レイトレーシングを高速処理する専用並列レンダリング・マシンのマクロパイプライン・アーキテクチャ

權, 五鳳
九州大学大学院総合理工学研究科情報システム学専攻

村上, 和彰
九州大学大学院総合理工学研究科情報システム学専攻

富田, 眞治
九州大学大学院総合理工学研究科情報システム学専攻

https://doi.org/10.15017/17212

出版情報：九州大学大学院総合理工学報告. 12 (4), pp.385-394, 1991-03-01. 九州大学大学院総合理工学研究科
バージョン：
権利関係：

九州大学大学院総合理工学研究科報告
第12巻 第4号 385-394頁 平成3年3月

Engineering Sciences Reports, Kyushu University
(KYUSHU DAIGAKU SOGORIKOGAKU KENKYUKA HOKOKU)
Vol. 12, No. 4 pp. 385-394 March 1991

レイトレーシングを高速処理する専用並列レンダリング・マシンのマクロパイプライン・アーキテクチャ

權　五鳳*・村上和彰**・富田眞治**

(平成2年11月30日　受理)

A Parallel Rendering Machine for High Speed Ray-Tracing —Macropipeline Architecture—

Oubong GWUN, Kazuaki MURAKAMI and Shinji TOMITA

A parallel rendering machine for high-speed ray tracing has been being built at Kyushu University. The machine exploits multiple-level parallelism as follows.

(1) Multiprocessing image space by dividing it among processors.
(2) Macropipelining every ray within each processor.
(3) Instruction-level parallelism with VLIW architectures.

This paper presents the design philosophy of the machine and discusses its macropipeline architecture.

1. はじめに

コンピュータグラフィックスは，効果的なマン－マシン・インターフェース手段として，CAD/CAMを始め，ビジュアライゼーション，アニメーション，芸術分野まで利用の幅が広がっている．それにともない，非常にリアリティの高い映像の高速生成が要求されている．

3次元映像生成アルゴリズムとして，視線探索法(ray tracing)がある[13]．このアルゴリズムは，3次元空間でイメージをレンダリングするので，反射，屈折，透過，陰影などの表現が優れている．よって，極めてリアルな映像が生成可能である．しかしその反面，映像生成に膨大な時間を要するといった短所がある．処理速度を向上するために，ソフトウェア[7),11),12)]およびハードウェア[3),4),6)]の両面から研究が盛んに行われている．

ハードウェアによる高速化としては，並列処理[2)]による方法がある．これは少なくとも，マルチプロセッサ・レベル[3),6)]と命令レベルの2つに分けられる．マルチプロセッサ・レベルでは，アルゴリズムに内在する並列性が画素[3)]ないし物体空間単位[6)]に分割されるのを利用する．命令レベルでは，命令に内在する並列性を利用する．さらに中間（プロセッサ）レベルでの並列化（視線探索法のマクロパイプライン化）も可能である．

我々は，これら3レベルの並列処理を活用し，以下の特長を有する視線探索法専用の並列レンダリング・マシン『熱視線』の開発を行っている[1)]．

①視線探索法に特化したレンダリング
②画面分割方式によるマルチプロセッサ処理
③1本の光線に対するマクロパイプライン処理
④負荷に応じたマクロパイプラインの各ステージの多重化
⑤空間等分割法による高速物体探索
⑥VLIW方式による命令レベル並列処理

本マシンは特に，各プロセッサ内部におけるマクロパイプライン処理に重点を置いている．本稿は，『熱視線』のマクロパイプライン方式および処理，各ステージの処理，負荷および構成，等を中心に述べる．

2. 高速化に関する方針

2.1 高速化技法に関する従来の研究

視線探索法は処理時間が膨大であり，その大部分は

*情報システム学専攻博士過程
**情報システム学専攻

交差判定に費やされる．したがって，本アルゴリズムを高速化するには，交差判定を高速化する必要がある．この方法としては，次の2つが考えられる．

①実行すべき交差判定の回数自身をなんらかの方法を使って減らす．

②単位時間当りの交差判定実行回数（スループット）を増やす．

(1)　交差判定回数の削減

交差判定に時間を要する主な理由としてその対処法には，次のものがある．

①［理由］探索すべき視線（1次光線）の数が膨大である．

［対処］他方式兼用法[12]：物体と交差しそうな一次光線に対応する画素を予め別の方式（たとえば，Zバッファ方式）で求めておいて，その一次光線のみを追跡する．

②［理由］反射および屈折によって光線（分岐光線）が増加する．

［対処］寄与値利用法[12]：1次光線あるいは2次以降の分岐光線が物体と交差した際の分岐光線の生成を，物体の輝度に影響を与える程度に応じて調整する．

③［理由］発生した光線は物体空間のすべての物体と交差判定を行うので，物体の数に比例して計算時間が増加する．

［対処］物体探索法：交差の可能性のない物体を交差判定の対象から外す．つまり，まず交差の可能性のある物体を探索し，見つかった物体に対してのみ交差判定を行うようにする．これは，次の2方式が提案されている．

a）外接体法[12]：プリミティブを木構造の階層的な外接体で取り囲む．上位レベルの外接体から交差判定を行い，交差した外接体のみ下位レベルへと交差判定を進める．

b）空間分割法[9]：空間を分割しておいて，光線が通る部分空間（ボクセル）内の物体のみを交差判定の対象とする．空間の分割方法には，オクトリー分割法[9]，空間等分割法[7),11]，などがある．

(2)　交差判定スループットの向上

①命令レベル並列処理：以下の交点計算の特長に注目して，命令レベル並列性を活かした専用交点計算器を設ける[4]．

- ポリゴンあるいは2次曲面に対して交点を求めるための方程式の各係数が独立に計算可能である．
- 対称性を持つ3次元ベクトル演算が主体である．

②プロセッサ・レベル並列処理：視線探索法のタスクは，基本的に交差判定と輝度計算の2個のサブタスクに分割でき，かつ，オーバラップ処理可能である．この点に注目して，各光線に関するタスクをパイプライン処理する[8]．

③マルチプロセッサ・レベル並列処理：視線探索法自身が有する並列性を活かす．すなわち，個々の光線あるいは物体に関する処理が互いに独立している点に注目して，各処理を各要素プロセッサで並列に行う．これには，次の2方式がある．

a）画面分割方式[3),4]：画面を画素単位に分割し，各画素を要素プロセッサに分配して並列に処理する．

b）物体空間分割方式[6]：物体空間を部分物体空間に分割して，各部分物体空間を要素プロセッサに担当させる．交点計算は，光線が実際の空間を進むように，隣接するプロセッサを移動しながら行われる．本方式はまた，交差判定の回数自身も削減する．つまり，光線が進む部分物体空間についてのみ交差判定すればよい．

2.2 『熱視線』の設計方法

前節で述べた各種高速化技法のうち，『熱視線』では以下のものを採用する．

①交差判定回数の削減;

a）寄与値利用法

b）空間分割法：空間等分割法

②交差判定スループットの向上;

a）命令レベル並列処理

b）プロセッサ・レベル並列処理：マクロパイプライン方式

c）マルチプロセッサ・レベル並列処理：画面分割方式

上記のうち，①－bの空間等分割法，および，②－cの画面分割方式に関しては，以下の考察に基づいてその採用を決定した．②－bのマクロパイプライン方式については，3章で詳述する．

(1)　空間等分割法の採用

交差判定回数の削減を目的とした物体探索法には，2.1節で述べたように次の3方式がある．

①外接体法

②オクトリー空間分割法

③空間等分割法

まず，外接体法はプリミティブが集中している場合，あるいは，プリミティブの数が少ない場合は，空間等分割法により有効である[7),11)]．しかし，以下の短所がある．

- 階層的なデータ構造を構成するのは，容易ではない．
- プリミティブに辿り着くまでの探索時間が一定でない．

次に，オクトリー空間分割法も外接体法同様，プリミティブが集中している場合は空間等分割法よりも有効である[7),11)]．しかし，以下の点で空間等分割法より劣る[7)]．

- 光線が次のボクセルへ移動するのに要する時間が長い．
- 外接体法同様，階層的に探索を行うので探索時間が一定でない．

これらの方法に比べて，空間等分割法は以下の長所を有する[7),11)]．

- 光線の次のボクセルへの移動が高速に行える．
- 探索時間が一定である．

上記の理由により，空間等分割法を採用する．

(2)　画面分割方式の採用

マルチプロセッサ・レベル並列処理の方法には，2.1節で述べたように次の2方式がある．

①画面分割方式

②物体空間分割方式

まず，物体空間分割方式は，以下の特徴を有する[10)]．

- 長所：各要素プロセッサは，担当する部分物体空間のみの物体データを持てばよい．よって，各要素プロセッサ当りのメモリ容量が小さくて済む．
- 短所：負荷分散が難しい．また，要素プロセッサ間で光線を授受するための通信が必要となる．

物体空間分割方式に対して，画面分割方式は以下の特徴を有する[10)]．

- 長所：負荷分散が容易である．また，要素プロセッサ間の通信が不要である．
- 短所：物体データの各要素プロセッサへの分割が困難である．理想的には，各要素プロセッサはすべての物体データを持つ必要があり，メモリ容量の爆発的な増加を招く．

このように，両方式の長所/短所は相補的関係にあるが，負荷分散の容易さ，および，要素プロセッサ間通信の不必要性を重視して画面分割方式を採用する．本方式の短所に対しては，物体データへの参照の局所性[10)]を活かす．つまり，キャッシュを導入することで対処する．

3.　マクロパイプライン方式

3.1　視線探索法のパイプライン化

2.1節で述べたように，視線探索法における個々の光線処理に関するタスクは，交差判定と輝度計算の2個のサブタスクに分割できる．さらに，交差判定回数削減のために，空間等分割法に基づく物体探索法を採

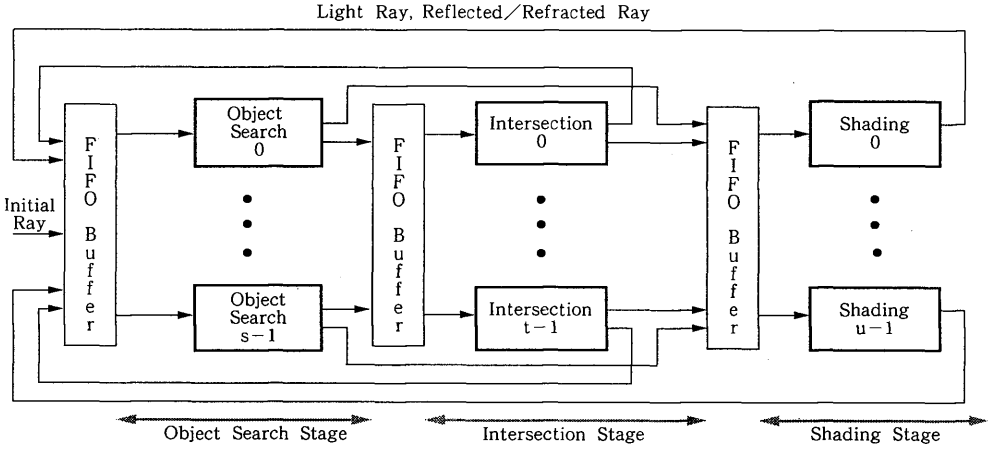


Fig. 1　Macropipeline Configuration.

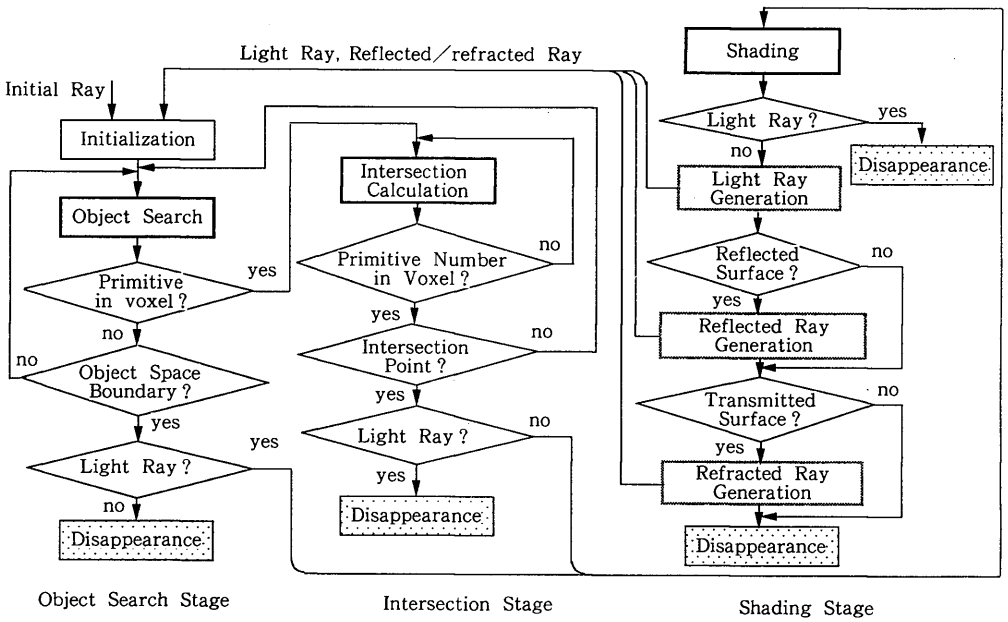


Fig. 2　Ray Proess Flow.

用しており，交差判定はさらに物体探索と交点計算の2個のサブタスクに分割できる．したがって，以下の計3個のオーバラップ可能なサブタスクに分割可能である．

①物体探索
②交点計算
③輝度計算

上記のサブタスクをそれぞれ1ステージとして得られる3ステージ・マクロパイプラインの構成を **Fig. 1** に示す．また，このときの光線処理の全体の流れを **Fig. 2** に示す．光線は基本的に，物体探索，交点計算，輝度計算へとステージを進めながら処理される．

Fig. 1 に示したように，マクロパイプラインのステージ間には FIFO バッファを備える．各ステージは，その入力 FIFO バッファから光線を取り出し，専用のデータベースを用いて処理を行う．処理は，ステージ毎に設けたプログラムの制御に従って行う．処理を終えると，光線を出力 FIFO バッファに格納する．

このように，各ステージの処理が独立したプログラムの制御下におかれ，かつ，FIFO バッファを介して非同期に行われる．このようなパイプライン処理方法を“マクロパイプライン”方式と呼ぶ．

次節で，このマクロパイプライン処理の全体の流れを述べる．各ステージにおける処理の詳細は，4章で述べる．

3.2　マクロパイプライン処理の概要

まず，処理対象となる光線には，以下の4種類がある．

①1次光線（Initial Ray）：画面の各画素に対応する光線で，物体探索ステージへの初期入力となる．処理中に新たに生成されることはない．

②分岐光線（Reflected Ray/Refracted Ray）：1次光線あるいは分岐光線が反射ないし屈折することによって生成される光線，輝度計算ステージにおいて生成され，物体探索ステージへ入力される．

③影光線（Light Ray）：交点が影か否かを判定するために光源に向かって生成される光線，輝度計算ステージにおいて生成され，物体探索ステージへ入力される．

④戻り光線：物体探索ステージからいったん交点計算ステージに進んだ後，物体と交差しなかったために再び物体探索ステージへと戻る光線．これには，1次光線，分岐光線，影光線のいずれの光線もなり得る．

さて，個々の光線は，以下のように処理される．

(1)　物体探索ステージ

光線が物体探索ステージに入力される．光線の進行方向に沿って，物体探索を行う．各ボクセルは，ボクセル内部のプリミティブの有無に関する情報を持っている．いま注目しているボクセル内にプリミティブが存在する場合，光線に当該ボクセルへの識別子を添付して交点計算ステージへ進ませる．そうでない場合は，次のボクセルに進む．

(2)　交点計算ステージ

ボクセル識別子を用いてボクセル内の各プリミティブの形状データを読出し，これらに対して交点計算を行う．1次ないし分岐光線が交差する場合，その交点座標と交差するプリミティブの識別子を光線に添付して輝度計算ステージに進ませる．影光線が交差する場合，それは消滅する．交差しなかったら，光線を物体探索ステージへ戻す．

(3)　輝度計算ステージ

プリミティブ識別子を用いてプリミティブの輝度データを読出し，輝度計算を行う．影光線を生成して，物体探索ステージへフィードバックさせる．さらに，プリミティブの種類によっては，反射および屈折光線といった分岐光線を生成して，物体探索ステージへフィードバックさせる．そして，元の光線は消滅する．

3.3　光線処理に要するデータ

前節で述べた光線処理では，以下の4種類のデータベースの存在を前提としていた．各データベースは，対応するステージに占有される．

①ボクセル・データ：物体探索ステージで用いる．ボクセル対応に設けられ，次の2つのフィールドから成る．
- フラグ：プリミティブの有無を示す．
- ボクセル識別子：プリミティブが存在する場合，対応するプリミティブ・リストの先頭へのポインタ．

②プリミティブ・リスト：交点計算ステージで用いる．ボクセル対応に設けられる単方向リスト．リストの各セルは，プリミティブ対応に設けられ，そのデータ部はプリミティブ形状データへのポインタとなっている．

③プリミティブ形状データ：交点計算ステージで用いる．プリミティブ対応に設けられ，次の2つのフィールドから成る．
- 形状値：交点計算に必要なデータ．
- プリミティブ識別子：プリミティブ輝度データへのポインタ

④プリミティブ輝度データ：輝度計算ステージで用いる．プリミティブ対応に設けられ，輝度計算に必要な色，反射率，透過率，屈折係数，等のデータを保持する．

4.　ステージの処理

4.1　物体探索ステージ

物体探索ステージの処理の詳細は，以下のようになる．

①物体探索 FIFO バッファから，光線データを読み出す．当該光線が戻り光線でなければ，これを初期化する．

②3DDDA (3D Digital Differential Analyzer) アルゴリズムにより，光線が辿るボクセルを決定する．

③当該ボクセルのボクセル・データを読み出し，プリミティブの有無を調べる．プリミティブが有る場合，光線にボクセル識別子を添付して交点計算 FIFO バッファに格納する．

④プリミティブが無い場合，上記②③を繰り返す．
もし，光線が物体空間の終わりに達した場合，
a）1次光線ないし分岐光線は消滅して，処理終了．
b）影光線は，輝度計算 FIFO バッファへ格納する．

3DDDA アルゴリズムには，Synder のアルゴリズム[11]と Fujimoto のアルゴリズム[7]がある．以下の理由により，Synder のアルゴリズムを用いる．
- カーネル・ループの実行ステップ数が少ない．ただし，Fujimoto のアルゴリズムに比べて浮動小数点演算が多いが，浮動小数点演算ユニットを複数備えれば問題ない．
- ボクセル探索に必要な変数が少ない．

4.2　交点計算ステージ

交点計算ステージの処理の詳細は，以下のようになる．

①交点計算 FIFO バッファから，光線データを読み出す．

②プリミティブ形状データを読み出す．

③形状データ内の係数を用いて交点計算を行う．

④ボクセル内のすべてのプリミティブに対して，上

記②③を繰り返す.

⑤交点が存在する場合,

a) 1次光線ないし分岐光線に対しては，最も手前の交点の法線ベクトルを求める．そして，交点ベクトル，法線ベクトル，プリミティブ識別子を光線に添付して輝度計算 FIFO バッファへ格納する.

b) 影光線は消滅し，処理終了

⑥交点が存在しない場合，当該光線を戻り光線として物体探索 FIFO バッファに格納する.

4.3 輝度計算ステージ

寄与植利用法[12]により，分岐光線の生成を制御する．1次光線の寄与値は1，分岐光線の寄与値は元の光線寄与値に反射率ないし透過率を乗じたものとなる.

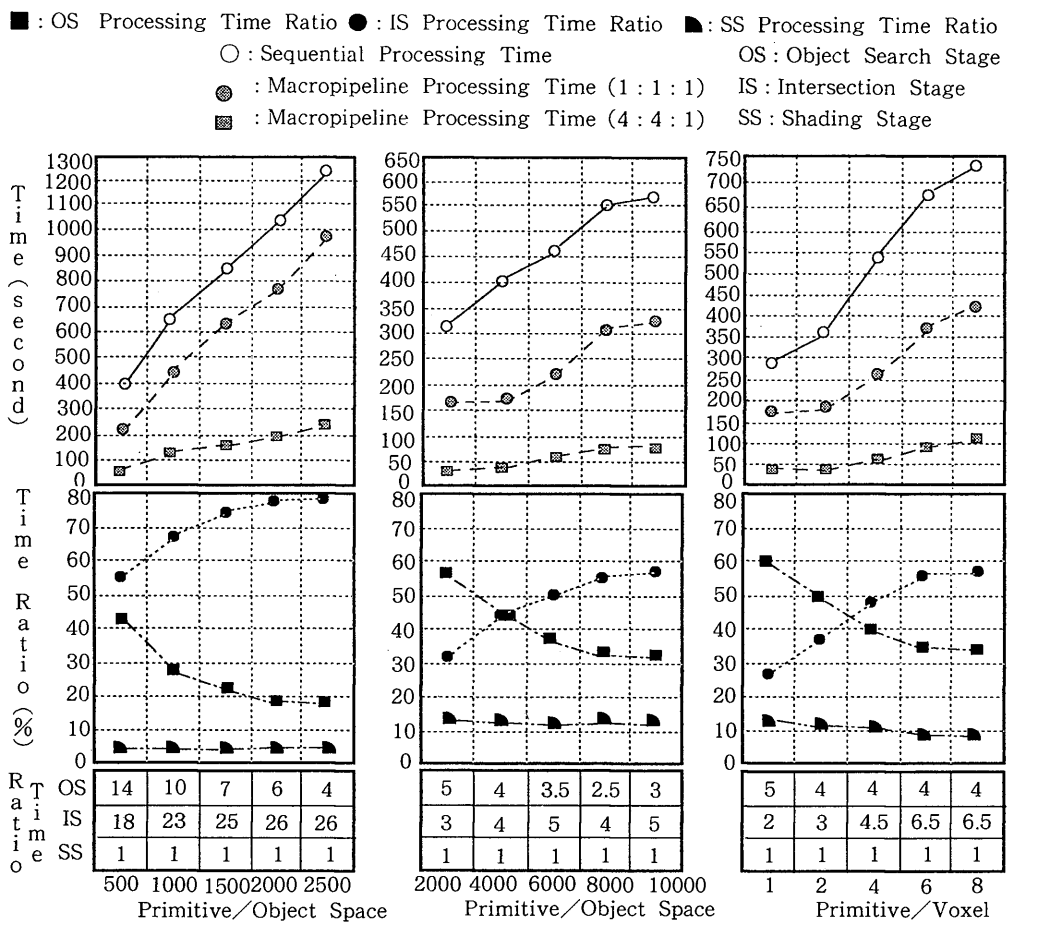
輝度計算ステージの処理の詳細は，以下のようになる.

①輝度計算 FIFO バッファから，光線データを読み出す.

②フレームバッファ内の該当するピクセルに輝度を加算する．輝度計算は光線の種類に応じて，以下のように行う.

a) 1次光線ないし分岐光線：プリミティブ輝度データ内の環境光輝度成分に寄与値を乗じて，該当するピクセルに加算する.

b) 影光線：光線に添付されている輝度成分を該当するピクセルに加算する．影光線は消滅して,



(a) 10Primitives／Voxel Fix　(b) 1Primitive／Voxel Fix (c) 600Primitives／Object Space Fix

Fig. 3 Stage Load.

処理終了.

③1次光線ないし分岐光線に対して，交点から光源へ向かう影光線を生成する．寄与値に拡散反射率を乗じて影光線に添付して，物体探索 FIFO バッファへ格納する．

④1次光線ないし分岐光線に対して，交差した物体の種類に応じて分岐光線（反射光線ないし屈折光線）を生成する．寄与値に反射率ないし透過率を乗じて分岐光線に添付して，物体探索 FIFO バッファへ格納する．

⑤元の1次光線ないし分岐光線は消滅して，処理終了．

5. ステージの負荷

5.1 負荷の計測

マクロパイプライン処理において各ステージの負荷が不均衡である場合，光線処理のスムースな流れが阻害される．この問題を解決するには，ステージ負荷が均衡となるように，負荷の大きいステージを並列化（細分化ないし多重化）する必要がある．そこで，各ステージに要する処理時間を測定した．

処理プログラムは，4章で述べた視線探索法に基づく．測定には，SPARC ワークステーション Solbourne Series 5/600を用いた．UNIX のプロファイル機能 gprof を用いて，以下の3通りについて処理時間の測定を行った．各測定結果を **Fig. 3** 上部に示す．また，各ステージの処理時間率を **Fig. 3** 中部に，輝度計算ステージの処理時間を1としたときの他ステージの処理時間の比を **Fig. 3** 下部に示す．

①**Fig. 3(a)**：ボクセル当りのプリミティブ（三角ポリゴン）数を10個に固定して，物体空間全体におけるプリミティブ数を500～2500個の範囲で変化させた．

②**Fig. 3(b)**：ボクセル当りのプリミティブ（2次曲面）数を1個に固定して，物体空間全体におけるプリミティブ数を2000～10000個の範囲で変化させた．

③**Fig. 3(c)**：物体空間のプリミティブ（2次曲面）数を600個に固定して，ボクセル当りのプリミティブ数を1～8個の範囲で変化させた．

Fig. 3 から，各ステージの負荷に関して以下のことがわかる．

(1)　物体空間全体のプリミティブ数の変化による影響

①ボクセル当りのプリミティブ数が多い場合（**Fig. 3(a)** 参照）：常に，交点計算，物体探索，輝度計算の順に負荷が大きい．物体空間全体のプリミティブ数が500個と少ないときのステージ負荷の比は，

物体探索：交点計算：輝度計算＝14：18：1
＝1：1.3：0.07

となり，物体探索ステージと交点計算ステージとの間にはそれほど負荷に不均衡がない．しかし，プリミティブ数の増加に伴い，不均衡の度合が大きくなる．プリミティブ数が2500個のときは，

物体探索：交点計算：輝度計算＝5：20：1
＝1：4：0.2

と，交点計算ステージの負荷が突出してステージ負荷が極めて不均衡となる．

②ボクセル当りのプリミティブ数が少ない場合（**Fig. 3(b)** 参照）：物体空間のプリミティブ数が少ないときは，物体探索，交点計算，輝度計算の順に負荷が大きいが，多くなると物体探索と交点計算の順序が入れ替わる．しかしながら，ボクセル当りのプリミティブ数が多い場合（**Fig. 3(a)** 参照）に比較して，物体探索ステージと交点計算ステージの負荷にそれほど不均衡が生じない．たとえば，最も負荷の不均衡が大きいときでも（プリミティブ数10000個），

物体探索：交点計算：輝度計算＝3：5：1
＝1：1.8：0.34

程度である．また，プリミティブ数が4000個のとき，

物体探索：交点計算：輝度計算＝4：4：1
＝1：1：0.25

と，物体探索ステージと交点計算ステージとの間の負荷の均衡はとれる．

(2)　ボクセル当りのプリミティブ数の変化による影響（**Fig. 3(c)** 参照）

ボクセル当りのプリミティブの数が少ないとき，物体探索，交点計算，輝度計算の順に負荷が大きいが，多くなると物体探索と交点計算の順序が入れ替わる．最も負荷の不均衡が大きいのは，ボクセル当りのプリミティブ数が1個のときで，

物体探索：交点計算：輝度計算＝5：2：1
＝2.5：1：0.5

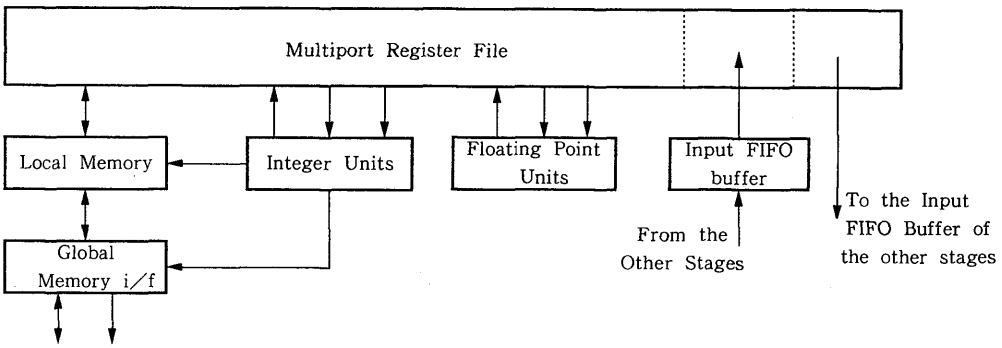


Fig 4　Data Path.

となる．また，プリミティブ数が3個のとき，

物体検索：交点計算：輝度計算＝4：4：1
＝1：1：0.25

と，物体探索ステージと交点計算ステージとの間の負荷の均衡はとれる．

5.2　負荷の均衡化

前節で見たように，物体探索ステージと交点計算ステージが処理時間全体に占める比率は，物体空間全体のプリミティブ数およびボクセル当りのプリミティブ数に強く影響を受ける．一方，輝度計算ステージの処理時間比率は，これらの要因に無関係にほぼ10％程度である．したがって，輝度計算ステージの処理時間を1としたときの物体探索および交点計算ステージの処理時間の比は，それぞれ2.5～14および2～26と大きく変化する（**Fig. 3** 下部参照）．このように，物体探索ステージおよび交点計算ステージをどの程度並列化（細分化ないし多重化）すればよいかは一概に決められない．

Fig. 3 上部に，各ステージの並列度比を

- 物体探索：交点計算：輝度計算＝1：1：1
- 物体探索：交点計算：輝度計算＝4：4：1

としたときのマクロパイプライン処理時間を示す．単純にマクロパイプライン化しただけの並列度比1：1：1における対逐次処理性能向上比は，1.2～2.3である．一方，並列度比4：4：1における性能向上比は，5.1～8.1とかなり高い．

さて，物体探索ないし交点計算ステージに関して，その対輝度計算ステージ並列度比をpとする，つまり，pだけ並列化する方法を考えよう．これには，一般に次の2通りの方法がある．

①細分化：ステージをp分割して，p個のサブステージに細分化する．

②多重化：同一ステージをp個設ける．

Fig. 2 に示したように，物体探索ステージおよび交点計算ステージのいずれの処理も，ステージ内にループを含んでいる．このようなステージを細分化するのは容易ではない．また，細分化されたp個のサブステージ間に新たなFIFOバッファまたはラッチが必要となり，ハードウェア量の増加を招く．また，ステージの多重化も，明らかにハードウェア量が元のp倍以上に増加する．

『熱視線』では，細分化や多重化は行わない．よって，**Fig. 1** において各ステージの多重度を表わしているs，t，uはすべて1となる．かわりに，命令レベルの並列度を活用する．次章で述べるように，物体探索および交点計算ステージにおける命令レベル並列処理により，実質的に並列度比4：4：1を実現している

6.　ステージの構成

6.1　概要

個々のステージを構成するハードウェアは，専用のプログラムおよびプログラム・カウンタを有する独立したプロセッサである．物体探索ステージおよび交点計算ステージは独自のVLIWプロセッサで，一方，輝度計算ステージは一般の汎用32ビット・マイクロプロセッサにより構成する．以下，物体探索および交点計算ステージを担当するVLIWプロセッサについて述べる．

Fig. 4 に，そのデータパスを示す．若干の相違を除いて，その基本構成は物体探索と交点計算ステージの

双方で共通である．主な諸元は，次の通りである．

①大容量のマルチポート・レジスタファイルにより，整数演算ユニット，浮動小数点演算ユニット，ローカル・メモリ，入力 FIFO バッファ（物体探索 FIFO バッファないし交点計算 FIFO バッファ），出力 FIFO バッファ（2個）を接続する．ポート数は，

a）物体探索ステージ：19ポート

b）交点計算ステージ：17ポート

である．

②整数演算ユニットは，整数論理演算およびメモリ・アクセスを行う．整数演算ユニット内には，

a）物体探索ステージ：2個の汎用機能ユニット

b）交点計算ステージ：1個の汎用機能ユニット

を備える．

③浮動小数点演算ユニット内には，

a）物体探索ステージ：2個の汎用機能ユニット

b）交点計算ステージ：3個の汎用機能ユニット

を備える．

④レジスタファイルには，128個の32ビット・レジスタを備える．このうち，2個のレジスタは，出力 FIFO バッファの最後尾エントリに相当する．また，

a）物体探索ステージ：3個のレジスタ

b）交点計算ステージ：1個のレジスタ

が，入力 FIFO バッファの先頭エントリに相当する．これらのレジスタから／への読出し／書込みは，自動的に入力／出力 FIFO バッファから／への読出し／書込みとなる．

⑤命令長は128ビットで，分岐および FIFO アクセスに加えて，以下の操作を同時に行うことが出来る．

a）物体探索ステージ：

- 2つの整数論理演算，あるいは，1つの整数論理演算と1つのローカルメモリ・アクセス（1ないし2語ロード，あるいは，1語ストア）またはグローバルメモリ・アクセス（ブロック転送起動）
- 2つの浮動小数点演算

b）交点計算ステージ：

- 1つ整数論理演算，あるいは，1つのローカルメモリ・アクセス（1ないし2語ロード，あるいは1語ストア）またはグローバルメモリ・アクセス（ブロック転送起動）
- 3つの浮動小数点演算

以上の諸元で物体探索ステージと交点計算ステージとの間に相違が生じた理由については，6.2節および6.3節で述べる．

6.2 物体探索ステージ

物体探索ステージ処理の静的ステップ数の内訳は，次のようになる．

①光線データの物体探索 FIFO バッファからのロード：25

②初期化：60

③ボクセル探索（1繰返し分）：23

④光線データの交点計算または輝度計算 FIFO バッファへのストア：25

(1) 初期化および光線データのロード／ストア

上記のうち最も静的ステップ数の多いのは，②の初期化である．初期化を行う方法には，次の2つの選択肢がある．

ⓐ光線が物体探索ステージに入る度に行う．したがって，毎回初期化のために60ステップを要する．

ⓑ光線が物体探索ステージに最初に入ったときだけ行う．このとき，初期化結果を光線データに添付するため，光線データ量が元の10語に対して25語と15語増加する．①および④のステップ数が25となるのは，このためである．しかし，2個の光線間でこのロード／ストア処理（出て行く光線の④と入って来る光線の①）がオーバラップ出来れば，実質的には5ステップの増加に留まる．

FIFO バッファーレジスタ間ロード／ストアのオーバラップ処理をハードウェア構成上可能として，選択肢ⓑを採用した．

(2) ボクセル探索

動的ステップが最も多くなるのは，③のボクセル探索である．たとえば，物体空間を1辺あたり30個のボクセルに分割すると，最大約50回ループを繰返すので動的ステップ数は最大約1000になる．したがって，このボクセル探索部分のコードを出来るだけ圧縮する必要がある．コード圧縮により，元々23ステップだったのが VLIW 命令では14ステップになる．これをさらにソフトウェア・パイプライン化[5]すると8ステップとなる．つまり，1繰返し当りのステップ数が23から8へと1/3に減り，ほぼ3倍の性能向上が得られる．

6.3 交点計算ステージ

交点計算ステージ処理の静的ステップ数の内訳は，

次のようになる.

①光線データの交点計算 FIFO バッファからのロード：25

②プリミティブ形状データのローカル・メモリからのロード（1繰返し分）：30

③交点計算［三角ポリゴン］（1繰返し分）：120,
交点計算［2次曲面］（1繰返し分）：130

④法線ベクトル計算［三角ポリゴン］：10,
法線ベクトル計算［2次曲面］：45

⑤光線データの物体探索 FIFO バッファへのストア：25,
光線データの輝度計算 FIFO バッファへのストア：15

交点計算は，三角ポリゴンおよび2次曲面ともに3次元ベクトルに対する浮動小数点演算が大部分を占める．法線ベクトル計算も同様である．よって，交点計算および法線ベクトル計算においては，3次元ベクトルのそれぞれの次元に関する浮動小数点演算を3つ同時に行うことが可能である．よって，交点計算ステージでは，浮動小数点演算ユニット内に3個の汎用機能ユニットを備えている．これにより，たとえば，平面を表わす式の係数を求める計算に47ステップ要していたのが17ステップと1/3に減少する．また，2次曲面の交点を与える2次方程式の係数を求める際，68ステップを要していたのが23ステップへと同じく1/3に減少する．

6. おわりに

以上，『熱視線』の設計方針，マクロパイプライン方式および処理，各ステージの処理，負荷および構成について述べた．

本稿の結論を以下に示す．

①各ステージの負荷率は物体空間全体のプリミティブ数およびボクセル当りのプリミティブ数に強く影響を受ける．

②各ステージの並列化（細分化ないし多重化）は一概に決められない．

③しかしながら，4：4：1に並列化すれば性能は5.1～8.1とかなり高い．

現在，各ステージの詳細設計を進めている．今後は，最上位のマクロプロセッサ・レベルに関して検討を進めていく予定である．特に，メモリ構成法，データベース分割法，光線分配法，等に対する検討および評価を計画している．

参考文献

1) 權ほか："レイトレーシング法を高速処理する専用並列レンダリング・マシンの構想"，情報40全大論文集，2P－4（1990年3月）.

2) 石井光雄：映像化マシン，pp. 116－120，オーム社，1989年.

3) 出口ほか："コンピュータグラフィックシステム LINKS-1 における画像生成の高速化手法"，情処論，vol. 25, no. 6, pp. 944－952（1984年11月）.

4) 吉田ほか："グラフイックス計算機 SIGHT の基本構成"，情処研報，85-CA-60-4（1985年12月）.

5) Charlesworth, A. E.: "An Approach to Scientific Array Processing: The Architectural Design of the AP-120B/164 Family", IEEE Computer, vol. 14, no. 9, pp. 18－27, Sept. 1981.

6) Dippe, M., et al.: "An Adaptive Subdivision Algorithm and Parallel Architecture for Realistic Image Synthesis", Proc. SIGGRAPH'84, pp. 149－158, July 1984.

7) Fujimoto, A., et al.: "ARTS: Accelerated Ray-Tracing Systems", IEEE Computer Graphics & Applications, vol. 6, no. 4, pp. 16－26, Apr. 1986.

8) Gaudet, S., et al.: "Multiprocessor Experiments for High-Speed Ray Tracing", ACM Trans. Graphics, vol. 7, no. 3, pp. 151－179, July 1988.

9) Glassner, A. S.: "Space Subdivision for Fast Ray Tracing", IEEE Computer Graphics & Applications, vol. 4, no. 10, pp. 15－22, Oct. 1984.

10) Green, S. A., et al.: "Exploiting Coherence for Multiprocessor Ray Tracing", IEEE Computer Graphics & Applications, vol. 9, no. 6, pp. 12－26, Nov. 1989.

11) Synder, J. M., et al.: "Ray Tracing Complex Models Containing Surface Tessellations", Proc. SIGGRAPH'87, pp. 119－128, July 1987.

12) Weghorst, H., et al.: "Improved Computational Methods for Ray Tracing", ACM Trans. Graphics, vol. 3, no. 1, pp. 52－69, Jan. 1984.

13) Whitted, T.: "An Improved Illumination Model for Shaded Disply", Comm. ACM, vol. 23, no. 6, pp. 343－349, June 1980.