

マルチトランスピュータシステム上での ラジオシティ法の並列化

上 嶋 明† 山 崎 勝 弘†
渡 部 透† 得 丸 英 勝†

ラジオシティ法では光の相互反射を考慮したリアルな画像を生成できるが、フォームファクタを求めるために非常に大きな計算量を必要とする。本論文では、フォームファクタ計算に用いるヘミキューブを細かい領域に分け、それを各プロセッサに割り当てて並列化する方法を述べる。プロセッサ間の負荷均衡を実現するために、インタリーブ方式による静的負荷分散法と、グループ分割による動的負荷分散法を考案した。5種類のベンチマークシーンの処理を対象として、トーラス結合された並列計算機トランスピュータ64台上で実行した結果、プロセッサ数の増加とともに処理時間がほぼ直線状に減少すること、および動的負荷分散法が静的負荷分散法よりもやや良いことが分かった。さらに、並列化の効率はパッチ数に依存し、パッチ数が少ないほど良くなることが分かった。速度向上の点からは、プロセッサ64台の場合、1台に対して平均で30倍の速度が得られた。

Parallelization of the Radiosity Method on a Multi-Transputer System

AKIRA UEJIMA,[†] KATSUHIRO YAMAZAKI,[†] TOHRU WATANABE[†]
and HIDEKATSU TOKUMARU[†]

Although the radiosity algorithm can generate very realistic images due to the global illumination effects, it requires an enormous amount of computation for calculating form factors. This paper describes how to parallelize the calculations by subdividing hemicubes into multiple elements and allocating them to multiple processors. A static load balancing algorithm and a dynamic one are devised to equalize the processing time in each processor. Five benchmark scenes are executed on a 64 transputers system connected by a torus network. The results demonstrate that the execution time reduces almost linearly as the number of processors increases and the dynamic load balancing algorithm is slightly better than the static one. In addition, the efficiency of parallel execution depends on the number of patches and the fewer the number of patches, the better that efficiency is. The processing speed is thirty times faster than the speed using one processor on average.

1. はじめに

ラジオシティ (radiosity) 法は、物体表面を構成するポリゴン細かいパッチ (patch) に分割し、各々について光の出入りを計算して輝度を求める方法である。ラジオシティ法の特徴は、物体間の光の相互反射を取り扱えるため、非常に現実感の高い画像を生成できることである^{1)~3)}。

初期の頃は光の平衡状態から得られる連立方程式を直接解いていたが、漸進 (progressive refinement) 法⁴⁾が発表されてからは、これが一般的に用いられる

ようになった。漸進法では徐々に解に近づけていくため、計算の途中でもそれなりの結果が得られる点、およびメモリを大幅に節約できる点が非常に有利である。

ラジオシティ法ではフォームファクタ (form factor) と呼ばれる、パッチ間で光が到達する割合の計算が重要となるが、間に障害物がある場合などにこれを積分計算で求めるのは非常に困難である。そのため、近似値を求める方法としてヘミキューブ (hemicube) 法⁵⁾が提案された。一般に、ラジオシティ法ではフォームファクタの計算時間が全計算時間の大部分を占める。

本研究では、ラジオシティ法の処理を、トーラス結合されたマルチトランスピュータシステム (T805×64台, 25 MHz) 上で並列化する方法と、その効果について述べる。従来の並列化の手法では、ポリゴンや

† 立命館大学理工学部
Faculty of Science and Engineering, Ritsumeikan
University

パッチ単位で分割を行うのが一般的である⁶⁾。本研究では、フォームファクタ計算の際にヘミキューブを分割して各プロセッサに処理を割り当て、ポリゴンのヘミキューブへの描画を並列処理する。同様の方法として、この描画処理を専用ハードウェアで一括処理した後、ルックアップテーブルを用いたフォームファクタの計算のみを並列化した例⁷⁾があるが、本研究では描画の処理段階も並列化している。この方法では、ヘミキューブに必要なメモリ領域を各プロセッサに分散できるため、ローカルメモリを有効に利用できる。

マルチトランスピュータシステム上での並列化の例として、ラジオシティの計算とレンダリングを同時に、それぞれ8台のプロセッサで並列処理した研究がある⁸⁾。この研究では、シーンのデータを完全に各プロセッサに分散させているため、ヘミキューブへの描画の際、必要なデータをプロセッサ間で転送する必要がある。それに対し、本研究でのシーンのデータの分散方法は、ポリゴンデータのコピーを全プロセッサに配置し、パッチデータを各プロセッサに分散させる。ここで、ポリゴンは物体を構成する多角形で、座標データを含んでいる。ポリゴンの形状が長方形の場合、それを縦方向と横方向それぞれ等間隔に分割して、そのひとつひとつをラジオシティの計算単位であるパッチとしている。この分散方法は、フォームファクタ計算の際に座標データを各プロセッサ間で転送する必要がないため、所要メモリ量の点では不利であるが、速度の点では非常に有利となる。

各プロセッサ間の負荷均衡を実現するために、全処理を単純に等分割し、インタリーブ方式による割当てを行う静的負荷分散 (static load balancing) と、実行時に割当てを行う動的負荷分散 (dynamic load balancing) を用いる。後者では、全プロセッサを複数のグループに分割し、グループ間の静的負荷分散、かつグループ内の動的負荷分散を行う。

本論文では、5種類のベンチマークシーンの計算を対象として処理時間の測定を行い、ヘミキューブの解像度による静的負荷分散法と動的負荷分散法の比較、およびパッチ数とポリゴン数各々を変化させた場合の比較を示し、考察を加える。

2. マルチトランスピュータシステム

本研究で用いるトランスピュータ T805 は 32 ビットのマイクロプロセッサで、動作クロック周波数は 25 MHz である。プロセッサ間の通信は速度 10 Mbps の 4 本のシリアルリンクを用いて行う。結合網として、 2×2 , 4×4 , 4×8 , 8×8 の 4 種類のトラス接

続、および 2^5 のハイパーキューブ接続を使用できる。このうち、ハイパーキューブ接続では、プロセッサ 2 台を組み合わせるとリンクを 6 本としている。本研究では、4 本のリンクを有効に利用でき、通信オーバーヘッドが小さいトラス接続を用いる。ローカルメモリは、1 台での逐次処理の場合にも使用することになるプロセッサに 4 M バイト、他の 63 台のプロセッサに 1 M バイトを備えている。なお、計算を担当するプロセッサ群とは別に、外部と入出力を行うためのルート・プロセッサを 1 台接続している。

本研究では言語として Parallel C を用いる。Parallel C は一般の C 言語に対して、プロセッサ間の通信やマルチスレッド処理を行うための関数などが拡張されている。スレッドとは、トランスピュータで実行される最小のプロセスで、複数のスレッドがデータを共有し、並行に実行される。マルチスレッド処理を行うことで、計算と通信を並列に行うことができる。

3. ラジオシティ法

3.1 計算手順

ラジオシティ法による画像生成の手順は、次のようになる^{1),3)}。

- (1) ポリゴンで物体を定義する。ただし、その表面は理想的な拡散反射面とする。
- (2) 各ポリゴンを適当な大きさに分割する。分割された 1 つをパッチと呼び、ラジオシティの計算単位とする。
- (3) 光のエネルギーが平衡状態に達しているとして、各パッチの光の入射と反射に注目すると、次の式が成立する。

$$B_i A_i = E_i A_i + \rho_i \sum_{j=1}^n B_j F_{ji} A_j \quad (1)$$

ここで、 B_i , B_j はパッチ i , j 各々のラジオシティ・エネルギー、 E_i はパッチ i の自己放射エネルギー、 A_i , A_j はパッチ i , j 各々の面積、 ρ_i はパッチ i の反射率、 F_{ji} はパッチ j からパッチ i へのフォームファクタ、 n はパッチ数である。フォームファクタとは、あるパッチから他のパッチへ光のエネルギーが到達する率で、対称性 $A_i F_{ij} = A_j F_{ji}$ により、式 (1) は次のように表せる。

$$B_i = E_i + \rho_i \sum_{j=1}^n B_j F_{ij} \quad (2)$$

- (4) 式 (2) より得られる次の連立方程式を解き、全パッチのラジオシティ B_i を求める。

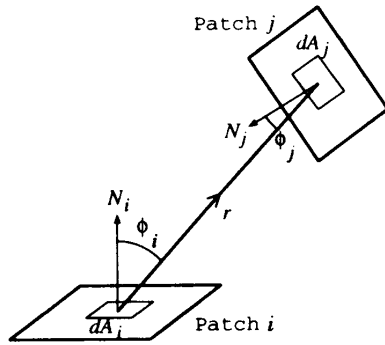


図1 フォームファクタの位置関係
Fig. 1 Form factor geometry.

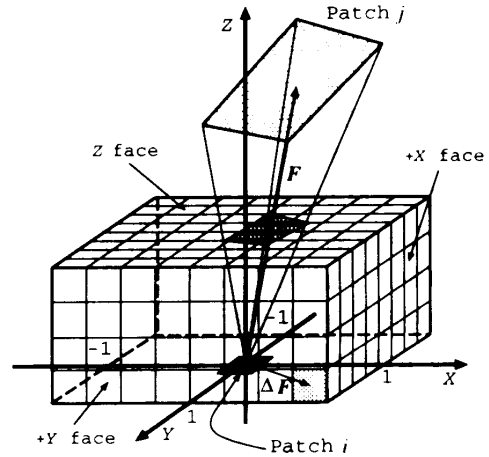


図2 ヘミキューブ
Fig. 2 Hemicube.

$$\begin{bmatrix} 1 - \rho_1 F_{11} & -\rho_1 F_{12} & \dots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1 - \rho_2 F_{22} & \dots & -\rho_2 F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \dots & 1 - \rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} \quad (3)$$

(5) 求めた各パッチの中心のラジオシティを頂点のラジオシティに変換し、パッチ内部を2次元の線形補間によりレンダリングする。

3.2 フォームファクタ

2つのパッチが図1に示すような位置関係にあるとき、フォームファクタ F_{ij} は、次のように表される。

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos \phi_i \cos \phi_j}{\pi r^2} dA_j dA_i \quad (4)$$

2つのパッチの間に障害物が存在するような複雑な位置関係のときに、この式によりフォームファクタを正確に計算するのは非常に困難である。そのため、近似値を求める方法として、次に述べるヘミキューブ法がある。

3.3 ヘミキューブ法によるフォームファクタの計算

図2に示すように、パッチiを中心にX軸・Y軸方向に対しては-1から1、Z軸方向に対しては、0から1の大きさのヘミキューブを考え、表面を適当な解像度のメッシュに分割する。たとえば、図2の場合の解像度は8である。そして、パッチiから各メッシュへのフォームファクタ ΔF を次の式を用いて計算する。

Z面については、

$$\Delta F = \frac{1}{\pi(x^2 + y^2 + 1)^2} \Delta A_i \quad (5)$$

±Y面については、

$$\Delta F = \frac{z}{\pi(x^2 + z^2 + 1)^2} \Delta A_i \quad (6)$$

この計算結果を後の計算のため、ルックアップテーブルに入れておく。

次に、Zバッファ法を用いて、パッチをヘミキューブに投影し、隠面除去によって距離が最も短いもののみを残す。ルックアップテーブルを用いて、パッチjが投影されたメッシュの ΔF の和を計算すると、 F_{ij} が得られる⁵⁾。なお、ヘミキューブの解像度を高くするほど計算時間が長くなるが、得られるフォームファクタの精度は良くなる。

本研究ではパッチではなく、ポリゴンをヘミキューブに投影して隠面除去を行った後、各メッシュがポリゴン内の、どのパッチに対応するかを決定している。これは、ポリゴンの形状が長方形の場合、縦方向と横方向それぞれ等間隔に分割してパッチとしているため、ポリゴン内の任意の点の座標が分かれば、その点を含むパッチを決定できることを利用している。なお、ポリゴンの形状が長方形以外の場合はパッチへの分割は行わず、1ポリゴン = 1パッチとしている。このように、本研究での描画は、ヘミキューブへのポリゴンの投影、Zバッファ法による隠面除去、パッチの決定という一連の作業を行うことになる。これによりパッチを直接描画したのと同じ結果が得られる。この方法では、ヘミキューブへの投影数はポリゴン数に等しく、パッチ数とは無関係となる。

3.4 漸進法

式(2)は各パッチの光の入射と自己放射に注目していた。一方、漸進法では各パッチの光の放射に注目し、あるパッチから他のパッチへの光の放射を繰り返すことで、全パッチのラジオシティを平衡状態に徐々に近

セッサが並列に、自分の保持するパッチへの放射の処理を行う。以上のように、最大ラジオシティを持つパッチの検索、フォームファクタの計算、そして漸進法の放射の過程を並列化している⁹⁾。繰返しの1回の中には、図4に示すように、全プロセッサが2カ所で同期をとることになる。

4.1 静的負荷分散

ヘミキューブを分割したものを、各プロセッサに均等に割り当てる。この際、物体の配置に偏りがある場合でも、なるべく各プロセッサに均等に負荷が分散できるようにインタリーブ方式で行う。まず、図5に示すように、分割したヘミキューブの領域にライン番号を振っておく。プロセッサを0番から $(Tn-1)$ 番までの Tn 台とすると、ヘミキューブのライン l は、 $(l \bmod Tn)$ 番のプロセッサが計算を担当する。たとえば、解像度8のヘミキューブをプロセッサ4台で並列化する場合、プロセッサ0はライン0, 4, 8, ..., 36の10カ所(図5の網掛部分)を担当することになる。

ヘミキューブの1つのメッシュに必要なメモリ量は8バイトである。したがって、解像度が192の場合、全体で864kバイトのメモリが必要である。しかし、プロセッサ64台で並列処理の場合、必要なメモリ量はプロセッサ1台あたり $864/64 = 13.5$ kバイトでよい。

4.2 動的負荷分散

各プロセッサでは、制御部(Allocator)、計算部(Calculator)の2つのスレッドを生成する。そして、各プロセッサの制御部の間でトークンを流す。トークンはライン番号と終了フラグを含んでいる。このライン番号により各プロセッサは計算するラインを指示される。終了フラグは、ヘミキューブのすべての部分の割当てが終了したことを意味し、各プロセッサに部分フォームファクタの通信の開始を指示する。図5でZ面が0~7, $\pm X$ 面と $\pm Y$ 面が8~39となるようにライン番号を振っているのは、処理量の大きいZ面の割当てを最初に行い、その後 $\pm X$ 面と $\pm Y$ 面を割り当てることで、負荷均衡を図るためである。

動的負荷分散法においてプロセッサ数が多くなると、1つの処理が終わった後、次の処理が到着するまでに間隔が空くようになる。その対策として、グループ分割による動的負荷分散を用いる。すなわち、図6に示すように、トーラスの横1列のプロセッサを1つのグループとして、そのグループ内で動的負荷分散を行う。各グループに対しては静的に処理を割り当てるので、グループ間静的負荷分散、かつグループ内動的負荷分散となる。

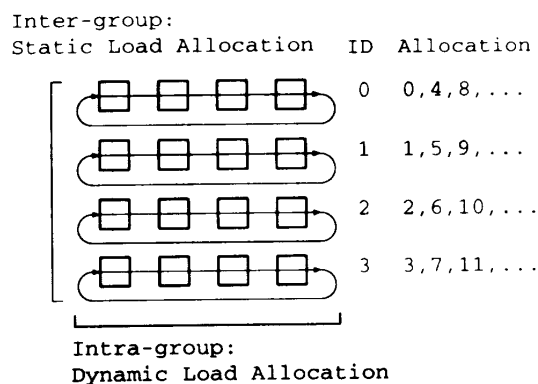
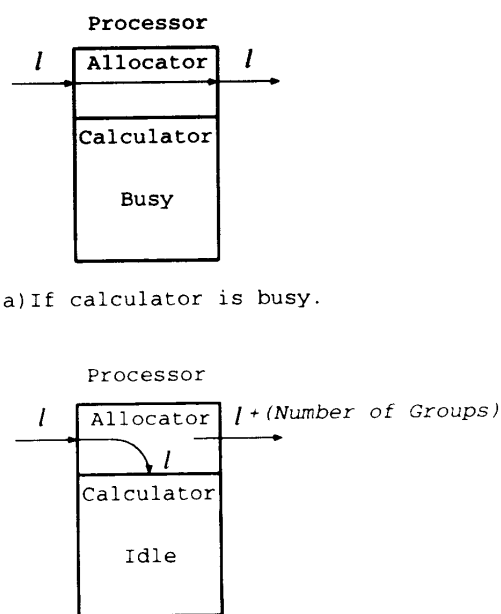


図6 グループ分割による動的負荷分散

Fig. 6 Dynamic load balancing by grouping.



(b) If calculator is idle.

図7 動的負荷分散のための2つのスレッド

Fig. 7 Two threads for dynamic load balancing.

各グループにおいて、ライン番号は各々のグループ番号(ID)から開始する。そして、図7に示すように、計算中の場合はライン番号をそのまま次のプロセッサに送り出す。待機中の場合にはそのライン番号を計算部に伝えると同時に、グループ数を加えたものを次のプロセッサへ送り出す。このようにして、空いているプロセッサに次々と処理を割り当てていく。

動的負荷分散法では各プロセッサの担当するライン数が不定のため、各プロセッサはヘミキューブ全体のためのメモリを確保している。

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

| シーン | 解像度 | 立方体数 | ポリゴン数 | パッチ数 |
|-----|-----|------|-------|------|
| 1 | 192 | 75 | 462 | 1074 |
| 2 | 160 | 75 | 462 | 1074 |
| 3 | 192 | 75 | 462 | 624 |
| 4 | 192 | 75 | 462 | 2874 |
| 5 | 192 | 25 | 162 | 1074 |

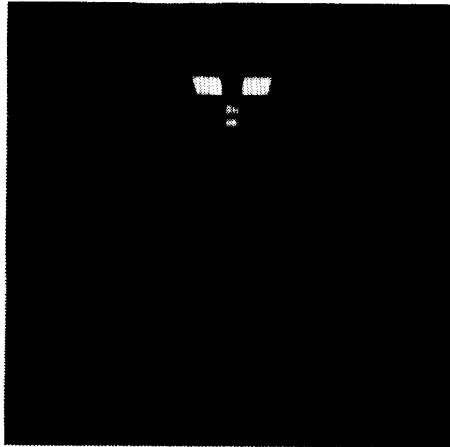


図8 ベンチマークシーン1
Fig. 8 Benchmark scene 1.

5. 並列化の実験と考察

5.1 実験方法

静的負荷分散, 動的負荷分散の2通りの並列化方法で, ラジオシティ法の計算を実行する. 1台のプロセッサを用いた場合と, トーラス結合された4, 16, 32, 64台のプロセッサを用いた場合, それぞれの処理時間を測定する. ここでの処理時間とは, データを全プロセッサに送るところから始め, ラジオシティの計算を行い, 結果をルート・プロセッサに回収し終わるまでの時間とする.

実行する内容は, 表1に示すような5通りのベンチマーク用のシーンにおいて, 漸進法の放射を100回行うものとする. 全シーンに共通して, 天井, 床, 4枚の壁で囲まれた空間内に光源がある. 天井と床そして4枚の壁のポリゴンは各々 5×5 個のパッチに, 光源の6個のポリゴンは, 各々 2×2 個のパッチに分割されている. シーン1のラジオシティの計算結果をレンダリングしたものを図8に示す. シーン1~4の立方体の配置は同じで, ヘミキューブの解像度とパッチへの分割数が異なる. シーン5は配置が異なり, 立方体群が3段の中央の1段のみになっている. なお, すべてのシーンはローカルメモリ上で実行可能な規模としており, メモリ量が最大のシーン4は約110kバイ

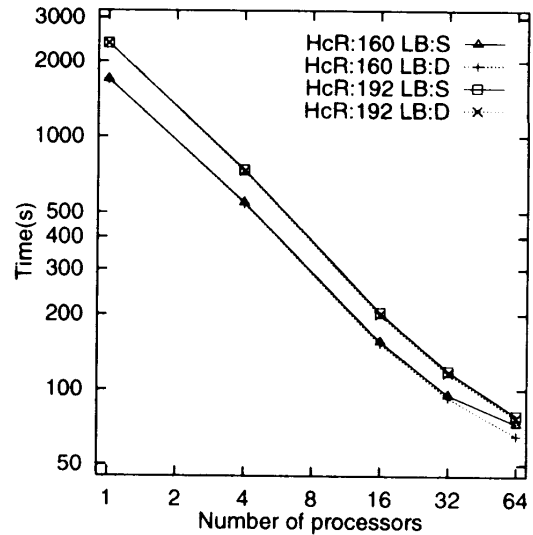


図9 ヘミキューブの解像度による比較
Fig. 9 Comparison based on the resolution of hemicycubes.

トである.

5.2 結果と考察

各シーンの処理時間のうち, 並列化可能部分(検索, フォームファクタ計算, 放射)の割合は99.2~99.5%となり, この部分によって全体の処理時間がほぼ決定されていることが分かる. なお, 並列化の評価において代表的な指標となる速度向上は, プロセッサ64台の場合, 1台に対して23~37倍, 平均で30倍となった.

5.2.1 ヘミキューブの解像度による比較

シーン1, 2を用いたヘミキューブの解像度による比較を図9に示す. ここで, HcRはヘミキューブの解像度, LBは負荷分散法で, S, Dがそれぞれ静的負荷分散, 動的負荷分散を表す. 図9より, プロセッサ数が32台まではプロセッサ数の増加とともに処理時間がほぼ直線状に減少することが分かる. また, 解像度192は160に対してプロセッサ1台の場合で約1.4倍, プロセッサ64台の場合で約1.2倍の処理時間となっている. ここで, $192/160 = 1.2$ であるから, 解像度を n とすると, プロセッサ数の増加とともに処理時間が $O(n)$ に近づいている. 負荷分散法による比較では, 解像度160でプロセッサ数64台の場合, 負荷分散による差が約12%と大きく表れている. これは, 160が64の倍数ではなく, 静的負荷分散では各プロセッサの担当ライン数に差が生じるためと考えられる. 一方, 動的負荷分散では, 160はプロセッサ64台の場合のグループ数である8の倍数であり, 各グループの担当ライン数が等しくなるため, 負荷分散がうまく働いている. プロセッサ32台以下の場合, 静的負荷

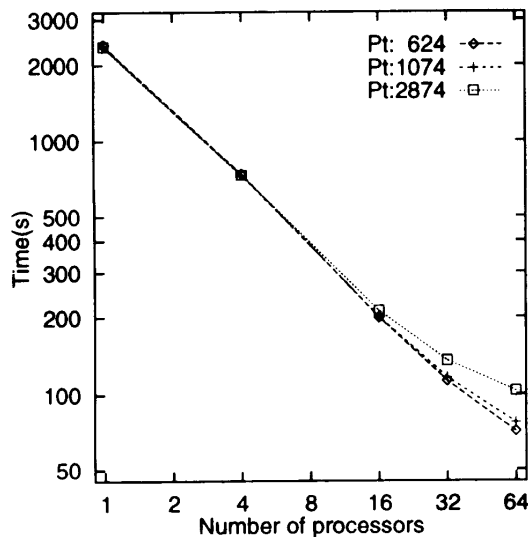


図 10 パッチ数による比較

Fig. 10 Comparison based on the number of patches.

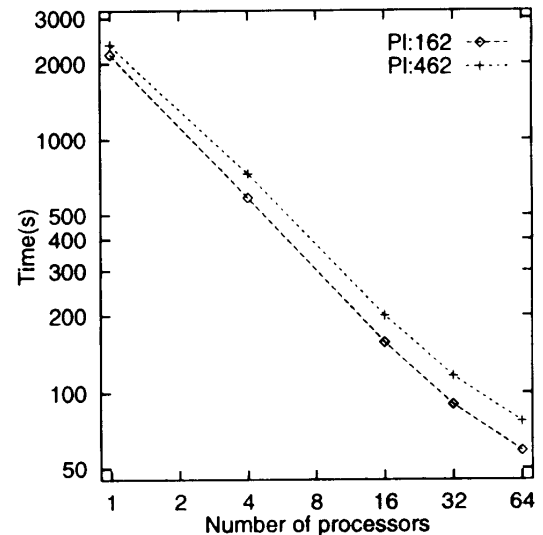


図 11 ポリゴン数による比較

Fig. 11 Comparison based on the number of polygons.

分散と動的負荷分散の差はほとんど表れていない。

5.2.2 パッチ数による比較

シーン 1, 3, 4 を用いたパッチ数による比較を図 10 に示す。ここで、Pt はパッチ数を表す。負荷分散法はすべて動的としている。図 10 より、プロセッサ 1 台の場合、処理時間はパッチ数とほとんど無関係であることが分かる。これは、3.3 節で述べたように、ポリゴンにヘミキューブに描画しているからである。ここで、プロセッサ 1 台での処理時間が $T(1)$ 、 n 台での処理時間が $T(n)$ の場合の並列化効率を $T(1)/nT(n)$ と定義すると、パッチ数が少ないほど並列化効率が高いことが分かる。これは、フォームファクタの並列計算の際、プロセッサ間の通信量がパッチ数に比例して増加するのに対し、計算量は一定であるため、パッチ数の増加とともに並列化効率が低下するものと考えられる。プロセッサ数が多くなるにつれて計算時間の割合は小さく、また、通信時間の割合は大きくなるため、この現象はより顕著に表れる。

5.2.3 ポリゴン数による比較

シーン 1, 5 を用いたポリゴン数による比較を図 11 に示す。ここで、Pl はポリゴン数を表す。負荷分散法はすべて動的としている。ポリゴンのパッチへの分割数を調整することにより 2 つのシーンともパッチ数を同じ 1074 にしている。図 11 より、ポリゴン数を増加させると処理時間が増加することが分かる。これは、ヘミキューブへ描画するポリゴン数が増加するためである。また、ポリゴン数を 162 から 462 にすると、プロセッサ 1 台の場合、処理時間は約 1.1 倍となるが、プロセッサ 4 台から 64 台の場合、ほぼ 1.3 倍

前後となっている。すなわち、並列処理の場合、処理時間が逐次処理の場合よりも増加している。この原因として考えられるのは、プロセッサ間の描画時間の差である。動的負荷分散を用いてもプロセッサ間の処理時間をまったく同一にはできない。ポリゴン数が多いほど、この処理時間の差が大きくなっていると考えられる。

6. おわりに

本論文では、マルチトランスピュータシステム上で、ヘミキューブを細分化してラジオシティ法の計算を並列化する方法と、その効果について述べた。

動的負荷分散法は、解像度がプロセッサ数の倍数でなくても、グループ数の倍数であるならば効率の低下が起こらず、静的負荷分散法よりも有効であることが分かった。また、処理時間はヘミキューブの解像度に比例することが分かった。さらに、並列化効率はパッチ数に大きく依存し、パッチ数が少ないほど良いこと、および、ポリゴン数が多いほど処理時間が増加し、逐次処理の場合よりも並列処理の場合の方が大きく増加することが分かった。

今後の課題として、ポリゴンやパッチ数を増加させた場合の並列化効率の改善、さらに、曲面を含む複雑なシーンの処理などがあげられる。

参考文献

- 1) Goral, C.M., Torrance, K.E., Greenberg, D.P. and Battaile, B.: Modeling the Interaction of Light Between Diffuse Surfaces, *Proc. SIG-*

- GRAPH '84*, Vol.18, No.3, pp.213-222 (1984).
- 2) 西田友是, 中前栄八郎: 陰および拡散照度を考慮した3次元物体の陰影表示, 情処学グラフィクスとCAD研報, 14-2, pp.1-8 (1984).
 - 3) Cohen, M.F. and Wallacek, J.R.: *Radiosity and Realistic Image Synthesis*, Academic Press (1993).
 - 4) Cohen, M.F., Chen, S.E., Wallance, J.R. and Greenberg, D.P.: A Progressive Refinement Approach to Fast Radiosity Image Generation, *Proc. SIGGRAPH '88*, Vol.22, No.4, pp.75-84 (1988).
 - 5) Cohen, M.F. and Greenberg, D.P.: The Hemisphere: A Radiosity Solution for Complex Environments, *Proc. SIGGRAPH '85*, Vol.19, No.3, pp.31-40 (1985).
 - 6) 大谷尚毅, 米田泰司, 日高教行, 浅原重夫, 鷺島敬之: ラジオシティ法の一並列化手法, 情処学グラフィクスとCADシンポジウム, Vol.93, No.6, pp.95-104 (1993).
 - 7) Baum, D.R. and Winget, J.M.: Real Time Radiosity Through Parallel Processing and Hardware Acceleration, *Proc. '90 Symposium on Interactive 3D Graphics*, pp.67-76 (1990).
 - 8) Feda, M. and Purgathofer, W.: Progressive Refinement Radiosity on a Transputer Network, *Photorealistic Rendering in Computer Graphics (Proc. 2nd Eurographics Workshop on Rendering)*, pp.139-148 (1994).
 - 9) 上嶋 明, 山崎勝弘, 渡部 透, 得丸英勝: マルチトランスピュータシステム上でのラジオシティ法の並列化, 並列処理シンポジウム JSP'95, Vol.95, No.2, pp.19-26 (1995).

(平成7年8月29日受付)

(平成8年2月7日採録)



上嶋 明 (学生会員)

昭和43年生。平成3年立命館大学理工学部情報工学科卒業。平成5年同大学院博士前期課程修了。現在、同大学院博士後期課程に在学中。計測自動制御学会、並列処理、コンピュータ・グラフィクスの研究に従事。



山崎 勝弘 (正会員)

昭和28年生。昭和51年京都大学工学部情報工学科卒業。昭和53年同大学院修士課程修了。昭和55年同大学院博士課程中退。同年より宇都宮大学工学部助手。昭和63年立命館大学理工学部情報工学科助教授。平成6年同教授。平成4年から1年間マンチェスター大客員研究員。工学博士。平成6年山内業績賞。並列処理、並列プログラミング、事例ベース推論の研究に従事。電子情報通信学会, ACM, IEEE 各会員。



渡部 透 (正会員)

昭和17年生。昭和40年京都大学工学部精密工学科卒業。昭和42年同大学院修士課程修了。同年京都大学工学部助手。昭和61年同大オートメーション研究施設助教授。昭和62年同大学院応用システム科学専攻助教授兼任。昭和63年立命館大学理工学部情報工学科教授。平成3年同大計算機センター長兼任。平成6年より同大総合情報センター副センター長兼任。現在に至る。工学博士。ロボットやNCなどの計算機制御、生産の管理と制御、CAD/CIM/CE, NN/GA 応用、並列処理計算機などの研究に従事。昭和61年日本自動制御協会(現システム制御情報学会)榎木記念論文賞受賞。電気学会, システム制御情報学会, 日本機械学会, 計測自動制御学会, 精密工学会, 日本ロボット学会, ASME, IIAS などの会員。



得丸 英勝

昭和2年生。昭和25年京都大学工学部応用物理学教室卒業。昭和26年同大工学部応用物理学教室助手。昭和28年同講師。昭和31年同大航空工学教室助教授。昭和37年同大数理工学教室教授。昭和62年同大応用システム科学教室教授。平成1年同大工学部長を兼任。平成3年立命館大学理工学部情報工学科教授となり。平成6年同大理工学部長を兼任。現在に至る。工学博士。昭和45年計測自動制御学会論文賞。昭和56年日本自動制御協会(現システム制御情報学会)榎木記念賞論文賞を受賞。計測自動制御学会, 日本機械学会, 日本材料学会, 日本レオロジー学会, エネルギー・資源学会, 日本防災システム協会, 日本工学アカデミー各会員。