計算機アーキテクチャ 111－8
（1995．3．10）

ラジオシティ法によるCG計算を高速化する専用計算機

成見 哲[1]　牧野 淳一郎[2]　戎崎 俊一[1]　大村 皓一[3]

[1]東京大学教養学部宇宙地球科学教室
[2]東京大学教養学部情報図形科学教室
[3]イメージ情報科学研究所

我々はラジオシティ法によるCG計算を飛躍的に加速する専用計算機の概念設計を行った。ラジオシティ計算において最も計算時間のかかる部分はレイとパッチとの交差判定部分である。そこで、ボクセルトラバースの時間を減らすブロックレイトラバースを提案する。これによりレイの並列処理も容易になる。専用計算機は交差判定部分についてのみ加速する。この専用計算機は、8枚のボードで構成され合計512個の交差判定専用プロセッサが搭載される。その他の計算はワークステーションで行う。このようなシステムを20台並列に用いるとラジオシティ計算が10秒以内で行えることが見積もられた。

A Special-Purpose Computer for Radiosity Method in Computer Graphics

Tetsu Narumi[1]　Junichiro Makino[2]　Toshikazu Ebisuzaki[1]　Koichi Omura[3]

[1]Department of Earth Science and Astronomy, College of Arts and Sciences, University of Tokyo
[2]Department of Information Science and Graphics, College of Arts and Sciences, University of Tokyo
[3]Laboratories of Image Information Science & Technology

We have designed a special-purpose computer for radiosity method. In the radiosity method, the most time-consuming part is the ray intersection test. We propose 'block ray traverse' which reduces the time for voxel traverse and makes parallel processing of ray intersection test easy. The special-purpose computer accelerates only the ray intersection test. This computer is composed of eight boards, and has 512 processors for the ray intersection test. Host computer, which is connected to the special-purpose computer, does all other operations. If we use twenty sets of this system in parallel, we can calculate radiosity equation within ten seconds.

1 はじめに

コンピュータグラフィクスの究極の目標は現実の物と見違えるほどのリアリティを持った画像を生成することである。ラジオシティ法[1]はその目標に最も近い方法の一つである（図1)。

ラジオシティ法を使えば、古典的なレイトレーシングによる方法ではうまく表現できなかった間接光や散乱光を正確に計算して求めることが出来る。このようなリアルな画像を生成できる理由は、ラジオシティ法が物体間の光の相互反射を忠実に計算しているからである。例えば、玄関やホールなどの間接照明による柔らかな照明もリアリスティックに表現出来る。このためラジオシティ法によって生成された画像は、照明設計をする上での事前の確認用としても用いられている。

ラジオシティ法はリアリスティックな表現が可能な反面、膨大な計算時間を必要とする。このためラジオシティ法による画像を実際の照明設計に使うためには高速な計算機が不可欠である。ラジオシティ計算を加速するために、並列計算機を開発する[7]、市販の並列コンピュータを使用する[8]などの方法がとられてきた。しかし、効率的な加速はあまりなされていない。

そこで我々は、専用計算機を開発することでラジオシティ計算を飛躍的に加速することを考えた。我々は図1の計算を数秒で行えることを目標として、ラジオシティ法専用計算機の概念設計を行った。画像生成が数秒の内に終れば、照明パラメーターを様々に変えながら思った通りの照明を簡単に設計することが出来る。本論文では、この専用計算機で用いるアーキテクチャを述べ、計算速度の定量的な評価を行う。

第2章では、ラジオシティ法について説明する。第3章では、レイの交差判定を高速化する手法について説明する。この中で、ボクセル分割法のボクセルトラバースを改良したブロックレイトラバースを提案する。第4章では、このブロックレイトラバースを行った時の計算量について定量的に評価する。第5章では、概念設計をした専用計算機のアーキテクチャについて説明する。第6章では、ワークステーションクラスターを作ることで並列処理する方法を説明する。最後に第7章で本論文についてまとめる。

2 ラジオシティ法

この章では、ラジオシティ法について説明する。まず、2.1ではラジオシティ法の基本方程式を説明する。次に2.2ではラジオシティ計算で計算時間の大部分を占めるフォームファクタの計算方法について説明する。

2.1 基本方程式

ラジオシティ法では、物体表面の相互反射（相互拡散反射）を計算する。その単位として、物体の各面を小さく分けた三角形のパッチを用いる。パッチiの単位面積当たりの輝度B_iは以下のような式により決める。

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

ここでB_iはパッチiのラジオシティ(単位面積単位時間あたりに放射する光のエネルギー)、E_iはパッチiの自発光ラジオシティ、ρ_iはパッチiの反射率、F_{ij}はパッチjからパッチiへのフォームファクタ(パッチjからパッチiに達するエネルギーの割合)、Nはパッチ数である。

フォームファクタは、パッチiとパッチjの間に障害物がなければ、

$$F_{ij} = \frac{\cos\phi_i \cos\phi_j}{\pi r^2} A_j \qquad (2)$$

で与えられる。ここでA_jはパッチjの面積、ϕ_i、ϕ_jはそれぞれパッチi、jの面の法線とパッチiからパッチjに向かうベクトルがなす角、rは二つのパッチの中心間の距離である。障害物がある場合は、隠される分だけフォームファクタは減る。

フォームファクタが求まったら(1)の連立一次方程式を解く。ラジオシティ法ではProgressive Refinement Approach[2]がよく用いられている。これはガウスザイデル法とよく似た反復法による解法で、ラジオシティ計算の場合はガウスザイデル法よりもかなり速く収束する。アルゴリズムは以下の通りである。

各パッチにはラジオシティB_iの他にアンショットラジオシティU_iの値を持っている。これは、そのパッチのラジオシティのうちまだ周りのパッチに分配していないラジオシティのことである。つまりパッチが持つアンショットラジオシティ分のエネルギーは他のパッチを明るくするのに役立っていないので、そのエネルギーを他のパッチに分配する必要がある。反復の一回のサイクルでは、まずA_iU_iが最大のパッチを選びそのアンショットラジオシティをフォームファクタに応じて他のパッチに分配する。また放射したパッチのアン

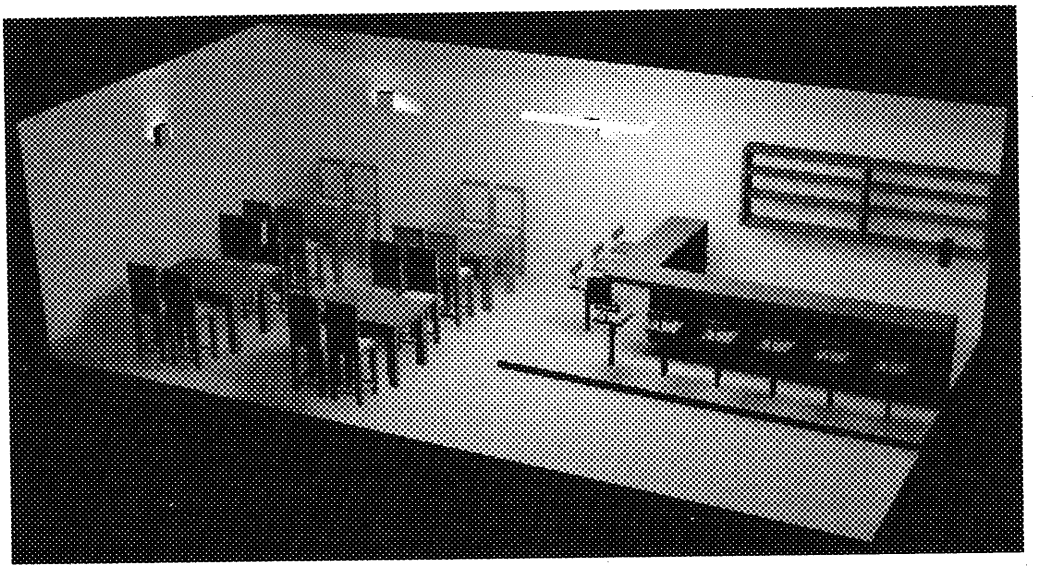

図 1: ラジオシティ法による生成画像例

パッチ数3万、計算時間約3時間、使用計算機IRIS INDIGO ELAN(R3000)。光源として、天井に白色ライト、壁に黄色のアップライト6個がある。次元あたりのボクセル分割数を32として、1000パッチからの放射計算を行った。

ショットラジオシティは0にする。

$$\begin{aligned} B_j^{(k+1)} &= B_j^{(k)} + \rho_j B_i F_{ji}, \quad (j \neq i) \\ U_j^{(k+1)} &= U_j^{(k)} + \rho_j B_i F_{ji}, \quad (j \neq i) \\ B_i^{(k+1)} &= B_i^{(k)}, \\ U_i^{(k+1)} &= 0. \end{aligned} \tag{3}$$

ここで $B_j^{(k)}$ と $U_j^{(k)}$ は k 回目のサイクルでのパッチ j のラジオシティとアンショットラジオシティ、i は A_iU_i が最大のパッチである。ただし初期値は以下のように設定する。

$$B_i^{(0)} = U_i^{(0)} = E_i. \tag{4}$$

2.2 フォームファクタ

ラジオシティ計算において最も計算時間のかかる箇所はフォームファクタの計算である。これは、フォームファクタを計算するには二つのパッチの間を他のパッチがさえぎらないかどうか判定する陰面消去計算が必要であり、しかも求めるフォームファクタの数がパッチ数の自乗個と多いからである。

このフォームファクタの計算には幾つかのアルゴリズムが考案されている [3,4,5]。レイトレーシングによる方法 [4] は、その精度の良さなどからよく用いられている。本論文ではレイトレーシングによる方法を用いる。以下でこの方法を簡単に説明する。

パッチ i からパッチ j のフォームファクタ F_{ij} を求めるとする。まず、パッチ i からパッチ j にレイ（光線）を飛ばす。そしてそのレイと他のパッチが交差しないかどうかの判定を行う（交差判定）。交差しなかったら式 (2) によりフォームファクタを求める。交差した場合はフォームファクタは0にする。

3 ブロックレイトラバースによる交差判定

前章で述べたように、フォームファクタを計算するためには多くのレイの交差判定を行わなければならない。さらに、特に工夫をしなければ一本のレイに対し全てのパッチとの交差判定を行う必要があり、計算量が膨大になる。そこで交差判定を高速化する手法が様々に提案されてきた。このセクションでは、専用計算機での並列処理に適したアルゴリズムとしてブロックレイトラバースによる交差判定を提案する。

3.1では今まで交差判定を高速化するために提案され

てきた手法について簡単に述べる。3.2ではブロックレイトラバースの基本となるボクセル分割法について説明する。最後に3.3でブロックレイトラバースについて説明する。

3.1 交差判定の高速化手法

交差判定を高速化する手法には様々なものがあるが、ボクセル分割法、バウンディングボリューム法、オクトツリー法が広く使われている [6]。いずれも空間を分割して交差判定の回数を減らすように工夫されている。

ボクセル分割法は、空間を均等な直方体（ボクセル）に分割してレイの通るボクセル内の物体とのみ交差判定する。バウンディングボリューム法は、物体ごとにそれを含む簡単な立体（直方体、球など）を用意し、レイとの交差判定はまずその立体と行うことで交差判定の回数を減らす。オクトツリー法は、空間を再帰的に8等分して木構造をつくり、レイとの交差判定はまず根に近い所から行うことで交差判定の回数を減らす。

本論文では、アルゴリズムが単純であることからボクセル分割法を基本としてそれを改良したブロックレイトラバースを提案する。

3.2 ボクセル分割法

以下、ボクセル分割法について説明する。

1. **前処理:** まず、ボクセル分割法では、前処理の第一段階として空間全体を等しい大きさの直方体に分割する。この直方体のことをボクセルと呼ぶ。

   次に第二段階として、全てのボクセルについて、そのボクセル内に入っているパッチを登録しておく。たとえパッチのほんの一部でもボクセルに入っていたならば、そのパッチ番号をメモリに格納しておく。

2. **ボクセルトラバース:** 初期設定が終り実際にあるレイの交差判定をする場合、まずそのレイがどのボクセルを通るかを決定する。その時、始点を含むボクセルから順に終点に向かってボクセルを移動していく。このボクセルの移動をボクセルトラバースと呼ぶ。ボクセルトラバースの処理は、ボクセルトラバースの初期設定と次のボクセルに移動する処理の二つに分けられる。ボクセルトラバースの初期設定ではブロックレイを構成する各平面の方程式などを求める。次のボクセルへ移動する処理は簡単な足し算や比較だけなので高速に行うことが出来る。3.3で説明するブロックレイトラバースではボクセルトラバースの部分が改良されている。

3. **ボクセル内での交差判定:** 一つのボクセル内では、そのボクセルに所属するパッチとの交差判定を行う。ボクセルトラバースによって今いるボクセルが決まったら、そのボクセル内のパッチのデータをメモリから読み出す。そしてそのパッチと交差判定を行う。ボクセル内の全てのパッチとの交差判定が終ったら次のボクセルに移る。

3.3 ブロックレイトラバース

ブロックレイトラバースは、レイのトラバースを高速に行えるよう改良したものである。また、ブロックレイトラバースを行うことによって並列化が容易になるという利点がある。

1. **ブロックレイ:** あるパッチからの放射計算を行う場合、放射パッチから他の全てのパッチへのレイについて交差判定する必要がある。この時レイの始点は放射パッチなので共通である。このためレイの終点が非常に近い二つのレイがあった場合、それらのレイはほとんど同じボクセルをトラバースしていくと考えられる。つまり、まとめて幾つかのレイのトラバースを行うことが出来る。この幾つかのレイの集まりのことをブロックレイと呼ぶ。

   従来のボクセルトラバースではレイごとにボクセルのトラバースを行うので、ブロックレイトラバースはブロックレイに入っているレイの数だけトラバースが速くなる。ただしブロックレイトラバースではその中のレイが通る可能性のあるボクセルには全てトラバースするため無駄なボクセルも通る。この無駄なボクセルを通ることによる計算量の増加は約2.6と見積もられる。

2. **ブロックレイトラバースの方法:** ブロックレイトラバースは次のようにして行われる。

   放射パッチが決まったら、まずあるボクセルを選ぶ。そしてその中にあるパッチへのレイを集めて一つのブロックレイにする。次にブロックレイのトラバースを行う。まず、x, y, z軸のうちブロックレイが一番大きく動く座標軸を求める。そしてその座標軸方向に一つボクセルを移動するごと

に、ブロックレイの断面の範囲を求め、断面が通るボクセル全てをトラバースする。

そのブロックレイのトラバースが終ったら次のボクセルを選び、その中のパッチへのレイを次のブロックレイとする。全てのボクセルへのレイをトラバースし終ったらその放射パッチに関しては終りである。

3. **レイの並列処理:** レイをブロック化することの利点はトラバースが速くなることだけではない。ブロックレイ内のレイについて並列処理が可能な点も大きな利点である。

   今、あるブロックレイに10本のレイが入っていたとする。するとブロックレイがトラバースする全てのボクセル内のパッチと、この10本のレイと交差判定を行うことになる。つまり、あるパッチについてみると10本のレイとの交差判定を並列に処理することが可能になる。一つのパッチのデータを10個のプロセッサが共有することが出来るので並列化しても全体のハードウェアの規模がそれほど大きくならない。

   これに対し、従来のボクセルトラバースはレイごとに行われるため、レイによって交差判定するパッチは変わってくる。そのためレイごとに並列処理するためにはそれぞれのプロセッサが全てのパッチのデータを持っていなければならない。これは全体のハードウェアの規模を大きくし大規模並列化に不利になる。

   このようにブロックレイトラバースは、ボクセルのトラバースを高速化するだけでなく大規模並列化に適している。

4 ブロックレイトラバースによる交差判定の計算量

この章では、ブロックレイトラバースを行った時の計算量について考察する。4.1では、一つの放射パッチから複数のパッチへのレイを一つのブロックレイとした時の計算量の理論値について述べる。4.2では、レイの並列処理をした場合の計算時間について述べる。4.3では、複数の放射パッチから複数のパッチへのレイを一つのブロックレイにした時の計算量について述べる。

4.1 計算量の理論値

まず、放射パッチの数は一つであるとする。一つの放射パッチから複数のパッチへのレイを一つのブロックレイにした場合の一本のレイの平均交差判定時間$\overline{T}$は、

$$\overline{T} = \overline{n}\,\overline{l}\,t_{I_0} + \frac{t_{BT_0} + (\overline{l}-1)\,t_{BT}}{\overline{L}} \tag{5}$$

と表せる。ここで、

t_{I_0} : 一本のレイと一つのパッチとの交差判定にかかる時間 (0.02μ sec)

t_{BT_0} : ブロックレイのボクセルトラバースの初期設定にかかる時間 (9.6μ sec)

t_{BT} : ブロックレイが次のボクセルにトラバースする時間 (0.4μ sec)

$\overline{l}$: 一本のレイがトラバースする平均ボクセル数

$\overline{n}$: 一つのボクセル内で交差判定をする平均パッチ数

$\overline{L}$: 一つのボクセル内にあるレイの端点数の平均

である。

式(5)の第1項は交差判定にかかる時間、第2項はボクセルトラバースにかかる時間である。従来のボクセルトラバースを行った場合、第2項の分母が1になる。つまりブロックレイトラバースではボクセルトラバースの時間が$1/\overline{L}$になる。

t_{I_0}は交差判定専用のLSIをクロック周波数50MHzで使用したとした時の値である。t_{BT_0}, t_{BT}はSun SPARC LXの10倍の実効性能を持つワークステーションを仮定している。

$\overline{n}$, $\overline{l}$, $\overline{L}$は以下のように近似できる。

$$\overline{n} = \frac{N}{v^p} \tag{6}$$

$$\overline{l} = k_l\, v \tag{7}$$

$$\overline{L} = \frac{k_L N}{v^p} \tag{8}$$

ここで、vは次元あたりのボクセル分割数、p, k_l, k_Lは定数であり、$p = 2.5$, $k_l = 2.9$, $k_L = 0.43$とした。これはシミュレーションをして求めた$\overline{n}, \overline{l}, \overline{L}$をうまく近似できるものを求めた。

4.2 並列化をした場合

プロセッサR個でR本のレイを同時に交差判定した場合の計算量について、シミュレーションと理論値とを比較した。理論値の方は式(5)で、t_{I_0}をt_{I_0}/Rに置き換えた。シミュレーションではブロックレイ内のレイの数によってR個のプロセッサが何回交差判定をする

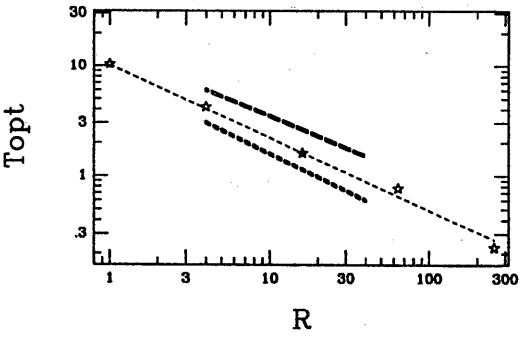

図 2: 並列化した時の計算時間
☆はシミュレーションによる値、破線は理論値。太い破線は、傾きが上から−0.6, −0.7 のインジケータである。

かをカウントした。レイの数が Rで割り切れない時は残りのプロセッサは遊んでいる。また、シミュレーションの方ではレイごとに交差判定の回数とトラバースしたボクセル数を数えて $t_{I_0}, t_{BT}, t_{BT_0}$ を掛けて求めた。

図 2は並列度 Rを変えた時の計算時間をシミュレーションによるものと理論値によるものとで比較している。各並列度でボクセル分割数を変えた時の最適値を求めそれを表示した。ここではパッチ数3013 の部屋についての計算を行った。これから理論値がほぼ正しいことがわかる。また、計算時間は並列度の 0.6〜0.7 乗で減少していることがわかる。

4.3 放射パッチが複数ある場合

複数の放射パッチから複数のパッチへのレイを一つのブロックレイにした場合の一本のレイの平均交差判定時間$\overline{T}$は、

$$\overline{T} = \overline{n}\,\overline{l}\,t_{I_0} + \frac{t_{BT_0} + (\overline{l}-1)\,t_{BT}}{\overline{L}^2} \qquad (9)$$

となる。放射パッチが一つの場合とは第 2 項の分母が $\overline{L}^2$に変わっている。

複数の放射パッチからのレイに関しても受光側と同様に並列処理が可能である。放射側のレイに関して S 個のプロセッサが並列処理した場合には t_{I_0} を t_{I_0}/Sにすればよい。

また、一つのボクセル内のパッチを P個に分割して並列処理した場合も同様に t_{I_0} を t_{I_0}/Pにすればよい。つまり、交差パッチ P、受光パッチ R、放射パッチ S の全ての並列処理を行う場合には、t_{I_0} が t_{I_0}/PRSに置き換わる。

5 専用計算機の概要と計算速度

この章では専用計算機のアーキテクチャと計算速度について述べる。

5.1 概要

専用計算機の概要は図 3のようになる。

まず、ホストコンピュータに 8 枚のボードがつながる。8 枚のボードはそれぞれ独立のパッチ用メモリを持つので、パッチを分割して $P = 8$ の並列化を行うことを意味する。また、一つのボードには $8 \times 8 = 64$ 個の交差判定専用プロセッサが搭載される。これは受光側で $R = 8$、放射側で $S = 8$ の並列化を行うことを意味する。

5.2 計算の流れ

ホストコンピューターにつながれた専用ボードはレイとパッチの交差判定のみを行う。それ以外の計算は全てホストコンピューターが行う。例えば、ボクセルトラバースや陰面消去後のフォームファクタ計算、ラジオシティの収束計算などはホストの担当である。ここでは、レイの交差判定を行う部分の計算の流れについて述べる。

初期設定として、まずホストコンピューターは空間のボクセル分割をし、パッチの位置などのデータを専用ボードに送っておく。

実際のレイの交差判定は以下のようにして行われる。

(*a*) まず、ホストコンピューターは交差判定をするブロックレイを求める。それらのレイの座標（始点と終点の座標）をボードに送る。

(*b*) 次にホストコンピューターはブロックレイによるボクセルトラバースの初期設定を行う。そして、実際にボクセルをトラバースさせて通るボクセルの番号を求め、それをボードに送る。この時は P枚のボード全てに同じデータを送ることになる。

(*c*) ボクセル番号を受けとったボードは、メモリからそのボクセル内のパッチデータを読み、交差判定専用プロセッサに送る。この時、一つのパッチデータが RS個の交差判定専用プロセッサに同時に送られることになる。ブロックレイ内の全てのレイとの交差判定が終ったら計算を終える。

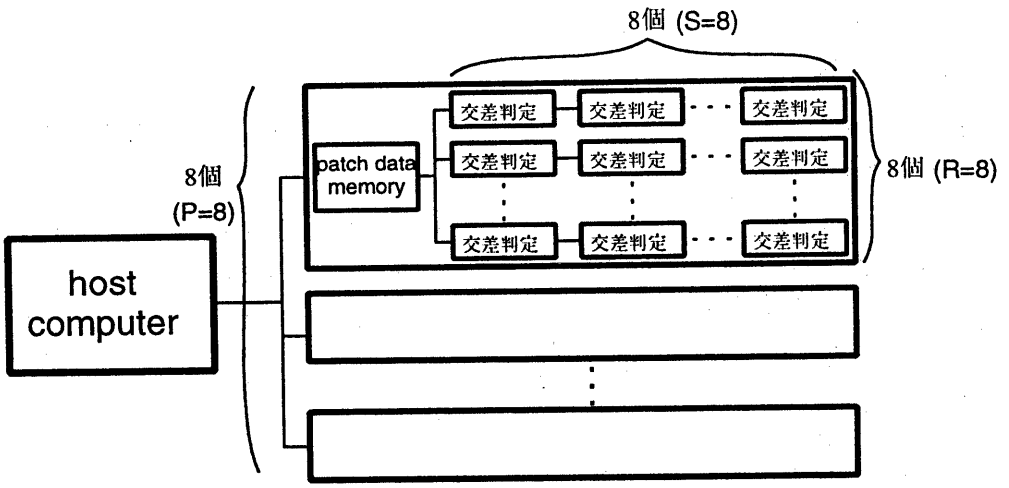

図 3: 専用計算機のブロック図

(*d*) ホストコンピューターは次々とトラバースするボクセルを求め、ボードに送っていく。受けとった各ボクセルについて、ボードは順々に交差判定を行う。

ホストが一つのブロックレイについてトラバースし終ると、ボードからレイごとの交差判定結果を回収する。

5.3 通信量

ホストコンピュータと専用計算機との間での一本のレイあたりの通信量は、

$$4 \times \frac{\bar{l}}{L^2} + 12 \times \frac{R+S}{RS} + \frac{4}{P} \quad \text{(byte)}$$

で与えられる。これは第一項目から順に、ボクセル番号、レイの始点と終点の座標、交差判定の結果をやりとりするのに必要な量である。これを見て分かるように、第2項と第3項は並列度を上げると減少する。また並列度を上げると最適なボクセル分割数は小さくなるので、第1項も小さくなる。つまり一本のレイあたりの通信時間は並列度を上げるほど小さくなる。

次節での評価では、ホストコンピュータと専用計算機との通信速度を 100Mbyte/sec と仮定した。

5.4 計算速度

図4はパッチ数3013の部屋について並列度を変えた時の一本のレイのあたりの計算時間を表したものである。これから計算速度は並列度の約0.7乗に比例して増加していることが分かる。また、並列度 $PRS = 1000$

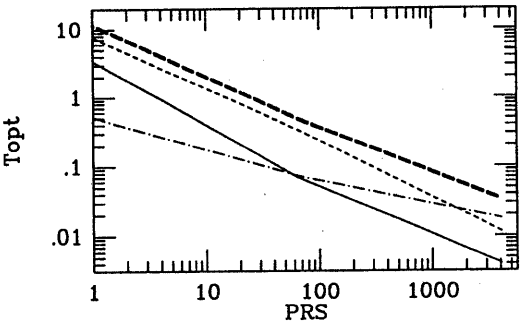

図 4: 専用計算機の計算速度

太い破線が一本のレイの計算にかかる合計時間を示す。細い線は、実線がホストコンピュータの計算時間、破線が専用計算機の計算時間、一点破線が通信時間である。

程度までなら通信時間はボトルネックとなっていないことが分かる。

$P = R = S = 8$ のシステムを考えた場合、一本のレイの計算に 0.12(μsec) かかる。この計算をホストコンピュータのみで行うと 34(μsec) かかる。つまり 280 倍の加速が出来る。

ラジオシティ計算の中でレイの交差判定にかかる時間は全体の 90%以上を占める。仮にこれを 95%としたとき、上記の専用計算機を接続すると全体の計算速度は 18.7 倍速くなる。

6 ワークステーションクラスターによる並列処理

前の章で述べた計算速度は、一台のワークステーションに専用計算ボードを接続した時のものであった。この組合せを何組も作れば並列処理することで更に加速できる。例えば一台のグラフィックワークステーションに20組の専用計算機の接続されたワークステーションをつなげば更に20倍の加速が可能である。

この時一台のワークステーションは、ガウスザイデル法によりあるパッチのラジオシティを求める計算を行う。通信時間は問題にならない。というのは、ワークステーション間で通信しなければならないデータ量は、周りの全てのパッチからのラジオシティを足し合わせて求めた一つのパッチのラジオシティだけでいいからである。このとき通信量は一台のワークステーションにつき数 kbyte/sec 程度でよい。このため 20 台でも問題なく通信を行うことが出来る。ただし、この場合 Progressive Refinement Approach による解法よりも計算時間が数倍程度かかる（完全に収束した値との誤差が1%以下になるまで収束させた場合）。

このようなワークステーションクラスターを考えた場合、図1の計算が10秒以内に終る。

7 まとめ

我々はラジオシティ法による CG 計算を高速化するための専用計算機を概念設計した。またブロックレイトラバースを提案し、これがボクセルトラバースの時間を減らし、並列化も容易であることが分かった。ブロックレイトラバースを採用した専用計算機は8枚のボードで構成され、合計512個の交差判定専用プロセッサが搭載される。この専用計算機を接続したワークステーションをさらに20台接続するとラジオシティ計算が10秒以内で行えることが見積もられた。このシステムは照明設計をする上で非常に役立つものになると思われる。

参考文献

[1] Goral, Cindy M., Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile, 'Modelling the Interaction of Light Between Diffuse Surfaces,' Computer Graphics (SIGGRAPH '84 Proceedings), vol. 18, no. 3, pp. 212-22, July 1984.

[2] Cohen, Michael, Shenchang Eric Chen, John R. Wallace, and Donald P. Greenberg, 'A Progressive Refinement Approach to Fast Radiosity Image Generation,' Computer Graphics (SIGGRAPH'88 Proceedings), vol. 22, no. 4, pp. 75-84, Aug. 1988.

[3] Cohen, Michael and Donald P. Greenberg, 'The Hemi-Cube: A Radiosity Solution for Complex Environments,' Computer Graphics (SIGGRAPH'85 Proceedings), vol. 19, no. 3, pp. 31-40, Aug. 1985. also in Tutorial: Computer Graphics: Image Synthesis, Computer Society Press, Washington, 1988

[4] John R. Wallence, Kells A. Elmquist, and Eric A. Haines. 'A Ray Tracing Algorithm for Progressive Radiosity,' In Proceedings of SIGGRAPH'88 (Atlanta, Georgia, August 1-5, 1988), pages 85-92. ACM, August 1988.

[5] Malley, T.J., 'A shading method for computer generated images,' Master's thesis, Dept. of Computer Science, University of Utah, 1988.

[6] Glassner, A. S., Ed. 'An Introduction to Ray Tracing,' Academic Press, San Diego, 1989

[7] 安部、西村、高畠、平井、中瀬、'画像生成システム SIG2,' 情報処理、グラフィックスと CAD 研究会資料、PRU88-127, pp.65-72, Feb. 1988.

[8] Michelin, S., Maffies G., Arques D., Grossetie J. C., 'Form Factor Calculation: A New Expression with Implementations on a Parallel T.Node Computer,' Computer Graphics Forum (Eurographics '93), vol. 13, no. 3, pp. 421-432, Barcelona, Spain, Sept. 1993.