

# 画像生成におけるスーパーコンピュータの応用例

## Computer Graphics Systems Using Supercomputers

計算機、VLSI技術の目覚ましい進歩に伴い、CGによってリアリティの高い画像を生成することが可能となり、CAD/CAM、工業デザイン、科学シミュレーション結果の画像表示、テレビジョン用商業フィルム、映画などに使用され、広く注目を集めている。しかし、CGは、その膨大な計算量が実用上の問題点となっていた。そこで、高品質な画像を高速に生成するために、スーパーコンピュータS-810を用いた高速画像生成システムのプロトタイプATRASを開発した。このシステムは、Zバッファアルゴリズムを用いた高速画像生成機能、レイトレーシングアルゴリズムを用いた高品質画像生成機能を持っている。本稿では、スーパーコンピュータに適した画像生成アルゴリズムについて述べる。ベクトル処理化によって、スカラー処理の5倍～10倍の高速化が得られた。

栗原恒弥\* *Tsuneya Kurihara*

矢島章夫\* *Akio Yajima*

上西博文\* *Hirofumi Jōnishi*

### 1 緒言

スーパーコンピュータをはじめとする計算機技術及びVLSI技術の目覚ましい進歩に伴い、3次元物体をリアリステックに表示するCG(Computer Graphics)<sup>1)</sup>が進歩してきている。計算機によってリアリティの高い画像を生成することが可能となり、CAD/CAM(Computer Aided Design/Computer Aided Manufacturing)、工業デザイン、科学シミュレーション結果の画像表示、テレビジョン用商業フィルム、映画などに使用され、広く注目を集めている。

CGでは、計算機内に定義された物体の形状、色、反射率、視点の位置、光源の位置などのデータからリアリステックな画像を生成する。画像は、画素と呼ばれる非常に小さな点から構成されている。一画像の画素数は $512 \times 512 \sim 2,000 \times 2,000$ 程度が多い。CGの具体的な処理内容は、各画素で視点から可視な物体を決定し(隠面消去)、その物体の表面での輝度を、面の方向(面の法線)、視点及び光源の位置関係、物体表面の反射特性から求める(シェイディング)ことである。これらの計算は比較的簡単な3次元の行列計算、ベクトル計算である。しかし、画像を構成する画素数は数十万から数百万( $512 \times 512 \sim 2,000 \times 2,000$ )に及び、一画像の生成には膨大な計算量が必要となる。更に、動画像を生成するためには、毎秒24～30フレームの画像が必要であり、画像生成処理の高速化が極めて重要となる。例えば、一画像の生成に1分の処理時間が必要であると仮定すれば、上映時間1分の動画像の生成には、24～30時間の処理時間が必要となる。

これに対して、日立製作所では高品質な画像を高速に生成する技術の研究を進めている。そして、その一環として、スーパーコンピュータHITAC S-810(以下、S-810と略す)を用いた高速画像生成システムのプロトタイプATRAS

(Advanced Three Dimensional Image Generator through Raster Graphics)を開発した。このシステムは、Zバッファアルゴリズムを用いた高速画像生成機能、レイトレーシングアルゴリズムを用いた高品質画像生成機能を持っている。画像生成で高速性が要求される場合には、処理手順の簡単なZバッファ法を用いる。一方、影、反射、屈折などの効果が必要な高品質画像の生成にはレイトレーシングアルゴリズムを用いる。

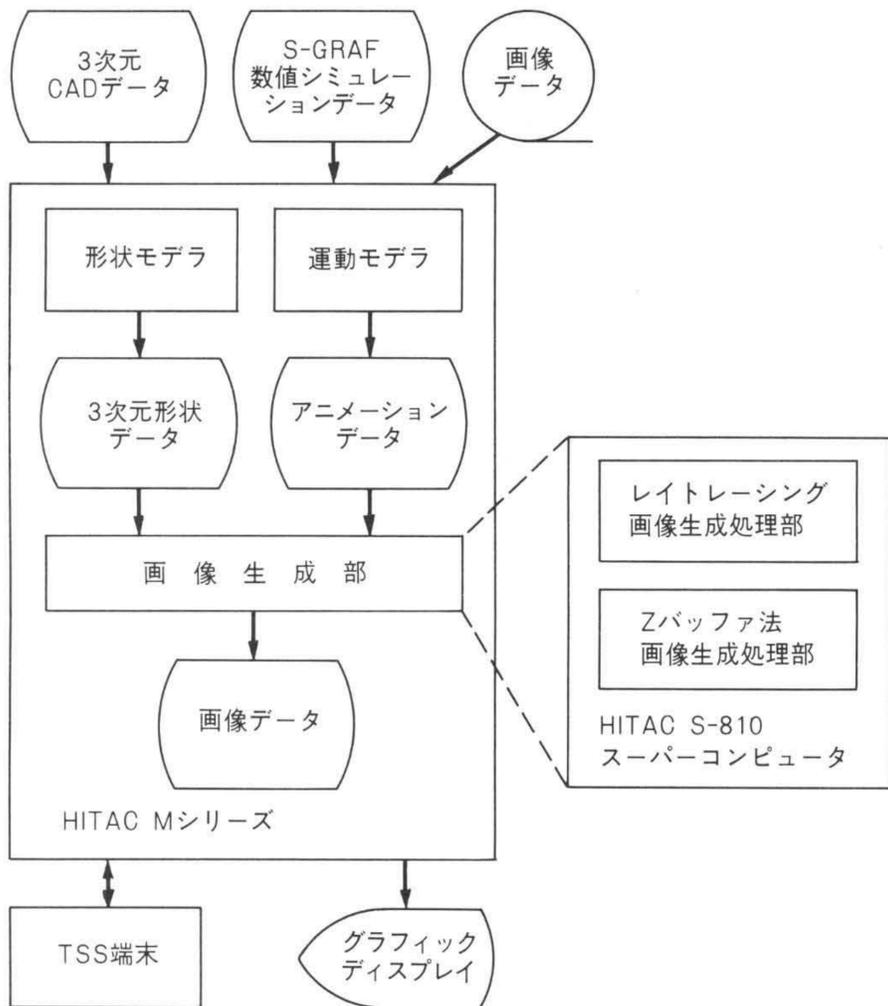
スーパーコンピュータは、表示モデルの柔軟性、システムの拡張性の点で、特に各種アルゴリズムの実験段階では、専用ハードウェアよりも優れていると考えられる。また、スーパーコンピュータの進歩がそのまま画像生成速度の向上に結び付くという利点がある。このため、ATRASでは画像生成処理にスーパーコンピュータを採用した。ベクトル計算機(スーパーコンピュータ)を用いて画像生成処理を高速化するためには、ベクトル計算機に適したアルゴリズムを用いることが重要である。本論文では、ベクトル計算機に適した画像生成アルゴリズムについて述べる。

### 2 システム概要

図1にATRASのシステム構成を示す。同図に示すように、ATRASはHITAC Mシリーズ大形コンピュータのTSS(Time Sharing System)環境下で稼動する。前述のように、計算量が膨大な画像生成処理については、スーパーコンピュータS-810を利用し、バッチ処理で行う。

3次元物体のモデリングは形状モデラによって行う。これは、言語形のモデラであり、物体の形状、属性などを専用の言語を用いて定義する。物体データとして、(1)多角形、(2)二

\* 日立製作所中央研究所



注：略語説明 CAD(Computer Aided Design)  
S-GRAF(Scientific Graphing Facilities)  
TSS(Time Sharing System)

図1 画像生成システムATRASのシステム構成 3次元形状のモデリング及び運動のモデリングには、HITAC Mシリーズ大形計算機を、画像生成にはスーパーコンピュータを用いる。

次曲面及びその集合演算、(3)自由曲面、(4)雲などの密度分布モデル<sup>2), 3)</sup>、(5)フラクタル曲面<sup>4)</sup>、(6)木、草などのプロシジャモデル<sup>5)</sup>を扱える。形状モデラは、物体データを階層的に管理し、それらの拡大、縮小、回転、平行移動を実行する。画像生成を行う前に線画でモデルのチェックを行う。このほかに、自由曲面を基本としたCADシステム<sup>6)</sup>、数値シミュレーション結果表示システムS-GRAF<sup>7)</sup>(Scientific Graphing Facilities)によって変換された形状データを扱える。

運動モデラはキーフレーム補間<sup>8)</sup>によってアニメーションデータを生成する。すなわち、キーとなる時刻での、物体の位置などのパラメータを指定し、キーフレーム間ではパラメータを(スプライン)補間によって求める。

画像生成処理は計算量が膨大なため、スーパーコンピュータS-810を利用する。現在、S-810上に、ベクトル化Zバッファアルゴリズムによる画像生成システム、及びベクトル化レイトレーシングアルゴリズムによる画像生成システムが実現されている。次章でこの二つのプログラムについて詳しく述べる。

### 3 スーパーコンピュータによる高速画像生成

#### 3.1 スーパーコンピュータ<sup>9)</sup>

画像生成処理の高速化を検討する前に、スーパーコンピュータの特性について簡単に説明する。

スーパーコンピュータは、数値計算を高速に処理するために作られた計算機である。数値計算では配列計算が支配的である。スーパーコンピュータは、配列データに対する同一の繰返し処理を演算パイプラインによって高速に実行する。このため、繰返し回数が数十以上の均質な繰返し計算に対しては、はん(汎)用計算機の20倍以上の処理速度が得られる。ここで、例えば配列要素A(i)が配列要素A(i-1)に依存するような場合、すなわち配列データに依存関係がある場合は、パイプライン処理が不可能である。このような場合にはパイプライン処理は行われず、スカラー処理を行うため、はん用の大形計算機並みの性能しか得られない。以下、このパイプライン処理をベクトル処理あるいは単に並列処理と呼ぶ。なお、ベクトル処理によって高速化されるのは最内側ループだけである。

スーパーコンピュータの性能を引き出すための条件は、次の2点である。

(1) プログラム中のベクトル処理される割合(ベクトル化率)が高いこと(条件1)。

スーパーコンピュータでは、スカラー処理の性能とベクトル処理の性能が大きく異なる。このため、ベクトル化率が低い場合には、スカラー処理の処理時間が支配的となり、ベクトル処理の効果が発揮されない。

(2) ベクトル長(繰返し回数)が十分に長いこと(条件2)。

1回のベクトル処理で処理されるデータ量をベクトル長と呼ぶ。ベクトル命令は、立上りのオーバーヘッドがあるため、ベクトル長を十分に長くしてパイプライン処理の効果が得られるようにする。

画像生成処理をスーパーコンピュータで高速実行するためには、以上の条件を満たしたアルゴリズムを採用することが重要である。

#### 3.2 画像生成アルゴリズム

今日、画像生成処理にしばしば用いられている隠れ面消去アルゴリズムは、

- (1) Zバッファアルゴリズム
- (2) レイトレーシングアルゴリズム

の二つである。(1)は高速な処理が特徴である。しかし、影の表現が困難なこと、反射・屈折の表現が不可能という表示処理上の問題点がある。これに対して(2)は、影、反射・屈折の表現が可能であり極めて高品質な画像が生成できる。しかし、計算量が(1)の10倍~100倍と膨大であるという問題点を持っている。このため、ATRASでは、二つのアルゴリズムを用意し、アプリケーションの必要性によって使い分けている。

#### 3.3 Zバッファアルゴリズム<sup>1)</sup>

Catmullによって提案されたこの方法は、隠れ面消去アルゴリズムのなかで最も簡単なものである。各画素のZ値(奥行き値)を保持するZバッファを用いて、画素単位に隠面消去を行う。

この方法の特徴は、

- (1) 処理が単純でハードウェア化に適する。
- (2) 扱える形状(多角形、プロシジャモデルなど)に制限がない。

(3) 形状データ量に制限がない。  
の3点である。  
その問題点は、

- (1) 各画素のZ値(奥行き値)を保持するZバッファのために大量のメモリが必要となる。
  - (2) 画素単位に隠面消去を行うため、透明表示、アンチエイリアシング(境界部分のぎざぎざの除去)、反射・屈折処理が困難である。
- の2点である。

アルゴリズムを以下に示す。

INTENSITY : array [ 1 : HEIGHT, 1 : WIDTH ] of PIXEL ;

(\* 画像データを格納する配列(フレームバッファ) \*)

DEPTH : array [ 1 : HEIGHT, 1 : WIDTH ] of REAL ;

(\* 奥行きデータを格納する配列(Zバッファ) \*)

(1) 各画素 <<IX, IY>> について

INTENSITY [IX, IY] := background value ;

DEPTH [IX, IY] := maximal value ;

(2) 各多角形について、多角形内部のすべての画素 <<IX, IY>> を求め、各画素 <<IX, IY>> について、

(a) 画素 <<IX, IY>> での奥行きZを求める。

(b) if Z < DEPTH [IX, IY]

then

begin

DEPTH [IX, IY] := Z ;

INTENSITY [IX, IY] := (その多角形表面の輝度)

end ;

(3) 全多角形の処理が終了後、配列INTENSITYには求める画像データが格納されている。

多角形で近似された曲面をZバッファアルゴリズムを用いて表示する処理手順を以下に示す。なお、曲面を多角形の集合で近似して滑らかに表示するスムーズシェイディングの方法としては、多角形の頂点の輝度を補間する輝度補間法を用いる。

(a) 座標変換

世界座標系で定義されている図形を透視変換によってスクリーン座標系に変換する。スクリーン座標系は、スクリーンの奥行き方向をZ軸とする座標系である。

(b) クリッピング、後方面の除去

与えられたウィンドウに関与する部分を切り出す(クリッピング)。更に、視点の反対方向を向いている面を処理対象から除去する。

(c) 輝度計算

Phongの反射モデル<sup>9)</sup>などを用いて多角形頂点での輝度を求める。

(d) 走査変換、隠面消去

すべての多角形について多角形内部の画素を求める(スキャンコンバージョン)。次に、各画素での奥行きZを線形補間によって求め、Zバッファを用いて隠面消去を行う。輝度を線形補間して滑らかな陰影付けを行う(スムーズシェイデ

ィング)。

上記の処理をベクトル化した。(a)~(c)については多角形単位あるいは多角形の頂点単位でのベクトル処理化が可能である。このとき、ベクトル長は全多角形数あるいは全多角形の頂点数となり、スーパーコンピュータの性能を引き出すために十分である。

(d)は、全体の処理時間の70~80%を占めるが、従来の方法を単純にベクトル化しただけではベクトル長が短いため十分な性能が得られない。このため、新たにベクトル計算機に適したアルゴリズムを開発した。詳細を以下に述べる。

(d)の処理内容の詳細を以下に示す。

多角形の存在するすべてのスキャンラインについて

(i) 多角形のすべての辺とスキャンラインとの交点を求める。

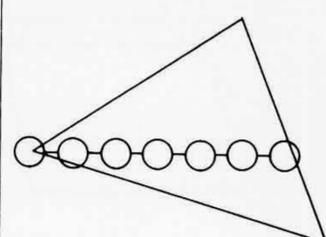
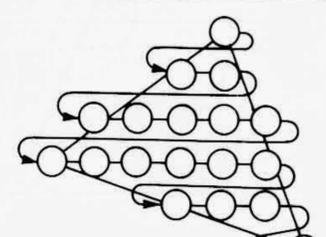
(ii) 交点对の間の画素について隠れ面消去、輝度補間を行う。

(i)の処理は多角形の各辺について行う。このとき、ベクトル長は各辺の両端点のY座標の差である。これは、表示する図形に依存するが、一般に小さい(1~20程度)。このため、ベクトル化してもほとんど高速化されない。そこで、多くの多角形の辺をいったんキュー(Queue)に格納し、キューが一杯になった時点で、すべての辺について一括して交点計算を行う方法を採用した。このとき、ベクトル長はキューの大きさ(ATRASでは500)となり、十分な性能が実現できる。

(ii)の処理もベクトル長は1本の走査線と多角形の辺との交点間の画素数になる(表1)。これも表示する図形に依存するが、一般に小さい(1~20程度)。このため、リストベクトル(間接アドレス指定)を用いて多角形の全画素をベクトル処理する方法を開発した。この方法では、前処理で、多角形内の全画素をいったんリストに格納する。リストの作成が終了した後、多角形内の全画素について隠れ面消去、輝度補間処理をベクトル処理で行う。この場合、ベクトル長は、多角形内部の全画素数(50~200程度)となり、大幅な高速化が実現できる。

以上、多角形の表示について述べた。

表1 スキャンコンバージョン処理方法の比較 リストベクトル法では、多角形内の全画素についてベクトル処理するため、ベクトル長は100程度となり高速処理が実現できる。

No.	1	2
方法	従来法	リストベクトル法
内容		
	○ : 画素	
ベクトル処理対象	単一の走査線	多角形内の全画素
ベクトル長	~10	~100

Zバッファ法による二次曲面の表示は、次のように行う。まず、曲面の存在する走査線を求める。これは、二次方程式の解の存在条件から簡単に求められる。次に、それらの走査線について、曲面が存在する領域を求める。この領域の画素について二次方程式を解き、奥行きZを求め、隠面消去、輝度計算を行う。ベクトル処理は、一走査線上の画素について行う。一般に、二次曲面では、一つの曲面で画面上で比較的大きな物体を定義することが多いため、一走査線上に存在する画素数は多く、上記のベクトル処理でも十分な性能が得られる。

### 3.4 レイトレーシングアルゴリズム<sup>10)</sup>

レイトレーシングアルゴリズムは、光源から照射された光が物体表面で反射し、視点に入射する現象を視点の側からシミュレートする。視点とスクリーン上の一点(画素)を通る半直線(これを視線と呼ぶ)を定義し、その視線と最初に交差する物体を求める。この物体が視点から可視な物体であり、交点での輝度を計算する。物体表面の輝度は散乱光、鏡面反射光、透過光から成ると仮定する。影処理は、輝度を計算する点を仮想的な視点として、各光源に向けて視線を定義し、各物体と交点計算を行う。視線と交差する物体の透過率を光源からの光強度に乗ずる。反射・屈折処理は、反射・屈折処理方向に視線を追跡することによって実現する(図2)。レイトレーシングアルゴリズムの特長を以下に示す。

(1) 反射・屈折が容易に表現できる。

(2) 視線単位の並列計算に適す。

問題点は、以下に述べるとおりである。

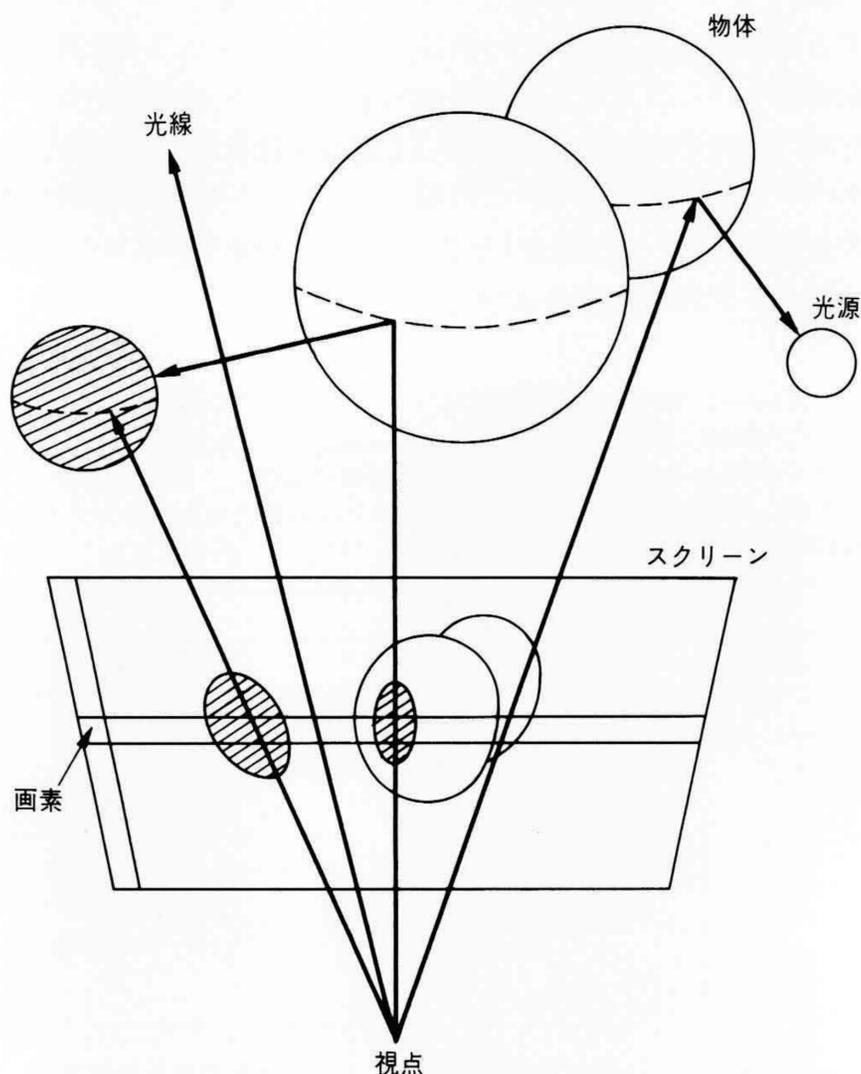


図2 レイトレーシングアルゴリズム 物体表面で反射して視点に入射される光を、視点から追跡して画像を生成する。

(1) 光線と物体との交点計算のために、膨大な計算時間を必要とする(Zバッファ法の $10^1 \sim 10^2$ 倍程度)。

レイトレーシングアルゴリズムでは、物体と視線との交点計算が全体の処理量の90%以上を占める。このため、この部分の高速化が重要である。なお、この方法は、前述のように視線単位の並列性があるため、ベクトル処理化は容易である。交点計算のベクトル化は以下のように行う<sup>11)</sup>。

(1) 交点計算を行う物体と視線とをキューに格納する。

(2) キューが一杯になった時点で、キュー内の物体と視線について交点計算をベクトル処理で実行する。このとき、ベクトル長はキューの大きさ(ATRASでは500)となり、大幅な高速化が実現できる。

以上により、レイトレーシングアルゴリズムで最も計算量の多い処理である交点計算をベクトル化することが可能となった。

なお、物体数が多い場合にはすべての物体に対して視線との交点計算を行うと、効率的でない。このため物体モデルが存在する3次元空間を等間隔に分割して(ボクセル(VOXEL)分割と呼ぶ)、交点計算の回数を削減する方法を採用した。各物体に対して、その物体が存在する領域(ボクセル)をあらかじめ求めておき、物体と視線との交点計算では、視線と交点を持つボクセルを視点に近い順に追跡し、そのボクセルに登録されている物体とだけ交点計算を行う。この結果、単純な方法と比較して、10倍~20倍程度の速度向上が得られている。

多角形、二次曲面については視線との交点計算は容易である。また、密度分布モデルについても、密度は3次元空間内の格子点に与えられるため、視線との交点計算は容易である。しかし、自由曲面については視線との交点計算は解析的には解けない。このため、ニュートン法を用いて交点計算を行っている。ニュートン法を用いる場合、収束計算での初期値の与え方が重要である。ATRASでは、隣接した画素については、同一の曲面と交点を持つ可能性が高いことに着目し、隣接した画素での交点のパラメータを初期値として利用する方法を採った。このため、ほとんどの場合ニュートン法は1回で収束し、計算量を削減できた。この方法はスカラー処理に適しているためベクトル処理化は行っていない。

木や草のプロシジャモデルは、非常に小さな粒子(パーティクルと呼ぶ)の集合によって構成されている。パーティクルの数は1本の木について1,000程度である。このため、視線との交点計算は、可能ではあるが計算量が膨大になってしまう。このため、木や草のプロシジャモデルのレイトレーシングによる表示は現在までのところ困難である。

## 4 適用例

ATRASの表示例を図3~10に、また、CPU時間を表2に示す。図3~4は、3次元CADシステムで作成したモデルを表示したものである。このCADシステムは、形状データとして自由曲面を用いている。この自由曲面データを多角形に分割して、ATRASでZバッファ法を用いて画像生成処理を行った。画像の解像度は $1,024 \times 1,024$ である。ベクトル処理化によってスカラー処理の5倍~6倍の高速化が実現されている。

図5はNaClの分子モデルをZバッファ法で表示したものであ

表2 画像生成のCPU時間 ベクトル化することにより、性能は5倍～10倍向上する。

No.	表示対象	a スカラー 処理 (従来法) 時間 (s)	b ベクトル 処理 (従来法) 時間 (s)	c ベクトル 処理 (提案法) 時間 (s)	速度比 (a/c)	備考
1	図3 受話器	0.55	0.192	0.091	6.1倍	Zバッファ法
2	図4 クリーナ	1.19	0.432	0.239	5.0倍	Zバッファ法
3	図5 分子モデル	7.69	2.45	0.80	9.6倍	Zバッファ法
4	図6 風景	580	255	124	4.7倍	Zバッファ法
5	図7 分子モデル	300	180	52	5.8倍	レイトレーシング
6	図8 球	97	48	17	5.7倍	レイトレーシング

使用計算機：HITAC S-810/20



図5 分子モデル Zバッファ法による二次曲面の表示例を示す。

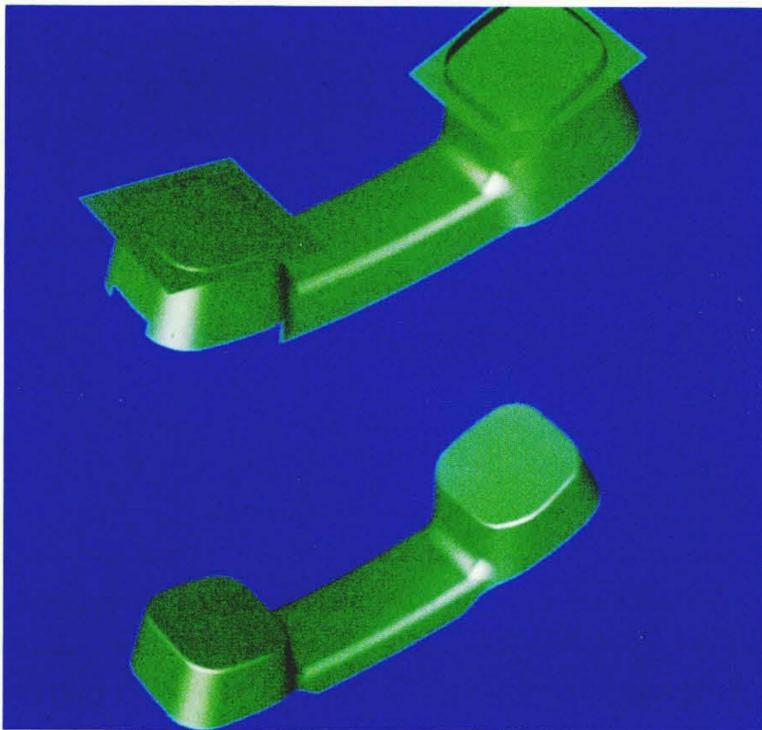


図3 受話器 Zバッファ法によるCADデータの表示例を示す。



図6 風景 Zバッファ法によるフラクタル曲面、草木のモデルの表示例を示す。

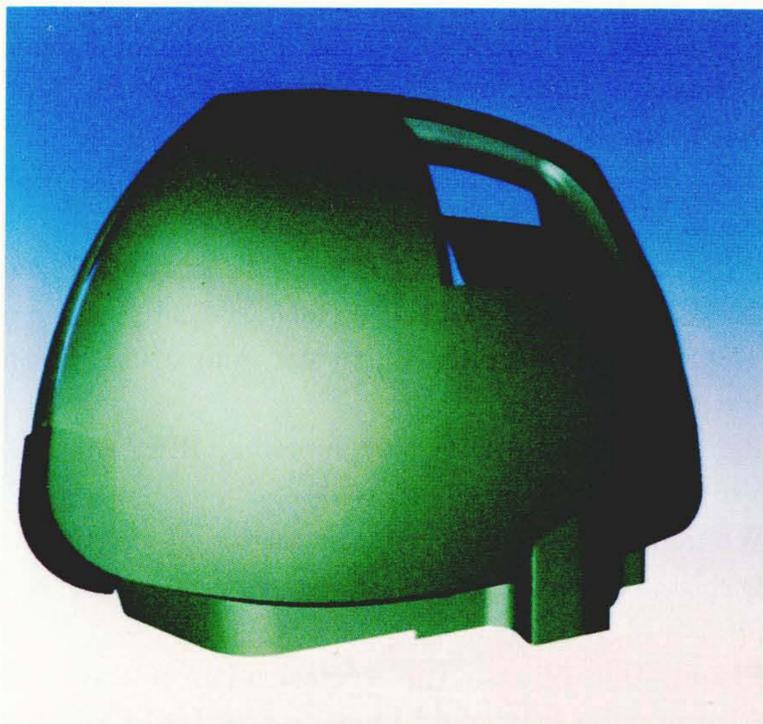


図4 クリーナ Zバッファ法によるCADデータの表示例を示す。

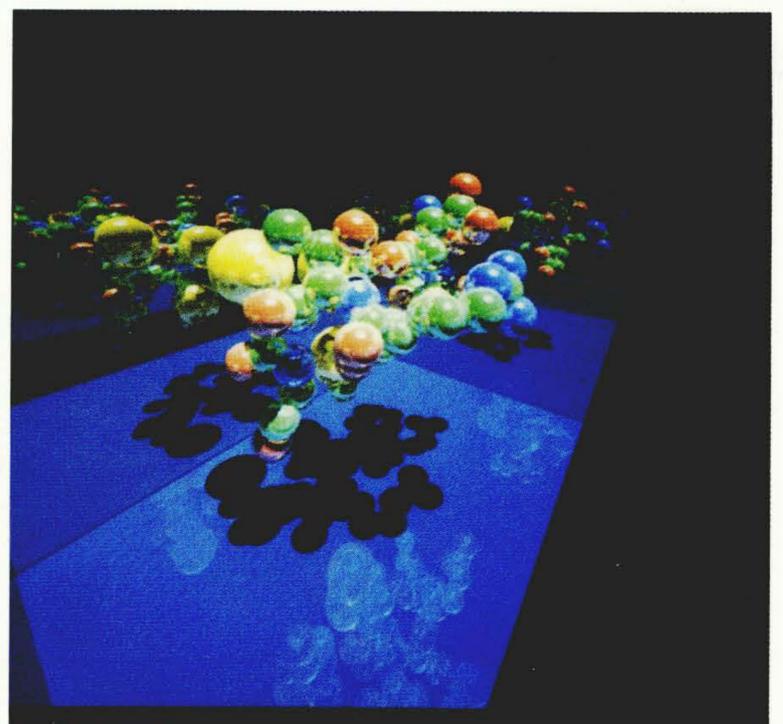


図7 分子モデル レイトレーシング表示例を示す。

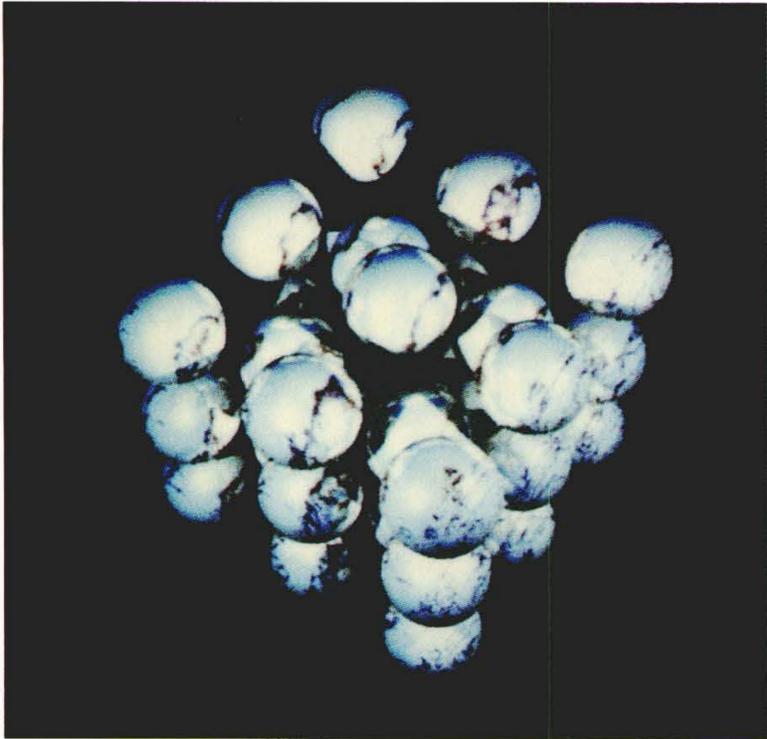


図8 球 レイトレーシング表示例を示す。

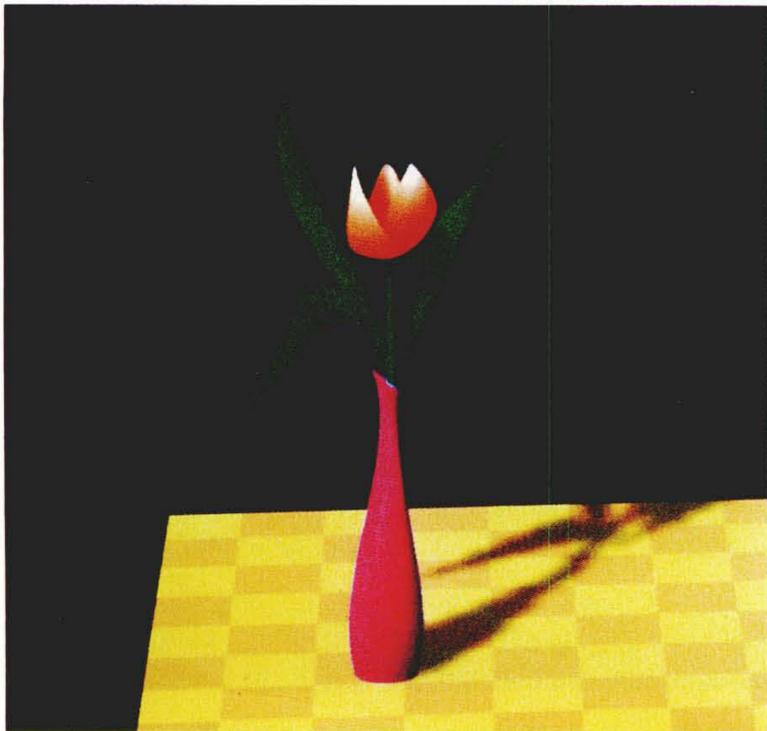


図9 花 レイトレーシングによる自由曲面の表示例を示す。



図10 ティーポット レイトレーシングによる自由曲面の表示例を示す。

る。ここで分子数は512である。これは、二次曲面の表示例であり、ベクトル処理化によってスカラー処理の9.6倍の高速化が実現されている。図6は、木や草のプロシジャモデル、雲の密度分布モデル、山を表すフラクタルモデルの表示を行ったものである。

図7は分子モデルを、図8は大理石の球をレイトレーシングアルゴリズムを用いて表示したものである。ベクトル化によってスカラー処理の約6倍の高速化が実現されている。

図9～10は、レイトレーシングアルゴリズムによる自由曲面の表示例である。画像生成処理時間はそれぞれ66秒、277秒(スカラー処理)である。

## 5 結 言

高品質な画像を高速に生成するために、スーパーコンピュータS-810を用いた高速画像生成システムのプロトタイプATRASを開発した。このシステムは、Zバッファアルゴリズムを用いた高速画像生成機能、レイトレーシングアルゴリズムを用いた高品質画像生成機能を持っている。スーパーコンピュータに適した画像生成アルゴリズムを開発し、ベクトル処理化によってスカラー処理の5倍～10倍の高速化が得られた。今後の課題としては、(1)並列計算機を用いたよりいっそうの高速化、(2)三次元物体の効率的なモデリング技術の開発、(3)複雑な運動の効率的なモデリング技術の開発などが挙げられる。今後、これらの問題を解決していきたいと考える。

## 参考文献

- 1) Foley J.D. et al. : Fundamentals of Interactive Computer Graphics, Addison Wesley, Reading, Mass(1982)
- 2) Kajiya J.T. et al. : Ray Tracing Volume Densities, Computer Graphics, Vol.18, No.3, pp.165~173(1984)
- 3) 栗原, 外 : 密度分布モデルの表示方法, 情報処理学会第34回全国大会 7E-5(1986)
- 4) A.Fournier, et al. : Computer Rendering of Stochastic Models, Communications of the ACM, Vol.25, No.6, pp.371~384(1982)
- 5) Reeves W.T. : Particle Systems-a technique for modeling a class of fuzzy objects, Computer Graphics, Vol.17, No.3, pp.359~376(1983)
- 6) 上西, 外 : 広域曲面補間法, 情報処理学会論文誌, Vol.27, No.4, pp.401~419(1986)
- 7) 大山, 外 : 科学技術計算出力結果表示システムの開発, 情報処理学会数値解析研究会資料16-2(1986.3)
- 8) Thalmann N.M. et al. : Computer Animation, Springer-Verlag(1985)
- 9) Riganati J.P. et al. : Supercomputing, IEEE Computer, Vol.17, No.10, pp.97~113(1984)
- 10) Whitted T. : An Improved Illumination Model for shaded Display, Communications of the ACM, Vol.23, No.6, pp.343~349(1980)
- 11) Plukett D.J. et al. : The Vectorization of Ray Tracing Algorithm for Improved Execution Speed, IEEE Computer Graphics and Applications, Vol.5, No.8, pp.52~64(1985)