

## 凸な立体の集合演算によって定義された形状の スキャンライン法による陰影画像生成†

床井浩平†† 北橋忠宏†††

CSG データからの陰影画像生成には、視線探索法を用いるのが一般的であるが、この方法は処理に膨大な時間を要するため、形状モデリングの際の形状把握を目的とした陰影画像生成等の用途には適しない。筆者らはこれを会話型システムにおいても利用できるような、スキャンライン法に基づく CSG データの高速な陰影画像生成アルゴリズムを開発した。スキャンライン法は元来境界表現データからの陰影画像生成に用いられる手法であり、形状データの境界に沿って処理が進められるため、境界情報を明示的な形で持っていない CSG データにそのまま適用することは困難である。そこで本研究では、①プリミティブとスキャンライン平面との交差線を求め、これに対してスキャンライン法を適用して、ディスプレイ上の特定の画素を覆い隠すプリミティブの選択を行う、②選択されたプリミティブ間で集合演算処理を行い、さらに奥行き比較を行って可視面を決定する、といった手順を用いた。しかしこのように集合演算処理等を画素単位に行った場合、複雑な形状の表示に時間がかかる。そのため本研究ではファレンスカウント法と呼ぶ集合演算処理の高速化技法を開発した。また直前に処理した画素における可視判定結果の再利用による高速化も行った。これらの高速化技法は、スキャンライン法の特徴を活用したもので、本手法は画素単位に独立した処理を行う視線探索法に比べて非常に高速な処理を可能にしている。

### 1. はじめに

ソリッドモデルの表現形式としては、CSG (Constructive Solid Geometry) 表現と境界表現の二つが代表的である。CSG 表現は単純な形状をした基本立体(これをプリミティブと呼ぶ)を回転・移動することによって空間中で位置決めした上、それらを集合演算を用いて組み合わせ、複雑な立体形状を表現しようとするものである。これは、立体を頂点・稜線・面といった境界情報を用いて表現する境界表現と比べて形状記述がより言語的となるため、人間にとって理解しやすく、形状定義の変更も容易である。また表現しようとする形状と用意されたプリミティブとの整合性が良ければ、形状記述に必要なデータ量も少なくてすむ。

これらは試行錯誤により形状データ作成を行おうとしたときに非常に有効な特徴であるといえる。ところが CSG により定義された形状の陰影画像生成に通常用いられる視線探索法は、処理に膨大な時間がかかり、形状把握のための陰影画像生成を頻繁に行うことができず、形状データ作成の際に CSG のこのような特徴を十分に生かすことができない。

そこで筆者らは、形状の把握を目的とした陰影画像生成には、視線探索法の特徴である鏡面反射や屈折などの光学的処理よりも、陰影画像の生成にかかる処理時間の短縮の方が重要であると考え、CSG で定義された形状の陰影画像生成をスキャンライン法を用いて行おうと試みた<sup>12),13)</sup>。スキャンライン法は元来境界表現データからの陰影画像生成に用いられる手法であり、鏡面反射や屈折などの光学的処理を行うことが難しいが、反面、視線探索法に比べ非常に高速に陰影画像を生成することが可能である。本研究では可視プリミティブの選択や集合演算処理において立体形状の一様性を利用する手法を開発することによって、これを CSG 表現による形状データに応用した。これにより本方式は画素ごとに独立した処理を行う視線探索法に比べ、非常に高速な処理速度を達成できた。本稿ではその具体的な手法について報告する。

### 2. CSG で記述された形状の陰影画像生成

形状データを CSG 表現による形状データの陰影画像生成が、視線探索法以外の方法では困難であると考えられる理由は、視線探索法以外のほとんどの描画アルゴリズムが物体形状の境界に沿って処理を進めていくのに対し、CSG が、

- ・データの基本要素が閉じた立体や半空間などのプリミティブであり、頂点、稜線、平面といった境界情報を明示的な形で持っていない、
- ・立体がプリミティブの集合演算によって表現され

† Half-Tone Representation of the Object Defined by Set Operations of Convex Primitives by a Scanline Method by KOHE TOKOI (Faculty of Economics, Wakayama University) and TADAHIRO KITAHASHI (Institute of Science and Industrial Research, Osaka University).

†† 和歌山大学経済学部産業工学科

††† 大阪大学産業科学研究所

ているので、描画時に集合演算処理によって常に実際の可視面を決定しなければならない。

という性質を持つため、物体形状が CSG によって記述された場合には、その境界に沿って処理を進めることが困難なためである。

この解決策として、形状データを CSG 表現からいったん境界表現に変換し、スキャンライン法やデプスバッファ法などの高速な描画手法を用いる方法<sup>1)</sup>が提案されているが、この方法ではデータ量が少なく済むという CSG 表現の特長が生かされない上、変換に時間を要し、その際の計算誤差による幾何学的な矛盾の発生により、システムの暴走を招く恐れもある。このほか、CSG 表現から、オクト・トリー・モデルに変換する方法は、特定の視点の位置からの隠面消去が高速に行えるが、表示精度を上げようとすると大量のメモリを必要とし、また任意の視点から見た場合の隠面消去には時間がかかる。また拡張デプスバッファを用いる方法<sup>2)</sup>は、最も会話性に優れた方法であるが、透視図が作成できず、実現には大量のメモリが必要となる。

一方、視線探索法を高速化することも考えられる。視線探索法による陰影画像生成に時間がかかる理由は、その処理時間のほとんどが物体と視線との交差判定に費やされていることに起因する。提案されている多くの高速化技法<sup>3)-7)</sup>は、いずれもこの交差判定の回数を減じることに努力が払われ、相当な効果をあげているが、会話的な操作環境に応用できるまでの高速化は今のところ得られてはいない。

筆者らのアプローチは、これらのものと異なり、立体を構成する個々のプリミティブについてスキャンライン法を用い、そのスクリーンへの投影像の一様性を利用して、隠面消去処理と集合演算処理を同時に高速に行おうとするものである。

### 3. スキャンライン法

スキャンライン法やデプスバッファ法は、境界表現データの陰影画像生成において、立体形状の空間的な一様性を利用することにより、高速に隠れ面消去処理を実行するアルゴリズムである。このうち、デプスバッファ法は CSG で定義された形状の表示にも応用されているが、この方法は立体を構成する要素（境界表現の場合はポリゴン、CSG 表現の場合はプリミティブ）単位で描画を行うため、CSG 表現された形

状データの表示のためには文献 1) のようにディスプレイ側に形状定義を保存しておく機構が必要になる。一方スキャンライン法は、スクリーン上のある領域を処理している際に、その領域を処理するのに必要な形状データをあらかじめ選択しながら処理を進める<sup>10), 11)</sup>。この選択は、立体の形状データが境界情報ならば、その空間的な一様性を利用できるため、非常に高速に行うことができる。しかしこれをそのまま境界情報を持たない CSG 表現に応用することは難しい。このため視線探索法の高速化技法の中には、個々のプリミティブごとに外接箱を設定し、それに対してスキャンライン法を用いることによって、第 1 世代の視線の限定や影の部分の処理の高速化を行っているものがある<sup>2), 3)</sup>。

そこでスキャンライン法によって CSG 表現された形状データの陰影画像生成の可能性を検証するために、スキャンライン法の処理手順について整理してみる。スキャンライン法による隠面消去アルゴリズムには、いくつかのものがあるが、いずれも基本的にはだいたい次のような処理手順を用いている。

(1) 立体を構成するすべての形状要素（この場合ポリゴン）について、そのスクリーンへの投影像の上端と下端を求める。

(2) ディスプレイの上端のスキャンラインから始めて、下端までの一つ一つのスキャンラインについて、

(2-1) そのスキャンラインと交差する要素を選択する。

(2-2) 選ばれた個々の要素がスキャンラインから切り取る線分を求める。

(2-3) 切り取られた線分間で奥行き比較を行い、可視であるものを表示する。

したがってこのような隠面消去処理を実行する際に形状要素から引き出さなければならないデータは、

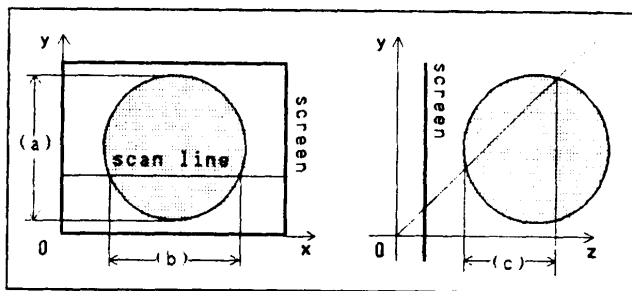


図 1 プリミティブの境界情報

Fig. 1 Boundary information of primitive.

① 物体形状のスクリーンへの投影像の上端および下端の位置 (図 1 (a)).

② 交差するスクャンラインと投影像が重なるスクャンライン上の範囲 (図 1 (b)).

である. ところが, CSG 表現された形状データにスクャンライン法を適用しようとするときには, 集合演算処理のためにこのような境界情報の算出を事前に行うことが難しい. そこで筆者らは立体を構成する個々のプリミティブに着目した. 通常個々のプリミティブには単純な形状のものが用いられることが多く, プリミティブの種類によってはこれらのものを算出することは不可能ではない. 本研究では CSG 表現に用いるプリミティブにこのような境界情報の算出が容易な凸形状の閉じた立体を用いる.

前述の処理手順では隠面消去の基本が線分間の奥行き比較であった. このため, これを一つ前のスクャンラインでの処理結果を利用して高速化することも可能であった. しかし形状要素がプリミティブの場合には, 投影像がスクャンラインから切り取る線分が, 実際には 3 次元空間上の閉曲線であるうえ, 集合演算処理も行われるため, 奥行き比較による可視判定は線分の場合ほど容易ではない. 本研究ではこの奥行き比較をディスプレイ上の画素単位で行い, 同時に集合演算処理も併せて実行することとする. これにより形状要素から引き出すデータは先の①②に,

③ 投影像内の 1 点に対応する形状要素上の奥行き方向の前端と後端の位置 (図 1 (c)).  
を加えなければならない.

#### 4. 集合演算の記述

次に, CSG 表現による形状記述の中心となる, 集合演算の記述方法について述べる. プリミティブの集合演算の記述には, 一般的に木構造が用いられるが, 本研究では文献 8) に示されているパターンマトリクスによる表現を用いる. 後節の準備のために, ここでこのパターンマトリクスによる集合演算の記述法について概説する.

いま, 形状要素 (これをエレメントと呼ぶ) の積集合演算による, 次のような組合せ  $S_j$ ,

$$S_j = E_{1j} \cap E_{2j} \cap \dots \cap E_{nj} = \bigcap_{i=1}^n E_{ij}. \quad (1)$$

をセグメントと呼ぶ. 立体形状  $S$  はこのセグメントの和集合演算処理によって定義される.

$$S = S_1 \cup S_2 \cup \dots \cup S_m = \bigcup_{i=1}^m S_i. \quad (2)$$

したがって, (1)(2)式より,

$$S = \bigcup_{i=1}^m \left\{ \bigcap_{j=1}^n E_{ij} \right\}. \quad (3)$$

これをマトリクス状に表現したものを,

$$S = \begin{pmatrix} E_{11}E_{12} \dots E_{1m} \\ E_{21}E_{22} \dots E_{2m} \\ \vdots \quad \vdots \quad \vdots \\ E_{n1}E_{n2} \dots E_{nm} \end{pmatrix} \quad (4)$$

をパターンマトリクスといい, 一つの列が一つのセグメントに対応している. 立体形状が任意の集合演算式によって記述されているとき, それを積和標準形に変形することによって, パターンマトリクス上に展開することができる. なお, 差集合演算処理は,

$$A - B \equiv A \cap \bar{B}. \quad (5)$$

であるので, エレメントに対し負のエレメントであることを示すフラグによって表現する. 本研究ではこのパターンマトリクスを用いた形状記述を文献 9) にならって積集合テーブルおよび和集合テーブルという二つのテーブルを用いて実現する (図 2).

なお, 文献 8) ではエレメントを半空間の境界面として取り扱い, ドメインエレメントを導入することによってその存在範囲を制限しているが, 本研究では, 後で述べるプリミティブ選択アルゴリズムの都合もあり, エレメントには凸形状を持った閉空間 (プリミティブ) を用いる.

#### 5. 陰影画像生成アルゴリズム

以降, 本研究において開発した陰影画像生成アルゴリズムについて説明する.

##### 5.1 プリミティブ選択アルゴリズム

提案手法はスクャンライン法によって, 処理の対象となる画素において可視となる可能性を持つプリミティブをあらかじめ選択し, その上で奥行き比較・集合演算処理を行うものである. このプリミティブ選択

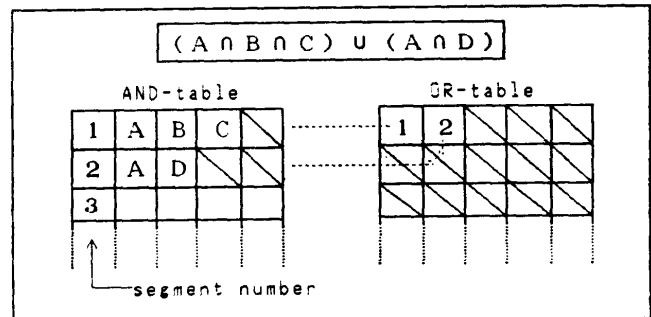


図 2 パターンマトリクスによる形状定義  
Fig. 2 Shape description by using pattern matrix.

手法には, Romney らの方法<sup>13)</sup>と呼ばれる, 基本形状に三角形板を用いた隠面消去アルゴリズムを応用しているが, 同方式で用いられている  $y$ -退出テーブルおよび  $x$ -退出テーブルを用いず, また  $y$ -占有テーブル,  $x$ -占有テーブルをリスト構造とすることによって, 処理を効率化している. このため, プリミティブはこれらのリストにつながるアトムとして取り扱われる.

(1) フレームに関する処理

(1-1) 立体を構成するすべてのプリミティブについて, そのスクリーンへの投影像の上端と下端の位置を求め (図3では, プリミティブ C, Dはそのスクリーンへの投影像の上端と下端, A, Bはそれが表示領域にクリップされる位置となる), 上端の位置についてバケットソートを行い,  $y$ -進入テーブルを作成する. 下端の位置は保存しておく.

(1-2)  $x$ -占有リストへのポインタおよび  $y$ -占有リストへのポインタを空に初期化する.

(1-3) ディスプレイの上端のスキャンラインから始め, 下端のスキャンラインまで(2)の処理を行う.

(2) スキャンライン上の処理

(2-1)  $y$ -進入テーブルの処理対象のスキャンラインの位置につながれているリストを  $y$ -占有リストに結合する. 図3ではいちばん上のスキャンラインにおいてAが, 3番目のスキャンラインにおいて, B, Cが  $y$ -占有リストに結合される. したがって  $t$  の位置のスキャンラインでは,  $y$ -占有リストには A, B, C の三つのプリミティブが登録されていることになる.

(2-2)  $y$ -占有リストが空でなければ, 以降の処理を行う. 空ならば, 次のスキャンラインの処理に移る.

(2-3)  $y$ -占有リストに登録されているプリミティブについて, 処理対象のスキャンライン上での左端と右端の位置を求め (図3では  $t$  の位置のスキャンラインが, その位置の  $y$ -占有リストに登録されている三つのプリミティブ A, B, C のスクリーンへの投影像

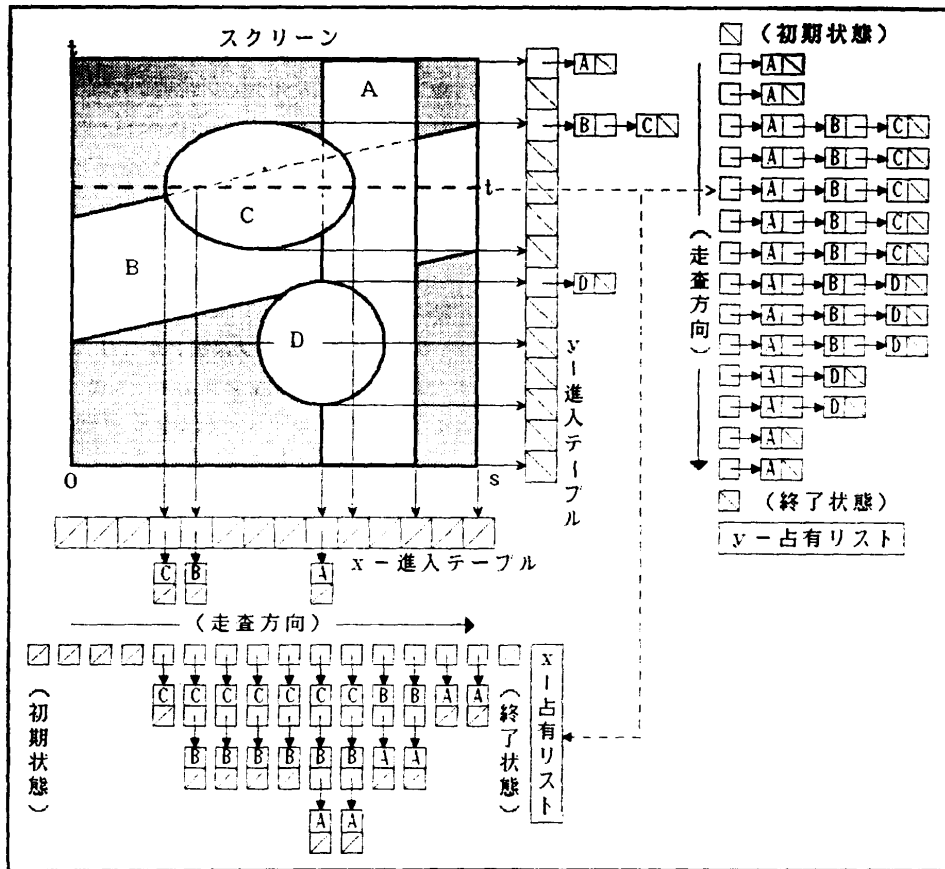


図3 プリミティブ選択アルゴリズム  
Fig. 3 Algorithm to select visible primitives.

と交差する位置を求めることになる), 左端の位置についてバケットソートを行い,  $x$ -進入テーブルを作成する. 右端は保存しておく.

(2-4) 処理対象のスカンラインのディスプレイの左端の画素から始めて, 右端の画素まで(3)の処理を行う.

(2-5)  $y$ -占有リストにつながれているプリミティブのうち, 下端の位置が処理対象のスカンラインの位置と一致するものを削除する.

(3) 画素に関する処理

(3-1)  $x$ -進入テーブルの処理対象の画素の位置につながれているリストを  $x$ -占有リストに結合する. たとえば図3において, プリミティブAが  $x$ -進入テーブルに登録されている位置では, 既に  $x$ -占有リストにはBおよびCが登録されており, これにAを追加することによって, この位置ではA, B, Cの三つのプリミティブが重なっていることがわかる.

(3-2)  $x$ -占有リストが空でなければ, その画素においてプリミティブの可視判定を行う. 空ならば, 次の画素の処理に移る.

(3-3)  $x$ -占有リストにつながれているプリミティブのうち, 右端の位置が処理対象の画素の位置と一致するものを削除する.

以上の処理により,  $x$ -占有リストには常に処理対象の画素を覆い隠すプリミティブが保持される.

5.2 集合演算処理と可視判定

提案手法は集合演算処理を画素単位に行うことを前提としている. また本研究では集合演算をパターンマトリクスによって記述している.

したがってこの場合の集合演算処理および可視判定は, 画素単位に,

(1) 個々のセグメントに対して, その画素における奥行き方向の積集合演算を行い (これはそのセグメントに属するプリミティブ間の, その画素における前端と後端の位置を比較によって実行できる), それらのセグメントの可視面を決定する.

(2) それらの可視面のうちもっとも視点に近いものを表示することにより, セグメント間の和集合演算処理と隠面消去処理を行

う. という処理を行うことによって実現できる.

この処理の対象となるセグメントは, それに属するプリミティブがすべて  $x$ -占有リストに登録されているものに限られる (負のプリミティブを除く). そこで筆者らはこのようなセグメントの抽出を高速に行うために, リファレンスカウントという技法を開発した.

5.2.1 リファレンスカウント

一つのセグメントに一つの変数  $RCx$  を与え, それをそのセグメントに属するプリミティブの数で初期化しておく. そして, いま図4(b)のように, スカンライン上の処理において,

- プリミティブを  $x$ -進入テーブルから  $x$ -占有リストに結合するとき, そのプリミティブが属するセグメントの  $RCx$  を1減ずる.
- プリミティブを  $x$ -占有リストから削除するとき, そのプリミティブが属するセグメントの  $RCx$  を1増す.

という処理を行う. これにより, あるセグメントが集合演算処理の対象となりうるかどうかの判定を, そのセグメントの  $RCx$  を調べることによって得ることができる. そこで上の処理のそれぞれに,

- $RCx$  を1減じた後, その結果が0となったセグメントを, 集合演算処理の対象となるセグメントを保持するリスト (これを可視セグメントリストと呼ぶ) に加える.
- $RCx$  を1増す前に  $RCx$  を調べ, それが0であれば

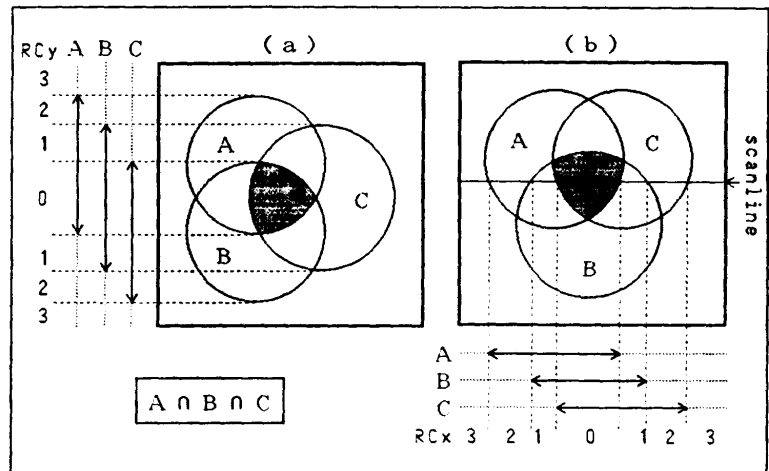


図4 プリミティブの占有領域とリファレンスカウントの値  
Fig. 4 The relation between occupied area of primitives and reference count.

ばそのセグメントを可視セグメントリストから削除する。

という処理を付け加える。したがって、ある画素における集合演算処理および隠れ面消去処理は、その時点で可視セグメントリストに登録されているセグメントについてのみ行えば良い。これにより、 $x$ -占有リストに加えられているプリミティブであっても、その属するセグメントが可視セグメントリストに加えられていなければ、その画素に関する処理は行わないで済む。

このリファレンスカウントは  $Y$  方向にも適用できる。 $X$  方向と同様に、一つのセグメントに対して一つの変数  $RCy$  を用意し、それをそのセグメントに属するプリミティブの数で初期化しておく。そして、

- ・あるスキャンライン上の処理を開始する際、プリミティブを  $y$ -占有リストに加えると同時に、そのプリミティブが属するセグメントの  $RCy$  を 1 減ずる。
- ・そのスキャンラインの処理が終了する際、 $y$ -占有リストからプリミティブを削除すると同時に、そのプリミティブが属するセグメントの  $RCy$  を 1 増す。

以上のようにすれば、処理対象のスキャンライン上にすべてのプリミティブが存在する場合のみ、そのセグメントの  $RCy$  は 0 となる (図 4 (a))。したがって、プリミティブが処理対象のスキャンラインと交差している場合でも、その属するセグメントのリファレンスカウントが 0 でないときは、そのプリミティブをスキャンライン上の処理の対象からはずすことができる。

なお、 $X$  方向、 $Y$  方向のいずれの場合も、負のプリミティブについてはこれらの処理を行わず、そのかわりこれが  $x$ -占有リストに組み込まれた際に、そのことを示すフラグをこのプリミティブ自身に立て、画素単位の集合演算処理の際に利用する。図 5 にリファレンスカウントを導入したセグメントの構成を示す。

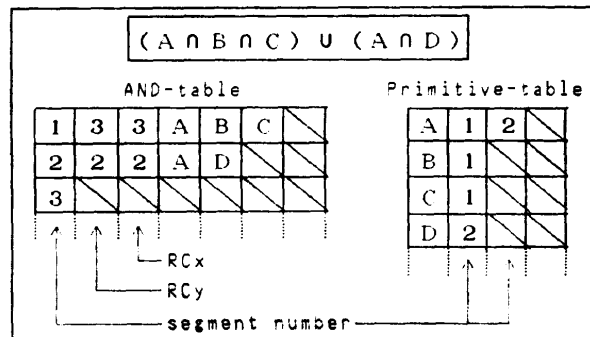


図 5 リファレンスカウントを加えたセグメントの記述  
Fig. 5 Segment description with reference count.

## 5.2.2 可視判定結果の再利用

ある画素における可視判定の結果は、その画素の次の画素における結果と一致することが多い。本研究で用いたスキャンライン法は、スキャンライン上の処理において、ある画素から次の画素に処理を移すときに、以前の画素において利用した情報を引き継ぐため、これを利用して可視判定の高速化を行える。スキャンライン上で隣合う二つの画素の間で判定結果が異なるのは、

- ①  $x$ -進入テーブルから新たなプリミティブが  $x$ -占有リストにつながる時。
- ②  $x$ -占有リストからプリミティブが削除される時。
- ③ プリミティブ間の奥行き方向の優先順位が入れ替わるとき。

の三つの場合である。したがってこれらのことが発生しなかった場合は、直前の画素の判定結果をそのまま利用でき、画素単位の集合演算処理を省くことができる。

## 6. インプリメンテーションの実際

本章では提案手法を実現するにあたって重要となる点について整理する。

試作したシステムにおいて、プリミティブは、

- ① プリミティブの種類を示すタグ。
- ② 負のプリミティブを示すフラグ。
- ③ プリミティブの局所座標系の原点の 3 次元空間上の位置。
- ④ プリミティブの局所座標系の原点を中心とする回転角。
- ⑤ プリミティブの局所座標系の原点を中心とする拡大係数。
- ⑥ プリミティブの色などの属性情報。

⑦ このプリミティブが属するセグメントテーブルへのポインタ。

⑧  $y$ -占有リストおよび  $x$ -占有リストを構成するための、次のプリミティブへのポインタ。

などのデータによって記述される。このうち①のタグは、実際にはこのプリミティブの各種の処理に関わる手続き群のポインタを保持する構造体へのポインタである。こうすることにより、プリミティブは陰影画像生成プログラムの本体からオブジェクトのような形式で取り扱いを受ける。このためプリミティブに対して各種の処理を行う手続き群は、プリミ

タイプの種類ごとに用意しなければならない。これには先に述べた境界情報を得るための、

① プリミティブの投影像の上端および下端の位置を求める手続き。

② プリミティブの投影像のスカンライン上の左端および右端の位置を求める手続き。

③ プリミティブの投影像内の1点における前端と後端の位置を求める手続き。

等の手続きに加えて、プリミティブの3次元空間上で位置決めや視点の移動を行うための、

④ プリミティブの平行移動、回転の手続き。および、陰影付けを行うための、

⑤ プリミティブの表面上の1点が与えられたときの、その点における色または法線ベクトルを求める手続き。

などが必要である。次にこれらの手続きの具体例について、プリミティブとして一般的によく用いられる球(楕円体)、円柱(楕円柱)、円錐(楕円錐)、および直方体・多角柱・多角錐などの凸多面体の各々の場合について概説する。

視点は図6のような左手系空間において、常に原点

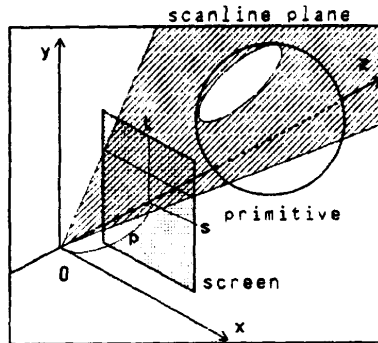


図6 スカンのライン平面  
Fig. 6 Scan line plane.

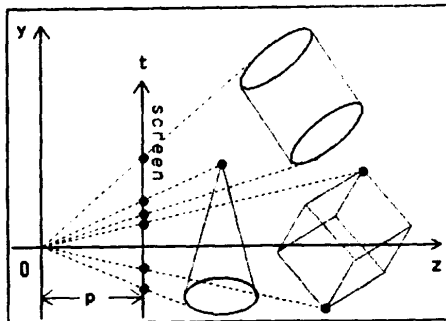


図7 プリミティブの投影像の上下端  
Fig. 7 The top and bottom of projections of primitives.

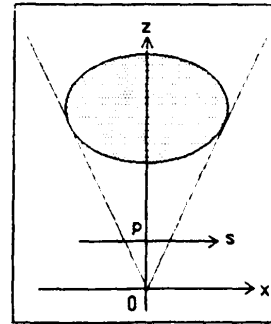


図8 交差線の左右端  
Fig. 8 The left end and right end of intersection line.

の位置にあり、Z軸上の正の方向を向いているとする。また、ディスプレイ面は仮想的にZ軸上のpの位置にXY平面と平行に存在し(これをスクリーンと呼ぶ)、スカンラインはX軸と平行で、Y=tであるスクリーン上の直線として表される。この直線と視点を含む平面をスカンライン平面と呼び、

$$py - tz = 0 \quad (6)$$

のように、tをパラメータとした平面群となる。したがって、球のスクリーンへの投影像の上端と下端は、この平面群のうちこの球と接するものtとして得られる。円柱の場合はその上面と底面の周囲の円について、この平面群のうちそれらに接する平面のtの最小値と最大値を求める。円錐の場合は頂点と底面の周囲の円について、円柱と同様の処理を行えば良い。多面体の場合はそれを構成する頂点の位置を透視変換し、その最大値と最小値を求める(図7)。

次に、スカンライン上の左端と右端の位置であるが、これにはまずプリミティブと処理中のスカンライン平面との交差線(のXZ平面への投影像)を求め、スカンライン上の位置sをパラメータとする直線群、

$$px - sz = 0 \quad (7)$$

について、上端・下端を求めた場合と同様な方式でsを求めれば良い(図8)。

形状要素がポリゴンの場合には、この範囲を一つ前のスカンラインにおける処理の結果に対してインクリメンタル計算を適用することで、高速に得ることができるが、これがCSG表現のプリミティブである場合には、その投影像の外形についてこのような線形性を期待できない。しかしCSG表現の表示の場合は境界表現の表示の場合ほど、1本のスカンラインと交差する形状要素が多量にならないので、これをスキャ

ンラインごとに独立に計算してもさほど問題とはならない。

こうして、ある  $s$  が与えられたときの奥行き方向の前端と後端の位置  $z$  は、交差線と上下との関係によって得ることができる。

なお、プリミティブの表面上の1点における法線ベクトルの計算は、その点の位置がわかれば、多くの場合それほど困難ではないため、ここではその解説を省略する。

## 7. 実験結果

本手法および視線探索法を用いて陰影画像を生成した場合の処理時間を表1および表2に示す。これらはパーソナルコンピュータ (i80286 [8 MHz]+i80287 [8 MHz]) 上で  $160 \times 100$  画素の画像を生成するのに要した時間である。表1は空間中のランダムな位置に置かれた球 (大きさは固定) の描画に要した時間であり、この例では本手法は何の高速化技法も用いない視線探索法に比べて、ほぼ100倍の処理速度を得ている。表2は形状データに集合演算が含まれている場合の処理時間である。これは集合演算により可視部分が変化しているので表の各行の間に厳密な関連はないが

表1  $N_p$  個のランダムな球の描画に要する時間 (160×100 画素)

Table 1 Calculation time in case of randomly placed spheres (160×100 pixels).

データ数 $N_p$	計算時間 (単位: 秒)		速度比
	視線探索法	本手法	
50	364	5	72.8
100	726	8	90.8
150	1,086	11	98.7
200	1,447	15	96.5
250	1,808	18	100.4
300	2,169	20	108.5

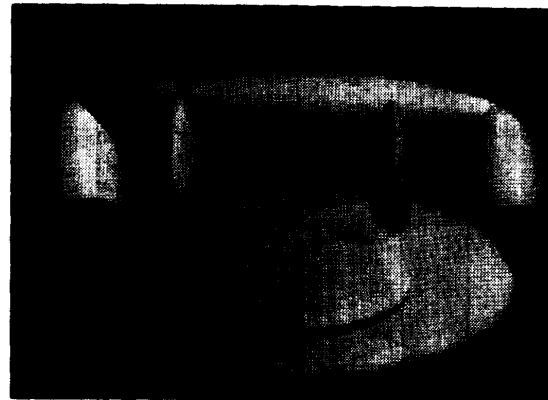
表2 プリミティブ総数  $N_p=120$  のとき、 $N_s$  個のセグメントの描画に要する時間 ( $N_p/N_s$  は1セグメント当りのプリミティブ数, 160×100 画素)

Table 2 Calculation time for  $N_s$  segments that constructed from 120 primitives (160×100 pixels).

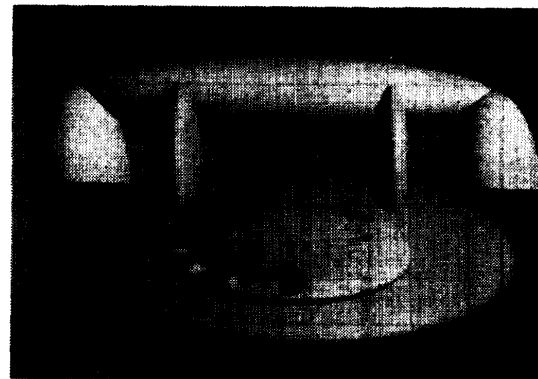
セグメント数 $N_s$	$N_p/N_s$	計算時間 (単位: 秒)		速度比
		視線探索法	本手法	
120	1	867	9	96.3
60	2	874	4	218.5
40	3	894	4	223.5
30	4	901	4	225.3

( $N_p/N_s$  が1から2へ変化した際の処理時間の変化は、集合演算による可視形状の変化が原因である)、この場合には本手法は視線探索法に比べ、ほぼ200倍の処理速度を得ており、なおかつセグメント形状が複雑化したときの処理速度の低下も少ないことがわかる。一方、本手法はデータ量に比べて可視部分が大きいようなデータに対しては、視線探索法に対する速度比が小さくなる。これは画素ごとの固定的な処理、特に陰影付け/ハイライト処理を視線探索法と同じ方法で行っていることに起因している。これを改善するには陰影付け/ハイライト処理をプリミティブの形状のスキャンライン上での一様性を利用するなどして、高速化する必要がある。

この二つはいずれも条件が本手法にとって有利であった場合の例であり、一般的な条件の下では本手法の視線探索法に対する速度比はだいたい30~80倍程度であると思われる。しかしこれはデータ量の増加、特にセグメント形状の複雑化に伴って大きくなる傾向



(a) 160×100 pixels, 3 bits/plane



(b) 640×400 pixels, 24 bits/plane

図9 本方式による陰影画像の生成例  
Fig. 9 Rendering example.



にある。

図9に本手法によって生成した画像の例を示す。この形状データはまず同図(a)に示す解像度を落とした画像(160×100画素)を生成しながら、何も無いところから場当たりに作成したのであるが、この程度の画像の生成は30秒以内で行えるため、この形状データは30分ほどで作成できた。同図(b)はこの解像度を上げ(640×400画素)、フルカラーのフレームバッファに表示したもので、この生成には約10分(このうちフレームバッファへのデータ転送時間が約2分)かかった。

## 8. む す び

本稿ではCSGによって定義された形状データからの陰影画像生成にスクリーン法に基づくアルゴリズムを用いることによって、可視プリミティブ選択や集合演算処理にプリミティブの形状のスクリーンへの投影像の一様性が利用できることを示し、その具体的な手法としてRomneyらの方式を基にしたプリミティブ選択アルゴリズム、リファレンスカウントを用いた可視セグメント選択アルゴリズム、および可視判定結果の再利用による集合演算処理の高速化技法について述べ、さらにCSGで定義された形状の陰影画像生成において、本手法が何の高速化技法も用いていない視線探索法に比べて最大で200倍程度の処理速度が得られることを確認した。

ただし、本手法はあくまで形状把握のための陰影画像生成を目的としており、視線探索法において可能な鏡面反射、屈折などの光学的処理については考慮していない。またスクリーン法でも対象が平面のみであれば影付け処理のほか、鏡面反射や屈折などの処理を行うことも不可能ではないが、本方式のように曲面や集合演算処理を含む場合にこれらを実現することは非常に困難である。しかしながら筆者は陰影画像生成を形状把握の補助のために行おうとする場合には、これらのものは応答時間の短縮に比べてそれほど重要ではないと考える。もしこれらのものが必要であるならば、本手法を用いて形状データを作成し、その後視線探索法を適用すれば良い。また本手法を視線探索法を高速化に応用することも可能である。

本研究ではプリミティブの形状に制限が多く、本手法の形状表現の自由度は高くない。それでもCSG表現は集合演算処理により多彩な形状表現が可能であるが、やはりこの制限は少ない方が望ましい。

リファレンスカウントを用いた可視セグメント選択アルゴリズムは、パターンマトリクスによる形状記述のように積集合演算処理を多用する場合には非常に有効である。しかしパターンマトリクスは複数のセグメントにわたって同一のプリミティブが登録されている場合は、木構造に比べて処理の効率が悪い。そこで今後の課題として、木構造で表現されたCSGデータに対して本方式を適用することが考えられる。

また、図9のダイヤル部分のように、多数の負のプリミティブで構成されたセグメントは、本手法では現在のところ負のプリミティブにリファレンスカウントなどの高速化技法を適用していないため、それ以外のセグメントに比べかなり処理速度が低下する。本手法を実用的なものとするためには、なんらかの方法でこれを改善しなければならない。

今後はこれらのほか、より多様なプリミティブへの対応や、本方式に適したアンチエイリアシング技法の開発、陰影付け/ハイライト処理の高速化などを行い、本手法を実用的なソリッドモデラに応用できるよう研究を進める予定である。

**謝辞** 本稿を執筆するにあたって助言をいただいた和歌山大学経済学部産業工学科竹内昭浩教授に感謝いたします。

## 参 考 文 献

- 1) Voelcker, H. B. and Requicha, A. G.: Geometric Modeling of Mechanical Parts and Processes, *IEEE Computer*, Vol. 10, No. 12, pp. 48-57 (1977).
- 2) 川島, 太田, 徳増: CAD 対話インターフェースのためのソリッドモデル隠面消去法, 情報処理学会論文誌, Vol. 28, No. 2, pp. 147-155 (1987).
- 3) Roth, S. D.: Ray Casting for Modeling Solids, *CG & IP*, No. 18, pp. 109-144 (1982).
- 4) 中前, 多田村, 西田: 局所レイトレーシングによる陰影画像表示, 情報処理学会論文誌, Vol. 27, No. 11, pp. 1077-1085 (1986).
- 5) Glassner, A. S.: Space Subdivision for Fast Ray Tracing, *IEEE CG & A*, Vol. 4, No. 10, pp. 15-22 (1984).
- 6) Fujimoto, A. and Iwata, K.: Accelerated Ray Tracing, *Proceedings of CG Tokyo '85*, pp. 41-65, Springer Verlag, Tokyo (1985).
- 7) 村上, 佐藤, 広田: Voxel 分割を用いた高速化レイトレーシング, 第32回情報処理学会全国大会論文集, pp. 2081-2082 (1986).
- 8) 木下: 形状モデリングを核としたCAD/CAMシステム TIPS-1, *PIXEL*, No. 24, pp. 70-81

- (1984).
- 9) 山本: *The 3 Dimensional Computer Graphics*, pp. 58-62, CQ 出版社, 東京 (1983).
- 10) Newman, W. M. and Sproull, R. F.: *Principles of Interactive Computer Graphics*, 2nd ed., pp. 372-376, McGraw-Hill, New York (1979).
- 11) 山口: コンピュータディスプレイによる図形処理工学, pp. 253-257, 日刊工業新聞社, 東京 (1981).
- 12) 床井, 藤阪, 北橋: 2次曲面を含む半空間式の集合演算によって定義された3次元形状のスキャンライン法による表示, 第32回情報処理学会全国大会論文集, pp. 2083-2084 (1986).
- 13) 床井, 北橋: スキャンライン法を用いた CSG データからの高速陰影画像生成アルゴリズム, 昭和61年電気関係学会関西支部連合大会講演論文集, P. S 49 (1986).

(昭和62年9月11日受付)

(昭和63年11月14日採録)



北橋 忠宏 (正会員)

昭和14年生。昭和37年大阪大学工学部通信工学科卒業。昭和43年同大学院博士課程修了。同年同大学基礎工学部助手, 助教授を経て, 昭和54年豊橋技術科学大学工学部情報工学科教授。現在, 大阪大学産業科学研究所教授。3次元物体・3次元運動認識のための視覚システム, 自然言語処理, 学習・推論機構に関する研究に従事。工学博士。電子情報通信学会, IEEE, 人工知能学会各会員。



床井 浩平 (正会員)

昭和36年生。昭和59年豊橋技術科学大学工学部情報工学科卒業。昭和61年同大学院修士課程修了。同年和歌山大学経済学部助手, 現在に至る。コンピュータグラフィックスに関する研究に従事。電子情報通信学会, 社会経済システム学会各会員。