ハードウェア向きの高速影生成アルゴリズム

3T-8

松本　尚

日本アイ・ビー・エム株式会社　東京基礎研究所

1. はじめに

近年，CAD のプリ・ビューやアニメーションやプレゼンテーション・グラフィックスなどで三次元物体の高速な描画の必要性が高まってきている。そこで，最近の多くのエンジニアリング・ワークステーションは三次元のグラフィックス表示の高速性を競い，特徴としている。これらの三次元高速描画機能はハードウェア化され，高速な物では何段ものパイプライン処理やパラレル処理を実行している。しかし，この高速ハードウェアはスムーズ・シェーディングと陰面消去ぐらいしかサポートしておらず，それ以上の機能を求めた途端に描画の高速性が激減する。そこで，現在あまりハードウェアでサポートされていない描画機能について，どういう手法で実現すればよいか考察を行った。その中の影の生成法について本稿で論じる。

具体的な手法について述べる前に，どのレベルの品位のグラッフィクスを狙っているのか，また，どのような方向でのハードウェア化を考えているのかを明らかにしておく。インターラクティブないしはリアルタイムの表示性能を持つグラフィックス・システム（ただし，規模はワークステーション程度）を念頭においている。品位としてはポリゴン・ベースのグラフィックス，つまり，現状のエンジニアリング・ワークステションの高速描画に用いられているものを仮定している。そして，影を付けたり，任意領域だけ切り出すといったことも高速に実行できるというように高速描画の高機能化の方向を目指す。ハードウェア化に関してはラスタ化時のピクセル毎の処理の並列性を利用し，パイプラインまたはパラレル処理を行うものを念頭においている。典型的には ISCC88 で発表された松下電気のグラフィックス・チップ[1]や IBM で研究開発中の SAGE [2]といった1 ライン分のピクセル・プロセッサを持ったチップでハードウェア化するのが望ましいと考えている。これらのチップのピクセル・プロセッサになるべく少ないハードウェアの追加を行うことにより他の機能もサポートすることができるようにもっていきたいと考えている。従来のアルゴリズムではあまりこの目的に適していないので，新しい手法を摸索した。

2. 基本アイデア

今，簡単のためポリゴンはすべて不透明とし，光源は単一点光源とする。ハードウェア化の簡単さにより陰面消去法としては，スキャンライン・Zバッファ法を仮定する。高速性とスキャンライン法と両立できるという条件よりシャドウ・ポリゴンの方法[3]を基本とする。あるスキャンライン上で，それに掛かるポリゴンを以下の順序で展開する。

1. 通常の物体を構成するポリゴンをすべて展開する。
2. シャドウ・ポリゴンをシャドウ・ボリューム毎に展開する。

後者の展開の際はZバッファや輝度バッファの内容は変更せず，次章で述べる手続きに従って視線とシャドウ・ボリュームの交点の数の偶奇性（図1 ）を調べるフラグを操作して，影の領域の決定を行う。すべてのポリゴンの展開の後，ピクセルごとに影でないときの輝度値と影内かどうかが判るので，これを用いて輝度を適当に修正してフレーム・バッファに格納する。

3. スパンデータ

まず，スパンデータと表記法を説明する。スキャンライン単位で処理を行うので，各ポリゴンのあるスキャンライン上のデータ（スパンデータ）として以下のものを用いることとする。

(attr, XL, XR, ZL, dZ, IL, dI)

XL, XR はスパンの両端の x 座標を表わし，今は便宜上それぞれ左端，右端とする。ZL, ILはそれぞれ XL におけるデプス値，輝度値を表わす。dZ, dI は XL 側から XR 側へスキャンライン上で，1 ピクセルずれたときのデプス値，輝度値の変化量（差分: 線形補完なので定数）を表わす。よって，スパンデータの展開は XL から XR に向かって，デプス値と輝度値にそれぞれ dZ, dI をピクセル毎に加算することによって行われる。

シャドウ・ポリゴンと普通のポリゴンを区別したり，影付けのアルゴリズムのための若干の情報を付加したりするために，属性項 attr (attribute) が含まれている。attr の実際に取りうる値（属性）は以下のものである。

OP	シャドウ・ポリゴンでない普通のポリゴンのスパン (Ordinary Polygon)
SP	シャドウ・ポリゴンのスパン (EV, IF でない) (Shadow Polygon)
EV	1 つのシャドウ・ボリューム内の最後のシャドウ・ポリゴンのスパン (End span in a Volume)
IF	シャドウ・ポリゴンのスパンの一種で影判定フラグを反転する 動作を伴う (Invert Flag)

4. シャドウ・ポリゴンの処理

普通のポリゴンのスパンデータに対しては スキャンライン・Zバッファ法に従って普通にピクセルごとのデータに展開する。この後，シャドウ・ポリゴンのスパンデータの処理をする（図2 ）。この処理のアルゴリズムを以下に述べる。各ピクセルに DSF, FSF という2個のフラグを設ける。DSF（影判定フラグ）は視線とシャドウ・ポリゴンが交差する毎に反転し， FSF はピクセルがあるシャドウ・ボリュームの中と確定した時にセットされる。

各ピクセルにおいて

1. シャドウ・ポリゴンの処理を開始するまでに， FSF := 0 ; DSF := 0 ; に初期化しておく。
2. 処理をパスカル風に記述すると，

```
REPEAT  all shadow volumes
  REPEAT
  (*all spandata in a shadow volume*)
    INPUT(spandata);
    IF attr='IF' THEN DSF:=NOT DSF;
    IF (XL<=pixelID<=XR) AND (Z < ZB)
      THEN DSF:=NOT DSF;
  UNTIL attr='EV';
  IF DSF<>0 THEN FSF:=1;
UNTIL end of shadow volumes
```

ただし，ここで pixelID はスキャンライン上の対象となる ピクセルの X座標値である。Z はシャドウ・ポリゴンのピクセル上でのデプス値 (ZL, dZ から計算される) であり。ZB は対象となるピクセルのZバッファの値であり，そのピクセルを占める普通のポリゴン上の点のデプス値である。デプス値のコンベンションは視点に対して手前にある程，値が小さいと仮定している。

A Fast Algorithm of Generating Shadows for Hardware

Takashi Matsumoto

IBM Research, Tokyo Research Laboratory

3. FSF が 1 になっていたら，そのピクセル上の点は影の中なので，そのピクセルの輝度値を修正する。

以上がシャドウ・ポリゴンのスパンデータの基本的な処理である。

5. 注意点

ここで，このアルゴリズムに対する注意点とその対処法について述べる。

- このアルゴリズムでは影判定に視線とシャドウ・ポリゴンの交差の数の偶奇性を用いている。このため，1 つのシャドウ・ボリュームに属するシャドウ・ポリゴンのつなぎめの所が問題になる。図3 の状況で AB, BC に対する (XL,XR) を (X1,X2), (X2,X3) としてスパンデータを作ると，X2 つまり点B で交差をダブル・カウントしてしまう。シャドウ・ポリゴンのスパンデータを作る時には，スパンの二端点を Xl, Xr とすると，シャドウ・ポリゴンは総て (Xl,Xr-1) (または (Xl+1,Xr)) としてスパンの範囲を指定すると不都合がなくなる。
- ソフトウェアで実行する時やスキャンライン上の全てのピクセルに対応する処理機構がない時は，DSF のチェックをすべてのピクセルについて行うのは効率的でない。この場合，EV の属性はやめ，シャドウ・ボリューム毎にチェックのためのダミーのスパンデータを各シャドウ・ボリュームに属するスパンデータの最後に加える。このダミーのスパンデータの XL, XR は，それぞれ，そのスパンデータが属するシャドウ・ボリュームのシャドウ・ポリゴンのスパンデータの XL の内の最小値，XRの内の最大値に取り，その範囲のみチェックする。

6. 拡張

広がりを持つ光源と複数の光源が作る影への拡張法を簡単に述べる。通常よくやられているように，広がりを持つ光源は複数個の同等な点光源で近似して扱う。それぞれの点光源についてシャドウ・ボリュームを生成し，ある点を影にしている点光源の数と影にしていない点光源の数の比で半影の度合を決定する。ピクセル毎にそのピクセルを占める点が幾つの点光源について影になっているかカウントする必要がある。このためスパンデータを展開するスキャンライン上の各ピクセルに対して1 つのカウンタを用意する。

次に複数の点光源への拡張について述べる。複数の光源による影を付けるとなると本質的には考慮している点がどの光源からの影の中にあるか決まるまでは，その点の輝度値を決めようがない。Zバッファ法を採用する限り，影判定が済むのはスパンデータ展開後になってしまう。そこで，スパンデータ展開後の処理をなるべく軽くして，高速性を維持する方法を考える。各光源からの影を無視した時の輝度への寄与を別々に展開しておけば，影判定後，影になっていない光源についてのみ，その寄与を足しあわせて輝度値を求めることができる。しかし，この方法ではピクセル毎に光源数分の輝度値を保持しなくてはならず，1 つの LSI 上に複数のピクセル・プロセッサを集積するようなアプローチのハードウェア化は現状では難しい。ここでは光源と物体がかなり離れているという仮定（1 つの光源からの光の方向は一定という仮定）を採用し，1 つの輝度値の近似修正法を提案する。影なしとして全部の光源の寄与を合算した輝度値とポリゴンの法線ベクタを共にスパンデータの要素とし，スパンデータを展開する。ただし法線ベクタとして，クラス分けして少し粗めに量子化されたものを用いて，そのクラスを示すインデックスとどの光源から影になっているかの情報で輝度値をどの程度修正する（割り引く）かの割合の入ったテーブルを引くことにする。テーブルから得た値とピクセルに展開された輝度値を掛けたものを修正された輝度値とする。影の範囲の判定は前述の手法の自然な拡張で行う。

7. おわりに

Zバッファ法とシャドウ・ポリゴンを組合せた手法としては，Brotman と Badler のアルゴリズム[4]がよく知られている。そこで，最後に彼等の手法との差異について述べる。最大の差は彼等の方法では凸のポリゴンが作ったシャドウ・ボリュームしか取り扱えないことである。本手法は凸でないポリゴンが作るシャドウ・ボリュームも取り扱え，三次元の形状データから大幅にシャドウ・ボリュームとシャドウ・ポリゴンを減らすことも可能である。また，彼等はハードウェア量の少さや高速性より品位にこだわっているようなので，本手法と同列に比較しがたいが，影の領域の判定法のみを見た場合でも，本手法の方が必要とする記憶領域も計算量も少ない。

文献

1. Nishizawa, T., et al.: A Hidden Surface Processor for 3-Dimension Graphics. *proc. 1988 IEEE Int. Solid-State Circuits Conf.* (1988) pp. 166-167.
2. Gharachorloo, N. and Pottle, C.: SUPER BUFFER: A Systolic VLSI Graphics Engine for Real Time Raster Image Generation. *Proc. 1985 Chapel Hill Conf. on VLSI* (1985) pp. 285-305.
3. Crow, F. C.: SHADOW ALGORITHMS FOR GRAPHICS. *Proc. SIGGRAPH'77* (1977) pp. 242-248.
4. Brotman, L. S. and Badler, N. I.: Generating Soft Shadows with a Depth Buffer Algorithm. *IEEE Computer Graphics & Applications* (1984.10) pp. 5-12.

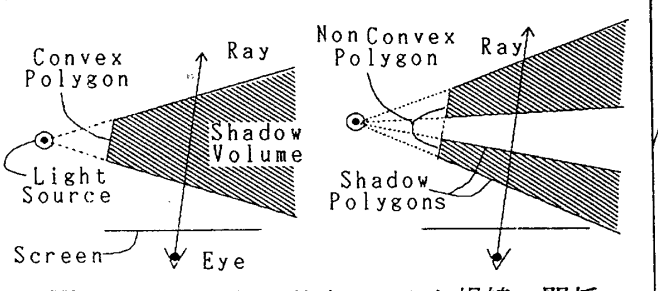

図1　シャドウ・ボリュームと視線の関係

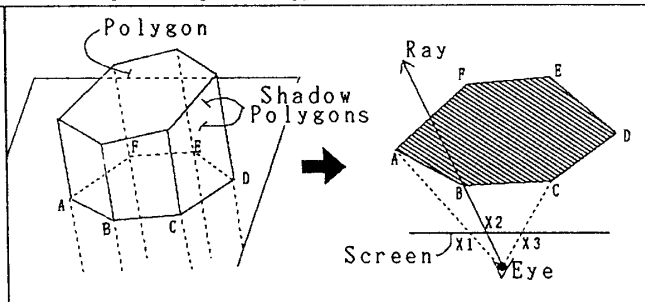

図3　シャドウ・ボリゴンのつなぎ目

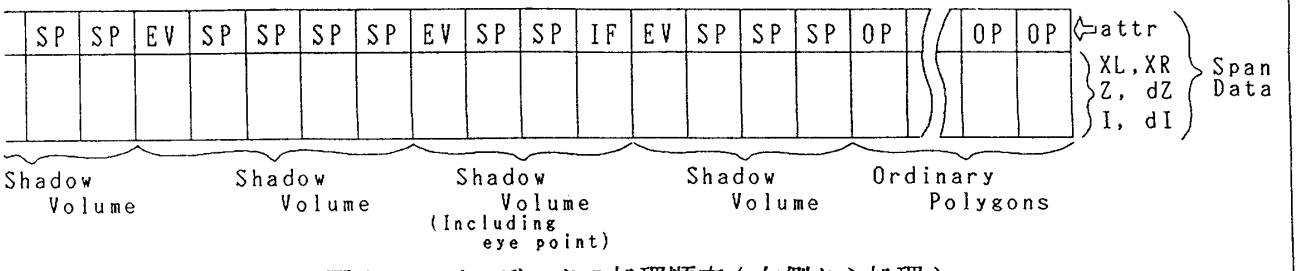

図2　スパンデータの処理順序（右側から処理）