 つ以上の視線が同一 ΔT 内で同一メモリバンクをアクセスすることはない．ゆえに，スライス群に対応するメモリバンク群は視角制限ピクセル並列処理が行えるバンクコンフリクトフリーなメモリ構成となっている．

4. 視角制限ピクセル並列処理のためのマクロパイプライン

4.1 マクロパイプラインの導入

1 枚の画像生成は，n 個の視線群を，1 個ずつ**図 8** に示した 6 ステージのマクロパイプラインに $n \times \Delta T$ ごとに投入することでなされる．システム全体のパイプライン構造は次のようになっている（図 8 参照）．

S_1： 視線群計算ステージ：視線群のスライス群への n 個の入射点を算出する．

S_2： 入射点ディスパッチステージ：計算済みの n 個の入射点をスライス群の対応位置のラッチ（LT）に ΔT に 1 個ずつ送出する．

S_3： ピクセル並列レイキャスティングステージ：このステージは S_3^0 と S_3^1 からなる．

  S_3^0： 視線群の各視線がスライス群に $n \times \Delta T$ で入射完了する．側面観察空間での視線群はこのステージですべてスライス群から出力される．底面観察空間での視線群はこのステージでスライス群内にすべて入力される．

  S_3^1： 側面観察空間での視線群に対してはこのステージでは処理は何もなされない．底面観察空間での視線群に対しては，進入した視線が各々レイキャスティングを行ってスライス群を通過していく．

S_4： 色バッファと深さバッファ構築ステージ：スライス群から出力する各視線のピクセル値を色バッファに，ある表面に交差するまで視線が進んだ前進ステップの数で求めた深さ値を深さバッファに

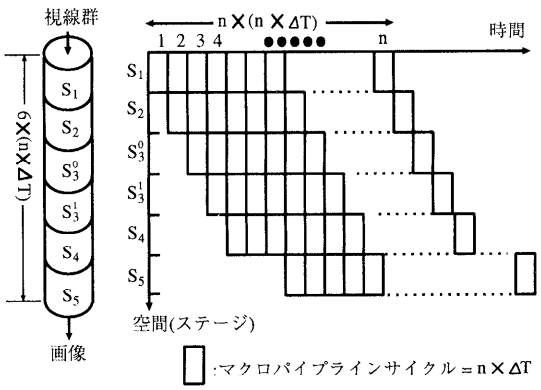


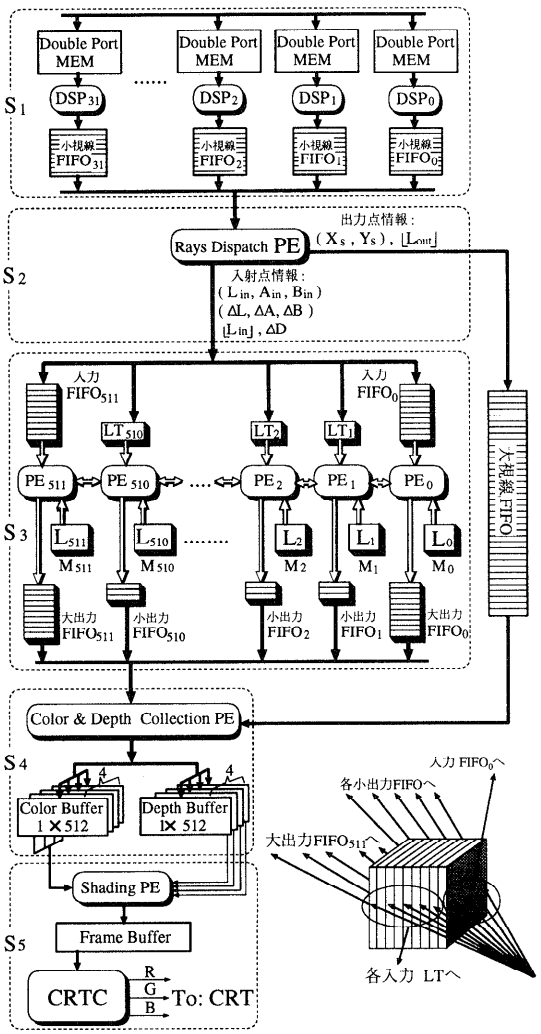
図 8 1 枚の画像生成の視線群マクロパイプライン
Fig. 8 The action of the ray-group parallel macro pipeline for a frame generation.

ΔT に1つずつ書き込む.

S_5： シェーディングステージ：周辺のピクセルの深さを参照して表面の法線を近似計算し，それをもとに各ピクセルの輝度を depth gradient shading 法により求める.

4.2 マクロパイプラインとプロトタイプ構成

4.1節で述べたマクロパイプラインに対するプロトタイプ構成を図**9**に図示した．ここでは，ボリューム空間，スクリーンの大きさは，各々512^3ボクセル，512^2ピクセルとしている．よって，画面生成速度は$(512^2 \times \Delta T)^{-1}$となっている．30画面/秒の描画速度を実現するため，少なくとも，$\Delta T = 120\,\mathrm{ns}$にする必要がある．プロトタイプシステムでは構成部品の状況より$\Delta T = 80\,\mathrm{ns}$の設定が可能であり，この場合には，$512^2$ピクセルの画面で48画面/秒の生成，あるいは，$640^2$ピクセルの画面で30画面/秒が生成できる．以後，$n = 512$，$\Delta T = 80\,\mathrm{ns}$としてマクロパイプラインの構成を示す.



LT：ラッチ L：スライス M：メモリバンク

図**9** メモリ構成を中心としたマクロパイプライン

Fig. 9 Overview of the ray-group parallel macro pipeline structure.

4.2.1 視線群計算ステージ S_1

S_1では，1走査線に対応する512本の視線（視線群）に対してそれらのスライス群への入射点，視線ベクトル成分等を計算する.

DSP TMS320C40をプロセッサとして使用し，40 nsの命令サイクルで動作させる．$\Delta T = 80\,\mathrm{ns}$で1視線の生成を行うとすると，$S_1$は40 nsの命令サイクルで動作するため，2サイクルで1視線の生成を行う必要がある．この速度でS_1を動作させるために，並列化を仮定し，S_1で必要なプロセッサ数を見積もる．DSPの視線生成処理は1ピクセルあたり平均50命令程度で行えるので，S_1のプロセッサ数は$\frac{50}{2} = 25$台程度あれば基本的な視線生成処理を行うことができる.

本計算機では，S_1は余裕を持たせてTMS320C40を32台並列に動作させるものとする.

視線群R_i $(i = 0,1,\cdots,511)$の各視線のうち，DSP_k $(k = 0,1,\cdots,31)$で視線$R_{k+32\times j}$ $(j = 0,1,2,\cdots,15)$が生成され，計算済みの入射点が小視線$FIFO_k$に送出される．$512 \times \Delta T$までに512視線の入射点がS_1の各小視線FIFOに格納される．各小視線FIFO長は16入射点分の容量をとる.

4.2.2 入射点ディスパッチステージ S_2

S_2のRays Dispatch PEは小視線$FIFO_k$ $(k = 0,1,\cdots,31)$からΔTごとに1入射点ずつ取り出してブロードキャスト方式でS_3へディスパッチする．各視線のディスパッチ順については4.2.4項で述べる.

各視線には以下の情報が付随している.

- **入射点情報**：A_{in}, ΔA, B_{in}, ΔB, L_{in}, ΔL, ΔD, $\lfloor L_{in} \rfloor$☆．これらは各視線がボリューム空間に入射される位置(L_{in}, A_{in}, B_{in})と傾き$(\Delta L, \Delta A, \Delta B)$とピクセル値の補正に使う$\Delta D$，スライス番号$\lfloor L_{in} \rfloor$からなる．これらの情報の全ビット長は48ビットである.
- **出力点情報**：(X_s, Y_s), $\lfloor L_{out} \rfloor$．これらは，各視線のスクリーン上の座標値(X_s, Y_s)，出力されるスライス番号$\lfloor L_{out} \rfloor$からなる（詳細は4.2.4項参照）．これらの情報の全ビット長は27ビットである.

☆ $\lfloor X \rfloor$：Xを超えない最大の整数.

4.2.3 ピクセル並列レイキャスティングステージ S_3

4.2.3.1 近接同期通信と並列同期計算

図9のS_3枠に示すように，同一構造をした512個のプロセッサをメモリバンクに1個ずつ1次元アレイ状に規則正しく配置し，直接隣接したプロセッサ間でのみ1ステップ（単位時間）で直接通信できる構成を採用している．

視線群のうちメモリバンク群（L）に進入した各視線は，各々各自が通過しているところのメモリバンクに配置したプロセッサで，2.3節のレイキャスティング算式に従って，計算を**並列**に行う．複数視線に対する漸化式(3)による累積カラー計算は，これらのパイプラインプロセッサ上を80 nsごとに重畳的に流れていく．

4.2.3.2 視線群の底面への逐次入射と側面への同時入射

スライス群上，底面に至る入射点はS_3ステージにおける対応プロセッサの**入力FIFO**に格納し，側面に至る視線の入射点はS_3ステージにおける相応プロセッサの**入力ラッチ（LT）**に格納することによって3.3.1項の前提条件3が満足される．入力FIFO長は1つの視線群の512個の入射点情報分の容量をとる．

また，その視線群がスライス群を出たとき，各視線の累積カラーの最終値，ピクセル値は，底面から出たものが底面に対応する**大出力FIFO**に，側面から出たものが，そのスライスに対応する**小出力FIFO**に書き込まれる．大出力FIFO長は1つの視線群の512個の出力結果分（ピクセル値等）の容量をとり，小出力FIFO長は，32出力結果分の容量をとる．

図9の右下に示した例では，右から見ているボリュームがある．ボリュームを右より見ているので，右側の底面に入射する視線が入力$FIFO_0$に入り，ボリュームの正面，すなわちスライス群の側面に入射する視線は相応スライスのラッチに入る．一方，視線群がスライス群を出る場合，左側の底面から出たものは大出力$FIFO_{511}$に入り，後の視線は相応スライスの小出力$FIFO_i$ $(i = 1, 2, \cdots, 510)$から出てくる．

4.2.3.3 S_3の詳細動作

S_3の各PEは処理が幾分複雑であるので，6段パイプラインとなっている（レイテンシは$6 \times 80\,\mathrm{ns} = 480\,\mathrm{ns}$）．視線に沿ってサンプリングするボクセル値の座標を$(\hat{L_j}, \hat{A_j}, \hat{B_j})$ $(j = 0, 1, \cdots, 511$ かつ $0 \leq \hat{L_j}, \hat{A_j}, \hat{B_j} < 512)$とすると，$S_3$の$PE_{L_j}$では，以下のパイプライン処理がなされる．

(1) 次のサンプリング座標$(\hat{L_j}+\Delta L, \hat{A_j}+\Delta A, \hat{B_j}+\Delta B)$を求める．L軸の座標$\hat{L_j}+\Delta L$は$0 \leq \hat{L_j}+\Delta L < 512$の場合，ボクセル値$v_j$がスライス（メモリバンク）$L_{j-1}(M_{j-1})$または$L_j(M_j)$または$L_{j+1}(M_{j+1})$に存在する可能性がある．$(\hat{L_j}+\Delta L, \hat{A_j}+\Delta A, \hat{B_j}+\Delta B)$と下記の(4)で生成される$K_j$，$\alpha_j$とを下記の式に従って隣接した$PE_{L_j+1}$または$PE_{L_j-1}$へ転送する．すなわち，$\hat{L_j}$の小数部分を$\hat{L}_{j_{小}}$として，以下の計算結果により方向を決定する．

$$\hat{L}_{j_{小}}+\Delta L = \begin{cases} \geq 1 & PE_{L_j+1} \text{へ転送} \\ \geq 0 \text{ かつ } < 1 & \text{転送なし} \\ < 0 & PE_{L_j-1} \text{へ転送} \end{cases}$$

(2) $\lfloor \hat{A_j} \rfloor$と$\lfloor \hat{B_j} \rfloor$をアドレスとして$M_{L_j}$メモリバンクからボクセル値$v_j$を読み出す．

(3) 2.4節のピクセルの補正方法により，
$v_j \times 2^{m+1} \| (\Delta D - 1) \times 2^m$
をk/αの拡張LUTのアドレスとして，補正後の$K'(v_j)$，$\alpha'(v_j)$を読み出す（$\|$は連接を表す）．

(4) 2.3節の漸化式(3)に$K'(v_j)$，$\alpha'(v_j)$を代入して，K_j，α_jを求める．

(5) depth gradient shadingをサポートするため，視線上のボクセル値を視点に近い側から調べ，ボクセル値が大きく変化する所をボリュームの表面と見なすことができる．よって，上記の処理を行うと同時に，視線が表面に交差するまでのdepthを求めるため，ΔDの深さを累積させておく．

(6) もし，座標$(\hat{L_j}+\Delta L, \hat{A_j}+\Delta A, \hat{B_j}+\Delta B)$がボリューム空間外となる場合，求めた$K_j$と$\Delta D$の累積深さを色とdepthとして，$PE_{L_j}$の出力$FIFO_{L_j}$に書き込む．

4.2.4 色バッファと深さバッファ構築ステージ S_4

4.2.4.1 色バッファと深さバッファへの書き込み要求

上記のように，スクリーン座標点(X_S, Y_S)のピクセルを通った視線はスライス群表面の点(L_{in}, A_{in}, B_{in})に入射して，それから，スライス群表面の座標点$(L_{out}, A_{out}, B_{out})$から出力される．その視線に沿って生成した色および深さ値を，色バッファおよび深さバッファの(X_S, Y_S)位置に書き込まなければならない．すなわち，その視線の色および深さ値を$\lfloor L_{out} \rfloor$番のプロセッサの出力FIFOから取り出して色バッファ

および深さバッファの (X_S, Y_S) 位置に書き込む必要がある．ここで，$\lfloor L_{out} \rfloor$ ($\lfloor L_{out} \rfloor = 0, 1, \ldots, 511$) は，L 軸に垂直な 512 個スライスの番号である．

最も単純な方式は S_3 で投入された各視線に (X_s, Y_s) 値を付随させる方法である．この方式では，大出力 $FIFO_0$，大出力 $FIFO_{511}$，およびすべての小出力 $FIFO_i$ ($i = 1, 2, \cdots, 510$) をチェックして出力視線が存在すればそれを対応する色バッファと深さバッファに出力する．しかし，視点の位置によって出力視線の大部分が大出力 FIFO に存在する場合や出力視線のほとんどが小出力 FIFO に存在する場合などがある．特に前者の場合には少数の出力視線をすべての小出力 FIFO にわたって探す必要があり，時間の無駄が多い．そこで次のような方式を採用した．

4.2.4.2 実現方法

(a) S_2 ステージと S_4 ステージとの間に 2 つの視線群分（2×512）個の出力点情報が格納できる大視線 FIFO をバイパスとして用意する．

(b) S_2 ステージでは入射点情報を S_3 ステージに送出するとともに，その視線の出力点情報 (X_S, Y_S)，$\lfloor L_{out} \rfloor$ を大視線 FIFO に書き込む．

(c) S_4 ステージでは，大視線 FIFO から (X_S, Y_S)，$\lfloor L_{out} \rfloor$ を取り出し，$\lfloor L_{out} \rfloor$ により S_3 ステージの $\lfloor L_{out} \rfloor$ 番のプロセッサの出力 FIFO から色と深さ値を取り出して，色バッファと深さバッファの (X_S, Y_S) 位置に書き込む．

(d) S_3 ステージの各出力 FIFO 内の色と深さ値の配列順を大視線 FIFO 内の (X_S, Y_S)，$\lfloor L_{out} \rfloor$ の配列順に一致させるため，S_2 ステージでは，以下に示す入射点のディスパッチ順序に従い，入射点情報と (X_S, Y_S)，$\lfloor L_{out} \rfloor$ を各々 S_3 ステージと大視線 FIFO に送出している．

- 視線群の各視線 R_i ($i = 0, 1, \ldots, 511$) からなる平面上に視点からスライス群の断面に平行な直線 r_A を引いておく．
- もし視線群が r_A の片側にあるならば，r_A となす角の大きい視線の入射点が優先ディスパッチされる．
- もし r_A が R_j と R_{j+1} ($1 \leq j \leq 510$) の間を通過すると，入射点のディスパッチは，$R_0, R_1, \ldots, R_j, R_{511}, R_{510}, \ldots, R_{j+1}$ の順である．

このようにすると，**図 10** に示すように視線のディスパッチ順序が大出力 FIFO に蓄積される順になっている．たとえば，図 10 (a) では，大出力 $FIFO_0$ の先頭より取り出されたデータが大視線 FIFO の先頭から取り出された (X_s, Y_s) 位置に順々に書き込まれる．その後，大視線 FIFO 上に登録された視線に対するデータのみが対応する小出力 FIFO より取り出される．

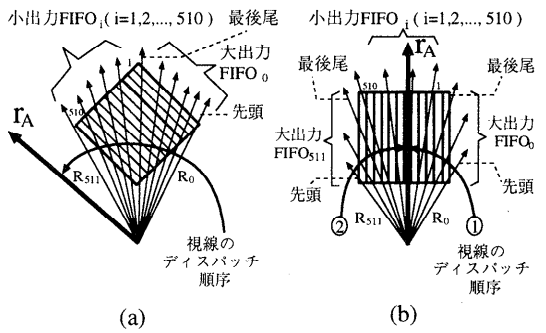


図 10 入射点のディスパッチ順
Fig. 10 Dispatch sequence of incident rays.

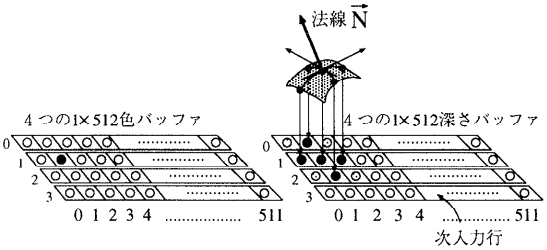


図 11 シェーディングステージ S_5
Fig. 11 Shading Stage S_5.

4.2.5 シェーディングステージ S_5 の構成

S_5 には，サイクル $\Delta T = 80$ ns の 7 段パイプラインからなる Shading PE が配置されている．シェーディングステージでは 3×3 の画素点に対する depth gradient shading がなされる．3×3 画素点の中央の画素の法線ベクトル $\vec{N}$ は**図 11** のように深さバッファから求められる[4]．この法線ベクトル $\vec{N}$ と光源ベクトル $\vec{I}$ を用い，シェーディングがなされる．すなわち，図 9 の S_5 枠のフレームバッファの画素値 P_s は，$P_s = P_c(\vec{I} \cdot \vec{N})$ で計算される．P_c は色バッファの色値である．ある画素の上下左右の画素の深さ値が必要となるので，このため，図 9 の S_4 に示した 4 行分の深さバッファが用意されている（図 11 参照．3 行のバッファの中心行にある画素に depth gradient shading がなされる．残りの 1 つのバッファは次の行のデータ入力に使用される）．

5. 関連研究

ニューヨーク州立大学の Cube-2 までの既存のボリュームレンダリング専用並列計算機について著者らの論文[4] [p.1716] にまとめているので，以下では最近提案された Cube-3 との比較を行うにとどめる．また，

Cube-3との実装上の比較については付録に述べる．

Cube-3のアーキテクチャ[2]は実時間レイキャスティングを実現するため，高度に並列化とパイプライン化し，512^3 ボクセルに対して1秒間に30画面が生成できる．

メモリ構成は以下のようになっている．

座標 (x, y, z) のボクセル値は次の式により

$$k = (x + y + z) \bmod n \quad 0 \leq k, x, y, z \leq n - 1$$

k番目のメモリバンクに写像される．したがって，n^3 ボクセル空間をn個のメモリバンクに格納し，各メモリバンクに n^2 ボクセル値を記憶している．CFBと呼ばれるこのメモリバンク群は，軸に平行な視線に対してコンフリクトフリーなアクセスが可能となる．任意視点の平行投影をサポートするため，視線群が通過する平面（PRPと呼ばれる）上の全ボクセルは，Fast Busを通して，2DSBと呼ばれるn個のメモリバンク（$2n \times n$ サイズ）に記憶される．各視線上のボクセルは一直線に並べ換えられ，コンフリクトフリーにアクセスされる．しかし，透視投影による視線群に対しては，PRP平面を構成する，軸に平行なビーム群は，いくつかのビームが平均された後，2DSBに記憶されるので，透視投影の計算精度が低下する．Fast Busは高速マルチプレクサ，トランシーバとマルチチャンネルバスで構成される．またマルチプレクサとトランシーバの操作は，実時間データ変更を行うlook-up tableによって，制御される．Fast Busが高速データ読み取りのボトルネックとなっており，ハードウェア量は膨大である．

これに対して，本計算機ではスライス群という非常に簡単なメモリ構成を用いて，平行投影でも透視投影でも描画が高速に行える．透視投影の計算精度にも影響がない．

また，Cube-3ではピクセル値計算ステージはRPCという逆ツリー構造をとり，ピクセル値を生成する．本計算機のこのステージは1次元アレイ状に配置するプロセッサ群の構成だけでピクセル値が生成できる．

6. おわりに

本稿では，ボリューム表示を超高速に行うアーキテクチャとして，視角制限ピクセル並列処理を提案し，それに基づいたシステム構成法について述べた．視角を実用上問題のない範囲で制限することによって，比較的単純なハードウェアでシステムが構成できる特徴がある．現在，DSP TMS320C40やFPGA XC4000などを用いて詳細設計を行っている．

謝辞 本研究に関して貴重なご意見をいただいた，明石英也氏（現在（株）日立製作所），ならびに日頃からご討論いただく京都大学工学研究科情報工学富田研究室の皆様に感謝いたします．

参考文献

1) 中嶋正之，川合 慧：グラフィクスとマンマシンシステム，岩波書店 (1995).
2) Pfister, H., et al.: Cube-3: A Real-time Architecture for High-resolution Volume Visualization, *Proc. 1994 Symp. on Volume Visualization*, pp.75–82 (1994).
3) Cohen-Or, D., et al.: A 3D Skewing and De-skewing Scheme for Conflict-free Access to Rays in Volume Rendering, *IEEE Trans. Comput.*, Vol.44, No.5, pp.707–710 (1995).
4) 對馬雄次，明石英也，金 喜都ほか：ボリュームレンダリング専用計算機 ReVolver アーキテクチャ，情報処理学会論文誌，Vol.36, No.7, pp.1709–1718 (1995).
5) 對馬雄次：ボリュームレンダリング専用計算機—ReVolver/C40，京都大学修士論文 (1995).
6) Kaufman, A. and Bakalash, R.: Memory and Processing Architecture for 3D Based Imagery, *IEEE Computer Graphics & Application*, Vol.8, No.6, pp.10–23 (1988).
7) Levoy, M.: Display of Surfaces from Volume Data, *IEEE Computer Graphics & Application*, Vol.8, No.5, pp.29–37 (1988).
8) Ohashi, T., Uchiki, T. and Tokoro, M.: A Three-dimensional Shaded Display Method for Voxel-based Repersentation, *Eurographics 85*, pp.221–232 (1988).
9) Westover, L.: Interactive Volume Rendering, *Proc. Chapel Hill Workshop on Volume Visualization*, pp.9–16 (1989).
10) Cohen, L.T., et al.: Real-time Discrete Shading, *Trans. Vis. Comput.* (West Germany), Vol.6, No.1, pp.16–27 (1990).

付 録

Cube-3との実装上の比較

Cube-3では，各クロックサイクルごとに，ボリューム空間から同時に取り出した主軸等間隔サンプリングによる視線上の全ボクセル値を，逆2進ツリー構造の階層パイプラインRPC（Ray Projection Cone）の相応の葉ノードLEAFに送入する．RPC上では，レイキャスティング法によるピクセル値の計算を階層パイプライン的に行い，各クロックサイクルごとにRPCのルートノードからピクセル値が流れ出てくるというボクセル並列処理構成を採用している．

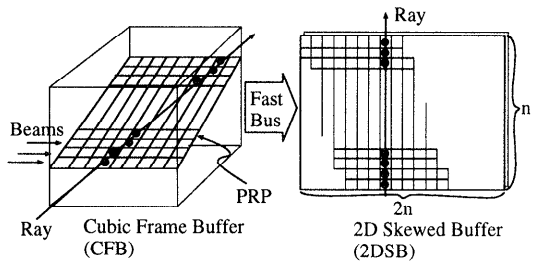


図12 Cube-3のメモリ構成メカニズム
Fig. 12 The memory system of Cube-3.

Cube-3の著者らは，ボリューム空間を格納するメモリ空間から任意な視線上のボクセル値を同時に読み取る直接的な方法がないとし，以下のような間接的な方法を採用している．

n^3 ボクセルを持つボリューム空間のうち，任意の座標軸と平行に並んでいるn個のボクセルをビームと呼ぶ．スクリーンの1走査線に対応する視線群がボリューム空間を切断する面PRP（Projection Ray Plane）はn本のビームから構成される．ボリューム空間を格納するCFB（Cubic Frame Buffer）は n^2 サイズのn個のメモリバンクから構成されている．切断面PRP上の全ボクセル値は，1ビーム分が後述のように並列アクセスされるので，nクロックサイクルごとに順番に取り出され，逆スキューを行うFast Busを経由して，2DSB（2D Skewed Buffers）にコンフリクトなしで書き込まれる．2DSBは2nサイズのメモリをn個有しており，同一アドレスで同一視線上のボクセル値を同時に取り出せるようになっている（平行視線の場合）．2DSBが2セットあり，交互に書き込ませることによって，各クロックサイクルごとに2DSBから1視線上の全ボクセル値を取り出して，RPC逆ツリー構造上の各葉ノードLEAFに送入することが保証される（**図12**参照）．

平行投影の場合には，視線群の各視線は，n個の異なるアドレスを使って2DSBからnクロックサイクルで順番に取り出せる．透視投影の場合には，オリジナルの視線上のボクセル値は2DSBから直接に取得できない．Cube-3の処理手法は，透視投影視線群の視線間の開度（発散程度）によって，Fast Busでの通過途中で，複数のビームが1ビームに平均化され，そして，2DSBに書き込まれる．したがって透視投影の計算精度にある程度の影響が生じる．

CFBメモリ空間からボリューム空間の任意のビームを同時に取り出すため，ボリューム空間からCFBメモリ空間への写像は次のスキュー写像算式によって行う．

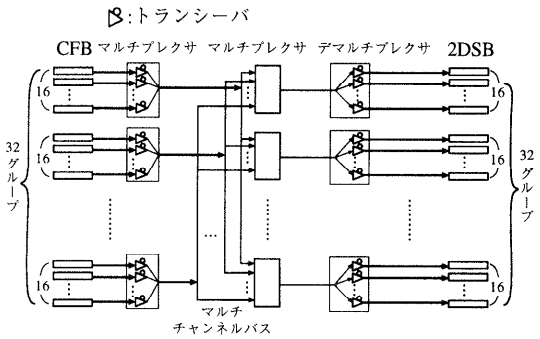


図13 Fast Busの接続構成
Fig. 13 Organization of Fast Bus.

$$k = (x + y + z) \bmod n \quad 0 \le k, x, y, z \le n - 1$$

この式で，ボリューム空間の座標点 (x, y, z) のボクセル値はk番目のメモリバンクの $(i = x,\ j = y)$ 位置に保存しておく．ボリューム空間のうち，任意のビーム上の各ボクセル値の3つの座標値は，ビームの方向に平行な軸上の座標値だけが0から $n-1$ まで変化し，あとの2軸の座標値が同じになっている．したがって，上の写像により，ボリューム空間のすべて（$3 \times n^2$ 本）のビーム上のn個のボクセル値はそれぞれn個のメモリバンクのCFBに格納され，同時アクセスできる．

一方，Fast Busでは，**図13**の接続構成に示すように，$n = 512$ の場合，それぞれCFBと2DSBの512個のメモリを，$m = 16$ 個のメモリを1グループとした32グループに分割する．CFBと2DSBの各メモリに接続した 2×512 個のトランシーバを，32本のマルチチャンネルバスと32個の32:1のマルチプレクサを用いて連接する．すなわち，CFBから2DSBへの512個のボクセル値の転送は32個ずつ $m = 16$ 回で行う必要がある．そこで，画面生成速度は1秒間に30画面であるとすると，1ビームの呼び出し時間は $1\text{秒}/(30 \times 512^2) = 128\,\text{ns}$ となるので，Fast Busは $128/16 = 8\,\text{ns}$ で1回転送を行う必要がある．したがってCube-3の画面生成速度はFast Busの転送速度で制約されている．明らかに，Fast Busはボリューム空間が大きくなるにつれて，高速描画を行うボトルネックとなる．

また，Fast Busのマルチプレクサとトランシーバの操作はLook-up Tablesから制御されている．視点の移動にともない，このLook-up Tablesのパラメータはリアルタイムでホストマシンからダウンロードされる．したがって，Fast Busは，複雑で，大量スイッチが必要である．

なお，Cube-3では，レイキャスティング法によるピクセル値の計算を行う逆2進ツリー構造RPCは，線

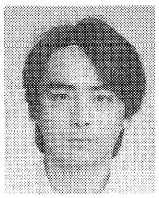
形補間などを行うn個の葉ノードLEAFとn−1個のボクセル値合成ノードVCUの合計2n−1個のノードプロセッサからなり，n＝512の場合，576専用チップ（512個の葉ノードチップと511/8＝64合成ノードチップ．各合成ノードチップに8個の合成ノードを配属）が用いられている．

本計算機は，Cube-3のCFB，Fast Busと2DSBに取り代わる，ボリューム空間からメモリ空間への非常に簡単な写像によるn個のメモリバンクから構成したスライス群から，平行投影でも透視投影でもオリジナルの視線群をピクセル並列に読み出せる構成になっている．また，Cube-3上の2n−1個のノードプロセッサを持つ逆2進ツリー構造RPCに相応するものとして，1次元アレイ状に配置したn個のプロセッサで，視線群の各視線に沿ってレイキャスティングを行っている．n＝512の場合，4プロセッサ分を3つのFPGAチップで制作するので合計384個のチップが必要である．画面生成速度は，Cube-3がFast Busによって制約されているのに対して，本計算機はボリューム空間を格納するメモリのサイクルタイムによって制約されている．したがって，より大きな解像度（たとえば768^3または1024^3ボクセル）のボリューム空間にしても，本計算機では高速な画面生成率が得られる特徴がある．

(平成8年9月13日受付)
(平成9年2月 5 日採録)

金　喜都（正会員）

1958年生．1982年中国ハルピン科学技術大学計算機工学科卒業．1982～1992年中国航空宇宙部第一研究院計算機応用技術研究所に勤務．その間，1988年中国ハルビン工業大学大学院計算機工学専攻修士課程修了．1996年京都大学大学院工学研究科情報工学博士課程修了．同年4月よりウッドランド（株）に勤務．現在，マルチメディアの研究に従事．



對馬　雄次

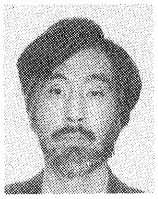
1968年生．1993年京都大学工学部情報工学科卒業．1995年同大学大学院修士課程修了．現在，(株）日立製作所に勤務．

中山　明則

1972年生．1994年京都大学工学部情報工学科卒業．1996年同大学大学院修士課程修了．現在，NTTデータ通信（株）に勤務．

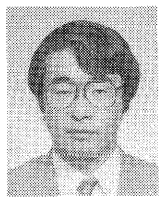
森　眞一郎（正会員）

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年同大学大学院総合理工学研究科情報システム学専攻博士課程単位取得退学．同年京都大学工学部助手．1995年同助教授．工学博士．並列分散処理．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）

1956年生．1981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．1992年より京都大学工学部助教授．1997年豊橋技術科学大学情報工学系教授．並列計算機のアーキテクチャ，プログラミング言語の実装方式に関する研究に従事．工学博士．1988年元岡賞，1993年坂井記念特別賞受賞．IEEE-CS, ALP各会員．



富田　眞治（正会員）

1945年生．1973年京都大学大学院博士課程修了，工学博士．同年同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．計算機アーキテクチャ，並列計算機システムに興味を持つ．著書「並列コンピュータ工学」「並列処理マシン」「コンピュータアーキテクチャI」など．電子情報通信学会，IEEE, ACM各会員．本会理事．