

1 次元 DSP アレイ上での光線追跡法の並列計算

小 畑 正 貴[†] 伊 藤 拓[†]

コンピュータグラフィックスの1手法である光線追跡法は高品質な画像を生成することができるが、そのために膨大な計算を必要とする。本論文では、1次元アレイプロセッサでの光線追跡法の並列計算方法とその評価結果について述べている。システムは256台の浮動小数点DSP（デジタルシグナルプロセッサ）をリングとバスの併用により接続した構成で、光線追跡法の実装においては巡回マッピングと呼ぶ負荷分散方法を提案している。この方法は2次元問題を1次元アレイに分配する場合の負荷の均等化に効果的であり、性能測定結果では256プロセッサで229倍の台数効果が得られた。また、プロセッサ間通信を必要とするアンチエイリアシング処理に対しては、本手法によって通信時間は増加するがそれ以上に負荷均等化の効果が大きいことを示した。

Parallel Computation of Ray Tracing on a One-dimensional DSP Array

MASAKI KOHATA[†] and TAKU ITO[†]

Ray tracing is a useful technique to generate photorealistic pictures, but it requires a huge amount of computation. This paper describes a parallel computation method of ray tracing on a one-dimensional processor array which employs an effective load distribution method, named the cyclic mapping method. An experimental system consists of 256 floating point DSPs. Several experimental results show that the cyclic mapping method is effective for balancing the computation load of ray tracing. The system with 256 DSPs is 229 times as fast as one DSP. Due to the 1-D processor array configuration, anti-aliasing such as filtering and super sampling requires additional interprocessor communication overhead, however, the cyclic mapping method is greatly effective to execute ray tracing algorithm.

1. はじめに

コンピュータグラフィックスの1手法である光線追跡法（レイトレーシング）¹⁾は光学系を忠実にシミュレートするもので、反射や屈折を含む極めて写実的な画像を生成することができる。しかしながら膨大な計算が必要で画像生成に時間がかかるという問題点があり、実用的なシステムを実現するためにはアルゴリズムの効率化とともに高速計算機の実現が必須である。

一方、光線追跡法のアルゴリズムには個々の画素の計算が独立して行えるという特徴があるため、並列処理に適している。そこで並列計算機により光線追跡法を高速化する試みが行われ、100台以上のプロセッサを持つシステムも試作されている。LINKS²⁾は1台のルートコンピュータと複数台のノードコンピュータか

らなるツリー型構造で、動的負荷分散により計算を進める。CAP³⁾やPixel Machine⁴⁾は2次元格子状の構成で静的負荷分散方式を採用している。単位プロセッサには、文献2), 3)では汎用マイクロプロセッサと浮動小数点ユニット、文献4)では汎用デジタルシグナルプロセッサ(DSP)が使用されているが、SIGHT⁵⁾のように専用プロセッサを開発している例もある。

我々は、256台の汎用DSPを1次元状に接続したシステム⁶⁾に光線追跡法を実装し、性能評価を行った。2次元構造で静的負荷分散をとるCAPやPixel machineなどのシステムに対し、本システムは1次元構造である。1次元構造はプロセッサ数を増してもハードウェア実装が容易な点で有利であるが、画像のような2次元問題に対して負荷分散の均等化に問題がある。本論文では1次元構造で静的負荷分散を行う場合の効果的な分散方法を提案し、実機上での実験結果を示す。

以下、2章でシステム構成、3章で光線追跡法の実装、4章で性能評価を示す。さらに、5章ではアンチ

[†] 岡山理科大学工学部情報工学科

Department of Information and Computer Engineering, Faculty of Engineering, Okayama University of Science

エリアシング処理に対する実験結果を示す。

2. 並列計算機 SPA

2.1 ハードウェア

並列計算機 SPA (signal processor array) のハードウェア構成を図 1 に、基本仕様を表 1 に示す。各プロセッサがローカルメモリを持つ MIMD 分散メモリ型マルチプロセッサで、単位プロセッサは 22 ビット精度で 20 Mflops の浮動小数点 DSP と 32 K バイトのメモリで構成される。プロセッサ台数は 256 台、結合は 1 次元型でバスとリングを併用した構成を採用している。

バスはホストコンピュータと DSP、および DSP と DSP の間のデータ転送を行う。ブロードキャスト機能により、ホストコンピュータのデータまたは指定した DSP のデータを全 DSP に同時に転送できる。また、リングはアレイ内のデータを右または左に一斉に回転し、隣接する DSP へのデータ転送を行う。データ転送はダイレクトメモリアクセス方式で、転送速度は 1 リンクあたり 6 M ワード/秒である。

2.2 ソフトウェア

通信はすべてホストが制御する。通信時、ホストは通信を行う DSP と同期を取り、その後データ転送を行う。バスによるブロードキャストとリングによる回転においては、ホストは通信時に全 DSP と一斉同期をとる。ホスト側プログラムと DSP 側プログラムの間では通信関数の対応がとられている必要がある。同期は通信時にとられるので通常は陽に記述しない。

応用プログラムは DSP とホストの協調作業により実行される。基本的にホスト側でデータ転送と入出力を行い、DSP 側で計算を行う。DSP に対しては全部に同一のプログラムを与える。

3. 光線追跡法の実装

3.1 光線追跡法

光線追跡法では、図 2 のように画面は球や直方体などの基本物体 (プリミティブ) の組み合わせで構成される。視点とスクリーン上の各画素を結ぶ線 (光線) を延長し、物体との交点における輝度を計算してその画素の値とする。物体表面で反射や屈折が起こる場合は交点から新たな光線を発生させ、追跡を続ける。

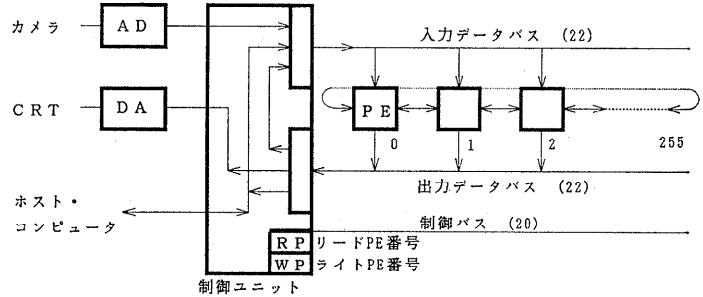


図 1 SPA ハードウェア構成
Fig. 1 Hardware configuration of SPA.

表 1 SPA システム諸元
Table 1 System specifications of SPA.

単位プロセッサ	
データ形式	22 ビット浮動小数点
サイクルタイム	100 ns
ピーク性能	20 MFlops (積和計算)
メモリ	32K バイト
全 体	
プロセッサ数	256
ピーク性能	5.12 GFlops
メモリ	8M バイト
接続	リング+バス
通信速度	6M ワード/秒/リンク

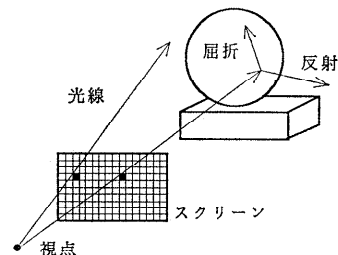


図 2 光線追跡法
Fig. 2 Ray tracing.

全体では画素数と同じだけの初期光線について上の計算を行う。図 3 に光線追跡法の基本アルゴリズムを示す。このアルゴリズムには光線単位の並列性と物体単位の並列性が含まれているが、光線単位の計算は光線ごとに独立しているため特に並列処理に向いている。一方、物体単位の並列計算ではプロセッサ間通信が必要になる。

3.2 1次元アレイでの負荷分散

SPA による並列実行の手順を以下に示す。まずブロードキャスト転送により、全 DSP のそれぞれに全物体データを持たせる。これにより計算途中での通信

は不要となる。次に各プロセッサは、あらかじめ分割されたスクリーンの担当画素に対して光線追跡法の計算を行う。最後にホストが計算結果を回収し、表示する。

ここで問題となるのはスクリーン上の画素の分割方法である。光線が物体と交差しない場合の計算量は少ないが、多くの物体と交差する場合は計算量は大きくなる。よって各プロセッサに同じ面積（画素数）を割り当てても計算量は同じにならず、スクリーン上での物体の片寄りによって計算負荷の不均衡を生じる。

動的負荷分散は最初から割り当てを固定せず、各プロセッサが計算を終えるたびに次の計算画素を与えていく。プロセッサ間での計算量の不均衡は少なくなるが、制御は複雑になり、計算中に負荷分散のための通信が必要となる。一方、静的負荷分散は最初に固定的にプロセッサに画素を割り当てる方法で、負荷バランスは悪くなるが制御は簡単で通信も不要になる。

SPA への実装では静的負荷分散を用いた。SPA は 1 次元構造であるので、単純には画面を行方向または列方向に分割してプロセッサに割り当てることになる。図 4 ではプロセッサ 0 と i が計算する画素を示している。図 4 (a) では単純に 1 列を 1 プロセッサに割り当てているが、近接する画素の計算量は同程度であるため物体の位置によってプロセッサごとの負荷の不均衡が大きくなる。

3.3 巡回マッピング

本論文では負荷分散の均等化のため、図 4 (b) に示す巡回マッピングと呼ぶ分配方法を提案する。 $N \times N$ 画素の画像に対して、単純な列分割ではプロセッサ i に画素 (j, i) ($j=0 \sim N-1$) を割り当てるが、巡回マッピングでは画素 $(j, (jp+i) \bmod N)$ ($j=0 \sim N-1$) を割り当てる。ここで p をオフセットと呼ぶ。図 4 のよ

```

procedure 光線追跡法
begin
  すべての画素に対して,
    視点からの光線を生成し, 光線処理を行う。
end;

procedure 光線処理
begin
  すべての物体に対して,
    交差判定を行い, 最初に交差する物体を求める。
    その交点に対して輝度を計算し, 現画素値に加算する。
    反射や屈折がある場合は新しい光線を生成し,
    光線処理を再帰的に行う。
end;
```

図 3 光線追跡法のアルゴリズム
Fig. 3 Ray tracing algorithm.

うに上の行に対して p だけずれるため、同一プロセッサに対する負荷の集中が軽減できる。 $p=0$ とすると単純な列分割（図 4 (a)）と同じになる。

CAP や Pixel Machine のような 2 次元マシンでは、プロセッサ数を $n \times n$ とすると、図 5 (a) のように行方向と列方向に n ずつとばした画素を 1 プロセッサに割り当てることにより良好な負荷分散が得られている。同じ n^2 台のプロセッサを持つ 1 次元アレイを考えた場合、図 5 (b) のように巡回マッピングのオフセットを $p=n$ とすると 2 次元マシンでのプロセッサ割り付けに類似した割り付けが行われる。図 5 (b) の $n \times n$ 画素の各小領域はそれぞれ全プロセッサによって

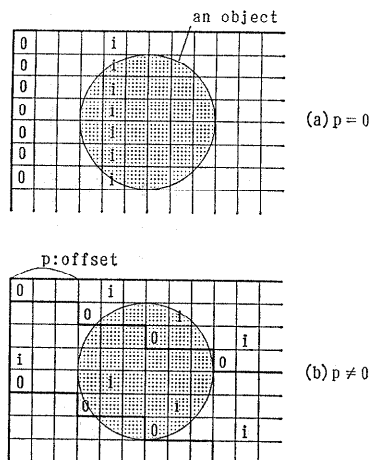


図 4 巡回マッピング
Fig. 4 Cyclic mapping.

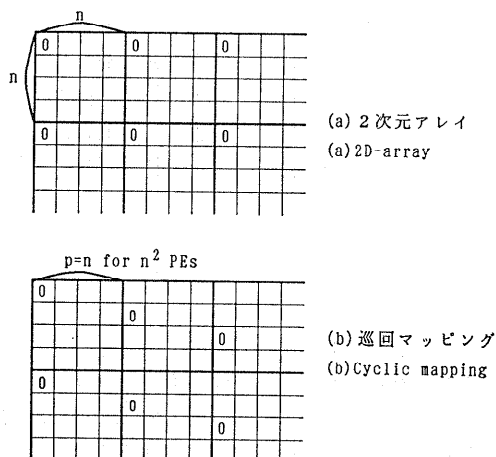


図 5 2 次元アレイとの比較
Fig. 5 Comparison with 2D-array.

1画素ずつ計算される。この図より均等化は $p=n$ のときが最良で、 p を n 以上にしても偏りの方向が変わるだけで不均衡は大きくなる。 $p=n^2$ は $p=0$ と同じ割り付けになる。また、近接通信では p は小さい方が有利なので、 p は台数の平方根までで設定すればよい。

4. 性能評価

実験プログラムの描画機能は、プリミティブは球と平面、モデラは Surface model である。多重反射、屈折については8回まで計算する。SPA は経済的制約から単位プロセッサの性能とメモリ容量が十分ではなく、基本的な描画機能しか実装できない。ここでは光線追跡法における負荷バランスを調べるのに必要最低限の機能のみを実装した。

実験では数種類のサンプル画像（物体数 6~30）に対して、 256×256 画素と 512×512 画素の画像生成を行った。巡回マッピングによる均等化の効果はすべてに対して得られたが、以下では負荷のばらつきが最も大きかった例（図 6）について詳しく解析する。これは反射、屈折を含む球が中央付近に集中した画像で、画素数は 256×256 である。

4.1 負荷分散

3章で提案した巡回マッピングにおいて、オフセット p を変化させたときの負荷バランスの様子を図 7 に示す。単純な列分割（図 7(a)）では物体の集中している中央部分で負荷が大きくなっているが、巡回マッピング（図 7(b), $p=16$ ）では負荷バランスが改善され、全体の計算時間が短縮されていることがわかる。

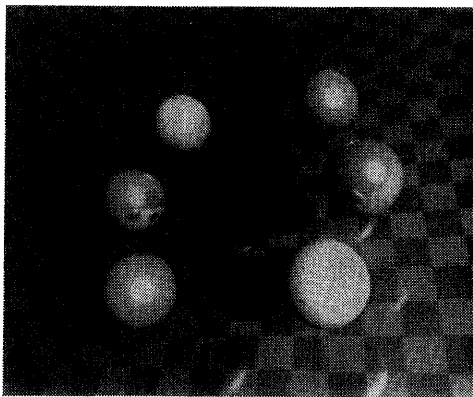


図 6 サンプル画像
Fig. 6 A sample picture.

次に、負荷のばらつきの度合い u を次式で定義する。

$$u = \frac{\text{最大負荷} - \text{平均負荷}}{\text{平均負荷}}$$

p を変化させたときの u の変化を図 8 に示す。プロセッサ数が $256 (=16 \times 16)$ なので、3.3 節の議論より p は 16 まで変化させればよい。負荷バランスは $p=16$ で最も良くなるが、 $p=8$ でも良好な負荷分散が得られることがわかる。次章で述べるアンチエイリアシング処理における通信に対しては、 p は小さいほどよい。

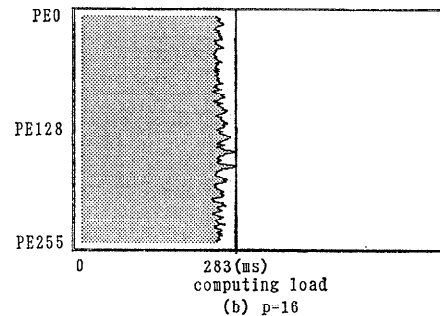
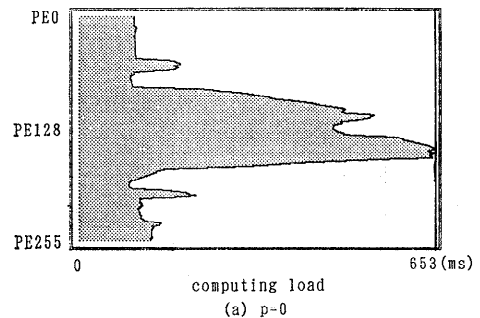


図 7 負荷分散
Fig. 7 Load distribution.

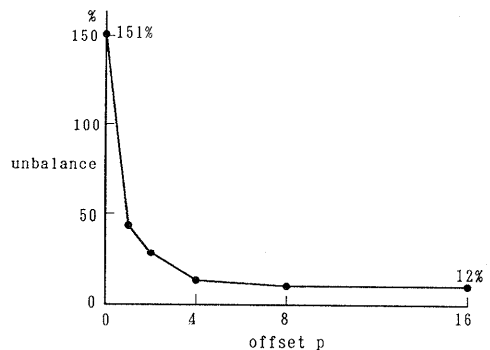


図 8 負荷のばらつき
Fig. 8 Load unbalance.

4.2 台数効果

プロセッサ台数に対する計算速度の向上を図9に示す。図では $p=0$ および $p=16$ の場合を示している。巡回マッピングにより負荷分散が改善され、プロセッサ数が大きくなっても台数に比例した速度向上が得られていることがわかる。2次元型マシンでの台数効果の例として、CAP において256台での64台の3.86倍³⁾、Pixel machine において64台で1台の60倍以上⁴⁾という報告があるが、図9はこれらに近いものとなっている。

また画像の違いによる台数効果の違いを調べた結果、単純な列分割では画像によって効率が大きく違ってくるのに対して、巡回マッピングでは画像によらず高い効率を得られることを確認した。

4.3 全体性能

同じ計算をパーソナルコンピュータおよびワークステーションで行ったときの総合的な処理時間を表2に示す。計算精度の違いなどがあるので、現在のSPAのデータがそのままマシン性能の比較にはならないが、並列マシンによる高速計算の可能性を示す参考データとして示す。

SPA の場合、表示をパーソナルコンピュータ上のフレームメモリで行っているため、計算とデータ転送以上に表示に時間がかかっている。表示の高速化のためにはSPA のバスに直結できるフレームメモリを持たせる必要がある。

5. アンチエイリアシング

光線追跡法では画面の格子点における計算値をその画素全体の値として用いる。画素は方形なので物体の輪郭部分などではジャギー（ギザギザ）が生じる。こういう現象はエイリアシングと呼ばれ、これを低減するためにアンチエイリアシングの手法が採られる。以下ではフィルタリングとスーパーサンプリングについての実験結果を示す。

5.1 フィルタリング

生成画像の各画素に対して、自分を含む 3×3 の隣接画素の平均をとることによりジャギーを低減する。画像が全体にぼけるという欠点があるが、実装が簡単なのでしばしば用いられる。

SPA でこのフィルタリングを行う場合、隣接画素を参照するためのプロセッサ間通信が必要となる。

表2 汎用機での計算時間

Table 2 Computing time on general purpose machines.

条 件				
機 種	プロセッサ	サイクルタイム	計算精度	記述言語
PC	CISC+FPU	50 ns	64 bits	C
EWS	RISC+FPU	25 ns	64 bits	C
SPA (256 PE)	DSP	100 ns	22 bits	アセンブラ

処理時間 (ms)					
	物体データ転送	計 算	生成画像転送	表 示	計
PC	0	295,910	0	1,130	297,040
EWS	0	28,800	0	300	29,100
SPA (256 PE)	2	283	140	1,130	1,555
SPA (1 PE)	2	64,800	140	1,130	66,072

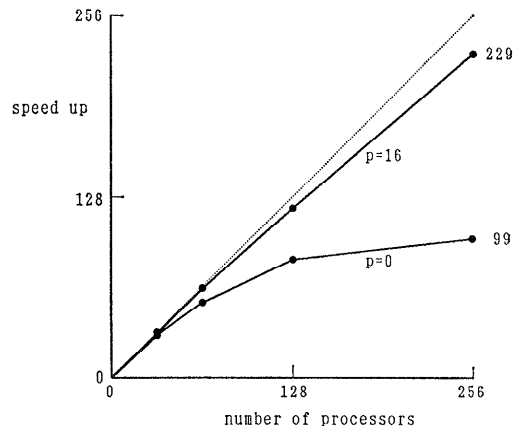


図9 台数効果

Fig. 9 Speed up.

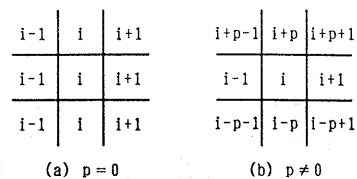


図10 隣接画素の位置

Fig. 10 Positions of neighbor pixels.

隣接画素をどのプロセッサが持っているかを表したのが図10である。単純な列分割では図10(a)のように左右の画素は左右のプロセッサが持ち、上下の画素は自分自身が持つ。したがってリングによる左右1回ずつの回転で必要なデータが得られる。一方巡回マッピングでは図10(b)のように、左右の画素は左右のプロセッサが持ち、上下の画素はそれぞれオフセット p だけ離れたプロセッサが持つ。 p だけ離れたプロセッサ

サのデータを得るには p 回の回転が必要なので、全体では左右 $(p+1)$ 回ずつの回転を要する。

3×3 フィルタの計算は、図11のように2ステップで行う。すなわち、左右の転送で3画素について計算し、その中間結果を上下に転送する。図では左右の転送が3回必要のように見えるが、上下ライン上の転送は上下画素の計算でも使われるので1画素に対して実質1回でよい。したがって、全画素に対して上下左右の1回ずつの転送で 3×3 フィルタが計算できる。

フィルタリングを含めた計算時間を表3に示す。巡回マッピングは、フィルタリングについては転送回数が多くなるので不利であるが、負荷均等化の効果が大きいので全体では2倍以上速くなっている。

5.2 スーパーサンプリング

この手法では、通常の計算の後、画像の変化の大きい場所においてその格子点のまわりの中間位置に新たな光線を追加する。さらにこれら複数の光線による計算値の平均をとってその画素の値とする。計算は複雑になるが、質の高い画像を得ることができる。

変化が大きいかどうかは隣接画素の比較によって判定するが、これはフィルタリングの時と同じプロセスな間通信によって行うことができる。計算時間も同等のものとなる。新たな光線に対する計算は基本的な光線追跡法の計算と同じである。画像の変化が大きいのは物体の輪郭部なので計算負荷には片寄りが生じるが、この場合も巡回マッピングによる負荷の均等化が有効となる。

実験で用いたスーパーサンプリングのアルゴリズムを以下に示す。

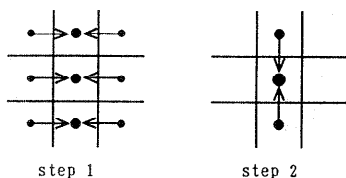


図11 3×3 フィルタにおけるデータ転送
Fig. 11 Data transfer for the 3×3 filtering.

表3 フィルタリングを含む計算時間
Table 3 Total computing time with filtering.
(単位 ms)

	$p=0$ (列分割)	$p=16$
Ray tracing	653	283
Filtering	9	19
Total	662	302

●各画素に対して、上下左右の画素と各色ごとに値(0~1)を比較し、0.125以上の差が1色でもあれば再計算する。

●再計算では4方向に1/2画素離れた4点を新たに計算し自分を合わせた5点の平均をとる。

負荷分散の測定結果(図12, 図13)より、スーパーサンプリングにおいても巡回マッピングによる負荷均等化の効果が得られていることがわかる。通信を含む計算時間(表4)では、フィルタリングの場合と同様

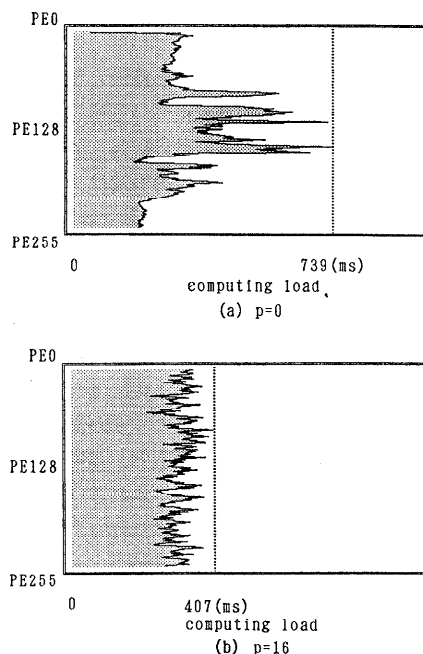


図12 スーパーサンプリングでの負荷分散
Fig. 12 Load distribution in the super sampling computation.

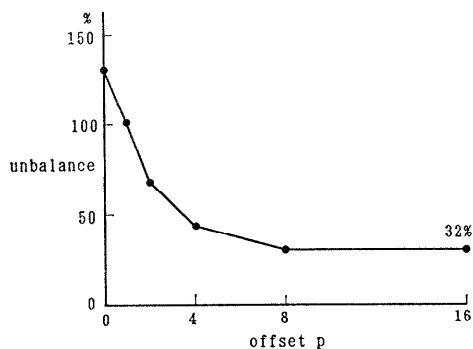


図13 スーパーサンプリングでの負荷のばらつき
Fig. 13 Load unbalance in the super sampling computation.

表 4 スーパーサンプリングの計算時間
Table 4 Computing time of super sampling.

	(単位 ms)	
	$p=0$ (列分割)	$p=16$
Communication	2	12
Computation	739	407
Total	741	419

に、通信時間の増加以上に負荷均等化の効果が大きくなっている。

6. おわりに

本論文では汎用 DSP を 1 次元接続した並列計算機での、光線追跡法の並列計算について述べた。特に、1 次元アレイへの画素の割り付けに対して巡回マッピングと呼ぶ方法を提案した。この方法は、フィルタリングなど隣接データを使う処理での通信時間は大きくなるが、負荷分散の均等化の効果が大きいので全体としては有効であった。この方法は、光線追跡法だけでなく、2 次元の問題を 1 次元システムに実装する場合の負荷分散方法として効果があると考えられる。

一般に 1 次元構造でも画像の 1 辺の画素数に対してプロセッサ数が小さく、1 プロセッサが複数行（または列）を飛び飛びに計算する場合は、単純な割り当てでもある程度負荷は分散される。本実験では、1 辺の画素数と同じ数のプロセッサにより行った上で高い効率が得られた。このことは、より精細な画像に対してはそれに合わせてプロセッサ数を増加して対応できることを示している。ただし、1 辺の画素数までであり、それ以上は 2 次元化などが必要になる。

現在の実験システムをより実用的なシステムとするには表示の高速化や描画機能の充実が必要である。また写実性向上のためには物体データ量が増大するので、物体データを分割して格納、処理できなければならない。これらの点を考慮した並列実装が今後の課題である。

参考文献

- 1) 千葉則茂, 村岡一信: レイトレーシング CG 入門, サイエンス社 (1990).
- 2) 西村ほか: LINKS-1: コンピュータグラフィックスシステム, 情報処理学会マイクロコンピュータ研究会, 24-1 (1982).
- 3) 石井光雄: 映像化マシン, オーム社 (1989).
- 4) Potmesil, M. and Hoffert, E.M.: The Pixel Machine: A Parallel Image Computer, *Computer Graphics*, Vol. 23, No. 3, pp. 69-78 (1989).
- 5) 吉田ほか: グラフィックス計算機 SIGHT の基本構成, 情報処理学会計算機アーキテクチャ研究会, 60-4 (1985).
- 6) 小畑正貴: 浮動小数点 DSP による高並列アレイプロセッサシステム, 情報処理学会論文誌, Vol. 32, No. 9, pp. 1142-1148 (1991).

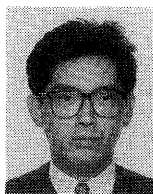
(平成 4 年 2 月 20 日受付)

(平成 5 年 2 月 12 日採録)



小畑 正貴 (正会員)

1957 年生。1980 年神戸大学工学部電子工学科卒業。1985 年同大学院自然科学研究科博士課程修了。学術博士。1984 年岡山理科大学理学部助手, 1990 年同大学工学部助教授, 現在に至る。計算機アーキテクチャ, 並列処理の研究に従事。電子情報通信学会会員。



伊藤 拓 (正会員)

1964 年生。1990 年岡山理科大学理学部電子理学科卒業。1992 年同大学大学院工学研究科修士課程修了。同年, 同大学大学院工学研究科博士課程入学, 現在に至る。並列計算機アーキテクチャ, 並列処理言語に関心をもっている。