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『熱視線』の要素プロセッサ・アーキテクチャ

——マクロパイプライン・アーキテクチャおよび性能評価——

権 五 鳳[†] 村 上 和 彰[††]
村 田 誠 治[††] 富 田 眞 治[†††]

筆者らは，レイトレーシング (ray tracing) 法を高速処理する専用並列レンダリング・マシン『熱視線』を開発している．『熱視線』は，マルチプロセッサ処理，マクロパイプライン処理，および，命令レベル並列処理の3レベルの並列処理を活用し高速化を図っている．すなわち，システム全体をマルチプロセッサ構成として画面分割型の並列処理を行い，次に個々の要素プロセッサを3ステージのマクロパイプライン構成として1本の光線に対する機能分割型のオーバラップ処理を行い，そして各ステージのプロセッサに VLIW (超長形式機械命令) アーキテクチャを採用している．本論文では，『熱視線』の要素プロセッサ・アーキテクチャ，特にマクロパイプライン・アーキテクチャについて述べている．最初に，物体探索，交点計算，および，輝度計算の3ステージでレイトレーシングをマクロパイプライン処理する 3MPRT (3-stage MacroPipelined Ray-Tracing) アルゴリズムを提案している．そして，3MPRT アルゴリズムを直接実装したプロセッシング・エレメントである 3MPPE (3-stage MacroPipelined Processing Element) について，構成，実行過程，性能を述べている．性能評価の結果，マクロパイプライン処理により最低 4%～最高 82% の性能向上が可能であることを示している．

Processing Element Architecture of a Parallel Rendering Machine for High-Speed Ray-Tracing

——Macropipeline Architecture and Preliminary Performance Evaluation——

OUBONG GWUN,[†] KAZUAKI MURAKAMI,[††] SEIJI MURATA[††] and SHINJI TOMITA[†††]

This paper introduces a parallel rendering machine for high-speed ray-tracing, which machine exploits three levels of parallel processing: i) multiprocessing at the system level, ii) macropipelining at the processing-element level, and iii) instruction-level parallel processing at the stage level. This paper focuses on macropipelining at the processing-element level. First, a ray tracing algorithm, called *3 MPRT (3-stage macropipelined ray tracing)* is presented to show how the ray-tracing task is divided into three subtasks: *OS* (object search), *IC* (intersection calculation), and *SC* (shading calculation). The paper then describes *3 MPPE (3-stage macropipelined processing element)* architecture which implements 3 MPRT directly. Both 3 MPRT and 3 MPPE are evaluated through simulation, varying the benchmark scene characteristics and the object-space subdivision. From the simulation results, the paper concludes that a single 3 MPPE is 1.04–1.82 times faster than a non-macropipelined implementation of 3 MPRT.

† 九州大学工学部情報工学科
Department of Computer Science and Communication Engineering, Faculty of Engineering, Kyushu University
†† 九州大学大学院総合理工学研究科情報システム学専攻
Department of Information Systems, Interdisciplinary Graduate School of Engineering Sciences, Kyushu University
††† 京都大学工学部情報工学科
Department of Information Science, Faculty of Engineering, Kyoto University

1. はじめに

レイトレーシング (*ray tracing*：光線追跡，視線探索) 法[24]は，反射，屈折，透過，影などの表現に優れ，現実感の高い3次元映像が生成可能なアルゴリズムである．しかしながら，計算量が非常に多いという短所があり，高速性を必要とする分野においてはあまり実用化されていない．Whitted[24] によれば，レイトレーシング法の処理時間の 75%～95% は光線と物体（オブジェクト）との交差判定に費やされている．し

たがって，レイトレーシング法の高速化にあたっては，①交差判定回数の削減，および，②交差判定処理自体の高速化・高スループット化が重要であり，以下の種々のアプローチが考案されている[2),5)]．

①交差判定の回数を削減する：

（a） 交差判定すべき光線の数を削減する：

ⅰ）Zバッファ併用法[23)]

ⅱ）画素選択型光線追跡法[1)]

ⅲ）寄与率法[19)]

（b） 1本の光線に関する交差判定回数を削減する：

ⅰ）外接体 (bounding volume) 法[23)]

ⅱ）ライト・バッファ (light buffer) 法[17)]

ⅲ）オクトリー (octree) 分割法[14)]

ⅳ）ボクセル (voxel) 分割法（空間等分割法）[12),21)]

②交差判定処理自体を高速化・高スループット化する：

（a） レイトレーシング法のアルゴリズム自体に内在する空間的並列性を活用して，負荷分散型のマルチプロセッサ処理を行う：

ⅰ）オブジェクト空間分割法[11),22)]

ⅱ）画面（イメージ空間）分割法：

A．データベース複写型[6),7)]

B．データベース共有型[13),15)]

（b） レイトレーシング法のアルゴリズム自体に内在する時間的並列性を活用して，機能分散型のマクロパイプライン処理を行う[13)]

（c） 交差判定計算に命令レベル並列処理を適用する[3),10)]

上記の交差判定削減法のうち，外接体法およびオクトリー分割法はメモリ利用率は高いものの探索時間が長くなる傾向にあり，ボクセル分割法はそれとは逆の特徴を有している[21)]．このように，各方法ともに一長一短があり，どれか1つの方法が多数を占める状況には至っていない．一方，交差判定処理の高速化・高スループット化の方法としては，マルチプロセッサ処理を採用したものが圧倒的に多い．しかしながら，マルチプロセッサ処理も万全ではなく（たとえば，オブジェクト空間分割法では負荷分散の難しさ，また，画面分割法ではデータベース共有／分散の難しさ，といった課題がある），単にプロセッサ数を増加させて高速化を図るというのも容易ではない．そこで，プロセッサ単体の高速化も相変わらず重要な課題となっている．

我々は以上の状況に鑑み，上記の高速化技法のうち次のものを採用した，レイトレーシング専用並列レンダリング・マシン『熱視線』を開発している[4),9),16)]．

- ①-a-ⅲ **寄与率法**：反射，屈折などの2次光線に対して，対応するピクセルの輝度値への寄与率があるしきい値以下になった場合，光線追跡を打ち切る．本方法は実装が容易なので，これを採用した．
- ①-b-ⅳ **ボクセル分割法**：オブジェクト空間をボクセルに等分割し，ボクセルごとにその内部に存在する物体の情報を付加する．交差判定回数が減少する反面ボクセルのトラバース処理が増加するが，これは3DDDA (3 Dimensional Digital Differential Analyzer)[21)] により高速化する．
- ②-a-ⅱ-B **画面分割（データベース共有）型マルチプロセッサ処理**：画面をピクセル単位で分割し，個々のピクセルに関する処理を各要素プロセッサに割り当てる．各要素プロセッサは，他の要素プロセッサとはまったく独立に処理を進めることができる．なお，データベースを要素プロセッサごとに複写するのは現実的ではないので，これを全要素プロセッサで共有させるようにする．
- ②-b **マクロパイプライン処理**：さらに，要素プロセッサ自身の処理能力（スループット）を向上させるために，これを3ステージにマクロパイプライン化する．
- ②-c **命令レベル並列処理**：マクロパイプラインの各ステージの負荷を均衡化するために，負荷の大きいステージを担当するプロセッサにVLIW (Very Long Instruction Word) アーキテクチャを採用する．

本論文では，『熱視線』の要素プロセッサ・アーキテクチャ，特にマクロパイプライン・アーキテクチャについて述べる．まず第2章で，我々が開発した3MPRT アルゴリズムと呼ぶレイトレーシング法について概説する．第3章では，『熱視線』の全体構成，および，『熱視線』の要素プロセッサで3MPRT アルゴリズムを直接実装した3MPPE と呼ぶプロセッシング・エレメントについて述べる．第4章で3MPPE 上での 3MPRT アルゴリズムの実行過程を詳述し，第5章でその性能をシミュレーションにより評価する．

2. 3 MPRT アルゴリズム

レイトレーシング法の個々の光線処理に関するタスクは，まず交差判定と輝度計算の2個のサブタスクに分割できる．『熱視線』では，1本の光線に関する交差判定回数を削減するために，ボクセル分割法[12),21)]を採用している．これにより，交差判定サブタスクは，さらに物体探索と交点計算の2個のサブタスクに分割可能となる．すなわち，1本の光線の処理は，①物体探索，②交点計算，③輝度計算の計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．そこで，**図1**に示すように，各サブタスクを1ステージとし，3ステージのマクロパイプラインにより1本の光線を処理する 3 MPRT (3-stage MacroPipelined Ray-Tracing) アルゴリズムを開発した[16)]．本アルゴリズムでは，光線は基本的に①物体探索→②交点計算→③輝度計算へと順にステージを進めながら個別に処理される．ステージ間にはFIFOバッファが設けられており，各ステージの処理は独立したプログラムにより非同期に行われる．

3 MPRT アルゴリズムの記述の前に，まず用語および使用するデータベースの定義を行う．

2.1 用　　語

3 MPRT アルゴリズムで用いる用語を以下のように定義する．

- 光線：始点および方向から成るベクトル．
  - —1次光線 (initial ray)：画面の各ピクセルに対応する光線．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．
  - —分岐光線 (secondary 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る．
  - —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射出される光線．輝度計算ステージにおいて生成される．
  - —戻り光線(return ray)：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戻る光線．これには，上記いずれの光線もなり得る．
- プリミティブ：物体（オブジェクト）を構成する基本要素（ポリゴン，球，円錐，ベジェー曲面，等）．
- ボクセル：オブジェクト空間を等間隔分割して得られる直方体．
- 寄与率：対応するピクセルの輝度値に光線が寄与する度合．

2.2 データベース

3 MPRT アルゴリズムは，以下の4種類のデータベースを用いる．光線処理過程では，いずれも read-only である．各データベースは，対応するステージに占有される．

- 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．
- プリミティブ・リスト：交点計算ステージで用いる．プリミティブが存在するボクセル対応に設けられる単方向リストである．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．
- 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．
- プリミティブ輝度データ：輝度計算ステージで用いる．プリミティブに対応に設けられ，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．

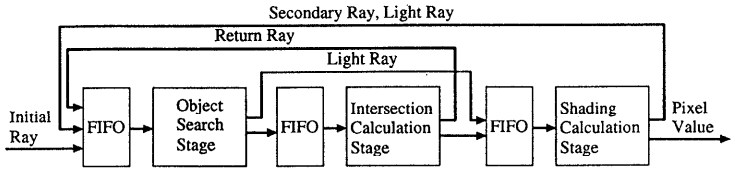


図 1 マクロパイプラインの論理構造
Fig. 1 Logical structure of macropipeline.

2.3 アルゴリズムの概要

3 MPRT アルゴリズムにおける各光線の処理過程は，以下のようになる．詳細は，第4章にて述べる．

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と，光線の進行方向に沿ってオブジェクト空間をトラバースする．
- 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，当該ボクセルの識別子を光線に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，次のボクセルへ進む．

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全プリミティブの形状データを読み出し，光線とプリミティブとの交点計算を行う．
- 1次光線ないし分岐光線があるプリミティブと交差する場合，最も視点に近い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．
- 影光線があるプリミティブと交差する場合，その影光線は消滅する．
- いずれの光線もプリミティブと交差しなかった場合は，戻り光線として物体探索ステージへ戻される．

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み出し，輝度計算を行う．
- 影光線を生成して，物体探索ステージへ進ませる．
- 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進ませる．

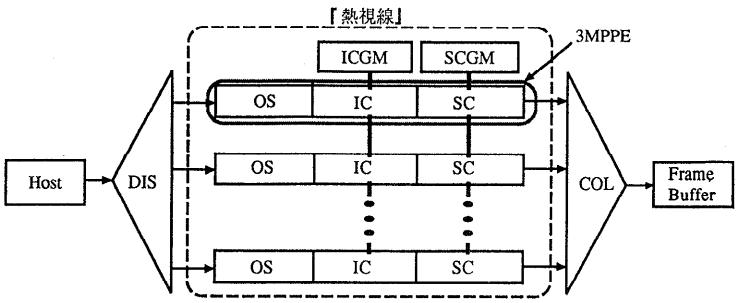
そして，元の光線は消滅する．

3. 要素プロセッサ・アーキテクチャ

3.1 『熱視線』の全体構成

図2に『熱視線』の全体構成を示す．フロントエンドにホスト・プロセッサを，また，バックエンドにフレーム・バッファを備える．ホストプロセッサは，ボクセル・データの生成，1次光線の生成，等の前処理を担当する．3 MPRT アルゴリズムを直接実装したプロセッシング・エレメント(3 MPPE : 3-stage MaroPipelined Processing Element)を要素プロセッサとするメモリ共有型マルチプロセッサである．構成要素は，以下のとおりである．

- 負荷分配器 (DIS)：1次光線生成に必要なデータ(負荷) を 3 MPPE に分配する．また，ホストによる動的負荷分散・均衡化を可能とするために，各 3 MPPE の負荷を監視し，それをホストに伝播する機能を有する．
- ピクセル輝度値コレクタ (COL)：各 3 MPPE の光線処理結果（ピクセル輝度値）を集めて，随時フレーム・バッファに書き込む．
- グローバル・メモリ：4種類のデータベース (2.2 節参照) のうち，交点計算ステージおよび輝度計算ステージに占有される3種類のデータベースを，それぞれステージ固有のグローバル・メモリに格納する (3.2 節参照)．
  - —交点計算用グローバル・メモリ (ICGM)：プリミティブ・リストおよびプリミティブ形状データを格納する．全 3 MPPE の交点計算プロセッサ (IC) に共有される．
  - —輝度計算用グローバル・メモリ (SCGM)：プリミティブ輝度データを格納する．全 3 MPPE の輝度計算プロセッサ (SC) に共有される．
- プロセッシング・エレメント (3 MPPE)：**図3**に 3 MPPE の構成を示す．3 MPRT アルゴリズム



3MPPE: 3-stage Macro Pipelined Processing Element
COL: Pixel Value Collector
DIS: Load Distributer
IC: Intersection Calculation Processor
ICGM: Intersection Calculation Global Memory
OS: Object Search Processor
SC: Shading Calculation Processor
SCGM: Shading Calculation Global Memory

図 2 『熱視線』の全体構成
Fig. 2 System configuration.

を直接実装するため，3ステージのマクロパイプライン構成を採っている．個々のステージは，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．各プロセッサは，以下の構成方式を採る．

—物体探索プロセッサ (OS)：整数演算ユニット2個および浮動小数点演算ユニット2個を有する VLIW プロセッサ．

—交点計算プロセッサ (IC)：整数演算ユニット1個および浮動小数点演算ユニット3個を有する VLIW プロセッサ．

—輝度計算プロセッサ (SC)：汎用機能ユニット1個を有する逐次プロセッサ．

さらに，各プロセッサは，データベース格納用のローカル・メモリを有する (3.2節参照)．これらプロセッサはFIFOバッファを介してリング状に接続されており，プロセッサ間通信をプロセッサとは独立に制御するための通信ユニットを備える (3.3節参照)．

次節以降，3MPPE のメモリおよび通信アーキテクチャについて述べる．

3.2 メモリ・アーキテクチャ

2.2節で述べた4種類のデータベースのメモリ配置にあたっては，以下の方針を採った．

3.2.1 ボクセル・データ

ボクセル・データのデータ量は，ボクセルの数，つまりオブジェクト空間の分割数に依存する．この空間分割数は任意に決めることができる．したがって，ボクセル・データに関しては，グローバル・メモリを設けずに，各 3MPPE の物体探索プロセッサ (OS) のローカル・メモリにその完全なコピーを格納する．このローカル・メモリの容量を上限として，その範囲内で空間分割数を調整する．

3.2.2 プリミティブ関連データベース

プリミティブ・リスト，プリミティブ形状データ，および，プリミティブ輝度データのデータ量は，モデリング・データの数に比例して増加する．したがって，ボクセル・データのように完全なコピーを各 3 MPPE のローカル・メモリに格納するのは困難である．

そこで，プリミティブ・リストおよびプリミティブ形状データについては交点計算用グローバル・メモリ (ICGM)，また，プリミティブ輝度データについては輝度計算用グローバル・メモリ (SCGM) と呼ぶ大容量のグローバル・メモリをそれぞれ設け，これに各データベースを格納することにした．すなわち，各データベースはグローバル・メモリ中にしか完全なコピーが存在せず，それが全 3MPPE に共有される．

このとき，グローバル・メモリへのアクセス時間が問題となる．レイトレーシング法の光線処理過程においても，データ参照の局所性があることは知られている[15]．よって，キャッシュ・メモリを導入することにより，実効メモリ・アクセス時間の低減が期待できる．しかも，光線処理過程では，すべてのデータベー

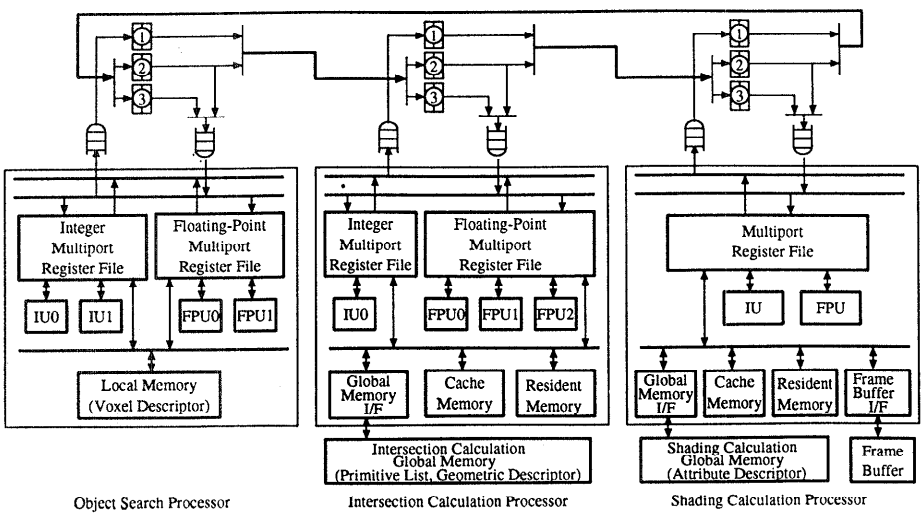


図 3　マクロパイプラインの物理構造
Fig. 3　Physical structure of macropipeline.

スがread-onlyなので，キャッシュの一貫性は保証する必要もない．

くわえて，光線処理においては，他のデータに比べて参照頻度の高いデータが存在する．これは，映像として表示されるのは視点から見える物体のプリミティブであることから，容易に想像できる．このような参照頻度の高いデータは，キャッシングされると同時にリプレース・アウトされないようにする必要がある．つまり，キャッシュ・メモリ内に常駐させた方がよい．

以上の特徴から，交点計算プロセッサ（IC）および輝度計算プロセッサ（SC）それぞれに対しては，最終的に次の2階層3種類のメモリから成る構成を採った．

- ローカル・メモリ：第1層メモリ．
  - —レジデント・メモリ：参照頻度の高いデータを前処理過程で選択し[20]，それの完全な同一コピーを各3MPPEのレジデント・メモリに格納する．これらのデータのコピーは，グローバル・メモリには格納する必要がない．
  - —キャッシュ・メモリ：グローバル・メモリから必要なデータのコピーを適宜キャッシングする．
- グローバル・メモリ：第2層メモリ．レジデント・メモリに格納すべきデータ以外をすべて格納する．

3.3 通信アーキテクチャ

図1に示したマクロパイプライン構成においては，以下の計5種類のステージ間通信，つまり，プロセッサ間通信がプロセッシング・エレメント内で起きる．

- 物体探索→交点計算
- 物体探索→輝度計算
- 交点計算→輝度計算
- 交点計算→物体探索
- 輝度計算→物体探索

これら5種類のプロセッサ間通信を直接的に実現しようとしたら，送信バッファ5個，受信バッファ5個，および，これらを1対1接続するプロセッサ間単方向リンク5本と，多くのハードウェアが必要となる．そこで，実際には図3に示したように，プロセッサ間を1本のリング網で結合する．

デッドロック・フリー，高スループット，低レイテンシといった要件を満たすために，文献8）で提案されている“ワームホール・ルーティング＋構造化バッファ・プール”方式を採用している．送信FIFOバッファ3個，受信FIFOバッファ3個，小量のフリット・バッファ9個，プロセッサ間単方向リンク3本を備える．フリット長は語長4バイトに等しく，各リンクの転送幅はフリット長に等しい．

光線データ・パケットは，次の3種類のフリットから成る．

- ヘッダ・フリット（1個）：パケットの宛先を指定する．
- データ・フリット（任意個）：光線データ．
- テール・フリット（1個）：パケットの終了を示す．

各プロセッサの通信ユニットには，小量のフリット・バッファを3個ずつ設ける．これらは，図3に示すように，クラス①，②，③がそれぞれ割り当てられる．次ステージにフリットを転送する際には，クラス①→②，あるいは，クラス②→③という具合に，1クラス上のフリット・バッファ宛に転送する．

プロセッサ間通信は，以下の手順で行う．

①通信を行いたいプロセッサは，光線データにヘッダおよびテールをつけてパケットを構成し送信FIFOバッファに入れる．通信ユニットは，クラス①のフリットバッファ経由でフリットを送り出す．

②ヘッダ・フリットを受信した通信ユニットは，宛先が自分宛か否かを判断する．
- 自分宛の場合：パケットを格納するために，受信FIFOバッファへの排他的書込み権を確保する（フリットはインタリーブできないため，排他的書込みとなる）．
- 自分宛でない場合：当該ステージは中継ステージとなる．ヘッダ・フリットを次ステージに転送し，転送経路を設定する．

③ヘッダ・フリットが確立した転送経路を後続のフリットは転送されていく．途中，フリット転送が継続できない状況（受信FIFOバッファへの排他的書込み権を確保できない，受信FIFOバッファが一杯である，等）が生じた場合は，プロセッサ間リンクをいったん解放する．

④テール・フリットを検出したら，転送経路を解除する．

4. マクロパイプライン処理

3MPRTアルゴリズムの処理の流れを**図4**に示す．

以下，3 MPPE 上での 3 MPRT アルゴリズムの実行過程を詳述する．

4.1 物体探索ステージ

物体探索プロセッサ (OS) は，以下の処理を行う．

①受信 FIFO バッファから，光線データを読み出す．

②3 DDDA[21] アルゴリズムにより，光線が辿るボクセルを決定する．

③そのボクセル・データを読み出し，プリミティブの有無を調べる．プリミティブが存在すれば，光線データにボクセル識別子を添付して交点計算プロセッサへ送る．

④プリミティブがない場合は，上記②，③を繰り返す．もし，光線がオブジェクト空間の終りに達したなら，以下のように処理する．

- 1次光線ないし分岐光線は，消滅して処理終了．
- 影光線は，輝度計算プロセッサへ送る．

4.2 交点計算ステージ

交点計算プロセッサ (IC) は，以下の処理を行う．

①受信 FIFO バッファから，光線データを読み出す．

②ボクセル識別子を用いてプリミティブ・リストを読み出し，さらに，リスト・セル内のポインタを用いてプリミティブ形状データを読み出す．

③形状データの係数値を用いて交点計算を行う．

④ボクセル内の全プリミティブに対して，上記②，③を繰り返す．

⑤交点が存在する場合，以下のように処理する．

- 1次光線ないし分岐光線は，最も視点に近い交点の法線ベクトルを求める．光線データに，交点ベクトル，法線ベクトル，および，プリミティブ識別子を添付して，輝度計算プロセッサへ送る．
- 影光線は，消滅して処理終了．

⑥交点が存在しない場合は，当該光線を戻り光線として物体探索プロセッサに送り返す．

4.3 輝度計算ステージ

輝度計算を行うと同時に，分岐光線および影光線を生成する．生成した光線に対して，3 DDDA アルゴリズム用に光線データを初期化する．

寄与率法[19]により，分岐光線および影光線の生成を制御する．1次光線の寄与率は1，分岐光線の寄与率は元の光線の寄与率に反射率ないし透過率を乗じたものである．

輝度計算プロセッサ (SC) は，以下の処理を行う．

①受信 FIFO バッファから，光線データを読み出す．

②プリミティブ識別子を用いてプリミティブ輝度データを読み出す．

③以下の輝度計算を行い，フレーム・バッファ内の対応するピクセルに輝度を加える．

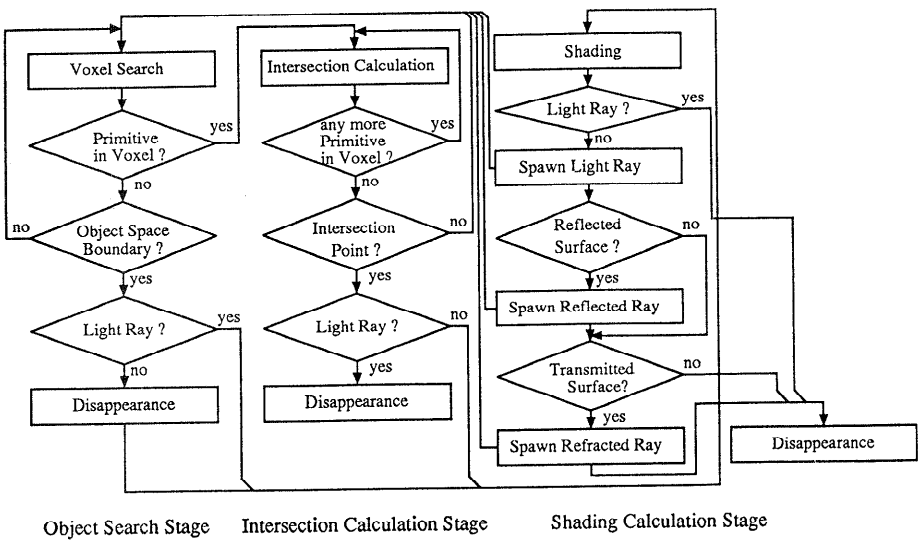


図 4 3 MPRT アルゴリズム
Fig. 4 3 MPRT algorithm.

- 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率を乗じて，対応するピクセルの輝度値に加算する．
- 影光線：光線データに添付されている輝度成分を対応するピクセルの輝度値に加算する．影光線自身は，消滅して処理終了．

④1次光線ないし分岐光線に対して，交点から光源に向かう影光線を生成する．輝度成分（法線ベクトルと方向ベクトルの内積に，拡散反射率および元の光線の寄与率を乗じたもの）を計算して光線データに添付し，物体探索プロセッサへ送る．このとき，輝度成分があるしきい値以下となったら，当該影光線の生成を取り止める．

⑤1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．元の光線の寄与率に反射率ないし透過率を乗じて新しい寄与率を求め，これを光線データに添付して物体探索プロセッサへ送る．このとき，求めた寄与率があるしきい値以下となったら，当該分岐光線の生成を取り止める．

⑥元の1次光線ないし分岐光線は，消滅して処理終了．

なお，図4には示してないが，輝度計算プロセッサでは，1次光線の生成，および，その光線データの初期化も行う．

5. 性能評価

3MPPE で採用したマクロパイプライン・アーキテクチャについて，ソフトウェア・シミュレータを用いて性能評価を行った．

5.1 評価方法

本シミュレータは 3MPRT アルゴリズムを忠実に実行し，3MPPE の動作を機能レベルで近似的にモデル化している（後述）．評価モデルとしては，マクロパイプライン・アーキテクチャの採用の有無に応じて次の2つを用いた．

- $\overline{MACRO}\cdot\overline{VLIW}$ (S)：マクロパイプライン・アーキテクチャおよび VLIW アーキテクチャともに採用していない，一般的な逐次プロセッサ・モデル．レジスタ数は，128 個．
- $MACRO\cdot\overline{VLIW}$ (M)：マクロパイプライン・アーキテクチャを採用した 3MPPE．ただし，物体探索プロセッサおよび交点計算プロセッサは VLIW アーキテクチャではなく，一般的な逐次プロセッサである．つまり，3つのプロセッサは同一構造を採る．各々，128 個のレジスタを備える．

両モデルとも，データ・キャッシュはヒット率 100 %とした．また，評価モデル M については，VLIW アーキテクチャを採用していない点に加えて，通信アーキテクチャが第3.3節で述べた内容と異なる．すなわち，送信 FIFO バッファおよび受信 FIFO バッファを無限長としてオーバフローが生じない点，送信 FIFO バッファの出口から受信 FIFO バッファの入口までの間のネットワーク・レイテンシを零としている点である．

ベンチマーク・シーンとしては，Haines[18] が提案している6シーンのうち**表1**に示した5つを用いた（**図5**参照）．ピクセル数は 512×512 である．

各ベンチマーク・シーンおよび各評価モデルに対して，オブジェクト空間の分割数，すなわち，次元当りの分割数を 20, 50, 100 と変化させて，以下の項目を測定した．

- 映像生成時間（表2参照）：ホストにおける前処理時間（空間分割処理，等）を除いた正味のレイトレーシング実行時間．評価モデル M に対しては，さらに各プロセッサにおける以下の所要時間内訳を調査した．
  - —処理時間：映像生成時間全体から，以下の通信時間および待ち時間を除いた時間．
  - —通信時間：送信 FIFO バッファへの書込み，および，受信 FIFO バッファからの読出しに要した時間．
  - —待ち時間：受信 FIFO バッファが空のために生じた待ち時間．

表1 ベンチマーク・シーン
Table 1 Benchmark scenes.

ベンチマーク・シーン		balls	mount	rings	tetra	tree
プリミティブ数		7382	8196	8401	4096	8191
	ポリゴン	1	8192	1	4096	1
	球	7381	4	4200	0	4095
	円錐	0	0	0	0	4095
	円柱	0	0	4200	0	0
光源数		3	1	3	1	7
反射面		有	有	有	無	無
透過面		無	有	無	無	無

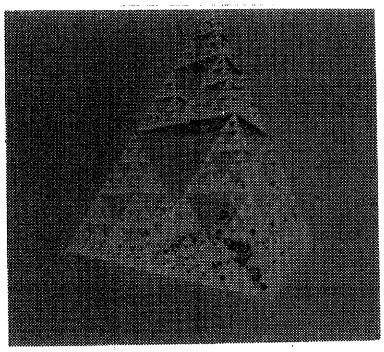
- 光線量（表3参照）：評価モデルMの各プロセッサにおいて，入力／消滅／生成／出力した光線の数．
- トラバースしたボクセル数（表4参照）
- 交点計算回数（表4参照）

5.2 評価結果および考察

表 2～4 および**図 6** に，測定結果を示した．

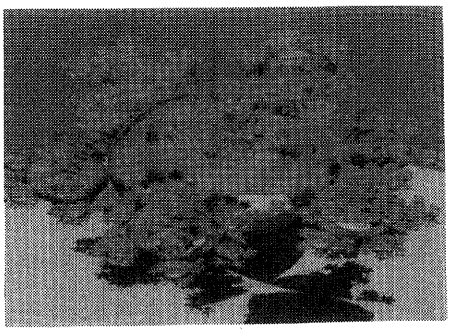
まず，表2および図6に，評価モデルSおよびMに対して，次元当りの分割数を 20，50，100 と変化させた場合の映像生成時間を示す．図6のモデルSの棒グラフの模様は，物体探索，交点計算，および，輝度計算の各処理時間を示している．また，モデルMの方は，マクロパイプライン処理において最も処理時間の大きかったステージを示している．これから，以下のことが判る．

①映像生成時間は，空間分割数に大きく依存する．映像生成時間が最小となる空間分割数はベンチマーク・シーンにより異なるが，両評価モデルで

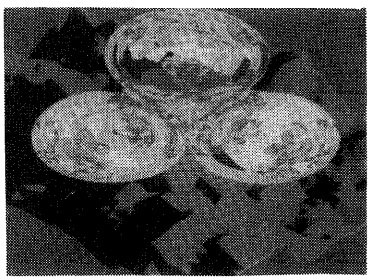


tetra.

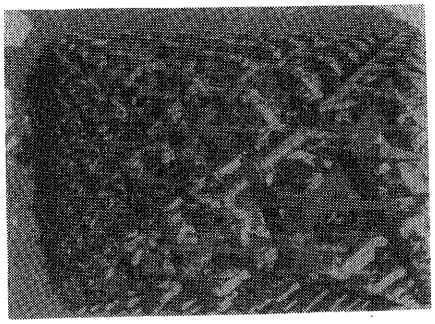
tree.



balls.



mount.



rings.

図 5 ベンチマーク・シーン
Fig. 5 Benchmark scenes.

表 2　映像生成時間［単位：1,000,000 クロック・サイクル］
Table 2　Ray tracing time.

ベンチマーク・シーン			balls			mount			rings			tetra			tree		
次元当り分割数			20	50	100	20	50	100	20	50	100	20	50	100	20	50	100
モデル S			35,173	11,526	7,308	5,711	3,109	3,288	13,930	8,420	7,726	2,096	1,951	2,976	38,552	17,839	10,651
［時間比］†			［100：32：20］			［100：54：57］			［100：60：55］			［90：65：100］			［100：46：27］		
モデル M			33,586	8,982	4,375	4,918	1,800	2,453	12,484	6,214	4,471	1,673	1,321	2,540	36,768	14,915	5,851
［時間比］†			［100：26：13］			［100：36：49］			［100：49：35］			［65：51：100］			［100：40：15］		
	物体探索プロセッサ (OS)	処理時間［割合］‡	994［3%］	1,975［22%］	3,641［83%］	562［11%］	1,129［63%］	2,095［85%］	807［6%］	1,552［25%］	2,605［58%］	493［29%］	1,237［94%］	2,459［97%］	1,301［4%］	2,542［17%］	4,627［79%］
		通信時間［割合］‡	406［1%］	449［5%］	549［13%］	207［4%］	225［13%］	284［12%］	501［4%］	554［9%］	619［14%］	76［5%］	83［6%］	81［3%］	470［1%］	471［3%］	470［8%］
		待ち時間［割合］‡	32,186［96%］	6,508［72%］	183［4%］	4,149［84%］	446［25%］	73［3%］	11,176［90%］	4,108［66%］	1,247［28%］	1,104［66%］	0.06［0%］	0.07［0%］	34,997［95%］	11,902［80%］	754［13%］
	交点計算プロセッサ (IC)	処理時間［割合］‡	33,148［97%］	8,429［94%］	2,346［54%］	4,651［95%］	1,440［80%］	544［22%］	11,967［96%］	5,604［90%］	3,725［83%］	1,429［85%］	513［39%］	329［13%］	36,217［99%］	14,264［96%］	4,995［85%］
		通信時間［割合］‡	346［1%］	388［4%］	489［11%］	193［4%］	212［12%］	271［11%］	460［4%］	514［8%］	578［13%］	67［4%］	74［6%］	72［3%］	419［1%］	421［3%］	420［7%］
		待ち時間［割合］‡	90［0%］	164［2%］	1,540［35%］	74［2%］	147［8%］	1,638［67%］	56［0%］	96［2%］	166［4%］	175［10%］	734［56%］	2,139［84%］	131［0%］	230［2%］	436［7%］
	輝度計算プロセッサ (SC)	処理時間［割合］‡	176［0%］	180［2%］	178［4%］	60［1%］	64［4%］	55［2%］	120［1%］	122［2%］	124［3%］	17［1%］	31［2%］	23［1%］	83［0%］	81［1%］	80［1%］
		通信時間［割合］‡	101［0%］	102［1%］	102［2%］	36［1%］	36［2%］	36［1%］	72［1%］	72［1%］	72［2%］	11［1%］	11［1%］	11［0%］	59［0%］	58［0%］	57［1%］
		待ち時間［割合］‡	33,309［99%］	8,650［96%］	4,095［94%］	4,822［98%］	1,700［94%］	2,362［96%］	12,292［98%］	6,020［97%］	4,275［96%］	1,645［98%］	1,279［97%］	2,506［99%］	36,626［100%］	14,776［99%］	5,714［98%］
	OS：IC：SC 負荷比§		5：120：1	8：31：1	14：10：1	7：49：1	13：16：1	25：8：1	6：64：1	10：31：1	16：21：1	19：51：1	31：13：1	73：11：1	12：256：1	21：104：1	36：39：1
モデル M の対モデル S 性能向上率			4%	28%	67%	16%	82%	34%	11%	35%	72%	25%	47%	17%	4%	10%	82%

†：分割数 20, 50, 100 の中で最も映像生成時間の大きいものを 100 とした場合の時間比
‡：各プロセッサの所要時間内に占める割合
§：負荷＝処理時間＋通信時間

表 3 光線量［単位：1,000 光線］
Table 3 The number of rays.

ベンチマーク・シーン			balls			mount			rings			tetra			tree		
次元当り分割数			20	50	100	20	50	100	20	50	100	20	50	100	20	50	100
物体探索プロセッサ (OS)	入力		6,797	7,515	9,182	3,494	3,810	4,792	8,392	9,274	10,345	1,384	1,494	1,460	7,912	7,929	7,905
		1次光線	262														
		IC から（戻り光線）	4,657	5,346	7,011	2,825	3,142	4,123	6,933	7,803	8,869	1,084	1,194	1,160	6,808	6,850	6,843
		SC から	1,878	1,906	1,907	406	406	406	1,196	1,208	1,213	38	38	38	842	816	799
	消滅		50	45	45	89	89	89	60	52	50	220	220	220	130	135	137
	出力		6,747	7,470	9,136	3,405	3,721	4,702	8,331	9,221	10,295	1,163	1,273	1,239	7,782	7,794	7,767
		IC へ	6,206	6,921	8,593	3,236	3,553	4,535	7,899	8,808	9,890	1,129	1,239	1,207	7,048	7,064	7,041
		SC へ	540	548	543	168	168	167	432	413	404	33	33	32	733	730	726
交点計算プロセッサ (IC)	入力 (OS からのみ)		6,206	6,921	8,593	3,236	3,553	4,535	7,899	8,808	9,890	1,129	1,239	1,207	7,048	7,064	7,041
	消滅		861	877	882	32	32	33	445	471	484	4.5	4.5	5.7	109	86	73
	出力		5,344	6,044	7,710	3,204	3,520	4,502	7,454	8,336	9,406	1,125	1,235	1,201	6,939	6,977	6,968
		OS へ（戻り光線）	4,657	5,346	7,011	2,825	3,142	4,123	6,933	7,803	8,869	1,084	1,194	1,160	6,808	6,850	6,843
		SC へ	686	696	697	377	377	377	520	532	536	41	41	41	131	127	124
輝度計算プロセッサ (SC)	入力＆消滅		1,228	1,246	1,242	547	547	545	953	946	940	74	74	73	864	857	851
		OS から	540	548	543	168	168	167	432	413	404	33	33	32	733	730	726
		IC から	686	696	697	377	377	377	520	532	536	41	41	41	131	127	124
	生成＆出力 (OS へのみ)		1,878	1,906	1,907	406	406	406	1,196	1,208	1,213	38	38	38	842	816	799
		分岐光線	476	481	481	206	206	206	319	323	325	0	0	0	0	0	0
		影光線	1,402	1,425	1,426	200	200	200	877	885	888	38	38	38	842	816	799
	生成抑止†		1,420	1,455	1,458	561	561	561	926	954	964	0	0	0	6	5	5

†：寄与率 ≤ しきい値のため，生成が取り止めとなった光線

表 4 各プロセッサの処理状況
Table 4 Statistics on each processor.

ベンチマーク・シーン		balls			mount			rings			tetra			tree		
次元当り分割数		20	50	100	20	50	100	20	50	100	20	50	100	20	50	100
物体探索プロセッサ (OS)	入力光線数 (×1,000)	6,797	7,515	9,182	3,494	3,810	4,792	8,392	9,274	10,345	1,384	1,494	1,460	7,912	7,929	7,905
	トラバースしたボクセル数 (×1,000,000)	20	45	88	11	26	52	15	35	64	3	9	20	28	64	124
	1光線当り平均トラバース・ボクセル数	3	6	9	3	6	10	1	3	6	2	6	13	3	8	15
	処理時間 (×1,000,000)	994	1,975	3,641	562	1,129	2,095	807	1,552	2,605	493	1,237	2,459	1,301	2,542	4,627
	1光線当り平均処理時間	146	263	397	161	296	437	96	167	252	356	828	1,684	164	321	585
	1ボクセル・トラバース当り平均処理時間	50	44	41	51	43	40	54	44	41	164	137	123	46	40	37
交点計算プロセッサ (IC)	入力光線数 (×1,000)	6,206	6,921	8,593	3,236	3,553	4,535	7,899	8,808	9,890	1,129	1,239	1,207	7,048	7,064	7,041
	交点計算回数 (×1,000,000)	1,376	346	92	99	31	12	146	61	37	28	10	6	490	191	66
	1光線当り平均交点計算回数	221	50	10	30	8	2	18	6	3	25	8	5	69	27	9
	処理時間 (×1,000,000)	33,148	8,429	2,346	4,651	1,440	544	11,967	5,604	3,725	1,429	513	329	36,217	14,264	4,995
	1光線当り平処理時間	5,341	1,218	273	1,437	405	120	1,515	636	377	1,266	414	273	5,139	2,019	709
	1交点計算当り平均処理時間	24	24	26	47	46	45	82	92	101	51	51	55	74	75	76
OS & IC	IC から OS への戻り光線数 (×1,000)	4,657	5,346	7,011	2,825	3,142	4,123	6,933	7,803	8,869	1,084	1,194	1,160	6,808	6,850	6,843
	1光線当り平均戻り回数	2.2	2.5	3.2	4.2	4.7	6.2	4.8	5.3	6.0	3.6	4.0	3.9	6.2	6.4	6.4
輝度計算プロセッサ (SC)	入力光線数 (×1,000)	1,228	1,246	1,242	547	547	545	953	946	940	74	74	73	864	857	851
	処理時間 (×1,000,000)	176	180	178	60	64	55	120	122	124	17	31	23	83	81	80
	1光線当り平均処理時間	143	144	143	110	117	101	126	129	132	230	419	315	96	95	94

共通している．

- mount では，分割数を 20→50 と増加させると性能が向上するが，さらに 50→100 と増加させると逆に低下する．tetra では，分割数を 100→50 と減少させると性能が向上するが，さらに 50→20 と減少させると逆に低下する．これから，mount および tetra の場合，分割数 20 と 100 との間に最適な空間分割数が存在すると言える．
- balls, rings, および, tree では，分割数を 20→50→100 と増加させるに従って性能が向上している．この場合，最適な空間分割数は 100 以上かもしれないし，あるいは，50 と 100 との間に存在するのかもしれない．

②評価モデル M の評価モデル S に対する性能向上率は最低 4%～最高 82% である．各ベンチマーク・シーンにおいて性能向上率が最高となるのは，映像生成時間が最小となる空間分割数のときである．

- balls, rings, および, tree の場合，分割数 100 のとき両評価モデルともに映像生成時間が最小になり，性能向上率もそれぞれ 67%，72%，および，82% と最高になる．
- mount および tetra の場合，分割数 50 のとき両評価モデルともに映像生成時間が最小となり，性能向上率もそれぞれ 82% および 47% と最高になる．

③評価モデル M において，空間分割数が 20→50→100 と増加するにつれて，物体探索プロセッサ (OS) の処理時間が増加する．一方，交点計算プロセッサ (IC) の処理時間は逆に減少する．輝度計算プロセッサ (SC) の処理時間は，ほぼ一定である．

④評価モデルMで映像生成時間が最小となるのは，物体探索，交点計算，輝度計算の各プロセッサの負荷 (OS : IC : SC 負荷比) が最も均衡している空間分割数においてである．つまり，空間分割数の増加に伴って増加する物体探索プロセッサの負荷と，逆に減少する交点計算プロセッサの負荷とが交差する点である．しかしながら，それでも，輝度計算プロセッサの負荷に比べて，物体探索プロセッサおよび交点計算プロセッサの負荷は 1 桁以上大きい．

⑤3 ステージのマクロパイプライン構成によりハードウェア量は 3 倍になったものの性能は最高で 1.8 倍程度にしかならないのは，まさしく上記の負荷不均衡に原因がある．輝度計算プロセッサの稼働率 (映像生成時間に占める処理時間の割合) は 4% 以内と極めて低く，これを独立した別ステージとする意味ははとんどない．つまり，2 ステージのマクロパイプライン構成でも十分効果があるので，輝度計算プロセッサは削除しても構わない．あるいは逆に，3 ステージ・マクロパイプライン構成のままで輝度計算プロセッサの稼働率

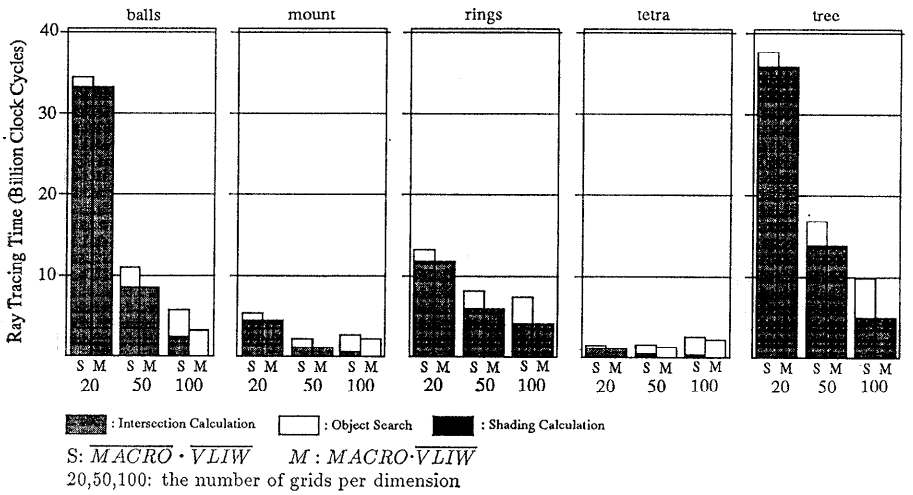


図 6　映像生成時間 [単位：1,000,000,000 クロック・サイクル]
Fig. 6　Ray tracing time.

を高めるには，物体探索プロセッサおよび交点計算プロセッサの高速化を図る，あるいは，輝度計算プロセッサへ負荷を分散させる必要がある．

⑥上述のとおり，ボクセル分割法では，空間分割数の最適化が非常に重要となる．その方法の1つとして，画素を粗くしたレイトレーシングを前処理として施し，その結果に基づいて空間分割数を決決することが可能である．たとえば，1000×1000の画素数でシーンを生成したい場合に，前処理で100×100 の画素数で空間分割数を3通り程度変えてレイトレーシングを行うとする．このとき，前処理に要する時間は，本処理に要する時間の(1/100)×3=3% 程度となる．

表3および表4に，評価モデルMの各プロセッサにおいて処理した光線量，物体探索プロセッサでトラバースしたボクセル数，および，交点計算プロセッサで行った交点計算回数を示す．これから，以下のことが判る．

- 空間分割数を 20→50→100 と増加させると，物体探索プロセッサおよび交点計算プロセッサへの入力光線がおおむね増加する傾向にある（表3参照）．これは，光線が交点計算プロセッサから物体探索プロセッサへ後戻りする回数が増加するためである（表4参照）．
- 同様に，物体探索プロセッサでトラバースするボクセル数も，空間分割数（すなわち，ボクセル数そのもの）の増加に従って当然増加する（表4参照）．これに伴って，1光線当りの平均トラバース・ボクセル数，および，1光線当りの平均処理時間も増加する．しかし，1ボクセル・トラバース当りの平均処理時間は，ほぼ一定である．
- 一方，交点計算プロセッサで行う交点計算回数は，空間分割数の増加に伴い逆に減少する（表4参照）．これにより，1光線当りの平均交点計算回数，および，1光線当りの平均処理時間も減少する．しかし，1交点計算当りの平均処理時間は，ほぼ一定である．この1交点計算当り平均処理時間は，映像を構成するプリミティブの種類および分布に，次のように依存する．
  - —balls における1交点計算当りの平均処理時間は 25 クロック・サイクル前後と最も小さい．これは，プリミティブのほとんどが球であり，交点計算が最も容易なことによる．
  - —次に，mount および tetra の1交点計算当り平均処理時間が 50 クロック・サイクル前後と2番目に小さい．これは，プリミティブのほとんどが，交点計算が比較的容易なポリゴンであることに起因する．
  - —最後に，rings および tree の1交点計算当り平均処理時間が 74〜100 クロック・サイクルと最も大きくなっている．これは，プリミティブの半数が，円錐や円柱といった非常に交点計算が複雑なもので占められているからである．
- 輝度計算プロセッサへの入力光線数は，空間分割数によらずほぼ一定である（表3参照）．また，1光線当り平均処理時間も同様に，空間分割数によらずほぼ一定となっている（表4参照）．
- 輝度計算プロセッサで生成される分岐光線および影光線の数はそれぞれ，反射面／透過面の有無，および，光源の数に依存する．
- 輝度計算プロセッサにおいて，寄与率法により生成が抑止された分岐光線および影光線がかなり存在する．特に，balls, mount, および，rings には反射面／透過面が存在しており，生成を抑止された光線は実際に生成された光線にほぼ匹敵した数になっている．これから，寄与率法が効果あることを確認できる．

6. おわりに

以上，『熱視線』の要素プロセッサ・アーキテクチャの特長の1つであるマクロパイプライン・アーキテクチャについて述べた．最初に，物体探索，交点計算，および，輝度計算の3ステージでレイトレーシングをマクロパイプライン処理する 3MPRT (3-stage Macro-Pipelined Ray-Tracing) アルゴリズムを提案した．そして，3MPRT アルゴリズムを直接実装したプロセッシング・エレメントである 3MPPE (3-stage Macro-Pipelined Processing Element) について，構成，実行過程，性能を述べた．

性能評価の結果，マクロパイプライン処理により最低 4%〜最高 82% の性能向上が可能であることが判明した．しかしながら，3ステージのマクロパイプライン構成としたことでハードウェア量は3倍になったものの，性能向上率は最高で 82% 程度にしか達していない．これは，輝度計算ステージの負荷が他の2ステージの負荷に比べて極端に小さいことに原因がある．これへの対処としてはまず，2ステージのマクロ

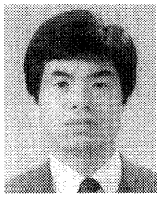
パイプライン構成で十分だとして，輝度計算プロセッサは削除することが考えられる．あるいは逆に，3ステージ・マクロパイプライン構成のままで輝度計算プロセッサの稼働率を高めるべく，物体探索プロセッサおよび交点計算プロセッサの高速化を図る，あるいは，輝度計算プロセッサへ負荷を分散させるという方法も考えられる．

謝辞 九州大学大学院総合理工学研究科情報システム学専攻の安浦寛人教授，ならびに，安浦研究室の諸氏に感謝します．

参考文献

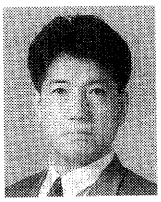
1) 秋本，間瀬：画素選択型光線追跡法，信学論，Vol. J 69-D, No. 12, pp. 1943-1952 (1986).
2) 石井：映像化マシン，オーム社 (1989).
3) 河合，山下，大野，吉村，西村，下條，宮原，大村：並列画像生成システム LINKS-2 のアーキテクチャ，情報処理学会論文誌，Vol. 29, No. 8, pp. 729-739 (1988).
4) 權，村上，富田：『熱視線』：視線探索法を高速処理する専用並列レンダリング・マシン―マクロパイプライン・アーキテクチャ―，情報処理学会研究会報告，ARC-85-6 (1990).
5) 鷲島，西澤，浅原：並列図形処理，コロナ社 (1991).
6) 出口，西田，西村，河田，白川，大村：視線探索法による画像生成のための木構造並列処理システム，信学論，Vol. J 69-D, No. 2, pp. 170-179 (1986).
7) 平井，日高，浅原，鷲島：画像生成用 SIMD 型マルチプロセッサシステム MC-2，情報処理学会研究会報告，MDP-29-5 (1986).
8) 堀江，池坂，石畑：並列計算機 CAP-II のルーティング・コントローラ，情報処理学会研究会報告，ARC-83-38 (1990).
9) 村田，權，村上，富田：『熱視線』：視線探索法を高速処理する専用並列レンダリング・マシン―マクロパイプラインの各ステージにおける命令レベル並列処理―，並列処理シンポジウム JSPP '91 論文集，pp. 109-116 (1991).
10) 吉田，成瀬，高橋，内藤：グラフィックス計算機 SIGHT の基本構成，情報処理学会研究会報告，CA-60-4 (1985).
11) Dippe, M. and Swensen, J.: An Adaptive Subdivision Algorithm and Parallel Architecture for Realistic Image Synthesis, *Proc. SIGGRAPH '84*, pp. 149-158 (1984).
12) Fujimoto, A., Tanaka, T. and Iwata, K.: ARTS: Accelerated Ray Tracing Systems, *IEEE Computer Graphics & Applications*, Vol. 6, No. 4, pp. 16-26 (1986).
13) Gaudet, S., Hobson, R., Chilka, P. and Calvert, T.: Multiprocessor Experiments for High-Speed Ray Tracing, *ACM Trans. Graphics*, Vol. 7, No. 3, pp. 151-179 (1988).
14) Glassner, A.S.: Space Subdivision for Fast Ray Tracing, *IEEE Computer Graphics & Applications*, Vol. 4, No. 10, pp. 15-22 (1984).
15) Green, S. A. and Padden, D. J.: Exploiting Coherence for Ray Tracing, *IEEE Computer Graphics & Applications*, Vol. 9, No. 6, pp. 12-26 (1989).
16) Gwun, O., Murata, S., Murakami, K. and Tomita, S.: A Parallel Rendering Machine for High-Speed Ray Tracing, *Proc. Int'l. Symp. Snpercomputing '91*, pp. 173-182 (1991).
17) Haines, E. A. and Greenberg, D. P.: The Light Buffer: A Shadow-Testing Accelerator, *IEEE Computer Graphics & Applications*, Vol. 6, No. 9, pp. 6-16 (1986).
18) Haines, E. A.: A Proposal for Standard Graphics Environments, *IEEE Computer Graphics & Applications*, Vol. 7, No. 11, pp. 3-5 (1987).
19) Hall, R. A. and Greenberg, D. P.: A Testbed for Realistic Image Synthesis, *IEEE Computer Graphics & Applications*, Vol. 3, No. 10, pp. 10-20 (1983).
20) Salmon, J. and Goldsmith, J.: A Hypercube Ray-tracer, *Proc. 3rd Conf. Hypercube Computers and Applications*, pp. 1194-1206 (1988).
21) Synder, J. M. and Barr, A. H.: Ray Tracing Complex Models Containing Surface Tessellations, *Proc. SIGGRAPH '87*, pp. 119-128 (1987).
22) Ullner, M. K.: Parallel Machines for Computer Graphics, Ph. D. Thesis, California Institute of Technology, Report Number 5112: TR: 83 (1983).
23) Weghorst, H., Hooper, G. and Greenberg, D. P.: Improved Computational Methods for Ray Tracing, *ACM Trans. Graphics*, Vol. 3, No. 1, pp. 52-69 (1984).
24) Whitted, T.: An Improved Illumination Model for Shaded Display, *Comm. ACM*, Vol. 23, No. 6, pp. 343-349 (1980).

(平成4年 6 月 26 日受付)
(平成4年 11 月 12 日採録)



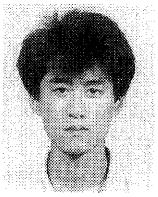
權　五鳳（正会員）

1954 年生．1980 年韓国高麗大学電気工学科卒業．1983 年同大学院制御工学専攻修士課程修了．同年韓国大丘工業専門大学専任講師．1992 年九州大学大学院総合理工学研究科博士課程退学．同年九州大学工学部情報工学科助手，現在に至る．計算機アーキテクチャ，並列処理システム，コンピュータ・グラフィックスの研究に従事．ACM, IEEE 各会員．



村上　和彰（正会員）

1960 年生．1982 年京都大学工学部情報工学科卒業．1984 年同大学院修士課程修了．同年富士通(株)本体事業部に入社，汎用計算機 M シリーズのアーキテクチャ開発に従事．1987 年九州大学工学部助手，1992 年同大学院総合理工学研究科講師，現在に至る．計算機アーキテクチャ，並列処理，性能評価などの研究に従事．著書「計算機システム工学（共著)」,「ヘネシー＆パターソン：コンピュータ・アーキテクチャ（共訳)」．本学会研究賞（平成 3 年度），本学会論文賞（平成 3 年度）受賞．電子情報通信学会，日本応用数理学会，ACM, IEEE, IEEE-CS 各会員．



村田　誠治（学生会員）

1968 年生．1991 年九州大学工学部情報工学科卒業．現在，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に在学中．並列処理，計算機アーキテクチャ，コンピュータ・グラフィックスの研究に従事．

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．