グラフィクスとCAD 57－6
（1992．7．24）

マルチ・パス・レンダリング法を用いた高速画像生成手法

川端 敦　渡辺 範人　坂井 俊雄
(株)日立製作所日立研究所

グラフィック・ワークステーションが装備しているハードウェア画像生成装置を用いて、高品位な画像生成を実現するマルチ・パス・レンダリング法に関して報告する。マルチ・パス・レンダリング法は、環境光、拡散反射光、鏡面反射光成分などの各画像成分を別々に画像生成し、生成された各画像成分を加算し最終結果を合成する手法である。マッピング処理なども各成分毎に別々に計算することになる。各画像成分はグラフィック・ワークステーションのハードウェア画像生成装置を用いて合成する。さらに、写り込み成分や屈折成分をレイ・トレイシング法で計算し足し込むことにより、フォト・リアリスティックな映像を生成する。レイ・トレイシング法では、陰面消去アルゴリズムにZバッファ法を用いることにより、1次レイの交点計算演算時間を低減している。

The Multi-Pass Rendering Algorithm for Fast Generating 3D Images.

Atsushi Kawabata　Norito Watanabe　Toshio Sakai
Hitachi Research Laboratory, Hitachi Ltd.

A new multi-pass rendering technique is developed. The image is generated by the conventional hardware multi-pass method which processes shadowing, anti-aliasing and some kinds of image mapping. Each element of shading calculation is rendered one-by-one, that is, the ambient component, the diffuse reflection component and the specular reflection component are rendered one-by-one. And then, the inter-object reflection is calculated by the ray tracing algorithm with the z-buffer method. This specular reflective image is added to the first image to produce the final image. The z-buffer method reduces the calculation time needed to cast the first ray to the nearest object from the camera.

1. まえがき

近年、機械設計の分野でコンピュータ・グラフィックスを用いることが、真剣に検討され始めている。例えば、列車や家電品のデザインを設計段階で検討したり、部品の動きをアニメーションにして確認したり、あるいは顧客に対して製品を説明する場合に用いられる。製品を設計する場合、形状のデザインは機械系CADシステムを用いて設計できるが、色、材質感、模様、ラベル等のデザインはコンピュータ・グラフィックスを用いて検討することになる。デザイン・ワークにコンピュータ・グラフィックスを用いた場合には、正確に形状のイメージを合成することが可能となり、ハイライト・ラインや写り込み、影の効果などを確認することが可能となる。また、コンピュータ・グラフィックスには製品の最終形態ををより具体的なイメージで提示する能力があり、将来はプレゼンテーションの道具としては必要不可欠な物になると予想される。前者の場合、つまり、デザイン・ワークにコンピュータ・グラフィックスを使用する場合には、デザイナと計算機とのやり取りが頻繁に発生するから、デザイナの創造的思考を阻害しないよう、十分に高速な画像生成が要求される。高速であっても、デザインに必須なハイライト表現、写り込み、影の効果などが当然要求される。後者の場合、完成されたデザインを用いて、計算機が単独で画像生成を実行するので、コストの問題はあるものの、高速性はさほど重要ではない。尤も、プレゼンテーションの現場で画像合成を行なう場合には、やはり高速性が要求される。

一方、コンピュータ・グラフィックスのアルゴリズムで代表的なものとしては、ラジオシティ法、レイ・トレイシング法、スキャンライン法、Zバッファ法等が提案されている。レイ・トレイシング法やラジオシティ法は、近年研究が進み、リアルな画像を生成することが可能になってきた。これらのアルゴリズムによって生成されたリアルな映像は、カタログを飾ったりテレビのコマーシャル・フィルムに用いられたりする。スキャンライン法は、上で述べたレイ・トレイシング法やラジオシティ法に比べると描画速度の点では、かなり有利であり、アニメーションを作成する場合等に用いられる。以上で述べたアルゴリズムはソフトウェアで実現され、処理に多くの時間を要するので、コストの高い映像となってしまう。例えば、レイ・トレイシング法の場合では、実用的な映像を生成するのに、最新のワークステーションを使用しても数時間を要してしまい、最も高速に動作するスキャンライン法でも映像生成に数分から数十分を要する。このため、試行錯誤を繰り返すデザイン・ワーク自体にこれらのアルゴリズムを用いることはできず、デザインが完成した後の確認用映像を作成するため、あるいはプレゼンテーション用映像を作成するために用いられているのが現状である。

Zバッファ法を装備したハードウェア・レンダラを用いて高速に画像を作成する手法も、考えられる。この場合、ハードウェアの高速性を生かすことが可能となるため、デザイナは作業を中断せずに、デザイン・ワークに励むことができる。画質の点においては、最新のグラフィックス・ワークステーションは、イメージ・マッピング[1]、マルチ・パス・アンチエイリアス[2]、ステンシル・バッファー等の機能を装備しており、生成された画像の品質に関してはかなり向上してきた。しかしながら、デザイン・ワークに使用する場合、まだ問題が残っている。例えば、影付けとか、属性マッピングを含む特殊なマッピング法等を実現することが困難であったり、同時に使用できる光源の総数も使用するハードウェアの種類に依存したものとなってしまう。そのため、例えば多数の照明に照らされている建造物の映像を作成することは困難となる。

マルチ・パス・レンダリング法は、グラフィックス・ワークステーションのハードウェア・レンダラを繰り返し用いることによって、影付け、属性マッピング等を実現する手法であり、スキャンライン・レンダラと同等の画像品質を、ハードウェアの高速性を生かして実現することを目標としている。さらに、写り込み成分、あるいは屈折成分のみをレイ・トレイシング法で計算することにより、フォト・リアリスティックな映像を高速に生成する。

本報告では、最初にマルチ・パス・レンダリング法の基本的な考え方を説明し、その後で、陰面消去にZバッファ法を用いたレイ・トレイシング法の実現方法、アンチ・エイリアスの実現方法等を順を追って説明する。

2. ハードウェア・レンダラの問題点

先にも述べたように、最新のグラフィックス・ワークステーションは多くの機能を実現しており、生成される画像の品質も大幅に改善されている。例えば、イメージ・マッピング機能を用いれば、テクスチャ・マッピングや、リフレクション・マッピング、リフラクション・マッピングを実現することも可能となる。さらに、画像どうしを画素毎に加算し平均化する機能を有するアキュームレーション・バッファを用いると、マルチ・パス・アンチ・エイリアシングや様々な要因によるボケの表現を実現すること

(a)ハードウェア・レンダラの出力（テクスチャ・マップ無し）　(b)ハードウェア・レンダラの出力（テクスチャ・マップ有り）　(c)マルチ・パス・レンダラの出力

図1　ハードウェア・レンダラのサンプル出力

もできる。

この様に機能が充実してきたが、生成された画像は、今ひとつリアリティーにかける場合が多い。図1にテクスチャ・マッピング処理を施さない場合、施した場合のハードウェア・レンダラの出力例とマルチ・パス・レンダラの出力例を示す。テクスチャ・マッピング機能を用いると、材質感を伝えることが可能となり、表現力は向上する。しかしこの出力例の場合、テクスチャ・マッピングを施したハードウェア・レンダラによる出力が艶消しとなってしまい、鏡面反射光成分の表現に問題が残る。これは、図2に示すように、環境光パラメータ、拡散反射光パラメータ、鏡面反射光パラメータ等のパラメータに基づいて、ハードウェアが物体表面の各点における輝度値を計算した後に、マッピング用のイメージと各輝度値を掛け合わせ、最終映像としているためである。つまり、環境光成分、拡散反射光成分、鏡面反射光成分が、全てマッピング用イメージによって変調を受けるために、艶の表現に問題が生ずる。例えば、赤いリンゴを表現するために赤いマッピング・イメージを使用すると、艶の色まで赤く染まってしまい、一見、艶消し表現の様に感じられるからである。さらに、同時に複数個のイメージ・マッピングを処理することができないため、ある物体にテクスチャ・マッピングを使用した場合には、その物体に対して同時にリフレクション・マッピングを施すことはできない。

マルチ・パス・レンダリング法は、ここで述べたハードウェア・レンダラの欠点を補うための手法であり、グラフィックス・ワークステーションのハードウェア資源を組み合わせて使用することにより実現する。ここで、マルチ・パス・レンダリング法が使用するハードウェア資源を図3で説明する。図3は、通常のグラフィックス・ワークステーションで用いられているZバッファ法により陰面消去を行なう、ハードウェア・レンダラの基本構成である。Zバッファ制御装置は、入力された形状の幾何情報とZバッファに蓄えられている奥行き値とを比較し、各画素毎にフレーム・バッファへの書き込みを制御する。この動作のことを「Zバッファ比較」と呼ぶことにする。またZバッファ制御装置は、Zバッファの各画素が最も視点に近い形状までの距離を保持するようZバッファを常に更新する。この動作のことを「Zバッファ更新」と呼ぶことにする。Zバッファ比較、Zバッファ更新とも別々に動作させたり、止めたりすることが可能である。αブレンディング機構は、すでに描画済みのフレーム・バッファの値と、新たに計算された画素値とを混ぜ合わせる機構であり、半透明表現に使用する。アキュームレーション・バッファはフレーム・バッファの内容を画素毎に累積するメモリーである。輝度計算機構に入力される霧パラメータは、視点からの距離に基づいて、計算される画素値を霧の色に近付けるパラメータである。以上のハードウェア資源を用いてマルチ・パス・レンダリング法の実現方法を説明する。

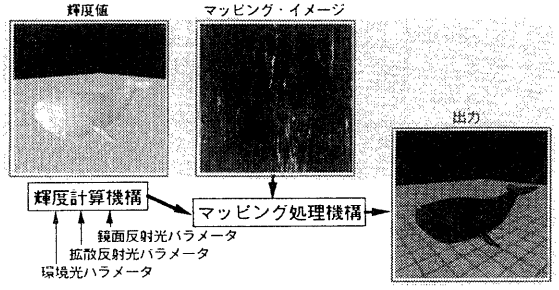

図2　ハードウェア・レンダラの画像生成機構

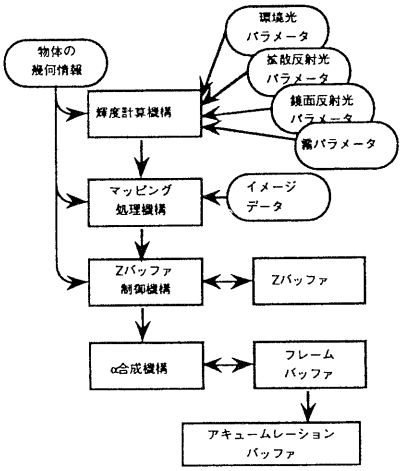

図3 グラフィックス・ワークステーションのハードウェア資源

3. マルチ・パス・レンダリング法の実現方法

まず、ハードウェア・レンダラの輝度計算式を式1に示す。

$$I(x, y) = \Sigma_i (Ia_i \cdot Ka + Id_i \cdot (Kd \cdot \cos\theta + Ks \cdot (\cos\alpha)^n) \cdot Map(s, t)) \quad (1)$$

太字は(r, g, b)を成分として持つベクトル量を示し、"・"はベクトルの成分毎の掛け算、あるいはベクトルの各成分にスカラを掛けることを意味する。I(x, y)は、最終画像の(x,y)に於ける画素値を示す。Ia_i、Id_iは光源を特徴付ける量、Ka、Kd、Ksは物体の表面属性を特徴付ける量、θは(x, y)におけるLambertの角度、αはPhongの角度を示し、nは面荒さを示す。添え字のiは光源番号を示す。ハードウェアの場合は、式1が多角形の各頂点に関して計算された後に内部が滑らかに補間される。Map(s, t)は、形状の各点に対応しているマッピング・イメージの画素値を出力する関数である。

マルチ・パス・レンダリング法は、環境光、拡散反射光、鏡面反射光成分などの各画像成分毎に別々に画像生成を実行し、各画像成分を加算し最終結果を合成する手法である。式2に示すようにマッピング処理なども各成分毎に別々に計算することになる。

$$\begin{aligned} Ia_i(x, y) &= Ia_i \cdot Ka \cdot Mapa(s, t) \\ Id_i(x, y) &= Id_i \cdot Kd \cdot \cos\theta \cdot Mapd(s, t) \\ Is_i(x, y) &= Is_i \cdot Ks \cdot (\cos\alpha)^n \cdot Maps(s, t) \\ Ie(x, y) &= Ks \cdot Mape(s, t) \\ I(x, y) &= \Sigma_i (Ia_i(x, y) + Id_i(x, y) + Is_i(x, y)) + Ie(x, y) \end{aligned} \quad (2)$$

各成分毎に画像生成を実行すると、各成分画像として高い品質のものを得られるので、加算して得られる最終画像の品質を保つことが可能となる。つまり、式2は式3に示すスキャンライン・レンダラの輝度計算式と類似しているため、ハードウェアを用いても高い品質の成分画像を生成することができる。(ただし、スキャンライン・レンダラの輝度計算は、補間演算をせずに画像の各画素毎に行なわれる事が多い)

$$I(x, y) = \Sigma_i (Ia_i \cdot Ka \cdot Mapa(s, t) + Id_i \cdot (Kd \cdot \cos\theta \cdot Mapd(s, t) + Ks \cdot (\cos\alpha)^n \cdot Maps(s, t))) + Ks \cdot Mape(s, t)) \quad (3)$$

図3のハードウェアを用いて式2の各画像成分、例えば、環境光成分の画像を合成する場合には、ハードウェア画像生成装置に環境光強度と、環境光用のテクスチャ・マップ・イメージを設定することにより、実現できる。ここで図3に示すハードウェアに設定するパラメータの組を次のように表現する。

(Ka = 物体属性、Kd = 0、Ks = 0、Ke = 0、
Map = 環境光用マップ・イメージ)

上で示すようなパラメータを設定して形状の幾何データを送り込むことによって描画が実行される。Keは光源に依存しない、自己発光成分を示すパラメータである。このような演算を各光源毎に別々に行う。拡散反射光、鏡面反射光、リフレクション・マッピング成分を生成する場合のパラメータの組は次のようになる。

(Ka = 0、Kd = 物体属性、Ks = 0、Ke = 0、
Map = 拡散反射光用マップ・イメージ)
(Ka = 0、Kd = 0、Ks = 物体属性、Ke = 0、
Map = 鏡面反射光用マップ・イメージ)
(Ka = 0、Kd = 0、Ks = 0、Ke = 1、
Map = リフレクション・マップ・イメージ)

(環境光成分画像)、(拡散反射光成分画像)、(鏡面反射光成分画像)は光源に依存する成分であるので、個々の光源に関して個別に画像生成し、足し込む。この演算は必要なだけ繰り返すことができるので、事実上光源の数に制限がなくなる。(環境マッピング成分画像)は、光源に依存しない成分であり一回だけ演算して、足し込む。以下では、幾つかの表現手法に関して、具体的に実現方法を述べる。

3.1 多重マッピング

周囲の光景が写り込んでいる大理石等を表現する場合には、テクスチャ・マッピングとレフレクション・マッピングを同時に用いる。通常、テクスチャ・マッピングの場合には、(環境光成分)と(拡散反射光成分)に、同じマッピング・イメージを用いて実行する場合が多い。しかし精密に大理石などを表現する場合には、大理石の微小なパターンの反射特性の違いを考慮した異なったマッピング・イメージを用いて、(環境光成分)、(拡散反射光成分)、(鏡面反射光成分)を計算する場合もある。式2を用いると、このような属性マッピングも実現できるし、さらに繰り返し演算することによって、例えば環境光成分として複数のイメージ・マッピングを実現することも可能である。

3.2 影

影の領域は環境光成分のみ、あるいは場合によっては環境光成分とリフレクション・マッピング成分だけで描画され、拡散反射光成分、鏡面反射光成分

は描画が抑制される。影領域の検出は、シャドウ・マッピング法[3]やシャドウ・ポリゴン法[4][5][6]によって実現できる。シャドウ・マッピング法は、物体の表面が、光源から見た場合直視できるか否かを各画素毎に調べる手法であり、光源から見た場合のZバッファ値を用いて計算する手法である。シャドウ・ポリゴン法は物体が落とす影を立体にみたて、表示物体との干渉を調べる手法であり、最終的には立体の集合演算に帰着する。最新のグラフィック・ワークステーションの中には、集合演算用のステンシル・バッファを備えている機種も存在するが、この機能を使用すれば影領域を高速に算出できる。式4に影領域を考慮した場合のマルチ・パス・レンダリング法の輝度計算式を示す。

$$
\begin{aligned}
&Ia_i(x, y) = Ia_i \cdot Ka \cdot Mapa(s, t) \\
&Id_i(x, y) = shadow_i(x, y) \cdot Id_i \cdot Kd \cdot \cos\theta \cdot Mapd(s, t) \\
&Is_i(x, y) = shadow_i(x, y) \cdot Is_i \cdot Ks \cdot (\cos\alpha)^n \cdot Maps(s, t) \\
&Ie(x, y) = Ks \cdot Mape(s, t) \qquad (4) \\
&I(x, y) = \Sigma_i \, (Ia_i(x, y) + Id_i(x, y) + Is_i(x, y)) + Ie(x, y)
\end{aligned}
$$

$shadow_i(x, y)$は光源 iに関する影領域を計算する関数であり、画面の該当する領域が影領域の場合は0を、そうでない場合には、1を返す関数である。図4に多重マッピングと影の効果を入れた、環境光成分画像、拡散反射光成分画像、鏡面反射光成分画像、リフレクション・マッピング成分画像、最終画像を示す。

3.3 光源

式4にも示したように、光源に依存する成分を各光源毎に演算することにより、光源の数に制限を受けない画像生成を実現する。さらに、適当な範囲で光源を移動させた画像を重ね合わせることにより、ソフト・シャドウも可能となる。自己発光体を表現するときには、発光体の形状を適切な範囲で移動させ重ね合わせれば、グレアが表示できる。

ある部品だけを選択的に照らす照明は、見栄えのする映像を作成する場合しばしば用いることがある。これは、Zバッファの制御と組み合わせて用いる事により実現できる。選択的照明を処理する場合には、その照明によって照らされる形状のみを描画して最終映像に足し込む。この場合、描画候補以外の形状によって、覆い隠されている部分の描画を抑制するために、全ての描画に先だって、全ての形状に関する奥行き値をZバッファに作成する必要がある。作成されたZバッファの内容を参照しながら描画を行なえば、実際に見える部分のみの描画を実行でき、正し

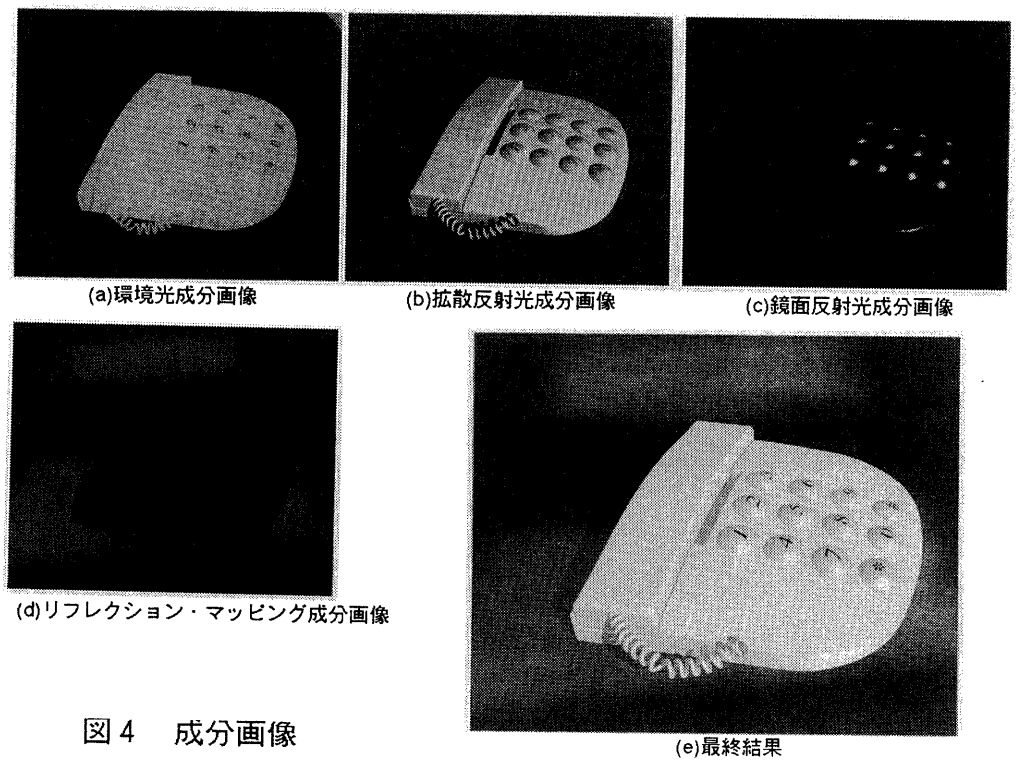

図4　成分画像

い、結果が得られる。

3.4 半透明表現

半透明物体を正確に表現するためには、レイ・トレイシング法等を用いる必要があるが、屈折のない半透明表現であれば、近似表現が可能となる。近似表現としては、ソーティング・バッファ[7]を用いた精度の高いものが提案されているが、本論文では簡略化した手法に関して述べる。半透明表現とは、物体を通して別の物体が見える表現であり、ハードウェアのαブレンディング機構を用いて実現できる。まず最初に全ての不透明物体を描画した後に、αブレンディング機構を作用させながら半透明物体を描画する。この場合、視点からの距離が遠い半透明物体から順に描画する必要がある。この条件が満足できない場合には、不透明物体を描画し終わった後にZバッファ更新を禁止して、適当な順序で半透明物体を描画する。ただしこの場合、半透明物体が重なっている部分に関しては描画が乱れる。4項で述べる写り込み、屈折成分を加算する場合には、半透明物体も不透明物体として描画し、透過成分はレイ・トレイシング法により、計算する。

3.5 霧

霧の表現は、物体の色と霧の色とを混ぜ合わせることにより実現する。混ぜ合わせる比率を距離によって変えることにより、遠くなるほどかすむ。マルチ・パス・レンダリングを用いて霧の表現を実現する場合、各画像成分に霧をかけると、影の部分に問題が生ずる。先にも述べたが、影は拡散反射光成分と鏡面反射光成分を描画する際に影領域の描画を抑制することによって表現している。そのため霧の成分も描画が抑制されてしまい、最終画像は、影の部分が黒く抜けた映像となってしまう。マルチ・パス・レンダリングで霧を表現するには、次に示す処理手順で画像を生成する。まず最初に、霧の色を黒として、通常のマルチ・パス・レンダリングの手順で画像生成を行なう。ここで得られた画像は、視点からの距離が大きくなるに従って、黒くなる映像が得られる。次に、霧の色を所定の値に設定し、影領域の描画を抑制せずに、半透明物体も含めた全ての物体を、黒い属性を設定して描画する。この場合、霧の成分画像が正確に得られるので、先に求まっている画像に足し込むことによって最終画像が得られる。

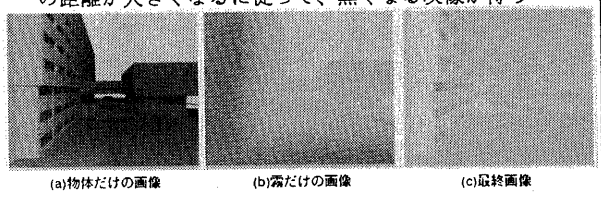

図5　霧の効果

4. 写り込み、透過成分の計算方法

レイ・トレイシング法による映像生成を行なう場合には、ハードウェア・レンダラによるマルチ・パス・レンダリングを実行した後に、写り込み成分、透過成分のみを足し込む。この場合の画像生成式を式5に示す。

$$
\begin{aligned}
&Ia_i(x, y) = Ia_i \cdot \mathbb{K}a \cdot Mapa(s, t) \\
&Id_i(x, y) = Id_i \cdot \mathbb{K}d \cdot \cos\theta \cdot Mapd(s, t) \\
&Is_i(x, y) = Is_i \cdot \mathbb{K}s \cdot (\cos\alpha)^n \cdot Maps(s, t) \\
&Ie(x, y) = \mathbb{K}s \cdot Mape(s, t) \\
&I(x, y) = \Sigma_i \,(Ia_i(x, y) + Id_i(x, y) + Is_i(x, y)) + \\
&\qquad\qquad Ie(x, y) + \underline{IrayTracing(x, y)}
\end{aligned}
\tag{5}
$$

式5の下線部分、つまりレイ・トレイシング成分が新たに加わった部分である。IrayTracing(x, y)はアイテム・バッファを使用した局所レイ・トレイシング法を用いて計算する。この手法は、1次レイの交点計算を加速する手法であり、基本的なアイデアはWaghorstによって提案されている[8]。アイテム・バッファはフレーム・バッファの各ピクセルに1対1に対応したエントリを持ち、該当するピクセルを覆う可能性のある形状を記憶する機構である。1個のピクセルを覆う可能性の有る形状は一般には複数存在するので、アイテム・バッファの各エントリは複数の形状を記憶できるようなリスト構造を要素として持つ

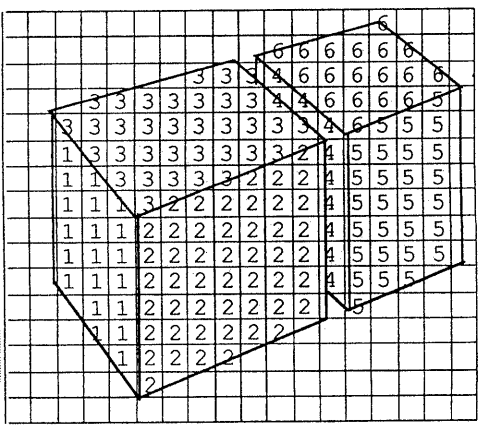

図6　アイテム・バッファが保持する値

ことになる。この手法では、アイテム・バッファの制御が複雑になるため、ハードウェアを使用して演算を加速することは困難である。本報告で使用するレイ・トレイシング・アルゴリズムは、各ピクセルに対応してただ1個の形状のみを記憶できる単純なアイテム・バッファとZバッファを組み合わせて使用することにより、1次レイの交点計算を加速する。図6に示すようにアイテム・バッファは表示されるポリゴンの識別子を記憶する。次に処理手順を説明する。

1)全ての表示形状をポリゴンに分割し、個々のポリゴンに識別子を与える。(このプロセスは$\mathbb{I}a_i(x, y),\cdots$を計算するときに実施済みである。)

2)ハードウェアのZバッファ法を用いて描画を行なう。この時、フレーム・バッファに書き込む内容は、シェーディングされた結果ではなく、各ポリゴンの識別子とする。フレーム・バッファが保持している内容は、該当するピクセルを覆うポリゴンの中で、最も視点に近いポリゴンの識別子である。

3)各画素毎に、登録されているポリゴンとの交点計算を行なう。

この手法は、2つの点で有利である。まず最初に、実際に交点追跡演算が行なわれるのは第2レイ以降であるという点である。反射光を計算する必要のない物体の場合は、一切交点追跡を実行する必要がなくなる。次に有利な点は、影演算は基本的には3項以前で述べた手法によりハードウェアを用いて行なうので、影演算用の交点追跡演算をする必要が無い点である。この場合、写り込んでいる部分に関してのみレイ・トレイシング法で通常用いている影演算を実行する必要がある。これらの利点により、交点追跡の回数は大幅に減少する。レイ・トレイシング法の場合、演算時間の大半が交点追跡に費やされているため、交点追跡の回数が減少することは、直接、演算時間短縮につながる。

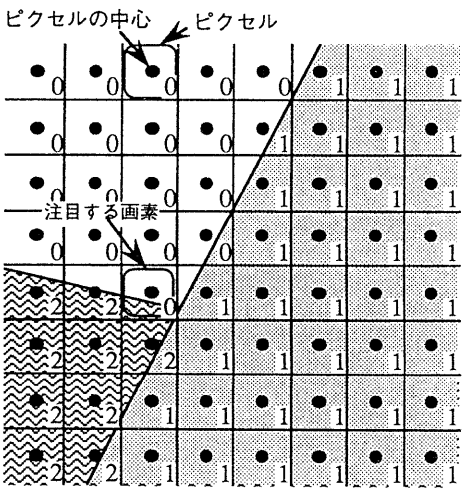

図7　アイテム・バッファのサンプリング法

次にこの手法を用いて、アンチ・エイリアスを施す手法に関して報告する。Waghorstの方法では、アイテム・バッファがあるピクセルに関連する全てのポリゴンに関する情報を保持しているため、アンチ・エイリアスを容易に実現できる。本報告で述べる手法はアイテム・バッファが各ピクセルに関してただ1個のポリゴン情報だけを保持しているため、アンチ・エイリアスを実行する場合には、注意が必要となる。まず最初に、アイテム・バッファの内容を参照しながら、適応的にサンプリング・レートを上げる手法が考えられが、この場合、小さなポリゴンに関するアンチ・エイリアスが問題となる。つまり、アイテム・バッファは、ピクセルの精度で情報を保持しているため、ピクセルよりも小さなポリゴンや、針状のポリゴンに関する情報を正しく内容に反映することができない。秋元らは、適応サンプリング型レイ・トレイシング法において、針状のポリゴンの描画を改善する手法に関して報告しているが[9]、微小ポリゴンに関する問題点は以前として存在している。

本報告では、単純にマルチ・パス・サンプリングを実行してアイテム・バッファを作成し、アンチ・エイリアスを実行する。以下で、簡単に原理を説明する。ハードウェア・レンダラは画像を生成する際に、図7に示すようにピクセルの中心でサンプリングし、色を決定する。従って、ハードウェア・レンダラを使用して、アイテム・バッファを作成した場合には、ピクセルの中心に於ける視点に最も近いポリゴンの識別子が保持される。ピクセルの中心位置を通過するレイはアイテム．バッファに登録されている形状と必ず交点を持つことになる。ここで、表示位置を1ピクセル以下の巾で上下左右に動かし、複数枚のアイテム・バッファを作成すると、図8に示すように特定のピクセルに対して様々なポリゴンが登録される。図8は図7で着目しているピクセルとその周辺に関するアイテム・バッファの値を示している。

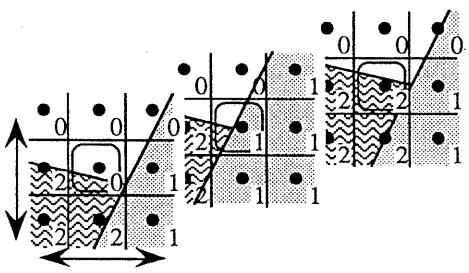
図8　表示位置移動による効果

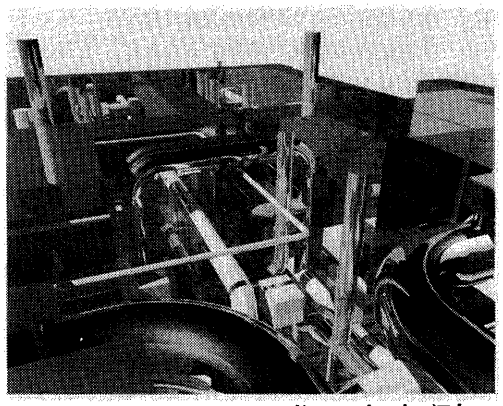

図9　写りこみ成分出力例

各アイテム・バッファに登録されているポリゴンと、そのときの表示位置におけるピクセルの中心位置を通るレイは必ず交点を持つ。各々のアイテム・バッファに関して反射、写り込み成分を計算し、ピクセル毎に単純平均、あるいは荷重平均を計算すれば、微小ポリゴンも確立的にサンプリングされ正しいアンチ・エイリアスを計算できる。表示位置を1ピクセル以下の巾で上下左右に動かすことは、ハードウェア・レンダラのサブ・ピクセル・ポジショニング機構を利用すると可能とる。

本方式は、陰面消去にハードウェア・レンダラを使用しているので、従来に比べて3倍から5倍の高速化が実現できた。例えば、図9の場合には(艶なしポリゴン-444、艶ありポリゴン-77,184)、IRIS 4D/340VGXのCPUを1個使用して、640X480、アンチ・エイリアスなし、第2レイまで追跡して、8分34秒で最終結果が得られた。

5. 結論

グラフィックス・ワークステーションの基本的なハードウェアを繰り返し使用することによって、ソフトウェアに匹敵する画像を高速に生成する手法を示した。画像生成は、十分に高速であるため初期の目的である、「デザイナーが試行錯誤的に使用するコンピュータ・グラフィックス・ツールの実現」を十分に満足するものが得られた。さらに、ハードウェア・レンダラを陰面消去に用いたアイテム・バッファ法を用いることにより、写り込み成分を高速に生成する手法を示した。この手法を用いて従来のレイ・トレイシング法に比べて3倍から5倍の高速化を実現できた。

6. 参考文献

[1] Kirk, D. and Voorhies, D. : The Rendering Architecture of the DN1000VS, Computer Graphics, Vol.24, No.2, pp.299-307(1990)

[2] Haeberli, P. and Akeley, K. : The Accumulation Buffer: Hardware Support for Hight-Quality Rendering, Computer Graphics, Vol.24, No.4, pp.309-318(1990)

[3] Williams, L. : Casting Curved Shadows on Curved Surfaces, Computer Graphics, Vol12, No.3, pp270-274(1978)

[4] Crow, F. C. : Shadow Algorithms for Computer Graphics, Computer Graphics, Vol.11, No.2, pp.799-805(1977)

[5] Bergeron, P. : A General Version of Crow's Shadow Volumes, CG & A, Vol.6, No.9, pp.17-28(1986)

[6] Brotman, L. S. and Badler, N. I. : Generation Soft Shadows with a Depth Buffer Algorithm, CG & A, Vol.4, No.10, pp.5-12(1984)

[7] Mammen, A. : Transparency and Antialiasing Algorithms Implemented with the Virtual Pixel Maps Technique, CG & A, Vol.9, No.4, pp.43-55(1989)

[8] Weghorst, H., Hooper, G. and Greenberg, D. P. : Improved Computational Methods for Ray Tracing, ACM TOG, Vol.3, No.1, January 1984, pp.52-69.

[9] Akimoto, Taka-aki et.al. : Pixel Selected Ray Tracing, EUROGRAPHICS'89, 1989, pp.39-50.