勉強会レジュメ
画像処理ハードウェアについて

秤谷 雅史

平成９年５月２９日(木)

1 高速化のための画像処理方式

従来の逐次型計算機では、膨大な２次元配列の画像データを処理操作することが不得手であるため、処理速度が遅いという欠点を有している。そのため、画像処理アルゴリズムを高速に実行する専用の画像処理ハードウェアの研究、開発が盛んに行われるようになってきた。これは、ＬＳＩ、ＶＬＳＩ技術の急速な進歩によるところが大きい。最近では商品化も進み、画像処理システムの核として広く使われ始めている。こうした画像処理ハードウェアについて、高速化のための画像処理方式についてまとめてみる。

1.1 高速化方式

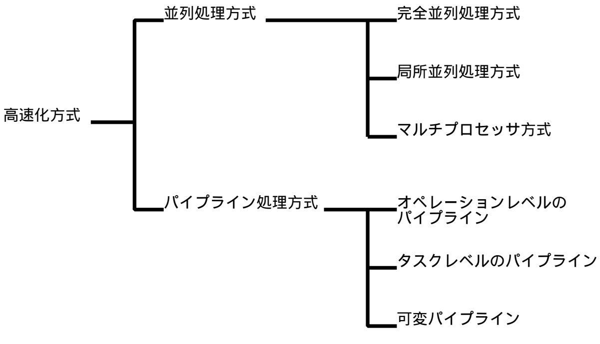

図 1: 画像処理の高速化方式

画像処理アルゴリズムの特質を考慮した効果的な方式やアーキテクチャが種々開発されており、それらをまとめると図 1となる。高速化方式の基本は、並列処理方式とパイプライン方式である。

並列処理方式は、画像処理アルゴリズムにおける２次元的な並列性を利用した高速化法で、これは、並列性の度合いによってさらに図 1のように分類できる。

一方、パイプライン処理方式は、パイプライン状に接続した演算モジュール群に画像データを順次入力し、ある遅延時間後連続して出力が得られるようにしたものである。これは演算モジュールの内容によって図 1のように細分類できる。

以下、図 1の各方式について述べる。

1.2 完全並列プロセッサ

Ｕｎｇｅｒは図 2に示すような完全並列型プロセッサの概念を提案した。これは、互いに接続された演算モジュール群と、それらに処理命令を出す主制御部とから構成されている。各演算モジュールを並列動作させることにより、全画素を同時に処理することができる。

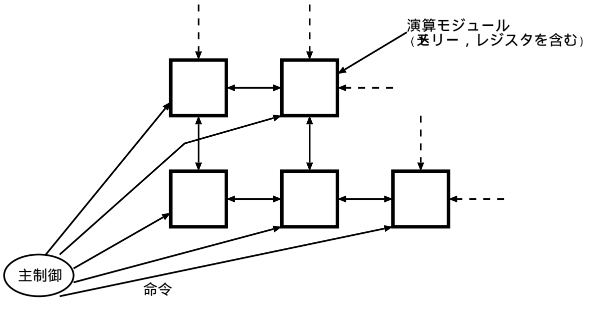

図 2: 完全並列型プロセッサの概念

このタイプのプロセッサは、回路接続が大規模で実装面で困難さがある。そのため、実用的な画像処理システムとしては限界があるように思える。

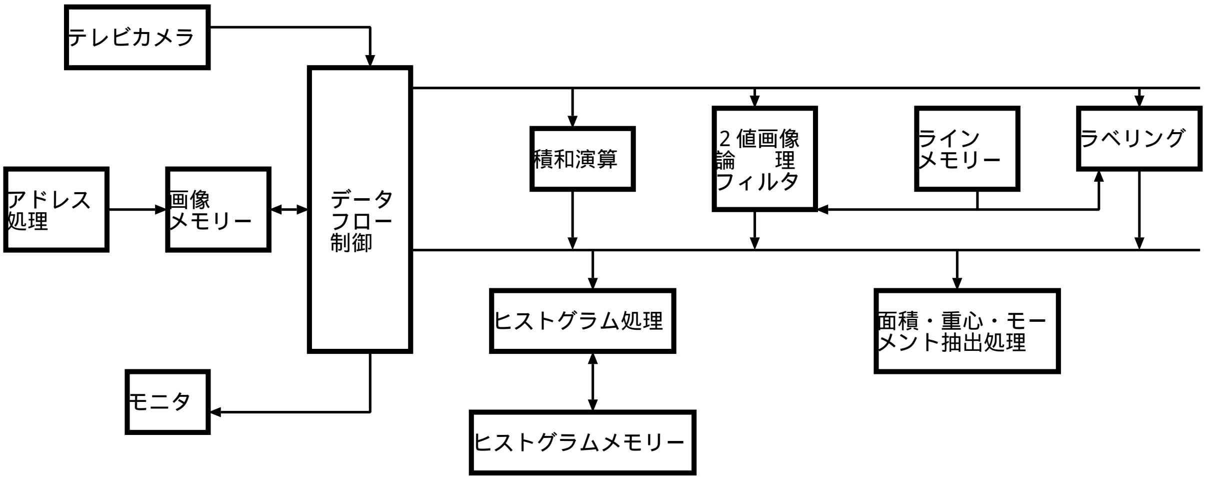

図 3: 局所並列型画像処理プロセッサ

1.3 局所並列型プロセッサ

画像処理で頻繁に用いられる平滑化、微分、細線化などは、各画素の値とその近傍画素の値を用いて計算する局所並列型の処理である。対象とする局所領域の大きさは $3\times3 \sim 16\times16$ 程度のことが多い。

この処理は、ハードウェア化に適したもので、局所領域内の画像データに対して、並列アクセスあるいは並列処理する回路を有し画像全体の処理はその局所処理を走査することで行う。ワンチップ化された局所並列型プロセッサと画像メモリーで画像処理システムを構成することができ、ハードウェア規模や価格の点で実現化しやすいものである。

局所並列型画像処理プロセッサの例を図3に示す。画像メモリーを中心に画像プロセッサと接続した構成をとっている。画像プロセッサとしては、局所並列性を利用する積和演算、2値画像論理フィルタやラベリングなどの前処理用プロセッサと、ヒストグラム処理、面積・重心・モーメント抽出処理などの特徴抽出用プロセッサがある。処理目的に応じて、これらのプロセッサを選択することができ、たとえば、テレビカメラからのデータに対して雑音除去と行って濃度頻度分布をとるといった一連の処理をパイプラインに実行できる。図3のシステムでは、上記の画像プロセッサ以外に画像メモリーに対するアドレス処理やデータフロー制御処理などのモジュールもＬＳＩ化しており、小型化と低価格化に寄与している。研究開発用のツールとしてだけでなく、画像処理機能と内蔵して自動化装置へ組み込むことも可能である。

1.4 マルチプロセッサ

複数の演算モジュールや汎用プロセッサと結合して、柔軟性のよい処理が実行できるようにしたシステムである。結合方式としては、バス結合、リング結合、クロスバ結合などがある。

図4はＭＡＣＳＹＭのシステム構成である。マスタプロセッサ１台、スレーブプロセッサ16台、および可変構造型共有メモリーが非同期型共通バスで接続されている。文書画像の理解を中心に、マルチプロセッサによる並列処理の有効性について研究開発が行われている。

このタイプのプロセッサでは、多数のプロセッサの結合方式、マルチプロセッサ向きの画像処理アルゴリズムなどが今後の課題である。

1.5 パイプライン型プロセッサ

パイプライン処理は、連続的に入力される画像データを次々に処理する場合に効果的な手法である。画像処理においては、次のようなレベルでのパイプラインが考えられる。

- オペレーションレベルのパイプライン
- タスクレベルのパイプライン

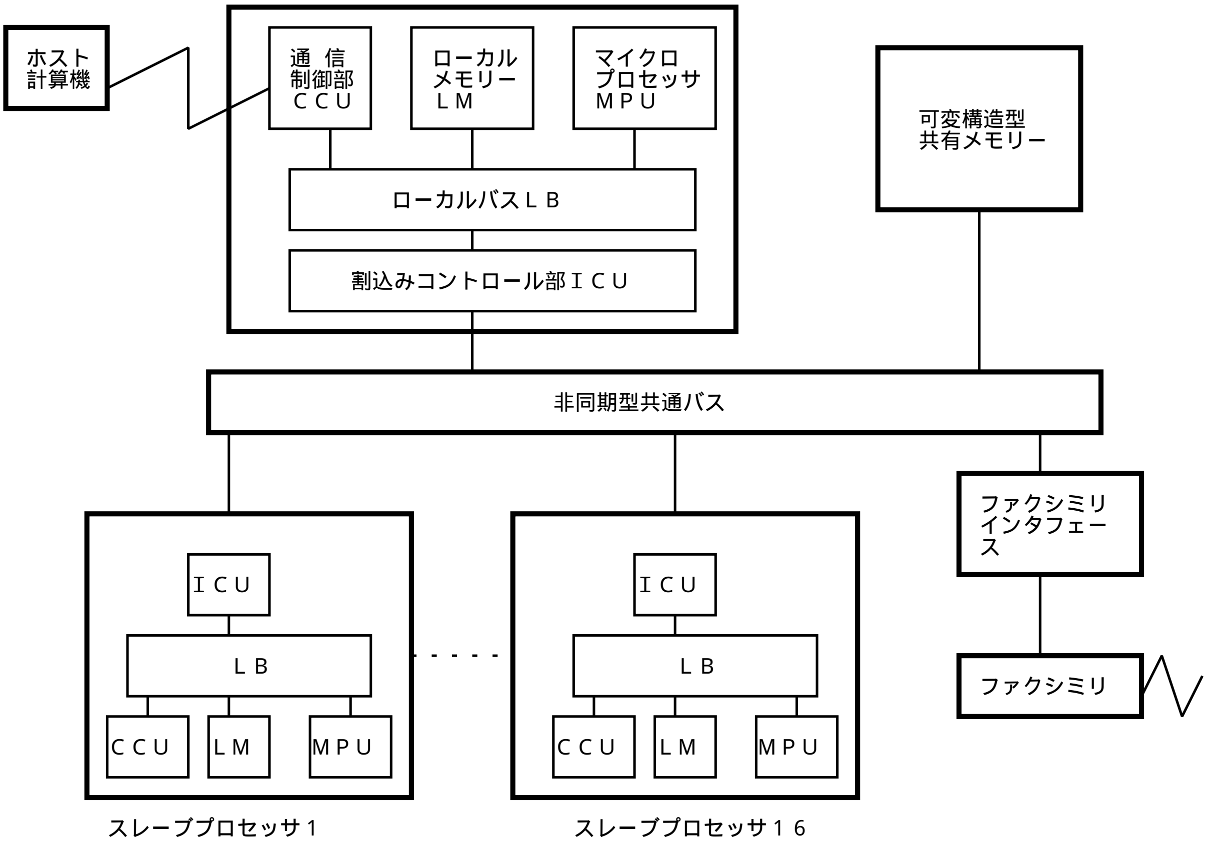

図 4: マルチプロセッサ型画像処理システムの構成例

- 可変パイプライン

である。以下、それぞれについて述べる。

1.5.1 オペレーションレベルのパイプライン

画像処理オペレーション (たとえば、微分や高速フーリエ変換など) を構成する基本演算 (乗算、加算など) をパイプライン動作させて高速化を図ったものである。

1.5.2 タスクレベルのパイプライン

画像処理オペレーションの系列をパイプライン動作させ、ある画像処理タスクの実行を高速化するものである。

また、画像処理システムにおける一連の画像操作 (画像データの入出力、処理など) を、画像処理タスクとしてとらえたパイプライン処理も考えられる。

1.5.3 可変パイプライン

並列処理、パイプライン処理によって、画像処理の高速性が図られるが、その反面、どちらも処理に融通性を欠くことが多い。そこで、システムの柔軟性の問題に対処するための試みが、可変パイプラインである。これは、複数の演算モジュールのパイプラインパスをプログラムによって任意に設定できることをいう。

パイプラインパスの設定方式としては、データにタグをつけることによりソフトウェア的にデータの流れを制御するもの、プログラマブルな交換機によってハードウェア的に切り換えるもの、などが提案されている。

このタイプのプロセッサは図 5のように多段スイッチによるネットワークによってハードウェア的に演算モジュール間の接続を可変にしている。このアーキテクチャによれば、テレビカメラから入力され画像データが、ネットワークと相互結合し

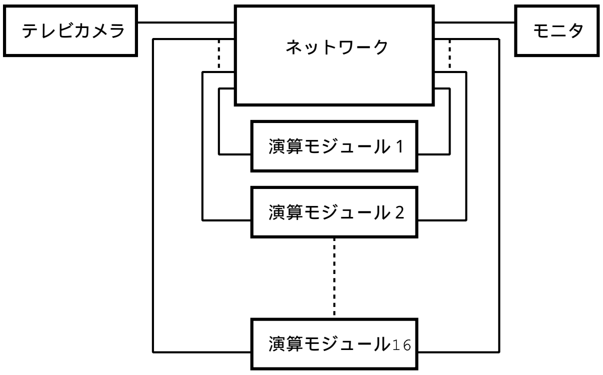

図 5: パイプライン可変型プロセッサの構成例

た演算モジュールを通過する間に、演算モジュールの処理機能に応じた画像処理を実行できる。

2　ボリュームレンダリング

2.1　ボリュームレンダリングとは

サーフィスモデルやソリッドモデルなどにより与えた３次元立体に対して光源情報をもとに明るさや色を付与して、より現実に近い質感を与える作業を**レンダリング** (rendering) といい、現在のＣＧにおける中心的なテーマとなっている。通常のレンダリング法では、物体の表面がどのように見えるかを計算し、画像として表す。物体をボクセル (用語の説明参照) の集まりとして表現したような場合でも、レンダリングの前にまず表面の抽出を行い、得られた表面に対してレンダリング法を適用するのが普通である。

これに対して表面抽出を行うことなく、ボリュームデータのまま描画を行う方法をボリュームレンダリングと呼ぶ。ボリュームレンダリングでは、物体の表面だけでなく、半透明表示などによって、物体の内部や背後の状況も画像として表現することが可能なので、ボリュームレンダリングは医療画像や科学技術計算結果の３次元データの可視化などに使用される。これらの処理では３次元物体の内部構造を調べる必要があり、ボリュームレンダリングは必要不可欠となりつつある。

2.2　ボリュームレンダリングの特徴と欠点

- ボリュームレンダリングの特徴
  - テクスチャ(図形表面に描かれた模様など)、フラクタル (自己相似)、動的に変形する物体などの表現がサーフィスレンダリングより容易に行える。
  - 物体の内部構造の表現が可能なので、油絵のように内部構造を反映した表現ができる。
- ボリュームレンダリングの欠点
  - ボリュームデータは３次元構造を持つので、非常に大きなメモリが必要であり、また処理に時間がかかる。
  - ３次元配列として離散化されているので、回転やズーム処理でゆがみが目立つようになる。
  - 法線ベクトルなど物体表面に関する情報がなく、これらを近接ボクセル値より計算する必要がある。

2.3　用語の説明

- シェーディング (**shading**) :立体をある光源の下で観察すると光が直接当たらない部分や、光源との方向により明るさがいろいろ変化する。ＣＧの分野において、立体を構成する各面に対して、光源位置や周囲環境条件により定まる明るさを決定する方法。
- レイトレーシング (**ray tracing**) :なめらかな平面に光線が照射されると、一部は反射し一部は内部へ向かって進む。これらの表面で分離した光は２本の新たな光としてさらにほかの表面に照射されて分離を繰り返す。従来の陰影モデルでは、光源から出発して反射分離を繰り返し演算を行うので視点まで到達する光は少なく、最終的なＣＧ画像生成には多くの演算が不要なものとなる。一方、レイトレーシング法では逆に、視点方向から出発して、最終的に得たい２次元ディスプレイ平面上の各画素点だけの輝度情報を得る。

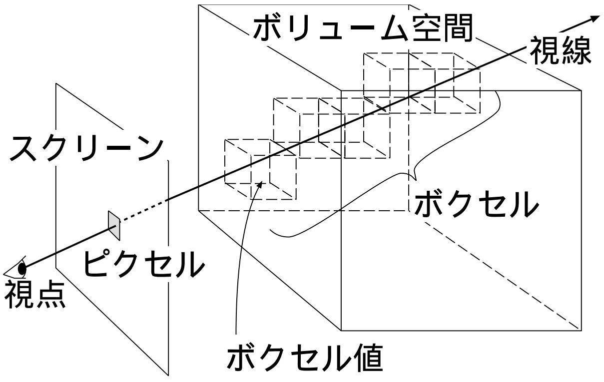

図 6: ボリュームレンダリングにおける用語

図 6における各用語においてあらかじめ説明を加えておく。

- ボクセル　:ボリュームレンダリングで扱う３次元データにおける個々の要素のこと。不透明、半透明、透明の色値をとりうる。
- ピクセル　:スクリーンの各画素点のこと。

2.4 ボリュームレンダリングにおける並列処理の形式

並列処理の方式としては次の３つが考えられる (図 7)。

- ボクセル並列処理　:視線と交差するボクセルを全て読み出し、ピクセル値計算を多数のプロセッサで行う方式。この方式では３次元画像メモリの構成が重要な役割を果たす。
- ピクセル並列処理　:複数の視線が同時に処理される方式である。具体的には、スクリーンをいくつかのサブスクリーンに分割し、各サブスクリーンにプロセッサを割り当て、ピクセル値計算を並列化する。
- ボリューム並列処理　:ボリュームをいくつかのサブボリュームに分割し、各ボリュームにプロセッサを割り当てる方式。各プロセッサでは担当ボリューム内の各ボクセルのスクリーンへの影響を計算し、計算結果をマージすることにより最終的な画像を得る。

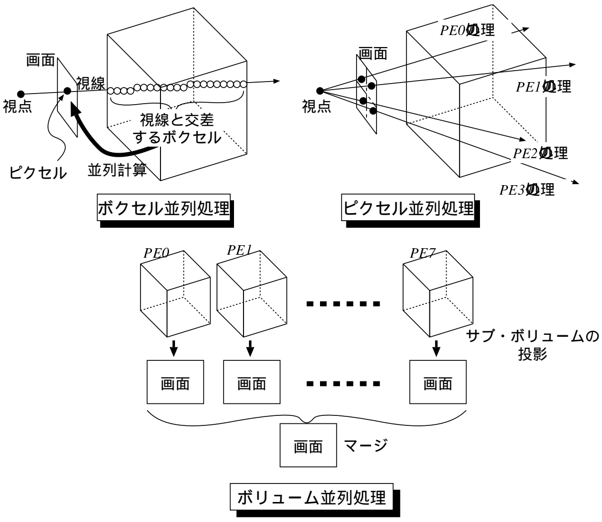

図 7: ボリュームレンダリングにおける並列処理の形式

さらにピクセル並列処理とボクセル並列処理を融合したハイブリッド並列処理という方式もある。

3 ＲｅＶｏｌｖｅｒ／Ｃ４０

3.1 全体構成

- システム制御ユニット（ＳＣＵ）:ＲｅＶｏｌｖｅｒ／Ｃ４０とホストマシンの中間に位置しており、ホストマシンからのボリュームデータ／制御コマンド／各種パラメータを各部に伝達する。
- 視線生成ステージ（ＲＣＳ）:図 8に示すように、多数のプロセッサが並列に配置されている。ＲＣＳでは、ＳＣＵから送られてくる視点、スクリーンの情報をもとに、視点から各ピクセルを通る視線ベクトル (視線) の生成を行う処理をスクリーンを分割してプロセッサに割り付けることにより並列に行う。
- ピクセル値計算ステージ（ＰＣＳ）:ＰＣＳは図 8に示すように、プロセッサが１次元に接続された構造であり、各プロセッサは、ボリュームデータ、ボクセル値を色、透明度に変換するための属性テーブルを保持している。各プロセッサの処理はＲＣＳから送られてくる視

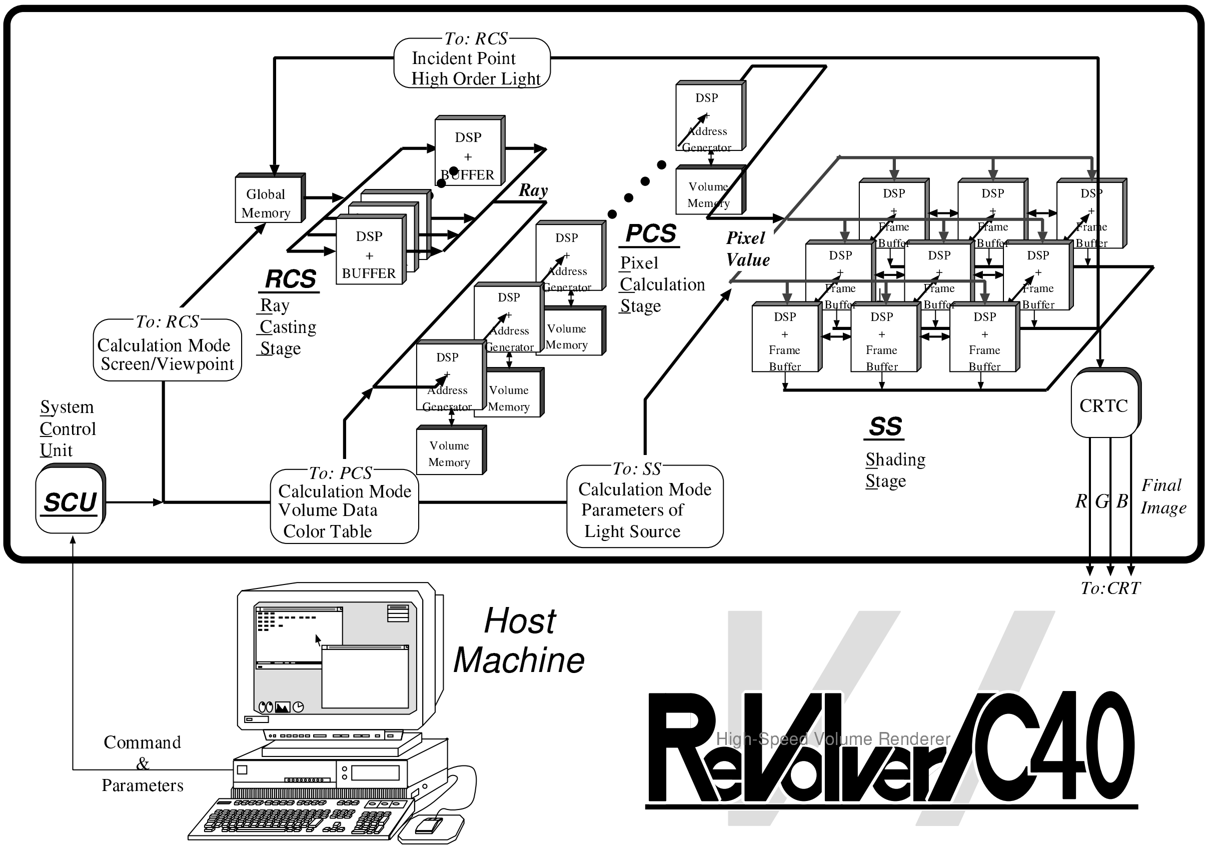

図 8: ＲｅＶｏｌｖｅｒ／Ｃ４０の全体構成

線データによりボクセルをサンプリングし、ピクセル値計算を行う。

- シェーディングステージ（ＳＳ）:図 8に示すように、プロセッサは画像処理に適したメッシュ状に配置されており、シェーディング処理などのフィルタ処理をスクリーン分割により並列実行する。またＳＳは画像のクリッピング処理や拡大縮小の処理を行う機能を持つ。

3.2　ＲｅＶｏｌｖｅｒ／Ｃ４０の特徴

3.2.1　半透明ボリュームの表示

ボリュームを半透明なものとして扱うには、図 9に示すように、ピクセル値計算の際に視線と交差する全ボクセルに対してアクセスする必要があり、計算量が増大するために従来のボリュームレンダリングではサポートされていなかった。ボリュームを半透明な物体として扱うことにより、ボリューム全体の状況を把握するのに役立つ。これは科学技術計算の結果の可視化による解析手法として非常に重要である。

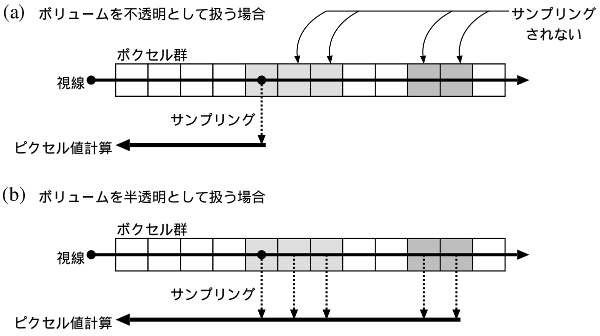

図 9: 参照するボクセルの違い

3.2.2　サンプリング方法の単純化

従来のサンプリング方法では、視線の方向に対して等間隔にボクセルをサンプリングしてきた(視

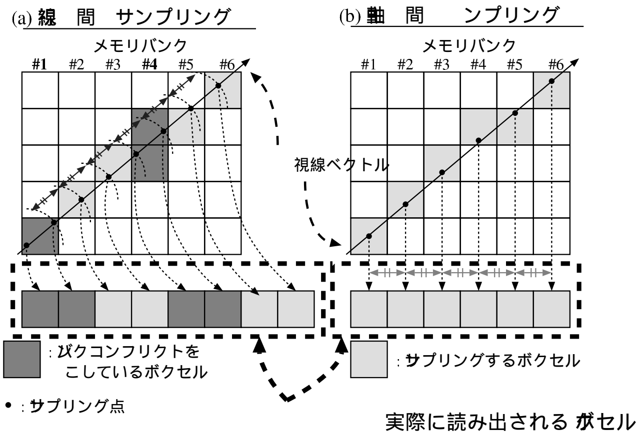

図 10: サンプリング方法の違い

線等間隔サンプリング法)。視線等間隔サンプリング法では図 10 （ａ）のように、ボリュームデータをメモリバンクに分割して格納すると、同一視線上の複数のボクセルが同一メモリバンクに格納されている状況が発生する。これは、ピクセル値計算の並列実行を妨げ、高速化の障害となる。

この問題を回避するためにＲｅＶｏｌｖｅｒ／Ｃ４０では、視線ベクトルの要素の絶対値のうち、最大値をもつ座標軸を主軸と定め、この主軸の方向に対して等間隔にボクセルをサンプリングする方法 (主軸等間隔サンプリング法) を採用している。この主軸等間隔サンプリング法により、図 10 （ｂ）に示すようにバンクコンフリクトが生じなくなる。

3.2.3 バンクコンフリクトフリーなメモリ構成

主軸等間隔サンプリング法では、ある座標軸に垂直な平面にボリュームデータを分割してメモリバンクに格納すれば、その座標軸を主軸として持つ視線に関してはバンクコンフリクトフリーとなる。しかし、任意方向の視線の主軸が、この座標軸を主軸として持つとは限らない。そこで、x,y,z 軸のいずれにも垂直な平面が存在するように分割すれば任意の方向の視線に対してバンクコンフリクトフリーとすることができる。

そこで、ＲｅＶｏｌｖｅｒ／Ｃ４０のメモリ構造は図 11のようにボリューム空間を x,y,z 軸に垂直なｎ枚の平面に分割し、x,y,z − 平面群とする。ただし、ｎはボリューム空間の１辺の大きさを表している。

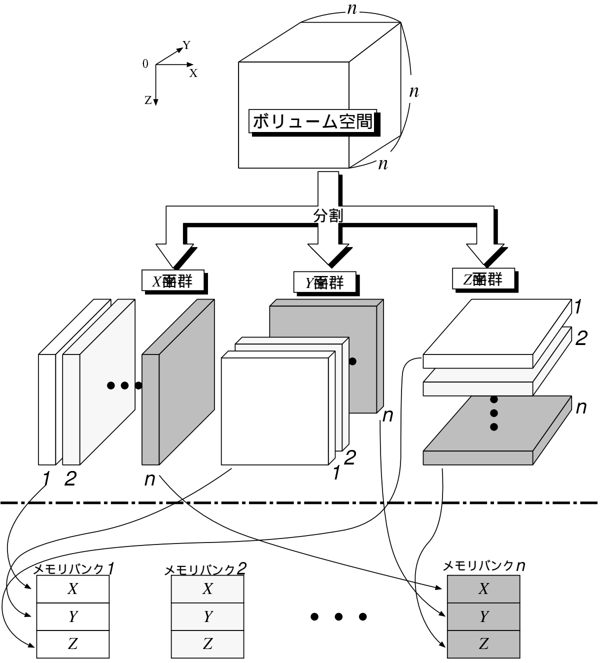

図 11: ボリュームデータの格納方法

各平面群に対して、平面に 1,2,· · ·,n と番号を付け、同じ番号を持つ平面のデータをその番号のメモリバンクの対応する領域に格納する。すなわち、ボリューム内の座標 (p,q,r) のボクセルは、メモリバンクｐのＸの領域、メモリバンクｑのＹ領域、メモリバンクｒのＺ領域の３箇所に格納されるので、ボリュームデータが３重化されて格納されることになる。

この構成により、各平面群の平面は全て別のメモリバンクに格納されているので、視線が任意の方向であっても、主軸等間隔サンプリング法を行うことによりバンクコンフリクトが生じないことを保証できるようになる。

3.2.4 シェーディング処理について

シェーディング処理では、視線と表面が交差する点での表面の法線を求めることで、光源に対して表面がどのような向きにあるかを計算し陰影を付ける。このため法線が必要不可欠であるが、ボクセルは離散的な座標により表されているため、法線は近似により求める以外に方法はない。

ＲｅＶｏｌｖｅｒ／Ｃ４０が採用しているシェー

ディング方法は depth gradient shading である。このシェーディン方法での法線の計算方法は各ピクセル毎にＺ値を用意し、スクリーンの水平垂直方向に隣接する７ピクセルのＺ値を用いて計算する方法である (図 12参照)。この方法は、画像生成処理と独立してシェーディング処理を行えるため、パイプライン化により高速化可能である。

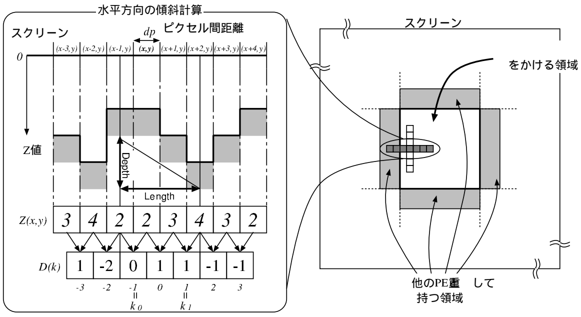

図 12: depth gradient shading

以下では水平方向のピクセルの傾斜の計算方法を示す。ただし、視点からスクリーンの距離を１として時のピクセルの間隔を dp、スクリーンからピクセル (x,y) までの距離を Z(x,y) で表すとする。

1. $k = 0, \pm1, \pm2, \pm3$ について、$D(k) = Z(x+k, y) - Z(x+k+1, y)$ を計算する。

2. $\{D(k) | k = 0, \pm1, \pm2, \pm3\}$ で D(0) を含み符号が異ならない最大の範囲を決定する。この範囲内での k の最大値、最小値を k_1、k_0とする。

3. $Depth = Z(k_1, y) - Z(k_0, y)$、$Length = |k_1 + 1 - k_0|$ を求める。

4. 傾斜角 $\theta_x = \arctan(\frac{Depth}{dp \times Length \times Z(x,y)})$

同様の処理をスクリーンの垂直方向でも行うことにより、ピクセル (x,y) の法線が求められる。

参考文献

[1] 對馬雄次 :ボリューム ·レンダリング専用並列計算機 — ReVolver/C40 — 修士論文 1994

[2] 藤原雅宏 :研究会資料 ReVolver グループ紹介と本年度の研究について 1997

[3] デレビジョン画像情報工学ハンドブック テレビジョン学会編 オーム社 1990

[4] テレビジョン ·画像情報工学ハンドブック 1990 オーム社

[5] 情報処理ハンドブック 1995 情報処理学会