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

立命館大学理工学部

ラジオシティ法による画像生成では，処理の大部分を占めるフォームファクタ計算の高速化を図ることが重要である．本研究では，ヘミキューブの処理を各プロセッサに分割して並列化し，かつ負荷均衡を図るための静的 / 動的負荷分散法を考案した．2 種類の画像生成を対象として，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実行時間がほぼリニアに減少すること，及び動的負荷分散法がやや有利となることがわかった．

This paper describes how to parallelize the enormous amount of calculations for form factors in the radiosity image generation using static/dynamic load balancing algorithms. Two sample scenes are executed on a torus-connected 64 transputers system. The execution time reduces almost linearly and the dynamic load balancing algorithm becomes better than the static one, as the number of processors increases.

1　はじめに

ラジオシティ法は，物体表面を構成するポリゴンを細かいパッチに分割し，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]．

ラジオシティ法ではフォームファクタと呼ばれる，パッチ間で光が到達する割合の計算が重要となる．これは 2 重積分を解くことによって求められるが，パッチ間に障害物がある場合などに積分計算で求めるのは非現実的である．そのため，近似値を求める方法としてヘミキューブ法 [4] が提案された．

初期のころは光の平衡状態から得られる連立方程式を解いて解を求めていたが，漸進法 [5] 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，及びメモリを大幅に節約できる点が非常に有利である．

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．従って，フォームファクタ計算の並列化が重要である．そこで本研究では，ラジオシティ法による画像生成を，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, 25MHz) 上で並列処理する方法と，その効果について述べる．

従来のラジオシティ法の並列化の手法では，ポリゴン，パッチやエレメント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る方法を用いる [7]．この方法は，各プロセッサ間の通信

Parallelization of the Radiosity Method on a Multi-Transputer System
Akira Uejima, Katsuhiro Yamazaki, Tohru Watanabe, and Hidekatsu Tokumaru
Faculty of Science and Engineering, Ritsumeikan University

量が少ないため，トランスピュータシステムのように通信速度が比較的遅い分散メモリ型の並列計算機で有効であると考えられる．ただし，単純にヘミキューブを各プロセッサに等分割したのでは，ポリゴンの配置の偏りが原因で各プロセッサ間の処理時間に差が生じる．そのため，本研究では各プロセッサへの割り当ての際に，単純に処理を各プロセッサに等分割する静的負荷分散法と，実行時に割り当てを行う動的負荷分散法を用いて実験を行い，それぞれの速度向上を示す．

2　トランスピュータシステム

本実験で用いるトランスピュータ T805 は 32 ビットの CMOS マイクロプロセッサで，動作クロック周波数は 25MHz である．メモリは各トランスピュータに 1M バイト備えている．外部との通信は，10Mbps の 4 本の INMOS シリアルリンクを用いて行う．結合網として，$2 \times 2, 4 \times 4$，4×8，8×8 の 4 種類のトーラス，及び 2 台ずつの組み合わせによる 2^5のハイパーキューブを切り替えて使用できる．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 ROOT 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，以下のようになる [1][3]．

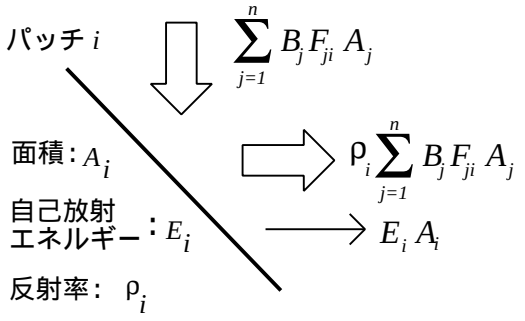

図 1　光の入射と放射

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（入射光を全方向に均等に反射する面）とする．
2. 各ポリゴンを適当な大きさに分割する．分割された一つをパッチと呼び，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，図 1 のように全てのパッチについて光の入射と反射に注目すると，次の式が成立する．

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

ここで，

B_i　：　パッチ i のラジオシティ・エネルギー
(エネルギー / 面積 / 時間)
E_i　：　パッチ i の自己放射エネルギー
(エネルギー / 面積 / 時間)
A_i　：　パッチ i の面積
A_j　：　パッチ jの面積
F_{ji}　：　パッチ jからパッチ i へのフォームファクタ
ρ_i　：　パッチ i の反射率
n　：　パッチ数

ここで，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ある．フォームファクタの対称性 $A_iF_{ij} = A_jF_{ji}$により，式 (1) は次のように表せる．

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

4. 式 (2) より次の連立方程式が立てられ，これを解けば全てのパッチのラジオシティが求められる．

$$\begin{bmatrix} 1-\rho_1F_{11} & -\rho_1F_{12} & \dots & -\rho_1F_{1n} \\ -\rho_2F_{21} & 1-\rho_2F_{22} & \dots & -\rho_2F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_nF_{n1} & -\rho_nF_{n2} & \dots & 1-\rho_nF_{nn} \end{bmatrix} \cdot \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} \qquad (3)$$

5. 求めた各パッチの中心のラジオシティから頂点のラジオシティに変換し，パッチ内部を線形補間してレンダリングする．

3.2 フォームファクタ

二つのパッチ i, jが図 2 の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．

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

二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．そこで，高速に近似値を求める方法として，ヘミキューブ法が研究された．

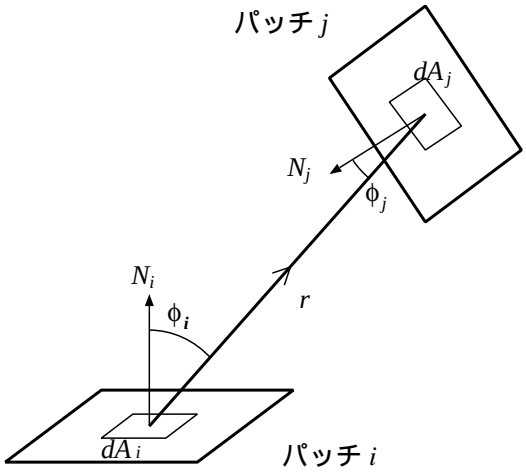

図 2 フォームファクタ

3.3 ヘミキューブ法によるフォームファクタの計算

この方法は，仮想的なヘミキューブ (半立方体) を用いることから，このように呼ばれている [4]．まず，図 3 に示すように，パッチ i を中心に X軸と 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る．次に，この表面を適当な大きさのメッシュに分割する (図 3 の場合の分割数は 8 である)．そして，パッチ i から各メッシュへのファームファクタΔFを次の式を用いて計算する．

上面については，

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

側面 ($y = 1, -1$) については，

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

この計算結果を後の計算のため，ルックアップ・バッファに入れておく．次に，Z バッファ法などを用いて，パッチ jをヘミキューブに投影し，隠面消去によって距離が最も短い物体のみを残す．ルックアップ・バッファを用いて，

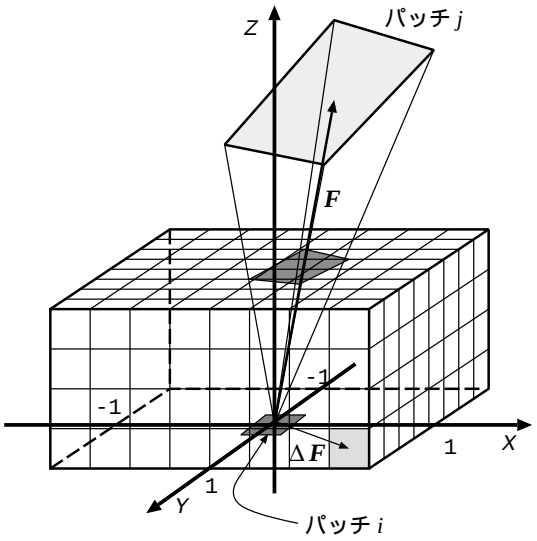

図 3 ヘミキューブ

投影されたメッシュのΔFの和 $(\Sigma\Delta F)$ を計算すると，F_{ij} が得られる．

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消去を行った後，各メッシュがポリゴン内部の，どのパッチに対応するかを計算している．従って，投影数はポリゴン数に等しく，パッチへの分割数と独立である．

3.4 漸進法

式 (2) は各パッチの光の入射に注目しているが，逆に，各パッチの光の放射に注目し，他のパッチへの放射を繰り返すことで，全パッチのラジオシティの平衡状態に徐々に近づけていく方法である [5]．

放射に注目すると，以下の式ができる．

$$B_j = B_j + B_i(\rho_j F_{ji}) \quad (j = 1,2,3,\ldots,n) \tag{7}$$

この式を，ラジオシティの大きいパッチ i から順に適用することで徐々に解に近づけていくことができ，これを収束するまで繰り返す．従来の連立方程式を解く方法では，計算の最

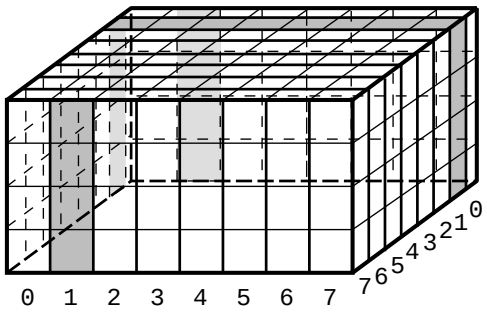

図 4 ヘミキューブの分割

中は式 (3) にある全フォームファクタを使用する必要がある．一方，この方法では一度に使用するフォームファクタは，式 (7) のように行列の縦 1 列分のみでよい．このため，メモリを大幅に節約することが可能である．

4 ヘミキューブの分割によるラジオシティ法の並列化

本研究では，ヘミキューブの 5 面のメッシュを図 4 に示すようにラインごとに分割しておき，ポリゴンをヘミキューブに投影する際に，各ラインを各プロセッサに割り当ててZバッファ処理を行わせる方法を用いる [8]．

各プロセッサでの計算後，全対全通信を行い，全プロセッサが完全なフォームファクタを得る．なお，パッチのデータは各プロセッサに分散して持たせてある．漸進法による計算の際には，図 5 のように，まず各プロセッサ内で最大のラジオシティを持つパッチを決定し，次に，このデータを一箇所に回収して全パッチの中で最大のラジオシティを持つパッチを決定する．そして，このデータを全プロセッサにブロードキャストする．その後，各プロセッサで並列にフォームファクタの計算を行い，各プロセッサが放射に必要とする結果を全対全通信で集める．最後に，各プロセッサが自分の保持するパッチへの放射の処理を行

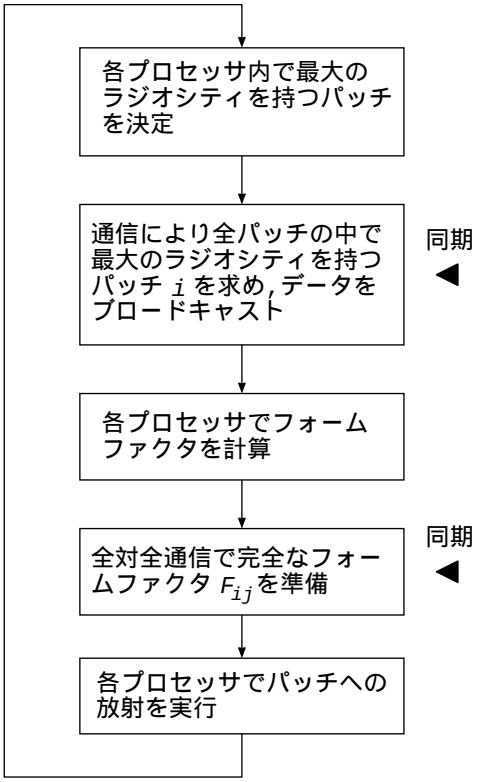

図 5　漸進法によるラジオシティ法の並列化

う . このように , 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 . 1 回の繰り返しの中では , 図 5 に示すように , 全プロセッサが 2 個所で同期をとることになる . なお , パッチのデータを各プロセッサに分散しているため , 1 台ではメモリ不足で処理できないような複雑な画像でも , プロセッサの台数を増やすことにより処理が可能となる .

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 . 具体的には , プロセッサを 0 から $T_n - 1$ までの T_n 台とすると , ヘミキューブのライン l は , $(l \bmod T_n)$ 番のプロセッサが計

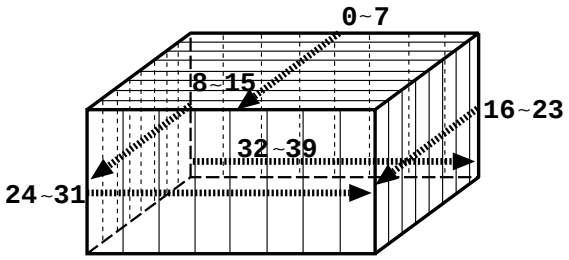

図 6　動的負荷分散での処理の割り当て順序

算を担当する . 例えば , 分割数 8 のヘミキューブをプロセッサ 8 台で並列化する場合 , プロセッサ 1 はライン 1 (図 4 の網掛部分全部) を担当することになる . 同様に , プロセッサ 4 台で並列化する場合には , プロセッサ 1 はライン 1 と 5 を担当することになる .

4.2　動的負荷分散

処理の割り当てに際して , ヘミキューブを (ヘミキューブの分割数)×5 の処理に分割し , 図 6 に示すような順で処理番号を振っておく . この図のように分割数 8 の場合 , 上面が 0 ~ 7 , 側面が 8 ~ 39 となるようにする . これは , 処理時間の大きいヘミキューブの上面の割り当てを最初に行い , その後側面を割り当てることで , 処理をできるだけ均等に分割できるようにするためである .

各プロセッサでは , 制御部 , 計算部の二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 . トークンは処理番号と終了フラグを含んでいる . この処理番号により各プロセッサは計算する部分を指示される . 終了フラグは , ヘミキューブの全ての部分の割り当てが終了したことを意味し , 各プロセッサに全対全通信の開始を指示する .

動的負荷分散法においてプロセッサ数が多くなると , 一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

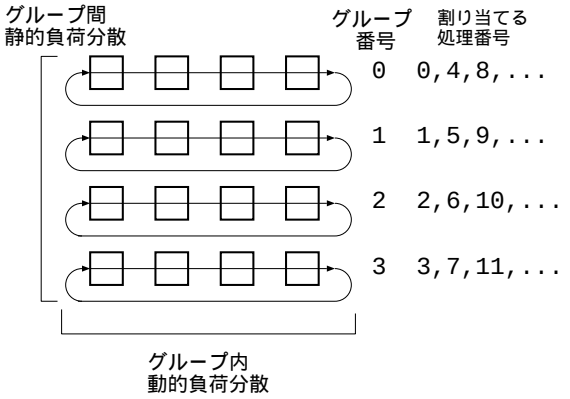

図 7　グループ分割による動的負荷分散

分散を用いる．すなわち，トーラスの横 1 列のプロセッサを一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 (図 7)．各グループには均等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考えられる．

各グループにおいて，図 7 に示すように処理番号は各々のグループ番号から開始し，もしプロセッサが待機中の場合はその処理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．もし計算中の場合は処理番号をそのまま次のプロセッサに送り出す (図 8)．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．

5　並列化の実験と考察

5.1　実験

静的負荷分散，動的負荷分散の 2 通りの並列化方法で，ラジオシティ法による画像生成を実行する．トーラス結合された 4 台, 16 台, 32 台, 64 台のマルチトランスピュータシステム上で，各々の実行時間を測定する．ここでの実行時間とは，データを全プロセッサに送るところから始め，計算を行い，結果を ROOT

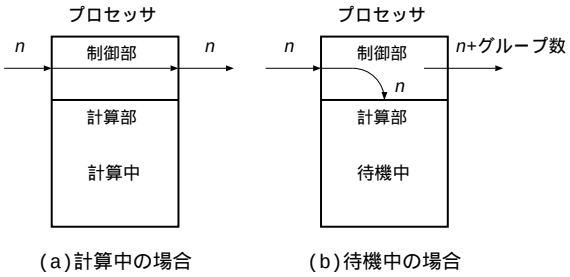

図 8　マルチスレッドによる動的負荷分散の制御

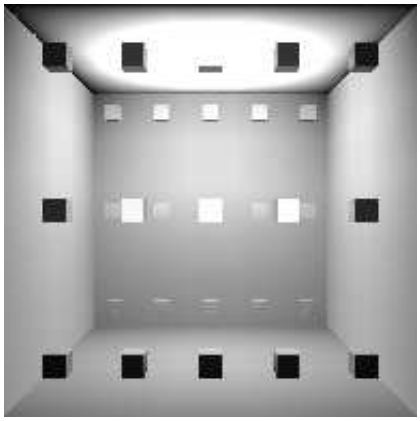

図 9　画像 1　(192 ポリゴン，894 パッチ)

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，図 9, 10 に示すような 2 通りのベンチマーク用の画像において，ヘミキューブの分割数 128, 192 各々について，漸進法の放射を 100 回行うものとする．

画像 1 は 192 ポリゴンを 894 パッチに分割し，画像 2 は 372 ポリゴンを 1614 パッチに分割している．画像 1 は 30 個の立方体，6 枚の壁と光源で構成される．立方体の 6 個のポリゴンは，各々$2 \times 2 = 4$ 個のパッチに分割している．天井と床，そして全ての壁は $5 \times 5 = 25$ 個のパッチに分割している．光源の 6 個のポリゴンは，各々$2 \times 2 = 4$ 個のパッチに分割している．従って，合計は 894 パッチとなる．画像 2 では立方体数が 60 である以外は画像 1 と

図 10　画像 2　(372 ポリゴン，1614 パッチ)

同じである．

5.2　実験結果

静的負荷分散と動的負荷分散の比較を図 11 に, ヘミキューブの分割数による比較を図 12 に示す．ここで，HC はヘミキューブの分割数，S, D はそれぞれ静的負荷分散，動的負荷分散を表している．図 11 より，プロセッサ台数が 32 台，64 台では動的負荷分散の処理時間がやや短いことがわかる．また図 12 より，ヘミキューブの分割数を 1.5 倍にしたとき，処理時間が約 2 倍になっている．

プロセッサ 64 台を用いた場合の実行時間は，ヘミキューブの分割数を 128 としたとき，画像 1 で約 39 秒，画像 2 で約 77 秒である．

5.3　考察

漸進法による計算時間は，パッチ数を n とすると，$O(n)$ となる．画像による計算時間を比べると，画像 2 は画像 1 に対して，プロセッサ 1 台のとき約 1.87 倍，複数台で並列処理のとき約 2 倍となる．パッチ数は 1.81 倍であるから，ほぼパッチ数に比例することがわかる．ただし，並列処理の場合では，7%のオーバーヘッドが生じている．

ヘミキューブの分割数の点からは，分割数

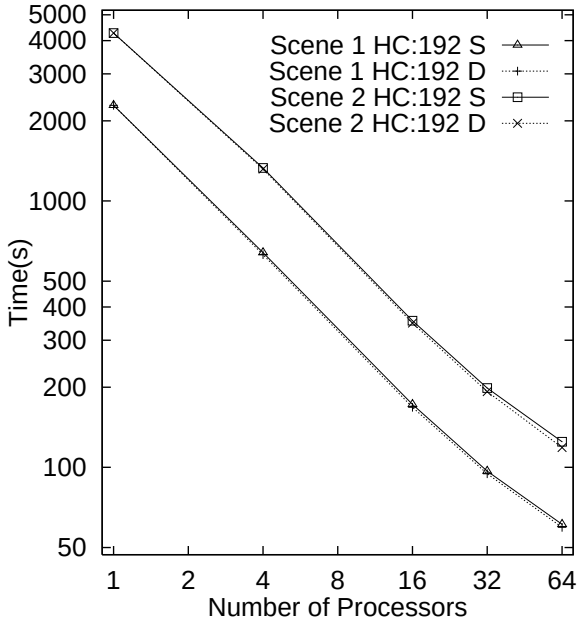

図 11　静的負荷分散と動的負荷分散の比較

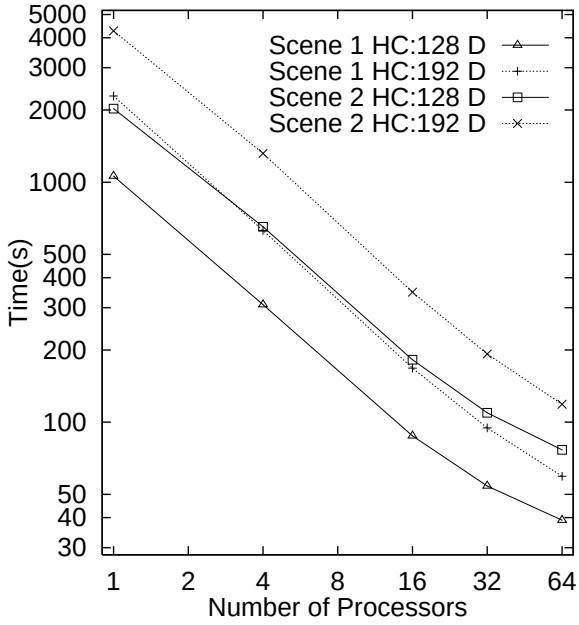

図 12　ヘミキューブの分割数による比較

128 に対して分割数 192 では，画像 1 で 2.16 ～ 1.53 倍，画像 2 で 2.11 ～ 1.55 倍の計算時間となる．つまり，プロセッサ数が増えるに従って，1.5 に近づいている．

メモリの点からでは，ラジオシティやフォームファクタなどのデータを含むパッチのデータを各プロセッサに分散させているため，プロセッサ数を増加させることで，より複雑な画像を扱えるようになる．

ヘミキューブ面の場所によって投影されるポリゴン数に大きな偏りがあるため，各プロセッサごとの処理時間が不均衡になる．ここで，フォームファクタの計算時間は描画処理時間が最大のプロセッサによって決定される．画像 2 では動的負荷分散の処理時間の最大値が，静的負荷分散の場合より約 7%小さいため，動的負荷分散の処理時間が短くなっている．ただし，本研究で用いた静的負荷分散法は，負荷の偏りを小さくするインタリーブ方式 (図 4) で各プロセッサに割り当てを行っているため，32 台以下では動的負荷分散法との差が，ほとんど現れていない．

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ラジオシティ法による画像生成を並列化する方法とその効果について述べた．

その結果，実行時間がほぼリニアに減少すること，実行時間はパッチ数に比例すること，及びプロセッサ台数が多い場合には，動的負荷分散法がやや有利であることがわかった．

今後の課題として，より複雑な画像生成の並列化，及び漸進法の繰り返し数と画質との関連の解明などがあげられる．

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”, *Computer Graphics(Proc. of SIGGRAPH ’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 Wallace J. R.: *“Radiosity and Realistic Image Synthesis”*, Academic Press(1993).

[4] Cohen M. F. and Greenberg D. P.: “The Hemi-cube : A Radiosity Solution for Complex Environments”, *Computer Graphics(Proc. of SIGGRAPH ’85)*, Vol. 19, No. 3, pp. 31–40(1985).

[5] Cohen M. F., Chen S. E., Wallance J. R. and Greenberg D. P.: “A Progressive Refinement Approach to Fast Radiosity Image Generation.”, *Computer Graphics(Proc. of SIGGRAPH ’88)*, Vol. 22, No. 4, pp. 75–84(1988).

[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”, *Proceedings 1990 Symposium on Interactive 3D Graphics*, pp.67–76(1990).

[8] 上嶋明, 山崎勝弘, 渡部透, 得丸英勝: “ヘミキューブの分割によるラジオシティ法の並列化”, 情処学グラフィクスと CAD シンポジウム, Vol. 94, No. 7, pp. 69–76(1994).