コンピュータグラフィックスを用いた

高品質画像のリアルタイム生成に関する研究

2001年9月

大阪大学大学院　基礎工学研究科

向井信彦

論文要旨

近年におけるコンピュータの計算処理能力の向上とその低価格による急速な普及により，コンピュータグラフィックス（CG）の実用化が進み，産業界や医療業界では CG をキー技術とするバーチャルリアリティ（VR）を使用した様々な応用システムが開発されている．本論文は，今後も急速に発展し続けていくであろう CG 応用システムにおいて，開発システムの要求を満足する品質を備えた画像を，その目的を達成するのに充分な応答速度で生成する方式に関する研究の成果について論ずる．

最初に，CG の基本である直線を取上げ，量子化されたディジタル直線が備える性質について考察する．続いて，導出されたディジタル直線の性質を利用して，ディジタル画像の輪郭を追跡しながら，同時に直線近似を行う方式を提案する．また，ディジタル直線の性質を利用した簡易直線近似方式により，移動する物体の形状をリアルタイムに認識する方法に関する提案を行い，その結果について報告する．

次に，ディジタル直線の性質を利用し，直線を構成する画素の位置を画素単位で逐一判断することなく，高速に直線を描画する方式について論じた後，CG で一般的に利用されるポリラインを対象として，ポリラインを構成する端点の状態遷移を利用した高速なクリップ判定方式について論述する．

さらに，高品質表示と高速表示を両立させるために CG で扱う大規模なデータを階層化し，表示の目的に応じて，高品質表示と高速表示の優先度を切り替えて表示する方式について論ずる．特に，概略データを用いた高速表示から詳細データを用いた高品質表示へ切り替える際に表示速度を犠牲にすることなく，階層化されたデータを徐々に詳細表示化する方式について論述する．

最後に，CG をキー技術とする VR への応用例として，近年医療業界から特に要請の強い医療応用システムである手術シミュレータを取上げて，実用的なシステムを構築した結果について詳述する．本システムは，本論文の前半で記載した CG を用いた高品質画像のリアルタイム生成に関する研究成果を実システムに適用したものであり，手術練習のために必要な品質を備えた画像を，手術練習装置として充分な応答速度で生成する方式を実現している．以上の結果から，本論文により CG を用いた高品質画像のリアルタイム生成に関する研究の基礎を築くことができたと考える．

第一章　序論

1.1　研究の背景

近年における計算機処理能力の向上により，数年前までは映画やテレビドラマだけの夢の世界であったものが次々と実現している．現在では日本国民の二人に一人は持っているという携帯電話，いつでもどこでも気軽に電話して相手にメッセージを伝えることができるだけでなく，電子メールを使用することにより，音声を出すことなく相手との会話を楽しむことができる．また，デジタルカメラで撮影した映像を即座に送信したり，超小型 CCD カメラを備えることにより，相手の顔を見ながら会話を楽しんだりすることも可能となってきた．また，計算機を構成するハードウェアコンポーネントの小型化及び低価格化に伴い，ファミリーコンピュータ（ファミコン）やパーソナルコンピュータ（パソコン）が急速に普及してきている．特に，低価格なファミコンに搭載されるゲームが大流行し，CPU からアプリケーションに至るまで米国の独占市場であったコンピュータ分野において日本のオリジナル製品が世界を支配し始めている．最近ではファミコンをネットワークで接続し，友達同士が離れた場所に居ても一緒に楽しむことのできるネットワーク型対戦ゲームが流行しており，パソコン分野においても多くのマニアがゲームソフトを開発し，フリーウェアやシェアウェアと称して，無料あるいは非常に低価格で提供している．これらのソフトにはゲーム以外にも計算機の開発，診断，新たなソフト開発に利用できるツール類など数多くのものが含まれており，コンピュータはもはや一部の専門家のみが大型計算機室と呼ばれる隔離された部屋で扱うものではなく，ごく普通の人達が日常生活において必要とされるものに変化してきている．

また，医療の分野においてもコンピュータの導入は大きな恩恵をもたらしている．従来，診察を担当する医師のみが手書きで作成していたカルテも電子化され，一旦ファイルとして保存された電子カルテはいつでもどこでもどの医師でも参照することができる様になってきている．この結果として，診察を担当する医師と手術を行う医師の間でのカルテの取り違えミスなどが減少すると共に，米国ではごく一般的になっている**セカンドオピニオン**が可能となりつつある．つまり，診察を担当した医師の意見だけでなく，別の医師の意見を聞いて，治療や手術の方法を患者自身が決めることが可能となってきている．さらに，一般に**インフォームドコンセント**と呼ばれているものがある．これは，医師は患者に対して診断，治療方法などについて十分な説明をする必要があることを表現する新しい概念であり，患者は自分自身の病気について正しい知識を持ち，自らが治療方法を選択するものである．当然のことながら，医師側としても患者に対する病気の説明として，コンピュータを使用した分かりやすい説明が求められると共に，治療や手術に対する医師側の技量が求められる．通常は動物実験を通して薬の有効性を確かめたり，動物を利用した手術の練習を行ったりしている．しかしながら，動物愛護という観点から，この動物を利用した実験や手術練習は好ましくない．さらに，動物には発病しない人間特有の病気も数多く存在し，これらの病気に対しては練習の機会が皆無に等しいと言っても過言ではない．当然の結果として，実技訓練を一度も行うことなく，実患者を対象とした手術をせざるを得ない医師も少なくないことは事実である．

そこで，近年ではバーチャルリアリティ技術やコンピュータグラフィックス技術を利用して動物には発病しない人間特有の病気を仮想的に創造し，仮想空間上で手術の練習を行うことができるシミュレータ（**手術シミュレータ**）や，コンピュータの支援を受けながら実際の手術をより効率的に行うシステム（**手術支援システム**）の開発が行われている．バーチャルリアリティ技術を利用した仮想空間上におけるシミュレータは何も医療分野に限定されたことではない．以前から有名なものとして，

航空機の飛行訓練を行うフライトシミュレータ，自動車教習所で路上運転を行う前に体験するドライブシミュレータ，原子炉など危険な場所での実作業を効率よく実行できるように予め訓練しておくための各種訓練シミュレータなどが存在する．また，家庭や個人で使用するファミコンだけでなく，繁華街のアーケードに設置されているゲームなども仮想空間上での体験であり，若年層から壮年層に至るまで幅広い年代での娯楽となってきている．

これらの市場動向を中長期的に捉えると，バーチャルリアリティ技術やコンピュータグラフィックス技術を利用した応用製品，応用システムは今後も発展する傾向にある．しかも，ハードウェア技術の急速な進歩により，システムの低価格化，高速化がさらに一層推し進められると共に，自然現象の解析がさらに進展し，コンピュータグラフィックスにより生成される映像のさらなる高度化が期待される．コンピュータグラフィックスで生成した従来の映像は，一見するだけで人工生成物と判別することが可能であった．その一つの要因は，現実世界の事象をコンピュータの世界に取り込む際に生ずる量子化（ディジタル化）である．例えば，一本の直線を考えてみる．定規を当てて紙の上に描かれた直線は真っ直ぐに延びているが，コンピュータのディスプレイ上に描かれた直線はディスプレイの表示解像度に応じてギザギザしており，とても真っ直ぐとは言えない．また，紙の上に描かれた直線の傾きは常に一定であるにも関わらず，コンピュータのディプレイ上に描かれた直線の傾きは局所的に変化している．これらのディジタル化により生ずる不具合については，ディスプレイの解像度を向上させたり，直線のギザギザが目立たないような表示方式（**アンチエイリアシング**）を考案したりするなどで課題の解決を図ってきた．

映像生成において，もう一つ重要な課題が残されている．それは，映像の生成時間である．コンピュータを構成するハードウェアコンポーネントの急速な技術革新により，従来では数日から数週間かけて生成していたコンピュータグラフィックスによる映像生成時間は急速に短縮され，数分から数時間で生成できるようになってきたが，それでも映像生成時間の点において充分満足されているとは言い難い．特に，各種シミュレータやゲームの世界ではリアルタイム表示が必須であり，映像の品質が多少悪くてもリアルタイムで表示することが第一の課題となっている．人間が違和感を感ずることなく，自然な動画（アニメーション）を生成するためには，1 秒間に約 30 枚の画像を生成する必要があり，この場合，一枚の画像生成に与えられた時間はわずかに約 33 ミリ秒である．映像の品質向上のために自然現象を詳細に解析した表示アルゴリズムを開発し，このアルゴリズムを基に自然な画像をコンピュータグラフィックスにより生成することを試みても，映像生成速度が間に合わない場合は表示アルゴリズムや表示モデルデータの簡略化を行わなければならない．あるいは，表示アルゴリズムを開発する初期の段階で，簡略化された高速な表示方式を検討する必要が生じてくる．

1.2 論文の目的

本論文では上記のような技術的課題を背景とし，近年益々身近な存在となっているコンピュータグラフィックスを用いて，現在では各家庭にも一般に普及しているパーソナルコンピュータを使用し，高品質な映像をリアルタイムに生成する方式の検討を目的とする．

そのためには，まずコンピュータグラフィックスを用いた映像生成方法を修得する必要があり，先人が築き上げてきたコンピュータグラフィックスに関する技術の習得を図る．勿論，高品質な映像の生成，あるいは高速な表示方式には様々な方式があり，全てを網羅することはできない．そこで，コンピュータグラフィックスにおける基礎的な課題を取り上げ，その課題解決方法についての概要を

記述することにする．コンピュータグラフィックスの基礎的な課題と，その解決方法を理解していれば，さらに高度な映像生成方式を検討する上での基礎固めを行うことができる．

その後，コンピュータグラフィックスで生成した映像と自然界にある物体の映像を異にする一つの要因であるディジタル化について考える．コンピュータグラフィックスで映像を生成する以上，このディジタル化は避けることのできないものであるから，ディジタル化とは何なのか，理想的なディジタル化はどうすればよいのか，さらに理想的なディジタル化によって得られた映像はどのようになっているのかを調べることにより，コンピュータグラフィックスで生成した映像と自然界の物体映像を異質にしている本質を探ることにする．

ディジタル化の本質が理解できれば，次に，コンピュータグラフィックスを用いて映像を生成するための高速化方式を検討する．勿論，ハードウェア技術の進歩により，同じアルゴリズムを用いていても自然に高速な描画が可能となるが，ここでは，ディジタル化の本質を理解した上での高速処理について検討する．さらに，充分なハードウェアリソースがなく，高速な表示を行うことができない場合にも，表示の目的に応じて表示方式を変更することにより，ユーザが満足できる画質と速度を保つ映像を提供することができる．例えば，山の上から街を見たとき，街中に住んでいる人や道路を走っている車の詳細までも見える必要はない．大きな物体としての山や川，あるいは鉄道やビルディングなどが見えていれば充分である．一方，山から下って街中を散策する場合には，一軒一軒の家や出会う人々が詳細に描かれていなければならない．このような状況は他にも存在する．例えば，表示物体が動いているときと，静止しているときである．表示物体が動いているときには表示速度が重要であり，ユーザと対話的な処理を行う場合，リアルタイムな応答速度が要求される．一方，表示物体が静止している際には，ユーザは表示対象物の詳細を見ようとするであろうから，表示物体の細部を細かく描く必要がある．このように，表示の目的に応じて表示方式を変更し，高品質な映像生成とリアルタイムな表示速度の両立性について検討する．

最後に，これらの検討結果を実際の応用分野に適用して，その効果を検討する．コンピュータグラフィックスを用いた応用分野としては，工業や教育など様々な分野が存在するが，近年特に話題となっている医療分野に検討結果を適用し，応用システムを製作すると共に実使用を通じた評価を行う．その際，高度な計算能力を備える大型計算機やスーパーグラフィックスワークステーションなどを用いることなく，一般に普及しているパーソナルコンピュータを用いて，画像の品質が重要となる医療分野においても充分満足できる高品質な映像をリアルタイムに生成する方式の検討を行う．

1.3　論文の構成

本節では，論文の構成と概要について記述する．

初めに第二章では，今日までにコンピュータグラフィックスが発展してきた歴史を振り返ると共に，コンピュータグラフィックスの基礎的な課題とその解決方法について概説する．近年における計算機処理能力の急速な向上と共に，コンピュータグラフィックスの標準化，開発ツールの高度化により，コンピュータグラフィックスの基礎的な知識がなくても，ある程度のソフトウェアを製作することは可能になってきている．しかしながら，コンピュータグラフィックスの基礎を充分に理解しないでソフトウェアの開発を行っていると，思わぬ落とし穴に落ち入ったり，非効率な作業が多発して応答速度が極端に悪くなったりすることがある．また後の章でも述べるが，本研究の目的はコンピュータグラフィックスを用いた画像生成において，高品質な画像をリアルタイムに生成する手法の研究であるため，高速表示を維持したまま高品質な画像を生成する手法の考案には，まず，コンピュータ

グラフィックスの基礎的課題を解決しておく必要がある．そのため，本研究における手法を考える上で，最低限必要なコンピュータグラフィックスの基礎的課題とその解決方法について述べることにする．

第三章では，全ての描画における基本単位である直線について議論する．コンピュータグラフィックスでは表示物体を全て多角形（**ポリゴン**）の集まりとして定義するが，多角形を構成する各辺は線分で構成されているため，コンピュータグラフィックスで扱う直線がどのような性質を持つのかを最初に調べておく必要がある．自然界における真っ直ぐなアナログ直線を計算機の世界に取り込む際に行われる量子化規則を定め，この規則に従ってディジタル化された直線を発生する．発生したディジタル直線を構成する各画素の列を詳しく調べることにより，ディジタル直線の性質が明確となる．このディジタル直線の性質を利用すれば，２次元平面上に投影された物体の境界を直線で近似する際，高速な直線近似方式を考案することができる．物体の境界を高速に直線で近似することにより，物体の形状を高速に認識することが可能となる．そこで，ディジタル直線の性質を利用した高速直線近似が有効であることを確かめるために，簡単な形状を持つ二つの物体をリアルタイムで認識する実験を行ったので，結果について報告する．

第四章では，直線の高速描画方式について議論する．通常の直線描画方式では，直線を構成する各画素の位置を直線の量子化規則に従って逐一定めていく．この際，DDA(Digital Differential Analyzer)[1,2]など様々な高速描画方式が考案されているが，基本的には各画素の位置を逐一決定する方式であるため，高速描画には向かない．第三章で求められたディジタル直線の性質を利用することにより，直線を構成する各画素の位置を逐一決定するのではなく，直線を構成する画素の中で同一方向に並ぶ画素の個数を求め，画素列単位での一括描画を行うことにより，高速な直線描画が可能となる．また，コンピュータグラフィックスでは，物体を描画する枠あるいは描画するウィンドウ内のみを表示するという規則があるため，表示対象物の一部でもこの描画枠から外にはみ出している部分があれば，その部分を描画対象から取り除かなければならない．この処理は**クリップ**と呼ばれ，物体を描画する枠は**クリップ枠**と呼ばれる．基本的には，表示対象物の境界を構成している直線のどの部分がクリップ枠からはみ出しているのかを調べ，クリップ枠内に入っている部分のみを描画することになる．この際，直線とクリップ枠との交点計算が必要となり，多大な処理時間を要する．そこで，本章では**ポリライン**と呼ばれる連続した線分列（折れ線）を描画する場合に，現在対象となっている線分の終点が次に対象となる線分の始点になることを考慮して，この折れ線を構成する各点の２次元空間上における状態の遷移を考えることにより，最小限の演算量で高速にクリップ処理を実行する方法について述べる．

前章までは，コンピュータグラフィックスにおいて最も基本となる直線について議論してきたが，第五章では，大規模なグラフィックスデータを高速に表示する方式について議論する．近年におけるコンピュータ処理能力の向上により，複雑なグラフィックスデータも高速に表示できるようになってきたが，表示ハードウェアが高速になると，今度はコンピュータグラフィックスで生成される画像の質が問われるようになり，高品質な画像を生成するために，グラフィックスデータが大規模で複雑となってくる．当然のことながら，大規模で複雑なグラフィックスデータを単純な処理を高速に行うハードウェアだけで対応させるには限界が生ずるため，大規模なグラフィックスデータを効率よく表示させる手法が必要となってくる．特に，単一の計算機のみを用いてコンピュータグラフィックス映像を生成するだけではなく，インターネットなどの通信ネットワークを介してコンピュータグラフィックス映像を高速に表示させるためには，通信回線に応じた対策が必要となる．最も有効な手段として考えられているのは，大規模なグラフィックスデータを階層的に構成し，概略データから詳細

データへと順次送信することにより，受信側では表示物体の概略イメージから詳細イメージへと徐々に変化する画像を得る方式である．このように，概略データから詳細データへと徐々に変化する画像を表示する方法は，画像通信の世界では業界標準となっている JPEG Progressive 方式と思想を同じにするものであり，今後の画像表示の方向性を示している．インターネットなどでウェブブラウザを介して色々なサイトにアクセスする場合，受信側では興味ある映像が送られている場合は，最終イメージが到着するまで徐々に画質が向上していく映像を見ながらユーザは待つであろうし，興味のない映像が送られてくる場合には，すぐに別のサイトへ移動することが可能である．また，単一の計算機のみを用いてコンピュータグラフィックス映像を表示する場合でも，物体を回転したり視点を移動したりしている間は概略データを用いた高速表示を行い，物体が静止した段階で詳細データを用いて高品質画像を生成することは，コンピュータグラフィックス映像の効率的な表示に繋がる．第二章から第五章までで，コンピュータグラフィックスの基礎的課題とその解決方法から，コンピュータグラフィックスを用いた映像生成において最も基本となる直線の高速描画，さらに大規模で複雑なグラフィックスデータを階層化することにより，コンピュータの処理能力あるいは通信回線速度に応じた効率的な表示方式を検討する．

第六章では，コンピュータグラフィックスと共に今後益々発展していくと思われるバーチャルリアリティ技術について概説する．バーチャルリアリティ技術とは，仮想空間上に現実には存在しない仮想物体を表示するが，仮想空間の物は現実空間の物と同じように表示され，本質的には仮想空間を現実空間と同一であると感じさせる技術である．そのために，単なるコンピュータグラフィックスの３次元表示だけではなく，両眼視差映像を与えることによる立体視表示，あるいは仮想空間上の表示物体に触れたときに感じる力覚や触覚の提示などの技術がある．バーチャルリアリティ技術はフライトシミュレータやドライブシミュレータあるいはゲームなど，産業分野での応用が盛んに行われているが，今後重要と思われる分野として医療分野が存在する．医療は手術などの人命に関わる重要な技術，技量が要求される分野であり，バーチャルリアリティ技術を利用した応用システムが数多く研究開発されている．医療分野において，特に人命に関わる手術を対象としたとき，どのような応用システムが存在するのかについて概説する．

第七章では，バーチャルリアリティ技術を医療分野に適用したシステムの一例として，手術シミュレータを取り上げて議論する．特に，**マイクロサージェリー**と呼ばれ，非常に精緻な技術を要求される眼科手術について，具体的な症例を対象として手術シミュレータを構築する方法について論ずる．眼科を対象とした手術シミュレータを構築するためには，当然のことながら，工学知識だけでなく眼科に関する医学知識も必要となるため，眼科で行われている手術の種類と術式について概説した後，実際に構築したシミュレータシステムについて記述する．システム構築の中でも特に，前章までに議論してきたコンピュータグラフィックスに関する様々な知識を利用して実際の手術における映像と本質的には同一である高品質な手術映像をリアルタイムに生成する方式について検討し，手術の練習として充分満足できる画質を備えた映像をリアルタイムに生成するため方策について述べる．

第二章　コンピュータグラフィックス

2.1　コンピュータグラフィックスの歴史

21 世紀となった現在では，ゲームソフトを始めとした一般消費者市場だけでなく，フライトシミュレータやドライブシミュレータといった産業市場，さらには手術の支援や計画などの医療市場にもコンピュータグラフィックスの技術が当たり前のように利用されている．しかしながら，コンピュータグラフィックスの歴史はまだ 40 年にも満たないほど浅いのである．コンピュータグラフィックスの始まりは，1963 年に米国マサチューセッツ工科大学において Ivan E. Sutherland 博士がスケッチパッド（Sketchpad: A Man-Machine Graphical Communication System）を発表した時とされている[1,2]．これは，スタイラスペンで画面上にインタラクティブに線を描くシステムであり，現在における対話型グラフィックスの元祖である．勿論，スケッチパッドシステム以前にもコンピュータの画面上に図形を描くシステムは存在した．例えば，1950 年に MIT の Whirlwind I と呼ばれる計算機には簡単な図形を描画するシステムが取り付けられていたが，その方法はバッチ処理によるものであり，対話処理に適したものではなかった[1]．Sutherland 博士が発明したスケッチパッドシステムの登場により，人間とコンピュータがグラフィックスという媒体を通して対話することができるようになったのである．

その後のコンピュータグラフィックスは，CAD(Computer Aided Design)と共に発展していく[2]．CAD にも数多くの種類が存在するが，これらを大別すると，電気 CAD に代表される**２次元 CAD** と金型 CAD に代表される**３次元 CAD** になる．電気 CAD はまさにコンピュータなどの電子回路設計に利用されるものであり，各 LSI の配置及び結線を製図としてまとめるシステムである．電気 CAD が使用される以前は，紙の上に鉛筆と消しゴムを持って設計図を描いていた．製図という作業はかなりの時間と労力を要するだけでなく，修正が非常に困難である．一度製作した図面を基に製品の試作版を作成すると，必ずと言ってよいほど不具合個所が発見される．この不具合個所が発見されるたびに，設計図を修正するわけであるが，鉛筆と消しゴムを持って紙の上に描いた図面の修正は非常に困難であり，ごく一部の小さな個所における修正を除いて，この修正という作業は新たな製図とほぼ同等な作業となる．つまり，修正のたびに相当な時間と労力が製図に費やされるのである．ところが，コンピュータグラフィックスの登場により，そして CAD システムの開発により，製図の修正が非常に容易となったのである．ある LSI の配置を変更する場合，新たな位置をペンやマウスで指定するだけで，周辺回路を自動的に再配置し，各部品間の配線を自動的に描画してくれる．鉛筆と消しゴムを持って紙の上で図面を修正する場合，変更箇所の部品を消去し，新たに変更された個所に書き加えるだけでなく，部品の位置が変更になったことによる配線の変更には大幅な時間が必要となる．ところが，電気 CAD を用いれば，部品の位置変更と共に各部品を繋いでいる配線も自動的に変更してくれ，修正のための時間も労力も大幅に短縮することができる．部品配置個所の修正がごくわずかであれば，元の位置から新たな位置まで部品をマウスでインタラクティブに移動できるだけでなく，移動の途中でも部品間の結線を繋いだまま表示してくれる．これはまさに，Sutherland 博士が開発した対話型グラフィックスの元祖であるスケッチパッドシステムの賜物である．勿論，これら図面修正の自動化には，コンピュータグラフィックスだけでなく，AI(Artificial Intelligence)技術も大いに貢献している．

一方，３次元 CAD を代表するものとして金型 CAD がある．図面の製作という点では電気 CAD と同じ２次元図面の製作であるが，複数枚（通常は３枚）の図面を製作し，最終的な製品イメージを３次元で表示するという点が異なる．２次元平面上に３次元物体を投影して表示するためには，表示物体を

正確に表現するための様々な技術が必要となる．例えば，**隠線/隠面消去技術**がその一つである．隠線あるいは隠面消去技術とは文字通り，隠れた線あるいは面を消去し，物体の 3 次元的な把握を容易にする手法である．例えば，サイコロを例に取って考えてみる．図 2-1 はサイコロを 2 次元平面上に展開した図であり，紙の上にこのような図を書いておく．この図を切り取り，サイコロ面の境界部分を折り曲げて離れた境界をセロハンテープで接着すると，図 2-2 (b)のようなサイコロができ上がる．しかしながら，この(b)の図をコンピュータグラフィックスで描画することは，それほど単純ではない．コンピュータグラフィックスでサイコロを描かせる場合，まず，サイコロが６つの面から構成していることを記述する．そして，各面の組み立て時における位置，つまり，各面を構成する頂点の 3 次元座標値を知らせる．コンピュータグラフィックスで各面を順に描画すると，図 2-2 (a)のような図が描かれる．これは，サイコロを見たとき，どの面が見え，どの面が隠れているのかを考慮していないからである．物体をある視点から眺めたとき，見える部分と見えない部分が存在する．この見える部分と見えない部分を正確に計算し，見える部分のみを描画することにより，図 2-2 (b)のようにサイコロを 3 次元的に表現することができる．その詳細な方法については，次節で述べることにするが，Romney(1967)， Warnock(1969)， Watkins(1970)らの研究によるところが大きい[2]．

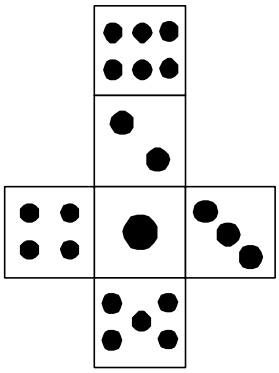


図 2-1 サイコロの展開図

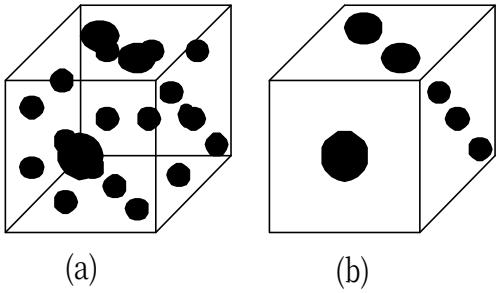


図 2-2 サイコロの立体図

また，別の課題としてクリッピングという処理が存在する．クリッピングとは，ある一定領域（通常は２次元空間における矩形領域あるいは３次元空間における視点を頂点とした四角錐）に入る物体のみを表示するという処理である．逆に言えば，指定された領域以外の物体は表示しないということになる．コンピュータグラフィックスで扱うデータ量は，数オブジェクトという小規模なものから，工場や街，あるいは国や地球，さらには宇宙全体という大規模なものまで様々である．例えば，電子回路設計を考えた場合，LSI を構成する回路図には様々なゲート部品が存在し，その間に無数の配線が施されている．勿論，LSI 全体を表示することもあるが，全体を表示した場合は全体的な配置を把握するのみであり，LSI 回路の詳細を掴むことは困難である．LSI 回路の詳細を見る場合には，LSI 回路を構成している一部の領域のみを切り出して，注目している領域のみを拡大表示することにより，LSI 回路の一部を詳細に検討することが可能となる．この拡大表示において，注目すべき領域を切り出し，その注目している領域のみを表示する処理がクリッピングである．クリッピングについては，Sutherland(1968)，Weiler(1977)らによって多くの研究[3]がなされてきたが，本論文でも第四章において従来の方法を詳しく説明した後，さらなる高速化についての検討を行う．

その他，自由曲面生成に関する研究（Coons，Reisenfield，Bezier ら）や陰影表示手法に関する研究（Gouraud，Phong，Whitted ら），さらに半透明表示方法（Newell-Shancha ら），材質感表現（Catmull ら），照明モデル（中前ら）など今日までに数多くの研究がなされている[3]．

2.2　コンピュータグラフィックスの基礎課題と解決策

本節では，今日までに築き上げられてきたコンピュータグラフィックス技術の中でも，特に基礎的な課題とその解決方法について概説することにする．但し，基礎的な技術の中でも直線の描画及びクリッピング処理についてはより詳しい研究を行ったので，第三章及び第四章で論ずることにする．

2.2.1　座標系と座標変換

2.2.1.1　コンピュータグラフィックスで扱う座標系

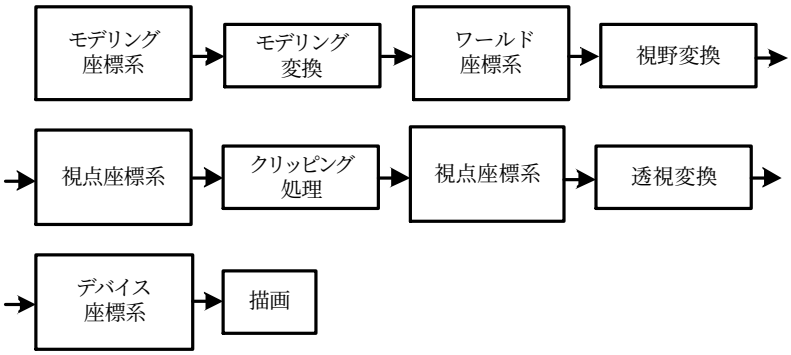


図 2-3 座標系と変換処理の流れ

図 2-3 に，コンピュータグラフィックスを用いて画像を生成する際の座標系と変換処理の一般的な流れを示す[2,4]．コンピュータグラフィックスで表示する物体は全て**ワールド座標系**という一つの大きな空間で定義されるが，最初から全ての物体をこの座標系上で定義したのでは表示処理の扱いが困難となる．何故なら，ワールド座標系で定義した個々の物体を独立に動かす必要性が多々存在するが，全ての物体をワールド座標系でのみ定義したのでは，個々の物体を独立に動かすことが困難となるからである．表示物体の移動，変形，及びワールド座標系で定義された空間を見る視点位置の移動は全てマトリックス演算で実現可能であるが，このマトリックス演算については後述することにし，ここではまず，**モデリング座標系**について説明する．表示物体を最初からワールド座標系という統一された座標空間上で定義するのではなく，表示物体固有の座標系で定義する．例えば，図 2-2 で定義したサイコロのモデリング座標系は図 2-4 のようになる．

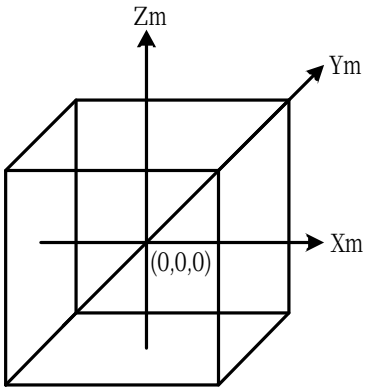


図 2-4 モデリング座標系

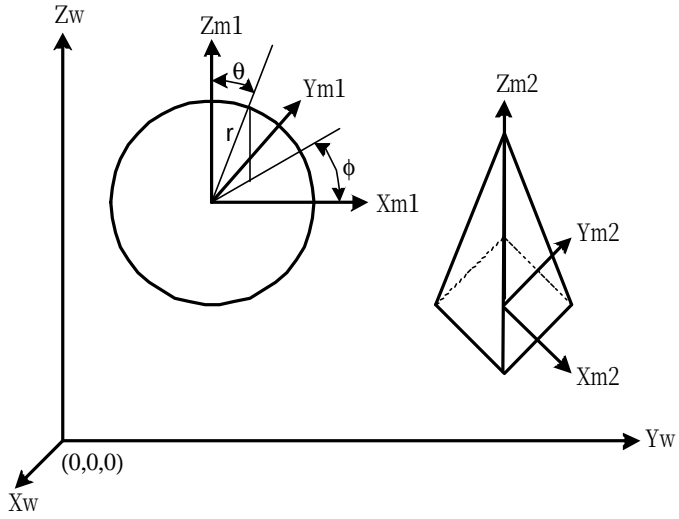


図 2-5 ワールド座標系

図 2-4 では，サイコロの中心を原点とし，右横方法にX軸(Xm:[注1])，右奥方向にY軸，上方向にZ軸を取っている．こうすると，単位立方体（一辺の長さが１の立方体）を構成する頂点座標は，(-0.5,-0.5,-0.5)， (0.5,-0.5,-0.5)， (0.5,0.5,-0.5)， (-0.5,0.5,-0.5)， (-0.5,-0.5,0.5)，(0.5,-0.5,0.5)，(0.5,0.5,0.5)，(-0.5,0.5,0.5)と表現することができ，全ての座標値を絶対値0.5の範囲内に収めることが可能となる．勿論，図2-4のように，必ずしも立方体の中心に原点を取る必要はない．例えば，左下手前の角(-0.5,-0.5,-0.5)を原点としてモデリング座標系を定義してもよいし，図2-4で示すような直行座標系ではなく，円筒座標系，あるいは極座標系など，個々の表示物体の特徴を生かした自由な座標系で定義することが可能である．

上記のように，個々の物体に固有のモデリング座標系上で定義された複数の表示物体をワールド座標系という統一された座標空間上に移動させる．図2-5では，球と四角錐という二つの表示物体をワールド座標系上に並べた例を示しているが，この場合，球は中心を原点とした極座標で，四角錐は底面の中心を原点とした直行座標でモデリング座標系が定義されている．但し，ワールド座標系は通常，直交座標を用いて定義することになっている．コンピュータグラフィックスではこのように，個々のモデリング座標系で自由に定義された表示物体を全て統一されたワールド座標系上に配置し，グラフィックス空間，あるいは仮想的な３次元世界を創造する．この世界は，我々が生存する３次元空間と同等である．そして，我々が３次元空間上を自由に移動し，好きな位置及び角度から自然の風景，あるいは車や建物などの人工物を写真に取ることができるように，コンピュータグラフィックスでも，任意の位置及び角度から広角や望遠といった任意のレンズを用いてグラフィックス空間を２次元平面上へ投影して描画することができる．このためには，**視点座標系**という新たな座標系を導入する必要がある．視点座標系とは，グラフィックス空間上でのカメラの位置（視点）を原点とし，見たい方向（被写体が存在する方向）に広がっている座標系である．図2-6にワールド座標系と視点座標系の関係を示す．

[注1]モデリング座標系(Modelling Coordinate)には m を添えることにより，他の座標系と区別することにする。例えば，ワールド座標系(World Coordinate)はw，視点座標系(Viewing Coordinate)はv，デバイス座標系(Device Coordinate)はd を添えることにする。

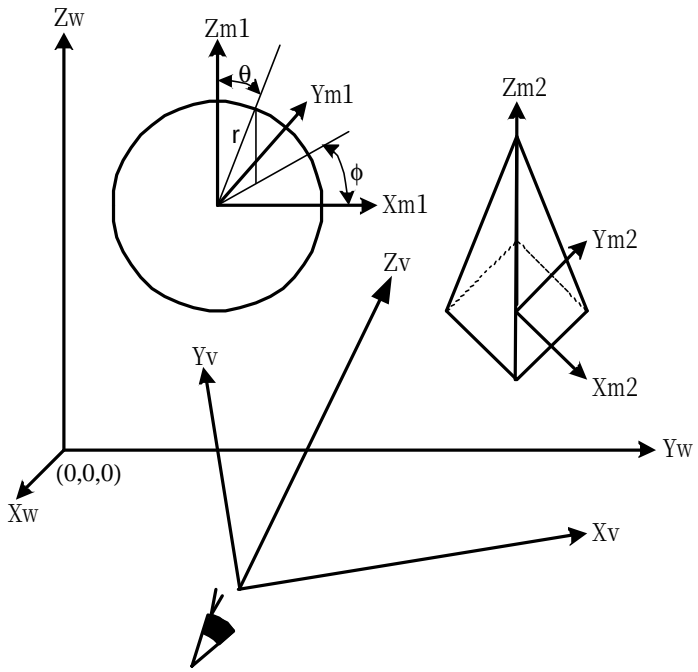


図 2-6 ワールド座標系と視点座標系の関係

図に示すように，視点座標系の原点（視点位置）は，ワールド座標系上の任意の位置で定義することができる．また，視点座標系のＺ軸方向（視線方向）により，見る方向（視野方向）が決定される．なお，図 2-6 からでは少し理解しずらいが，視点座標系は通常，右横方向にＸ軸，上方向にＹ軸，奥行き方向にＺ軸を取ることになっている．従って，視点座標系は通常，左手系座標系として定義される．これに対して，モデリング座標系及びワールド座標系は通常，右手系座標系として定義される．視点座標系において，視点位置から見た映像をある投影面（通常は XY 平面に平行な面）上に投影した像が最終画像として表示される．この最終画像が表示される空間は２次元平面であり，コンピュータの性能，特にビデオボード（あるいはグラフィックスボード）に搭載されている表示メモリなどのデバイスに依存するため，デバイス座標系と呼ばれる．図 2-7 にデバイス座標系を示す．

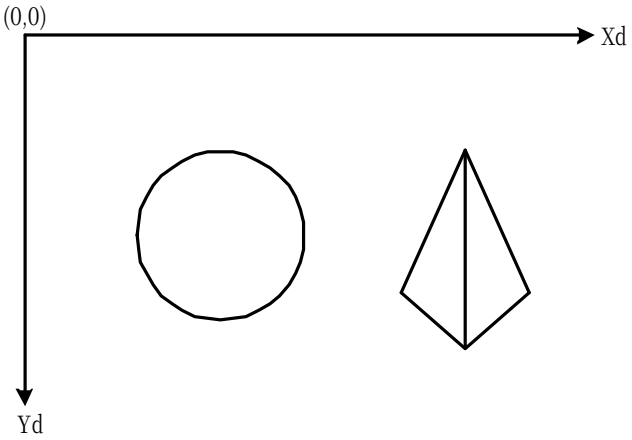


図 2-7 デバイス座標系に表示された物体

表示投影面に投影され，デバイス座標系上に表示された映像は多くの画素から構成されており，画像と同様に扱われる．通常，画像処理分野では左上を原点し，右方向にX軸，下方向にY軸を取る２次元座標系が用いられるため，３次元コンピュータグラフィックスでもデバイス座標系は画像処理における座標系に従うこと場合が多い．

以上で，基本的な座標系の流れ（モデリング座標→ワールド座標→視点座標→デバイス座標）を説明した．次に，各座標系間の繋ぎ役としての処理を行う座標変換について概説する．

2.2.1.2 表示処理に必要な座標変換

３次元グラフィックスで扱う座標はＸＹＺの三つであるが，平行移動や透視変換（視点からの距離に応じて表示物体の大きさが小さくなる変換）を考えると，次元を一つ増やして，４次元同次座標として扱うと便利である．**モデリング変換**とは，モデリング座標系上で定義された表示物体をワールド座標系上に配置するための変換処理であり，表示物体の拡大縮小，回転，せん断及び平行移動という四つの変換処理の組み合わせとして定義される．

$$(X_m, Y_m, Z_m, 1)M_m = (X_w, Y_w, Z_w, 1) \quad \cdots\cdots(2-1)$$
$$M_m = M_e M_r M_s M_t$$

但し，

M_m : モデリング変換マトリックス

M_e : スケール変換マトリックス（X,Y及びZ座標の拡大縮小）

M_r : 回転マトリックス（X,Y及びZ軸周りの回転）

M_s : せん断マトリックス（固定点からの距離に応じた歪を生じさせる変形）

M_t : 平行移動マトリックス（X,Y及びZ方向の平行移動）

$$M_e = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad M_s = \begin{pmatrix} 1 & c & e & 0 \\ a & 1 & f & 0 \\ b & d & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad M_t = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{pmatrix} \quad \cdots\cdots(2-2)$$

$$M_r = R_x R_y R_z \quad \cdots\cdots(2-3)$$

R_x : X軸周りの回転マトリックス

R_y : Y軸周りの回転マトリックス

R_z : Z軸周りの回転マトリックス

$$R_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos q_x & \sin q_x & 0 \\ 0 & -\sin q_x & \cos q_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad R_y = \begin{pmatrix} \cos q_y & 0 & -\sin q_y & 0 \\ 0 & 1 & 0 & 0 \\ \sin q_y & 0 & \cos q_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_x = \begin{pmatrix} \cos q_z & \sin q_z & 0 & 0 \\ -\sin q_z & \cos q_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{.....(2-4)}$$

上記マトリックスによりモデリング変換を実行し，モデリング座標系上で定義した表示物体をワールド座標系上の任意の位置に任意の大きさ及び形に変形させて表示することが可能である．次に，**視野変換**について述べる．視野変換とはワールド座標系から視点座標系への変換であり，ワールド座標系上の任意の位置に視点を移動し,視線方向をＺ軸の正方向として左手座標系を定義することと等価である．通常，ワールド座標系，視点座標系とも直行座標系で定義し，スケールの変化や空間的な歪みが存在しないため，基本的には回転と平行移動のマトリックス（M_r及びM_t）で構成される．但し，**2.2.1.1**にも記述したように，ワールド座標系は通常右手座標系として定義されるのに対し，視点座標系は通常左手座標系として定義される．従って，右手座標系から左手座標系への変換マトリックスが必要となるが，これはＺ軸の向きを入れ替えるだけの簡単なマトリックスである．最終的に，視野変換マトリックスは次のように記述することができる．

$$M_v = M_r M_t M_i \quad \text{.....(2-5)}$$

M_v : **視野変換マトリックス**

M_i : **座標系変換マトリックス（*Z*軸の方向を**180**度反転）**

コンピュータグラフィックスを用いた画像生成における最後の変換処理として，透視変換が存在する．透視変換とは，視点座標系上で捕らえた表示物体の集まり（シーン）をある２次元平面上へ投影する変換処理である．勿論，２次元平面上へ投影する際に，視点座標空間内に存在する物体を投影面上に平行に投影する方法（**平行投影法**）も存在する．平行投影の場合，視点座標系のＺ軸が視線方向と一致している限り，Ｚ座標を無視し，Ｘ，Ｙ座標のみを有効にして描画すればよいため，平行投影のマトリックスは極めて単純であり，Ｘ，Ｙ座標に対応する部分を１に，他の部分を０にすれば生成できる．しかしながら，通常，カメラなどで風景写真を撮影した場合，手前にある物が大きく，遠方にある物ほど小さく写っており，この被写体の大きさにより遠近感が得られている．従って，コンピュータグラフィックスを用いて画像を生成する際にも，手前にある物ほど大きく，遠方にある物ほど小さく描画して遠近感を出す手法が要求される．

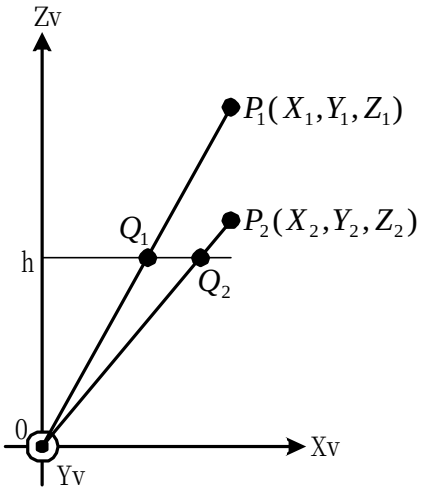


図 2-8 透視変換の原理

図 2-8 は，視点座標系をＹ軸上空からＹ軸の負の向きに眺めた様子を示している．図中， $P_1(X_1,Y_1,Z_1)$ **及び** $P_2(X_2,Y_2,Z_2)$ **という２つの点を表示しているが，２点の(X, Y)座標値は同一であり，Ｚ座標値のみ異なるとする．また，投影面をZ=hで表されるＸＹ平面に平行な平面とする．こうすると，２点(** P_1 **及び** P_2 **)を投影面(Z=h)上へ投影して得られる座標値(** Q_1 **及び** Q_2 **)は，三角形の相似形原理から，次の様にして得られる．**

$$Q_1(hX_1/Z_1, hY_1/Z_1, h) \qquad Q_2(hX_2/Z_2, hY_2/Z_2, h) \quad \cdots\cdots(2-6)$$

ここで， $X_1 = X_2$ **及び** $Y_1 = Y_2$ **であり，かつ，** $Z_1 > Z_2$ **であることを考慮すると，投影面上において，** Q_2 **の方が** Q_1 **よりも大きなＸ及びＹ座標値を持つことが分かる．つまり，近くにあるもの(** P_2 **)ほど，遠くにあるもの(** P_1 **)よりも大きく表示され，我々が常日頃から感じている遠近法の原理と一致する．ここで，透視変換マトリックスは次のように記述することができる．**

$$(X^*,Y^*,Z^*,1)(1/W^*)M_u = (X_d,Y_d,X_d,1)$$
$$(X_v,Y_v,Z_v,1)M_pM_j = (X^*,Y^*,Z^*,W^*) \quad \cdots\cdots(2-7)$$

$$M_u = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad M_p = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1/h \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad M_j = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \cdots\cdots(2-8)$$

M_u **:単位マトリックス**

M_p **:透視変換マトリックス**

M_j **:座標系変換マトリックス（Ｙ軸の方向を180度反転）**

以上で，コンピュータグラフィックスを用いて描画処理を行う上で，最低限必要な座標系と座標変換について述べてきた．なお，図 2-3 における視点座標系上で行われるクリップ処理については，本論文の第四章で詳しく議論することにする．

2.2.2 隠面消去技術

コンピュータグラフィックスが発展してきた歴史の中で，もう一つ重要な基礎的課題がある．それは，表示対象物体の中で，見えない部分を如何にして隠すのかという問題（隠面消去問題）である．今日のコンピュータ技術の発展，特にハードウェアの大量生産による低価格化のお陰で，今日ではハードウェア資源を充分活用した方法が主流となっているが，歴史的には様々な方法が存在し，コンピュータグラフィックスの原理を理解する上で，それらの方法を知っておくことは重要である．ここでは，歴史的に見て代表的と思われる四つの方法について概説する．

2.2.2.1 優先順位法

世の中に存在する風景や物体を写真で撮影する場合，隠れて見えない部分には必ず，その物体の前に別の物体が存在している．つまり，視点から遠い距離に存在する物体は，視点とその物体との間に別の物体が存在し，この物体によって視点が遮られるために，隠れて見えなくなるのである．従って，表示対象物体を視点からの距離に応じて順番に並べ，遠い物体から順に描いていけば，最終的に手前にあって他の物体に遮られない物体（あるいは物体の一部）が残り，隠面消去問題が解決するという考えに基づく手法が**優先順位法**である．

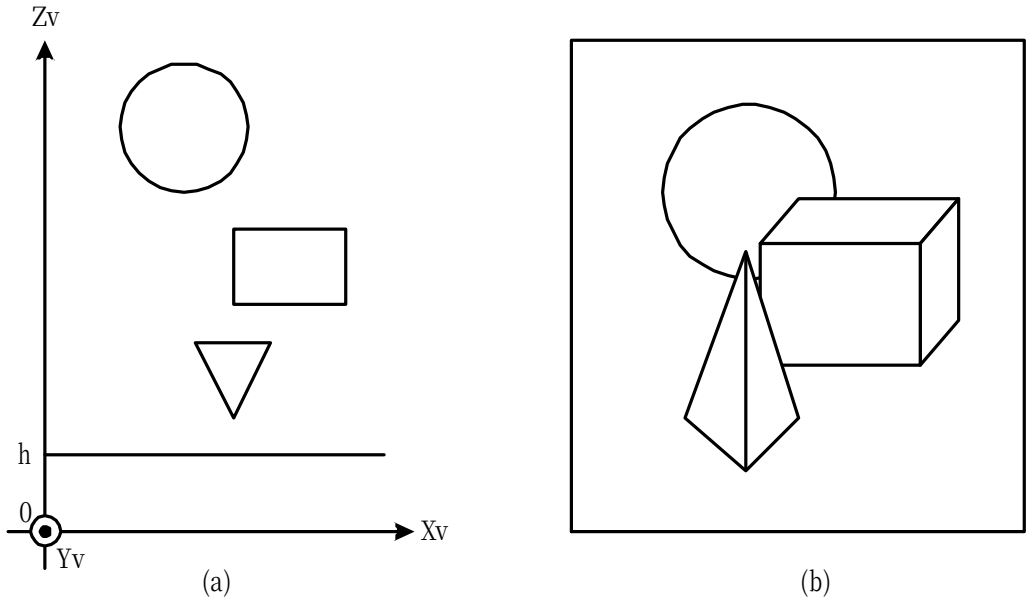


図 2-9 優先順位法による隠面消去処理

図 2-9 に，優先順位法により隠面消去処理問題が解決できる簡単な例を示す．図中(a)は視点座標系のＹ軸上空からＹ軸の負の向きに眺めた風景であり，球，直方体，三角錐の３つの物体が存在する．これらの物体を Z=h という投影面上に投影して描いた図形が(b)である．視点から最も遠い距離に存在する球は，立方体及び三角錐により物体の一部が隠れて見えなくなっている．直方体は球と三角錐との中間に位置するため，球の一部を隠している一方で，三角錐により直方体の一部も隠されて

いる．三角錐は全ての物体の中で最も視点に近い距離にあるため，どの物体にも隠されることなく，全てが見えており，隠面消去問題が解決されている．勿論，各表示物体は複数の面（三角錐は４面，直方体は６面，球は分割数に依存した面数）から構成されており，それぞれの面単位に優先順位を決めて描画する必要がある．こうすることにより，最も手前にある三角錐でも視点方向を向いている面は他の面により隠されることはないが，視点と逆方向を向いている面は他の面により隠されて最終的には表示されない．表示対象物体の優先順位が簡単に決定できる場合は，単純に視点から遠くにある物体（あるいは面）から順に描画すればよく，非常に高速な表示方法である．しかしながら，表示物体の数や物体を構成する面数が増加するにつれ，この優先順位が簡単に求められなくなるという欠点が存在する．

2.2.2.2 スキャンライン法

優先順位法は，表示物体を視点から遠くにあるものより順位をつけ，この順番に描画することにより隠面消去問題を解決するという方法であった．しかしながら，物体数や物体を構成する面の数が増加するにつれ，優先順位の決定が困難となる．一方，CRT や LCD などの表示デバイスは，**走査線**（あるいは**スキャンライン**）と呼ばれるラインにより，構成されている．図 2-10 に表示デバイスの走査線を示す．走査線は通常，左上を原点とし，右方向にX軸，下方向にY軸を取るデバイス座標系において，Xの正方向に走査するラインであり，左から右へ，上から下へと走査を続ける．右から左へ，下から上へと線が戻る帰線時間は表示されない．スキャンライン法とは，このスキャンライン単位で隠面消去問題を解決しようという方法である．つまり，表示対象物体を**スキャンライン面**（スキャンラインを含み，ＸＹ平面に垂直な平面）で切断した際の切り口から，視点に最も近い線分列を求めようという方法である．優先順位法が表示物体単位，あるいは表示物体を構成している面単位で隠面消去問題を解決しようとしていたのに対し，スキャンライン法では対象とするスキャンライン上に存在する物体の**線**のみを問題にすればよく，隠面消去問題がより単純化されている．

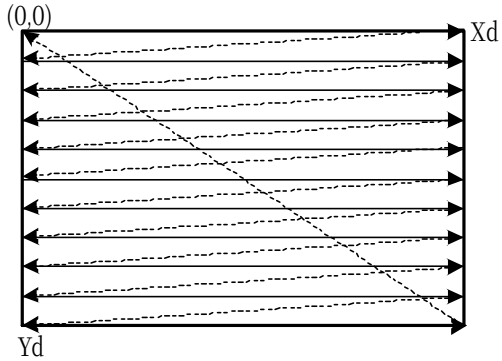


図 2-10 表示デバイスの走査線

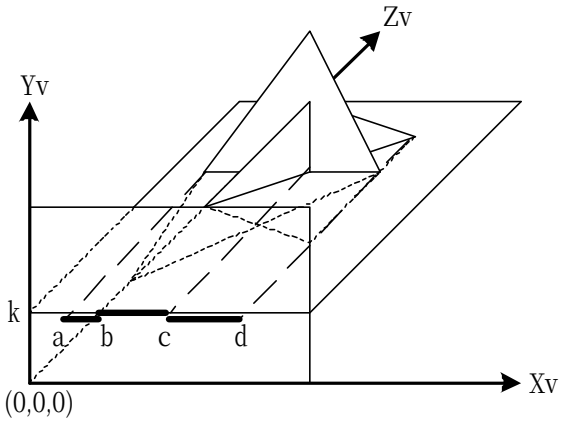


図 2-11 スキャンライン法

図 2-11 にスキャンライン法の原理を示す．表示物体として，少し大きめの三角形とこれよりも小さな四角形を対象とする．また，説明の単純化のために，透視変換ではなく平行投影を考える．今，Z=0 という投影面に表示対象物である三角形と四角形を平行投影し，Y=k というスキャンラインのみ

を対象として考えると，それぞれの表示物体に対応した線分が得られる．図において，三角形は線分 ad，四角形は線分 bd が得られ，点 c において両者が交差している．これらの線分を分割し，線分を構成する端点のＺ値を比較することにより三角形は線分 ab 及び線分 cd が，また，四角形は線分 bc が視点に最も近い線分として判断され，隠面消去問題が解決される．線分の端点の比較だけでは視点からの距離の判定が不可能な場合，線分をさらに分割して判定することになる．このように，スキャンライン毎に隠線消去問題を解決し，最終的に隠面消去問題を解決する方法がスキャンライン法である．

2.2.2.3　Ｚバッファ法

優先順位法は表示物体あるいは表示物体を構成している面単位に，また，スキャンライン法は投影面の走査線上で表示対象物を構成する線単位に，隠面消去問題を解決してきた．Ｚバッファ法は，さらに細かく，表示対象物を構成している画素単位に隠面消去問題を解決する方法である．

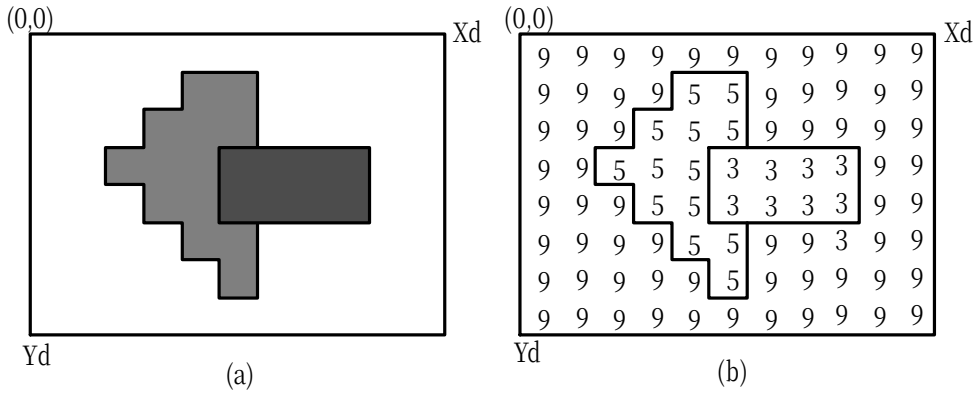


図 2-12　Ｚバッファ法の原理

図 2-12 にＺバッファ法の原理図を示す．Ｚバッファ法とは，表示対象物を描画する**フレームバッファ**以外に，対象物の奥行き情報を格納する**Ｚバッファ**という２種類のバッファを用いて隠面消去問題を解決する方法である．最初に，フレームバッファを背景色（図 2-12 では無色）で，また，Ｚバッファを無限遠点の距離（図 2-12 では９）で初期化する．この方法では優先順位法のように，表示対象物を奥行きから順に並べる必要はない．対象となる物体を無作為に選んで，物体を構成している画素単位に，物体の画素の色をフレームバッファに描画すると共に，画素の奥行き情報（Ｚ値）をＺバッファに書き込む．但し，このフレームバッファ及びＺバッファへの書き込みは無条件に行うのではなく，書き込み時に条件判定を行う．つまり，これから書き込もうとする画素のＺ値とその画素に対応するＺバッファの位置に既に書き込まれている値とを比較し，これから書き込もうとする画素のＺ値が既に書き込まれているＺ値よりも小さい（つまり，視点に近い）場合にのみ，フレームバッファ及びＺバッファへの書き込みを行い，これから書き込みをするＺ値が既に書き込まれているＺ値よりも大きい場合は，フレームバッファ及びＺバッファの双方とも書き込みを行わない．書き込みをするＺ値と既に書き込まれているＺ値とが等しい場合は，デバイス依存となる．図 2-12 では表示対象物の色として，三角形を斜線で，四角形をクロスパターンで表している．また，三角形の距離

情報（Ｚ値）を５，四角形の距離情報を３としている．勿論，図形が投影面に対して斜めになっている場合は，距離情報は単一物体内でも変化するが，ここでは説明を単純化するため，図形はＸＹ平面に平行であり，物体内のＺ値は同一であると仮定している．二つの表示対象物である三角形と四角形を，各物体を構成している画素単位にＺバッファとの比較を行いながら，フレームバッファ及びＺバッファへの書き込みを行った結果が図 2-12(a)及び(b)である．図では，四角形の一部と三角形の一部が重なっているが，四角形のＺ値（３）が三角形のＺ値（５）よりも小さいため，四角形のＺ値が優先され，四角形の一部が三角形の一部を隠す結果を示している．

2.2.2.4 光線追跡法

上記で説明した手法は全て，表示物体あるいは表示物体を構成している面，線，点を単位として隠面消去問題を解決する手法であったが，ここで述べる**光線追跡法**は発想が少し異なる．元々，カメラで撮影した映像というのは，カメラに入射してくる光線により結像されるものである．従って，このカメラに入射する光線を逆に辿ることにより，映像画面を構成する各画素に写っている物体と，その色を調べようというものである．光線追跡法は視線を逆に辿ることから，**視線逆探索法**とも呼ばれる．

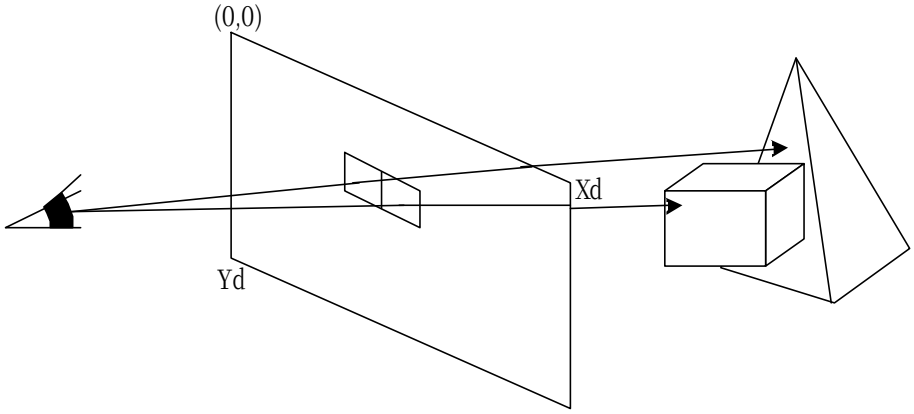


図 2-13 光線追跡法の原理

図 2-13 に光線追跡法の原理図を示す．光線追跡法とは，視点に入ってくる光線を逆に辿ることにより，視点に最も近い物体を判別する手法であるため，まず，視点と映像画面を構成する各画素とを結ぶ直線を考え，この直線と表示物体（あるいは表示物体を構成している面）との交点を調べる．そして，この表示物体との交点の距離を比較し，最も小さい距離を与える物体が表示されるという考えに基づく方法である．原理は非常に単純であるだけでなく，光線を辿る際に光線が鏡などにぶつかった場合には，鏡などからの２次反射も考慮されている．つまり，同じアルゴリズムを再帰的に用いることにより，高次の反射特性をも考慮することができる．また，反射時におけるエネルギーの減少や，光線が別の物体に遮られて光源まで辿り着けない場合には，影を生成する処理も同一のアルゴリズムで実現可能である．このように，光線追跡法は映像が生成される基となる光線を逆に辿ることにより，隠面消去問題を解決するだけでなく，半透明表示，付影処理など，高品質画像を生成する上で非常に有効な手法である．しかしながら，ただ一つの欠点が存在する．それは，非常に膨大な計算コ

ストを要するということである．例えば，1,000×1,000 の画面上に一体が 1,000 ポリゴン（多角形）で構成されている表示物体を 100 体表示することを考える．視点と画面上の各画素とを結ぶ光線の数は 100 万本存在し，この 100 万本の光線と 1,000 ポリゴンとの交点計算を行う必要があるため，表示物体 1 体につき必要な交点計算は 10 億回となる．さらに，表示物体は 100 体存在するから，全ての表示物体との交点計算は 1,000 億回となり，1,000 億回の計算の中から視点に最も近い物体との交点を調べる必要がある．さらに，1,000 億回の交点計算と 1,000 億回の交点距離比較を行っただけでは，視点から最も近い物体の交点を調べただけであり，鏡などからの 2 次反射や，影処理，半透明処理などを行うためにはさらなる時間を要することになる．

一方，光線追跡法は非常に並列化し易い手法であり，各画素単位で独立の処理が行えるため，多くの並列化手法が検討されてきた．また，現在におけるコンピュータ計算能力の向上や，コンピュータグラフィックス描画能力の向上により，かなりの高速化が実現できているが，現状では毎秒 30 枚あるいは 60 枚という画像を生成する**リアルタイム表示**には適していない．現在におけるコンピュータグラフィックスは，大量生産によるハードウェアコストの低価格化に伴い，安価で高速なメモリを大量に利用できるＺバッファ法が主流となっており，Ｚバッファ法を用いたグラフィックスボードが広く利用されている．ただ，Ｚバッファ法は隠面消去問題を解決しただけであり，付影処理など高品質画像を生成する上での課題も多い．そこで本研究では，コンピュータグラフィックスを利用した映像生成方式としては，現在主流となっているＺバッファ方式を採用して高速な映像生成を行うが，このＺバッファ方式を採用するだけでは解決できなかった高品質画像の生成という課題を解決することにより，コンピュータグラフィックスを用いた高品質な画像のリアルタイム生成に関する手法について論ずることにする．

第三章　ディジタル直線

3.1　アナログ直線の量子化

コンピュータグラフィックスとは，与えられた数値データから所望の画像を生成する技術であり，この画像生成において最も基本となるのは直線の描画である．これは丁度，絵画の世界における線描画を基本としたデッサンの重要性に類似している．我々が通常，紙の上に鉛筆や筆で描く直線とコンピュータのディスプレイ上にディジタル化された直線とは少し性質が異なる．従って，最初にコンピュータグラフィックスを扱う上で最も基本となるディジタル化された直線の性質を詳しく調べることにする．ディジタル化された直線の性質を調べるためには，その前に，紙の上に鉛筆などを使用して描画する直線をコンピュータ内で取り扱えるように変換する量子化規則について定義しておく必要がある．なお，本論文では線幅を持たない真っ直ぐな直線を**アナログ直線**と呼び，これに対してコンピュータ内で取り扱われる量子化された直線を**ディジタル直線**と呼ぶことにする．言い換えると，アナログ直線とは直線の始点及び終点を与えることにより一意に定まる直線であるのに対し，ディジタル直線とは直線の始点及び終点を与えただけでは直線を構成する始点から終点に至る画素列を一意に定めることができず，アナログ直線をディジタル化する際の量子化規則を定めることにより初めて，直線を構成する始点から終点に至る画素列を一意に定めることができる直線である．

二次元平面上の線幅を持たない真っ直ぐなアナログ直線を量子化して，コンピュータ内で取り扱うことのできるディジタル直線を生成すると，ディジタル直線を構成する各画素間には繋がりがあり，この画素の繋がりの方向を記述するのにチェイン符号を用いると便利である．チェイン符号[5-8]とは，画素の連なりを符号化したもので，その基本となる方向及び符号は図 3-1 に示すように 8 の剰余系により定められる．図 3-1 において点 P を注目する画素とすると，右方向から順に 0，1，…，7 の符号を割当てていく．例えば 0010000100 は，図 3-2 で示すディジタル直線を構成する画素列の繋がりをチェイン符号を用いて符号化したものである．

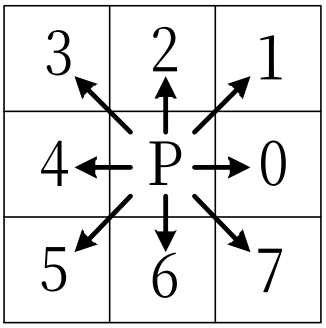


図 3-1 チェイン符号

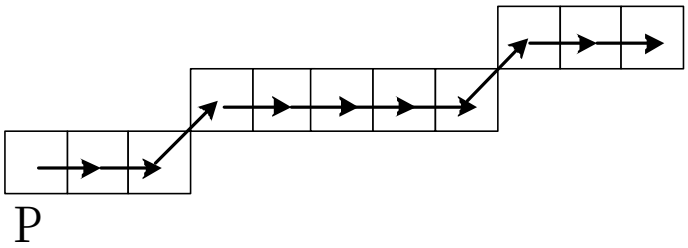


図 3-2 ディジタル直線

次に，ディジタル直線を一意に定めるためのアナログ直線の量子化について考える．点 P を始点とする直線は点 P から任意の方向へ延びる無数の直線が定義可能であるが，直線の性質を調べる上で直線の対称性を考慮して，ある限定された範囲内の直線のみを対象としても一般性を失うことはない．点 P を原点とし，そこから様々な方向に延びる様々な直線を図形の対称性を考慮して分類する

と，図 3-3 のように分類することができる．そこで，図 3-3 のA領域（直線の傾きが 0 度から 45 度の範囲内に入る第１象限の直線）について考え，量子化規則を次のように規定する(図 3-4 参照)．

＜直線の量子化規則＞

1) X座標値に関しては整数値のみを取るように量子化を行い，i 番目の量子化されたX座標値を X_i とする．
2) X_i に対応するアナログ直線上の点のY座標値を Y_a とし，Y_a の小数点以下に関しては Y_a の絶対値を四捨五入して Y_a と同符号与えることにより，i 番目の量子化されたY座標値 Y_i を定める．

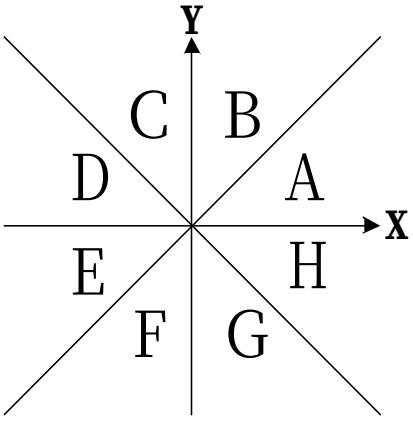


図 3-3 直線の分類

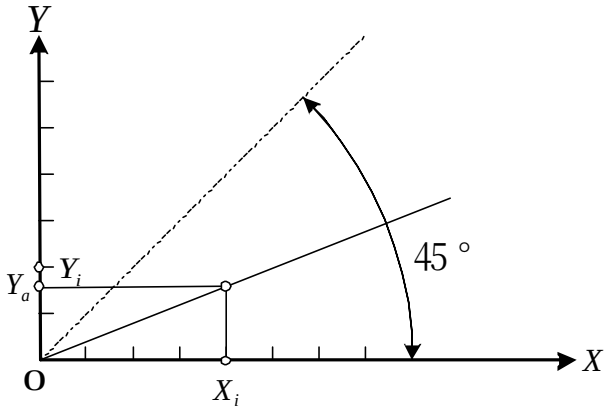


図 3-4 アナログ直線の量子化

ここで，上記量子化規則を図形の対称性を考慮して，点Pから様々な方向に延びる直線へと一般化すると，図 3-3 の各領域に対しての量子化規則は次の様になる．

＜領域A，D，E，Hに対する量子化規則＞

1) X座標値に関しては整数値のみを取るように量子化を行い，i 番目の量子化されたX座標値を X_i とする．
2) X_i に対応するアナログ直線上の点のY座標値を Y_a とし，Y_a の小数点以下に関しては Y_a の絶対値を四捨五入して Y_a と同符号与えることにより，i 番目の量子化されたY座標値 Y_i を定める．

＜領域B，C，F，Gに対する量子化規則＞

1) Y座標値に関しては整数値のみを取るように量子化を行い，i 番目の量子化されたY座標値を Y_i とする．
2) Y_i に対応するアナログ直線上の点のX座標値を X_a とし，X_a の小数点以下に関しては X_a の絶対値を四捨五入して X_a と同符号与えることにより，i 番目の量子化されたX座標値 X_i を定める．

3.2　ディジタル直線の性質

前節において，図形の対称性を考慮しながら直線の量子化規則を定めた．本節では，量子化された直線（ディジタル直線）の性質について調べる．図 3-3 のA領域における直線を量子化し，その量子化された直線を構成しているチェイン符号を調べると，次の３種類のタイプに分類することができる．

＜A領域におけるディジタル直線の分類＞

1) X軸上の直線を構成するチェイン符号列は，全て0から構成される（図 3-5 (a)）．
2) 傾きが斜め 45 度の直線（Y＝X）を構成するチェイン符号列は，全て１から構成される（図 3-5 (b)）．
3) その他の直線を構成するチェイン符号列は，0と１の組み合わせである（図 3-5 (c)）．

上記の事柄を他の領域にも広めて考察を加えると，ディジタル直線には次の一般的な性質が備わっていることが分かる．

＜ディジタル直線の一般的な性質＞

1) ディジタル直線を構成するチェイン符号化列は高々２種類のチェイン符号から構成されており，その符号間の差は８の剰余系で１である．
2) 二種類のチェイン符号のうち，一方は常に孤立して現れる（連続して現れない）．
3) 孤立して現れるチェイン符号間の距離はほぼ一定である．

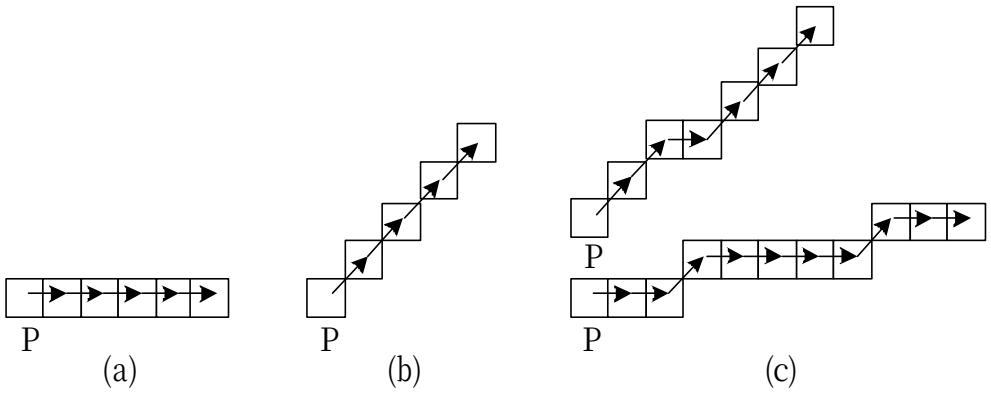


図 3-5 ディジタル直線の分類

さらに，上記＜ディジタル直線の一般的な性質＞における 3)のチェイン符号間の距離，つまり同一チェイン符号で繋がりが表現されている画素列を構成する画素の個数を詳しく調べてみることにする．図 3-5 (a)及び(b)のタイプの直線は常に一種類のチェイン符号から構成されているため，詳しく調べる対象から除外することにする．図 3-5 (c)についてチェイン符号化列を調べると，傾きが 1/2

以上の直線はアナログ直線の量子化規則における四捨五入という特性上，最初のチェイン符号が１となるのに対し，傾きが1/2未満の直線は最初のチェイン符号が０となることが分かる．つまり，直線の傾き1/2が，最初のチェイン符号が０になるのか１になるのかを決める閾値となっている．従って，図3-5(c)の直線はその性質上，次の２種類の直線に分類することができる．

＜A領域におけるディジタル直線の詳細分類＞

① 直線の傾きが 0 度を超え，かつ$\tan^{-1}(1/2)$（約 26．6）度未満の直線はチェイン符号０の列が並んだ後，チェイン符号１が単独で現れて直線の補正を施し，さらにチェイン符号０の列が並ぶという繰り返しにより直線が構成されている（図3-6の line 1）．

② 直線の傾きが$\tan^{-1}(1/2)$（約 26．6）度以上で，かつ 45 度未満の直線はチェイン符号１の列が並んだ後，チェイン符号０が単独で現れて直線の補正を施し，さらにチェイン符号１の列が並ぶという繰り返しにより直線が構成されている（図3-6の line 2）．

図3-6 より，上記＜A領域におけるディジタル直線の詳細分類＞における①は，チェイン符号０の方向に直線を構成するチェイン符号列が並び，１の方向に補正を施した後，再び０の方向にチェイン符号列が並んでいる．一方，上記＜A領域におけるディジタル直線の詳細分類＞における②は，チェイン符号１の方向に直線を構成するチェイン符号列が並び，０の方向に補正を施した後，再び１の方向にチェイン符号列が並ぶ構成となっている．そこで，この直線の趨勢を表すチェイン符号列（上記①の場合は０のチェイン符号列，②の場合は１のチェイン符号列）を**趨勢列**，また，その方向を**趨勢方向**と呼び，補正を表すチェイン符号（上記①の場合は１のチェイン符号，②の場合は０のチェイン符号）を**補正項**，また，その方向を**補正方向**と呼ぶことにする．

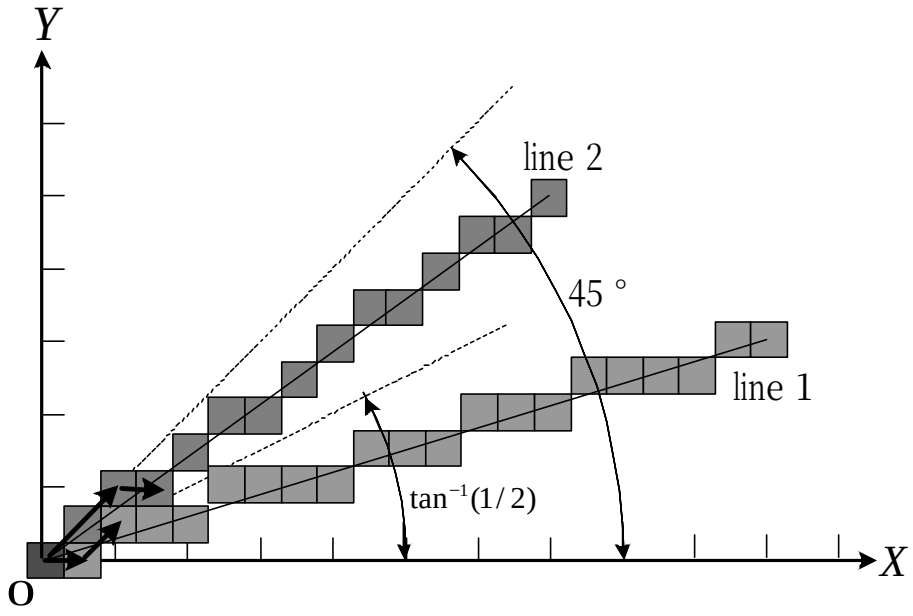


図3-6 ディジタル直線の詳細な分類

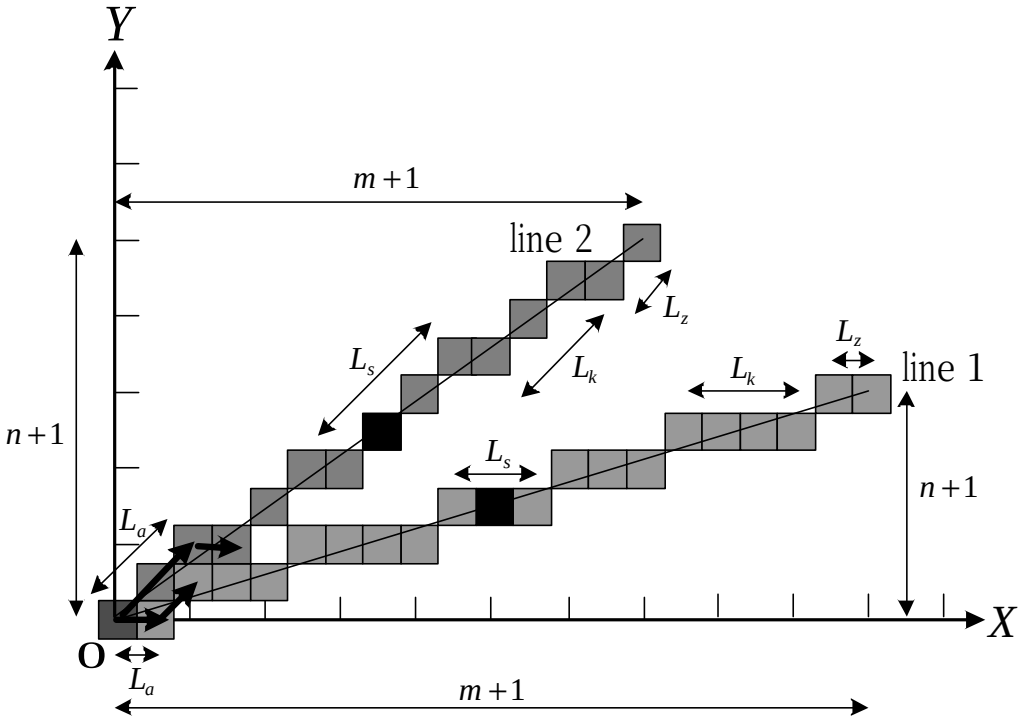


図 3-7 趨勢方向の画素列

ここで，直線の始点を$(0,0)$，終点を(m,n)（但し，m と n は互いに素な正の整数）とすると，直線の傾きは$a = n/m$ となる．今，直線の傾きが 45 度未満の場合を考えているから$m > n$ であり，始点から終点までの直線を構成する画素の総数は$m+1$個，Y軸方向の段差の数は$n+1$段となる．次に m と n が互いに素ではない正の整数の場合を考える．m と n が互いに素ではないから，二つの最大共役数を c とすると，$(m,n) = (c \times m', c \times n')$（但し，$m'$ と n' は互いに素な正の整数）と記述することができ，始点$(0,0)$ から終点(m,n) に至る直線は，$(0,0)$ から(m', n') までの基本となる直線パターンの繰り返しになっており，この場合には，基本パターンを c 回繰り返すことになる．そして，この基本となる直線パターンの繰り返しにおいて，(m', n') という点は最初の直線パターンの終点であると同時に，後の直線パターンの始点にもなっている．そこで，このディジタル直線を構成する途中の画素で基本パターンの終点と始点が重なる点を**中継点**（図 3-7 における黒色の点）と呼ぶことにする．

次に，単独で現れるチェイン符号間の距離，つまり同一チェイン符号で繋がりが表現されている画素列を構成する画素の個数を調べてみることにする．図 3-7 に示すように，直線の始点から始まり趨勢方向に並ぶ最初の画素列の個数を L_a，直線の途中における中継点を含まない k 番目の趨勢方向に並ぶ画素列の個数を L_k（但し，$1 \le k \le n-1$），直線の途中における中継点を含む趨勢方向に並ぶ画素列の個数を L_s，直線の終点に至る趨勢方向に並ぶ最後の画素列の個数を L_z とすると，ディジタル直線は以下の性質を持っていることが分かる．なお，数式導入の過程は付録として記すことにする．

＜ディジタル直線の性質＞

Ⅰ) $0<a=1/h<1/2(n=1,m=h)$ **の場合（** L_k **は存在せず，途中の画素列は全て** L_s **）**

$$L_a=\lceil h/2\rceil, L_s=h, L_z=(h+1)-\lceil h/2\rceil$$
$$2L_a-1\le L_s\le 2L_a, L_a\le L_z\le L_a+1$$

Ⅱ) $0<a=n/m<1/2(n\neq 1, n,m$ **は互いに素な正の整数)の場合（図 3-7 の** line 1**）**

$$L_a=\left\lceil\frac{m}{2n}\right\rceil, L_k=\left\lceil\frac{(2k+1)m}{2n}\right\rceil-\left\lceil\frac{(2k-1)m}{2n}\right\rceil$$

$$L_s=\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil+m, L_z=m+1-\left\lceil\frac{(2n-1)m}{2n}\right\rceil$$

$$2L_a-2\le L_k\le 2L_a, 2L_a-2\le L_s\le 2L_a, L_z=L_a$$

Ⅲ) $1/2\le a=n/m<1(n,m$ **は互いに素な正の整数)の場合（図 3-7 の** line 2**）**

$$L_a=\left\lfloor\frac{m}{2(m-n)}\right\rfloor+1, L_k=\left\lfloor\frac{(2k+1)m}{2(m-n)}\right\rfloor-\left\lfloor\frac{(2k-1)m}{2(m-n)}\right\rfloor$$

$$L_s=\left\lfloor\frac{m}{2(m-n)}\right\rfloor-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+m, L_z=m-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor$$

$$2L_a-2\le L_k\le 2L_a, 2L_a-2\le L_s\le 2L_a-1, L_a-1\le L_z\le L_a$$

但し， $\lceil X\rceil$ **は** X **以上の最小の整数（つまり，少数点以下切り上げ）を，** $\lfloor X\rfloor$ **は** X **を超えない最大の整数（つまり，小数点以下切り捨て）を求める関数である．**

3.3 ディジタル直線による点列の直線近似

本節では，前節でその性質を調べたディジタル直線を用いて，自然界に存在する物体の境界を直線で近似する方法（**ディジタル直線近似**）を提案する．例えば，対象物体をカメラで撮影し印画紙に印刷した後にイメージスキャナを用いてディジタル画像として取り込んだり，あるいは最初からディジタルカメラを用いて撮影したりすると，対象物体をディジタル画像データとして入力することができる．この対象物の境界を直線で近似するには，通常，微分オペレータを用いて境界部分の点列を抽出した後，これらの点列を近似する直線を想定し，近似直線と元の点列との誤差が最小となるように最適な近似直線を見つける．近似直線と元の点列との誤差解析には一般に誤差の二乗が最小となる最小二乗誤差法が用いられるが，元の点列と近似直線との最大誤差を最小とする**ミニマックス近似法**もよく用いられる近似手法である[9]．また，与えられた点列に対して，点列の両端点を結ぶ一本の直線を割当てて直線と点列との誤差を計算し，次に誤差が最大となる点を順次加えて近似直線を反復的に分割することにより，点列を複数の直線で近似する方法（本論文では**反復近似法**と呼ぶ）も報告されている[10]．

最初に，両近似方法について簡単に説明しておく．図 3-8 はミニマックス近似方法を説明する図であり，図 3-8 では a から i までの８点を与えている．最初に点 a 及び b を対象として，これら２点を

結ぶ直線 L1 を考えると，両点とも近似直線 L1 上に乗っているので，その誤差は 0 となり，与えられた直線 L1 は 2 点 a 及び b を最もよく近似する直線となる．次に，点 c を含めた 3 点を考え，これらの点を近似する直線 L2 を考える．近似直線の誤差として，各点から近似直線 L2 までの距離を計算し，最大距離（つまり最大誤差）が最小となるように，直線のパラメータを計算する．もし，最大誤差が与えられた閾値以下であれば次の点 d を含めた近似直線 L3 を考え，最大誤差が閾値を超えるまで一本の直線で近似する点列の対象範囲を拡大する．一方，近似直線 L2 の段階で最大誤差が閾値を超えている場合には，一段階前の近似直線 L1 を採用し，点 c から新たな近似直線 L4 を考えて，直線近似を続ける．最終的には得られた近似直線の交点（図 3-8 の○印の点）を結ぶことにより，与えられた点列を直線で近似することができる．

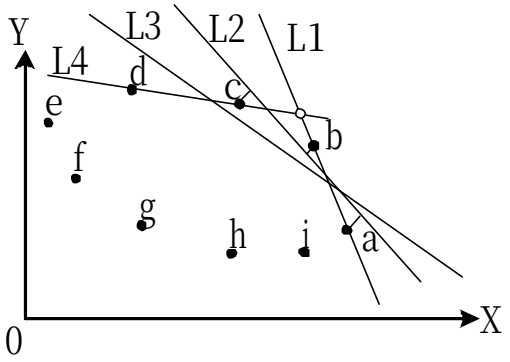


図 3-8 ミニマックス近似方法

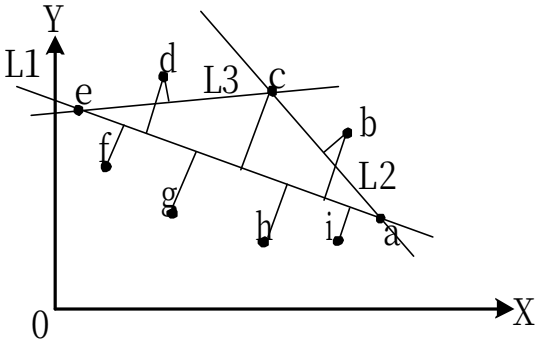


図 3-9 反復近似方法

一方，反復近似方法では全ての点列を対象として，2 点間の距離が最大となる点を求める．図 3-9 では点 a と e の距離が最大となっているので，この 2 点を採択された点とする．そして，この 2 点を結ぶ直線 L1 を近似直線として考え，残りの点から近似直線 L1 までの距離を計算する．次に，これらの点の中で最大距離を与える点（図 3-9 では c）を求め，この点を採択された点に追加すると共に，この点からの距離を最大とする近似直線 L1 を構成する点 a 及び e と，この点 c とを結ぶ直線 L2 及び L3 を近似直線として追加する．そして再び，各点から近似直線 L1，L2 及び L3 までの距離を求め，最大距離を与える点を採択された点として追加すると共に，この点からの距離を最大とする近似直線を構成する点と，この点とを結ぶ直線を近似直線として追加し，各点から近似直線までの距離が閾値以下になるまでこの計算を繰り返す．全ての点から近似直線までの距離が閾値以下になれば，採択された点を結ぶことにより，与えられた点列を直線で近似することができる．ミニマックス近似方法では近似直線の交点を計算し，新たに生成された交点列が与えられた点列を直線で近似するための点列となるが，反復近似方法では新たな点は生成されず，近似直線を構成する点は全て与えられた点の一部であるという点が異なる．

次に，ディジタル直線近似について説明する．前節では線幅を持たない真っ直ぐなアナログ直線を量子化することにより得られるディジタル直線についてその性質を調べたが，ディジタルカメラなどで撮影した自然物のディジタル画像データにはノイズが混入する．このため，紙の上に定規などを用いて真っ直ぐに引いたアナログ直線をディジタル画像として取り込んでも前節で証明したディジタル直線の性質を満足する直線とはならない．一例を図 3-10 に示す．図において，□は傾き 1/6 のノイズが混入しない状態でのディジタル直線を構成する画素列であり，○はノイズが混入した状態

の量子化直線を構成する画素列である．また，直線が単独で現れることはほとんどなく，通常は物体の境界線として現れる．このため，前記２方法を含めた従来の方法では，入力されたディジタル画像に対して微分オペレータや二値化及び細線化処理などを施すことにより，物体の境界線を抽出した後，これらの点列を対象に直線近似を行っていた．一方，3.1 にて説明したチェイン符号を用いると，微分オペレータや二値化及び細線化処理などの物体の輪郭を抽出する前処理を行うことなく，対象物体の境界を追跡しながら，直線近似の対象となる点列を直接抽出することが可能であり，高速な直線近似を期待することができる．

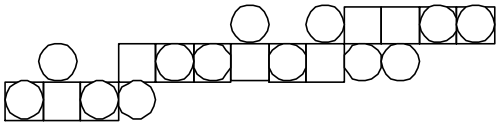


図 3-10 ノイズのないディジタル直線と実際の直線

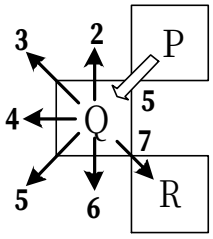


図 3-11 境界追跡方法

そこで，ディジタル直線近似を次の様にして行う．

＜ディジタル直線近似方法＞

1) 対象物体の境界をチェイン符号を用いて追跡する．
2) 境界追跡により得られた複数の画素列から，ノイズが混入していない状態のディジタル直線を推測して発生させる．
3) さらに対象物体の境界追跡を続け，逐次得られる画素と既に発生させているノイズが混入していない状態でのディジタル直線を構成する画素との差分を取り，差分が閾値以下であれば同一直線での近似を続ける．
4) 逐次得られる画素とノイズが混入していない状態でのディジタル直線を構成する画素との差分が閾値を超える場合は同一直線での近似を終了し，1)からの処理を対象物体の境界追跡が一周するまで続ける．

以下，上記アルゴリズムについて解説する．まず，境界追跡は次のようにして行うことができる．今，図 3-11 の点Ｐから点Ｑが見つかったと仮定すると，点Ｐから点Ｑへのチェイン符号は５である．従って，チェイン符号５と逆方向のチェイン符号である１から反時計回りに一つ進めたチェイン符号である２から始め，チェイン符号で指す先に次の境界点が存在するかどうかを調べる．もし，境界点が存在していない場合は，順次反時計回りにチェイン符号を更新しながら，次の境界点が発見されるまで続ける．図 3-11 ではチェイン符号７で次の境界点が見つかっているため，このチェイン符号７の逆方向のチェイン符号３から反時計回りに一つ進めたチェイン符号４より，次の境界点が存在するかどうかを調べる．ここでチェイン符号は８の剰余系で成り立っているため，チェイン符号 C1 の逆方向のチェイン符号 C2 及び最初の検索対象となる画素の方向を示すチェイン符号 C3 は一般に，次のようにして計算することができる．

C2 = (C1+4)%8, C3 = (C1+5)%8 (但し，x%y は x を y で除算した際の余りを表す)

また，境界追跡を始める最初の点は，例えば対象物体の画像を上から順次走査し，最初に見つかった点を取ればよい．画像走査は通常，左から右へ行われるため，このときのチェイン符号は０であり，次の境界点を見つけるためのチェイン符号は５(=(0+5)%8)から始めればよい．また，境界追跡を行う場合，基本的には入力画像を二値化してから行うが，濃淡画像やカラー画像でも輝度や各カラー要素における二値化の閾値を明確にしておけば，最初から二値化を施しておく必要はない．従って，対象物体の境界を追跡しながらその物体の境界を直線近似する方法は，前処理としての二値化や微分オペレータ，あるいは細線化などの輪郭抽出処理が不要となり，高速な直線近似を期待することができる．

次に，境界追跡により得られた複数の画素列から，ノイズが混入していない状態のディジタル直線を推測する方法について説明する．

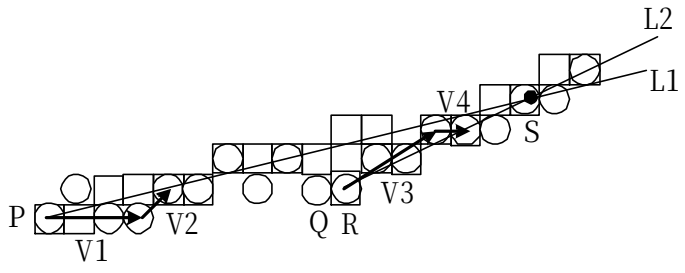


図 3-12 ディジタル直線近似

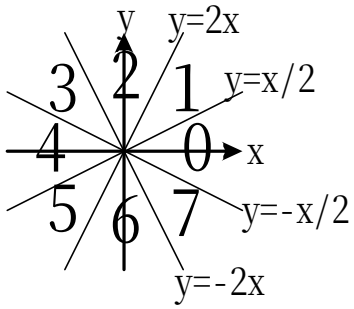


図 3-13 チェイン符号の領域

図 3-12 において，点Ｐから物体の境界を追跡し，順次境界点○印が見つかるものとする．ディジタルカメラなどで入力した自然画像のデータではノイズなどの混入により，3.1 の量子化規則に従うディジタル直線とは異なるため，開始点から次の境界点を抽出しただけでは直線の趨勢方向は得られない．そこで，直線近似の開始点からの数画素を参照してノイズが混入していない状態のディジタル直線の趨勢方向を決定する．この参照すべき画素を**参照画素**，その個数を**参照点数**と呼ぶことにする．物体の境界追跡において，次の境界点の方向を表すチェイン符号のベクトル（例えばチェイン符号０なら(1, 0)，１なら(1, 1)）を加算し，その合成ベクトルを直線の趨勢方向とする．そして，直線の趨勢方向決定後，その趨勢方向を表すベクトルが属する領域のチェイン符号と異なるチェイン符号が現れると，その方向を直線の補正方向とする．例えば，図 3-12 において，点Ｐからの参照点数を４（チェイン符号の数は３）として，境界追跡により得られた境界点の方向を表すチェイン符号のベクトルを加算して直線の趨勢方向を表すベクトル V1=(3, 0)を生成し，このベクトル V1 が属する領域のチェイン符号を図 3-13 より０と求める．その後，境界追跡を進める過程で，直線の趨勢方向を表すチェイン符号０と異なる方向のチェイン符号１が現れると，この方向 V2 を補正方向とし，これら２つのベクトル（V1 と V2）の合成ベクトルからノイズが混入していない状態のディジタル直線の傾きを求め，3.1 の量子化規則に従って，ディジタル直線（図 3-12 の□印）を発生させる．なお，直線の趨勢方向を表すベクトル（x, y）から，そのベクトルが属する領域のチェイン符号を求める方法は図 3-13 を参照すると，次のアルゴリズムで記述することができる．

＜ベクトルが属する領域のチェイン符号の求め方＞

もし， $(x > 0, 2|y| < |x|)$ **なら，チェイン符号は0**

もし， $(x > 0, y > 0, 2|y| \geq |x|, |y| \leq 2|x|)$ **なら，チェイン符号は1**

もし， $(y > 0, |y| > 2|x|)$ **なら，チェイン符号は2**

もし， $(x < 0, y > 0, 2|y| \geq |x|, |y| \leq 2|x|)$ **なら，チェイン符号は3**

もし， $(x < 0, 2|y| < |x|)$ **なら，チェイン符号は4**

もし， $(x < 0, y < 0, 2|y| \geq |x|, |y| \leq 2|x|)$ **なら，チェイン符号は5**

もし， $(y < 0, |y| > 2|x|)$ **なら，チェイン符号は6**

もし， $(x > 0, y < 0, 2|y| \geq |x|, |y| \leq 2|x|)$ **なら，チェイン符号は7**

もし， $(x = 0, y = 0)$ **なら，対応するチェイン符号はなし**

3.1 の量子化規則に従ったディジタル直線が発生できると，境界追跡により得られる物体の境界点と発生させたディジタル直線との差分を求め，この差分がある一定の閾値以内であれば同一直線による近似を続け，閾値を超える場合は同一直線での近似を終了し，発生したディジタル直線との差分が閾値を超る境界点を始点として新たな直線近似を始める．図 3-12 では境界点とディジタル直線との差分の閾値を 1 とし，点Qまでは差分が閾値 1 以内であるが，点Rにおいて差分が 2 となるため，同一直線 L1 での直線近似を終了し，点Rを開始点として新たなディジタル直線 L2 を用いた近似を行っている．ここで，境界点とディジタル直線との距離差分算出方法は 3.1 の量子化規則に従う．つまり，図 3-3 における領域A，D，E，Hに対しては同一のX座標値に対応するY座標値の差分を 2 点間の距離差分とし，領域B，C，F，Gに対しては同一のY座標値に対応するX座標値の差分を 2 点間の距離差分とする．

ここで，閾値の決め方について説明する．同一直線での近似を続けるか否かの判断基準となる閾値は直線近似における許容誤差を考慮して決定すべきものである．図 3-14 に示すように，傾き q が 0 度から 45 度の範囲内に入るアナログ直線を 3.1 の量子化規則に従って得られたディジタル直線を構成する各画素は，元のアナログ直線とY軸方向に最大 1/2 の距離誤差を持つ．従って，この点PからY軸方向に l だけ離れた点Qのアナログ直線までの距離誤差は $\{l + (1/2)\}\cos q$ となる．境界追跡により得られた参照画素を基に算出した趨勢方向が図 3-3 の領域A，D，E，Hに属する場合，境界点とディジタル直線との距離差分は，同一X座標値に対応するY座標値の差により計算されるので，点Pと点QとのY軸方向の距離 l を閾値として決定しなければならない．ディジタル直線近似における許容誤差を e とすると，$\{l + (1/2)\}\cos q \leq e$ を満足する l を求めることになる．$l \leq (e/\cos q) - (1/2)$ であるから，$l = \lfloor (e/\cos q) - (1/2) \rfloor$ となり，直線近似の許容誤差 e を用いて同一直線での近似を続けるか否かの判断基準となる閾値 l を算出することができる．

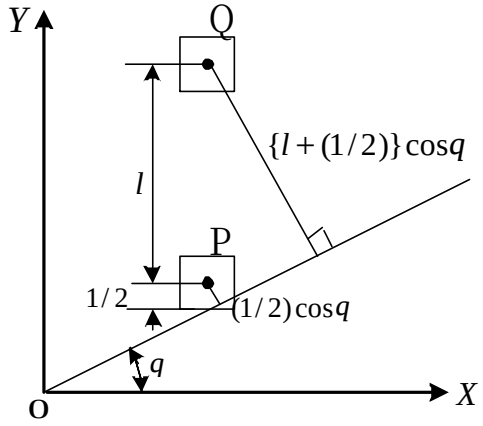


図 3-14 ディジタル直線近似の誤差

このようにして，対象物体の境界を追跡しながら，その境界を構成している画素列を部分的に直線で近似し，これらの直線の交点を計算して得られた交点を結ぶことにより，ディジタル直線での近似を行う．例えば，図 3-12 において，最初に算出された近似直線 L1 と次に算出された近似直線 L2 の交点Ｓを求め，これらの交点を結ぶことにより，最終的な近似直線を得ることができる．但し，２直線の交点を求める際，２直線の傾きが平行に近い場合には適当な位置に交点を求めることができないため，このような場合には次の補正を行う．

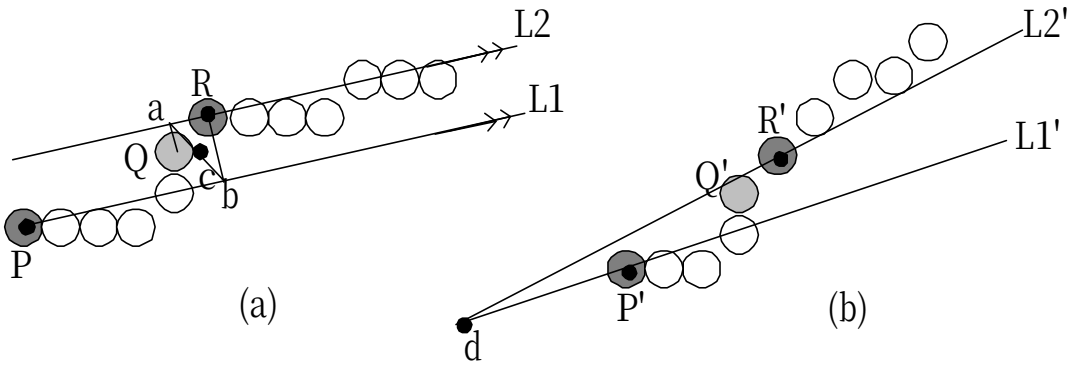


図 3-15 近似直線の交点補正

図 3-15 (a)は点Ｐを始点としてディジタル直線近似を行い，点Ｑまで近似した時点で近似直線 L1 を抽出し，点Ｒから新たに別の近似直線 L2 にて近似する場合であり，抽出された近似直線 L1 及び L2 が平行で両直線の交点が存在しない場合である．このような場合には，最初のディジタル直線近似における終点Ｑから次の近似直線 L2 に下ろした垂線の足を a，また，次のディジタル直線近似における始点Ｒから最初の近似直線 L1 に下ろした垂線の足を b として a と b の中点 c を算出し，この点を直線 L1 及び L2 の交点として代用する．一方，図 3-15 (b)は点Ｐ’を始点としてディジタル直

線近似を行い，点Q’まで近似した時点で近似直線 L1’を抽出し，点R’から新たに別の近似直線 L2’で近似する場合であり，この場合は２直線 L1’と L2’の傾きが非常に近いことから，その交点 d が本来の点（Q’あるいはR’）から離れている場合である．このような場合には，後のディジタル直線における始点R’を２直線 L1’及び L2’の交点として代用する．こうすることにより，ディジタル直線近似により得られる最終画像が本来の画像から大きくずれることを防ぐことができる．但し，最終的な近似直線が補正されているため，対象物体から近似直線までの誤差に変動が生ずるので，最終的な近似直線を対象とした誤差評価を行う必要がある．

以上のようにして得られたディジタル直線近似（DL 法）を実際の画像に適用し，従来の方法であるミニマックス近似（MM 法）及び反復近似（RA 法）と比較した．直線近似の方法及び近似精度の評価については，次のように行っている．まず，入力された画像から微分オペレータ及び細線化処理により物体の輪郭部分を抽出し，この図形を **Original** とする．ここではこの物体の輪郭である Original を抽出する前処理を**輪郭抽出**と呼び，物体の境界を直線で近似する際に行う**境界追跡**と区別する．つまり，輪郭抽出は Original を求める前処理であるのに対し，境界追跡は直線近似中に行う処理である点が異なる．輪郭抽出では対象物体に微分オペレータを施した後，細線化処理を行うことにより，Original を１ドットの点列で構成する図形としているが，境界追跡はその性質上，物体の輪郭を構成する１ドットの点列を追跡する手法であるため，最終的には Original と同じ図形を得ることができる．従って，微分オペレータ及び細線化処理による輪郭抽出処理ではなく，境界追跡を用いて Original 図形を求めてもよいが，従来の手法では境界追跡による輪郭抽出についての言及はなかったため，従来用いられている微分オペレータ及び細線化処理を輪郭抽出の手法として採用している．

上記のようにして得られた Original に対して MM 法及び RA 法を適用し，Original 図形を構成する点列を直線で近似して，近似直線同士の交点を求める．次に，求められた直線の交点を結んで得られる直線に対して 3.1 で記述した量子化規則に従い，８連結の画素列として近似図形を表現する．そして，近似された図形と Original 図形の誤差を次のようにして計算する．まず，Original 図形の内点であるが近似図形の外点である画素の数を E1，次に Original 図形の外点であるが近似図形の内点である画素の数を E2 とし，(E1-E2)を Original 図形の面積で除した値を誤差として定義している．

また前述しているとおり，DL 法に関しては物体の境界を直接追跡しながら同時に直線近似も行うため，輪郭抽出時間は不要となる．各方法の比較一覧を表 3-1 に，その平均値を表 3-2 に，また，処理結果画像を図 3-16～3-22 に示す．

評価尺度としては，直線近似により得られる交点の数，近似誤差，直線近似に要する総時間，及び結果としての図形を比較することによる主観評価などを取り上げている．但し，結果を示す図（図 3-16～3-22）は直線近似結果（交点を結ぶ直線を量子化した結果）をプリントアウトした後，イメージスキャナによりコンピュータ内に取り込み，論文掲載のための縮小処理を施しているために近似直線の一部が欠けた状態となっている．

表 3-1 直線近似結果一覧

	方法	頂点数	面積	近似時間	E1	E2	圧縮率	誤差
1. ティーポット	MM	49	37,847	33分06秒	819	136	0.060	0.018
輪郭点数： 811画素	RA	38	37,660	2分26秒	880	10	0.047	0.023
面　積：38,530画素	DL	47	37,356	3分04秒	1,233	59	0.057	0.030
輪郭抽出時間： 1分56秒		(57)*						(0.070)**
2. モンキースパナ	MM	53	10,838	29分35秒	391	343	0.076	0.004
輪郭点数： 697画素	RA	23	10,849	1分42秒	197	160	0.033	0.003
面　積：10,886画素	DL	30	10,830	2分30秒	322	266	0.043	0.005
輪郭抽出時間： 1分43秒		(32)*						(0.046)**
3. 電話の受話器	MM	43	11,462	22分35秒	321	279	0.075	0.004
輪郭点数： 571画素	RA	23	11,394	1分31秒	188	78	0.040	0.010
面　積：11,504画素	DL	29	11,191	2分03秒	436	123	0.050	0.027
輪郭抽出時間： 1分23秒		(40)*						(0.070)**
4. コーヒーミル	MM	51	30,818	50分22秒	418	462	0.046	-0.001
輪郭点数：1,120画素	RA	48	30,762	3分23秒	271	237	0.043	0.001
面　積：30,796画素	DL	54	31,496	3分48秒	278	978	0.048	-0.023
輪郭抽出時間： 2分37秒		(161)*						(0.144)**
5. フライパン	MM	53	33,057	30分36秒	478	351	0.064	0.004
輪郭点数： 825画素	RA	28	33,150	2分13秒	197	164	0.034	0.001
面　積：33,184画素	DL	32	32,618	2分56秒	625	59	0.039	0.017
輪郭抽出時間： 2分06秒		(52)*						(0.063)**
6. ハサミ	MM	55	11,400	50分02秒	405	501	0.056	-0.009
輪郭点数： 985画素	RA	42	11,025	2分58秒	371	102	0.043	0.024
面　積：11,294画素	DL	49	11,109	3分23秒	561	376	0.050	0.016
輪郭抽出時間： 2分33秒		(113)*						(0.115)**
7. 数字の５	MM	53	8,898	24分40秒	378	325	0.076	0.006
輪郭点数： 695画素	RA	42	8,922	2分20秒	186	157	0.060	0.003
面　積：8,951画素	DL	44	8,896	2分30秒	381	326	0.063	0.006
輪郭抽出時間： 2分40秒		(76)*						(0.100)**

MM：ミニマックス近似（Kurozumi and Davisの方法）
RA：反復近似（Ramerの方法）
DL：ディジタル直線近似(本論文の提案方法)
E1：Original図形の内点であるが，近似図形の外点である画素の数
E2：Original図形の外点であるが，近似図形の内点である画素の数
圧縮率＝頂点数/輪郭点数
誤差＝（E1－E2）/Original図形の面積
＊　DL法の近似結果に対して，近似直線までの距離が閾値を超える元図形の輪郭点数 但し，閾値は全て2画素
＊＊　精度＝（＊の数）/（輪郭点数）

表 3-2 近似結果の平均値

	MM	RA	DL
圧縮率	0.06	0.04	0.05
誤　差	0.007	0.009	0.018
精　度	0	0	0.09
頂点数比	1.25	0.86	1
輪郭抽出時間	2分10秒	2分10秒	0秒
直線近似時間	34分25秒	2分21秒	2分53秒

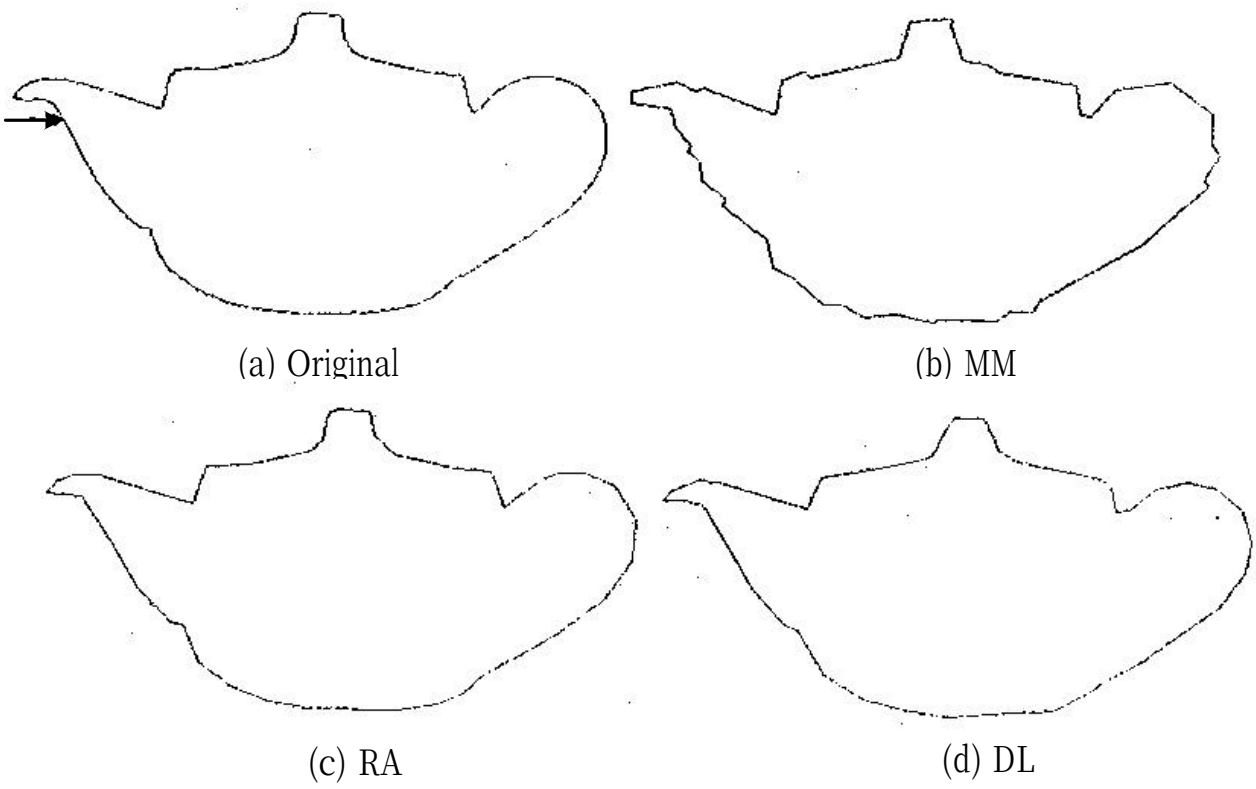


図 3-16 直線近似結果の比較（ティーポット）

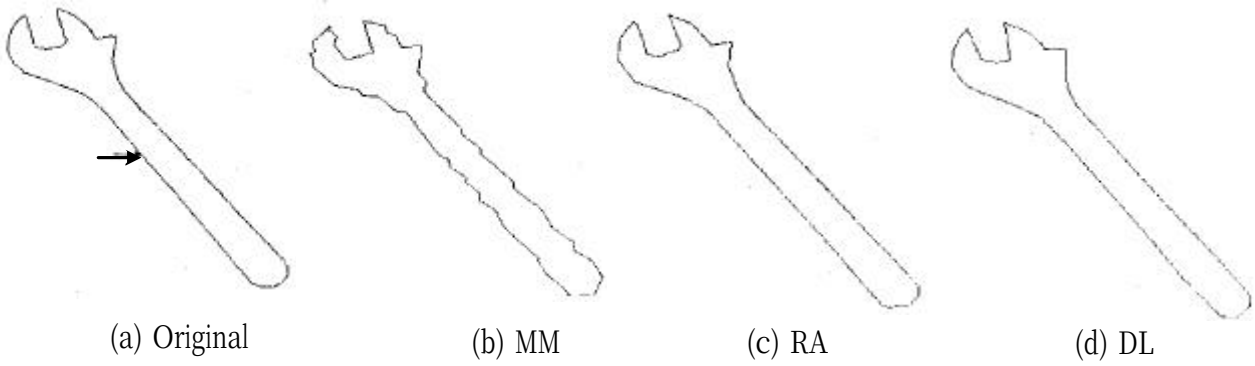
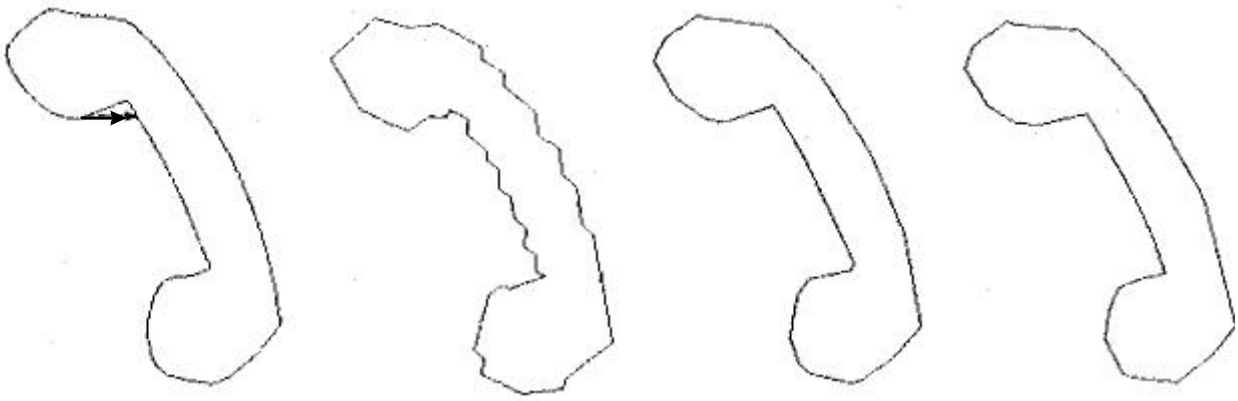
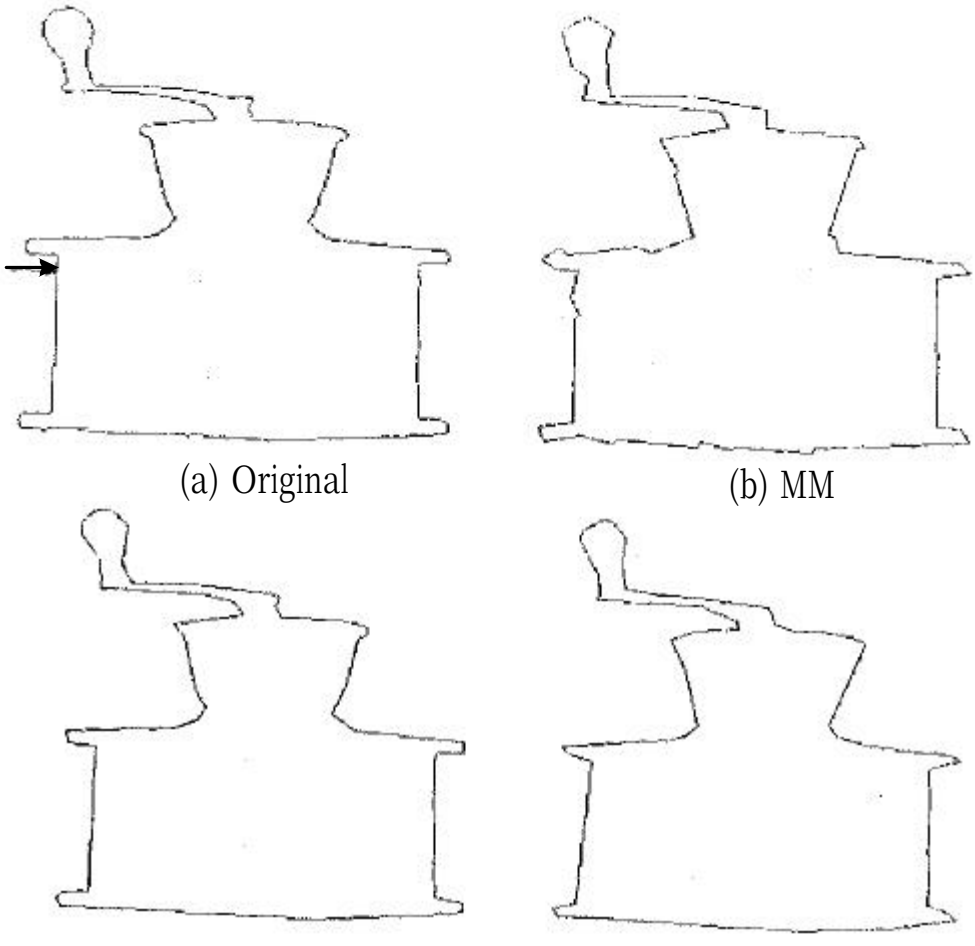


図 3-17 直線近似結果の比較（モンキースパナ）



(a) Original　(b) MM　(c) RA　(d) DL

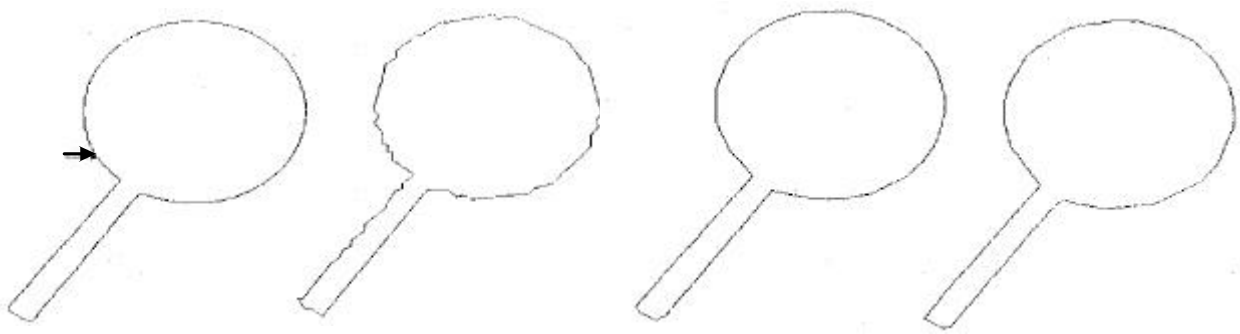
図 3-18 直線近似結果の比較（電話の受話器）



(a) Original　(b) MM

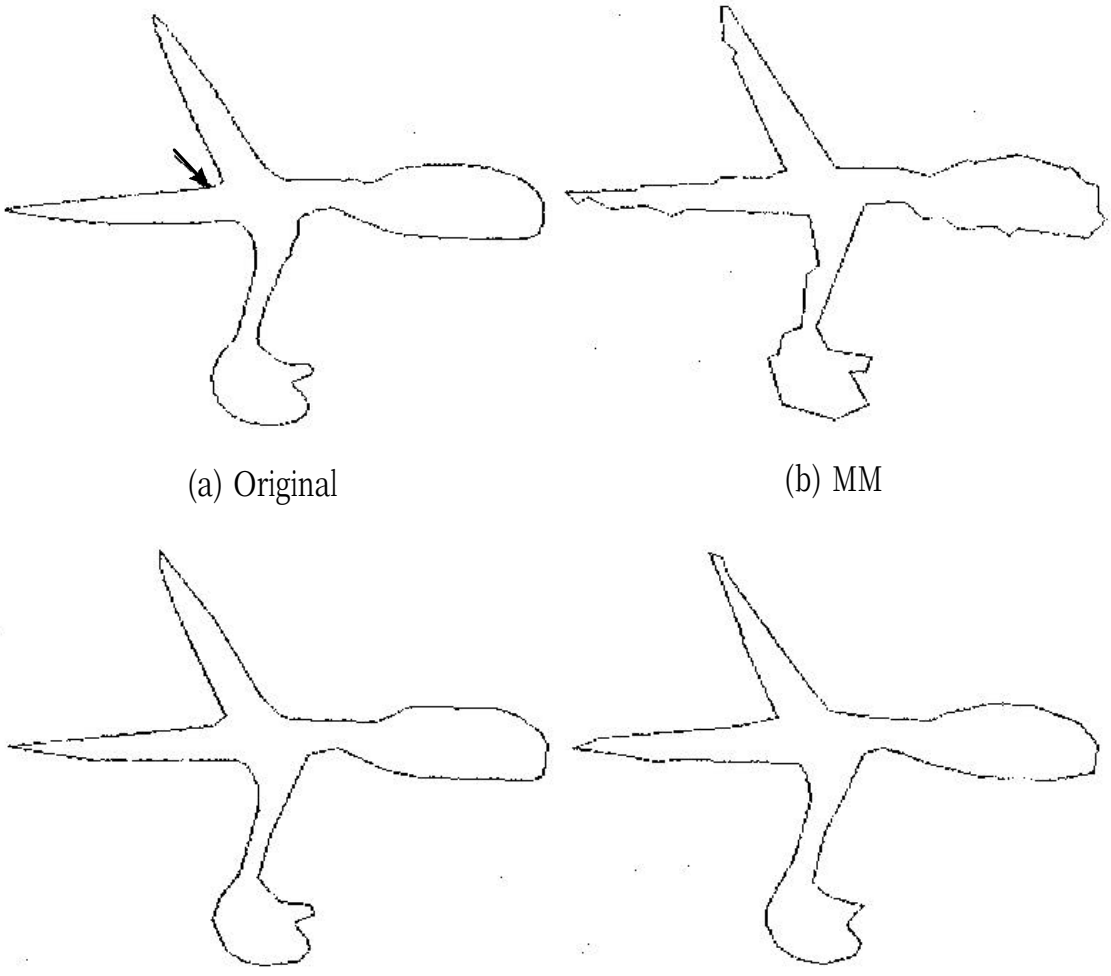
(c) RA　(d) DL

図 3-19 直線近似結果の比較（コーヒーミル）



(a) Original　(b) MM　(c) RA　(d) DL

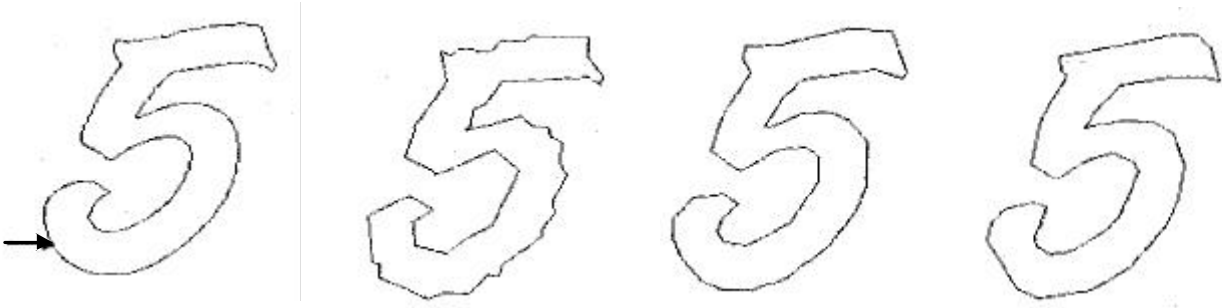
図 3-20 直線近似結果の比較（フライパン）



(a) Original　(b) MM

(c) RA　(d) DL

図 3-21 直線近似結果の比較（ハサミ）



(a) Original　(b) MM　(c) RA　(d) DL

図 3-22 直線近似結果の比較（数字の５）

本実験では，安立電気社製 PacketⅡe を使用し，BASIC を用いてプログラムを組んでいる．ディジタル直線近似における参照点数は３とし，閾値誤差は全てにおいて２画素としている．また，直線近似の開始点を矢印にて示しているが，これは，次のようにして求められる．

1) 画面の中央を走査し，最初に発見された境界点を直線近似の開始点とする．
2) 上記走査により境界点が見つからなければ，画面を中央線により２分割し，各画面に対して1)の走査を繰り返す．

図 3-16～3-22 より，DL 法の画質（主観評価）は MM 法と RA 法の中間に位置することが分かる．また，物体の境界を直線近似したことによる圧縮率（近似直線の交点数を Original 図形の輪郭を構成する画素の総数で除した値）に関しても，表 3-2 より，DL 法は MM 法と RA 法の中間に位置していることが分かる．但し，誤差についてはあまりよい結果が出ているとは言えず，特に，近似直線の交点補正を行っているために，近似誤差が閾値を超えている画素も存在する．この閾値を超える画素の割合を精度として表 3-2 で表している（この値は０に近いほど精度が高いことになる）．

MM 法は３つの方法の中で最も誤差が小さいのにも関わらず，圧縮率及び主観評価の画質が悪い．これは，順次与えられる点列を逐一直線で近似しており，最大誤差を最小とする誤差補償を行っているために，細かな近似直線が多数抽出された結果である思われる．これに対して，RA 法は近似の最初に２点間の距離が最大となる点を求め，これらの点を結ぶ直線を基本とし，基本直線を順次二分割することにより近似を行っていることから，近似処理としては大局的な処理方法であると言える．この大局的な処理方法のお陰で圧縮率がよく，同時に主観評価の画質もよい結果となっている．しかしながら，大局的な処理を行うということは，必ず前処理としての微分オペレータ及び細線化処理による輪郭抽出時間が必要となる．本実験では MM 法も引用文献に従って輪郭抽出を行ってから直線近似処理を施したが，MM 法は RA 法と異なり大局的な処理は施していないため，DL 法と同じく，境界追跡を行いながら同時に直線近似を行うことが可能である．但し，このことを考慮して MM 法から輪郭抽出時間を削除しても MM 法の直線近似にかかる総時間はあまり変わらない．MM 法

は最大距離を最小とする直線を求める計算に多大な時間がかかっており，このことが近似処理に膨大な時間を要する原因となっている．

一方 DL 法は，物体の境界を直接追跡しながら直線近似も行っており，輪郭抽出時間が不要である．前述したように，境界追跡が行えるということは近似処理が局所的に行えるということであり，上記の評価より，局所処理を行う方法（MM 法）は大局処理を行う方法（RA 法）に比べて主観評価が悪いということになるが，図 3-16〜3-22 から判断して DL 法は RA 法とほぼ同じ画質を保っていることが分かる．結果として，DL 法は局所処理による高速近似を行っているにも関わらず近似結果として得られる画質もよいので，近似誤差が厳密ではなく，むしろ直線近似の高速性が要求される応用分野では効率的な直線近似を提供する方式であると言える．

3.4 ディジタル直線を用いた移動物体の認識

前節では，ディジタル直線を用いて物体の境界を効率よく近似する方式を考案した．そこで本節では，この直線近似方式を動画像に適用し，移動物体のリアルタイムな形状認識が可能かどうかを検証する．但し，ここでは 33ms 毎に CCD カメラから入力されてくる映像を基にリアルタイムな形状認識を行うため，ディジタル直線近似における参照点数を基に直線の趨勢方向を決定すると，その趨勢方向を表すベクトルが属する領域のチェイン符号(図 3-13 参照)から８の剰余系で１だけ異なるチェイン符号を同一直線で近似する際の許容範囲とし，これ以外のチェイン符号が現れると，別の直線として近似を続ける．

実験では，縦×横＝130×136cm^2 の矩形リング上面に黒色画用紙を敷き詰めて入力画像の背景とし，このリング上に，リモートコントロールで移動できる物体の上面に白画用紙で作成した三角形あるいは四角形の図形を貼り付けて認識対象物とした．認識対象物体である三角形は底辺 15㎝，高さ 20㎝の二等辺三角形であり，四角形は一辺 15㎝の正方形である．この状況を地上から 229㎝の天井に取り付けた CCD カメラより撮影し，得られた映像を基に物体の形状認識を行う．この様子の一部(対象物のある部分を拡大表示した映像)を図 3-23 に示す．背景となる矩形リングを黒画用紙で覆い，認識対象物の上面に白画用紙で作成した図形を貼り付けることにより，入力される画像データは適当な閾値を設けることで，容易に二値化することができる．CCD カメラにより得られた映像の解像度は 128×128 であり，得られた画像を走査し，各物体の境界を追跡しながら境界部分を直線で近似する．近似された物体が３本の直線から構成されていれば三角形，４本の直線から構成されていれば四角形である．但し，実際の画像にはかなりのノイズが混入するため，認識対象物以外にも数種類の物体を認識するが，物体の面積がある一定の閾値以下のものはノイズと判断している．

次に，認識が正しく行われていることを検証するために，各物体の形状を認識した後，三角形の物体を四角形の物体に近づけるプログラムを組み込んだ．三角形は底辺 15㎝，高さ 20㎝の二等辺三角形であるから，三角形として認識された物体の各直線間の角度を求めることにより，二等辺三角形の頂角方向が分かるので，三角形物体の姿勢制御を容易に行うことができる．そして，図 3-24 に示すように，二等辺三角形の重心から頂角に至るベクトルと二等辺三角形の重心から四角形の重心に至るベクトルのなす角度を求め，この角度がある一定の閾値q （実験では 40 度）を超えていれば二等辺三角形に回転（右回転あるいは左回転）という制御命令を，閾値以下の角度であれば前進という制御命令を三角形の物体に送信し，三角形の物体が備えている車輪を回転させることにより，四角形の物体に近づけることができる．

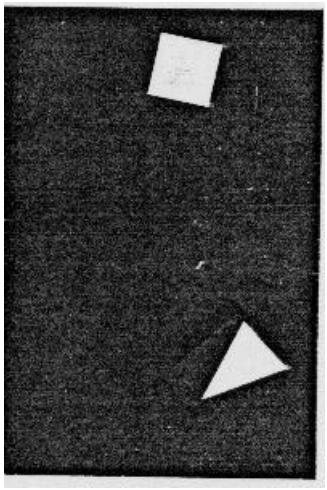


図 3-23 認識対象移動物体

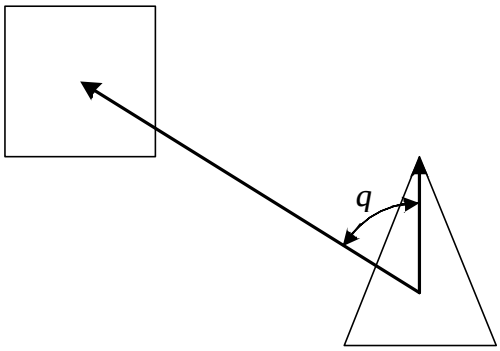


図 3-24 制御命令

実験は次のようにして行った．天井に取り付けられた CCD カメラから 33ms 毎に入力される映像を CPU 68000 を搭載する安立電気社製 Packet IIe で処理し，境界部分の直線近似及び物体の形状認識を施した後，三角形に与える制御命令を，CPU Z-80A を搭載する NEC 社製 PC-8001 mk IIに RS-232C を介して送信する．そして，PC-8001 mk IIは 40MHz あるいは 20MHz の無線信号により，三角形の物体に制御命令を伝え，三角形の物体は姿勢を制御しながら四角形の物体に近づく．これら一連の処理は全て 33ms 以内に行われ，ディジタル直線によるリアルタイムな形状認識を実証することができた．

図 3-25 は，カメラを露光したまま実験の様子を捉えたものであり，二等辺三角形が右回転することにより四角形の方向に向きを変え，さらに前進することにより四角形に近づいている様子を示している．また，図 3-26 は四角形の物体をリモートコントロールにより人間が操作できる状況での実験結果を示すものである．物体の形状認識により，三角形の物体が四角形の物体に近づいてくると，リモートコントロールにより四角形の物体を三角形の物体から遠ざけているが，四角形の物体が移動しても新たな位置を認識し，三角形の物体が自律的に姿勢を制御しながら四角形の物体を追跡している様子を理解することができる．

なお，3.3 及び 3.4 における研究は 1984 年に行ったものであり，使用している計算機（CPU 68000 を搭載する安立電気社製 Packet IIe）の処理能力は現在の計算機に比べてかなり低いものであるが，考案したアルゴリズムは計算機に依存するものではない．このため，現在の高速計算機を使用すれば CPU のクロック数で 100 倍以上の性能向上が見込めることから，表 3-2 に掲載している DL 法の平均直線近似時間（2 分 53 秒＝173 秒）は数秒で実現することが可能となり，非常に高速な直線近似方法を提供することができる．また，3.4 の実験もリアルタイムな形状認識を行うために直線近似方法を簡略化したが，3.3 の DL 法を適用し，ディジタル直線の性質を用いた移動物体の高速形状認識も可能となる．簡略化した直線近似方式でなく，ディジタル直線の性質を用いた近似方式（DL 法）を適用することにより，三角形や四角形などの単純な形状だけでなく，五角形や六角形，あるいは星形のような複雑な形状をリアルタイムで認識することも可能となる．

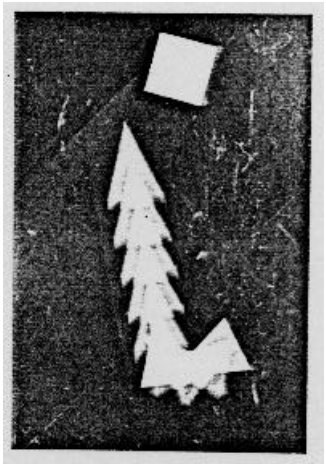


図 3-25 移動物体の認識（目標物は静止）

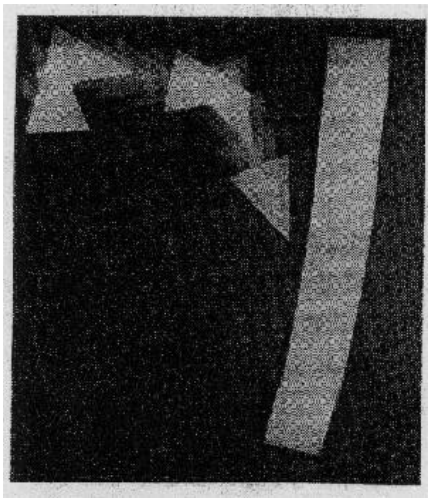


図 3-26 移動物体の認識（目標物も移動）

第四章　高速描画

第三章では，アナログ直線を量子化することによって得られるディジタル直線の性質を調べると共に，このディジタル直線の性質を利用して，物体境界部の直線近似，あるいは移動物体のリアルタイムな形状認識を行った．本章では，さらにディジタル直線の性質を利用することによって，コンピュータグラフィックスでの直線描画を高速に行う方法について議論する．また，コンピュータグラフィックスではクリップという概念があり，クリップ枠と呼ばれるある一定の矩形領域内に存在する部分のみを表示する必要がある．このため，描画する直線とクリップ枠を構成する直線の内，どの直線とが交差するのかを判定する必要があり，この判定を高速に行う方式について述べる．

4.1　直線描画方式

4.1.1　従来の直線描画方式

従来の代表的な直線描画方式としては，DDA(Digital Differential Analyzer)を挙げることができる[1,2]．DDA とは，直線の位置に対する微分値である傾きを計算し，描画すべき画素の位置を直線の傾き(微分値)の加算により，逐一求める方法である．アナログ直線をディジタル化する際の量子化規則は 3.1 に記載した方法と同一であり，ＸＹの各座標値を四捨五入することは各座標値に予め 0.5 を加算しておき，各座標値の小数部分を切り捨てることと同値となる．このことを考慮し，直線の始点を (X_0, Y_0)，終点を (X_1, Y_1) とすると，DDA による直線描画アルゴリズムは次の様に記述することができる．なお，図形の対称性を考慮して，図 3-3 におけるＡの領域について議論する．

＜DDA アルゴリズム＞

1) $L = X_1 - X_0, A = (X_1 - X_0)/L, B = (Y_1 - Y_0)/L$ を計算する．
2) $X = X_0 + 0.5, Y = Y_0 + 0.5, C = 0$ と初期設定する．
3) $(\lfloor X \rfloor, \lfloor Y \rfloor)$ を点描画し，$X = X + A, Y = Y + B, C = C + 1$ とする．
4) $C = L + 1$ なら終了．そうでなければ 3)に戻る．

上記方法は非常に単純であり，直線の傾きを加算するだけで自動的に描画すべき画素の位置を決定することができる．しかしながら，ディジタル直線の描画を考えると，次に描画すべき画素はＸ方向，Ｙ方向あるいは両方向に高々一つ移動するだけであり，図 3-3 のＡの領域を対象とする限り，上記Ａは常に１となる．このことを考慮して，Bresenham[11]は次の直線描画アルゴリズムを考案している．

＜Bresenham のアルゴリズム＞

1) $a = (Y_1 - Y_0)/(X_1 - X_0)$ を計算する．
2) $X = X_0, Y = Y_0, E = a - 0.5$ と初期設定する．
3) (X, Y) を点描画し，$E > 0$ なら $Y = Y + 1, E = E - 1$ とする．
4) $X = X + 1, E = E + a$ とし，$X = X_1 + 1$ なら終了．そうでなければ 3)に戻る．

Bresenham のアルゴリズムでは，常にX座標値を１更新しながらディジタル直線とアナログ直線との誤差Ｅを四捨五入の閾値である 0.5 と比較することにより，Y座標値を１更新するかどうかの判断を行っている．初期設定としてＥから 0.5 を引いておくことにより，Ｅを０と比較するだけでよく，非常に高速な処理が期待できる．さらに，(X, Y)は常に整数値を取るため，切り捨て関数処理が不要となる．しかしながら，ここでも問題点が一つ存在する．それは誤差Ｅの蓄積である．通常，直線の傾きは小数値となるため，有効桁数で切り捨てられた部分が蓄積し，最終的には直線の終点に到達しない可能性もある．そこで，整数値のみを用いて上記アルゴリズムを実行できる改良型 Bresenham アルゴリズムが提唱されている．基本的な考え方は，$\{(Y_1-Y_0)/(X_1-X_0)\}-(1/2)$ を０と比較しているので，$2(X_1-X_0)$ を乗算して $2(Y_1-Y_0)-(X_1-X_0)$ としても０との比較に影響は出ない．このとき，誤差Ｅに加算すべき値は $2(Y_1-Y_0)$ となり，１を引く操作は $2(X_1-X_0)$ を引く操作となることを考慮すれば，アルゴリズムは次のように記述することができる．

＜改良型 Bresenham アルゴリズム＞

1) $a=2(Y_1-Y_0), b=2(X_1-X_0)$ を計算する．
2) $X=X_0, Y=Y_0, E=a-(X_1-X_0)$ と初期設定する．
3) (X,Y)を点描画し，$E>0$ なら $Y=Y+1, E=E-b$ とする．
4) $X=X+1, E=E+a$ とし，$X=X_1+1$ なら終了．そうでなければ 3)に戻る．

上記アルゴリズムでは除算を一切用いず，整数演算のみで計算できるため，高速処理が可能なだけでなく，除算による切り捨て誤差の蓄積が全く発生しない．従って，通常は上記改良型 Bresenham アルゴリズムを用いて直線を描画するのが一般的である．

4.1.2　ディジタル直線を利用した高速直線描画方式

前節で説明した改良型 Bresenham アルゴリズムを用いると，整数演算のみで高速に，しかも誤差を蓄積することなく直線を描画することができる．しかしながら，描画すべき画素の位置を決定するためには，逐一Ｅを０と比較する必要があった．一方，前章で述べたように，ディジタル直線には趨勢方向が存在し，この方向に趨勢列の数だけ直線を構成する画素が同じチェイン符号を持って並んでいる．また，その並びの数は前節で証明したディジタル直線の性質から明確に計算できるため，このディジタル直線の性質を利用すれば，描画すべき画素の位置を逐一判断することなく，高速な直線描画を期待することができる．以下，前節同様，図形の対称性を考慮して，傾きが０から 45 度の範囲（図 3-3 の領域Ａ）に入る直線について，アルゴリズムを検討する．

前章 3. 2 で記載したディジタル直線の性質より，直線の傾きが $1/h$ となる場合は，$n=1, m=h$ と置き換えることにより，傾き n/m の場合に包含することができる．また，付録-2 の(A-9)及び(A-28)より，k 番目の直線の補正方向を与える画素のX座標値は，直線の傾き a が $0<a<1/2$ のときは $\lceil\{(2k-1)m\}/(2n)\rceil$，$1/2\le a<1$ のときは $\lfloor\{(2k-1)m\}/\{2(m-n)\}\rfloor+1$ と記述できるので，このことを考慮し，直線の始点を (X_0,Y_0)，終点を (X_1,Y_1) とすると，ディジタル直線の性質を利用した直線描画アルゴリズムは次のようになる（図 4-1 及び 4-2 参照）．

Ⅰ）直線の傾き $(a=0)$ **の場合**

1) $X=X_0, Y=Y_0$ と初期設定する．
2) (X,Y) を点描画し，$X=X+1$ とする．
3) $X=X_1+1$ なら終了．そうでなければ 2)に戻る．

Ⅱ）直線の傾き $(0<a<1/2)$ **の場合**

1) $m=(X_1-X_0), n=(Y_1-Y_0)$ を計算する．
2) $X=X_0, Y=Y_0, k=0, X_k=0$ と初期設定する．
3) $X_{k+1}=\lceil\{(2k+1)m\}/(2n)\rceil$ を計算し，(X,Y) から $(X+X_{k+1}-X_k-1, Y)$ まで $X_{k+1}-X_k$ 個の画素をチェイン符号 0 の方向に描画し，$X=X+X_{k+1}-X_k, Y=Y+1, k=k+1, X_k=X_{k+1}$ とする．
4) $k=n$ なら，(X_k,n) から (m,n) まで $m-X_k+1$ 個の画素をチェイン符号 0 の方向に描画して終了．そうでなければ 3)に戻る．

Ⅲ）直線の傾き $(1/2\le a<1)$ **の場合**

1) $m=(X_1-X_0), n=(Y_1-Y_0), n'=m-n$ を計算する．
2) $X=X_0, Y=Y_0, k=0, X_k=0$ と初期設定する．
3) $X_{k+1}=\lfloor\{(2k+1)m\}/(2n')\rfloor+1$ を計算し，(X,Y) から $(X+X_{k+1}-X_k-1, Y+X_{k+1}-X_k-1)$ まで $X_{k+1}-X_k$ 個の画素をチェイン符号 1 の方向に描画し，
$X=X+X_{k+1}-X_k, Y=Y+X_{k+1}-X_k-1, k=k+1, X_k=X_{k+1}$ とする．
4) $k=n'$ なら，(X_k, X_k-m+n) から (m,n) まで $m-X_k+1$ 個の画素をチェイン符号 1 の方向に描画して終了．そうでなければ 3)に戻る．

Ⅳ）直線の傾き $(a=1)$ **の場合**

1) $X=X_0, Y=Y_0$ と初期設定する．
2) (X,Y) を点描画し，$X=X+1, Y=Y+1$ とする．
3) $X=X_1+1$ なら終了．そうでなければ 2)に戻る．

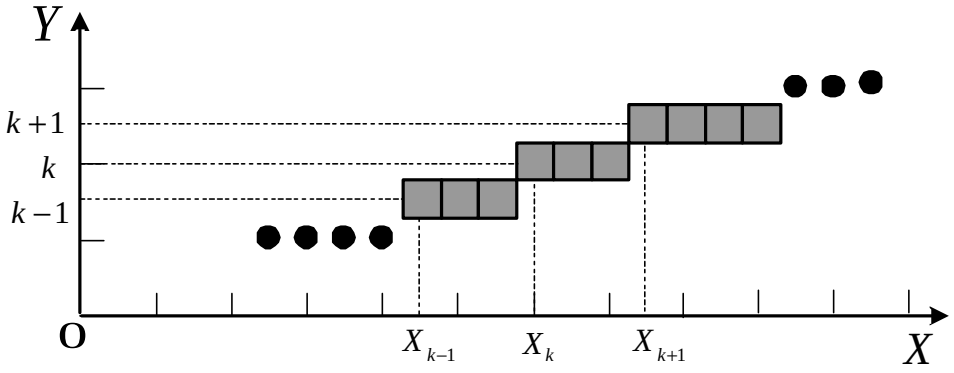


図 4-1 傾き $(0<a<1/2)$ の直線

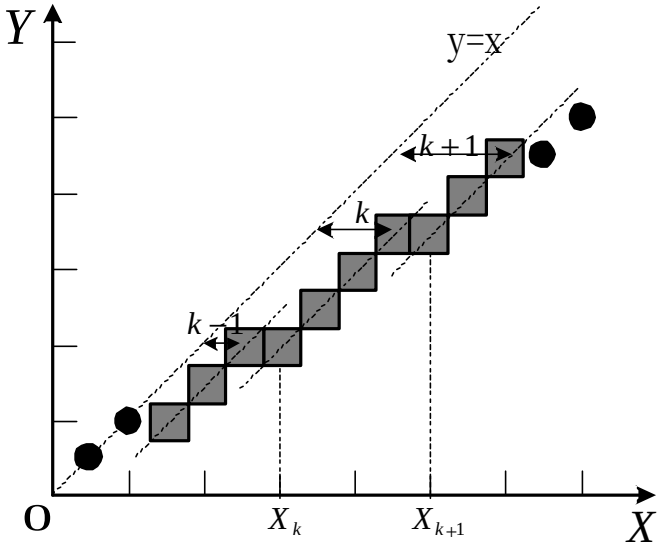


図 4-2 傾き $(1/2 \le a < 1)$ の直線

上記アルゴリズムを用いれば，描画すべき画素の位置を逐一判断する必要がなく，高速に直線を描画することができる．しかしながら，除算や切り上げ，あるいは切り捨て関数を使用しているため，これらの処理を改良型 Bresenham アルゴリズム同様，整数の演算処理に置き換えることができれば，さらなる高速処理が可能となる．そこで，上記アルゴリズムを整数の演算処理で実現可能かどうかを検討する．

本節では図形の対称性を考慮して，図 3-3 の領域Aについて議論している．従って，直線の傾き n/m は 1 以下の値であるが，その逆数 m/n は 1 以上の値となるため，m を n で除した際の商を Q_0，余りを R_0 とすれば，m/n は次のように表現することができる．

$$\frac{m}{n} = Q_0 + \frac{R_0}{n} \qquad (0 \le R_0 < n) \qquad \cdots\cdots (4-1)$$

また，直線の補正方向を与える画素のX座標値は，直線の傾き a が $0 < a < 1/2$ のとき，一般に $\lceil \{(2k-1)m\}/(2n) \rceil (0 < k \le n)$ と表記されるため，切り上げ関数の中にある数値も同様に，以下のように表現することができる．

$$\frac{(2k-1)m}{2n} = Q_k + \frac{R_k}{2n} \qquad (0 \le R_k < 2n) \qquad \cdots\cdots (4-2)$$

特に，$k = 1$ の場合について考えると，

$$\frac{m}{2n} = Q_1 + \frac{R_1}{2n} \qquad (0 \le R_1 < 2n) \qquad \cdots\cdots (4-3)$$

と表され，本式を式(4-1)と比較すると，次のことが言える．

$$Q_1 = \left\lfloor \frac{Q_0}{2} \right\rfloor, R_1 = \begin{cases} R_0 & (Q_0\text{は偶数}) \\ R_0 + n & (Q_0\text{は奇数}) \end{cases} \quad \cdots\cdots(4-4)$$

まず，$R_0 \neq 0$ 場合について考える．式(4-4)より $R_1 \neq 0$ が成り立ち，式(4-2)，(4-3)より，

$$Q_k + \frac{R_k}{2n} = \frac{(2k-1)m}{2n} = (2k-1)\frac{m}{2n} = (2k-1)(Q_1 + \frac{R_1}{2n}) = (2k-1)Q_1 + \frac{(2k-1)R_1}{2n} \quad \cdots\cdots(4-5)$$

となる．従って，$R_k = (2k-1)R_1$ であり，$R_1 \neq 0$ から，$R_k \neq 0$ も成り立つ．すると，k 番目の直線の補正方向を与える画素のX座標値 X_k は次のようになる．

$$X_k = \left\lceil \frac{(2k-1)m}{2n} \right\rceil = \left\lceil Q_k + \frac{R_k}{2n} \right\rceil = Q_k + 1 \quad (\because R_k \neq 0) \quad \cdots\cdots(4-6)$$

次に，$k+1$ 番目の直線の補正方向を与える画素のX座標値 X_{k+1} を考えると，次の漸化式が成り立つ．

$$X_{k+1} = \left\lceil \frac{\{2(k+1)-1\}m}{2n} \right\rceil = \left\lceil \frac{(2k-1)m}{2n} + \frac{2m}{2n} \right\rceil = \left\lceil Q_k + \frac{R_k}{2n} + Q_0 + \frac{2R_0}{2n} \right\rceil$$

$$= \left\lceil (Q_k + Q_0) + \frac{R_k + 2R_0}{2n} \right\rceil = \begin{cases} Q_k + Q_0 + 1 & (0 < R_k + 2R_0 \leq 2n) \\ Q_k + Q_0 + 2 & (R_k + 2R_0 > 2n) \end{cases} \quad \cdots\cdots(4-7)$$

ここで、$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k + 2R_0 \quad \cdots\cdots(4-8)$

となる。

従って，X_k から $X_{k+1} - 1$ までの画素列の長さ L_k は，式(4-6)及び(4-7)より次のようになる．

$$L_k = \mathrm{X}_{k+1} - X_k = \begin{cases} Q_0 & (R_{k+1} \leq 2n) \\ Q_0 + 1 & (R_{k+1} > 2n) \end{cases} \quad \cdots\cdots(4-9)$$

また，最初の画素列の長さ L_0 は，式(4-6)より次のようになる．

$$L_0 = X_1 - 0 = Q_1 + 1 \quad \cdots\cdots(4-10)$$

一方，直線の終点に至る最後の画素列の長さ L_n は，n 番目の直線の補正方向を与える画素のX座標値 X_n を計算することにより，次のように得られる．

$$X_n = \left\lceil \frac{(2n-1)m}{2n} \right\rceil = \left\lceil m - \frac{m}{2n} \right\rceil = \left\lceil m - (Q_1 + \frac{R_1}{2n}) \right\rceil = \left\lceil (m - Q_1) - \frac{R_1}{2n} \right\rceil$$

$$= m - Q_1 \quad (\because R_1 < 2n) \quad \cdots\cdots(4-11)$$

$$\therefore L_n = m - X_n + 1 = m - (m - Q_1) + 1 = Q_1 + 1 \quad \cdots\cdots(4-12)$$

従って，式(4-8)，(4-9)，(4-10)，(4-12)より，$R_0 \neq 0$ 場合についてまとめると次のようになる．

$$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k + 2R_0 \quad (0 < k < n-1)$$

$$L_0 = L_n = Q_1 + 1$$

$$L_k = \begin{cases} Q_0 & (R_{k+1} \leq 2n) \\ Q_0 + 1 & (R_{k+1} > 2n) \end{cases} \quad \cdots\cdots(4-13)$$

次に，$R_0 = 0$場合について考える．式(4-6)より，

$$X_k = \left\lceil \frac{(2k-1)m}{2n} \right\rceil = \left\lceil Q_k + \frac{R_k}{2n} \right\rceil = \begin{cases} Q_k & (R_k = 0) \\ Q_k + 1 & (R_k \neq 0) \end{cases} \quad \cdots\cdots(4-14)$$

となる．また，式(4-7)より，

$$X_{k+1} = \left\lceil \frac{\{2(k+1)-1\}m}{2n} \right\rceil = \left\lceil \frac{(2k-1)m}{2n} + \frac{2m}{2n} \right\rceil = \left\lceil Q_k + \frac{R_k}{2n} + Q_0 + \frac{2R_0}{2n} \right\rceil$$

$$= \left\lceil (Q_k + Q_0) + \frac{R_k}{2n} \right\rceil = \begin{cases} Q_k + Q_0 & (R_k = 0) \\ Q_k + Q_0 + 1 & (R_k \neq 0) \end{cases} (\because R_0 = 0) \quad \cdots\cdots(4-15)$$

$$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k = \ldots = R_1 \quad \cdots\cdots(4-16)$$

従って，式(4-14)，(4-15)より，趨勢方向の画素列を表す長さは一定で，

$$L_k = \mathrm{X}_{k+1} - X_k = Q_0 \quad (0 < k < n) \quad \cdots\cdots(4-17)$$

となる．また，最初の画素列の長さL_0は，式(4-14)より次のようになる．

$$L_0 = X_1 - 0 = \begin{cases} Q_1 & (R_1 = 0) \\ Q_1 + 1 & (R_1 \neq 0) \end{cases} \quad \cdots\cdots(4-18)$$

一方，直線の終点に至る最後の画素列の長さL_nは，n番目の直線の補正方向を与える画素のX座標値X_nを計算することにより，次のように得られる．

$$X_n = \left\lceil \frac{(2n-1)m}{2n} \right\rceil = \left\lceil m - \frac{m}{2n} \right\rceil = \left\lceil m - (Q_1 + \frac{R_1}{2n}) \right\rceil = \left\lceil (m - Q_1) - \frac{R_1}{2n} \right\rceil$$

$$= m - Q_1 \quad (\because R_1 < 2n) \quad \cdots\cdots(4-19)$$

$$\therefore L_n = m - X_n + 1 = m - (m - Q_1) + 1 = Q_1 + 1 \quad \cdots\cdots(4-20)$$

従って，式(4-16)，(4-17)，(4-18)及び(4-20)より，$R_0 = 0$場合についてまとめると次のようになる．

$$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k = R_1 \quad (0 < k < n-1)$$

$$L_0 = \begin{cases} Q_1 & (R_1 = 0) \\ Q_1 + 1 & (R_1 \neq 0) \end{cases}, \quad L_n = Q_1 + 1$$

$$L_k = Q_0 \quad (0 < k < n) \quad \cdots\cdots(4-21)$$

次に，直線の傾きaが$1/2 \le a < 1$の場合について考える．$m - n = n'$とすると，式(4-1)同様，次の様に表現できる．

$$\frac{m}{m-n} = \frac{m}{n'} = Q_0 + \frac{R_0}{n'} \quad (0 \le R_0 < n') \quad \cdots\cdots(4-22)$$

また，直線の補正方向を与える画素のX座標値は，一般に$\left\lfloor \{(2k-1)m\}/(2n') \right\rfloor + 1$と表されるため，切り捨て関数の中にある数値も同様に，以下のように表現することができる．

$$\frac{(2k-1)m}{2n'} = Q_k + \frac{R_k}{2n'} \quad (0 \le R_k < 2n') \qquad (4-23)$$

特に，$k=1$の場合について考えると，

$$\frac{m}{2n'} = Q_1 + \frac{R_1}{2n'} \quad (0 \le R_1 < 2n') \qquad (4-24)$$

と表され，本式を式(4-22)と比較すると，次のことが言える．

$$Q_1 = \left\lfloor \frac{Q_0}{2} \right\rfloor, R_1 = \begin{cases} R_0 & (Q_0\text{は偶数}) \\ R_0 + n' & (Q_0\text{は奇数}) \end{cases} \qquad (4-25)$$

次に，k番目の直線の補正方向を与える画素のX座標値X_kを計算すると，式(4-23)より次のようになる．

$$X_k = \left\lfloor \frac{(2k-1)m}{2n'} \right\rfloor + 1 = \left\lfloor Q_k + \frac{R_k}{2n'} \right\rfloor + 1 = Q_k + 1 \qquad (4-26)$$

また，$k+1$番目の直線の補正方向を与える画素のX座標値X_{k+1}は，式(4-22)及び(4-26)より次のようになり，漸化式が得られる．

$$X_{k+1} = \left\lfloor \frac{\{2(k+1)-1\}m}{2n'} \right\rfloor + 1 = \left\lfloor \frac{(2k-1)m}{2n'} + \frac{2m}{2n'} \right\rfloor + 1 = \left\lfloor Q_k + \frac{R_k}{2n'} + Q_0 + \frac{2R_0}{2n'} \right\rfloor + 1$$

$$= \left\lfloor (Q_k + Q_0) + \frac{R_k + 2R_0}{2n'} \right\rfloor + 1 = \begin{cases} Q_k + Q_0 + 1 & (R_k + 2R_0 < 2n') \\ Q_k + Q_0 + 2 & (R_k + 2R_0 \ge 2n') \end{cases} \qquad (4-27)$$

$$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k + 2R_0 \qquad (4-28)$$

従って，X_kから$X_{k+1}-1$までの画素列の長さL_kは，式(4-26)及び(4-27)より次のようになる．

$$L_k = \mathrm{X}_{k+1} - X_k = \begin{cases} Q_0 & (R_{k+1} < 2n') \\ Q_0 + 1 & (R_{k+1} \ge 2n') \end{cases} \qquad (4-29)$$

また，最初の画素列の長さL_0は式(4-26)より，

$$L_0 = X_1 - 0 = Q_1 + 1 \qquad (4-30)$$

となる．一方，直線の終点に至る最後の画素列の長さ$L_{n'}$はn'番目の直線の補正方向を与える画素のX座標値$X_{n'}$を計算することにより，次のように得られる．

$$X_{n'} = \left\lfloor \frac{(2n'-1)m}{2n'} \right\rfloor + 1 = \left\lfloor m - \frac{m}{2n'} \right\rfloor + 1 = \left\lfloor m - (Q_1 + \frac{R_1}{2n'}) \right\rfloor + 1 = \left\lfloor (m - Q_1) - \frac{R_1}{2n'} \right\rfloor + 1$$

$$= \begin{cases} m - Q_1 + 1 & (R_1 = 0) \\ m - Q_1 & (R_1 \ne 0) \end{cases} \qquad (4-31)$$

$$\therefore L_{n'} = m - X_{n'} + 1 = \begin{cases} Q_1 & (R_1 = 0) \\ Q_1 + 1 & (R_1 \ne 0) \end{cases} \qquad (4-32)$$

従って，式(4-28)，(4-29)，(4-30)及び(4-32)より，直線の傾きa が$1/2 \le a \le 1$の場合についてまとめると次のようになる．

$$Q_{k+1} = Q_k + Q_0, R_{k+1} = R_k + 2R_0 \qquad (0 < k < n' - 1)$$

$$L_0 = Q_1 + 1, L_{n'} = \begin{cases} Q_1 & (R_1 = 0) \\ Q_1 + 1 & (R_1 \neq 0) \end{cases}$$

$$L_k = \begin{cases} Q_0 & (R_{k+1} < 2n') \\ Q_0 + 1 & (R_{k+1} \geq 2n') \end{cases} \qquad (0 < k < n') \quad \cdots\cdots (4-33)$$

ここでもし，$R_0 = 0$とすると，式(4-24)及び(4-28)より，$R_{k+1} = R_k + 2R_0 = R_k = \ldots = R_1 < 2n'$であるから，$L_k = Q_0 =$一定となる．

上記結果に対して，直線の傾きn/mが0ということは$n = 0$と同値，$n/m = 1/2$ということは$2n = m$と同値，$n/m = 1$ということは$n = m$と同値であることを考慮すると，図 3-3 の領域Aを対象とした，ディジタル直線の性質を用いた高速直線描画アルゴリズムは図 4-3～4-5 のようになる．なお，$\mathrm{int}(x)$はxの小数点以下を切り捨てる関数，$x\%y$はxをyで除した際の余りを求める演算子である．

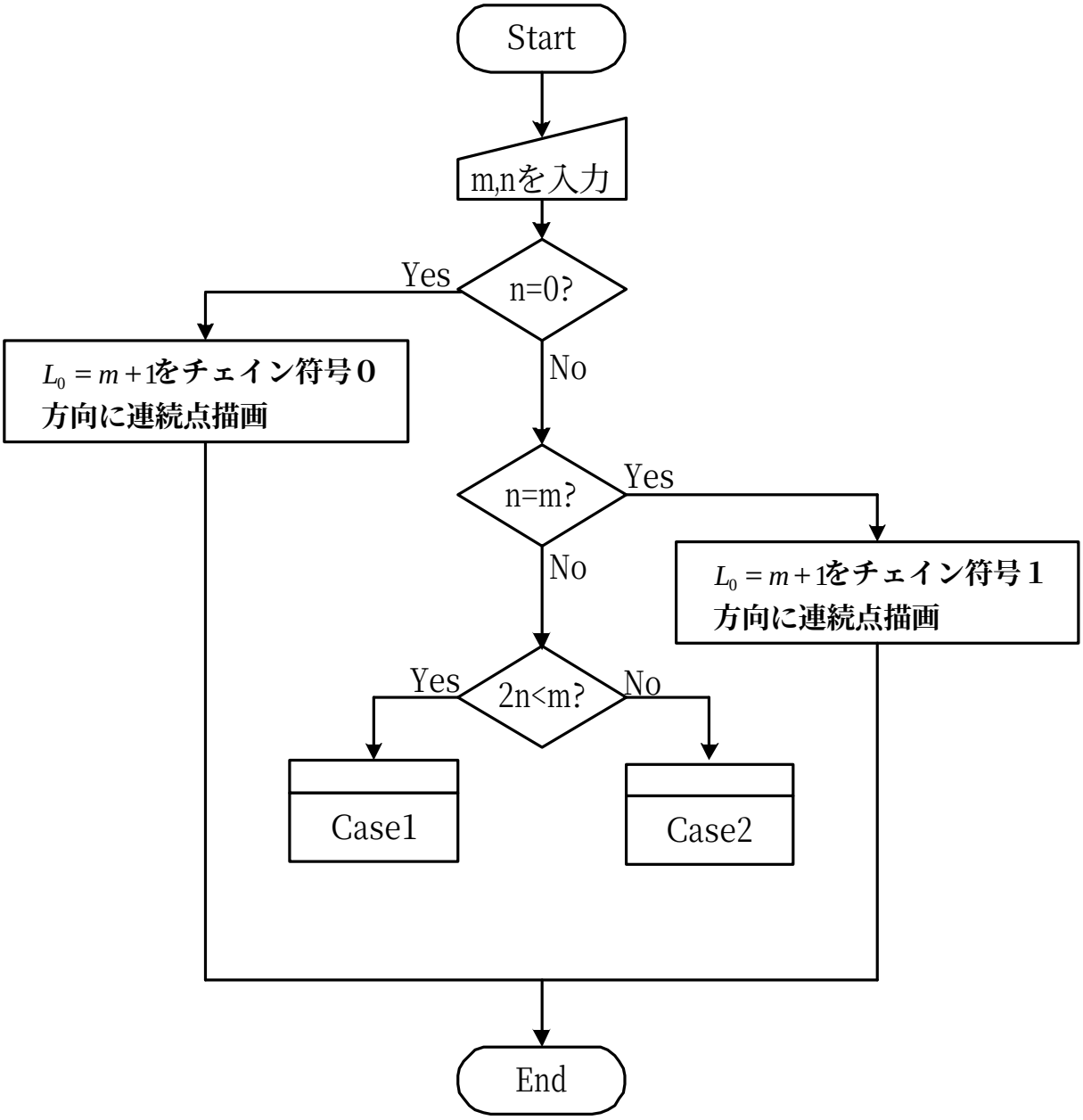


図 4-3 ディジタル直線の性質を利用した高速直線描画アルゴリズム（全体処理）

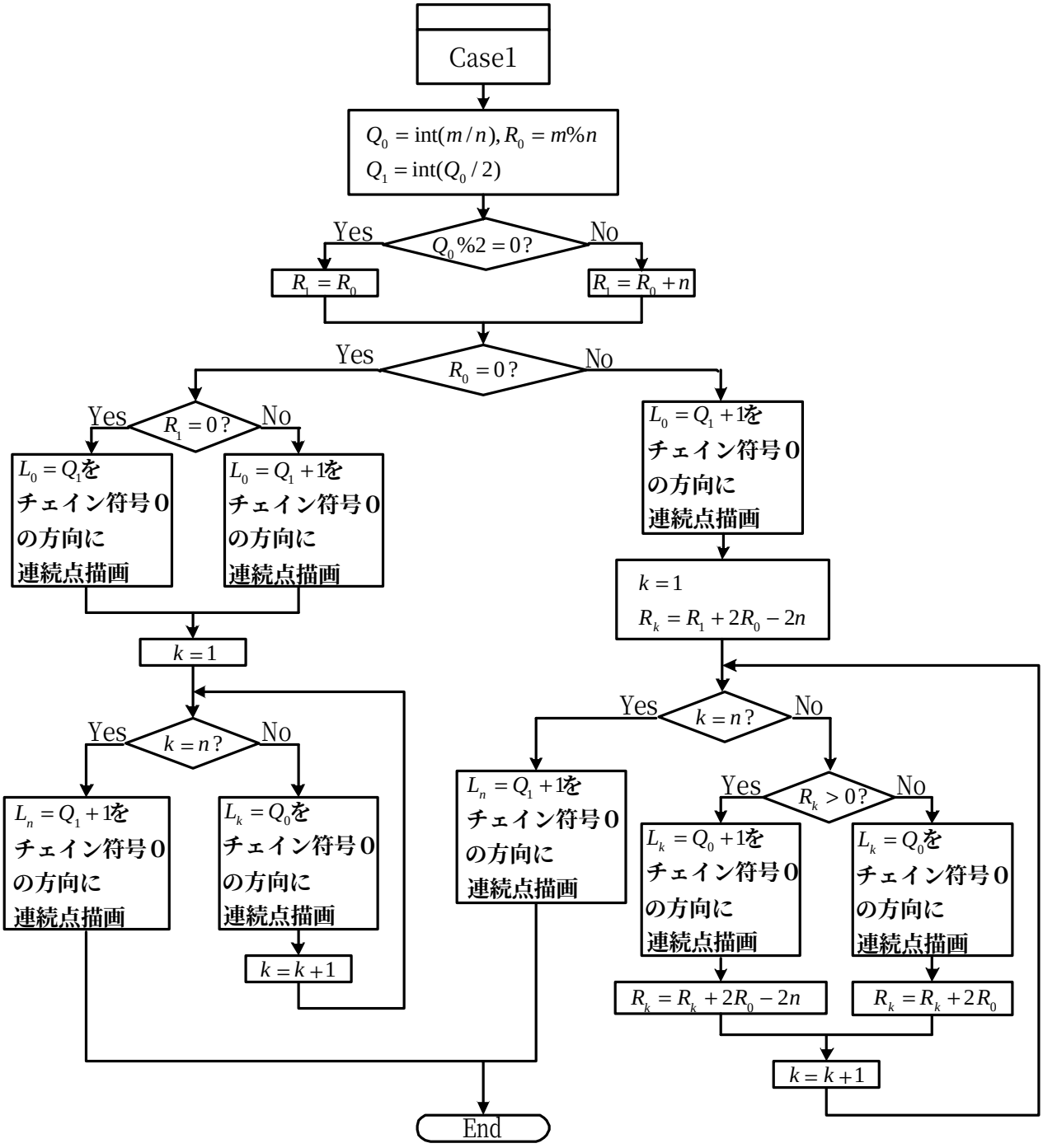


図 4-4 ディジタル直線の性質を利用した高速直線描画アルゴリズム（傾き 1/2 未満の場合）

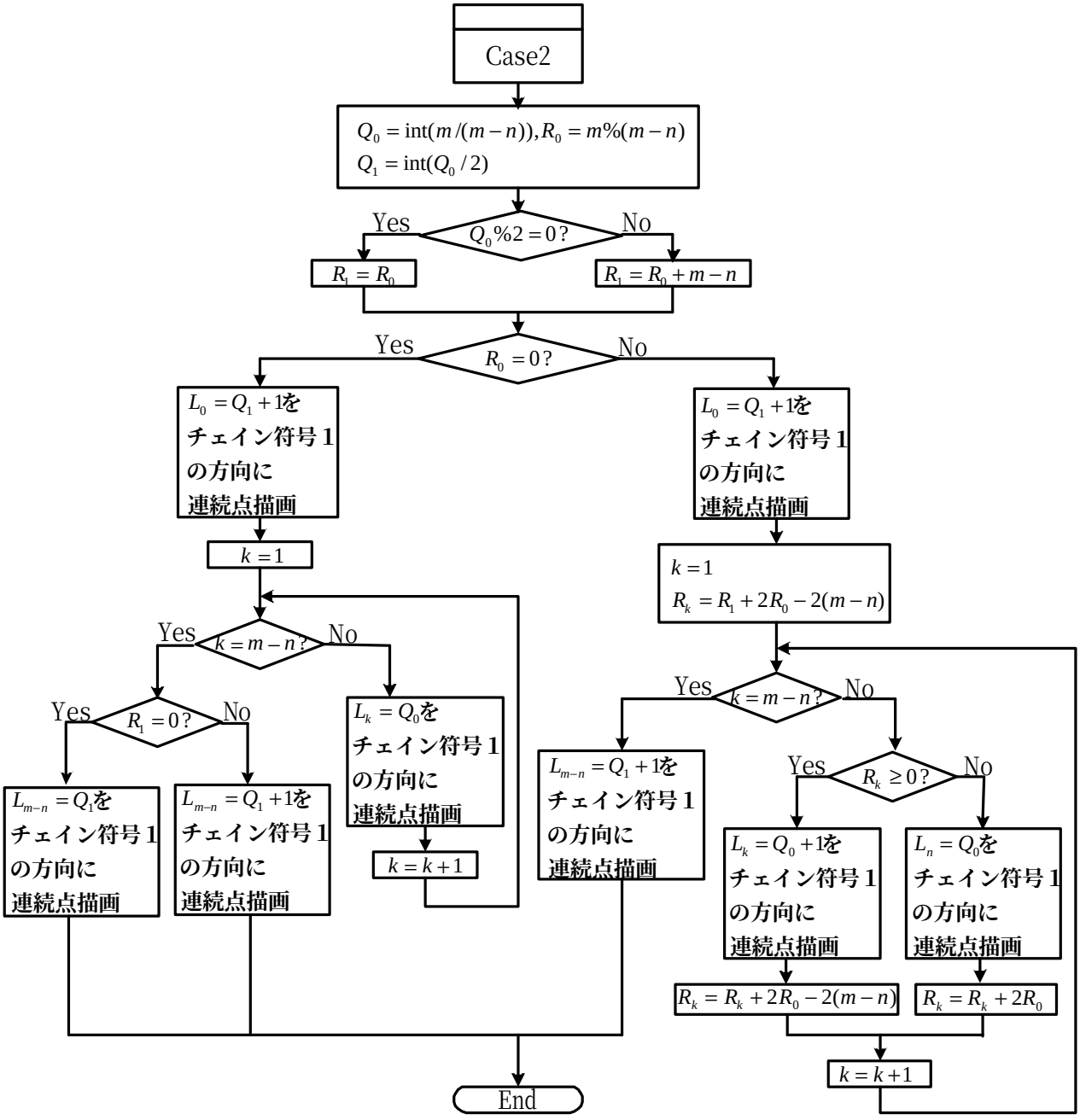


図 4-5 ディジタル直線の性質を利用した高速直線描画アルゴリズム（傾き 1/2 以上の場合）

4.2 直線のクリップ

コンピュータグラフィックスの用途は工業製品製作のための CAD/CAM，映画/ゲームなどのエンターテインメント，教育用エデュテイメント，医療あるいは産業用各種シミュレータなど様々であるが，いずれの分野をとっても扱うグラフィックスデータの量は非常に多い．これらの大規模データを一度に全てを表示していたのでは，いくら最近のハードウェア技術が進歩していてもリアルタイムな表示は困難である．一方，コンピュータグラフィックスの応用分野からすると，構築した大規模データを一度に全てを見ることは稀であり，大規模なデータを構築してもその一部を拡大表示して見る

ことが多い．つまり，全体を見るときは各部の詳細なデータは不要であり，全体的なデータの配置状況が分かればよく，詳細データを見るのは一部を拡大表示するときである．全体データの概略状況をリアルタイムに把握するためには大規模データをそのまま表示するのではなく，詳細データを簡略化してグラフィックス表示データを削減する必要があると同時に，一部の領域を詳しく表示することを可能とするための詳細データも保持しておく必要がある．つまり，データの階層化が重要となる．一方，一部の領域を詳しく表示する際には，全てのグラフィックスデータを表示対象とする必要はなく，表示される領域に入るデータを高速に抽出すればよい．この表示領域に入るグラフィックスデータの抽出処理をクリップと呼び，コンピュータグラフィックスの高速表示に重要な役割を果たす．データの階層化については次章で述べることにし，本節では高速表示に重要なクリップ処理について論ずることにする．

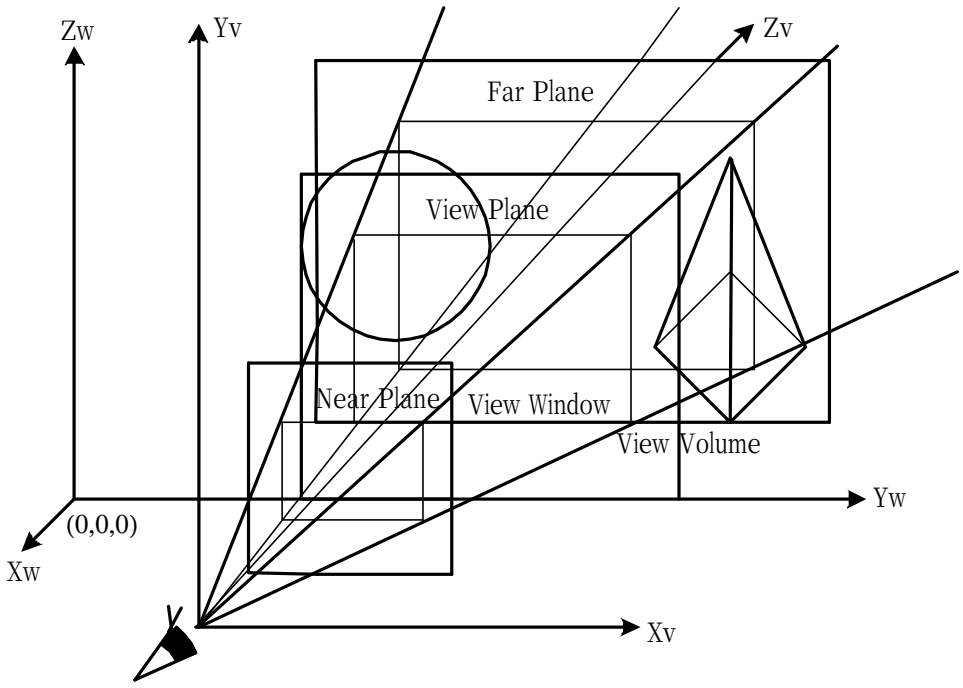


図 4-6 ３次元コンピュータグラフィックスのクリップ処理

第二章でも記述したように，コンピュータグラフィックスで扱うデータは最初，モデリング座標系上で定義され，各モデルデータをモデリング変換によりワールド座標系上に配置した後，視野変換により視点座標系のデータとして記述される．視点座標系は視点を原点とした座標系であり，変換されたデータのどの部分を表示するのかを決定するためのクリップ処理が，この視点座標系上で行われる（図 2-3 参照）．図 4-6 に示すように，視点を頂点としたピラミッドのうち，視点に最も近い平面（Near Plane）と視点から最も離れている面（Far Plane）との間に挟まれた部分（View Volume）が表示対象となり，表示対象範囲内に存在する物体（あるいは物体の一部）が View Plane に投影されて，最終的な画像が表示される．ここで，表示対象となる View Plane は平面であり，この View Plane のうち，View Volume により切り取られた部分（View Window）が，デバイス座標系におけるクリップ枠となる[11]．３次元コンピュータグラフィックスにおける View Volume は一般に台形体となるため，側面と表示物体との交点計算に多大な計算時間を要する．そこで，視点座標系上のデータを全

て，一辺が単位長さ(1.0)となる立方体形状の座標系（**正規化視点座標系**）上のデータに変換した後，クリップ処理を行う場合もある．この場合，View Volume を構成する面は全て視点座標系を構成する各軸(X,Y,Z)と直交し，クリップ処理は容易となる．あるいは，View Volume を対象とした３次元クリップ処理は表示物体を包含する矩形体（**バウンディングボックス**）で概略のクリップ処理を行い，少なくともその一部が表示される物体は全て，透視変換によりデバイス座標系上のデータに変換した後，View Window を対象とした２次元クリップ処理に帰着させる場合が多い．従ってここでは，コンピュータグラフィックスにおいて表示対象となるデータに対し，View Window（つまりクリップ枠）内に存在する物体，あるいは物体の一部を高速に抽出する方式について検討する．

4.2.1 従来のクリップ方式

２次元クリップ処理で最も有名な方法は，Cohen-Sutherland(CS)の手法[1,12]である．

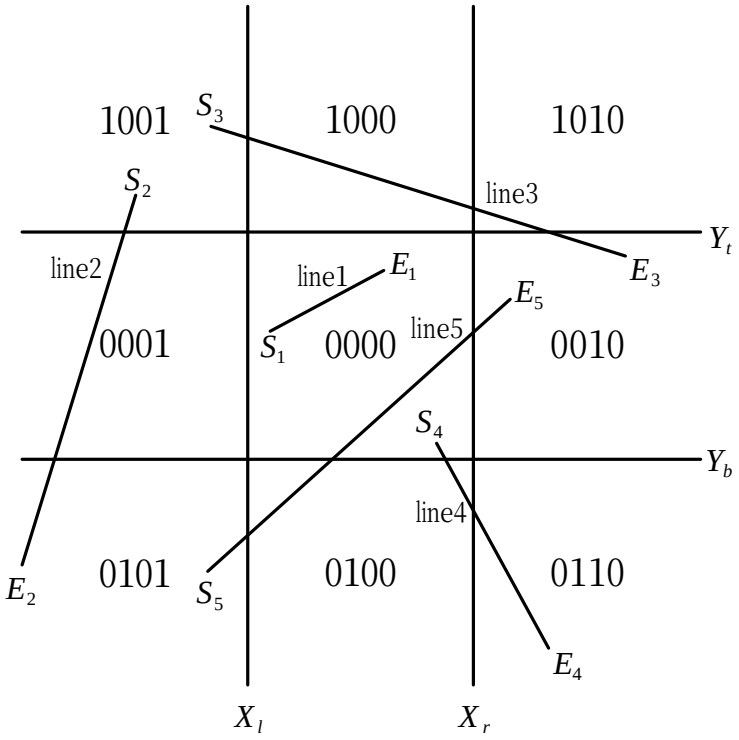


図 4-7 Cohen-Sutherland のクリップ方法

CS 法では，図 4-7 に示すように２次元デバイス座標系において，クリップ枠を構成する４本の直線を $X = X_l$，$X = X_r$，$Y = Y_b$ 及び $Y = Y_t$ とし，直線の両端点（X,Y）に対して，それぞれ次の符号（ABCD）を求める．

$Y \le Y_t$ なら A=0，そうでなければ A=1
$Y \ge Y_b$ なら B=0，そうでなければ B=1
$X \le X_r$ なら C=0，そうでなければ C=1
$X \ge X_l$ なら D=0，そうでなければ D=1

すると，クリップ枠を構成する４本の直線により９つの領域に分割された２次元空間の各領域に対し，図 4-7 に示す符号が得られる．左端の符号Ａ＝１は対象となる点が$Y = Y_t$よりも上側にあることを，次の符号Ｂ＝１は対象となる点が$Y = Y_b$よりも下側にあることを，その次の符号Ｃ＝１は対象となる点が$X = X_r$よりも右側にあることを，最後の符号Ｄ＝１は対象となる点が$X = X_l$よりも左側にあることを示す．図 4-7 には様々な直線を示しているが，例えば line1 の両端点S_1及びE_1に対する符号は共に(0000)となり，line2 のS_2に対する符号は(1001)，E_2に対する符号は(0101)となる．従って，次のことが言える．

＜クリップ処理アルゴリズム＞

1) 直線の両端点に対する符号が共に(0000)であれば，その直線はクリップ枠内に存在する．
2) 直線の両端点に対する符号のビット毎の AND が(0000)でなければ，その直線はクリップ枠を構成する４本の直線の内，いずれかの直線よりもクリップ枠の外側に存在する．
3) 上記いずれにも当てはまらない場合は，直線を分割してさらに詳しく調べる．

上記アルゴリズムの 3)において，さらに詳しく調べるための分割には様々な方法がある．一つは Clipping Divider[12]であり，直線の両端点の平均を取ることにより，直線の中点を求めて直線を二分割する．二分割された各直線に対し，上記アルゴリズムを適用して 1)あるいは 2)により，直線がクリップ枠内に存在するか，あるいはクリップ枠外に存在するかが判断されるまで繰り返す．直線の中点を求める手法は非常に単純であり，ハードウェア化が容易であることから高速処理が期待できる．しかしながら，この中点分割方法では直線のクリップ処理を行うためには数回の中点計算が必要となり，最終的に得られた直線の端点には数回の中点計算による誤差が蓄積されている．一方，直線がクリップ処理により切断される点は直線とクリップ枠との交点であるため，直線の中点を求めるのではなく，直接，クリップ枠との交点計算を行う方法が提案されている[2]．アルゴリズムのフローチャートを図 4-8 に示す．

図 4-8 に示すアルゴリズムを用いれば，直線とクリップ枠との交点を直接計算することにより，計算誤差の蓄積を抑えた精度の高いクリップ処理が実行できる．しかしながら，図 4-8 のアルゴリズムにも短所は存在する．図のアルゴリズムに従えば，直線がクリップインでもクリップアウトでもない場合，必ず直線の端点のどちらかはクリップ枠の外にあるため，その点を対象として，直線がどのクリップ枠と交わるのかを判断しているが，不必要な交点計算を伴うことがある．例えば，図 4-7 の line3 では最初のループで$X = X_l$との，次のループで$X = X_r$との交点を計算するが，最終的にはクリップアウトと判断され，非表示となる．また，line5 の場合，アルゴリズムに従うと，$X = X_l$，$X = X_r$及び$Y = Y_b$に対して３回の交点計算を行うが，図より判断すれば$X = X_r$及び$Y = Y_b$に対して２回の交点計算でよいことが分かる．

そこで，直線を構成する両端点の２次元座標系上における位置を考慮しながら，最小の交点計算で行えるクリップ処理方法が，二人の Nicholl と Lee によって考案されている[13]．この方法では，クリップ枠を構成する４本の直線とクリップ対象となる直線との交点計算を行う前に，クリップ対象となる直線の始点とクリップ枠のコーナー点とを結ぶ直線を考え，この直線とクリップ対象となる直線とを比較することにより，クリップ対象となる直線がクリップ枠を構成する直線の内、どの直線と交差しているのかを判断して最小限の交点計算でクリップ処理を実行している．

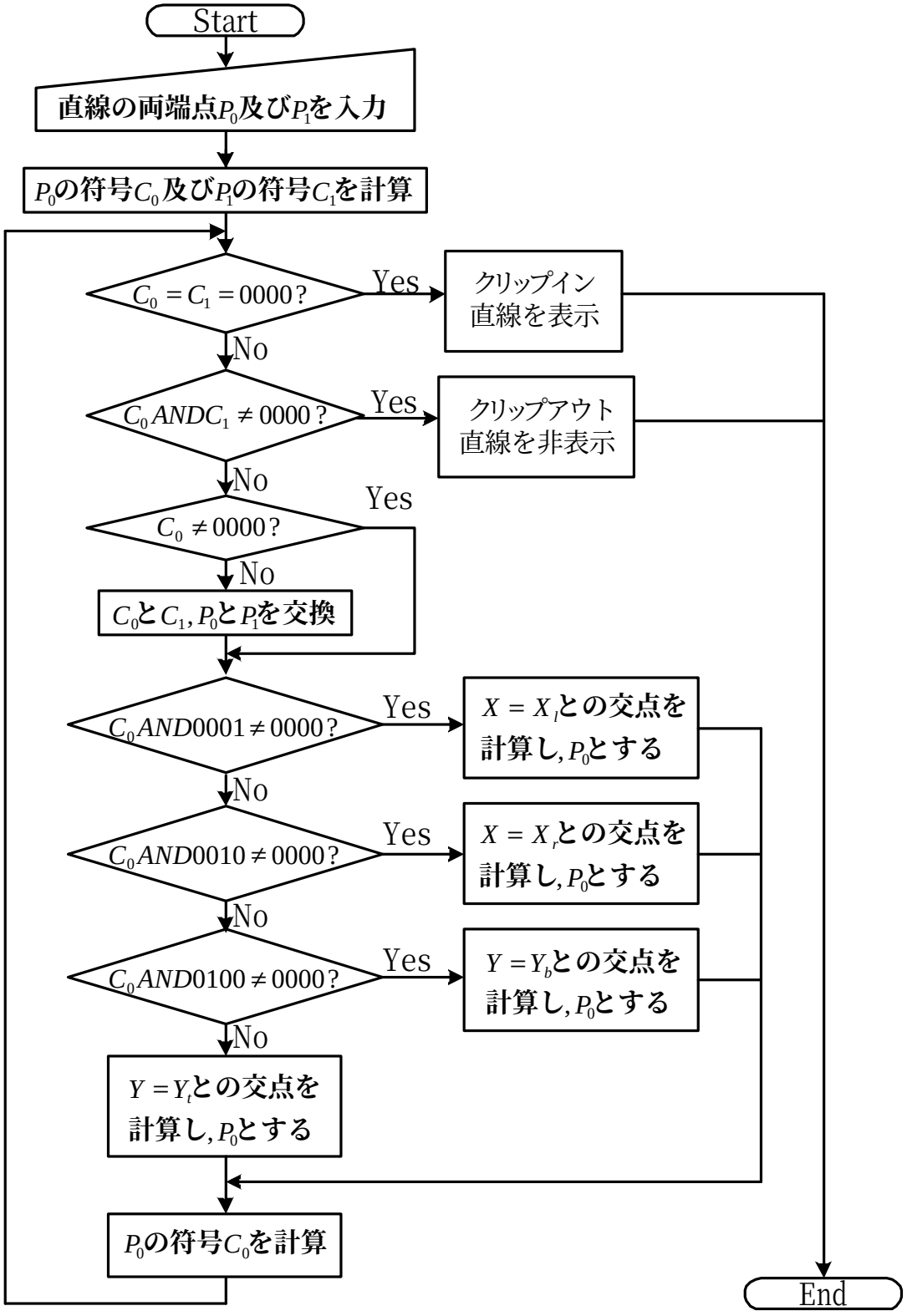


図 4-8 クリップ処理フローチャート

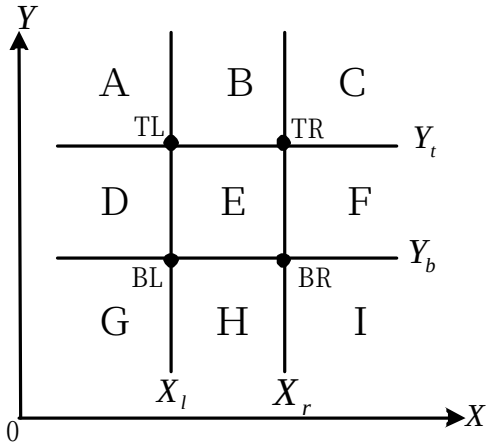


図 4-9 クリップ領域の分類

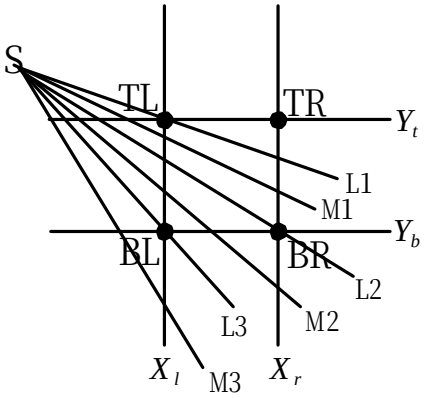


図 4-10 領域Aのクリップ処理

Nicholl-Lee-Nicholl の方法(NLN 法)では，まずクリップ枠を構成する４本の直線により分割される９つの領域を図 4-9 に示すように区分し，図形の対称性を考慮することにより，領域A，D及びEのみを検討対象とする．例えば，領域BはTLとBRを結ぶ直線(y=-x)に対して領域Dと対称となっているため，この直線に対して折り返すことにより領域Dに帰着することができる．また，領域Cは図形を反時計回りに 90 度回転することにより，領域Aに帰着できる．領域AあるいはDに帰着してクリップ処理を行った後，逆の手順を辿ることにより元の図形に戻すことができる．

直線の始点座標を(X_0,Y_0)，終点座標を(X_1,Y_1)とすると，クリップ処理アルゴリズムは次の様になる．

＜NLN 法のクリップアルゴリズム＞

1) $X_0 < X_l$なら，LeftColumn へ．
2) $X_0 > X_r$なら，図形を 180 度回転させて LeftColumn で処理後，逆方向に 180 度回転させて元に戻す．
3) 上記いずれでもなければ CenterColumn へ．

＜LeftColumn＞

1) $X_1 < X_l$なら，直線はクリップアウト．
2) $Y_0 > Y_t$なら，TopLeftCorner へ．
3) $Y_0 < Y_b$なら，$Y = (Y_t + Y_b)/2$で折り返して TopLeftCorner で処理後，逆方向に折り返して元に戻す．
4) 上記いずれでもなければ LeftEdge（領域Dの処理ルーチン）へ．

＜TopLeftCorner＞

1) $Y_1 > Y_t$ なら，クリップアウト．
2) 始点（$S(X_0,Y_0)$）とクリップ枠の左上コーナー点(TL)とを結ぶ直線(図 4-10 の L1)を考え，クリップ対象となる直線が L1 より下にあれば，つまり，$(Y_t - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば，LeftBottomRegion へ．
3) クリップ対象となる直線が L1 よりも上にあれば，y=-x で折り返して LeftBottomRegion で処理後，逆手順で元に戻す．

＜LeftBottomRegion＞

1) $Y_1 > Y_b$ なら，図 4-10 の L1 と L3 に挟まれる直線(M1 あるいは M2)となり，$X_0 < X_l$ であるから $X = X_l$ との交点計算を行う．さらに，$X_1 > X_r$ を満足していれば $X = X_r$ との交点計算も行い，対象となる直線はクリップインとなる．
2) $X_1 \le X_r$ の場合は，始点(S)とクリップ枠の左下コーナー点(BL)とを結ぶ直線(図 4-10 の L3)を考え，クリップ対象となる直線が L3 より下にあれば，つまり，
$(Y_b - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば，直線はクリップアウトとなる．クリップ対象となる直線が L3 の上にある場合は $X = X_l$ との交点計算を行っており，$Y_1 > Y_b$ かつ $X_1 \le X_r$ であるから，対象となる直線はクリップインとなる．
3) $Y_1 > Y_b$ でない場合は，$Y_1 \le Y_b$ が成り立つ．従って，$X_1 < X_r$ であれば，$Y = Y_b$ との交点計算を行う．$X_1 \ge X_r$ の場合は，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線(図 4-10 の L2)を考える．直線が L2 の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば $Y = Y_b$ と，そうでなければ $X = X_r$ との交点計算を行う．いずれの場合にも，$X_0 < X_l$ であるから，$X = X_l$ との交点計算を行い，対象となる直線はクリップインとなる．

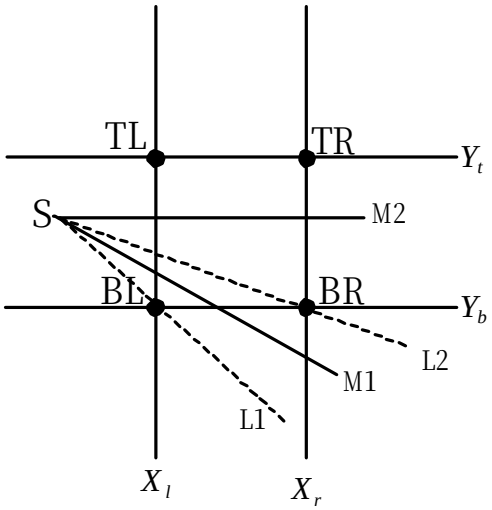


図 4-11 領域Dのクリップ処理

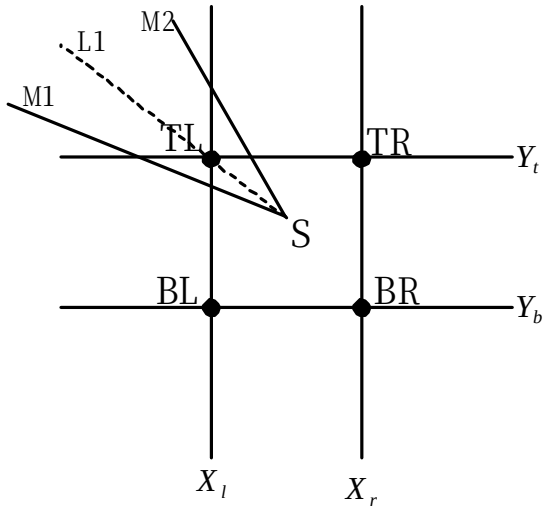


図 4-12 領域Eのクリップ処理

＜LeftEdge＞

1) $X_1 < X_l$ なら，直線はクリップアウト．
2) $Y_1 < Y_b$ なら，P2Bottom へ．
3) $Y_1 > Y_t$ なら，$Y = (Y_t + Y_b)/2$ で折り返して P2Bottom で処理後，逆方向に折り返して元に戻す．
4) 上記いずれでもなければ，$Y_b < Y_1 < Y_t$ となり，図 4-11 の M2 となる．従って，$X_0 < X_l$ であるから $X = X_l$ との交点計算を行うと共に，$X_1 > X_r$ であれば $X = X_r$ との交点計算も行い，対象となる直線はクリップインとなる．

＜P2Bottom＞

1) 始点(S)とクリップ枠の左下コーナー点(BL)とを結ぶ直線(図 4-11 の L1)を考え，クリップ対象となる直線が L1 より下にあれば，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば，直線はクリップアウトとなる．
2) クリップ対象となる直線が L1 より上にある場合，<LeftEdge>の 2)より，$Y_1 < Y_b$ が成り立っているから，$X_1 < X_r$ なら $Y = Y_b$ との交点計算を行う．$X_1 \geq X_r$ の場合，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線(図 4-11 の L2)を考える．クリップ対象となる直線が L2 の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば $Y = Y_b$ と，L2 の上にあれば $X = X_r$ との交点計算を行う．1)，2)いずれの場合にも，$X_0 < X_l$ であるから $X = X_l$ との交点計算を行い，対象となる直線はクリップインとなる．

＜CenterColumn＞

1) $Y_0 > Y_t$ なら，図形を 90 度反時計回りに回転させて，LeftEdge で処理後，逆方向に回転させて元に戻す．
2) $Y_0 < Y_b$ なら，図形を 90 度時計回りに回転させて，LeftEdge で処理後，逆方向に回転させて元に戻す．
3) 上記いずれでもなければ Inside へ．

＜Inside＞

1) $X_1 < X_l$ なら，P2Left へ．
2) $X_1 > X_r$ なら，図形を 180 度回転させて P2Left で処理後，再び 180 度回転させて元に戻す．
3) 上記いずれでもない場合，$X_l < X_1 < X_r$ となる．従って，$Y_1 > Y_t$ なら $Y = Y_t$ と，$Y_1 < Y_b$ なら $Y = Y_b$ と交点計算を行う．どちらにも当てはまらない場合は $Y_b < Y_1 < Y_t$ となり，交点計算の必要はなく，対象となる直線はクリップインとなる．

＜P2Left＞

1) $Y_1 > Y_t$ なら，P2LeftTop へ．
2) $Y_1 < Y_b$ なら，図形を 90 度時計回りに回転させて P2LeftTop で処理後，逆方向に回転させて元に戻す．
3) 本ルーチンでは<Inside>の 1)より，$X_1 < X_l$ が成立しているから，上記いずれでもない場合は $X = X_l$ との交点計算を行い，対象となる直線はクリップインとなる．

＜P2LeftTop＞

1) 始点(S)とクリップ枠の左上コーナー点(TL)とを結ぶ直線(図 4-12 の L1)を考える．クリップ対象となる直線が L1 の下，つまり，$(Y_t - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば $X = X_l$ と，L1 の上であれば $Y = Y_t$ との交点計算を行い，対象となる直線はクリップインとなる．

4.2.2 ポリラインの状態遷移を利用した高速クリップ判定方式

前節で述べた NLN 法は，クリップ枠を構成する四つのコーナー点とクリップ対象となる直線を比較することにより，最小限の四則演算でクリップ処理を実行することができる．しかしながら，このクリップ処理には非常に多くの条件判断を伴う．特に，高速性が要求されるクリップインの判断はあらゆる条件判断の最後となり，四則演算の数を減少させたとはいえ，これらの条件判断に要する時間を無視することはできない．近年，コンピュータの処理速度は大幅に向上しているが，その性能向上の要因としてプロセッサのパイプライン化を挙げることができる[14-16]．つまり，プロセッサはクロック数を向上させるだけでなく，複数の命令をパイプライン化することにより，性能向上を図っているのである．しかしながら，条件判断に伴う分岐命令は，このプロセッサのパイプラインを乱し，処理速度を低下させてしまう．

一方，コンピュータグラフィックスで扱う直線は，通常ポリラインと呼ばれ，複数の連続した点列を結ぶ折れ線である．つまり，ポリラインを構成する複数の端点を連続して並べることにより，点の座標変換量を削減して描画性能の向上を図るものである．そこで本論文では，コンピュータグラフィックスで通常扱われるポリラインを対象とし，クリップ処理に要する四則演算の回数だけでなく，比較及び分岐命令の数をも最小とするクリップ方法について論ずる．

従来のクリップ方式では図 4-9 に示すように，クリップ枠を構成する 4 本の直線で分割される九つの領域を考え，図形の対称性を利用することにより三つの場合に分類してアルゴリズムを構築してきた．本クリップ方式ではこの領域をさらに拡張して，図 4-13 に示す 21 種類の領域を考える．

基本的な考え方は NLN 法と同じく，直線を構成する両端点の位置情報を調べながら，最小限の演算量でクリップ処理を行うことである．しかしながら，本クリップアルゴリズムは連続した点列を結ぶ折れ線を対象としているため，1 本の直線のクリップ処理を行った後，連続して次の直線のクリップ処理を行う必要がある．その際，最初の直線の終点は次の直線の始点となるため，終点の位置情報を用いることにより高速なクリップ処理が可能となる．但し，NLN 法によるクリップ処理を終えた段階では，従来の 9 分割された領域のいずれに属するかは不明なことが多いため，二つ，あるいは三つという複数の領域に跨る範囲を考え，これらの領域にも符号を割当てる．さらに，終点情報を利用することにより，NLN 法では様々な条件判断の最後に位置していたクリップイン処理を高速に実行することも可能となる．つまり，最初の直線のクリップ処理により，その終点の位置情報が上記 21 種類の領域符号で記述され，この領域符号を用いて次のクリップ処理が実行される．直線のクリップ処理を実行する度にその終点の位置情報は変化（**状態遷移**）し，状態の変化に応じた最適なクリップ処理アルゴリズムを適用することができる．最初の直線がクリップアウトであれば，次の直線も同じような領域に属するためクリップアウトとなる可能性が高く，最初の直線がクリップインであれば，次の直線もクリップインとなる確率が高い．このように，ポリラインを構成している点列の状態遷移に応じた最適なクリップ処理アルゴリズムを適用することにより，最少量の演算及び分岐命令で，高速なクリップを実行することが可能となる．本論文で提案する手法は，ポリラインを構成する点列の状態遷

移に応じて最適なクリップ処理アルゴリズムを適用する方法であるため，本手法を State Transition(ST)法と呼ぶことにする．

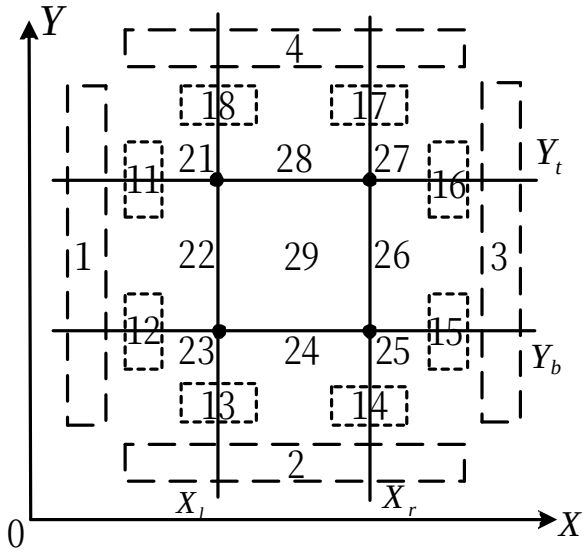


図 4-13 拡張クリップ領域

本クリップ方式も NLN 法と同様，図形の対称性を考慮してアルゴリズムを記述するが，実際のアルゴリズムでは NLN 法のように図形の折り返しや回転を行うのではなく，あらゆる場合のプログラムコードを記述しておいた方が，より高速なクリップ処理を実行することができる．以下，アルゴリズムを記述する．なお，クリップ枠を構成する４本の直線，クリップ対象となる直線の始点及び終点の座標は NLN 法と同一のものを使用する．

＜状態遷移を利用したクリップアルゴリズム＞

1) $X_0 < X_l$ なら State1 へ．
2) $X_0 > X_r$ なら State3（図形の対称性を考慮すれば State1 に相当）へ．
3) $Y_0 < Y_b$ なら State24（図形の対称性を考慮すれば State22 に相当）へ．
4) $Y_0 > Y_t$ なら State28（図形の対称性を考慮すれば State22 に相当）へ．
5) 上記いずれでもなければ，State29 へ．

＜State1＞

1) $X_1 < X_l$ ならクリップアウト．終点の領域符号は 1 で，次のクリップは State1 から開始．
2) $Y_0 < Y_b$ なら State23 の 2)（図形の対称性を考慮すれば State21 の 2）に相当）へ．
3) $Y_0 > Y_t$ なら State21 の 2)へ．
4) 上記いずれでもなければ，State22 の 2)へ．

＜State11＞

1) $X_1 < X_l$ならクリップアウト．終点の領域符号は 1 で，次のクリップは State1 から開始．
2) $Y_0 > Y_t$ なら State21 の 2)へ．
3) 上記いずれでもなければ，State22 の 2)へ．

＜State21＞

1) $X_1 < X_l$ならクリップアウト．終点の領域符号は 1 で，次のクリップは State1 から開始．
2) $Y_1 > Y_t$ ならクリップアウト．終点の領域符号は 17 で，次のクリップは State17（図形の対称性を考慮すれば State11 に相当）から開始．
3) 始点(S)とクリップ枠の左上コーナー点(TL)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_t - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば 4)へ．そうでなければ 8)へ．
4) $Y_1 < Y_b$ なら 5)へ．そうでなければ 7)へ．
5) 始点(S)とクリップ枠の左下コーナー点(BL)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であればクリップアウト．終点の領域符号は 14 で，次のクリップは State14 から開始．そうでなければ $X = X_l$ でクリップして 6)へ．
6) $X_1 < X_r$ なら $Y = Y_b$ でクリップ．終点の領域符号は 24 で，次のクリップは State24 から開始．そうでなければ，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば $Y = Y_b$ でクリップ．そうでなければ $X = X_r$ でクリップ．いずれの場合も，終点の領域符号は 25 で，次のクリップは State25 から開始．
7) $X = X_l$ でクリップ後，$X_2 > X_r$ なら，$X = X_r$ でクリップ．終点の領域符号は 26 で，次のクリップは State26 から開始．そうでなければ終点はクリップイン．終点の領域符号は 29 で，次のクリップは State29 から開始．
8) $X_1 > X_r$ なら 9)へ．そうでなければ 11)へ．
9) 始点(S)とクリップ枠の右上コーナー点(TR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の上，つまり，$(Y_t - Y_0)(X_1 - X_0) < (X_r - X_0)(Y_1 - Y_0)$ であればクリップアウト．終点の領域符号は 15 で．次のクリップは State15 から開始．そうでなければ $Y = Y_t$ でクリップして 10)へ．
10) $Y_1 > Y_b$ なら $X = X_r$ でクリップ．終点の領域符号は 26 で，次のクリップは State26 から開始．そうでなければ，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば $Y = Y_b$ でクリップ．そうでなければ $X = X_r$ でクリップ．いずれの場合も，終点の領域符号は 25 で，次のクリップは State25 から開始．
11) $Y = Y_t$ でクリップ後，$Y_1 < Y_b$ なら $Y = Y_b$ でクリップ．終点の領域符号は 24 で，次のクリップは State24 から開始．そうでなければ終点はクリップイン．領域符号は 29 で，次のクリップは State29 から開始．

＜State22＞

1) $X_1 < X_l$ならクリップアウト．終点の領域符号は1で，次のクリップはState1から開始．
2) $Y_1 < Y_b$なら3)へ．そうでなければ5)へ．
3) 始点(S)とクリップ枠の左下コーナー点(BL)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であればクリップアウト．終点の領域符号は14で，次のクリップはState14から開始．そうでなければ$X = X_l$でクリップして4)へ．
4) $X_1 < X_r$なら2)より$Y_1 < Y_b$であるから，$Y = Y_b$でクリップ．終点の領域符号は24で，次のクリップはState24から開始．そうでなければ，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_b - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば$Y = Y_b$でクリップ．そうでなければ，$X = X_r$でクリップ．いずれの場合も終点の領域符号は25で次のクリップはState25から開始．
5) $Y_1 > Y_t$なら6)へ．そうでなければ8)へ．
6) 始点(S)とクリップ枠の左上コーナー点(TL)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の上，つまり，$(Y_t - Y_0)(X_1 - X_0) < (X_l - X_0)(Y_1 - Y_0)$ であればクリップアウト．終点の領域符号は17で．次のクリップはState17から開始．そうでなければ，$X = X_l$でクリップして7)へ．
7) $X_1 < X_r$なら5)より$Y_1 > Y_t$であるから，$Y = Y_t$でクリップ．終点の領域符号は28で，次のクリップはState28から開始．そうでなければ，始点(S)とクリップ枠の右上コーナー点(TR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の上，つまり，$(Y_t - Y_0)(X_1 - X_0) < (X_r - X_0)(Y_1 - Y_0)$ であれば$Y = Y_t$でクリップ．そうでなければ，$X = X_r$でクリップ．いずれの場合も終点の領域符号は27で次のクリップはState27から開始．
8) $X = X_l$でクリップ後，$X_1 > X_r$なら$X = X_r$でクリップ．終点の領域符号は 26 で，次のクリップはState26から開始．そうでなければ終点はクリップイン．終点の領域符号は29で，次のクリップはState29から開始．

＜State29＞

1) 図4-7 に示すCohen-Sutherlandの４ビット符号ABCDを作成する．
2) ABCD=0000なら直線はクリップイン．終点の領域符号は29で次のクリップはState29から開始．
3) ABCD=0001なら$X = X_l$でクリップ．終点の領域符号は22で次のクリップはState22から開始．
4) ABCD=0010なら$X = X_r$でクリップ．終点の領域符号は26で次のクリップはState26から開始．
5) ABCD=0100なら$Y = Y_b$でクリップ．終点の領域符号は24で次のクリップはState24から開始．
6) ABCD=1000なら$Y = Y_t$でクリップ．終点の領域符号は28で次のクリップはState28から開始．
7) ABCD=0101なら，始点(S)とクリップ枠の左下コーナー点(BL)とを結ぶ直線LLを考える．クリップ対象となる直線が LL の上，つまり，$(Y_b - Y_0)(X_1 - X_0) < (X_l - X_0)(Y_1 - Y_0)$ であれば$X = X_l$でクリップ．そうでなければ$Y = Y_b$でクリップ．いずれの場合も終点の領域符号は23で，次のクリップはState23から開始．
8) ABCD=1001なら，始点(S)とクリップ枠の左上コーナー点(TL)とを結ぶ直線LLを考える．クリップ対象となる直線が LL の下，つまり，$(Y_t - Y_0)(X_1 - X_0) > (X_l - X_0)(Y_1 - Y_0)$ であれば$X = X_l$でクリップ．そうでなければ$Y = Y_t$でクリップ．いずれの場合も終点の領域符号は21で，次のクリップはState21から開始．

9) ABCD=0110 なら，始点(S)とクリップ枠の右下コーナー点(BR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の上，つまり，$(Y_b - Y_0)(X_1 - X_0) < (X_r - X_0)(Y_1 - Y_0)$ であれば $X = X_r$ でクリップ．そうでなければ $Y = Y_b$ でクリップ．いずれの場合も終点の領域符号は 25 で，次のクリップは State25 から開始．

10) ABCD=1010 なら，始点(S)とクリップ枠の右上コーナー点(TR)とを結ぶ直線 LL を考える．クリップ対象となる直線が LL の下，つまり，$(Y_t - Y_0)(X_1 - X_0) > (X_r - X_0)(Y_1 - Y_0)$ であれば $X = X_r$ でクリップ．そうでなければ $Y = Y_t$ でクリップ．いずれの場合も終点の領域符号は 27 で，次のクリップは State27 から開始．

以上で，直線を構成する点列の状態遷移を利用したポリラインクリップ判定方式について述べた．ここで，従来のクリップ方式である CS 法及び NLN 法と本手法(ST 法)に関して，クリップ処理に要する演算量（四則演算及び比較/分岐命令）の比較を行う．但し，従来の方法はポリラインを対象としていないため，参考として CS 法でポリラインを対象とした場合に一つ前のクリップ符号（４ビットコード）を保持し，次のクリップ処理においては終点のクリップコードのみを計算するように改良した方法を用いることとした．この改良版 CS 法を **Updated CS(UCS)法**と呼ぶ．図 4-14，4-15 及び 4-16 は，始点の状態を固定した場合に，終点の状態に応じてクリップ処理に必要な演算量を算出し，ST 法と NLN 法及び UCS 法とを比較したものである．例えば，図 4-14 はクリップ対象とする直線の始点が State21 の状態にある場合を示す図であり，終点が同じく State21 にある場合は左上の欄を参照することにより，この場合に ST 法で必要な演算は比較及び分岐がそれぞれ１回ということが分かる．また，図 4-15 は直線の始点が State22 の状態にある場合を示す図であり，終点が State25 にある場合には ST 法で必要な演算量は右下の欄を参照することにより，比較５，加算２，減算４，乗算２，除算２及び分岐５となる．図 4-16 には直線の始点の状態を示していないが，図 4-14 及び 4-15 と同じく，直線の始点が State29 に属する場合に，終点の状態に応じて必要な演算量を算出している．図形の対称性を考慮すれば，始点の状態が State21，22 及び 29 で全ての場合を包含しているので，これらの図で示されている演算量を比較すれば，直線の始点及び終点の状態に応じて各手法で必要な演算量を比較/分析することができる．

図に示されている演算量を比較すると ST 法は，従来最小の演算量でクリップ処理を行うと評価されていた NLN 法に比べて全ての場合において，より少ない演算量でクリップ処理を実現していることが分かる．また，UCS 法と比較しても，ほとんどの場合においてより少ない演算量でクリップ処理を実行している．UCS 法と比較して ST 法の方が演算量が多くなっている項目は，図 4-14 及び図 4-16 において○印を付けた部分である．この部分は，実際に直線とクリップ枠との交点計算を行う前に，直線がどのクリップ枠との交点計算を行う必要があるのかをチェックするために必要な減算及び乗算であり，UCS 法がこのチェック処理を不要としているために，ST 法及び NLN 法の方が演算量が多くなっている．しかしながら，多くなっている演算量はわずか 0.5，あるいは 1.0 であるのに対し，UCS 法ではこのチェック処理を不要としたことにより，その他の演算量（比較，加算，除算及び分岐）の演算量がかなり多くなっており，特にパイプライン制御を乱す原因となる比較及び分岐の回数が ST 法に比べて非常に多くなっている．これらの事柄を総合すると，ST 法は全ての場合において，NLN 法及び UCS 法よりもより少ない演算量でクリップ処理を実行できることになる．

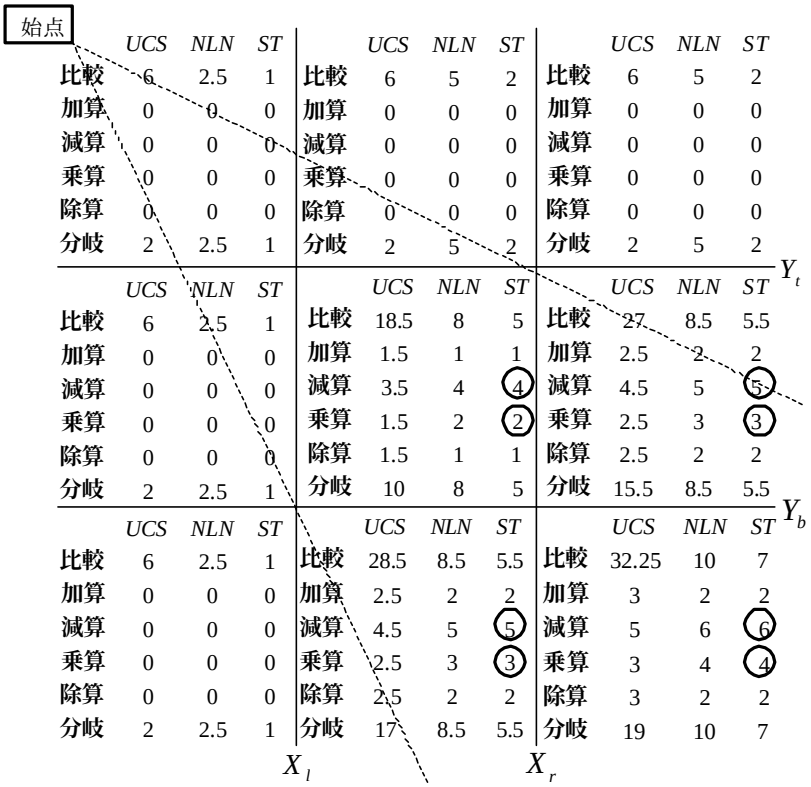


図 4-14 クリップ処理に要する演算量の比較（始点が State21 にある場合）

図 4-15 クリップ処理に要する演算量の比較（始点が State22 にある場合）

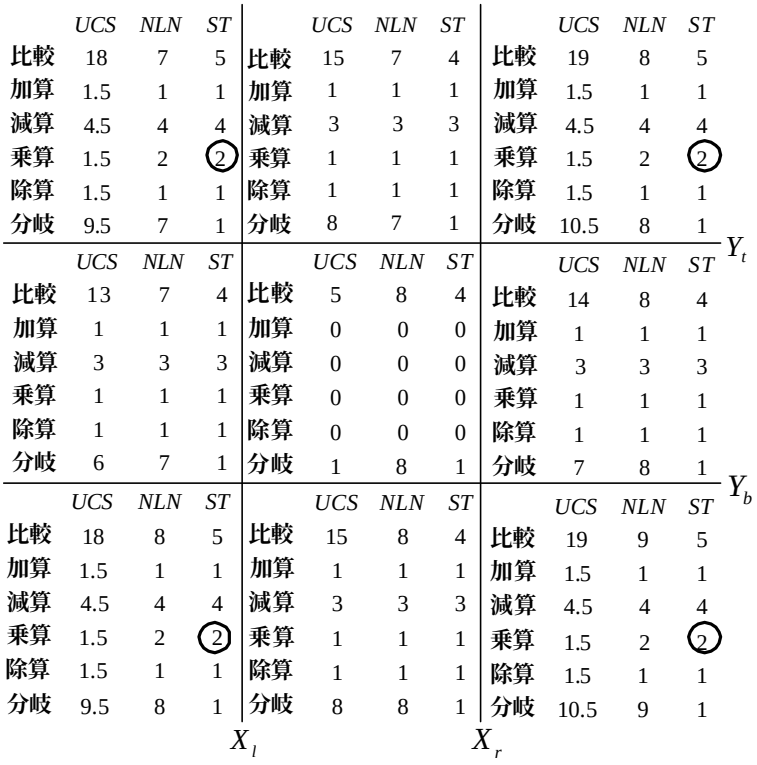


図 4-16 クリップ処理に要する演算量の比較（始点が State29 にある場合）

第五章　大規模グラフィックスデータの高速表示

前章の後半では，クリップ処理の高速化について述べた．コンピュータグラフィックスで構築される描画用データには大規模なものが多く，全ての詳細なデータをリアルタイムに表示するためには多大な計算コストが要求される．一方，応用面からすると全てのデータを詳細に表示することは稀であり，全体的な概要を把握するか，あるいは一部の領域を詳細に表示するかのどちらかである．従って，表示領域内に属するデータを高速に抽出するクリップ方式は重要である．また，全体的な概要を表示する際は，個々の部分に対する詳細な表示を行う必要はなく，詳細データを基に削減されたデータを用いれば全体的な概要を高速に表示することができる．さらに，構築されたグラフィックスデータが小規模で個々のデータのみを表示する際にも，表示対象物体が動的に動く場合と静止している場合とでは要求される表示画像の品質が異なる．つまり，物体が動いている場合にはリアルタイムな表示が重要であり，特に人間の操作に応じてインタラクティブに物体が動く場合には，人間に表示の遅延を感じさせない程度の高速処理が必要となる．ところが，動いている物体が一旦静止すると，表示物体の詳細な部分までも正確に表現する必要がある．

このような，概略データの高速表示と詳細データの高品質表示を同時に満足させるためには，対象とするグラフィックスデータを階層化し，全体表示あるいは動画表示の際には概略データを用いた高速表示を，また，一部領域の拡大表示あるいは静止画表示の際には詳細データを用いた高品質表示を行う方法が一般的である．そこで本章では，グラフィックスデータの階層化と階層化されたデータの高速表示方式について論ずることにする．なお本論文では，**削除**とは元データから一部のデータを取り除くこと，**削減データ**とは元データからデータを削除することにより減少したデータとする．

5.1　従来のグラフィックスデータ削減方法

グラフィックスデータの階層化は LOD(Level of Details)と呼ばれ，1996 年の SIGGRAPH でかなり多くの論文が発表された[17-20]．本研究もほぼ同時期に行ったものであるが，データ階層化の基本的な考えは Hoppe[18]の考え方と同じであるため，データ削減の一手法として Hoppe の方法を採用する．

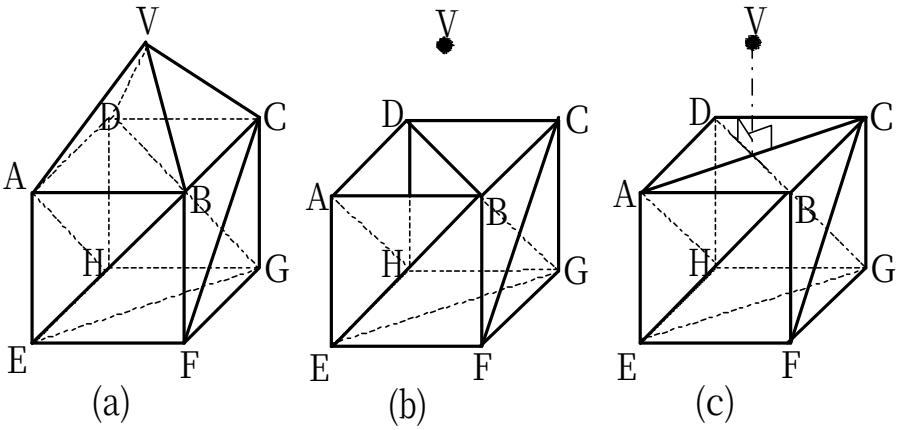


図 5-1 グラフィックスデータの削減（頂点マージ法）

コンピュータグラフィックスのデータは通常ポリゴンで構成されているが，最も基本となるポリゴンは三角形であり，三角形を定義する平面の方程式は三角形を構成する三つの頂点から唯一に決定されることから，ポリゴンを三角形に分割してデータとして保存したり，あるいは描画時に三角形に分割して表示処理したりすることが多い．そこで，本章で扱うグラフィックスデータは全て三角形から構成されているものとする．

図 5-1(a)のような図形を構成するグラフィックスデータに対して，Hoppe の考えを適用してデータの削減を行うと次の様になる．今，頂点Ｖを削除すると仮定してみる．この場合，単に頂点Ｖを取り除いただけでは 図 5-1(b) に示すように，頂点Ｖを用いて生成されていた ⊿ＶＡＢ， ⊿ＶＢＣ，⊿ＶＣＤ及び⊿ＶＤＡが消滅し，四角形ＡＢＣＤという穴が生成される．コンピュータグラフィックスのデータは全てポリゴン（基本的には三角形）で構築されており，ポリゴンのない部分は描画されない．つまり，表示物体に穴が開いてしまい，幾何学的な位相が異なることになる．このような穴を開けることなく頂点Ｖを削除するためには，単に頂点Ｖを取り除くのではなく，頂点Ｖを用いて生成されている三角形の他の頂点に，頂点Ｖを移動（**マージ**）することを考える．つまり，頂点Ｖを用いて生成されている三角形としては，⊿ＶＡＢ，⊿ＶＢＣ，⊿ＶＣＤ及び⊿ＶＤＡの四つがあり，これらの三角形を構成する他の頂点とは頂点Ａ，Ｂ，Ｃ及びＤの四つであるから，頂点Ｖをこれらの一つの頂点にマージすることを考える．例えば，頂点Ｖを頂点Ａにマージすると，図 5-1(c)のように新たな面として⊿ＡＢＣ及び⊿ＡＣＤが生成され，図 5-1(b)の場合に生成された穴は塞がれるため，幾何学的な位相の変化は生じない．この方法は，削除対象となる頂点を，その頂点を用いて生成されている三角形を構成する他の頂点にマージすることにより頂点を削除する方法なので，ここでは**頂点マージ法**と呼ぶことにする．ここで頂点Ｖを頂点Ａにマージすることにより，⊿ＶＡＢ,⊿ＶＢＣ，⊿ＶＣＤ及び⊿ＶＤＡはそれぞれ，直線ＡＢ，⊿ＡＢＣ，⊿ＡＣＤ及び直線ＡＤとなり，直線は面を構成しないから削除すると，新たに生成される⊿ＡＢＣ及び⊿ＡＣＤだけが残ることになる．

次に，頂点マージ法による誤差を計算する．頂点Ｖを頂点Ａにマージすることにより，⊿ＶＡＢ，⊿ＶＢＣ，⊿ＶＣＤ及び⊿ＶＤＡが消滅し，新たに⊿ＡＢＣ及び⊿ＡＣＤが生成されているから，頂点Ｖから新たに生成された ⊿ＡＢＣ 及び ⊿ＡＣＤ までの距離を計算し，その最大値を頂点Ｖを頂点Ａにマージしたことにより発生する誤差と定義する．図 5-1(c)では頂点Ｖから ⊿ＡＢＣ 及び⊿ＡＣＤまでの距離は等しく，どちらの距離を採用してもよいが，通常は全て異なる距離誤差を持つため，これらの最大距離を誤差とする．

上記では一例として，頂点Ｖを頂点Ａにマージした場合について説明したが，頂点Ｖを他の頂点であるＢ，Ｃ及びＤのどの頂点にマージしても構わない．そこで，上記で定義した誤差に対して，最小誤差を与える頂点へのマージを採択する．つまり，最大誤差を最小に抑えるようにデータ削減を行う方法であり，データ削減に対する閾値としての誤差に関しては，第三章で述べたミニマックス法[9]の考えを取り入れている．さらに，上記では頂点Ｖを削除するという仮定の基にデータの削減方法を検討してきたが，データの削減が目的であるため，削除対象となる頂点はＶに拘る必要はない．つまり，ここでもミニマックス法の考えを適用し，グラフィックスデータを構成している各頂点を削除すると仮定した場合に計算される誤差を調べ，最も小さな誤差を与える頂点を削除対象とする．そして，削除対象となった頂点はその頂点から新しい面までの最大距離が最小となるようにマージすべき頂点を調べ，その頂点へマージすることによりデータを削減する．アルゴリズムを用いて自動的なデータ削減を行う際，プログラムの終了条件としては次の２種類が考えられる．一つは，距離誤差を閾値として与え，上記で定義した誤差がこの閾値を超えない限りデータ削減を続け，誤差が閾値を超えた時点で終了する方法であり，もう一つは，削減目標となるデータ数（頂点数や面数）を与え，目

標以下のデータ量となるまで削減を繰り返し行う方法である．Hoppeのデータ削減方法に誤差解析としてミニマックス処理を適用したアルゴリズムを以下に示す．

＜頂点マージ法によるデータ削減アルゴリズム＞

1) データを構成する全ての頂点に対して，その頂点を他の頂点にマージすることにより生成される面までの距離を計算し，その最大距離を誤差とする．この誤差の中で最も小さな値を与える場合を採択し，削除対象となる頂点をマージする先の頂点を決定する．
2) 全ての頂点に対して計算された誤差の中で最小値を与える場合を採択し，削除対象となる頂点及びそのマージ先の頂点を決定する．
3) 今までに削除された全ての頂点に対し，上記削除により新たに生成される面までの距離誤差を調べる．もし，距離誤差が閾値を超えている場合にはデータ削減を終了させる．そうでなければ4)へ行く．
4) 上記により決定された削除対象となる頂点をそのマージ先の頂点に移動することにより，頂点を一つ削除すると同時に，頂点の削除により直線となる三角形も削除する（但し，新たに生成される面は頂点の移動により自動的に生成される）．
5) データ削減の終了条件（頂点数や面数あるいは閾値誤差）が満足されていれば終了．そうでなければ1)へ戻る．

5.2 複数頂点マージによるデータ削減方法

前節では，削除対象となる頂点をその頂点を用いて生成されている三角形の他の頂点にマージすることにより，頂点を削除するというHoppeの方法について検討してきた．しかしながら，頂点の削除方法としては別の方法も考えられる．例えば，削除対象となる頂点をただ単純に他の頂点にマージするのではなく，削除対象となる頂点及び他の頂点をも含めてマージすべき新たな頂点を生成し，この新たに生成される頂点に削除対象となる頂点及び他の頂点の全てをマージする方法である．この方法を用いると新たに頂点を生成する必要はあるが，削除対象となる頂点だけでなく他の頂点をも同時にマージすることができるため，一度に複数の頂点を削除することが可能となる．そこで，この方法を**複数頂点マージ法**と呼ぶことにする．

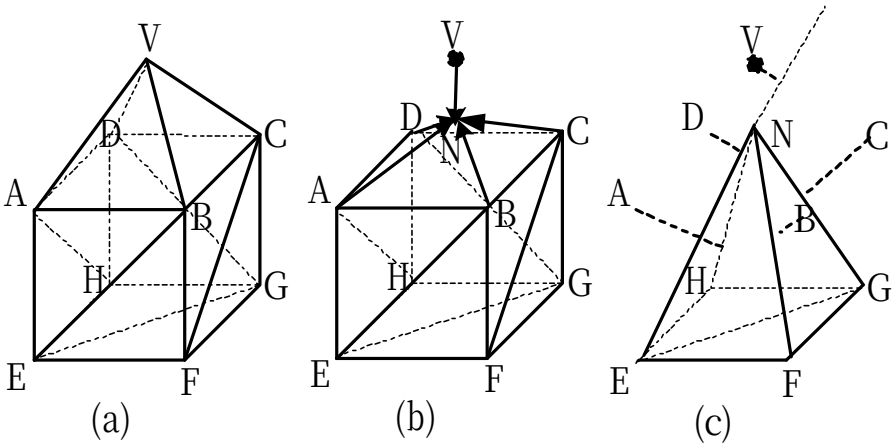


図5-2 グラフィックスデータの削減（複数頂点マージ法）

図 5-2(a)に示す図形に対して，頂点Vを削除すると仮定する．頂点Vを用いて生成されている三角形は⊿ＶＡＢ，⊿ＶＢＣ，⊿ＶＣD及び⊿ＶＤＡであるから，それらの三角形を構成する頂点V以外の頂点は頂点Ａ，Ｂ，Ｃ及びDの４つである．従って，削除対象となる頂点V及び他の４頂点（Ａ，Ｂ，Ｃ及びD）の平均を取ることにより新たな頂点Nを生成する．そして，頂点Nに全ての頂点をマージすると，図 5-2(c)に示すような四角錐が得られる．頂点マージ法では一回の削除計算に対して１頂点のみの削除を行ったが，複数頂点マージ法では５頂点を１頂点にマージするため，一回の計算で４頂点の削除を行うことができる．

誤差計算は頂点マージ法と同様に行う．つまり，削除された各頂点から新たに生成される面までの距離を計算し，その最大値を誤差距離とする．そして，最も小さい誤差距離を与える頂点を削除対象としてデータの削減を行う．ここで，図 5-2 に従って，新たに生成される面及び消滅する面について調べておく． 図 5-2(a)では，図形を構成する面は 14 個の三角形から成り立っており，⊿ＶＡＢ，⊿ＶＢＣ，⊿ＶＣＤ，⊿ＶＤＡ，⊿ＢＡＥ，⊿ＢＥＦ，⊿ＣＢＦ，⊿ＣＦＧ，⊿ＤＣＧ，⊿ＤＧＨ，⊿ＡＤＨ，⊿ＡＨＥ，⊿ＧＨＥ及び⊿ＧＥＦである．頂点Ｖ，Ａ，Ｂ，Ｃ及びＤを頂点Ｎにマージすることにより，上記三角形は次のように変形する．点Ｎ，点Ｎ，点Ｎ，点Ｎ，直線ＮＥ，⊿ＮＥＦ，直線ＮＦ，⊿ＮＦＧ，直線ＮＧ，⊿ＮＧＨ，直線ＮＨ，⊿ＮＨＥ，⊿ＧＨＥ及び⊿ＧＥＦである．点及び直線は面を構成しないため削除すると，⊿ＮＥＦ，⊿ＮＦＧ，⊿ＮＧＨ，⊿ＮＨＥ，⊿ＧＨＥ及び⊿ＧＥＦだけが残ることになる．但し，⊿ＮＥＦ，⊿ＮＦＧ，⊿ＮＧＨ及び⊿ＮＨＥの４面は新たに生成された面であり，⊿ＧＨＥ及び⊿ＧＥＦの２面は変化しない面である．複数頂点マージ法によるアルゴリズムを以下に記述する．

＜複数頂点マージ法によるデータ削減アルゴリズム＞

1) データを構成する全ての頂点に対して，対象となる頂点と，その頂点を用いて生成される三角形を構成する全ての頂点の平均を取ることにより新たな頂点を生成し，対象となる頂点を含む全ての頂点を新たに生成される頂点にマージする．マージされた全ての頂点から，複数頂点のマージより新たに生成される面までの距離を計算し，その最大距離を誤差とする．
2) 全ての頂点に対して計算された誤差の中で最小値を与える場合を採択し，削除対象となる頂点及び新たに生成するマージ先の頂点を決定する．
3) 今までに削除された全ての頂点に対し，上記削除により新たに生成される面までの距離誤差を調べる．もし，距離誤差が閾値を超えている場合にはデータ削減を終了させる．そうでなければ 4)へ行く．
4) 上記により決定された削除対象となる全ての頂点を新たに生成されたマージ先の頂点に移動することにより，複数の頂点を削除すると同時に，頂点の削除により直線あるいは点となる三角形も削除する．（但し，新たに生成される面は頂点の移動により自動的に生成される．）
5) データ削減の終了条件（頂点数や面数あるいは閾値誤差）が満足されていれば終了．そうでなければ 1)へ戻る．

5.3　グラフィックスデータの階層化と削減方法の評価

前節及び前々節で述べた二つのデータ削減方法（頂点マージ法及び複数頂点マージ法）を用いて一つのグラフィックスデータに対して複数のデータ削減を行い，その削減されたデータの量に応じて段階的に表示を行うと，グラフィックスデータの階層化表示を行うことができる．本章における研究では，対象とするグラフィックスデータとして，VRML(Virtual Reality Modeling Language)[21]を用いることにした．VRML とは，インターネット上の表記言語である HTML(HyperText Markup Language)の３次元グラフィックス版であり，インターネットを介して３次元コンピュータグラフィックスを扱う上での業界標準となるデータ形式である．VRML は元々，SGI 社の Open Inventor を基にしたものであるが，現在では，SGI，IBM，SUN など主なコンピュータメーカが参画する VAG(VRML Architecuture Group)により，その仕様が決定されている．VRML は単なるグラフィックスデータの形式を定めたものではなく，インターネット上での動きを記述することもできる言語であるが，本研究ではデータ形式だけを用いてデータの削減と階層化の実験を行った．

図 5-3 に示すドルフィンは 285 頂点，563 三角形から構成される VRML 形式のグラフィックスデータである．図 5-4 及び 5-5 に頂点マージ法及び複数頂点マージ法により階層化されたデータの表示結果を示す．

図 5-3　ドルフィン（285 頂点，563 三角形）

(a) 200頂点395三角形

(b) 150頂点296三角形

(c) 100頂点196三角形

(d) 50頂点96三角形

図 5-4 頂点マージ法によるグラフィックスデータの階層化

(a) 198頂点391三角形

(b) 147頂点289三角形

(c) 100頂点194三角形

(d) 50頂点92三角形

図 5-5 複数頂点マージ法によるグラフィックスデータの階層化

図 5-4 では，グラフィックスデータの削減に伴いドルフィンの形状も徐々に劣化しており，ほぼ予想通りの階層化を実現することができた．しかしながら，一度に複数の頂点を削除した複数頂点マージ法では図 5-5 に示すように，データを削減するにつれてドルフィンの形状が急激に劣化しているのが分かる．これは，一度に複数の頂点を削除するために新たな頂点を生成したことから，データ削減を重ねるにつれて新たに生成された頂点の影響により，ドルフィン形状が元形状から急激に離れていったものと推測される．そこで，頂点マージ法及び複数頂点マージ法をもう少し定量的に評価するために，各方法を用いた場合のデータ削減に要する時間及びデータ削減に伴い発生する誤差の解析を行った．

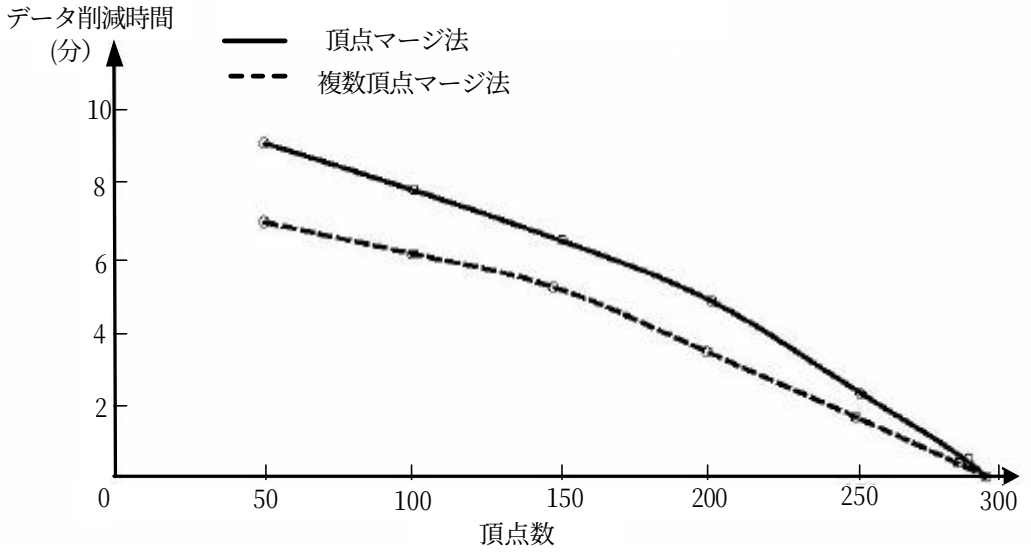


図 5-6 頂点マージ法及び複数頂点マージ法に要するデータ削減時間

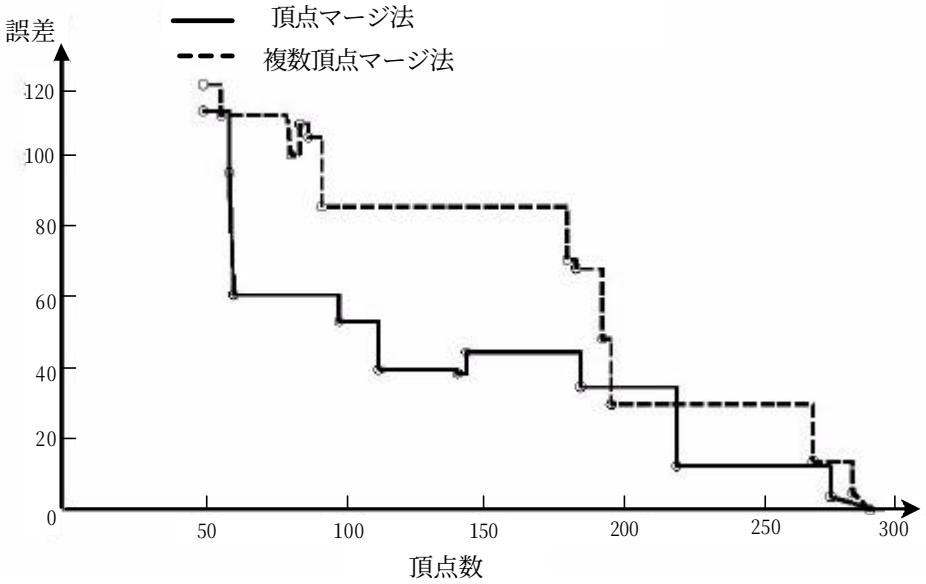


図 5-7 頂点マージ法及び複数頂点マージ法の誤差

データ削減プログラムはLispを用いて作成し，Sun Workstaion(SPARC Station 1)にて実行させた結果が図5-6である．複数頂点マージ法は新たな頂点を生成することにより一度に複数の頂点削除を目的としたものであるため，一度に一つの頂点しか削除できない頂点マージ法に比べてデータ削減時間が短いことが分かる．しかしながら図5-7に示すように，同じ頂点数まで削減したデータを比較した場合，元の頂点から新たに生成される面までの距離誤差が，頂点マージ法に比べて複数頂点マージ法の方が大きいことが分かる．つまり，図5-4及び図5-5を比較した際に，複数頂点マージ法のグラフィックスデータの方が頂点マージ法のグラフィックスデータに比べて形状の劣化が大きい定量的な裏付けが出来たことになる．

結局，新たな頂点を生成して一度に複数の頂点削除を行うよりも，元の頂点座標を変更することなく一度に一つの頂点削除を行った方が，最終的な誤差及び形状の劣化が少ないことが分かった．そこで次節では，頂点マージ法により削減及び階層化されたデータを基に，粗い形状データから詳細な形状データへと徐々に復元する階層化表示方法について検討する．

5.4　高速復元のための階層化データ作成方法

本節では，頂点マージ法により削減された複数のグラフィックスデータに対して頂点数に応じた階層化を行うと共に，階層化されたデータ間に関連性を持たせることにより高速に元のグラフィックスデータを復元する方法を検討する．

頂点マージ法では複数頂点マージ法と異なり，新たな頂点を生成することなく，必ず元のグラフィックスデータの頂点にマージさせることにより対象となる頂点を削除する．従って，削減されたグラフィックスデータを構成する頂点は全て元のデータの頂点と同一であり，頂点数の増減はあっても，頂点の座標値に変更はない．従って，頂点データに関して粗いデータから詳細なデータへ復元するためには，基本的には削除された頂点データを追加すればよいわけである．しかしながら面データに関しては，単純に削除された面を追加するだけでは元の詳細なグラフィックスデータを復元することはできない．何故なら，幾何的な位相の変化を発生させることなく，つまり，グラフィックスデータに穴を生じさせることなく頂点データの削除を行ったため，元の面データが削除されるだけでなく，新たな面データが生成されているからである．従って，面データに関しては元のグラフィックスデータを復元するためには，単純に削除された面データを追加するだけでなく，新たに生成された面データを削除する必要がある．

以下，具体的な例を用いて説明する．図5-8に示すのはVRML Ver.1.0データの一例であり，頂点データ(Coordinate3)及び面データ(IndexedFacetSet)の一部を割愛しているが，図5-4(d)の50頂点で構成されるドルフィンのVRMLデータである．Transform， ShapeHints及び Materialは属性データであり，ドルフィンデータをモデリング座標系からワールド座標系に変換するモデリング変換で必要となるデータ，面を構成する頂点の並びや法線ベクトル生成時に使用されるデータ及びドルフィンの色属性データである．これらのデータは最初に設定しておけばよいものであり，グラフィックスデータの階層化やデータの高速復元に関して特に留意すべきものではない．一方，Coordinate3及び IndexedFaceSetはグラフィックスデータを構成する頂点及び面データであり，データの階層化及び高速復元に深く関係するデータである．

Coordinate3は3次元座標を持つ頂点データであり，point[の後に，x y z の座標値が順に並んでいる．座標値と座標値の間はカンマ(,)で区切られ，] まで座標値データは続く．グラフィックスデータの形式によっては，最初に格納されている座標値の数を表示しているものもあるが，この形式で

は座標値データの数に関する記述はなく，座標値データを入力する際に数える必要がある．また，IndexedFaceSet は面を構成する頂点座標の番号を表すデータであり，三角形なら三つ，四角形なら四つの番号が並ぶ．頂点データの x y z がブランクにより区切られていたのとは異なり，面データを構成する頂点の番号は全てカンマ(，)で区切られているため，面の区切りを示すために，-1 が用いられる．つまり，カンマ(，)で区切られた番号データを-1 が現れるまで順に読み込むことで，一つの面を構成する頂点の数とその番号が分かる．頂点データと同様に，面を構成する頂点の数に関する記述はなく，また，表示物体を構成する面の数に関する記述もない．なお，頂点データの番号は 0 からではなく，1 から始まる．従って，図 5-8 のグラフィックスデータを構成する最初（1 番目）の面は 27, 6 及び 1 番目の頂点から構成される三角形であり，1 番目の頂点座標は(88.8501,0,496.294)であることが分かる．

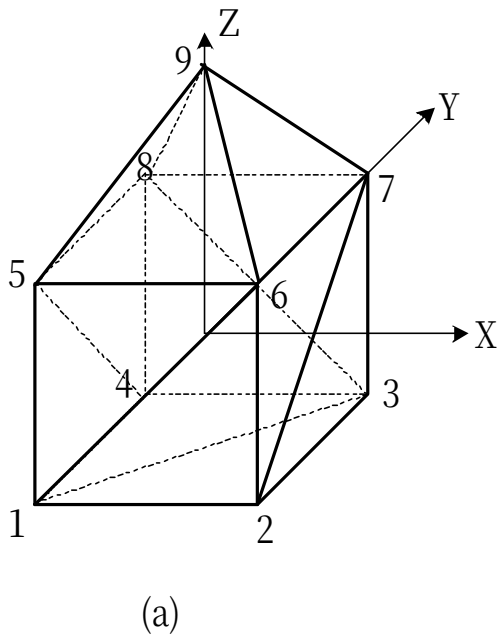
```
#VRML V1.0 ascii

Separator {
    Transform {
rotation            1 0 0  4.71239
center              -63.8645 0 193.138
                    }
    ShapeHints {
vertexOrdering      COUNTERCLOCKWISE
creaseAngle         1.5708
                    }
    Material {
ambientColor        0.092332 0.0928379 0.101263
diffuseColor        0.38087 0.382957 0.417708
specularColor       0.533333 0.533333 0.533333
emissiveColor       0 0 0
shininess           0.933333
transparency        0
                    }
    Coordinate3 {
point [ 88.8501 0 496.294,
        39.0601 -11.68 511.024,
        -201.22 0 402.574,
        -177.09 -13.97 423.654,
        ・・・・・・・・・・・・・・・・・・
        -25.9599 77.22 353.294,
        236.17 86.61 277.854,
        214.58 52.32 317.734 ]
                    }
    IndexedFaceSet {
coordIndex [ 27, 6, 1, -1,
             28, 27, 1, -1,
             5, 4, 3, -1,
             6, 5, 3, -1,
             ・・・・・・・・・・・・
             49, 25, 45, -1,
             48, 25, 49, -1,
             49, 25, 48, -1 ]
                    }
}
```

図 5-8 VRML のデータ例

図5-1(a)の図形を対象にし，図5-9(a)のように表示物体のモデリング座標系を取ると，図形のVRMLデータ（頂点データ及び面データ）は図 5-9(b)のように記述される．このデータに対し，頂点９を頂点５にマージして，対象となっている頂点９を削除すると，図 5-10 のようになる．但し，図5-10(b)では，削除される頂点データ９ 及び 頂点９を頂点５にマージすることにより，直線となる⊿５６５と⊿８５５を横線にて抹消している．また，図 5-9(b)における⊿６７９及び⊿７８９は図5-10(b)において，⊿６７５及び⊿７８５となり，新しい三角形が生成されている．

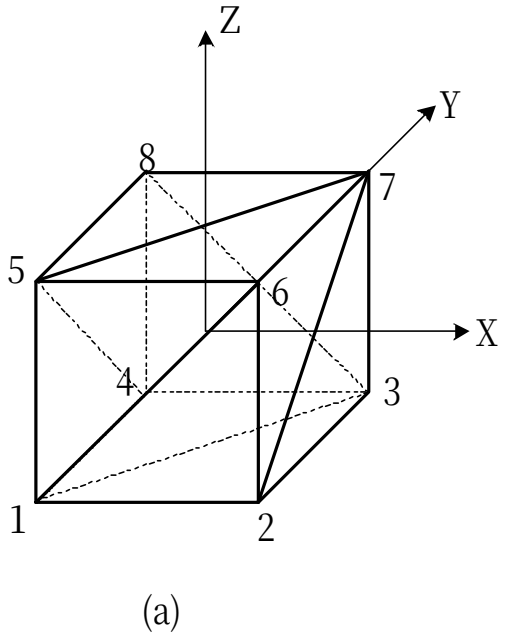
従って，図5-9(b) 及び 図5-10(b)を比較すると，削減されたデータ（図5-10(b)）から元のデータ（図5-9(b)）を復元するためには，頂点データに対して削除された頂点９を追加すると共に， 面データに対しては新たに生成された⊿６７５ 及び ⊿７８５を削除すると共に，削除されていた⊿５６９，⊿６７９，⊿７８９及び⊿８５９を追加する必要がある．




```
Coordinate3 {
point [ -0.5 -0.5 -0.5,
0.5 -0.5 -0.5,
0.5 0.5 -0.5,
-0.5 0.5 -0.5,
-0.5 -0.5 0.5,
0.5 -0.5 0.5,
0.5 0.5 0.5,
-0.5 0.5 0.5,
0 0 1.5] }
lndexedFaceSet {
coordlndex [ 1, 2, 3, -1,
1, 3, 4, -1,
1, 6, 5, -1,
1, 2, 6, -1,
2, 7, 6, -1,
2, 3, 7, -1,
3, 8, 7, -1,
3, 4, 8, -1,
4, 5, 8, -1,
4, 1, 5, -1,
5, 6, 9, -1,
6, 7, 9, -1,
7, 8, 9, -1,
8, 5, 9, -1] }
```

(b)

図 5-9 表示物体の形状と VRML データ（削減前のデータ）




```
Coordinate3 {
point [ -0.5 -0.5 -0.5,
0.5 -0.5 -0.5,
0.5 0.5 -0.5,
-0.5 0.5 -0.5,
-0.5 -0.5 0.5,
0.5 -0.5 0.5,
0.5 0.5 0.5,
-0.5 0.5 0.5,
0 0 1.5 ] }
IndexedFaceSet {
coordIndex [ 1, 2, 3, -1,
1, 3, 4, -1,
1, 6, 5, -1,
1, 2, 6, -1,
2, 7, 6, -1,
2, 3, 7, -1,
3, 8, 7, -1,
3, 4, 8, -1,
4, 5, 8, -1,
4, 1, 5, -1,
5, 6, 5, -1,
6, 7, 5, -1,
7, 8, 5, -1,
8, 5, 5, -1] }
```

(b)

図 5-10 表示物体の形状と VRML データ（削減後のデータ）

図 5-10 の場合は，削除される頂点データの番号が最後の番号であったために頂点の順番を表す番号に変化は生じず，結果的に面を構成する頂点の番号を入れ替える必要はなかった．しかしながら，一般に削除される頂点の番号が最後とは限らず，最後ではない番号の頂点を削除すると削除した頂点以降の番号が一つ繰り上がることになる．従って，データ削減の場合には，削除した番号以降の頂点番号を持つ面データを書き換える必要がある．削減されたデータを元のデータに復元する際にも同様の処理が必要とされるが，削除された頂点データを必ずしも元の順番に戻す必要はない．面データは面を構成する頂点の番号のみで構成されているため，面を構成する頂点の番号が正しく記述されていれば，削除された頂点データを必ずしも元の位置に戻さなくてもよいわけである．しかも，削減されたデータから元のデータを復元する際に，削除された頂点データを頂点データ列の最後に追加すれば，追加される以前から存在している頂点の番号に影響を及ぼすことはない．従って，面データの追加及び削除を行うだけでよく，追加される以前から存在している面データを書き換える必要はなくなる．そこで，削減されたデータから元データを復元する際には，削除された頂点及び面データはデータリストの最後に追加することにする．

上記では説明を簡単にするために，単純な図形を対象とし，しかも，一つの頂点を削除した場合を例に取ってデータリストを比較しながら説明を行ったが，グラフィックスデータを階層化する場合，通常は複数の頂点を削除して削減データを作成する．例えば，図 5-4 では頂点数 50，100，150 及び 200 から成る 4 階層のデータを作成している．そこで，複数の頂点を削除したデータから元の，あるいは一つ前の詳細なデータを復元する際に必要となる追加あるいは削除すべきデータ（頂点及び面データ）を作成するアルゴリズムを次に示す．但し，アルゴリズムを起動する前に，対象となるグラフ

ィックスデータには同一座標を持つ複数の頂点がないことを確認しておく．もし，同一座標を持つ複数の頂点データがあれば，それらは一つにまとめることができる．重複している頂点データを一つの頂点データにマージすることにより重複している頂点データを一つにまとめると共に，重複している頂点データを一つにまとめることにより直線や点となって表示に影響を与えない三角形も削除しておく．

＜**階層化データ作成アルゴリズム**＞

1) 削減データ及び元データ（あるいは一つ前の詳細データ）を比較し，頂点データに関する差分を求める．
2) 頂点データに関する差分は削減データから元データを復元する際に追加される頂点群であるから，これらを**頂点追加リスト**として作成する．
3) 頂点追加リストに記述されている頂点群を削減データに追加し，**新データ**とする(図5-11(a))．
4) 元データ及び新データを比較し，差分を求める．頂点の数は同じであり，差分は頂点データの番号の違いである．従って，この番号の違いを調べ，元データの頂点番号を新データの頂点番号へ変換する**頂点番号変換表**を作成する(図5-11(b))．
5) 上記で作成した頂点番号変換表を用いて，元データの面データを構成する頂点番号を書き換え，これを新データの面データとする(図5-11(c))．
6) 削減データの面データと新データの面データを比較し，差分を求める．差分は削減データから新データを復元する際に追加すべき面データ及び削除すべき面データであるから，これらを**面追加リスト**及び**面削除リスト**とする(図5-11(d))．
7) 得られた頂点追加リスト，面削除リスト及び面追加リストを，削減データの頂点データ及び面データに追加すると，階層化データが作成される(図5-11(e))．

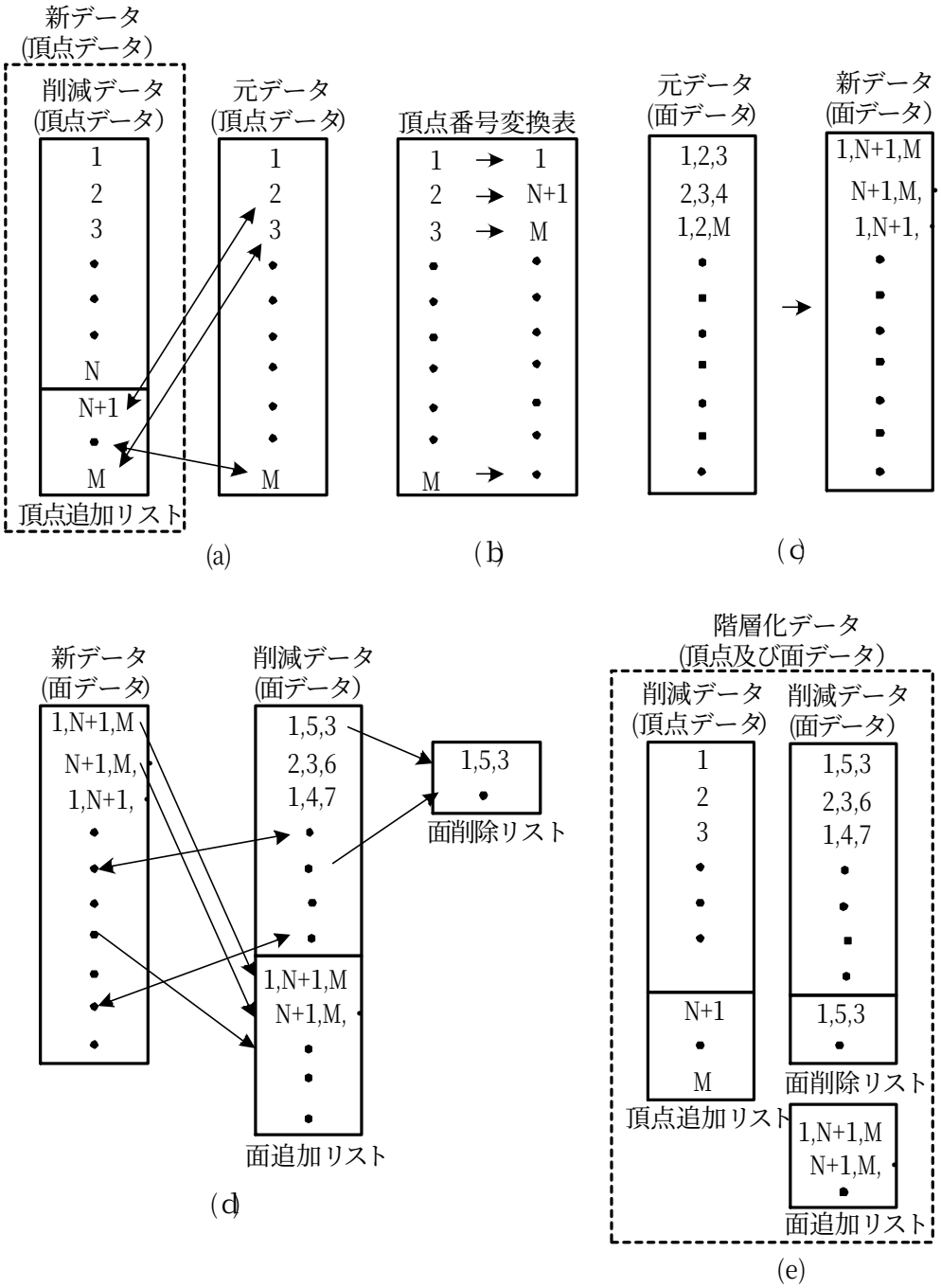


図 5-11 階層化データの作成

現状の VRML 仕様には，上記頂点追加リスト，面削除リスト及び面追加リストに対応する記述様式が存在しないため，VertexAddition，FacetDeletion 及び FacetAddition を追加して階層化データファイルを作成することにした．図 5-3 のドルフィンを 3 階層化したデータファイルを図 5-12 に示す．最も粗いデータが Coordinate3 及び IndexedFaceSet に記述されており，このデータに最初の VertexAddition を追加し，FacetDeletion を削除すると共に FacetAddition を追加することにより，次の詳細なデータが得られる．さらに，その後にある VertexAddition を追加，FacetDeletion を削

除して FacetAddition を追加すれば最も詳細なデータを得ることができる．なお，紙面の都合上，本来は縦に並ぶデータ列を便宜上，横に記述している．

```
#VRML V1.0 ascii

Separator {
  Transform {
    rotation          1 0 0  4.71239
    center            -63.8645 0 193.138
  }
  ShapeHints {
    vertexOrdering    COUNTERCLOCKWISE
    creaseAngle       1.5708
  }
  Material {
    ambientColor      0.092332 0.0928379 0.101263
    diffuseColor      0.38087 0.382957 0.417708
    specularColor     0.533333 0.533333 0.533333
    emissiveColor     0 0 0
    shininess         0.933333
    transparency      0
  }
  Coordinate3 {
    point [ 88.8501 0 496.294,
            39.0601 -11.68 511.024,
            -201.22 0 402.574,
            -177.09 -13.97 423.654,
            ..................
            -25.9599 77.22 353.294,
            236.17 86.61 277.854,
            214.58 52.32 317.734 ]
  }
  IndexedFaceSet {
    coordIndex [ 27, 6, 1, -1,
                 28, 27, 1, -1,
                 5, 4, 3, -1,
                 6, 5, 3, -1,
                 ..........
                 49, 25, 45, -1,
                 48, 25, 49, -1,
                 49, 25, 48, -1 ]
  }
```

```
VertexAddition {
  point [ -19.6099 -8.13 512.804,
          96.2101 0 561.824,
          78.1801 0 521.184,
          ..........
          233.63 9.65 310.114,
          222.45 24.13 336.024,
          236.42 21.59 294.364 ]
}
FacetDeletion {
  coordIndex [ 27, 6, 1, -1,
               28, 27, 1, -1,
               5, 4, 3, -1,
               ..........
               49, 25, 45, -1,
               48, 25, 49, -1,
               49, 25, 48, -1 ]
}
FacetAddition {
  coordIndex [ 0, 52, 53, -1,
               27, 50, 1, -1,
               1, 54, 27, -1,
               ..........
               146, 147, 49, -1,
               148, 146, 49, -1,
               8, 146, 148, -1 ]
}
```

```
VertexAddition {
  point [ 13.6601 0 548.364,
          -24.9399 0 513.564,
          21.2801 -6.1 547.094,
          ..........
          -59.9999 78.23 350.244,
          219.66 73.15 302.744,
          217.63 52.32 320.784 ]
}
FacetDeletion {
  coordIndex [ 10, 2, 26, -1,
               18, 11, 20, -1,
               27, 50, 1, -1,
               ..........
               146, 147, 49, -1,
               148, 146, 49, -1,
               8, 146, 148, -1 ]
}
FacetAddition {
  coordIndex [ 150, 151, 50, -1,
               50, 152, 150, -1,
               153, 150, 152, -1,
               ..........
               232, 233, 112, -1,
               256, 255, 125, -1,
               185, 74, 188, -1 ]
}
```

図 5-12　3階層化したドルフィンデータ

5.5　大規模グラフィックスデータの階層化表示

前節で作成した階層化データファイルを用いれば，頂点追加リストにある頂点群を頂点データに追加し，面削除リストにある面群を面データから削除すると共に，面追加リストにある面群を面データに追加すれば，粗いデータから詳細なデータへと段階的にグラフィックスデータを切り替えて画像の品質を徐々に向上させることができる．本研究により作成した階層化データを表示するネットワークブラウザを，JAVA を用いて作成した．ドルフィンデータの表示例を図 5-13 に示す．

作成したネットワークブラウザは GraNet(Graphics for Network)と呼ばれ，次の機能を備える．

1) **表示データの入力：**表示すべきデータファイルの入力には2種類の方法がある．一つは，Input file というウィンドウでファイル名を入力する方法であり，もう一つは，WRL files というウィンドウ上に表示されているファイルをダブルクリックする方法である．入力されたデータは Loaded Images というウィンドウに表示され，一度入力されたデータはそのファイル名をダブルクリックすることにより，表示データを切り替えることができる．なお，通常の VRML ファイルは拡張子として.wrl を使用しているため，本研究で作成した階層化データファイルの拡張子を.ref と定義する．
2) **色の変更：**Change Color の下に示されているＲＧＢのカラーバーを移動させると，R，G，B 各色要素を別々に変更することができる．また，白色バーを移動させると，色相を変更せずに，全体の輝度を変更することができる．Color Reset というボタンをクリックすると，元の色に戻る．
3) **拡大縮小及び回転：**Zoom の下にある＋ボタンをクリックすると表示物体が拡大され，－ボタンをクリックすると表示物体は縮小される．Reset ボタンをクリックすると元の状態に戻る．Rotation の下にあるＸ，ＹあるいはＺボタンをクリックすると，表示物体はそれぞれＸ，ＹあるいはＺ軸周りに回転する．Reverse ボタンをクリックしてからＸ，ＹあるいはＺボタンをクリックすると，逆方向に回転する．All Reset というボタンをクリックすると，全ての属性データ（色，大きさ，軸周りの回転角など）が初期状態に戻る．
4) **マウスオペレーション：**マウスによる操作も可能である．マウスを上下方向に移動させると，Ｘ軸回りの回転，左右方向に移動させるとＹ軸回りの回転，左上から右下あるいは右下から左上に移動させるとＺ軸周りの回転，左下方向に移動させると拡大，右上方向に移動させると縮小する．

本研究では，グラフィックスデータの削減方法として頂点マージ法を用いた．頂点マージ法は，元々存在するグラフィックスデータの頂点に削除対象となる頂点をマージすることによりデータ削減を行う方法であり，頂点を削除する際にグラフィックスデータに穴が開くことはなく，幾何学的な位相を保つことができる．また，階層化したデータを用いて，元のグラフィックスデータを徐々に復元（Progressive Refinement）する場合，頂点データに関しては削除されたデータを追加するだけよい．コンピュータグラフィックスを用いて物体を表示する際，対象物体を構成している頂点データの座標変換を行い，変換された座標値を基に表示プリミティブ（通常は三角形）を描画する．データを階層化することにより複数のデータを持つと，各段階の表示処理に多大な時間を要し，最終となる詳細データの表示が遅くなる可能性がある．しかしながら，作成したデータの頂点データは全て元の頂点データであり，途中段階で削除される余分な頂点データは存在しない．従って，座標変換に関する限り余分な変換処理を行う必要はなく，Progressive Refinement の各段階では頂点追加リストに記述されている頂点データのみの座標変換を行えばよいことになる．結果として，面データの追加削除処理を伴うものの，最終的な詳細データを最初から表示する場合とほぼ同等の処理時間で途中段階の画像表示を行うことができる．

ブラウザを使用してマウスで物体の移動（拡大縮小あるいは回転など）を行っている間は粗いデータを用いて物体が表示され，リアルタイムな拡大縮小あるいは回転を行うと共に，マウスを静止させた瞬間に次の段階のデータを用いた表示処理が始まる．こうすることにより，表示物体の画質は徐々に向上し，最終的には最も詳細なデータを用いた高品質画像を得ることができる．

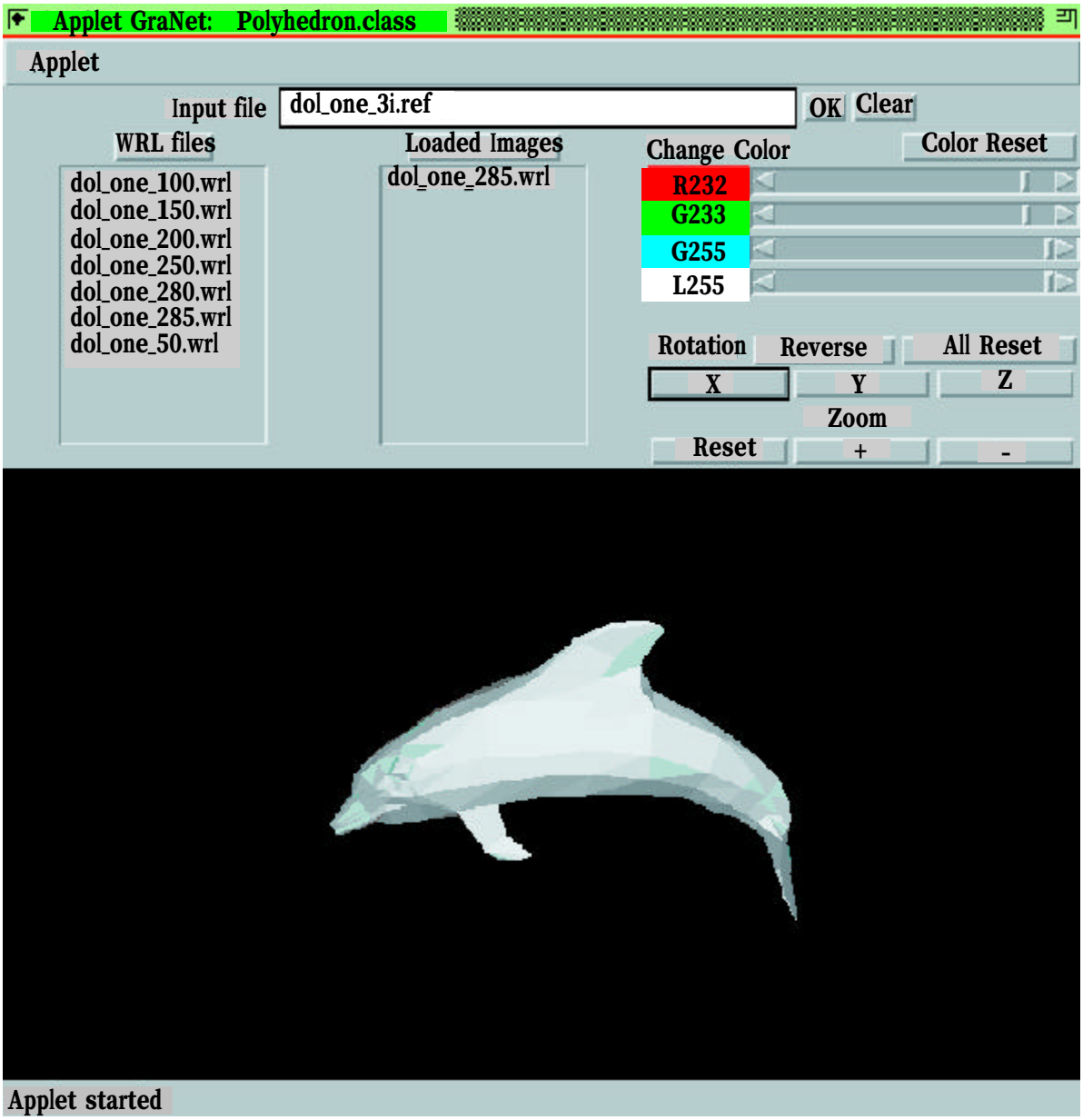


図 5-13 階層化データを表示するネットワークブラウザ

第六章　バーチャルリアリティ技術と医療応用

6.1　バーチャルリアリティ技術

近年におけるコンピュータグラフィックスの発展に伴い，コンピュータグラフィックスを利用した様々な応用技術が生まれてきた．その一つが**バーチャルリアリティ（Virtual Reality）**技術である．バーチャルリアリティ技術とは，現実には存在しない事物を仮想的に創造し，コンピュータグラフィックスを使用してインタラクティブに表示する技術であり，**仮想現実感**と訳される．しなしながら，バーチャルリアリティ技術の第一人者である東京大学の舘教授によれば，この訳は適切ではない．Virtual とは”Existing in effect or essence through not in actual fact or form”であり，日本語に訳すと，「表面あるいは名目上そうではないが事実上の，本質としては」となり，Virtual Reality は”Entity that has the essence of real entity through not the real entity itself”となり，日本語では「現物そのものではないが現物のエッセンスを有するもの」となる[22]．つまり，バーチャルリアリティ技術とは，現実には存在しないがそのものの本質を備えているもの，あるいは現実に存在するものと同様の効果を与えるものを生み出す技術であり，バーチャルリアリティという言葉をそのまま用いるか，あるいは**人工現実感**と訳した方がよい．

具体的な例を示すと，テレビや冷蔵庫などの家電製品，あるいは自動車や電車などの乗り物を製作する前には必ず本物そっくりの**モックアップ**（模型）を作成し，製品に対する様々な評価を行う．色や形状などの外見上のデザイン評価に加え，製品の操作感覚なども重要なポイントとなる．そこで，コンピュータグラフィックスを用いて本物そっくりの映像を作成し，その映像をリアルタイムに表示することにより，まさに製品そのものが目の前に存在しているかのような錯覚を引き起こすことにより，製品の評価を行おうとするものである．視点を変えることにより，どこからでも自由な角度からの映像がリアルタイムに生成され，思考を中断させることなく，製品の評価を行うことができる．また，製品そのものを手にとって触れ，製品の操作感覚を確かめるのと同様に，バーチャルリアリティ技術により生み出された映像に触れ，製品に触れたときの感触を味わうこともできる．勿論，製品評価においては破壊試験もあり，製作した製品がどこまでの危険に耐えられるのか，あるいは製品が破砕された際の人間への影響はどの程度などかについての評価も重要であるが，現状の技術では，バーチャルリアリティ技術を用いて，これらの破壊検査及びその評価までを包含しようとはしていない．現状では，破壊試験などの最終評価に使用するモックアップは別途製作するとしても，最終評価を行う以前の製品評価のために繰り返し製作される数多くのモックアップに代替するものとして，バーチャルリアリティ技術を使用して作成したモックアップ映像を利用している．

バーチャルリアリティ技術とは，「現実には存在しないが，そのものの本質を備えているもの，あるいは現実に存在するものと同様の効果を与えるものを生み出す技術」であるから，コンピュータグラフィックスにより生成された映像は，製品が本来備えている品質を持ち，製品に触れたときと同じ感覚を与えることができるだけの特性を備えている必要性を要する．まさに，目の前に製品そのものがあるかのような錯覚を引き起こすことが必要であり，人間に錯覚を引き起させるためには，人間の五感に対して，現実に存在する製品が与えるのと同じ効果を与える必要がある．人間の五感には，視覚，触覚，聴覚，嗅覚及び味覚が存在するが，人間の五感の約 80%を占めると言われる視覚が最も重要である．その他の重要な五感としては，聴覚が挙げられる．人間の聴覚に対しては，音声認識，音声合成などの独自の技術分野があり，音楽や電話のオペレータへの応用として盛んに研究されている．一方，近年急速に研究が進められてきた技術として，触覚がある．バーチャルリアリティ技術

を用いて作成した映像は後で述べる立体視表示技術を使用することで３次元空間上に投影されるが，この空中に浮いた製品に触れることは以前においては非常に困難であった．しかしながら，近年では反力を表現できるデバイスの開発が盛んに進められており，このデバイスを使用すれば，３次元空間上に浮いた物体に触れ，その感触を得ることが可能である．このように，バーチャルリアリティ技術ではコンピュータグラフィックスで生成した物体がまさに目の前にあるかのような錯覚を引き起こすだけでなく，目の前にある物体に触れて，その感触を味わうことができる．そこで，本章では特に，人間の五感の約 80%を占めると言われる視覚と，近年盛んに研究が進められている触覚を取り上げてその技術動向を探ることにし，次章では具体的な応用システムを取り上げて，高品質画像のリアルタイム生成に関する検証を行うものとする．

6.1.1 視覚インタフェース

6.1.1.1 立体視

バーチャルリアリティ技術により生成されたコンピュータグラフィックス映像があたかも３次元空間上に存在するかのような錯覚を人間に与えるためには，立体視が不可欠となる．人間は左右両眼の目で３次元物体を捕らえるが，左右両眼の目の奥にある網膜上に結像した映像は視神経を通じて脳に伝えられ，脳に伝達された２枚の映像の差分から各物体までの距離を計算し，３次元立体映像として認識している．

ここで少し用語について定義しておく．絵画や漫画などで描かれている映像あるいは画像は通常，**２次元画像**と呼ばれる．２次元画像とは文字通り，ＸＹという２軸を持つ平面上に描かれた画像である．絵画手法には遠近法など，描く物体を立体的に見せる方法もあるが，描かれた画像は一方向から見た画像のみであり，角度を変えて別の方向から見ることはできない．これに対して，コンピュータグラフィックスで生成した画像は，元々ＸＹＺという３軸を持つモデリング空間上で定義されており，ワンショットとして捉えた画像が一方向から見た映像であっても，角度を変えて別の方向からの画像を生成することも可能である．このように３次元空間上で定義されたデータを基に生成された画像を**３次元画像**と呼ぶ．３次元画像は３次元空間上で定義されたデータを基に生成される画像であるから，基になる３次元空間上で定義されるデータには幾つかのものが存在する．一つは上記で述べたモデリング空間上で定義されたモデリングデータである．別のものとして，医療分野で使用されている**Ｘ線 CT（Computed Tomography）**や**磁気共鳴画像診断（MRI: Magnetic Resonance Imaging）装置**などによって得られる人体の断層像データがある[23,24]．断層像の一枚一枚は２次元画像であるが，複数枚の断層像を並べることにより３次元空間上で定義された画像データが得られるため，**ボリュームレンダリング**[25]という複数点列の可視化手法を用いると，視点を変更して任意の角度からの画像を生成することが可能である．

一方，**立体画像**とは，人間の左右両眼に対して視差を考慮した２枚の画像を与えることにより，脳において，与えられた２枚の画像から各物体までの距離を計算して生成された，３次元空間上に浮いている，あるいは飛び出して見える画像のことを言う．なお，**映像**と**画像**にも，本来それぞれ別の定義が存在し，信号処理の分野では映像あるいは映像信号，画像処理の分野では画像という言葉を用いることが多いと思われるが，本論文では特に明確な区別は行わず，ほぼ同義語として扱う．

さて，上記にて定義した３次元立体画像（映像），つまり，視点位置を変更して任意の角度から見える３次元空間上に浮かぶ映像を生成するためには，コンピュータグラフィックスを利用してモデリ

ング座標空間上で定義したモデリングデータを基に左右両眼に対して，視差を考慮した画像（**視差画像**）を生成する必要がある．そして，生成された二つの画像は確実に分離されて，左眼用画像は左眼に，右眼用画像は右眼に入力されなければならない．そのためには，HMD(Head Mounted Display)と呼ばれるデバイスを使用する方法が一般的であり，様々な装置が開発されている[26]．これらのデバイスには，メガネ式のものや頭から被るもの，あるいは双眼鏡のように覗き込むタイプのものまで様々なものがある．しかしながら，コンピュータグラフィックスを用いて人工的に生成した映像を，左右両眼の目に強制的に入力したのでは自然な立体視を行うことができず，逆に，強制的な立体視による障害が報告されている[27,28]．そこで，これらのデバイスを使用することなく，裸眼で自然な立体視を実現する研究が盛んに行われており，それらの方式をまとめると次の様になる．

1) **レンティキュラ方式：レンティキュラ**と呼ばれるカマボコ状の板から左右両眼の眼に対応する二つの映像を投影する．投影される映像の面はカマボコの凸凹に応じた角度が付けられているため，左眼用の映像は左眼に，右眼用の映像は右眼に入力されるように設計されている．このため，メガネなどのデバイスを用いることなく，裸眼状態で自然な立体視を行うことができる．しかしながら，この方式ではレンティキュラ板に設けられている映像面の傾斜が一定であるために視野領域が固定され，観察者は頭を移動させることができないという欠点を持つ[29]．同様な方式として，**パララックスバリア方式**や**イメージスプリッタ方式**[30]などがあり，視野領域を拡大するために，これらの方式を組み合わせるなどの様々な工夫がなされている[31]．

2) **視点追従法：**レンティキュラなどの方式における視野領域固定の問題を解決するために，観察者の頭の位置を認識してこの位置を追跡することにより，観察者の両眼には分離された左右の視差画像が確実に入力されるよう工夫した方法である[32,33]．

3) **多眼ステレオグラム：**上記二つの方法は基本的に左右両眼に入力される画像を分離し，左眼には左眼用の画像を，右眼には右眼用の画像を入力するように設計することで，メガネなし立体視を実現していた．多眼ステレオグラムも基本的な考えは同じであるが，左右それぞれの眼に入力される画像が一つではない点が異なる．つまり，左右それぞれの眼には複数の画像が重畳された状態で入力される．これらの合成画像を左右それぞれの眼で認識した後，人間の脳の処理により立体視を実現するものである．一つの眼に複数の映像が入力されるため，観察者は頭を移動させると，これら複数映像の重畳度合いが異なり，別の角度から見た映像として認識することができる．従って，観察者はメガネを用いずに立体視を行うことができるだけでなく，頭を自由に動かせることも可能となる[34,35]．

4) **ホログラフィ：**オーストリア・ハンガリー二重帝国の科学者ガボールによって 1947 年に発明された方法で，当時の論文には「新しい顕微鏡の原理」として紹介された．電子顕微鏡は光学顕微鏡と異なり，レンズによる集光での拡大を行うことはできない．従って，入力される電子線の波長と出力される可視光の波長との倍率で拡大表示することになる．この原理を応用し，電子線により撮像した像を可視光により再生すると，その倍率で拡大像が得られる．この像がホログラフィである．発明当時は，得られる像が二重になるなどの問題が発生し，解決策として参照光を斜めから照射する方法，レーザーを光源として使用する方法などが考案された．ホログラムは電子線により撮影した像を可視光で再生する方法であり，メガネをかけることなく 3

次元立体映像を得ることができる．しかも，視点を変更して任意の角度から見ることも可能であるが，解像度や色の問題，さらには動画への対応など課題も多い．

6.1.1.2 高臨場感

コンピュータグラフィックスで生成された映像が，あたかも目の前の３次元空間上に実在するかのような錯覚を人間に与えるためのもう一つの技術課題は，**臨場感**あるいは**没入感**である．通常のパーソナルコンピュータに備え付けられているディプレイは 14 インチから 21 インチくらいのものが多く，コンピュータのディスプレイを見ていても，同時に周囲環境が目に入ってくる．従って，コンピュータのディプレイ上に表示されているものは，距離をおいた仮想世界で生成されているものであり，現実の３次元空間に存在するものと錯覚することはない．しかしながら，画面が大きくなり，コンピュータグラフィックスで生成された映像以外，周囲環境が一切視界に入ってこない状況になると，観察者自身がコンピュータの中に入り込んだような没入感を与えることができる．そして，この没入感を実現する表示デバイスを **IPD(Immersive Projection Display)**と呼ぶ．

最も有名な IPD として，Illinois 大学で開発された CAVE(Cave Automatic Virtual Environment) を挙げることができる．CAVE は正面，左右の側面及び床面の４面に大型スクリーンを設け，背面投射により 3 次元立体映像を映し出す．観察者は液晶シャッターメガネを装着することにより，映し出された映像を立体視することができるだけでなく，４面の大型ディプレイが観察者を取り囲んでいるため，没入感も与えることが可能となる[36]．Illinois 大学で CAVE が開発されてから，日本国内でも同様な IPD が数多く開発された．有名なものとして，東京大学で開発された CABIN(Computer Augmented Booth for Image Navigation)，筑波大学の CoCABIN，郵政省の UNIVERS，岐阜県の COSMOS などがある．CABIN は CAVE に上面を加えて５面構成とし，COSMOS はさらに背面を加えて６面構成としている．

CAVE を始めとするこれらの IPD は全て平面ディプレイを用いて構成しているが，面と面との境目で映像の不連続性が発生し，没入感を損なう．そこで，球面スクリーンで IPD を構成しようという研究も行われている．(財)イメージ情報科学研究所で開発された半球面ドーム型ＶＲ体験システムは，水平３台，垂直２台の合計６台のプロジェクタを配置し，水平 180 度，垂直 90 度という大型半球面ドーム上に前面投射で映像を映し出す．平面映像を球面に投影することにより歪みが生ずるが，球面へのマッピングを２回行うことで歪みを解消している[37]．また，筑波大学では全周球面ディスプレイの研究も行われている[38,39]．

6.1.2 力覚インタフェース

バーチャルリアリティ技術のもう一つの特徴は，**フォースフィードバック**である．コンピュータグラフィックスを用いて作成した物体がまさに目の前にあるかのような錯覚を与えるだけでなく，実際に，その物体を手で触れて感じることができる．重い物体を持つと重く感じるし，軽い物体を持つと軽く感じる．また，硬い物体に触れた場合と柔らかい物体に触れた場合とで感じる力の大きさが異なる．人間の五感で言えば触覚ということになるが，人間の皮膚に感じるような微妙な感触を発生させることは困難であるため，通常は**力覚**という言葉で表現している．但し，現在では触覚を表現できるような力制御の研究も数多くなされている．

フォースフィードバックを実現するためには，力を表現するデバイスが必要であり，そのデバイスを**力覚提示装置（ハプティックデバイス）**と呼ぶ．表示されている物体から返される力（反力）を表現するためには様々な機構が考えられ，その機構によりハプティックデバイスを分類すると次の様になる．

1) **把持型：**ペン状，あるいは球状のものを掴んで反力を感じるタイプで，SensAble Technologies 社の PHANToM，筑波大学の Haptic Master などがある[40]．基本的には DC モータとパンタグラフ機構を用いて反力を発生させるものであり，入力はＸＹＺという位置３自由度に各軸周りの回転３自由度を加えた６自由度が一般的であるが，出力は位置の３自由度のみのものや，回転も加えた６自由度のものまで様々である．出力自由度が大きくなればなるほど，様々な反力を発生することができるという長所の反面，位置センサや DC モータなどの数が増えることにより，把持しているペンあるいは球などが重くなるという短所もある．このために，重力補償を行わない限り，自重による重みを感じたり，速く動かそうとしても DC モータの制御により遅く感じたりすることがある．基本的な反力発生アルゴリズムは次のようになる．例えば，球や立方体など簡単な仮想モデルを用意し，ペンや球など把持している物体で，仮想空間上に設定されているモデルを押すと，その仮想的モデルを構成する面あるいは点の位置と把持しているペン先あるいは，球の中心位置がずれる．物体を押す方向を正としてペン先とモデルとの差分を計測し，その差分に応じた反力を発生させる．実際の物体，例えば，机や壁などを手で押す場合，押す手の位置は不変であり，押す力に応じて反力が発生するが，ハプティックデバイスを用いた場合は，ペン先の位置と対象物との差分により力を発生しているため，対象物にめり込んだ状態で力を感じることになる．従って，実際に計測される位置を補正し，対象物上にペン先が乗っているようにグラフィックスの描画を行う必要がある．

2) **グローブ型：**手にグローブを装着して指を曲げると，指関節の曲がり度合いに応じて反力が発生する．グローブには関節の曲がり度合いを検出する位置センサとワイヤーが取り付けられており，指の曲げ度合いに応じた力でワイヤーを引っ張ることにより，反力が発生する．Virtual Technologies 社の CyberGrasp や，New Jersey 州立大学 Rutgers の Handmaster などがある[40,41]．把持型では手で握っているものに反力が発生し，ペンや球を介して人間の手に反力が伝達されるのに対し，グローブ型では直接手の指に反力を発生することができるので，実際の力覚に近いものを表現することができる．また，グローブに装着したワイヤーを引っ張る方式であるため，手が壁にぶつかった瞬間に反力を発生し，手が壁の中に入り込むと，壁の位置まで戻すように反力が発生するため，把持型のように手の位置を補正してグラフィックス表示する必要はない．また，５本の指に対して，それぞれ別々の反力を発生することが可能であり，力覚表現の自由度が高いと言える．但し，グローブに接続されたワイヤーをコントロールするための計算機が必要であり，かなり高価なシステムとなる．

3) **ワイヤーテンション型：**手や指に直接糸を装着し，この糸を引っ張る際の張力により反力を発生させる．このタイプのハプティックデバイスとしては，東京工業大学の SPIDAR[42]や東京大学のウェアラブルフォースディスプレイ[43]などがある．SPIDAR の場合は，親指，人差し指，中指及び薬指の４本の指に糸を装着し，各指に対する反力を発生させる．把持型やグローブ型のように，何かを持ったり装着したりする必要がないので，自然な反力の発生が可能となる．特

に，装着するものの重みがないため，自重による重力補償を行う必要がない．また，ウェアラブルフォースディスプレイとは，衣服を着るように人間の体に装着して自由に動くことのできるハプティックデバイスであり，ユーザが自由に動き回れるというメリットは大きい．**ウェアラブルコンピュータ**（衣服に装着し，大容量のバッテリを備えることにより野外でも使用可能なコンピュータ）の研究も盛んであるように，ハプティックデバイスの小型軽量化と共に，今後の研究における一つの方向性を示している

4) **接触型：**上記のハプティックデバイスは基本的に何かを把持したり，あるいは装着したりすることにより反力を発生させるものであるが，人間の指で直接対象物に触れることにより，物に触れた感触（いわゆる触覚）に関する研究も行われている．例えば，穴の空いた振動子を用いて人間が感じる触覚の精度を調べる研究[44]やステッピングモータにより回転するシリンダに触れることにより人間が滑り現象をどこまで認識できるのかについての研究[45]，あるいは弾性波動を用いた皮膚感覚の表現に関する研究[46]などがある．これらの研究は，力覚からさらに進んだ触覚についての研究であるが，解決する課題も多い．

6.2 医療応用システム

バーチャルリアリティ技術の応用として，6.1 節でも述べたが，実際のモックアップを製作することなく製品を評価するシステムがある．このバーチャルリアリティ技術を利用して製作したモックアップは**ディジタルモックアップ**と呼ばれ，近年急速に普及しているものの一つであるが，もう一つ，近年急速に発展してきているバーチャルリアリティ技術の応用として，医療応用システムが挙げられる．従来の医療では，専門の教育を受けた医師が長年培ってきた知識と経験を基にして医療行為を行っていた．その方法はまさに徒弟制度であり，名医と呼ばれる優秀な先生の医療行為を見ることにより，その手法を学ぶ方法が一般的であった．特に外科手術においては，全く同じ症例を持つ患者はほとんどなく，一人一人の患者に対して手術を執刀する医師（**執刀医**）の知識と経験から判断される手術の方式（**術式**）を，実際の手術現場（**術場**）で見学することにより学び取ってきたわけである．

しかしながらレントゲンが発明されて以来，人体を切り裂くことなく，その内部を見ることができるようになった．そして，MRI の登場により人体の３次元画像が得られるようになり，従来では複数のレントゲン写真を並べることにより判断していた人体内部に潜む病巣を，ボリュームレンダリングを用いることで簡単に３次元表示することが可能となってきた．この複数のレントゲン写真から人体内部に潜む病巣を発見するのにはかなりの熟練を要するが，ボリュームレンダリングのお陰で高度な技量を持たない医師でも病巣を容易に発見することができるようになってきた．それだけではなく，得られた３次元画像はコンピュータ内部に取り込まれているため，コンピュータを利用した高速解析が可能となり，病巣の可能性がある部位までコンピュータが教えてくれるようになってきている．このように，コンピュータは医療の世界に深く入り込み，現在ではコンピュータを使用した機器がなければ医療行為を行うことができないほどになっている．そこで，コンピュータグラフィックスの応用技術であるバーチャルリアリティ技術を用いて，現在使用あるいは研究されている医療システムについて本節でまとめる．そして，次章においては，バーチャルリアリティ技術を応用した医療システムを実際に構築し，コンピュータグラフィックスを用いて，医療システムにも充分通用できる画質を備えた高品質な映像を，医療行為を行う上で充分な応答性を持って生成する方式についての検討を行う．

6.2.1 手術ナビゲーションシステム

実際の手術中（**術中**）における手術操作の支援あるいはガイド（ナビゲーション）を行うシステムである．手術が効率よく行えるように執刀医を支援するものであり，一般に手術支援システムと呼ばれる．近年の手術は，**低侵襲手術**と呼ばれる手術が一般的になってきている．低侵襲手術とは，人体を大きく切り開くのではなく，必要最小限の小さな穴を開け，その穴から内視鏡など人体内部を見ることのできるカメラを挿入し，このカメラにより得られる映像を見ながら別の穴から挿入された手術器具（**術具**）を用いて行う手術の総称である．人体内部を見るカメラとして内視鏡が用いられる場合には，特に**内視鏡手術**と呼ばれる．低侵襲手術を行うことにより術中の出血量を抑えることができ，手術時間も短縮される．結果として，手術後（**術後**）の入院日数が減少し，早期退院，早期社会復帰が可能となる[47]．しかしながら，従来のように患者の体を大きく切開するわけでないので患者の体内を直接見ることができず，内視鏡などの人体内部を見ることができる装置で得られる手術映像を基に，その映像を表示するモニタを見ながら手術を行うことになる．当然のことながら，執刀医にかかる負荷は増大するため，執刀医にかかる負荷を減少し，コンピュータの支援による手術（**CAS：Computer Aided Surgery**）[47]をサポートするシステムが数多く開発されている．現在開発中の CAS システムには，次のような特徴がある．

1) **手術支援ロボット：**低侵襲手術では患者の体内を大きく切開することなく，必要最小限の穴を開け，その穴より術具を挿入すると共に，内視鏡などの体内に挿入したカメラで得られる映像を基に手術を行う．そのため，人体を大きく切開する**開腹手術**に比べ，術中における作業では細やかな神経と器用さが要求される．また，執刀医が術中，両手に持つ術具で手術を行っている間，内視鏡などのカメラを持つ助手が必要となる．しかしながら，人間が内視鏡を持つと，手ぶれなどにより映像が安定せず，手術の効率が低下することがある．そこで，この内視鏡を支持するロボット（Computer Motion 社製 AESOP）が開発されている．このロボットは，執刀医が発する音声を理解し，予め定められている音声命令により，執刀医の意のままに内視鏡の位置を変更することができる．また，患者の体内で行われる手術の操作には極めて繊細な技量が要求されるため，執刀医の動きを縮小して患者の体内に挿入された術具に伝達するロボットがあれば，この繊細な技量が要求される手術も容易となり，低侵襲手術の普及を促進することができる．現在では，術中における手術の手助けを行うマスタースレーブ型ロボット（Intuitive Surgical 社製 da Vinchi）も開発されている．このロボットを用いた手術では，執刀医は患者から離れた位置で顕微鏡を覗く．すると，患者体内の状況が 3 次元立体映像として現れ，執刀医が両手に持つマニピュレータを操作すると，その動きが 5:1 あるいは 3:1 の割合で患者の体内に挿入された術具に伝達される．結果として，体内における細かな作業を実行することができ，手術効率が向上する[48]．但し，このマスタースレーブ型ロボットにはフォースフィードバック機能が備わっていないという欠点もあり，現在さらに機能を向上中である．

2) **オーグメンティドリアリティ：**6.1 ではバーチャルリアリティ技術とは，現実には存在しないが，そのものの本質を備えているもの，あるいは現実に存在するものと同様の効果を与えるものを生み出す技術であると説明した．つまり，バーチャルリアリティ技術はコンピュータグラフィックスで生成した本物そっくりのものを表現する技術である．これに対して，現実世界には存在し得

ないが，コンピュータグラフィックス技術を使用することで，現実世界のものを扱う以上に都合がよくなるものを表現する技術，あるいは現実世界では見えないものを表現することにより，現実世界での表現を補足することができる技術を，**オーグメンティドリアリティ（Augmented Reality）**技術と呼ぶ[49]．例えば，手術を行う患者の病巣は患者の体内に存在するため，体の外からは見ることができない．しかしながら，手術前（**術前**）に撮影したX線CT，あるいはMRI装置により得られた３次元画像を用いれば，患者の体内に潜んでいる病巣を表示することができる．そして，この患者の体内にある病巣を患者の体表面上に重畳表示することができれば，低侵襲手術において必要最小限の穴を開ける位置を正確に把握することができ，結果としての手術の効率，あるいは成功率が高まる．また，脳外科などでは放射線を照射することにより，体内の腫瘍を攻撃する**放射線治療**と呼ばれる手術方法がある．この場合も，体内に潜む腫瘍の位置を正確に把握して，術前に得られた３次元映像を術中に，患者の体表面上に正確に位置合わせ（Registration）を行って表示することができれば，ピンポイントで腫瘍に放射線をあてることが可能となり，良性の組織にダメージを与えることなく，悪性腫瘍のみを攻撃することが可能となる．

6.2.2　手術シミュレーションシステム

前節で述べた手術ナビゲーションシステムは，術中に手術の支援を行うシステムであり，手術を成功させるためには極めて重要なシステムであるが，執刀医を支援する手術ロボットの安全性やオーグメンティドリアリティで表示する画像の品質が問題となる．術中に使用するロボットにおいては，患者の体内で暴走しないこと，及び問題があった場合にはすぐに停止する**フェールセーフ**機能[48]が備わっていることなどが必須となる．一方，オーグメンティドリアリティでは患者の体表面上に表示される患者体内の映像は，正確な位置合わせが必要であると共に，手術時間の経過に伴う臓器の変形が問題となる．つまり，いくら術前に高品質の画像を得，正確な位置合わせの基に表示を行ったとしても術中における変形までは考慮されていない．従って本来は，術中に変形した患者体内の映像を即座に取り込み，その映像を基に患者の体表面上に表示するシステムが必要となる．術中に患者体内の映像を取り込むための研究もなされているが，X線CTの場合は執刀医と患者への被爆量の問題など実現するための課題も多い．

そこで，術中では不十分となる部分を補い，手術を成功へと導くためには，できるだけ正確に手術の計画を立て，手術のシミュレーションを行っておく必要がある．このシステムを手術シミュレーションシステムと呼び，次の大きな二つの目的がある．

1) **術前計画：**実際の手術における手順を計画し，手術が手順どおりスムーズに遂行されるかどうかを模擬（シミュレーション）する．このためには，実際の患者の個人差に基づいたモデル作りが重要である．例えば，小耳症耳介形成術という手術がある．これは，小さな耳を大きくするための手術であり，シリコン樹脂を用いて形成した軟骨を耳に埋め込む手術である．この際，元々存在する耳の一部を削り取り，その削り取られた助軟骨から最適な耳のモデルを生成する．その際，実際の手術を行う前に，どの部分の助軟骨を採取し，どのような形のモデルを生成すればよいのかをシミュレーションしておく[50]．あるいは，低侵襲手術で脳腫瘍を取り除くために，細い管を脳に穿刺することがある．この場合も，どの位置からどの方向にどれだけの深さまで管を挿入す

ればよいのかを事前に把握しておく必要がある．術中支援も重要であるが，術前に得られた３次元画像を表示すると共に，**ファントム**と呼ばれるダミーの患者を使用して実際に穿刺を行うことにより，脳腫瘍だけを正確に採取できるのかどうかをシミュレーションしておくことは重要である[51]．

2) **手術練習：**手術を成功させるためには，実患者を対象とした手術計画や術中支援が重要となるが，これには個々の患者の特徴を反映させた正確なモデル作りが必要であり，この正確なモデルを基にした正確な模擬が必要となる．しかしながら，実患者の個々の特性を充分反映させたモデル作りは非常に困難であり，また時間の要する作業となる．従って，全ての病気，全ての患者に対して充分な術前計画を行うことはできない．また，この術前計画が有効となるのは，ある程度の熟練された技量を持つ医師に対してであり，最初に手術の基本的な技量を修得する必要がある．そこで，一般的な症例を対象とし，手術の基本技量の修得を目的としたシミュレータが数多く開発されている．以下，主な手術シミュレータを挙げる．

   a) **器官縫合：**BDI 社製 Virtual Surgery というシステムがある．血管，尿管などの器官の縫合を練習するシステムであり，SensAble Technologies 社製 PHANToM を用いてフォースフィードバックを実現している．シミュレーション後には，練習に対する評価結果が表示される．

   b) **脳外科手術：**脳の手術に関しては，郵政省，国立がんセンター，東京大学，東京医科歯科大学などかなり多くの機関で研究が進められている．基本的には，X線 CT あるいは MRI により得られた３次元画像を Volume Rendering あるいは，３次元画像からポリゴンを生成して Surface Rendering により表示し，脳穿刺の練習をするシステムが多い．

   c) **内視鏡手術：**脳外科手術と同様，X線 CT あるいは MRI により得られた３次元画像を基に体内の表示を行うが，実際に内視鏡を挿入した場合と同じ状況を作成し，内視鏡手術における器官の分岐をシミュレーションする．これは名古屋大学で積極的に研究が進められているが，HT Medical 社製 PreOp Endscopic Simulator という製品もある．

   d) **内臓手術：**肝臓を始めとする各種内臓のシミュレータであり，慈恵医大では３本の指で肝臓を摘んだ感覚を表現できるハプティックデバイスをススギ㈱技術研究所と共同で開発している．また，肝臓のような柔らかいものの変形を表現するために，剛体球が詰まった臓器モデル（球充填臓器モデル）の開発も行っている[52,53]．

   e) **眼科手術：**眼の手術シミュレータであり，Georgia Institute of Technology が開発した Eye Surgery Simulation，及び Colorado 大学が開発した Opthalmic Surgical Simulator がある．対象症例は共に白内障手術であり，白く白濁した眼球内部に存在する水晶体を人工レンズと置き換えることが最終目的であるが，現状では眼球表面の切開練習のみである．また，Mannheim 大学では，眼球奥に存在する眼底画像を見ることができるシミュレータ（EyeSi）を開発している．

第七章　手術シミュレータ

前章で述べたように，コンピュータグラフィックスを利用した応用技術であるバーチャルリアリティが様々な分野で活用され，数多くの応用システムが生み出されている．その中でも特に，医学と工学の技術が融合し，人間にとって最も大切な”命”を守るための医療応用システムが様々な研究機関で開発されている．

この医療応用システムを大別すると，術中に執刀医の支援を行う手術ナビゲーションシステムと，術前に手術の計画を立てたり，あるいは医師の技量を向上させるために手術の訓練を行ったりする手術シミュレーションシステムの二つに分けられる．手術ナビゲーションシステムは，執刀医の手術を支援するためのロボットの開発や，実際には見えない患者体内の病巣を患者の体表面上に重畳表示することにより，手術の効率を高めるオーグメンティドリアリティ技術の向上が重要な課題である．一方，手術シミュレーションシステムは，実際の患者を対象として手術の手順を確認したり，あるいは新しい術式を仮想的に試みることによって手術の成功率を高めたりする術前計画と，医師の技量向上を目的とする手術訓練の二つがある．今後の医療分野においては，個々の患者を対象とし，常に新しい術式を試みることによって，今まで助からなかった患者をも救うことのできる術前計画は非常に重要であると考えられる．

しかしながら，術前計画を行うためには，個々の患者の特性を充分に反映した非常に精密なモデルを作成する必要があり，この精密なモデルの作成には現状の技術で解決すべき課題が山積している．また，例え精密なモデルが完成したとしても，この非常に精密なモデルを基にして現実の物理現象に忠実に従うシミュレーション技術が必要となり，このためには膨大な計算時間を要する．個々の患者に対して術前計画のための時間を充分にかけることができる状況であれば，現状の技術でも術前計画は一つの解を与えている．ところが，医療の現場を見てみると，毎日のように数多くの手術が行われ，予定されていた手術だけでなく，突然運び込まれた急患があるという現状を考えると，残念ながら個々の患者に対して充分な術前時間があるとは言えない．また，医療現場では患者の数に比較して，医師や看護婦の絶対数が不足しており，人為的な医療ミスが多発しているだけでなく，全ての医師が充分な手術の技量を持っているとは言えないのが現状である．

医師の技量を向上させるためには通常，動物や遺体を使用した手術の練習を行うが，実際の患者である生体と，動物や遺体との間にはかなりの差がある．特に，遺体を使用する場合には死後硬直を回避することはできないため，術中の手術感覚は術場でしか学ぶことができない．また，動物を使用する場合でも，人間特有の病気は動物に発病しないため，微妙な手術の技量はやはり術場でしか学ぶことができないのが現状である．さらに，最近の工学技術の発達により，最新式手術器具が数多く開発されているが，これらの最新式手術器具の使用方法を学ぶのも重要である．最もよい例は低侵襲手術の代表例である内視鏡手術の術式学習方法である．内視鏡により捕らえられる映像のみを頼りにして，最小限の切開により開けられた小さな穴から術具を挿入した手術は非常に困難であり，実際の手術を行う前に充分な訓練が必要となる．術前計画においては，膨大な時間をかけても充分に正確な解を得ることが重要であるが，一方，手術練習に関しては，実際の手術を再現できるだけの充分な応答速度を持つシステムの構築が要求される．患者の体を切開したのに何の変化も現れず，数分後に切開部分が裂けていくようでは，手術シミュレーションとしての役割を果たしているとは言えない．

そこで，本章では今までに検討してきたコンピュータグラフィックスの高速化技術を用いて，その応用システムとして近年特に注目を集めている手術シミュレータの高品質な映像生成を，手術訓練という目的を充分果たすことのできる高速描画処理により実現するシステムの開発について述べ，

本論文の最終目的であるコンピュータグラフィックスを用いた高品質画像のリアルタイム生成に関する研究の最終成果を示すことにする．

7.1 眼科手術

6.2.2 で記述したように，手術練習を目的とした手術シミュレータにも様々なシステムが存在する[54-56]．対象部位も脳，眼，鼻，歯，肺，心臓，肝臓，骨など様々であり，現在のところ，これらを全て包含するような万能手術シミュレータは存在せず，それぞれ個別のシステムとなっている．コンピュータグラフィックスを用いた高品質画像のリアルタイム生成に関する研究を，実際のシステムを開発して検証するためにはどの部位を選んでもよいのであるが，実際の手術についての医学的な知識がなければ手術シミュレータを開発することは不可能である．そのような状況の中，三菱電機㈱情報技術総合研究所と，旭川医科大学眼科教室との間で共同研究契約を締結することになり，眼科手術に関する知識を習得する機会を得ることができるようになった．

眼科医は通常動物の眼，特に豚の眼を使用して手術の練習を行うが，使用される豚の眼は死後硬直により硬くなっており，生体の眼を対象とした手術練習を充分に行えないのが現状である[57]．また，網膜剥離などの症例は人間特有の病気であるため，このような人間特有の病気を持つ動物眼は存在せず，本当の練習は実患者を対象とした実際の手術に頼らざるを得ないという現実が存在する．これらの症例では動物や遺体を用いた手術練習が不可能であり，医療現場では手術教育が大きな課題となっている[58]．そこで，近年急速に発展しているバーチャルリアリティ技術を使用して，医療側から早急な開発を要望されている手術シミュレータを，医学的知識を充分習得することのできる眼科手術を対象として構築し，コンピュータグラフィックスを用いた高品質画像のリアルタイム生成に関する実検証を行うことにした．

本節ではまず，眼科手術について概説することにする．眼科手術はマイクロサージェリー[59]と呼ばれ，非常に精細な熟練技量が要求される手術である．成人の眼球は直径約 24㎜ の球とみなすことができる．この小さな眼球に直径約 1㎜ 程度の小さな三つの穴（**ポート**）を開けて術具を挿入し，一つの術具を固定すると共に他の二つの術具を左右両手に持って手術を行う，いわゆる低侵襲手術の一つである．手術を行う医師（**術者**）は，手術顕微鏡と呼ばれる双眼鏡型の顕微鏡を覗くと眼球の立体映像を見ることができる．左右両手にはそれぞれ別の術具を持ち，これらの術具を駆使して，直径 24㎜ という非常に小さな空間内で手術を遂行する．また，足元には**フットスイッチ**と呼ばれる 2 種類のペダルが用意されており，このペダルを踏むことで，手術顕微鏡の位置設定，手術映像の拡大/縮小，あるいは術具動力の制御などを行うことができる．従って，眼科用手術シミュレータの構築には，次のような要求仕様がある．

1) **立体視：**両眼視差を利用した左右２枚の立体視映像を生成するだけでなく，生成された画像は精緻な立体感覚を与えるだけの充分な画質が要求される．例えば，成人の眼球半径は約 12㎜ であり，眼底付近は半径約 6㎜ のほぼ平面とみなせる円となる．このほぼ平面とみなせる円上にほとんど平面である膜が増殖しているが，このほぼ平面とみなせる円とほとんど平面である膜との距離が分かるくらいの精緻な立体感覚を生み出す必要がある．何故なら，この立体感覚が欠如すると，誤って術具で眼底に触れてしまい，眼底から出血する恐れがあるからである．立体視の方法としては，生成した一つの画像を人間の瞳孔間距離だけ左右にずらすことにより立体画像を生成する方法もあるが，この方法では，左右両眼

の輻輳角を考慮していないため本来は隠れて見ない部分が見えたり，逆に見えるはずの部分が隠れたりする結果となる．従って，輻輳角を考慮した正確な左右２画像を生成する必要がある．

2) **術具感覚：**眼科手術では，眼球表面に設けられた小さなポートから左右の術具を挿入し，この眼球内部に挿入された二つの術具を左右両手に持って手術を行う．この際，左右両手に持った術具を眼球内部に挿入するときの感覚が重要である．このためには，左右両手に持つ術具の先端における３次元空間での座標値，及び各軸周りの回転角を位置センサなどで読み取り，術者が持っている術具の位置制御を行って正確な位置，及び姿勢情報を基に術具を描画しなければならない．同時に，術具を眼球内部に挿入する際には，術具と眼球との間に摩擦力が発生するため，この摩擦力を正確に模擬しなければならない．このためには，ハプティックデバイスを使用して，正確に計算された反力を術者に伝達する必要がある．

3) **応答性能：**シミュレータとしての必須要件はリアルタイムな応答速度である．いくら正確な物理モデルを基に物理現象を忠実に模擬しても一枚の画像が表示されるまでに，数分あるいは数時間かかるようでは手術の練習は行えない．手術の練習を行うためには，通常，１秒間に 30 枚もの画像を生成する必要があると言われている．つまり，30Hz の応答性能が必要となる．しかも，上記で述べたように眼科手術では立体視可能な手術顕微鏡を用いた手術を行うため，両眼視差に基づく左右２枚の画像を生成しなければならない．結果として，30Hz の応答性能を確保するためには，１秒間に 60 枚もの画像を生成する必要がある．さらに，左右両足に設けられたフットスイッチ及び左右両手に持つハプティックデバイスからの情報を基に，術者に対して反力を発生すると同時に，手術の進行状況に応じて，腫瘍部位の変形や出血などの処理を行わなければならない．そのためには，手術シミュレータを構築するプラットホームの選定や，搭載する CPU の数，さらに動作するプログラムの負荷分散などを考慮して，手術シミュレータの構築を行わなければ必要な応答性能を確保することはできない．

眼科手術にも様々な症例が存在するため，手術シミュレータを構築する上でターゲットとする症例を決めなければならない．そこで次に，眼科手術の代表的な症例について記述することにする．眼科手術の主な症例を列挙すると，次のようになる．

1) **白内障手術：**眼の中には**水晶体**というレンズが存在し，外界から入射される光は水晶体で集光されて**網膜**上に像を結ぶ．網膜上に結像された映像は眼底奥にある**視神経乳頭**から視神経を介して脳に伝達され，映像として認識される．ところが，年齢と共に，この水晶体の収縮が悪くなり，硬直して網膜上に像が結ばれなくなる．これが**遠視**と呼ばれるものであるが，さらに症状が悪化すると，水晶体が白濁して外界からの光が網膜上に到達しなくなる．この結果，徐々に目が見えなくなるという病気が白内障である．従って，白濁して硬くなった水晶体を細かく砕いて**眼外**に取り出すと共に，水晶体の代わりとなる人工レンズを**眼内**に挿入する．この手術が白内障手術である[57]．

2) **緑内障手術：**正常な人の目では，眼内の圧力（**眼圧**）は一定に保たれている．ところが，眼圧が一定に保たれなくなると，様々な病気を引き起こす．このように，眼圧が一定に保たれなくなったことにより引き起こされる病気が緑内障である．通常は眼圧が正常値よりも高くなることが多いため，上昇した眼圧を下げるための手術が多い．このためには，目の奥に穴を開けて眼内の圧力を下げる方法などがあり，非常に難しい手術の一つである[60]．

3) **網膜・硝子体手術:** 眼内にある水晶体の下は大きな空洞となっており，この中には**硝子体**（しょうしたい）と呼ばれるゲル状組織（卵の白身のような液体）が存在している．通常この硝子体には弾力があり眼圧を一定に保っているが，年齢と共に硝子体が収縮を始める．すると，眼の底の部分（**眼底**）には，水晶体を通過してきた光が結像する網膜があり，硝子体と網膜は接しているために硝子体の収縮により，網膜が引っ張られて剥離を引き起こす．これが**網膜剥離**であるが，それ以外に，詰まった血管が破裂することにより新たな膜が生成されたり，あるいは網膜上の一部に穴が開いたりする病気がある．このように，網膜と硝子体とは一体となっているため，これらに関する病気の手術を総称して網膜・硝子体手術と呼ぶ[61]．

4) **角膜手術：**眼球組織の中で唯一外気に触れている部分であり，最も丈夫に作られているが，逆に角膜に傷がつくと眼内を守るものがなくなるため，様々な病気が引き起こされる．弱くなった角膜を正常な角膜と取り替える**角膜移植**もこの手術の一つであるが，通常は角膜に混入した異物を取り除くことが手術の主な目的となる[60]．

7.2 対象症例

前節では，充分な医学的知識を習得することのできる環境を考慮して，手術シミュレータの対象を眼科手術としたが，眼科手術にも大きく分けると前述した４種類の手術が存在する．この中で，眼科医が最初に取り組むのが白内障である．白内障は，年齢と共に硬くなった水晶体を人工レンズと置き換える手術であり，眼科手術の中でも最も症例数が多い手術である．また，死後硬直しているとはいえ，豚眼を用いた練習が行える状況にあるため，まず白内障で一流の手術技量を身につけた後，他の手術を行うのが一般的な教育方針となっている．これに対して，他の三つ手術はどれを取っても困難な手術であるが，特に網膜・硝子体手術は対象となる範囲が広く，網膜剥離のように動物の眼には発病しない人間特有の病気が多い．従って，バーチャルリアリティ技術を使用した手術シミュレータの開発が他の手術に比べて強く要求されているため，網膜・硝子体手術を対象手術として手術シミュレータを構築することにした．

しかしながら，網膜・硝子体手術を対象手術としても，個々の症例はそれぞれ異なるため，具体的な症例を設定し，しかも個々の症例に依存する多様性を取り除いて，対象とする症例を単純化しなければ，手術シミュレータとして取り扱うことのできるモデルを作成することはできない．そこで，網膜・硝子体手術の中でも比較的頻繁に現れ，しかも動物眼を使用した練習が困難な次の２症例を対象症例として取り上げることにした．

1) **黄斑前膜症：**眼底中心部（**黄斑部**）にある血管がつまった後に破裂して，その破裂した部分には新たな膜（**増殖膜**）が生成される．この生成された増殖膜は，例えば手に擦り傷を負った場合にできる**かさぶた**のようなものであり，体表面上に生成したかさぶたは自然に剥がれてなくなるが，眼内に生成したかさぶたは自然には剥がれず，放置しておくとさらなる増殖を繰り返す恐れがあるため取り除かなければならない．このように黄斑部の網膜上面（あるいは前面）に増殖膜が生成される症例を黄斑前膜症と呼ぶ．

2) **加齢性黄斑変性症：**黄斑部に増殖膜が生成されるという点では黄斑前膜症と同じであるが，黄斑前膜症では網膜上面に増殖膜が生成されるのに対し，網膜の一部が盛り上がり，その中に増殖膜が生成する点が異なる．年齢を重ねるにつれて黄斑部が変形することにより生ずる症状であるため，加齢性黄斑変性症と呼ばれるが，網膜の下に生成された増殖膜を取り除くことが主目的であるため，この手術は**網膜下手術**と呼ばれる．

上記では対象となる二つの症例を取り上げ，それぞれの症例の特徴について述べた．次に，対象となる症例の手術手順について概略を述べることにする[62]．

＜黄斑前膜症の手術手順＞

1) **麻酔：**眼球の奥に注射して麻酔を施す**局所麻酔**が基本である．従って，眼球には麻酔がかけられているために手術中の痛みは感じないが，手術中も患者の意識はあり，術後，医師が患者に話しかけて眼を動かせることにより，手術の最終確認を行うこともある．但し，かなりの痛みを伴うことが予想される場合などには**全身麻酔**を施し，術中は患者の意識が全くないこともある．

2) **開瞼：**人間は自然な状態では瞬きをしたり，瞼を閉じたりするが，術中は常に瞼を開けた状態を保たなければならない．そのために，**開瞼器**という器具を使用して強制的に瞼を開く．眼の幅（**瞼裂幅**）が小さい人は瞼を大きく開くことができないため，瞼を少し切開して眼を大きく開かせることもある．

3) **結膜切開：**瞼の根元部分の膜を**結膜**といい，眼の瞳以外の大部分を覆っている膜（**強膜**）を露出するために，結膜を切開する．360 度にわたる円状の切開と，強膜を部分的に露出させるための部分切開とがある．当然のことながら出血を伴うために，電気コテ（**焼灼器**）で止血する．

4) **制御糸：**瞼を強制的に開かせると共に，眼があまり動かないようにするための糸をかける．通常は，眼の耳側にある筋肉（**外直筋**）と鼻側にある筋肉（**内直筋**）の２箇所に制御糸をかけるが，360 度にわたる円状の結膜切開を行った場合には，眼の頭側にある筋肉（**上直筋**）及び足側にある筋肉（**下直筋**）にもかける．

5) **強膜切開：強膜刀**と呼ばれる先の尖ったメスで，外直筋の下方に穴（**強膜切開創**）をつくる．この場合，強膜切開創より眼内に存在する硝子体が流出する恐れがあるため，止金で抑える．出血した場合には焼灼器で止血を行う．

6) **灌流液注入：**後で述べるが，眼内に存在する硝子体を切除吸引して，眼外に取り出す．眼内にある硝子体が除去された分だけ眼圧が低下するので，眼圧を一定に保つために，**灌流カニューレ**という装置を上記にて設けた強膜切開創より挿入し，術中，外れないようにナイロン糸で縫合/固定する．その後，灌流カニューレより，**灌流液**（通常は生理的食塩水）を眼内に注入し，術中，眼圧を一定に保つ．

7) **強膜切開：**上記同様，強膜刀にて強膜を切開し，左右の術具を挿入するための強膜切開創を設ける．強膜切開創を設ける位置は，患者の頭を 12 時とした場合に，2 時と 10 時の位置である．また上記同様，眼内の硝子体が流出しないように，止金で抑えておく．

8) **レンズリング装着：**網膜・硝子体手術は，眼の奥の手術であるために，眼（角膜）の上にコンタクトレンズを置き，このコンタクトレンズを通して眼内を観察する．**レンズリング**とはコンタクトレンズを置くための金属の輪であり，術中外れないように縫付ける．

9) **術具挿入：**左右の手に持つ術具を挿入する．右利きの医師は通常，左手に**ライトガイド**と呼ばれる術具を持つ．眼内は真っ暗であり，術中，手術室の明かりも消されるために，何の光もない状態となる．このような状況の中では，左手に持つライトガイドが唯一の明かりとなり，眼科外科医はこのライトガイドの明かりのみを頼りにして，眼内の様子を観察する．一方，利き手である右手には，症例に応じて様々な術具を持つ．黄斑前膜症の場合は，後述する硝子体切除吸引と増殖膜剥離が主な作業工程となるため，硝子体切除吸引を行うための**硝子体カッター**，及び増殖膜剥離を行うための**ピーラー**と**鑷子**が主な術具となる．

10) **コンタクトレンズ設置：**助手がレンズリングの上にコンタクトレンズを置くと，眼内の手術が開始される．コンタクトレンズには様々な種類のものが存在し，通常は，コンタクトレンズの表面が水平にカットされているものを使用する．このコンタクトレンズでは，レンズを通過する光が屈折することなく直進するため，眼底中央部を観察することができる．一方，レンズ表面が斜めにカットされていると，カットの角度に応じてコンタクトレンズを通過する光が屈折し，眼底周辺部を観察することができる．また，硝子体切除吸引により除去された硝子体の代わりに生理的食塩水を注入する場合と，特殊なガスを注入する場合とでは，これらの物質の屈折率が異なる．このため，硝子体の代わりに注入されている物質の屈折率に応じて，眼内の様子をより明確に観察するための様々な種類のコンタクトレンズが用意されている．

11) **硝子体吸引切除：**上記にて挿入したライトガイドと硝子体カッターの先端が硝子体の中に入っていることを確認してから硝子体切除吸引を始める．足元に用意されているフットスイッチの一つは手術顕微鏡の視点位置，フォーカス，ズームなどを調整するためのものであるが，もう一つ別に用意されているフットスイッチは硝子体切除吸引のための吸引圧力調整用である．従って，フットスイッチを押下すると硝子体カッターの先に取り付けられている刃が稼

動し，刃の部分に触れている硝子体はカッターの吸引圧力によりカッター内部に吸引された後に切除され，カッターの内部を通過して眼外へ排出される．フットスイッチの踏込み量に応じて吸引圧力が変化するので，硝子体同士の癒着が強い部分では吸引圧を上げ，逆に，硝子体と網膜との癒着が強い部分では網膜剥離を引き起こす可能性があるため，吸引圧を下げる．硝子体切除吸引の順番としては，水晶体後方部分から始め，手前にある**前部硝子体**，中央部分にある**中央硝子体**の切除吸引を済ませた後，網膜に近い**後部硝子体**の切除吸引を行う．この間，眼内にある硝子体をくまなく切除吸引するために，レンズ表面が斜めにカットされているコンタクトレンズと交換したり，コンタクトレンズを回転させることによりカット面の方向を変更させたりすることにより，眼内のあらゆる部分を観察しながら硝子体切除吸引を行う．

12) **増殖膜剥離：**黄斑部上面に生成している増殖膜の剥離を行う．増殖膜は薄く透明な膜であり，所々に網膜との癒着個所が存在する．丁度，和紙の上に張られたセロハンテープを剥がす動作に類似している．膜の周辺部分に探りを入れて，掴みやすそうな個所を見つけると共に，発見された掴みやすそうな個所をさらに少し持ち上げることにより，容易に全体の膜を剥がすことを試みる．そのために，２種類の術具が用意されている．一つはピーラーと呼ばれる術具であり，注射針の先端を少し折り曲げたような器具である．この折り曲げられた先端部で，掴みやすそうな個所を探すと共に，その部分を少し持ち上げる．もう一つの術具は鑷子と呼ばれる術具であり，ピンセットに似た器具である．術具を持つ手元の部分に鑷子の先端部を開閉させるスイッチがあり，このスイッチを押下することにより，鑷子の先端部が閉じるように設計されている．この鑷子を用いて，先ほどピーラーで少し持ち上げられた増殖膜の個所を掴んでゆっくり剥がすことにより，増殖膜を剥離することができる．剥離された増殖膜はそのまま眼外へ排出される．

13) **網膜剥離復位：**網膜が眼底から剥離している個所があれば，その部分をレーザーにより眼底に接着する．この際，次の加齢性黄斑変性症のように，網膜の下に生理的食塩水が入り込んでいる場合は，灌流カニューレより特殊なガスを眼内に注入し，網膜下に入り込んでいる生理的食塩水を眼外へ排出した後，レーザーによる接着を行う．

14) **器具除去：**左右両手に持つ術具を眼内から取り出すと共に，コンタクトレンズ及びレンズリングを外す．

15) **強膜縫合：**上記にて作成した三つの強膜切開創を縫合する．

16) **結膜縫合：**取り付けた制御糸及び開瞼器を外し，切開した結膜を縫合する．

＜加齢性黄斑変性症の手術手順＞

本症例の手術手順は黄斑前膜症の手術手順とほぼ同一であり，異なる部分は 12) 増殖膜剥離のみであるため，この部分についてのみ手順を記述する．

a) **網膜膨張：**網膜の一部が山のように盛り上がり，その下に増殖膜が生成されている症例であるため，網膜の下に術具を挿入して増殖膜を取り出さなければならない．そのためには，増殖膜が生成している部分を大きく膨張させて，膜を掴みやすくするための空間を作成する必要がある．このために，**サブレチナルカニューレ**と呼ばれる先の曲がった術具を挿入して少し盛り上がった網膜上の側面に穴を開けると共に，開けられた穴から生理的食塩水を注入することで，盛り上がった網膜内部の圧力を上昇させ，増殖膜剥離を行う空間を大きくする．サブレチナルカニューレの先端が曲がっていることにより，網膜に傷をつけることなく，山のように盛り上がった網膜の側面に穴を開けることができる．

b) **増殖膜剥離：**上記サブレチナルカニューレの挿入と生理的食塩水の注入により膨張した網膜に**サブレチナル鑷子**と呼ばれる術具を挿入して，網膜下に生成している増殖膜を掴み，網膜外へ取り出す．この術具は黄斑前膜症で使用した鑷子と基本構造は同じであるが，術具の先端がサブレチナルカニューレと同じ様に曲げられている点が異なる．先端の開閉は手元のスイッチにより行い，網膜外へ取り出された増殖膜はそのまま眼外へと排出される．

c) ***ガス置換：***サブレチナルカニューレを使用して網膜下に生理的食塩水を注入したために，網膜剥離復位を行う前には，眼内に存在する生理的食塩水を特殊なガスで置換して網膜下に存在する生理的食塩水を眼外へ排出しておく必要がある．**フルートニードル**と呼ばれる術具を眼内に挿入し，灌流カニューレからは生理的食塩水の代わりに特殊なガスを眼内に注入すると，ガス球が眼内で膨張して，このガス球の圧力により眼内に存在する生理的食塩水はフルートニードルの内部を通過して眼外へ排出される．

上記２症例とも手術手順としては，麻酔及び開瞼に始まり，強膜及び結膜縫合で終了する非常に長い工程がある．しかしながら，麻酔や開瞼あるいは切開や縫合は動物の眼を利用すれば練習可能であるため，必ずしも全工程の練習が行える手術シミュレータを構築する必要はなく，動物の眼を使用しても練習が不可能な手術の工程を対象として，バーチャルリアリティ技術を利用した手術シミュレータを開発すればよい．上記手術手順の中で，動物の眼を使用しても練習不可能な部分は，動物の眼では発病しない病気である増殖膜を剥離する動作の練習ということになる．しかしながら，増殖膜剥離にあたって，その前の工程である硝子体切除吸引は重要な手順の一つである．何故なら，硝子体切除吸引を充分に行わない状態で増殖膜剥離を行うと，増殖膜及び網膜が残存している硝子体と癒着し，網膜剥離を引き起こす恐れがあるからである．また，加齢性黄斑変性症の場合には，網膜下に生理的食塩水を注入しているため，ガス置換により眼内の生理的食塩水を完全に眼外へ排出しておかなければ網膜剥離復位を行うことができない．

従って全工程の中で，黄斑前膜症の場合には硝子体切除吸引及び増殖膜剥離を，加齢性黄斑変性症の場合には硝子体切除吸引，網膜膨張，増殖膜剥離及びガス置換を対象症例における対象手術手順として，手術シミュレータを開発することにした．

7.3　眼科用手術シミュレータ

前節までに記述した症例及び手術手順を対象として，手術シミュレータを開発した．シミュレータのハードウェア構成を図 7-1 に示す．

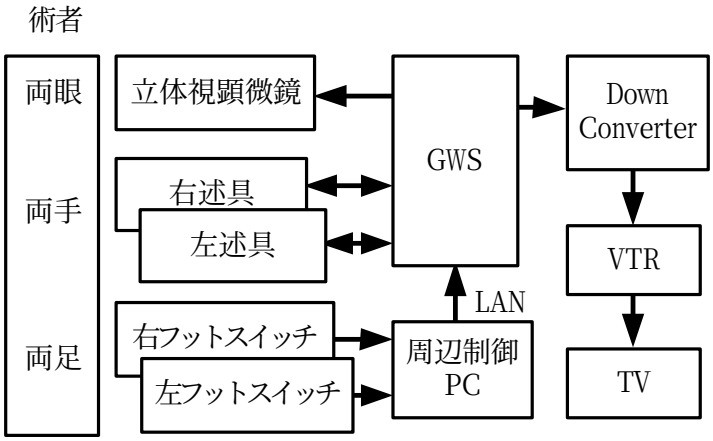


図 7-1 手術シミュレータのハードウェア構成

図 7-1 に示すように，術者は両眼で手術顕微鏡を覗きながら，両手に術具を持ち，両足ではフットスイッチを使用して手術を遂行する．従って，シミュレータでも同様の環境を構築する必要がある．バーチャルリアリティの立体視技術には様々なタイプのものが存在するが，通常用いられるのは偏光メガネあるいは頭から被る HMD タイプのものが一般的である．しかしながら，手術顕微鏡は両眼で手術の対象（**術野**）を覗き込むタイプなので，眼科用手術シミュレータとしては，双眼鏡型の立体視ディスプレイが立体視のユーザインタフェースとしては最適である．そこで，n.Vison 社製 Virtual Binoculars を採用することにした．他にも双眼鏡型立体視ディスプレイは製品化されているが，安価な装置のほとんど全ては液晶を用いたタイプである．液晶を用いることにより，デバイスのコストを削減して安価な表示装置を作成することが可能となるが，表示される映像の画質が手術の練習として充分とは言えない．特に，画像を構成している各画素の境目が認識できたり，輝度や色調が均一ではないため，手術練習用シミュレータとしては不適格であると判断した．これに対して，CRT を使用したタイプは手術の練習としては充分満足できる画質を提供しており，画素の境目が見えることなく，また輝度や色調も均一であり，実際の手術映像を撮影したビデオと比較したところ，ほぼ同等の画質を備えていることが分かった．Virtual Binoculars も液晶タイプと CRT タイプの双方が用意されているが，本シミュレータでは CRT タイプを採用することにした．

次に術具であるが，7.1 でも述べたように眼球と術具との間に発生する摩擦力を模擬し，ユーザにはこの摩擦力により生ずる反力を伝達する必要がある．従って，反力を発生することのできるハプティックデバイスを使用することとし，ハプティックデバイスの先駆けであり，世界的なシェアと共にその有効性が実証されている SensAble Technologies 社製 PHANToM を選択した．PHANToM は入力に関しては位置ＸＹＺの３自由度に加え，各軸周りの回転３自由度を加えた６自由度を持つが，出力に関しては反力ベクトルの方向を表す３自由度しか持たず，回転力であるトルク力を発生することはできない．現在では出力６自由度を持つ PHANToM も開発/製造されているが，非常に高価であると共に，開発当初には存在しなかったために，出力３自由度の PHANToM Model 1.5 を採用することにした．開

発当初はトルク力発生の不可能性が問題視されたが，模擬すべき重要な反力は眼球と術具との摩擦力であり，回転に関しては，発生するベクトルの方向を術具回転の方向を考慮して変更することにより，トルク力の簡易的な模擬を行うことで問題を解決している．

フットスイッチであるペダルも，実際の手術を模擬する上で重要な役割を果たす．手術シミュレータとして実環境と同等の環境を構築するためには，術場で用いられている手術顕微鏡及び硝子体切除吸引圧などの制御を行うための手術装置に接続されているフットスイッチと同等の機能を持つペダルを用意しなければならない．そのため，実際の手術で用いられている手術顕微鏡（ZEISS 社製手術顕微鏡 OPMI CS）及び手術装置（Alcon 社製手術装置アルコンアキュラス VS400）に接続されているフットスイッチを購入し，内部構造を調査すると共に，これらのペダルを計算機に接続できるように改修を加えた．但し，実際の手術で使用されているフットスイッチには様々な機能が用意されているが，本シミュレータでは対象となる症例及び手術手順を実現する上で必要な機能のみを抽出し，これらの機能のみを使用可能としている．

バーチャルリアリティ技術を駆使した手術シミュレータを構築するためには，その中心となる計算機の選択は極めて重要である．特に，コンピュータグラフィックスで生成する立体視画像は，手術映像という極めてリアルな画像を 1 秒間に 30 回，しかも両眼合わせて 60 枚生成しなければならない．さらに実際の手術では，手術顕微鏡内に映し出されている映像は術者だけでなく，執刀医を補助する助手も見ることができる．ただ単純に同じ映像を表示するだけであれば，術者用に生成した立体視画像の映像信号を分配器を用いて分ければよいことになる．しかしながら，これではバーチャルリアリティ技術を駆使した手術シミュレータの有効性が半減する．例えば，運転免許取得の際に使用するドライブシミュレータや航空機の操縦訓練のために使用するフライトシミュレータでは，実際には発生してはいけない事故を仮想的に発生させることにより，事故回避能力の向上を訓練の目的の一つとしており，訓練後にはコンピュータが判断した客観的評価を得ることができる．このことを考慮すると，手術シミュレータの場合もただ単純に術者と同じ映像を表示するのでなく，実際には得られないが訓練としては有効な映像（例えば，手術映像を別の角度から見た画像など）を表示することにより，シミュレータとしての有効性を高めることができる．そのためには，中心となる計算機にはさらなる画像生成能力が要求される．さらに，上記にて採用を決定した PHANToM や Virtual Binoculars の計算機への接続可否，つまり，接続インタフェースとそのデバイスドライバの有無をも考慮しなければならない．そこで，高度なグラフィックス生成能力を持ち，PHANToM や Virtual Binoculars の接続も可能な高性能グラフィックスワークステーション（GWS: Graphics WorkStation）である SGI 社製 Onyx2(R10000 195MHz 2CPU Infinite Reality)を採用することにした．但し，実際の手術に使用されている手術顕微鏡や手術装置に接続しているフットスイッチを Onyx2 に接続するためのインタフェースは非公開であり，ドライバも存在しないため，これらのフットスイッチは周辺制御 PC に接続し，フットスイッチより入力された情報は周辺制御 PC から GWS へ LAN を経由して伝達される構成とした．

また，手術教育の現場では，実際の手術をビデオに録画して指導医が研修医に術式を教授したり，あるいはビデオ録画されている手術についての反省や新方式の検討なども行ったりしているため，本シミュレータでも Down Converter を介して，計算機の映像信号をビデオ録画できる構成としている．

次に，手術シミュレータのソフトウェア構造について説明する．

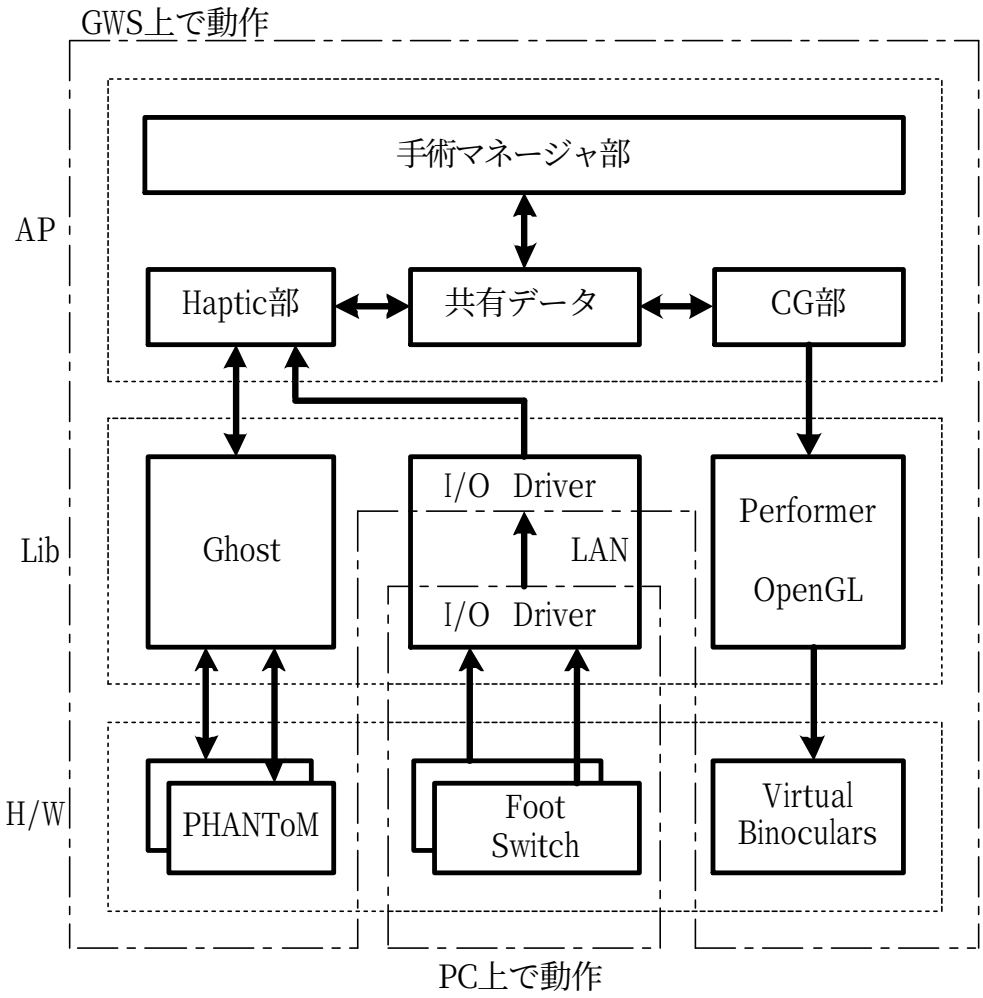


図 7-2 手術シミュレータのソフトウェア構造

図 7-2 に示すように手術シミュレータのアプリケーション（AP: APplication）部分は，手術マネージャ部，Haptic 部及び CG(Computer Graphics)部の三つに大別される．手術マネージャ部は手術の進行状況を司る部分であり，例えば眼内に挿入した術具が眼底に接すると出血を引き起こすとか，手術装置のフットスイッチを押下すると硝子体切除吸引が開始されるなどの手術の進行状況に応じて発生する現象を模擬する．Haptic 部は基本的にハプティックデバイスを用いて反力を発生させる部分であるが，反力を発生させるためには，例えば術具が眼内に挿入されて摩擦力が生じているとか，術具が増殖膜を掴んでいるのかなどの判断も行わなければならない．従って，反力発生に必要な干渉計算や黄斑部上面に生成している増殖膜の変形計算も行っている．CG 部は基本的に表示処理を行う部であり，眼球内外部や術具の他に，Haptic 部にて計算された増殖膜の変形結果を表示する．これら三つの部の間には共通データが存在し，この共通データを介してお互いの情報をやり取りしている．例えば，Haptic 部にて計算された増殖膜の変形結果も，この共通データを介して Haptic 部から CG 部へ情報が伝達される．

反力を発生させるハプティックデバイスは SensAble Technologies 社製 PHANToM を採用しているため，その制御には同社より提供されている Ghost Lib. を使用している．通常，映像生成におけるリアルタイムとは 30Hz を意味し，1 秒間に 30 枚の画像を生成できればよいが，ハプティック制御の応

答速度は30Hzでは充分とは言えず，通常は1kHzの高速応答性能が要求される．特に，硬い物質を表現するためには最低1kHzの応答性能が必要であり，1kHzが保証されないとハプティックデバイスの正常動作も保証されない．一方，CG部のリアルタイム応答は30Hzであるから，CG部に情報を提供する30Hzのルーチンとハプティックデバイスを制御する1kHzのルーチンが混在することになる．しかしながら，Ghost Lib.にはこれらの制御用ライブラリが用意されているため，細かなチューニングを行わない限り，応答性能の異なる２種類のルーチン制御は可能である．但し，1kHz という応答性能はコンピュータグラフィックスでの応答性能である30Hzの30倍以上であり，術具を挿入した際に眼球上に設けたポートと術具との間で発生する摩擦力の力覚と，目にフィードバックされる視覚との間の整合性を保つ必要がある．

前述したように，手術顕微鏡及び手術装置に備え付けられているフットスイッチを直接GWSであるOnyx2に接続することは困難であったために，一旦PCに接続し，LANを介して情報を送信している．従って，GWS及びPC双方には情報をやり取りするためのI/O Driverが必要であり，このDriverはPC及びGWSの仕様に合わせて自作した．なお，PCにはmtt社製heronシリーズ（CPUはAMD-K6 MMX 233MHz 64MB Main Memory）を使用している．また，PCを介して得られたフットスイッチの情報はHaptic部により読み取られた後，共通データに書き込まれることによって，手術マネージャ部及びCG部へ伝達される．

手術シミュレータの心臓部とも言えるコンピュータグラフィックスによる映像生成であるが，コンピュータグラフィックスの世界では業界標準となっているOpenGLを採用することにした．OpenGLとGhost Lib.を混在させたサンプルプログラムはPHANToMの開発元であるSensAble Technologies社より提供されおり，両ライブラリの整合性は非常によい．但し，シミュレータの世界ではOpenGLの上位ライブラリとして知られているPerformerがよく用いられる．Performerには，シミュレータの開発で重要なフレームレイト制御や，CADで作成したモデルデータのローダーなどが予め用意されており，開発工程を短縮することができる．そこで，コンピュータグラフィックスの映像生成にはPerformerを用いることとし，高速性能が必要な部分は直接OpenGLを用いて記述することにより性能向上を図っている．

立体視映像の生成にも様々な方法が存在するが，ここでは**Quad Buffer + フィールドシーケンシャル方式**を採用した．通常のコンピュータグラフィックス描画方式では，フレームメモリへの書き込み途中の映像が表示されることを防ぐために，２枚のフレームメモリを用意し，メモリへの書き込みが終わった段階で瞬時に書き込むフレームメモリと表示するフレームメモリを切り替える．このようにすることで，フレームメモリに書き込んでいる途中の映像が表示されることはない．この方式を**Double Buffer方式**と呼ぶ．このDouble Buffer方式を立体視映像生成にも適用すると，立体視を行うために左右２枚のフレームメモリが必要であり，さらにそれぞれのフレームメモリに対してDoubleのBufferが必要となるから，合計４枚つまり，QuadのBufferが必要となる．この方式をQuad Buffer方式と呼ぶ．Quad Buffer方式を採用すると，左右両眼に対して生成された映像の切り替えに対する同期が完全に保証されるため，左右の画像表示に時間的なずれを生ずることない．手術シミュレータとして採用したGWSであるOnyx2にはこのQuad Bufferを実現するための充分なフレームメモリが用意されており，この同期が取れた左右両眼の映像をフィールドシーケンシャル信号として出力している．フィールドシーケンシャル信号とは，ＲＧＢの信号を別々の信号線ではなく，同一の信号線上に時分割で各色に対応するフィールドに分けて伝送する方式の信号であり，同一の信号線を用いて送信することにより，左右両眼に対する同期を取ることができる．従って，Quad Buffer + フィールドシーケンシャル方式を採用すると，完全に同期の取れた左右両眼に対応する信号がVirtual

Binoculars に表示され，立体視において左右両眼に対応する映像の間で時間的なずれを生ずることはない．

7.4 PC ベースリアルタイムシミュレータ

前節では，高性能 GWS をベースとした手術シミュレータの構築について述べた．高性能 GWS をベースとした手術シミュレータでは高品質な映像をリアルタイムに生成することができる反面，コンピュータグラフィックスを用いた映像生成能力の高さに比例して高価となる．また，立体視の実現にあたっては，左右両眼の同期を完全に取ることのできる Quad Buffer + フィールドシーケンシャル方式を採用し，この方式を実現することのできる Virtual Binoculars を使用したが，システム構築に用いた Virtual Binoculars は片眼 SXGA(1280x1024)の解像度の映像を表示する能力を持つこともあり，非常に高価な装置である．さらに，左右の術具を実現するために採用した PHANToM Model 1.5 も世界に先駆けて製品化されたこともあり，３自由度を持つ反力しか表現できないものの，やはり高価な装置である．従って，これらの装置を組み合わせて構築した手術シミュレータシステムは，生成する映像の品質及び応答性能に関して手術練習として使用するための充分な条件を備えているが，システムが高価となるため，多くの眼科医に気軽に利用して頂き，手術技量の向上に貢献するという本来の目的を達成することが困難となる．

そこで，より多くの眼科医に気軽に利用して頂くという本来の目的を達成するために，近年急速に普及すると共に，年々性能が向上している PC をベースとした手術シミュレータの構築を検討した．但し，PC を使用してシステムを構築した結果，システム価格を下げることができたとしても，生成する映像の品質低下あるいは応答性能の劣化を引き起こしては本末転倒となる．本節では，PC を使用して手術シミュレータを構築するが，映像の品質及び応答性能をできるだけ保ったまま，システム価格を低下させるという方針で，PC ベースのリアルタイム手術シミュレータを構築する方法について述べる．

最初に検討すべき課題はベースとなる PC を何にするかであり，現実感のある手術映像を高速に生成するために搭載するグラフィックスアクセラレータの選択である．GWS ベースのシミュレータを PC 化する最大の目的は，必要とされる画質及び応答性能を維持したまま，より多くの眼科医に使用して頂けるようにシステムの低価格化を図ることであり，世の中での普及度及び搭載されるグラフィックスアクセラレータの種類及び性能を考慮すると，WindowsNT ベースの PC が最適であると考えられる．現在では，Windows2000 に搭載される高速なグラフィックスアクセレータの種類も増えたが，開発当初，Windows2000 は未だ出荷されていなかった．また，Linux に搭載されるグラフィックスアクセラレータの種類も徐々に増えてきているが，普及度及び搭載されるグラフィックスアクセラレータの性能面では，WindowsNT に勝るものは現状ない．

そこで，コンピュータグラフィックスを用いた手術映像生成において，最も影響力の強いグラフィックスアクセラレータの選択にあたり，各種グラフィックスアクセラレータの性能評価を行った．性能評価の項目は次の２点である．

1) OpenGL のベンチマークテストである ViewPerf(CDRS)
2) OpenGL で生成されたアプリケーションの性能評価

1)に関しては，グラフィックスアクセラレータの製造元より対象となる基板の性能評価結果が提供されているが，ベンチマークテストのソースも公開されているので，評価対象となる PC での測定が可能である．2)に関して本来は手術シミュレータのデモ版を作成し，GWS と PC の双方でデモ版による性能評価を行うと，PC に搭載されるグラフィックスアクセラレータを変更した場合の性能比較だけでなく，GWS と PC を比較して，PC を用いた場合に GWS を用いた場合とほぼ同等性能が出せるのかどうかを調べることも可能となる．

しかしながら，手術シミュレータのデモ版を製作するということは，手術シミュレータ自身を開発することとほぼ同程度の工程を要することから，既に市販されているソフトウェアを用いた評価が可能かどうかを調査した．調査の結果，GWS ベースの手術シミュレータで採用した Onyx2 と開発対象としている WindoesNT 版 PC の双方で動作するグラフィックスライブラリとして，Vega というソフトウェアが存在することが分かった．Vega とは，MultiGen・Paradigm 社が開発した OpenGL の上位ライブラリであり，GWS をベースとした手術シミュレータで採用した Performer とほぼ同等の機能を持つソフトウェアである．そこで，これら二つの項目に対し，対象となる GWS 及び PC を用いて性能評価を行った．性能評価結果を表 7-1 に示す．

表 7-1 グラフィックスアクセラレータの性能比較

計算機	ＣＰＵ速度	アクセラレータ	ViewPerf 性能	Vega 性能
GateWay	266MHz	FireGL4000	32.92	13fps
GateWay	266MHz	AccelGALAXY	56.04	14fps
GateWay	400MHz	AccelGALAXY	80.72	20fps
Intergraph	450MHz	WildCat4000	71.02	25fps
Onyx2	195MHz	InfiniteReality	161.29	30fps

注１） ViewPerf は高い値ほど高性能であることを示す．
注２） fps: Frames Per Second: １秒間あたりの生成画像枚数であり，高い値ほど高性能である．

表 7-1 より，同一のグラフィックスアクセラレータ（例えば AccelGALAXY）を用いた場合，CPU 速度の向上（266MHz から 400MHz）に対応して ViewPerf 性能（56.04 から 80.72）及び Vega 性能（14fps から 20fps）も向上していることが分かる．しかしながら，GateWay(400MHz) + AccelGALAXY と Intergraph(450MHz) + WildCat4000 を比較すると，ViewPerf の性能が高いからと言って必ずしも Vega の性能が高いわけではないことが分かる．これは一般的なベンチマークテストである ViewPerf の性能向上のために様々な高速化対策が取られていることに拠るものである．手術シミュレータは一つのアプリケーションであり，GWS では Performer を用いて記述していたことより，Performer とほぼ同等機能を持つ Vega による評価を優先させて，グラフィックスアクセラレータとしては Intense3D 社製 WildCat4000 を採用することにした．

Intense3D 社製 WildCat4000 を搭載した Intergraph が現状の性能評価では最も高い値を示しているが，それでも従来のシミュレータでベースとなっていた Onyx2 の性能には及ばない．従って，PC を使用した手術シミュレータの構築にあたっては，ソフトウェアの最適な負荷分散，あるいは画質を劣化させない程度のポリゴンの削減などを行い，最大限の性能を引き出すための工夫を行う必要がある．この高品質画像のリアルタイム生成に関する検討については次節で述べることにする．

次に，立体視の実現方法について検討しなければならない．Onyx2 では Quad Buffer + フィールドシーケンシャル方式を採用することにより，完全に同期の取れた左右両眼に対応する立体視画像を生成することができた．しかしながら，上記性能評価にて採用を決定した WildCat4000 では Quad Buffer を実現するだけの充分なフレームメモリ容量を持たない．これは WildCat4000 に限らず，ほとんど全ての PC に搭載されているグラフィックスアクセラレータに対して言えることでもある．従って，Quad Buffer + フィールドシーケンシャル方式とは別の方式を考えなければならない．通常の PC に搭載されているフレームメモリでは SXGA(1280 x 1024)を表示することは可能である．また，n.Vision 社製 Virtual Binoculars には，片眼で SXGA を表示することのできる高性能仕様と，片眼 VGA(640×480)を表示することのできる低価格仕様があり，低価格仕様も液晶ではなく CRT を使用している．さらに眼科外科医の評価では，立体視用ディスプレイとして CRT タイプのものを使用した場合，VGA の解像度でも手術練習としては充分使用可能であるという結果が得られている．そこで，PC から出力される SXGA(1280×1024)の画面を４分割し，４分割された画面うち二つの画面に左右両眼に対応する手術映像を生成すると共に，この分割された２画面を別々の Scan Converter を使用して切り出せば，Virtual Binoculars へ入力する二つの映像信号を生成することができる．この際，立体視用の左右両眼に対応する画像の同期についての問題が発生するが，PC から出力される映像信号を一度分配器に入力し，この分配器を介して二つの Scan Converter に入力すると，実験上では，左右の画像信号にずれを認めることはできなかった．従って，立体視の実現方法としては，Scan Converter による切り出し方式を採用することにした．この際，SXGA(1280x1024)の解像度を持つ全画面を４分割し，そのうち分割された二つの画面を左右両眼に対応する画像生成に使用すると，分割された画面領域のうち二つの画面が残ることになる．そこで，この残った画面領域の一つに執刀医を補助する助手用の画像を，また別の領域にユーザインタフェース用入力画面を表示すれば，分割された画面の全てを有効に利用することができる．この様子を図 7-3 に示す．

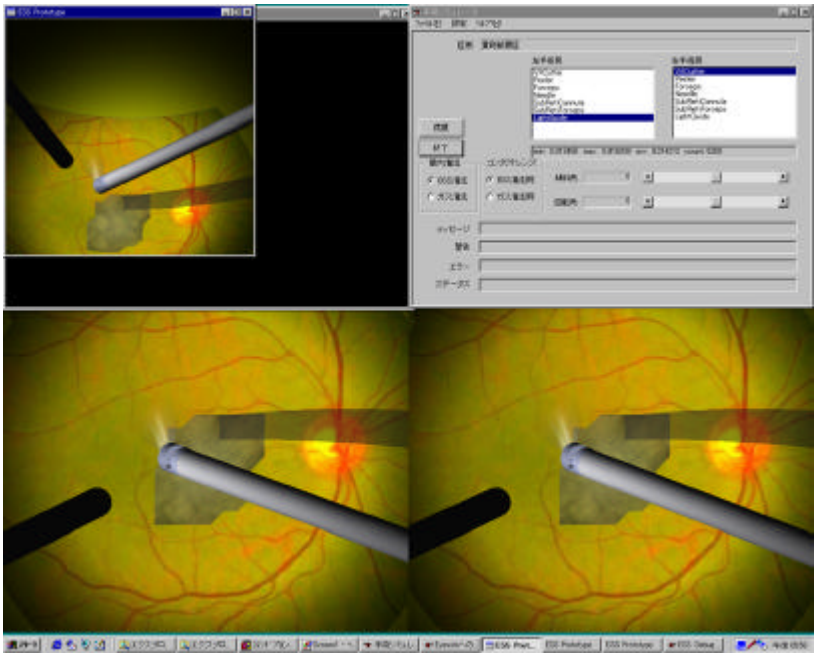


図 7-3 立体視実現のための画面分割

次に，左右の術具について検討する．Onyx2 に左右の術具として接続した PHANToM Model 1.5 は出力３自由度しか持たないとは言え，かなり高価な装置である．一方，眼科の手術において，最も重要な力覚は術具を挿入した際の術具と眼球との間に発生する摩擦力である．従って，高価なハプティックデバイスを用いなくても，物理的に摩擦力を発生させることは可能である．摩擦力発生の方法としては，術具挿入部分に**ブッシュ**と呼ばれる輪を取り付け，このブッシュの締め付け具合を調整することにより，摩擦力を発生すると共に，発生する力を調整することも可能となる．また，PHANToM を用いた場合には，PHANToM の先端に発生している反力を，術具と眼球との接触部から発生しているように反力の位置を変更する必要があった．また，出力３自由度しか持たないためにトルク力を発生することができず，発生する力の方向を変更することにより，擬似的なトルク力を生成していた．これに対してブッシュを利用する方法では物理的な力を発生することができるため，術具に発生させる力を計算する必要がない．従って，Onyx2 に直接接続していた PHANToM の代わりにブッシュを用いた模擬術具を周辺制御 PC に接続して LAN 経由でメインの PC に情報を伝達することにより，反力発生に要していた時間をコンピュータグラフィックスによる画像生成に振分けることができる．こうすることにより，性能評価時点では Onyx2 と PC との間に残っていた性能差分を吸収し，高品質な映像をリアルタイムに生成できる可能性が出てきた．

さらに，PHANToM の先端には位置センサが取り付けられており，この位置センサと PHANToM の先端部分が干渉するために，左右の術具先端を 50㎜ 以下にすることができないという問題があった．これに対して，成人の眼球は直径約 24㎜ であるから，実際の手術の環境を構築するためには，左右の術具先端を 20㎜ 程度にまで近づける必要がある．ブッシュを用いた方式では，ブッシュの取り付け位置に位置センサを左右対称構造で取り付けると共に，術具先端位置をブッシュ取り付け位置の先に仮想的に設定することにより，この仮想的な術具先端位置では左右の間隔を 20㎜ とすることが可能となる．結果として，PHANToM では解決できなかった左右の術具間距離の問題を解決することができた．図 7-4 にブッシュを用いて製作した模擬術具を示す．

また，Onyx2 をベースとした手術シミュレータでは，実際の手術顕微鏡（ZEISS 社製手術顕微鏡 OPMICS）及び手術装置（Alcon 社製手術装置アルコンアキュラス VS400）に接続しているフットスイッチを購入して改造を加えることにより周辺制御 PC へ接続して，必要な情報を LAN 経由で Onyx2 に伝達していたが，実際の手術顕微鏡に接続しているフットスイッチは高価であるため，必要な機能のみを抽出して簡易なフットスイッチを製作することにした．この簡易版フットスイッチの製作も，システム価格の低価格化に寄与している．図 7-5 に製作した簡易版フットスイッチを示す．

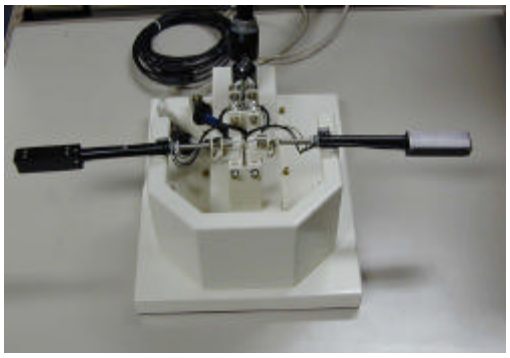


図 7-4 ブッシュを使用した模擬術具

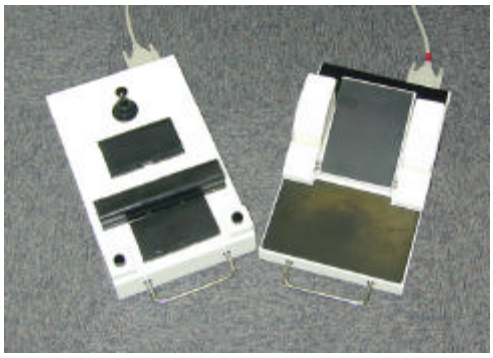


図 7-5 簡易版フットスイッチ

上記のように検討した結果，PC をベースとしたリアルタイム手術シミュレータの構成は，図 7-6 のようになる．なお，最終的にはメイン PC として，WildCat4000 を搭載可能な WindowsNT 版 PC とし，開発時点における価格対性能比を考慮して，DELL 社製 Precision 610（Pentium Ⅲ 550MHz 2CPU）を採用している．

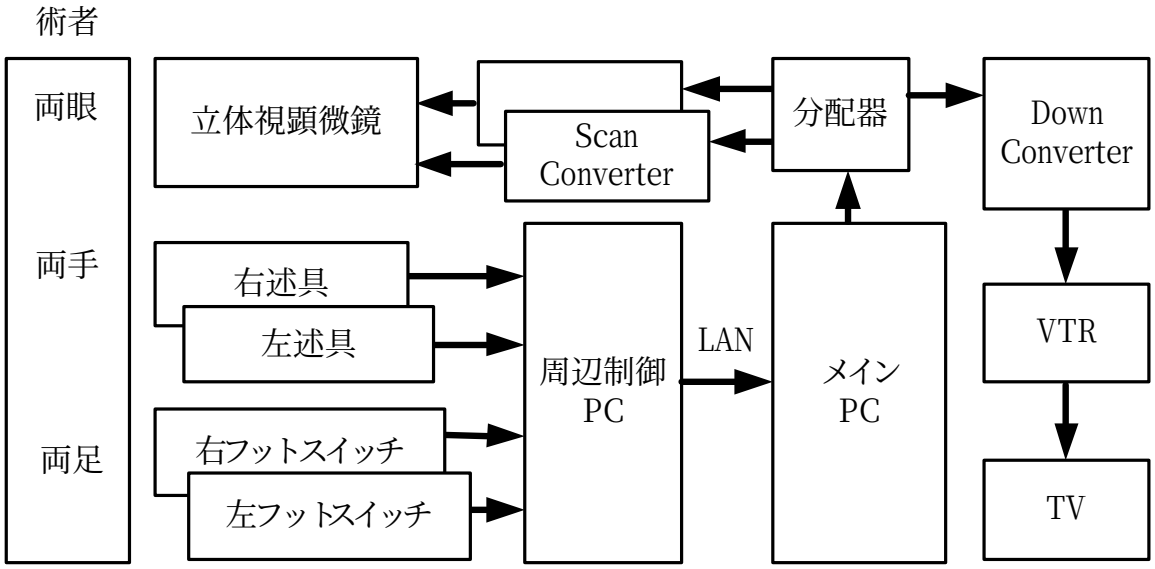


図 7-6　PC ベースリアルタイム手術シミュレータのハードウェア構成

なお，PC ベースリアルタイム手術シミュレータのソフトウェア構成は，図 7-7 のようになる．図 7-6 に示すように，左右の術具は周辺制御 PC に接続し反力を発生させる必要がないため，左右のフットスイッチと同じ I/O Driver で位置及び各種スイッチの情報を入力し，LAN 経由にてメイン PC 上で動作する Device 部に伝達している．ここで，GWS ベースの手術シミュレータではハプティックデバイスである PHANToM を採用していたが，PC ベースの手術シミュレータではハプティックデバイスを用いず，ブッシュを使用した模擬術具を使用しているため，Haptic 部を Device 部と名称変更している．また，CG 部で行うコンピュータグラフィックスを用いた手術映像の生成に関して，性能評価時点では Performer と同等機能を持つ Vega が候補となったが，使用ライセンスの問題及びリアルタイム制御機能など必要な機能の全てが網羅されていないことなどが判明したため，OpenGL を用いて必要な機能を全て C++言語にてコーディングしている．

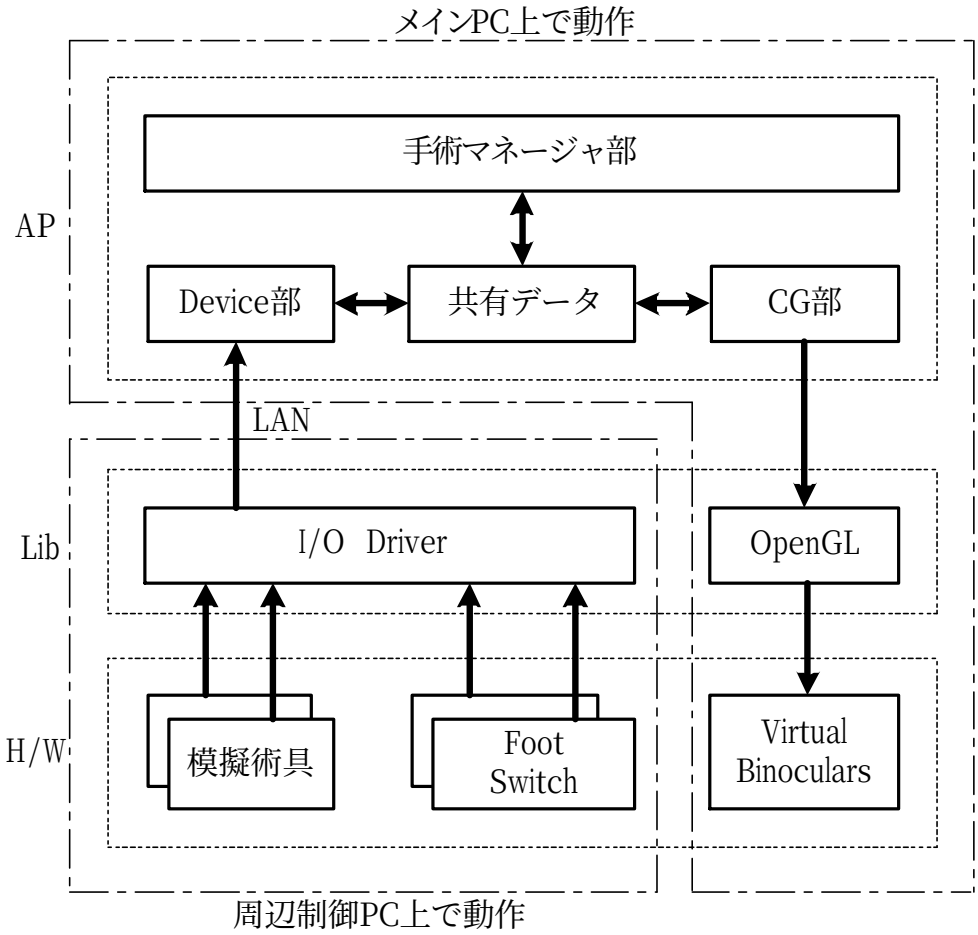


図 7-7 ＰＣベースリアルタイム手術シミュレータのソフトウェア構造

7.5 高品質画像のリアルタイム生成方式

前節では，高性能 GWS をベースとして開発した眼科用手術シミュレータをより多くの眼科医に使用して頂くことを目的に，PC をベースとして開発するための方策について述べた．必要な機能を抽出して，機能面での劣化を引き起こすことなく，低価格化するための指針については明確となったが，最終的に生成される手術映像の画質及び応答性能に劣化が生ずるようでは本末転倒となる．本節では，高性能 GWS から PC へとベースとなる計算機を変更して低価格化を図ったにも拘わらず，生成される手術映像の画質と応答性能を満足させるための方式（**高品質画像のリアルタイム生成方式**）について述べる．

7.5.1 黄斑前膜症の増殖膜剥離モデル

7.2 節で記述したように，本論文では黄斑前膜症と加齢性黄斑変性症の二つの症例を対象として手術シミュレータを構築し，高品質画像のリアルタイム生成に関する実証を行うことにした．手術という非常に高度な品質が要求される映像を，手術練習を行うにあたり充分な応答速度を持って生成するためには，物理現象を忠実に模擬することのできる精密なモデルを基にして，映像品質を劣化させることなくリアルタイムに画像を生成することのできる高速なモデルを考案する必要がある．本節

では，黄斑前膜症の手術シミュレータを構築するにあたり，高品質な映像を生成するためには膨大な時間を必要とする増殖膜剥離を対象として，映像品質を劣化させることなくリアルタイムに立体視画像を生成する方法について記述する．

通常，自由曲面の変形を扱う場合には，**有限要素法**や**質点バネモデル**が用いられる[63]．有限要素法とは，対象物を細かな要素に分割し，各要素における変形が全体としてどの様に影響を及ぼすのかを調べて，細かく分割された多数の要素から成る物質の変形を取り扱うものであるが，各要素の変形処理には膨大な計算量が必要とされることから，リアルタイム応答には不向きであると言われている．一方，質点バネモデルは細かく分割された各要素にある程度の質量を与えると共に，この質量を持つ点（**質点**）の間をバネで結合する．物質の変形に応じてバネの長さも変化するので，バネの変位に応じた内力が発生すると仮定して，物質の変形を取り扱うものである．例えば，一枚の増殖膜を有限要素法あるいは質点バネモデルにより各要素に分解すると，図 7-8(a)のようになる．図において，ハッチされた部分が増殖膜の内部を表し，これを有限要素法あるいは質点バネモデルで扱えるように，細かな要素に分解したものの外周を太線で示す．基本的には四角形あるいは三角形で構成される要素の集まりとなる．図 7-8(a)に示す全ての要素に対して逐一質点とバネ力に基づく運動方程式を解いていたのでは非常に時間がかかり，高速に変形画像を生成することはできないので，リアルタイムに画像を生成するためには分割する要素の数を減らして，図 7-8(b)のような構成を考える．図 7-8(a)及び(b)の太線を比較すると明らかなように，画像生成のリアルタイム性を重視して物質の構成要素数を削減すると，画質が劣化していることが分かる．

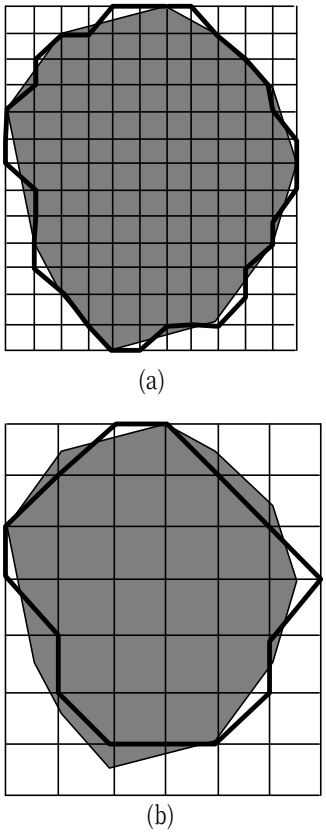


図 7-8 モデル分割数と画質の関係

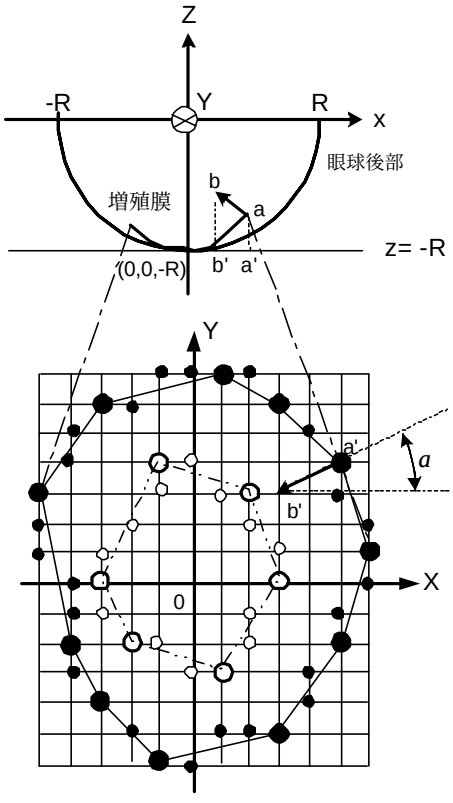


図 7-9 増殖膜モデル

そこで，画質を劣化させることなくリアルタイムな画像生成を実現するために，図 7-9 のような増殖膜のモデル（**高速膜剥離モデル**）を考える．図 7-9 は，図 7-8(a)と同じ分割数で増殖膜の要素を構成している．しかしながら，増殖膜内部の点全てに対して運動方程式を解いていたのでは，リアルタイムな応答性能を実現することはできない．従って，分割数は同じであるが，変形計算を行う重要点を抽出し，この重要点に対する計算のみを行う．眼科手術において膜の剥離を行う場合，膜の外周部分に触れて周辺部を少し持ち上げた後，持ち上がった部分を摘んで膜を剥離する．このとき重要となる点は膜の外周部分であるが，眼底網膜上に接着している増殖膜を剥離するためには，増殖膜と網膜との接着部分を調べなければならない．この接着部分が広いと，網膜と増殖膜との癒着力が大きく，膜は剥がれにくい．膜を剥離するにつれて接着部分が小さくなり，癒着力も下がるため，膜は剥離し易くなる．図 7-9 において●で示した点が増殖膜の外周部分を構成する点であり，○で示した点が網膜との癒着を示す境界線を構成する点である．増殖膜モデルを作成する際，膜を構成する格子上の点を全て与えるのは困難であるため，大きな丸（●及び○）を与えると，それらの間を補間して自動的に小さな丸（●及び○）を生成するようにしている．

ここで，増殖膜上の点 a を摘んで図中の点 b の方向に膜を持ち上げることを考える．点 a 及び b から z=-R 平面上へ下ろした垂線の足を a ’ 及び b ’ とする．ここで，$\overrightarrow{ab}$ 及び $\overrightarrow{a'b'}$ を含む平面を考える．この平面はX軸と α の角度をなしているから，この平面を Z 軸周りに時計方向へ α だけ回転するとＸＺ平面に一致し，図 7-10(a)のように表される．

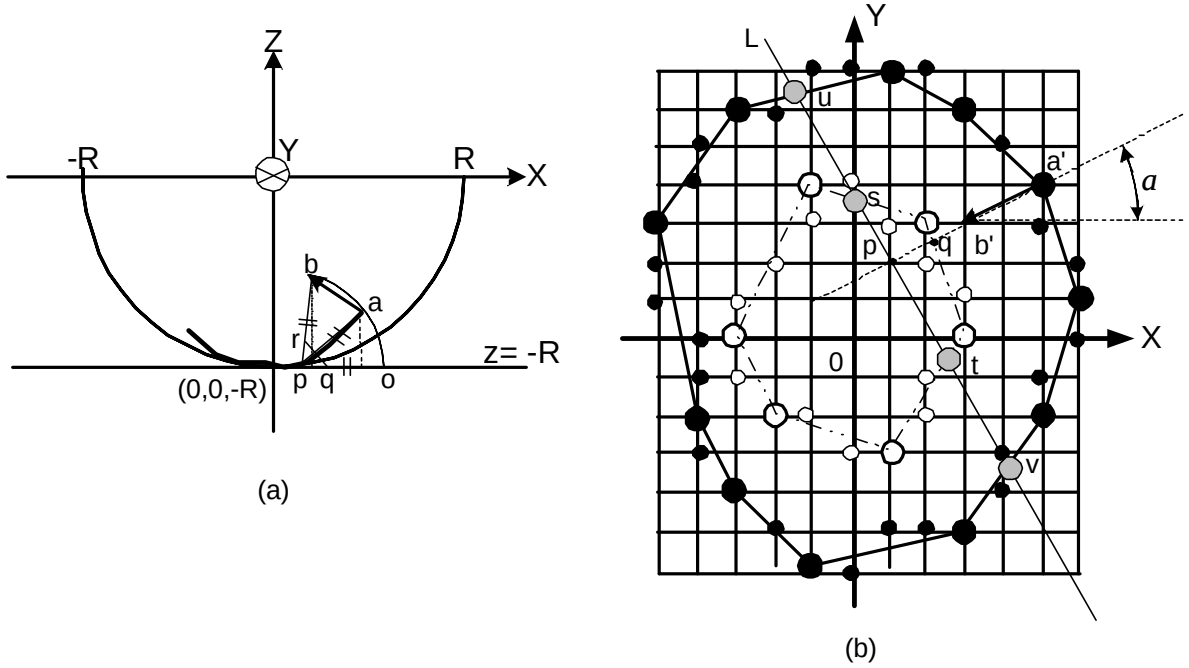


図 7-10 膜剥離モデル

点 a から点 b まで円孤を描きながら膜を持ち上げたとすると，その膜剥離における回転中心 p を求めことができる．点 p は z=-R 上にあり，点 a 及び b より等距離にある点である．また，点 a も元々は z=-R 上で点 p から点 a あるいは点 b と等距離にある点 o に存在していたとする．すると，増殖膜と網膜との接着境界線上の点 q も点 p を中心にして回転し，線分 bp 上の点 r に移動すると考えられ

る．ここで，点 q と点 r がバネで結合されていたと仮定すると，この距離が大きくなり弾性限界を超えると，バネは切断されて膜の復元力は失われることになる．従って，線分 qr がある一定の閾値を超えると膜は剥がれて，増殖膜と網膜との接着の境界点は点 q から点 p へ移動する．この様子をＸＹ平面に投影したものが図 7-10(b)である．この場合，線分 pq と直交する直線Ｌに対して折れ曲がる形で増殖膜は変形する．また，増殖膜と網膜との境界部分は点 s 及び t（薄くハッチされた○印）に移動する．また，点 a の移動に伴って増殖膜の外周を構成している点も移動するため，その移動量を計算する．直線Ｌと交差する点より左下にある部分の外周は変化しないから，直線Ｌと交差する点（u 及び v）で移動量が 0 となるように，各点の移動量を比例配分にて計算する．

上記のようにして増殖膜の外周部分及び網膜との接着境界部分の点が求められるので，これらの点を基に他の点の補間計算を行うことにより，増殖膜を構成する全ての点の位置を求めることができる．このように，膜を構成する全ての点に対する計算を行うのではなく，膜剥離という特性を考慮して重要な点を抽出すると共に，重要な点に対してのみ変形計算を行い，他の点に関しては重要点からの補間計算にて移動量を計算することにより，画質を劣化させることなく，高速に画像を生成することができる．結果としての図及び性能評価については，7.6 で述べることにする．

7.5.2　加齢性黄斑変性症の網膜モデル

前節では，黄斑前膜症のリアルタイムなシミュレーションを実現するために，生成する画像の品質を劣化させることなく，剥離する増殖膜の変形動作を高速に計算することのできる方式について述べた．本節では，加齢性黄斑変性症を実現するにあたり，生成する画像の品質を劣化させることなく，高速に網膜と術具との接触判定を行う方式について述べる．

7.2 で記述したように，加齢性黄斑変性症では黄斑部の網膜が盛り上がり，その中に増殖膜が生成する病気であり，網膜下にある増殖膜を取り除くためにサブレチナルカニューレと呼ばれる術具を挿入して生理的食塩水を注入することにより，網膜を持ち上げる．その際，術具と網膜が接触しているかどうかを判定しなければならない．術具と網膜が接触していれば術具を網膜に挿入して生理的食塩水を注入することができるが，術具と網膜が接触していなければ術具から網膜内に生理的食塩水を注入することはできず，網膜を膨張させることはできない．通常，網膜や増殖膜というグラフィックスモデルはポリゴンで生成されているため，この接触判定にはポリゴンとポリゴン，あるいはポリゴンと直線との交点計算を必要とする．このため，ポリゴン数が増加するにつれて接触判定の時間がかかることになり，リアルタイムな接触判定を行うためには，ポリゴン数を削減しなければならない．しかしながら，ポリゴン数を削減すると上記の増殖膜でも説明したように画質が劣化し，手術練習としての充分な画質を持つ映像を生成することはできなくなる．

そこで，グラフィックス描画としては生成されたポリゴンを使用するが，接触判定にはポリゴンを使用せず，盛り上がった網膜を数学関数にて表現する方法を考える．つまり，ポリゴンモデルと数学モデルの両方を備える新しいモデル（**ハイブリッドモデル**）を導入するわけである．例えば，直線と球との接触判定において，球を構成するポリゴンと直線との交点計算を行うよりも，直線と球の方程式を解いた方がはるかに高速である．しかしながら，球をコンピュータグラフィックスを用いて描画するためには，球を構成するポリゴンを必要とする．勿論，光線追跡法を用いる場合，ポリゴンに分割することなく球の方程式をそのまま使用することは可能であるが，光線追跡法ではリアルタイムな描画は困難である．一方，Ｚバッファ法などの手法ではポリゴン単位の描画となり，ポリゴンの分割数に応じた品質を持つ球を描画することができる．

ここで問題となるのは，如何にして盛り上がった網膜を数学関数で表現するのかという点である．幾つかの手術ビデオを観察した結果，加齢性黄斑変性症の網膜の盛り上がり方には特徴があることが分かった．網膜は小さな山，あるいは丘のように少し盛り上がっており，サブレチナルカニューレの挿入により，その山，あるいは丘は相似的に膨張するという点である．従って，網膜の盛り上がり部にcos関数を割当てて，網膜全体を表現する方法を考案した．図 7-11 のように盛り上がった網膜は次のように記述することができる．

$$F(x,y,z)=\begin{cases}(R\sin\theta\cos\phi, R\sin\theta\sin\phi, -R\cos\theta) & if\ R\sin\theta > r\\ (R\sin\theta\cos\phi, R\sin\theta\sin\phi, -R\cos\theta + h(\cos\eta + 1)) & if\ R\sin\theta \le r\end{cases}$$
$$where,\ 0 \le \theta \le \pi/2,\ 0 \le \phi \le 2\pi,\ \eta = (R\sin\theta / r)\pi,\ 0 \le \eta \le \pi\ (for\ R\sin\theta \le r) \quad .....(7\text{-}1)$$

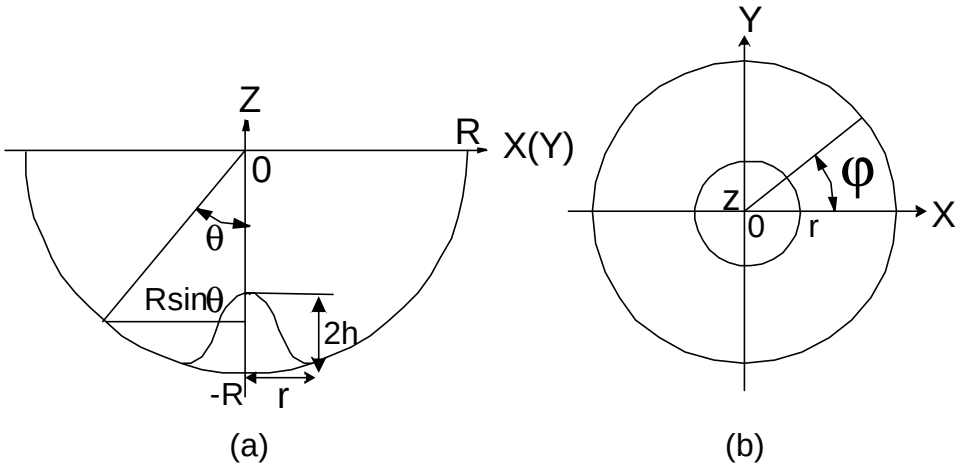


図 7-11 網膜モデル

ここで，網膜の盛り上がりが黄斑部で生じていることを考慮すると，θ をほぼ0とみなすことができるので，式(7-1)において $R\sin\theta \le r$ の場合，

$$z = -R\cos\theta + h(\cos\eta + 1) \cong -R + h(\cos\eta + 1) \quad .....(7-2)$$

となる．従って，

$$\eta = \cos^{-1}((R+z)/h - 1) \quad .....(7-3)$$

と求めることができる．また，式(7-1)より，$\eta = (R\sin\theta / r)\pi$ であるから，

$$(R\sin\theta / r)\pi = \eta = \cos^{-1}((R+z)/h - 1) \quad .....(7-4)$$

となる．結果として，

$$R\sin q = (r/p)\cos^{-1}((R+z)/h-1) \quad \text{.....(7-5)}$$

と計算することができ，術具先端の座標値を(x，y，z)とすると，術具先端が盛り上がった網膜の中にあるのか外にあるのかという接触判定のアルゴリズムは次のようになる．

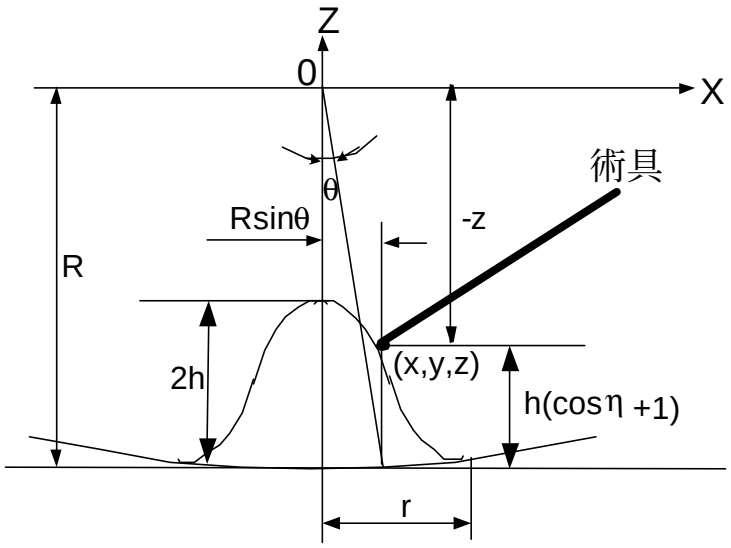


図 7-12 術具と網膜の接触判定

＜術具と網膜の接触判定アルゴリズム＞

1）術具先端のZ座標値に対して，$-R \le z$かつ，$z \le -R+2h$が成り立っていなければ，
術具先端は網膜外にある．成り立っていれば２）へ進む．

2）式(7-5)に従い，$R\sin q = (r/p)\cos^{-1}((R+z)/h-1)$を計算する．また，術具先端の座標値に対して，Z軸からの距離$\sqrt{(x^2+y^2)}$を計算する．

3）$R\sin q \ge \sqrt{(x^2+y^2)}$が成り立っていれば，術具先端は網膜内に存在する．成り立っていなければ術具先端は網膜外にある．

7.5.3 模擬画像のリアルタイム生成

前節及び前々節では，手術映像生成に時間の要する処理を取り上げ，生成する画像の品質を劣化させることなく，高速に画像を生成する方式について検討してきた．本節ではさらに，手術映像生成に必須となる４項目，つまり，1）硝子体吸引画像，2）光照射と付影処理，3）焦点調節，及び 4）出血表現について，高速な画像生成方法について検討する．４項目はいずれも，正確な模擬をしていてはとてもリアルタイムな画像生成をすることができない項目であり，現実感のある画像を簡易に生成する方法について述べる．

１） 硝子体吸引画像

術具として硝子体カッターを持ち，手術装置用フットスイッチを押下すると，硝子体切除吸引が開始される．硝子体は複数の繊維から構成されているゲル状組織であるから，正確な模擬を行うためには，一本一本の繊維の吸引動作を正確に模擬する必要がある．しかしながら繊維の本数は非常に多く，一本一本の挙動を正確に模擬することは現実的ではない．また，繊維の塊を考えて，束となった繊維の吸引動作を模擬しても，実際の手術で行われている硝子体吸引動作に近い映像を生成することはできない．そこで，硝子体カッターの先端にある吸引口に仮想的な面を考え，この面に硝子体吸引画像をテクスチャマッピングにより貼り付ける方法を採用することにした．

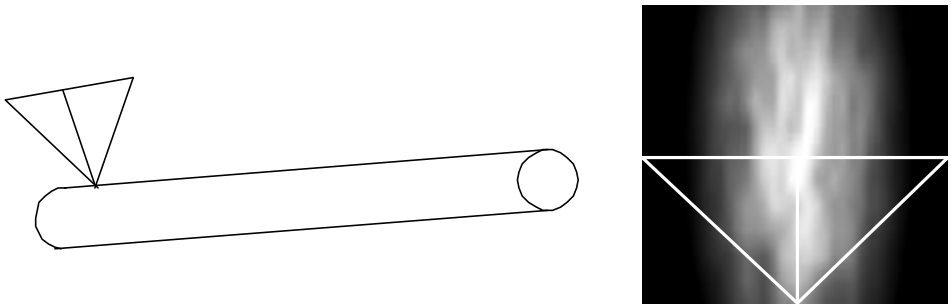


図 7-13 硝子体吸引画像の生成

図 7-13(a)に示すように，硝子体カッターの先端に仮想的な三角形を２枚用意し，この三角形に図 7-13(b)に示す硝子体吸引画像を貼り付ける．形状を三角形とすることにより，眼内に広く存在する硝子体が硝子体カッターの小さな口から吸い込まれている様子を模擬することができる．また，硝子体が吸引されている様子を示すために，貼り付ける画像の位置を時間の経過と共に移動させる．図 7-13(b)に示す三角形が図 7-13(a)の三角形に貼り付けられるが，この貼り付けられる部分が時間の経過と共に上へ移動する．そして，貼り付ける範囲が上端に達すると再び初期状態に戻ることにより，硝子体吸引が延々と続けられることになる．但し，硝子体吸引が進行するに従って，眼内に残る硝子体の量は少なくなるため，時間の経過と共に貼り付ける画像の透明度を上げている．こうすることにより，最初は濃い硝子体吸引が行われるが，硝子体吸引の進行と共に眼内に残る硝子体の量は減少し，吸引される硝子体も次第に薄くなっていく様子を模擬することができる．本方式はほとんど全てのグラフィックスアクセラレータで採用されている方式であるため，他の時間を要する処理に影響を及ぼすことなく，現実感のある硝子体切除吸引画像を高速に生成することができる．

２）光照射と付影処理

眼科手術は片手にライトガイドを持ち，他の手には硝子体カッター，ピーラー，鑷子などの術具を持って行う手術である．従って，ライトガイドを当てることにより照射される領域の模擬と，ライトガイドの照射による他方の術具の眼底上に生成される影の表現が必要となる．PC をベースとした手術シミュレータでは，コンピュータグラフィックスによる画像生成用ライブラリとして OpenGL のみを使用しているが，OpenGL ではスポットライトを表現する関数が用意されているため，照射領域や

光の減衰に関するパラメータを調整することにより，ライトガイドの光照射を模擬することができる．しかしながら影を生成する関数はなく，影の生成には多大な計算時間を必要とする．

影とは，光源から放たれた光が届かない部分にできるものである．従って，光源を視点としてグラフィックス処理により生成した画像には映っていないが，人間の目あるいはカメラの位置を視点として生成した画像上には映っているものの輝度を落とすことにより，付影処理を行うことができる．しかしながら，この方法では常に，光源と人間の目という二つの視点を考え，しかも一方には映っているが他方には映っていないものを探し出す必要があり，通常の倍以上の描画処理時間が必要となる．

そこで，リアルタイムに影を生成するために，影ポリゴンを作成することにした．図 7-14 に示すように，ライトガイドから照射された光により，術具は眼底に影を落とす．従って，ライトガイドの先端である光源の位置と術具を構成する各ポリゴンの頂点位置を結ぶ直線を考え，この直線が眼底と交わる点を求めると，これらの点が影ポリゴンを構成する頂点となる．眼底上に生成されたこれらの点列を結ぶことにより，影を描画することができる．但し，図 7-14 にも示すように，ライトガイドの光源位置と術具を結ぶ線を考えると，術具全体に対して影ポリゴンが生成されるが，スポットライト光源の照射領域の調整，及び光の減衰係数の変更などにより，濃い影と薄い影を生成することができる．

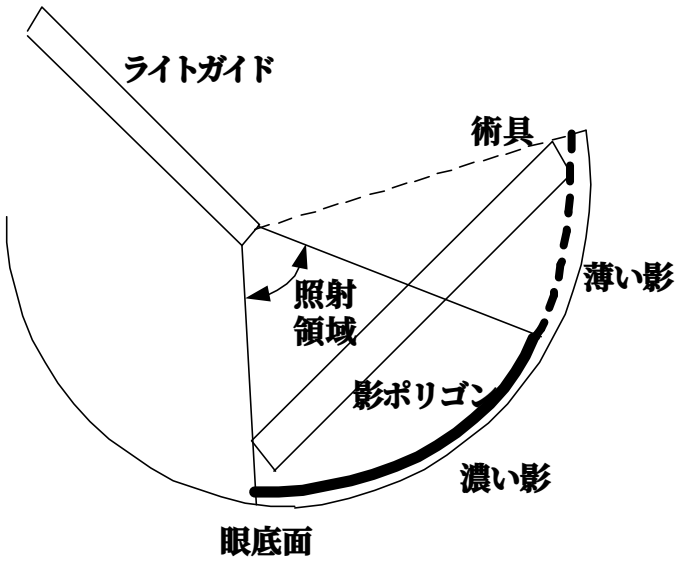


図 7-14 影ポリゴンによる影生成方法

３）焦点調節

眼科用の手術顕微鏡は約 50 倍という高倍率なズームが行えるレンズを使用しており，また被写体深度も深く，非常に鮮明な３次元立体映像を得ることができる．その反面，焦点調整を怠ると，画像がボケて鮮明な映像を得ることができない．術具を交換する際には焦点を角膜上に合わせ，術具を眼内に挿入して黄斑部にある増殖膜を剥離する際には焦点を眼底上に合わせるべきである．従って，術具を交換する度に常に焦点を合わせることが習慣となるような訓練を実施しなければならない．

ボケを生成するためには，通常平滑化フィルタを用いる．あるいは対象となる画像をフーリエ変換により周波数領域に変換した後，低周波成分のみを抽出して逆フーリエ変換することにより元の画

像に戻せば，ボケた画像を生成することも可能である．しかしながら，いずれの方法を採用したとしても画像を構成する画素単位の処理が必要となり，リアルタイムなボケの生成は困難である．

一方，眼科の手術において，焦点調整の練習は実物の顕微鏡を使用して行うこともできる．従って，生成された映像のボケの品質はさほど重要ではなく，上記にも記したように，眼科の手術練習を行う際には常に焦点調整を行う習慣を身に付けさせることが重要である．つまり，焦点がずれていれば鮮明ではない手術映像を呈示し，焦点がずれていることをユーザに認識させることができればよいわけである．そこで，手術映像と視点との間に一枚の仮想的な面を用意し，焦点のずれ具合に応じて仮想面の色を変更する．例えば，焦点が合っているときには仮想面は100%透明となり，鮮明な手術映像を提供するが，焦点がずれてくるに従って仮想面の透明度は減少し，逆に不透明度が増加するので次第に手術映像が見えにくくなる．こうすることにより，映像のボケを完全に模擬しなくとも，手術練習中に常に焦点調整を行うという訓練の目的を達成することができる．

4）出血表現

手術中に出血はつきものであり，メスで切開した際には必ずと言ってよいほど出血を引き起こす．しかしながら，最初から予想されている出血だけではなく，予期せぬところから出血することもある．出血は合併症の一つであり，手術中もできるだけ少ない出血を心がけるべきである．眼科の手術においては，眼底に**アーケード**と呼ばれる太い血管がある．この血管に触れて出血を引き起こすことはまず避けなければならないことであるが，このアーケード以外にも細い血管が数多くあり，また，網膜の下にある血管は観察することができない．従って，増殖膜剥離中に誤って血管のある部分に触れて，出血を引き起こす可能性もある．

本来の出血表現は，眼底に存在する血管を全てグラフィックスのモデルとして登録し，この登録された血管にふれた場合にのみ，出血を引き起こすようにするべきである．また，血管の太さも一様ではないことから，太い血管と細い血管とで出血の量に違いが現れ，血管から出血する様子も流体力学を基にした血液の流れモデルを解析して模擬すべきである．しかしながら，これらの事柄を全て行っていたのでは，術具と血管との干渉計算に多大な時間を要するだけでなく，流体力学を基にした出血画像の生成には膨大な計算及び画像生成時間を要する．従って，本シミュレータでは高速に，しかも術具と網膜との接触状況に応じて出血状態が変化するように，次の仮定を設けて出血モデルを構築した．

＜出血モデル＞

1) 網膜には弾力があり，術具が網膜に触れただけでは出血しないことから，術具が眼底下一定以上の深さにまで達すると出血を引き起こす．
2) 出血量は眼底下に入り込んだ術具の深さに依存し，術具の侵入度が深いほど出血量は多い．また，出血は円状に発生するとし，出血量が多いほど出血の円は大きくなる．
3) アーケードやその支線以外に，網膜下にも多数の血管があり，手術観察下では認識不可能なものもあるため，出血はあらゆる場所から起る．

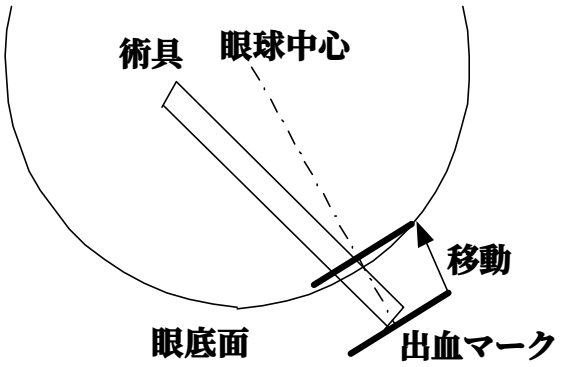


図 7-15 出血モデル

図 7-15 に出血のモデルを示す．術具が眼底下一定以上まで侵入した場合に出血が起るため，術具先端に円状の出血マークを付けたのでは，出血マークが眼底より下にあるために，顕微鏡から観察することはできない．従って，図で示すように，術具侵入の深さに依存した大きさの出血マークを生成し，このマークを眼底面上まで移動させる．また，出血は時間とともに出血量を増していくため，出血マークの透明度を時間と共に下げている．こうすることにより，最初は薄く出ていた出血が時間と共に出血量が増大し，次第に濃くなっていく様子を模擬することができる．

7.6 画像の品質と性能評価

本章では，手術シミュレータというバーチャルリアリティ技術を応用した医療システムを対象とし，コンピュータグラフィックスを用いて手術練習という目的を満足することのできる高品質な画像のリアルタイム生成について検討を進めてきた．本節では，その結果としての画像と性能について評価を行う．

図 7-16　PC ベース手術シミュレータの構成

図 7-16 に PC ベース手術シミュレータの構成を示す．左下に見える計算機が画像生成を行うメイン PC であり，DELL 社製 Precision 610 （Pentium Ⅲ 550MHz 2CPU 512MB Main Memory Intense3D 社製 WildCat4000 搭載）を使用している．その横にあるのが，手術顕微鏡及び手術装置を制御するフットスイッチであり，左が手術顕微鏡，右が手術装置用スイッチである．右下にある計算機は模擬術具及びフットスイッチからの情報を入力し，LAN でメイン PC に情報を伝達する周辺制御 PC であり，mtt 社製 heron シリーズ PC（AMD-K6 MMX 233MHz 1CPU 64MB Main Memory）を使用している．

机の上右側に乗っているのが，上から周辺制御 PC のモニタ，メイン PC からの映像を分配する分配器，映像を切り出す Scan Converter（デジタルアーツ社製 DSC06j 入力対応解像度 640×400～1600×1200　出力対応解像度 640×480～1280×1024），双眼鏡型手術顕微鏡（n.Vision 社製 Virtual Binoculars）の制御ボックスである．その隣にあるのが，メイン PC により生成された映像を表示する CRT であり，CRT の画面上で，下の段に表示されている映像が Scan Converter により切り出されて左隣にある Virtual Binoculars に送られる．Virtual Binoculars の下にあるのが模擬術具であり，入力６自由度を持つが，ハプティックデバイスのように出力の制御は行わず，ブッシュにより術具と眼球との摩擦力を模擬している．この装置も必要な機能のみを抽出して自作することにより，術具間距離を接近させると共に，低価格化に寄与している．

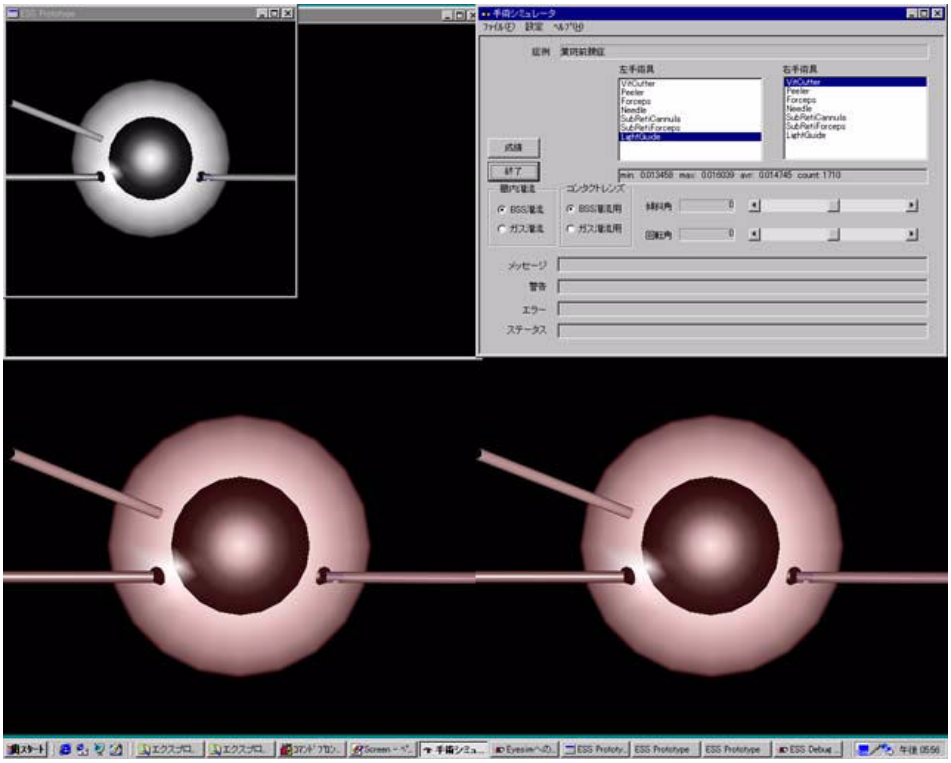


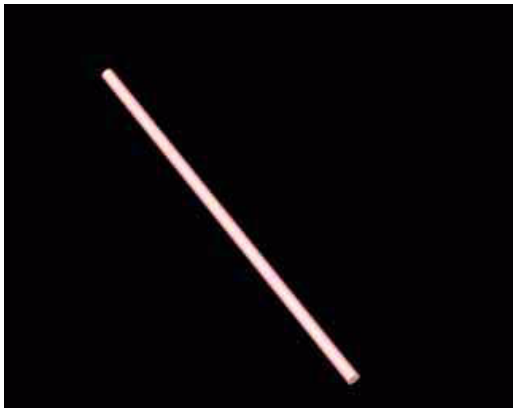
図 7-17　PC ベース手術シミュレータの初期画面

図 7-17 に，PC をベースとした手術シミュレータの初期画面を示す．眼球は黒目（角膜）を示す半球と白目（強膜）を示す球の２つの球から構成されている．眼球左斜め上には生理的食塩水を注入する灌流カニューレが既に装着されており，また，左右の切開創が設けられて，左右の術具が挿入されている．左に挿入されている術具はライトガイドであり，左の術具から少し光が漏れている様子が理解できる．下側に描画されている二つの画像は立体視用画像であり，これらの画像は Scan Converter を介して切り出された後，双眼鏡型手術顕微鏡（Virtual Binoculars）に入力されて立体的な眼球を観察することができる．左下にある画像が左目に，右下にある画像が右目に入力される．

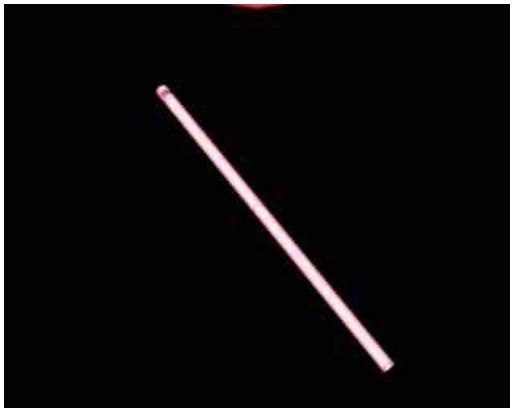
左上にある画像は助手用の画像であり，術者と同じく上から見た手術の様子を示しているが，図 7-3 に示すように，別の角度から見ることも可能である．右上にある画像はユーザインタフェース用画面であり，左右の術具がメニューとして表示されている．挿入されている術具を抜いてから，メニュー上にある術具をクリックすれば，術具の交換を行うことができる．但し，右利きの手術者は必ず左手にライトガイドを持たなければならない．また，角膜上に載せているコンタクトレンズの交換や回転などを行い，任意の方向を見ることも可能となっている．メニュー画面下の方には，手術進行状況や警告，さらにエラーや手術後の成績なども表示できるようになっている．

図 7-18 に手術シミュレータで使用した術具の例を示す．ライトガイドは単純な円柱であり，23 ポリゴンから構成されているが，硝子体カッターには硝子体を切除吸引する口があり，このために 327 ポリゴンから構成されている．また，サブレチナルカニューレやサブレチナル鑷子などは先が曲がっているのが特徴であり，先が曲がっていることにより，網膜に平行に術具を動かし，盛り上がった網膜内にある増殖膜を摘んで剥離することができる．鑷子とよばれるものはピンセット状になってお

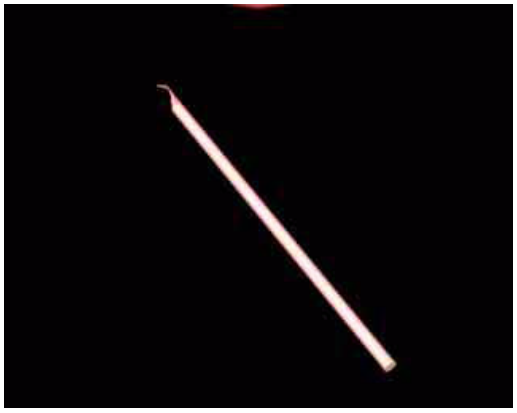
り，術具の手元にあるスイッチを押すと先端が閉じる仕様となっている．また，これらの術具は全て実際の手術で使用する術具とほぼ同じである．



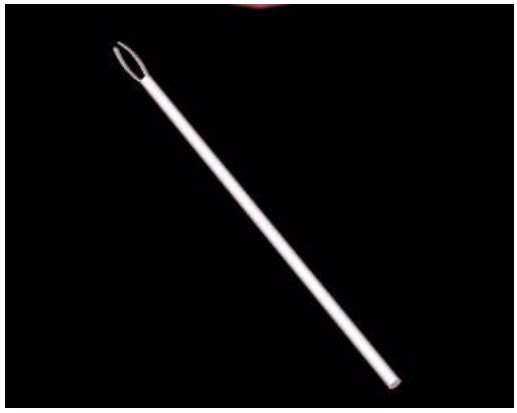
ライトガイド（23ポリゴン）



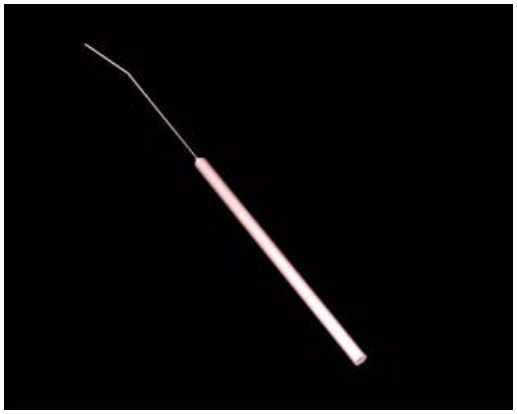
硝子体カッター（327ポリゴン）



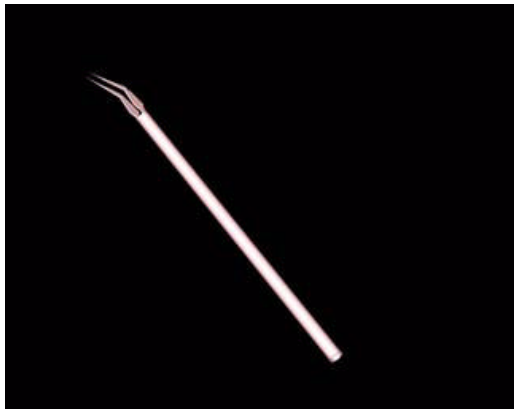
ピーラー（226ポリゴン）



鑷子（132ポリゴン）



サブレチナルカニューレ（101ポリゴン）



サブレチナル鑷子（110ポリゴン）

図 7-18 手術シミュレータで使用する術具例

図 7-19 にモデルの違いによる増殖膜の比較を示す．図 7-19(a)は質点バネモデルで作成した増殖膜であり，30Hz というリアルタイム応答性能を出すために，103 ポリゴンから構成されている．一方，

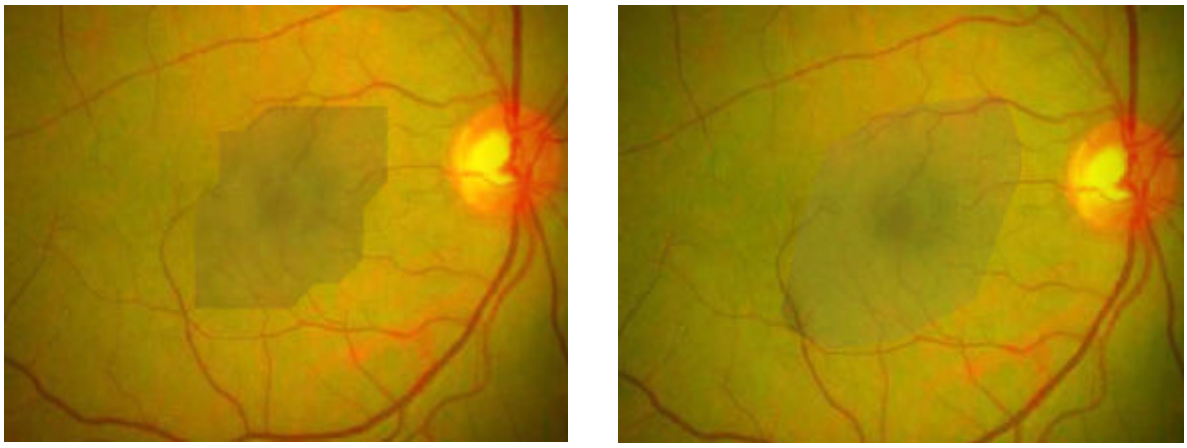
図 7-19(b)は本論文で提案した黄斑前膜の高速膜剥離モデルで作成したものであり，973 ポリゴンから構成されているが，質点バネモデルと同等の応答性能を出すことができる．図 7-19 の(a)及び(b)を比較すると，質点バネモデルによる膜は構成するポリゴンが少ないために，同じ大きさの増殖膜を構成すると，一つ一つのポリゴンの大きさが大きくなり増殖膜を構成するポリゴンの辺を認識することができるくらい画質が劣化しているのが分かる．一方，本論文で提案した増殖膜モデルを構成するポリゴンの数は質点バネモデルの約 10 倍であり，ポリゴンが細かくなっているために，ポリゴンを構成する辺を認識することは困難であり，増殖膜の周辺は滑らかな曲線で構成されている．これは，増殖膜を構成する全てのポリゴンに対して，その頂点の変形を計算するのではなく，重要と考えられる増殖膜の外周及び網膜と癒着している境界部分の点についてのみ，その移動量を求め，他の点については重要と考えられる点の計算結果より，その補間計算で求める．このことが，増殖膜の画質を保ったまま，高速な画像生成を行える理由である．

図 7-20 に黄斑前膜症の手術映像を示す．図 7-20(a)は左からライトガイドで光を照射しながら，右手で硝子体カッターを用いて眼内にある硝子体の切除吸引を行っている様子であり，手術装置に接続されているフットスイッチを押下すると硝子体切除吸引が開始され，スイッチを離すと硝子体切除吸引は中断される．左からライトガイドで光を照射することにより，右手に持った術具の影が眼底網膜上に生成されている様子が理解できる．また，時間と共に眼内に存在する硝子体の量は減少し，代わりに灌流カニューレより生理的食塩水が流入し，切除吸引する硝子体の色は徐々に薄くなる．これは，硝子体カッターの先に仮想的に設けられた三角形に貼り付ける硝子体吸引画像の透明度を増すことにより，吸引される硝子体の色を時間の経過と共に薄くすることで対応している．

一方，図 7-20(b)に示すのは，左からライトガイドで光を照射しながら，右手でピーラーを用いて増殖膜の剥離をしている様子を示すものである．増殖膜の変形は増殖膜の周辺部と，増殖膜と網膜との接着境界部の計算のみを行っているが，現実感のある剥離の様子が伺える．また，右端に大きくみえる明るい円状の物体は視神経乳頭と呼ばれる神経の集まりであるが，そこからアーケードと呼ばれる太い血管が延びているのが分かる．視神経乳頭近くとライトガイド近くには出血の痕があり，時間の経過と共に出血の色は濃くなっている．これは，出血マークに貼り付ける画像の不透明度を時間の経過と共に増加させることにより対応している．

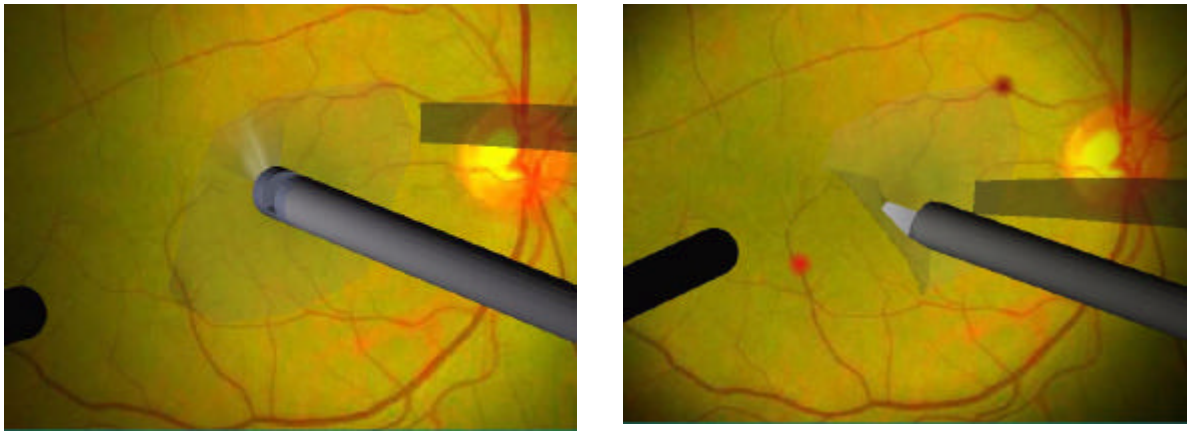
図 7-21 には，加齢性黄斑変性症の手術映像を示す．中心にある丸いものが盛り上がった網膜であり，その下に増殖膜が発生している．図 7-21(a)はサブレチナルカニューレを盛り上がった網膜に差し込み，生理的食塩水を注入して網膜を膨張させている様子を示す映像である．一方，図 7-21(b)は膨張した網膜にサブレチナル鑷子を挿入し，サブレチナルカニューレで開けた穴から網膜下にある増殖膜を取り出している様子を示す映像である．加齢性黄斑変性症で発生する増殖膜は黄斑前膜症で発生する増殖膜と異なり，増殖膜の大きさも小さく，網膜から取り出すと小さく固まる性質を持つ．網膜下から取り出された増殖膜はそのまま眼外へ排出される．

図7-21(a)及び(b)いずれの場合も30Hzというリアルタイムな応答性能を達成している．これは，加齢性黄斑変性症の網膜モデルを，ポリゴンと数学関数という二つのモデルから構成するハイブリッドモデルを採用しているためで，術具と網膜との干渉計算には数学関数のモデルを用いることにより高速な計算を行うと共に，コンピュータグラフィックスでの描画に際しては，ポリゴンを用いることにより高速なグラフィックスアクセラレータの性能を充分に引き出している．その結果として，盛り上がった網膜の形状を粗くすることなく，リアルタイムな画像生成が行えるわけである．なお，盛り上がった網膜は 1560 ポリゴンから構成されており，手術映像の画質としては充分である．



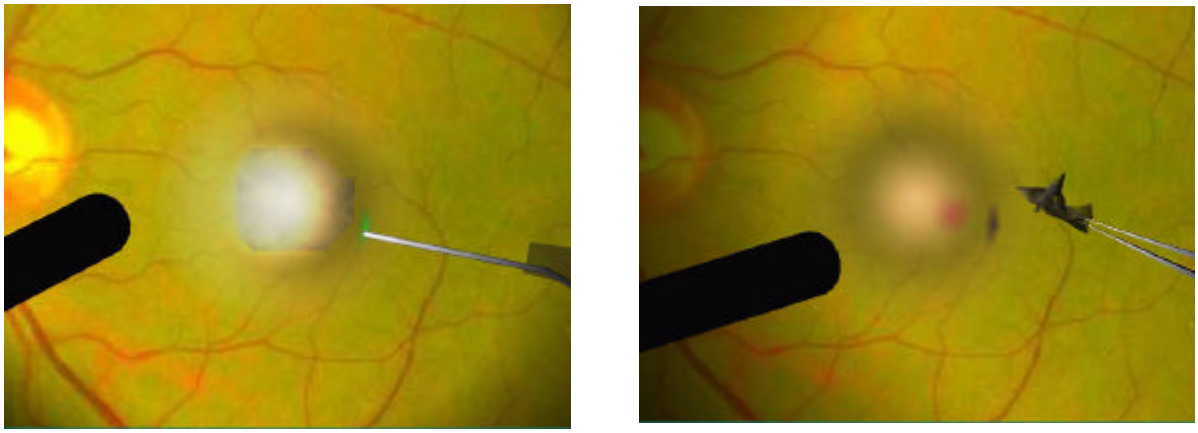
(a) (b)

図 7-19 モデルの違いによる増殖膜の比較



(a) (b)

図 7-20 黄斑前膜症の手術映像



(a) (b)

図 7-21 加齢性黄斑変性症の手術映像

第八章　結論

本論文では，近年益々身近な存在となり，今後さらに発展を遂げていくであろうコンピュータグラフィックスを取上げ，コンピュータグラフィックスを用いた画像生成における基礎的な知識を土台として，コンピュータグラフィックスの基本要素に関する高速化手法について検討した．その後，コンピュータグラフィックスから発展したバーチャルリアリティ技術を利用して，その応用分野の一つである医療システムを考え，実際に手術シミュレータを構築することにより，高品質な画像をリアルタイムに生成する方策について検討を重ねてきた．

まず第二章では，先人が切り開いてきたコンピュータグラフィックスに関する基礎的な課題とその解決方法について論じ，コンピュータグラフィックスを用いた画像生成及び高速描画についての土台を築いた．コンピュータグラフィックスを用いて図形を描画するためには，モデリング座標系上で個々の表示物体を定義した後，ワールド座標系上において全ての表示物体の位置関係を定め，さらにワールド座標系上に存在する全ての表示物体を描画するのではなく，一部の領域を切り出して高速に描画するためのクリップ処理を行う．この結果として得られた表示対象物体のみを描画することにより，ワールド座標系上に定められている表示物体が多数存在しても高速に描画することができる．また，コンピュータグラフィックスを用いて３次元的な映像を生成するためには，視点から見えている面と隠れている面を区別し，如何にして隠れた面を消去するのかという隠面消去問題が重要である．これには様々な手法が存在するが，常に高品質画像と高速描画という相反する二面性を持っている．特に，光線追跡法を用いて画像の描画を行えば，隠面消去問題だけでなく，２次反射や影処理あるいは半透明表示などの問題も同時に解決することができる．しかしながら，光線追跡法による描画には多大な計算時間を要するため，現状ではリアルタイムな描画は困難であるとされている．一方，近年の大量生産によるハードウェアコストの低下に伴い，大量のメモリを使用したＺバッファ法がコンピュータグラフィックスの描画手法では主流となってきている．Ｚバッファ法を用いれば，非常に高速に隠面消去問題を解決することができるため，通常の PC に搭載されている多くのグラフィックスアクセラレータはＺバッファ法を採用している．そこで，通常の PC に搭載されているグラフィックスアクセラレータを用いて高速に，しかも高品質な画像を生成する手法について検討することにした．

第三章では，コンピュータグラフィックスを用いた描画において，最も基本となる直線について検討した．線幅を持たない真っ直ぐなアナログ直線をディジタル化した際に得られる直線（ディジタル直線）の性質を調べるために，まず直線の量子化規則を定め，この量子化規則に従って量子化されたディジタル直線の性質をチェイン符号を用いて調べた．すると，同じチェイン符号が連続して現れることが分かり，同じチェイン符号が連続して現れる部分の直線を構成する画素の個数を調べると，その個数は数式で表現できるというディジタル直線の性質を示した．そして，このディジタル直線の性質を利用して，日常生活で用いられる物体の境界部分を直線で近似する方法を検討した結果，ディジタル直線を用いた直線近似方法は物体の境界を追跡しながら局所的に直線近似も行える方法であるため，従来必要とされていた物体の境界を抽出するという前処理が不要となり，高速な直線近似が行えることを証明した．さらに，この高速な直線近似方法を用いて移動する物体の輪郭を調べることにより，その物体の形状を認識し，一方の物体を他方の物体が追跡するという実験を行い，ディジタル直線を用いた直線近似方法が物体形状の高速な認識にも使用可能であることを示した．

第四章では，このディジタル直線の性質を用いて高速に直線を描画する方法について検討した．従来の直線描画手法は DDA に代表されるとおり，ディジタル直線を構成する画素の位置を一点一点判断

する方式であった．これに対して，ディジタル直線を利用した描画手法では同じチェイン符号を持つ画素の集まりの個数を調べることにより，これらの画素の位置を一点一点判断することなく一度に描画することができる．従って，従来の DDA 法に比べて高速な直線描画を行うことができる．また，コンピュータグラフィックスにおける描画において，重要なクリップ処理の問題に取り組んだ．ワールド座標系上に非常に多くの表示物体を定義していても，詳細な表示を行う場合には一部の物体のみを抽出して描画するため，クリップ処理は高速描画にとって必須の課題である．従来のクリップ処理手法は，常に両端点を持つ直線を対象としてクリップ処理を扱っていた．しかしながら，コンピュータグラフィックスで非常によく用いられる直線は連続した点列を結ぶポリラインであるため，ポリラインを対象とした高速なクリップ処理を検討する必要がある．しかも，従来の手法では計算時間の対象として四則演算のみを取上げていたが，近年におけるパイプライン処理によるプロセッサの高速化を考慮すると，四則演算に加えて比較及び分岐も計算時間の対象に加える必要がある．そこで，ポリラインを対象とした場合，最初の直線の終点が次の直線の始点となることに着目し，連続した点列の２次元座標空間上における状態がどの様に遷移するのかを考え，この状態の遷移を利用して高速なクリップ判定手法を考案した．結果として，四則演算に比較及び分岐を加えた計算時間に対し，最小の演算回数でクリップ処理を実行できることを証明した．

第五章では，大規模グラフィックスデータの階層化について議論した．コンピュータグラフィックスを用いた画像生成において高品質な画像を生成しようとすると，表示対象物体を構成するポリゴンを細かくする必要が生ずる．ポリゴンを細かくすると当然のことながら表示対象となるポリゴンの数が増え，より多くの描画時間が必要とされる．しかしながら，表示物体を拡大したり，あるいは回転させたりと，対象物体を移動している場合には物体の細部を詳細に表示する必要はなく，表示物体の概略が表現できればよい．一方，対象となる物体が静止すると，物体の細部に至るまで細かな表示を行わないと高品質な画像は得られない．そこで，非常に多くのポリゴンから構成される大規模グラフィックスデータを対象とし，そのデータを階層化する方法を検討した．階層化したデータを別々に保存し，動画生成あるいは静止画生成の場合を使い分け，動画生成の場合は粗いデータを，また静止画生成の場合には細かなデータを用いて画像生成してもよいのであるが，動画から静止画に移った瞬間は粗いデータを用いた描画を行った後に細かなデータを用いた描画を行うという二重描画が必要となる．これでは，折角グラフィックスデータを階層化して高速描画を試みたとしても結果として，最終的な高精細な描画の表示が遅くなってしまう．そこで，グラフィックスデータを階層化して，複数のデータを持つが，粗いデータから細かなデータを高速に復元することのできる方式を考案した．本方式では，元のデータに存在する頂点座標を変更することなく，粗いデータから細かなデータを復元する際には，削除された頂点データを追加するだけで行える．従って，データ復元の際に頂点に対する余分な座標変換が発生することはなく，最初から細かなデータを用いて描画した場合とほぼ同じ時間で，粗いデータから細かなデータへと徐々に変化しながら，最終的には詳細なデータを用いた表示を行うことができる．

第六章では，コンピュータグラフィックスから発展したバーチャルリアリティ技術について調査した．バーチャルリアリティ技術では人間の五感を対象とし，その実現方法を研究しているが，五感のうち約 80%をも占めると言われている視覚が最も重要である．また，近年におけるハプティックデバイスの登場により，今までには実現できなかった感覚である力覚（触覚）が実現できるようになってきたため，視覚及び触覚を取上げ，それらの感覚を実現する各種装置についての調査を行った．さらに，バーチャルリアリティ技術は工学分野だけでなく，医療の領域にも応用されており，医学分野

の応用システムである手術ナビゲーションシステム及び手術シミュレーションシステムについての調査も行った．

第七章では，バーチャルリアリティ技術の医療応用である手術シミュレータを取上げ，その構築方法について検討した．手術シミュレータを構築するためには手術そのものについての医学的な知識が必要である．そこで，構築するシミュレータの対象である眼科手術についての調査を行った．眼科手術で用いられる器械類とその特徴，眼科手術の種類と特徴を調査し，対象となる手術を網膜・硝子体手術に決定した．これは，網膜・硝子体手術で発生する病気には人間特有の病気が多く，動物の眼を用いた手術練習が困難であることから，バーチャルリアリティ技術を利用した手術シミュレータの早期開発が要望されていたからである．さらに，網膜・硝子体手術の中でも練習対象とする症例を黄斑前膜症と加齢性黄斑変性症の二つとし，その手術手順について調べた．手術手順の大部分は動物の眼を用いた練習が可能であることから，動物の眼を用いても手術練習が行えない部分を取り出し，この部分についての手術シミュレータを構築することにした．

眼科手術の練習という目的を達成するためには，実際の手術映像に近い高品質な映像を作成する必要があり，しかも，手術練習を行うことのできる充分な応答速度が必要となる．そのため，開発当初は高性能グラフィックスワークステーションをベースとした手術シミュレータを構築し，手術の練習という目的を達成することのできる映像の品質と応答性能について検証した．しかしながら，高性能グラフィックスワークステーションは映像生成能力の高さに比例して高価な計算機であるため，高性能グラフィックスワークステーションをベースとした手術シミュレータは非常に高価なシステムとなる．そこで，近年急速に普及し，年々性能が向上している PC を用いて低価格な手術シミュレータを構築する方策について検討した．但し，低価格なシステムを構築しても，映像の品質が劣化したり，応答性能が悪くなったりして，手術練習という本来の目的を達成することができない．そこで，映像の品質を劣化させることなく，高速に画像を生成することのできる手法についての検討を行った．黄斑前膜症の増殖膜剥離で採用した高速膜剥離モデルでは，増殖膜を構成するポリゴンの数を削減することなく，増殖膜の変形において重要な点を抽出し，この重要点のみの変形計算を行うと共に，他の点については計算された重要点からの補間により求める．この結果として，増殖膜の品質を劣化させることなく，リアルタイムな増殖膜剥離練習を実現することができた．また，加齢性黄斑変性症で採用したハイブリッドモデルでは，盛り上がった網膜と術具との干渉計算を高速に行うために，盛り上がった網膜をポリゴンで定義するだけでなく，網膜形状を数学関数でも記述できるようなモデルを考案した．この結果，網膜形状を表す数学関数を用いることにより，術具と網膜との干渉計算を高速に行うことができると同時に，網膜を構成するポリゴンを用いることにより高速な描画をも実現することができた．その他，硝子体吸引画像，光照射と付影処理，焦点調整，出血表現などについても，現実感のある画像を高速に生成する方法について検討し，PC ベースの手術シミュレータが手術練習という目的を達成することのできる高品質な画像をリアルタイムに生成することができることを実証した．

本論文で検討してきた内容はいずれもコンピュータグラフィックスで高品質な画像を高速に生成するための基本的な研究であり，今後さらに研究を深める必要がある．コンピュータの処理能力も年々向上してきており，数年前までは実現不可能と思われていた事柄が数年先には実現可能なものになりつつあり，要求される画像の品質と応答性能の問題は永遠のテーマである．従来では計算機の処理能力不足により実現不可能と思われていたリアルタイムな映像生成が実現してくると，今度はさらに高度な品質を持つ画像の生成が要求され，そのためにコンピュータ処理能力のさらなる向上が期待されることになる．その意味では，本研究で行った事柄は画像の品質と応答性能という永遠の

テーマに対する研究の一過程であり，今後とも世の中の技術動向に柔軟に対応しながら，研究活動を続けて行きたいと考えている．

謝辞

本論文をまとめるにあたり，始終懇切なるご指導ご鞭撻を賜りました大阪大学大学院・基礎工学研究科の宮崎文夫教授に心より感謝申し上げます．宮崎先生には大阪大学・基礎工学部及び基礎工学研究科・博士前期課程在学中にも多大なご指導を頂きました．重ねて御礼申し上げます．また，本論文をまとめるにあたり，数多くの貴重なご意見を頂きましたシステム科学分野主任の井口征士教授，並びに機械科学分野主任の田中正夫教授に，心より感謝申し上げます．さらに，学位取得に関しましては，立命館大学の有本卓教授に深く感謝申し上げます．有本先生には，大阪大学・基礎工学部及び基礎工学研究科・博士前期課程在学中には，多大なるご指導ご鞭撻を賜り，有本先生が東京大学に移られてからも機会がある度にお会いし，学位取得という大きな目標に向かって励ましのお言葉を頂きました．おそらく，有本先生の励ましがなかったなら，この論文は存在しなかっただろうと思います．ここに，改めて心より感謝申し上げます．

大阪大学大学院・基礎工学研究科の升谷保博講師，並びに西川敦助手にはゼミナールなどで貴重なご意見を頂き，同林清重技官には大阪大学在学中より，常日頃から心温かいご支援を賜りました．ここに，感謝の意を表します．また，宮崎研の方々からはゼミナールなどで貴重なご意見を頂きました．特に，現在三菱電機㈱勤務の小荒健吾氏からは数多くのご助言を頂きました．ここに御礼申し上げます．また，大阪大学大学院・博士後期課程入学及び学位取得に際し，数多くのご助言を頂きました㈱東芝勤務の吉見卓氏に深く感謝申し上げます．

東京工芸大学の小野文孝教授には，小野先生が三菱電機㈱情報技術総合研究所在職中より，表示インタフェース技術部長として，論文原稿の査読など数多くのご指導ご鞭撻を賜りました．また，小野先生が東京工芸大学に移られてからも，本論文に関して多大なご指導を賜りましたことを心より感謝申し上げます．

大阪大学大学院・基礎工学研究科・博士後期課程入学に際し，数々のご支援を賜りました尾形仁士三菱電機㈱開発本部長（元情報技術総合研究所長），並びに中島邦男専任（元インタフェース部門統括）に深く感謝の意を表します．また，大学入学後も引き続きご支援を賜りました瀬政孝義メディア基盤技術部長（元表示インタフェース技術部長），並びに室井克信チームリーダを始めとするグラフィック応用チームの皆様に感謝の意を表します．

手術シミュレータという医療応用システムを構築するにあたり，数々の医学知識をご教授下さいました旭川医科大学の吉田晃敏教授に心より感謝申し上げます．また，手術シミュレータの構築及び評価にあたり，数多くのご助言を頂きました引地泰一講師に心より感謝申し上げます．

大規模グラフィックスデータの階層化に関しては，米国コーネル大学 Bruce Land 博士より数多くのご指導ご鞭撻を賜りました．また，米国コーネル大学在学中には地理情報システムなどに関し，Chris Pelkie 博士より数多くのご助言を頂きました．ここに，深く感謝致します．

最後に，学位取得を陰ながら支えて頂いた義父黒田昭三氏，妻多恵子，息子祐貴，そして両親に感謝の意を表します．本論文を見ることなく昨年秋に他界した故黒田允子義母に感謝の意を表すると共に，本論文をできれば天国にまで届けたい．

参考文献

[1] W.M. ニューマン, R.F. スプロール著, 大須賀監訳：対話型コンピュータグラフィックス, マグロウヒルブック社 (1984)

[2] 山口：コンピュータディスプレイによる図形処理工学, 日刊工業新聞社 (1981)

[3] 中前, 西田：３次元コンピュータグラフィックス, 昭晃堂 (1986)

[4] J. Neide et al: OpenGL Programming Guide(日本語版), The Official Reference Document for OpenGL Release 1, Addison Wesley (1993)

[5] H. Freeman: Computer Processing of Line Drawing Images, Comput. Surv., Vol.6, No.1, pp.57-97 (1974)

[6] 木村, 矢代, 今井：ディジタル画像における線図形可変ベクトル符号化法, 信学論(D), Vol.J66-D, No1, pp.73-80 (1983)

[7] 木本, 安田：Hilditch 法を用いた細線化図形の方向類別チェイン差分符号化方式, 第４回情報理論とその応用研究会資料, pp.238-247 (1983)

[8] 中島, 安居院：ディジタル輪郭線のデルタ符号化について, 信学論(D), Vol.J64-D, No2, pp.109-115 (1981)

[9] Y. Kurozumi and W. Davis: Polygonal Approximation by the Minimax Method, Comput. Gr. Image Process., Vol.19, pp.248-264 (1982)

[10] U.E. Ramer: An Interactive Procedure for the Polygonal Approximation of Plane Curves, Comput. Gr. Image Process., Vol.1, pp.244-256 (1972)

[11] A. Watt: 3D Computer Graphics, Addison Wesley (1993)

[12] R.F. Sproull and I.E. Stherland: A Clipping Divider, Fall Joint Computer Conference, Vol.33, pp.765-775 (1968)

[13] T.M. Nicholl et al: An Efficient Algorithm for 2-D Line Clipping — Its Development and Analysis, ACM Computer Graphics, Vol.21, No.4, pp.253-262 (1987)

[14] 矢野:マイクロプロセッサの命令セットアーキテクチャ，情報処理，Vol.29，No.12，pp.1420-1427 (1988)

[15] 鹿子木:プロメテウスの火―マイクロプロセッサ(完)，ディジタル化の主役 DSP，bit 誌，Vol.19，No.14，pp.83-89 (1989)

[16] 持田：DSP の現状と動向，情報処理，Vol.30，No.11，pp.1291-1299 (1989)

[17] A. Certain et al: Interactive Multiresolution Surface Viewing, SIGGRAPH 96 Conference Proceedings, pp.91-98 (1996)

[18] H. Hoppe: Progressive Meshes, SIGGRAPH 96 Conference Proceedings, pp.99-108 (1996)

[19] P. Lindstorm et al: Real-Time Continuous Level of Detail Rendering of Height Fields, SIGGRAPH 96 Conference Proceedings, pp.109-118 (1996)

[20] J. Cohen et al: Simplification Envelopes, SIGGRAPH 96 Conference Proceedings, pp.119-128 (1996)

[21] J.R. Vacca: VRML Bringing Virtual Reality to the Internet, AP Professional (1996)

[22] 舘：Virtual Reality and R-Cubed，第 8 回産業用バーチャルリアリティ展 IVR2000，リードエグジビションジャパン，pp.6-35 (2000)

[23] 信太他：人体内を自由に観察できる高速ヘリカル CT，東芝レビュー，Vol. 52，No. 5，pp. 37-40 (1997)

[24] (社)日本放射線機器工業会編集：医用画像・放射線機器ハンドブック，名古屋美術印刷 (1995)

[25] 藤代：ボリュームビジュアリゼーション&グラフィックス ボリューム表現の共通基盤の確立を目指す，NIKKEI COMPUTER GRAPHICS，2000 年 1 月号，pp.68-69 (2000)

[26] HMD 特集 HMD 製品紹介，日本バーチャルリアリティ学会誌，Vol.3，No.2，pp.27-41 (1998)

[27] 柴田他： ２眼式立体映像観察時の視機能に関する研究，3D 映像，Vol.12，No.4，pp.8-12 (1998)

[28] 井上他： HMD における立体画像の再生位置と視覚負担，3D 映像，Vol.12，No.4，pp.13-18 (1998)

[29] 梶木：自然な立体視を目指した立体表示技術の動向, 画像電子学会 第174回研究会講演予稿, pp. 43-48 (1999)

[30] 山下他：イメージスプリッタ方式 2D/3D ディスプレイのインターネットへの応用，３次元画像コンファレンス’97 講演論文集，pp.225-230 (1997)

[31] 能瀬他：リアレンチ方式めがねなし 3D 液晶ディスプレイ，３次元画像コンファレンス’97 講演論文集，pp.219-224 (1997)

[32] 今井他：画像シフト光学系を用いた視点追従型立体プロジェクタ，3D 映像，Vol.12，No.4，pp.45-54 (1998)

[33] 小林：広視域複数者対応メガネなし 3D ディスプレイ　シーフォン 3D ディスプレイのしくみと製品化，画像電子学会 第174回研究会講演予稿，pp.25-30 (1999)

[34] 梶木他：超多眼領域の立体表示における単眼視差の効果，３次元画像コンファレンス’97 講演論文集，pp.166-171 (1997)

[35] 梶木他：超多眼領域の立体視覚，3D 映像，Vol.12，No.4，pp.25-30 (1998)

[36] 内田：CAVE システムの動向と課題, 画像電子学会 第174回研究会講演予稿, pp. 49-53 (1999)

[37] 澤田他：半球面スクリーンを用いた多人数型及びパーソナル型 3D 没入ディスプレイ―都市環境ヒューマンメディアにおける半球面ドーム型 VR 体験システムー，３次元画像コンファレンス 2000 講演論文集，pp.79-82 (2000)

[38] 岩田：シームレス全周球面ディスプレイ, 日本バーチャルリアリティ学会 第３回大会論文集，pp.155-158 (1998)

[39] 橋本，岩田：凸面鏡を用いた全方向球面ディスプレイの光学設計，日本バーチャルリアリティ学会 第３回大会論文集，pp.159-162 (1998)

[40] 広田，坂口：仮想空間のためのハプティックス技術の現状，日本機械学会誌，Vol.102，No.971，pp.40-44 (1999)

[41] T.H. Massie: Trends and Issues in the Application of Haptic Interfaces，第 8 回産業用バーチャルリアリティ展 IVR2000，リードエグジビションジャパン，pp.71-90 (2000)

[42] M. Sato: 4+4 Fingers Manipulating Virtual Objects in Mixed Reality Environment, ISMR 2001，pp.27-34 (2001)

[43] 広瀬他：ワイヤーテンションを用いたウェアラブルフォースディスプレイの開発，日本バーチャルリアリティ学会大会論文集，Vol 3，pp.1-4 (1998)

[44] 浅村他：選択刺激子による触感の呈示，日本バーチャルリアリティ学会大会論文集，Vol.3，pp.15-18 (1998)

[45] 久米他：触滑り感覚ディスプレイの検討，日本バーチャルリアリティ学会大会論文集，Vol.3，pp.19-20 (1998)

[46] 奈良他：弾性波動を用いた皮膚感覚ディスプレイ（第二報），日本バーチャルリアリティ学会大会論文集，Vol.3，pp.21-24 (1998)

[47] 土肥他：特集 CAS(Computer Aided Surgery) コンピュータ外科の現状，医器学，Vol.66，No.12，pp.685-689 (1996)

[48] 斎藤： 医療・福祉用ロボットの制御，日本機械学会誌，Vol.103，No.984 pp.765-769 (2000)

[49] 伊関他：特集 CAS(Computer Aided Surgery) バーチャルリアリティ技術と外科手術，医器学，Vol.66，No.12，pp.698-702 (1996)

[50] 金子他：シリコン樹脂製助軟骨モデルを用いた小耳症耳介形成術の手術シミュレーション，日本コンピュータ支援外科学会誌，Vol.6，No.2，p.11 (1998)

[51] 加藤他：術前画像誘導下脳穿刺マニピュレータの開発，第７回コンピュータ支援画像診断学会大会/第６回日本コンピュータ外科学会大会 合同論文集，pp.73-74 (1997)

[52] 江積他：手術シミュレーションシステムにおける弾性臓器モデルの開発，第７回コンピュータ支援画像診断学会大会/第６回日本コンピュータ外科学会大会 合同論文集，pp.91-92 (1997)

[53] N. Suzuki: Virtual Surgery System using Deformable Organ Models and Force Feedback System with Three Fingers, Lecture Notes in Computer Science, Vol.1496, pp.397-403 (1998)

[54] 鈴木他：触覚を伴った手術作業が可能なバーチャル手術システムの開発，日本バーチャルリアリティ学会論文誌，Vol.3，No.4，pp.237-243 (1998)

[55] P.F. Neumann et al: Virtual Reality Vitrectomy Simulator, Lecture Notes in Computer Science, Vol.1496. pp.910-917 (1998)

[56] 江他：脳外科手術のためのナビゲーションシステムについて，日本コンピュータ支援外科学会会誌，Vol.6，No.2，p.13（1998）

[57] 大鹿他：超音波白内障手術の修得，メディカルビュー（1997）

[58] 佐竹，田中：VOPS(VideoOverlay Parameters System)を用いた白内障手術教育，眼科手術，Vol.11，No.4，pp.441-443（1998）

[59] 永田監修：眼科マイクロサージェリー 第3版，㈱ミクス（1997）

[60] 臼井，坪田編集：カラーアトラス眼科手術/Special Technique，診断と治療社（1996）

[61] 山中他：アトラス　網膜・硝子体手術，メディカル葵出版（1995）

[62] 増田監訳：眼科手術手技アトラス，メディカルサイエンスインターナショナル（1991）

[63] 広田，金子： 柔らかい仮想物体の表現，日本バーチャルリアリティ学会大会論文集，Vol.3，pp.299-302（1998）

関連文献

学術論文誌

[101] 向井，有本：ディジタル直線の性質に基づく高速多角形近似を用いた移動物体の認識，情報処理学会論文誌，Vol.28，No.2，pp.156-167（1987）

[102] 向井：状態遷移を利用した高速ポリライン・クリップ方式，情報処理学会論文誌，Vol.32，No.2，pp.188-196（1991）

[103] 向井，原田，室井，宮本，浦谷，矢野：PCベースリアルタイム手術シミュレータの開発，電子情報通信学会論文誌 D-Ⅱ，Vol.J84-D-Ⅱ，No.6，pp.1213-1221（2001）

国際学会論文集

[104] N. Mukai, M. Harada, K. Muroi, T. Hikichi, and A. Yoshida: New Graphics Models for PC Based Ocular Surgery Simulator, Medicine Meets Virtual Reality 2001, J.D.Westwood et al.(Eds.), IOS Press, pp.329-335 (2001)

[105] T. Hikichi, A. Yoshida, S. Igarashi, N. Mukai, M. Harada, K. Muroi, and T. Terada: Vitreous Surgery Simulator, Archives of Ophthalmology, Vol.118, pp.1679-1681 (2000)

[106] N. Mukai, M. Harada, K. Muroi, T. Hikichi, and A. Yoshida：New Model for Membrane Peeling in Ocular Surgery Simulator, MODSIM 2001 (2001) （採録決定 2001年12月に発表予定）

学術研究会予稿集

[107] 向井，森田，有本：ディジタル直線の性質を利用した境界追跡による高速特徴点抽出アルゴリズム，第29回情報処理学会全国大会，pp.1043-1044（1984）

[108] 田中，向井，志賀： 高速画像回転処理の実現法，第 35 回情報処理学会全国大会，pp.2005-2006（1987）

[109] 向井，亀山，風間：効率的な隠面消去領域抽出方式，第 35 回情報処理学会全国大会，pp.2283-2284（1987）

[110] 飯塚，向井，亀山：ME400のグラフィック・エンジン(1)―ハードウェア方式―，第39回情報処理学会全国大会，pp.1908-1909（1989）

[111] 向井，飯塚，亀山：ME400のグラフィック・エンジン(2)―ファームウェア方式と性能評価―，第39回情報処理学会全国大会，pp.1910-1911（1989）

[112] 向井，亀山：3 次元地形形状の 3 角形パッチ生成方法，第 45 回情報処理学会全国大会，pp.2-393 - 2-394（1992）

[113] 向井，原田，室井，寺田，引地，吉田：硝子体手術シミュレータシステム，第38回日本ME学会大会，p.282（1999）

[114] 向井，原田，室井，寺田：眼科手術シミュレータシステム，三菱電機技報，Vol.73，No.11，pp.34-37（1999）
[115] 原田，向井，室井：手術シミュレータにおける力覚提示方式，第 4 回知能メカトロニクスワークショップ，pp.247-249（1999）
[116] 原田，向井，室井：眼球手術シミュレータにおける膜剥離モデル，日本バーチャルリアリティ学会第５回大会論文集，pp.421-424（2000）
[117] 向井，原田，室井，瀬政：眼科用手術シミュレータの開発，画像電子学会第 182 回研究会，pp.7-13（2000）

付録

付録－１

本節では，第三章で紹介したディジタル直線の性質を導くために必要となる整数に関する基本的な定理について証明する．

＜定理１＞

正の実数 x に対して次式が成立する．[注2)]

$\lceil x \rceil - 1 \le \lfloor x \rfloor \le \lceil x \rceil$

（証明）

$x = q + r$ (q は正の整数，r は $0 \le r < 1$ を満たす小数) とおくと，

$$\lceil x \rceil = \lceil q + r \rceil = \begin{cases} q+1 & (0 < r < 1) \\ q & (r = 0) \end{cases}$$

従って，$$\lceil x \rceil - 1 = \lceil q + r \rceil - 1 = \begin{cases} q & (0 < r < 1) \\ q-1 & (r = 0) \end{cases}$$

一方，$\lfloor x \rfloor = \lfloor q + r \rfloor = q$ であるから，

$\therefore \lceil x \rceil - 1 \le \lfloor x \rfloor \le \lceil x \rceil$ となる．

＜定理２＞

m を正の整数，x を正の実数（但し，$x \le m$）とすると，次式が成立する．

$\lceil m - x \rceil = m - \lfloor x \rfloor$

（証明）

$x = q + r$ (q は正の整数，r は $0 \le r < 1$ を満たす小数) とおくと，

$$\lceil m - x \rceil = \lceil m - (q + r) \rceil = \lceil (m - q) - r \rceil = \begin{cases} (m - q - 1) + 1 & (0 < r < 1) \\ m - q & (r = 0) \end{cases}$$

一方，$\lfloor x \rfloor = \lfloor q + r \rfloor = q$ であるから，

$\therefore \lceil m - x \rceil = m - q = m - \lfloor x \rfloor$

注2) $\lfloor x \rfloor$ は x を超えない最大の整数を，$\lceil x \rceil$ は x 以上の最小の整数を求める関数である

＜定理３＞

正の実数 x に対して，次式が成立する．

$\lceil x \rceil \le 2\lceil x/2 \rceil \le \lceil x \rceil + 1$

（証明）

$x = q + r$ **(q は正の整数，r は $0 \le r < 1$ を満たす小数) とおくと，**

$\lfloor x \rfloor = \lfloor q + r \rfloor = q$

1）$0 < r < 1$ の場合，$\lceil x \rceil = \lceil q + r \rceil = q + 1$ だから，

a）q=2j（j は正の整数）のとき

$$2\lceil x/2 \rceil = 2\lceil (q+r)/2 \rceil = 2\lceil (2j+r)/2 \rceil = 2\lceil j + (r/2) \rceil = 2(j+1)$$
$$= (2j+1) + 1 = (q+1) + 1 = \lceil x \rceil + 1$$

b）q=2j+1（j は正の整数）のとき

$$2\lceil x/2 \rceil = 2\lceil (q+r)/2 \rceil = 2\lceil (2j+1+r)/2 \rceil = 2\lceil j + \{(r+1)/2\} \rceil = 2(j+1)$$
$$= (2j+1) + 1 = q + 1 = \lceil x \rceil$$

$$\therefore \lceil x \rceil \le 2\lceil x/2 \rceil \le \lceil x \rceil + 1$$

2）$r = 0$ の場合，$\lceil x \rceil = \lceil q + r \rceil = q$ だから，

a）q=2j（j は正の整数）のとき

$$2\lceil x/2 \rceil = 2\lceil (q+r)/2 \rceil = 2\lceil (2j)/2 \rceil = 2\lceil j \rceil = 2j = q = \lceil x \rceil$$

b）q=2j+1（j は正の整数）のとき

$$2\lceil x/2 \rceil = 2\lceil (q+r)/2 \rceil = 2\lceil (2j+1)/2 \rceil = 2\lceil j + (1/2) \rceil = 2(j+1)$$
$$= (2j+1) + 1 = q + 1 = \lceil x \rceil + 1$$

$$\therefore \lceil x \rceil \le 2\lceil x/2 \rceil \le \lceil x \rceil + 1$$

従って，1)及び 2)より，$\lceil x \rceil \le 2\lceil x/2 \rceil \le \lceil x \rceil + 1$

＜定理４＞

m を正の整数，x を整数ではない正の実数とすると，次式が成立する．

$\lceil m - x \rceil = m - \lceil x \rceil + 1$

（証明）

$x = q + r$ **(q は正の整数，r は $0 < r < 1$ を満たす小数) とおくと，**

$$\lceil m - x \rceil = \lceil m - (q+r) \rceil = \lceil (m-q) - r \rceil = m - q = m - \lceil x \rceil + 1$$
$$(\because \lceil x \rceil = \lceil q + r \rceil = q + 1)$$
$$\therefore \lceil m - x \rceil = m - \lceil x \rceil + 1$$

＜定理５＞

正の実数 x_1 **及び** x_2 **に対して，次式が成立する．**

$\lceil x_1 - x_2 \rceil - 1 \le \lceil x_1 \rceil - \lceil x_2 \rceil \le \lceil x_1 - x_2 \rceil \le \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

（証明）

$x_1 = q_1 + r_1 (q_1$ **は正の整数，** r_1 **は** $0 \le r_1 < 1$ **を満たす小数）及び**

$x_2 = q_2 + r_2 (q_2$ **は正の整数，** r_2 **は** $0 \le r_2 < 1$ **を満たす小数）とおくと，**

1） $r_1 \ne 0 \wedge r_2 \ne 0$ **の場合**

$$\lceil x_1 - x_2 \rceil = \lceil (q_1 + r_1) - (q_2 + r_2) \rceil = \lceil (q_1 - q_2) + (r_1 - r_2) \rceil = \begin{cases} q_1 - q_2 + 1 & (r_1 - r_2 > 0) \\ q_1 - q_2 & (r_1 - r_2 \le 0) \end{cases}$$

一方， $\lceil x_1 \rceil - \lceil x_2 \rceil = (q_1 + 1) - (q_2 + 1) = q_1 - q_2$ **であるから，**

$\therefore \lceil x_1 - x_2 \rceil - 1 \le \lceil x_1 \rceil - \lceil x_2 \rceil \le \lceil x_1 - x_2 \rceil \le \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

2） $r_1 \ne 0 \wedge r_2 = 0$ **の場合**

$\lceil x_1 - x_2 \rceil = \lceil (q_1 + r_1) - q_2 \rceil = \lceil (q_1 - q_2) + r_1 \rceil = q_1 - q_2 + 1$

一方， $\lceil x_1 \rceil - \lceil x_2 \rceil = (q_1 + 1) - q_2 = q_1 - q_2 + 1$ **であるから，**

$\therefore \lceil x_1 - x_2 \rceil - 1 \le \lceil x_1 \rceil - \lceil x_2 \rceil = \lceil x_1 - x_2 \rceil \le \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

3） $r_1 = 0 \wedge r_2 \ne 0$ **の場合**

$\lceil x_1 - x_2 \rceil = \lceil q_1 - (q_2 + r_2) \rceil = \lceil (q_1 - q_2) - r_2 \rceil = q_1 - q_2$

一方， $\lceil x_1 \rceil - \lceil x_2 \rceil = q_1 - (q_2 + 1) = q_1 - q_2 - 1$ **であるから，**

$\therefore \lceil x_1 - x_2 \rceil - 1 = \lceil x_1 \rceil - \lceil x_2 \rceil \le \lceil x_1 - x_2 \rceil = \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

4） $r_1 = 0 \wedge r_2 = 0$ **の場合**

$\lceil x_1 - x_2 \rceil = \lceil q_1 - q_2 \rceil = q_1 - q_2$

一方， $\lceil x_1 \rceil - \lceil x_2 \rceil = q_1 - q_2$ **であるから，**

$\therefore \lceil x_1 - x_2 \rceil - 1 \le \lceil x_1 \rceil - \lceil x_2 \rceil = \lceil x_1 - x_2 \rceil \le \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

従って，1），2），3)及び 4)より， $\lceil x_1 - x_2 \rceil - 1 \le \lceil x_1 \rceil - \lceil x_2 \rceil \le \lceil x_1 - x_2 \rceil \le \lceil x_1 \rceil - \lceil x_2 \rceil + 1$

＜定理６＞

正の実数 x に対して，次式が成立する．

$\lfloor x \rfloor - 1 \le 2\lfloor x/2 \rfloor \le \lfloor x \rfloor$

（証明）

$x = q + r$（q **は正の整数，** r **は** $0 \le r < 1$ **を満たす小数**）**とおくと，** $\lfloor x \rfloor = \lfloor q + r \rfloor = q$

1）q=2j（j は正の整数）のとき，

$$2\lfloor x/2 \rfloor = 2\lfloor (q+r)/2 \rfloor = 2\lfloor (2j+r)/2 \rfloor = 2\lfloor j + (r/2) \rfloor = 2j = q = \lfloor x \rfloor$$

2）q=2j+1（j は正の整数）のとき，

$$2\lfloor x/2 \rfloor = 2\lfloor (q+r)/2 \rfloor = 2\lfloor (2j+1+r)/2 \rfloor = 2\lfloor j + \{(r+1)/2\} \rfloor = 2j = q-1 = \lfloor x \rfloor - 1$$

$$\therefore \lfloor x \rfloor - 1 \le 2\lfloor x/2 \rfloor \le \lfloor x \rfloor$$

＜定理７＞

正の実数 x_1 及び x_2 に対して，次式が成立する．

$\lfloor x_1 \rfloor - \lfloor x_2 \rfloor - 1 \le \lfloor x_1 - x_2 \rfloor \le \lfloor x_1 \rfloor - \lfloor x_2 \rfloor \le \lfloor x_1 - x_2 \rfloor + 1$

（証明）

$x_1 = q_1 + r_1$（q_1 **は正の整数，** r_1 **は** $0 \le r_1 < 1$ **を満たす小数**）**及び**

$x_2 = q_2 + r_2$（q_2 **は正の整数，** r_2 **は** $0 \le r_2 < 1$ **を満たす小数**）**とおくと，**

$$\lfloor x_1 - x_2 \rfloor = \lfloor (q_1 + r_1) - (q_2 + r_2) \rfloor = \lfloor (q_1 - q_2) + (r_1 - r_2) \rfloor = \begin{cases} q_1 - q_2 & (r_1 - r_2 \ge 0) \\ q_1 - q_2 - 1 & (r_1 - r_2 < 0) \end{cases}$$

一方， $\lfloor x_1 \rfloor - \lfloor x_2 \rfloor = q_1 - q_2$ **であるから，**

$$\therefore \lfloor x_1 \rfloor - \lfloor x_2 \rfloor - 1 \le \lfloor x_1 - x_2 \rfloor \le \lfloor x_1 \rfloor - \lfloor x_2 \rfloor \le \lfloor x_1 - x_2 \rfloor + 1$$

＜定理８＞

m **を正の整数，** x **を正の実数とすると，次式が成立する．**

$m - \lfloor x \rfloor - 1 \le \lfloor m - x \rfloor \le m - \lfloor x \rfloor$

（証明）

$x = q + r$（q **は正の整数，** r **は** $0 \le r < 1$ **を満たす小数**）**とおくと，** $\lfloor x \rfloor = q$

1） $r \ne 0$ **のとき，** $\lfloor m - x \rfloor = \lfloor m - (q+r) \rfloor = \lfloor (m-q) - r \rfloor = m - q - 1 = m - \lfloor x \rfloor - 1$

2） $r = 0$ **のとき，** $\lfloor m - x \rfloor = \lfloor m - q \rfloor = m - q = m - \lfloor x \rfloor$

$$\therefore m - \lfloor x \rfloor - 1 \le \lfloor m - x \rfloor \le m - \lfloor x \rfloor$$

付録－２

本節では，第三章で紹介したディジタル直線の性質について証明する．

Ｉ） $0 < a = 1/h < 1/2$ **（** a **は直線の傾き，** h **は正の整数）の場合**

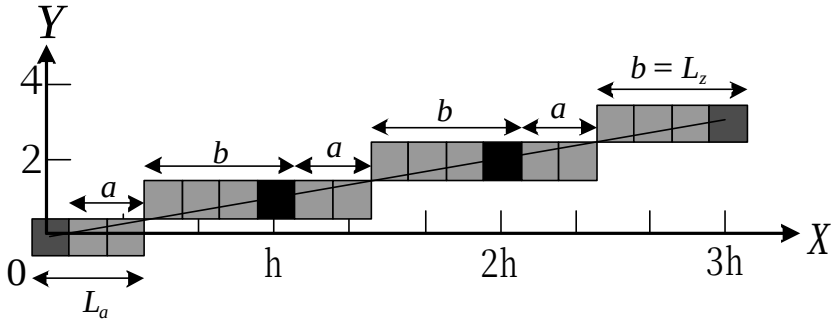


図 A-1 傾き 0<1/h<1/2 のディジタル直線

この場合の直線は図 A-1 に示すように，X座標値０からｈまでのディジタル直線を構成する画素列のパターンを繰り返すことにより，全体のディジタル直線が構成されている．図において黒色画素は中継点であり，この中継点は一つ前のディジタル直線（例えばX座標値が０からｈまでのディジタル直線）の終点であると同時に，次のディジタル直線（X座標値がｈから２ｈまでのディジタル直線）の始点でもある．そこで，最初の画素列の並びである L_a よりも一つ少ない数を $a(=L_a-1)$，終点に達する画素列の数（つまり L_z）を b とすると，$a+b=h$ となる．ここで，最初にＹ座標値が１となる画素のＸ座標値 x は次式を満たす．

$$(1/h)x \geq 1/2 \quad \cdots\cdots (A-1)$$

ゆえに，$x=\lceil h/2 \rceil$ であり，L_a を構成する画素列のＸ座標値は０から $\lceil h/2 \rceil -1$ までであるから，$L_a=\lceil h/2 \rceil$ となる．最初のディジタル直線を構成する画素列のＸ座標値は０からｈまでであり，その個数はｈ＋１であるから，$b=(h+1)-\lceil h/2 \rceil$ となり，これ値が L_z となる．

また，始点あるいは終点を含まない途中状態におけるディジタル直線は，常に中継点を含むので，$L_s=a+b=h$ となる．つまり，

$$L_a=\lceil h/2 \rceil, L_s=h, L_z=(h+1)-\lceil h/2 \rceil \quad \cdots\cdots (A-2)$$

が成立する．次に，これらの関係について調べる．前節の定理３より，

正の実数 x に対して，$\lceil x \rceil \leq 2\lceil x/2 \rceil \leq \lceil x \rceil +1$ が成立するので，正の整数ｈに対しては，

$h=\lceil h\rceil \le 2\lceil h/2\rceil \le \lceil h\rceil+1=h+1$ が成立し，$2\lceil h/2\rceil-1\le h\le 2\lceil h/2\rceil$ となる．ここで，(A-2)を代入すると，$2L_a-1\le L_s\le 2L_a$ .....$(A-3)$

が成立する．ここで，hは正の整数であるから，hを偶数$(h=2j,\ j$は正の整数$)$と奇数$(h=2j+1)$に分けて考える．

a) h=2j のとき，$\lceil h/2\rceil=\lceil (2j)/2\rceil=j=h/2$ であるから，

$$L_z=(h+1)-\lceil h/2\rceil=(h+1)-(h/2)=(h/2)+1=\lceil h/2\rceil+1=L_a+1$$

b) h=2j+1のとき，$\lceil h/2\rceil=\lceil (2j+1)/2\rceil=\lceil j+(1/2)\rceil=j+1=(2j+1+1)/2=(h+1)/2$ であるから

$$L_z=(h+1)-\lceil h/2\rceil=(h+1)-(h+1)/2=(h+1)/2=\lceil h/2\rceil=L_a$$

$$\therefore L_a\le L_z\le L_a+1 \quad .....(A-4)$$

よって，式(A-2)，(A-3)及び(A-4)より，

$$L_a=\lceil h/2\rceil, L_s=h, L_z=(h+1)-\lceil h/2\rceil$$
$$2L_a-1\le L_s\le 2L_a, L_a\le L_z\le L_a+1 \quad .....(A-5)$$

Ⅱ） $0<a=n/m<1/2$ **（**a **は直線の傾き，** n,m **は互いに素な正の整数）の場合**

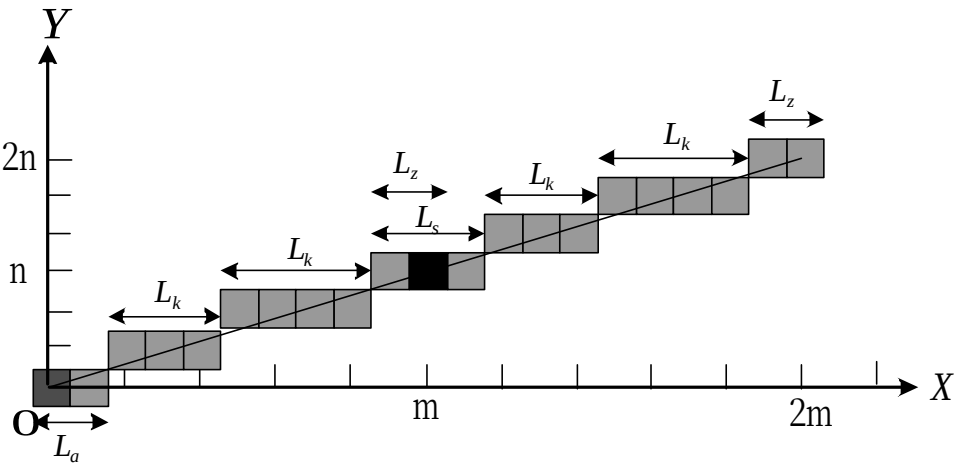


図 A-2 **傾き** 0<n/m<1/2 **のディジタル直線**

Ⅰ）の場合と同様に，最初にY座標値が1となる画素のX座標値 x_1 **は次式を満たす．**

$$(n/m)x_1\ge 1/2 \quad .....(A-6)$$

よって，$x_1 = \lceil m/(2n) \rceil$であり，$L_a$を構成する画素列のX座標値は0から$\lceil m/(2n) \rceil - 1$までであるから，$L_a = \lceil m/(2n) \rceil$.....(A−7)

となる．一般に，Y座標値がkとなる最初のX座標値x_kは，

$(n/m)x_k \geq (k-1) + (1/2)$.....(A−8)

を満足する最小の整数であるから，$x_k = \left\lceil \frac{(2k-1)m}{2n} \right\rceil$.....(A−9)

となる．従って，Y座標値がkとなるディジタル直線を構成する画素列の個数L_kは，次式で求められる．

$$L_k = x_{k+1} - x_k = \left\lceil \frac{\{2(k+1)-1\}m}{2n} \right\rceil - \left\lceil \frac{(2k-1)m}{2n} \right\rceil = \left\lceil \frac{(2k+1)m}{2n} \right\rceil - \left\lceil \frac{(2k-1)m}{2n} \right\rceil \quad (A-10)$$

ここで，全体のディジタル直線はX座標値が0からmまでの範囲のディジタル直線を繰り返すことにより構成されていることを考慮すると，最初の中継点を含む画素列の長さL_sの一部に，終点を含む画素列の長さL_zが含まれている．最初にY座標値がnとなるX座標値x_nは式(A-9)より，$x_n = \left\lceil \frac{(2n-1)m}{2n} \right\rceil$であり，最初の中継点のX座標値はmであるから，$L_z$は次式にて求められる．

$$L_z = m - \left\lceil \frac{(2n-1)m}{2n} \right\rceil + 1 = m + 1 - \left\lceil \frac{(2n-1)m}{2n} \right\rceil \quad (A-11)$$

最後に，中継点を含む画素列の長さL_sを求める．中継点は一つ前のディジタル直線を構成する画素列の終点であると同時に，次のディジタル直線を構成する画素列の始点でもあるため，(A-7)及び(A-11)によって求めた画素列から，次の様にして求めることができる.

$$L_s = L_z + L_a - 1 = m + 1 - \left\lceil \frac{(2n-1)m}{2n} \right\rceil + \left\lceil \frac{m}{2n} \right\rceil - 1 = \left\lceil \frac{m}{2n} \right\rceil - \left\lceil \frac{(2n-1)m}{2n} \right\rceil + m. \quad (A-12)$$

よって，式(A-7)，(A-10)，(A-11)及び(A-12)より，

$$L_a = \left\lceil \frac{m}{2n} \right\rceil, L_k = \left\lceil \frac{(2k+1)m}{2n} \right\rceil - \left\lceil \frac{(2k-1)m}{2n} \right\rceil$$

$$L_s=\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil+m, L_z=m+1-\left\lceil\frac{(2n-1)m}{2n}\right\rceil \quad\cdots\cdots(A-13)$$

となる．次に，これらの関係について調べる．前節の定理５より，

正の実数x_1及びx_2に対して，$\lceil x_1-x_2\rceil-1\le\lceil x_1\rceil-\lceil x_2\rceil\le\lceil x_1-x_2\rceil\le\lceil x_1\rceil-\lceil x_2\rceil+1$が成立するので，$x_1=\dfrac{(2k+1)m}{2n}, x_2=\dfrac{(2k-1)m}{2n}$とすると，$x_1-x_2=\dfrac{2m}{2n}=\dfrac{m}{n}$であるから，

$$\left\lceil\frac{m}{n}\right\rceil-1\le\left\lceil\frac{(2k+1)m}{2n}\right\rceil-\left\lceil\frac{(2k-1)m}{2n}\right\rceil\le\left\lceil\frac{m}{n}\right\rceil \quad\cdots\cdots(A-14)$$

また，定理３より，正の実数xに対して，$\lceil x\rceil\le 2\lceil x/2\rceil\le\lceil x\rceil+1$が成立するので，$x=\dfrac{m}{n}$とすると，

$$\left\lceil\frac{m}{n}\right\rceil\le 2\left\lceil\frac{m}{2n}\right\rceil\le\left\lceil\frac{m}{n}\right\rceil+1 \quad\cdots\cdots(A-15)$$

となり，

$$\left\lceil\frac{m}{n}\right\rceil-2\le 2\left\lceil\frac{m}{2n}\right\rceil-2\le\left\lceil\frac{m}{n}\right\rceil-1 \quad\cdots\cdots(A-16)$$

が成立する．従って，式(A-14)，(A-15)及び(A-16)より，

$$2\left\lceil\frac{m}{2n}\right\rceil-2\le\left\lceil\frac{m}{n}\right\rceil-1\le\left\lceil\frac{(2k+1)m}{2n}\right\rceil-\left\lceil\frac{(2k-1)m}{2n}\right\rceil\le\left\lceil\frac{m}{n}\right\rceil\le 2\left\lceil\frac{m}{2n}\right\rceil \quad\cdots\cdots(A-17)$$

ここで，式(A-13)より，$L_a=\left\lceil\dfrac{m}{2n}\right\rceil, L_k=\left\lceil\dfrac{(2k+1)m}{2n}\right\rceil-\left\lceil\dfrac{(2k-1)m}{2n}\right\rceil$であるから，

$$2L_a-2\le L_k\le 2L_a \quad\cdots\cdots(A-18)$$

となる．さらに定理５より，正の実数x_1及びx_2に対して，

$\lceil x_1-x_2\rceil-1\le\lceil x_1\rceil-\lceil x_2\rceil\le\lceil x_1-x_2\rceil\le\lceil x_1\rceil-\lceil x_2\rceil+1$が成立するので，

$x_1=\dfrac{(2n-1)m}{2n}, x_2=\dfrac{m}{2n}$とすると，$x_1-x_2=\dfrac{(2n-1)m}{2n}-\dfrac{m}{2n}=\dfrac{(2n-2)m}{2n}=m-\dfrac{m}{n}$

であるから，

$$\left\lceil m-\frac{m}{n}\right\rceil-1\le\left\lceil\frac{(2n-1)m}{2n}\right\rceil-\left\lceil\frac{m}{2n}\right\rceil\le\left\lceil m-\frac{m}{n}\right\rceil\quad\cdots\cdots(A-19)$$

となる．ここで，m と n は互いに素な正の整数であるから，m/n は整数ではない正の実数となる．

すると，定理４より，正の整数 m ，整数ではない正の実数 x に対して，

$\lceil m-x\rceil=m-\lceil x\rceil+1$

が成り立ち，$x=m/n$ とすると，$\left\lceil m-\frac{m}{n}\right\rceil=m-\left\lceil\frac{m}{n}\right\rceil+1\quad\cdots\cdots(A-20)$

となる．式(A-19)及び(A-20)より，

$$m-\left\lceil\frac{m}{n}\right\rceil=\left\lceil m-\frac{m}{n}\right\rceil-1\le\left\lceil\frac{(2n-1)m}{2n}\right\rceil-\left\lceil\frac{m}{2n}\right\rceil\le\left\lceil m-\frac{m}{n}\right\rceil=m-\left\lceil\frac{m}{n}\right\rceil+1$$

$$\therefore -m+\left\lceil\frac{m}{n}\right\rceil-1\le\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil\le-m+\left\lceil\frac{m}{n}\right\rceil$$

よって，$\left\lceil\frac{m}{n}\right\rceil-1\le\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil+m\le\left\lceil\frac{m}{n}\right\rceil\quad\cdots\cdots(A-21)$

となる．式(A-15)，(A-16)及び(A-21)より，

$$2\left\lceil\frac{m}{2n}\right\rceil-2\le\left\lceil\frac{m}{n}\right\rceil-1\le\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil+m\le\left\lceil\frac{m}{n}\right\rceil\le2\left\lceil\frac{m}{2n}\right\rceil$$

ここで，式(A-13)より，$L_a=\left\lceil\frac{m}{2n}\right\rceil, L_s=\left\lceil\frac{m}{2n}\right\rceil-\left\lceil\frac{(2n-1)m}{2n}\right\rceil+m$ であるから，

$\therefore 2L_a-2\le L_s\le 2L_a\quad\cdots\cdots(A-22)$

となる．また，定理４より，正の整数 m ，整数ではない正の実数 x に対して，

$\lceil m-x\rceil=m-\lceil x\rceil+1$ が成り立ち，m と n は互いに素な正の整数であるから，$\frac{m}{2n}$ は整数ではない正の実数となる．$x=\frac{m}{2n}$ とすると，$m-x=m-\frac{m}{2n}=\frac{(2n-1)m}{2n}$ であるから，

$\left\lceil\frac{(2n-1)m}{2n}\right\rceil=m-\left\lceil\frac{m}{2n}\right\rceil+1$ となり，式(A-13)より，

$L_a = \left\lceil \frac{m}{2n} \right\rceil, L_z = m+1-\left\lceil \frac{(2n-1)m}{2n} \right\rceil$を考慮すると,

$$L_a = \left\lceil \frac{m}{2n} \right\rceil = m+1-\left\lceil \frac{(2n-1)m}{2n} \right\rceil = L_z \quad \text{.....} (A-23)$$

となる．式(A-13)，(A-18)，(A-22)及び(A-23)より，

$$L_a = \left\lceil \frac{m}{2n} \right\rceil, L_k = \left\lceil \frac{(2k+1)m}{2n} \right\rceil - \left\lceil \frac{(2k-1)m}{2n} \right\rceil$$

$$L_s = \left\lceil \frac{m}{2n} \right\rceil - \left\lceil \frac{(2n-1)m}{2n} \right\rceil + m, L_z = m+1-\left\lceil \frac{(2n-1)m}{2n} \right\rceil$$

$$2L_a - 2 \le L_k \le 2L_a, 2L_a - 2 \le L_s \le 2L_a, L_z = L_a \quad \text{.....} (A-24)$$

となる.

III）$1/2 \le a = n/m < 1$（a は直線の傾き，n, mは互いに素な正の整数）の場合

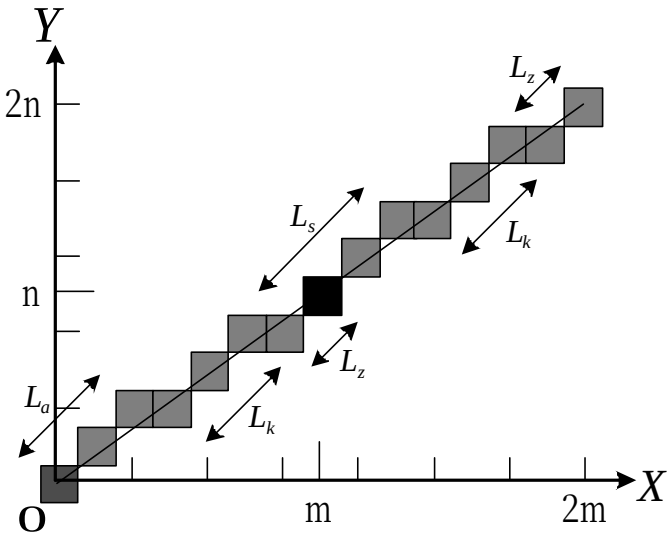


図 A-3 傾き$1/2 \le n/m < 1$のディジタル直線

この場合はⅠ)及びⅡ)と異なり，ディジタル直線の趨勢方向はチェイン符号１の方向である．従って，ディジタル直線を構成する画素列のうち，最初にＸ座標値とＹ座標値の差が１となる画素のＸ座標値 x_1 は，次式を満足する最小の整数である．

$$x_1 - \frac{n}{m}x_1 > \frac{1}{2} \quad \cdots\cdots(A-25)$$

従って，$x_1 > \dfrac{m}{2(m-n)}$ であるから，$x_1 = \left\lfloor \dfrac{m}{2(m-n)} \right\rfloor + 1$ と記述できる．

よって，ディジタル直線を構成する最初の画素列，つまり画素のＸ座標値とＹ座標値との差が０の個数は

$$L_a = \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1 \quad \cdots\cdots(A-26)$$

となる．一般に，ディジタル直線を構成する画素のＸ座標値とＹ座標値との差が k となる最初の画素のＸ座標値 x_k は，次式を満足する最小の整数である．

$$x_k - \frac{n}{m}x_k > (k-1) + \frac{1}{2} \quad \cdots\cdots(A-27)$$

従って，$x_k > \dfrac{(2k-1)}{2}\dfrac{m}{(m-n)}$ であるから，$x_k = \left\lfloor \dfrac{(2k-1)m}{2(m-n)} \right\rfloor + 1$ $\cdots\cdots(A-28)$

と記述できる．

よって，ディジタル直線を構成する画素列で，画素のＸ座標値とＹ座標値との差が k の個数は

$$L_k = x_{k+1} - x_k = \left\{ \left\lfloor \frac{\{2(k+1)-1\}m}{2(m-n)} \right\rfloor + 1 \right\} - \left\{ \left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor + 1 \right\} = \left\lfloor \frac{(2k+1)m}{2(m-n)} \right\rfloor - \left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor$$

$\cdots\cdots(A-29)$

ここで，Ⅱ)と同様に，全体のディジタル直線はX座標値が0からmまでの範囲のディジタル直線を繰り返すことにより構成されていることを考慮すると，最初の中継点を含む画素列の長さ L_s の一部に，終点を含む画素列の長さ L_z が含まれている．最初にX座標値とY座標値の差がm－nとなるX座標値 x_{m-n} は式(A-28)より， $x_{m-n}=\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+1$ である．最初の中継点のX座標値はmであるから， L_z は次式にて求められる．

$$L_z=m-\left\{\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+1\right\}+1=m-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor\cdots\cdots\cdots\cdots\cdots\cdots(A-30)$$

最後に，中継点を含む画素列の長さ L_s を求める．中継点は一つ前のディジタル直線を構成する画素列の終点であると同時に，次のディジタル直線を構成する画素列の始点でもあるため，(A-26)及び(A-30)によって求めた画素列から，次の様にして求めることができる．

$$L_s=L_z+L_a-1=m-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+\left\lfloor\frac{m}{2(m-n)}\right\rfloor+1-1$$

$$=\left\lfloor\frac{m}{2(m-n)}\right\rfloor-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+m\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots(A-31)$$

よって，式(A-26)，(A-29)，(A-30)及び(A-31)より，

$$L_a=\left\lfloor\frac{m}{2(m-n)}\right\rfloor+1,L_k=\left\lfloor\frac{(2k+1)m}{2(m-n)}\right\rfloor-\left\lfloor\frac{(2k-1)m}{2(m-n)}\right\rfloor$$

$$L_s=\left\lfloor\frac{m}{2(m-n)}\right\rfloor-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor+m,L_z=m-\left\lfloor\frac{\{2(m-n)-1\}m}{2(m-n)}\right\rfloor\cdots\cdots\cdots\cdots\cdots\cdots(A-32)$$

となる．次に，これらの関係について調べる．前節の定理7より，正の実数 x_1 及び x_2 に対して， $\lfloor x_1\rfloor-\lfloor x_2\rfloor-1\le\lfloor x_1-x_2\rfloor\le\lfloor x_1\rfloor-\lfloor x_2\rfloor\le\lfloor x_1-x_2\rfloor+1$ が成立する．

$x_1=\frac{(2k+1)m}{2(m-n)},x_2=\frac{(2k-1)m}{2(m-n)}$ とすると，

$x_1-x_2=\frac{(2k+1)m}{2(m-n)}-\frac{(2k-1)m}{2(m-n)}=\frac{2m}{2(m-n)}=\frac{m}{m-n}$ であるから，

$$\left\lfloor \frac{m}{m-n} \right\rfloor \le \left\lfloor \frac{(2k+1)m}{2(m-n)} \right\rfloor - \left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor \le \left\lfloor \frac{m}{m-n} \right\rfloor + 1 \quad \cdots\cdots (A-33)$$

となる．ここで，m と n は互いに素な正の整数であるから，m/n は整数ではない正の実数となる．また，定理６より，正の実数 x に対して，$\lfloor x \rfloor - 1 \le 2\lfloor x/2 \rfloor \le \lfloor x \rfloor$ が成立するので，$x = \dfrac{m}{m-n}$ とすると，

$$\left\lfloor \frac{m}{m-n} \right\rfloor - 1 \le 2\left\lfloor \frac{m}{2(m-n)} \right\rfloor \le \left\lfloor \frac{m}{m-n} \right\rfloor \text{ となり,}$$

$$\therefore \left\lfloor \frac{m}{m-n} \right\rfloor - 1 \le 2\left\{ \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1 \right\} - 2 \le \left\lfloor \frac{m}{m-n} \right\rfloor \quad \cdots\cdots (A-34)$$

$$\therefore \left\lfloor \frac{m}{m-n} \right\rfloor + 1 \le 2\left\{ \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1 \right\} \le \left\lfloor \frac{m}{m-n} \right\rfloor + 2 \quad \cdots\cdots (A-35)$$

よって，式(A-33)，(A-34)及び(A-35)より，

$$2\left\{ \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1 \right\} - 2 \le \left\lfloor \frac{m}{m-n} \right\rfloor \le \left\lfloor \frac{(2k+1)m}{2(m-n)} \right\rfloor - \left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor \le \left\lfloor \frac{m}{m-n} \right\rfloor + 1 \le 2\left\{ \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1 \right\}$$

$$\cdots\cdots (A-36)$$

ここで式(A-32)より，$L_a = \left\lfloor \dfrac{m}{2(m-n)} \right\rfloor + 1, L_k = \left\lfloor \dfrac{(2k+1)m}{2(m-n)} \right\rfloor - \left\lfloor \dfrac{(2k-1)m}{2(m-n)} \right\rfloor$ であるから,

$$\therefore 2L_a - 2 \le L_k \le 2L_a \quad \cdots\cdots (A-37)$$

となる．また，$\left\lfloor \dfrac{2(m-n)-1}{2} \dfrac{m}{(m-n)} \right\rfloor = \left\lfloor m - \dfrac{m}{2(m-n)} \right\rfloor$ であり，さらに，定理８より，正の整数 m，正の実数 x に対して，$m - \lfloor x \rfloor - 1 \le \lfloor m - x \rfloor \le m - \lfloor x \rfloor$ が成立するので，$x = \dfrac{m}{2(m-n)}$ とすると，

$$m - \left\lfloor \frac{m}{2(m-n)} \right\rfloor - 1 \le \left\lfloor m - \frac{m}{2(m-n)} \right\rfloor \le m - \left\lfloor \frac{m}{2(m-n)} \right\rfloor$$

$$\therefore m - \left\lfloor \frac{m}{2(m-n)} \right\rfloor - 1 \le \left\lfloor \frac{\{2(m-n)-1\}}{2} \frac{m}{(m-n)} \right\rfloor \le m - \left\lfloor \frac{m}{2(m-n)} \right\rfloor$$

$$\therefore -m+\left\lfloor \frac{m}{2(m-n)} \right\rfloor \le -\left\lfloor \frac{\{2(m-n)-1\}}{2}\frac{m}{(m-n)} \right\rfloor \le -m+\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1 \cdots\cdots\cdots (A-38)$$

$$\therefore 2\left\lfloor \frac{m}{2(m-n)} \right\rfloor \le \left\lfloor \frac{m}{2(m-n)} \right\rfloor -\left\lfloor \frac{\{2(m-n)-1\}}{2}\frac{m}{(m-n)} \right\rfloor +m \le 2\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1$$

$$\therefore 2\left\{\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1\right\}-2 \le \left\lfloor \frac{m}{2(m-n)} \right\rfloor -\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor +m \le 2\left\{\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1\right\}-1$$

ここで，式(A-32)より，

$L_a=\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1, L_s=\left\lfloor \frac{m}{2(m-n)} \right\rfloor -\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor +m$ であるから，

ゆえに，$2L_a-2 \le L_s \le 2L_a-1 \cdots\cdots\cdots (A-39)$

となる．また，式(A-38)より，

$$\left\lfloor \frac{m}{2(m-n)} \right\rfloor \le m-\left\lfloor \frac{\{2(m-n)-1\}}{2}\frac{m}{(m-n)} \right\rfloor \le \left\lfloor \frac{m}{2(m-n)} \right\rfloor +1$$

$$\therefore \left\{\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1\right\}-1 \le m-\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor \le \left\lfloor \frac{m}{2(m-n)} \right\rfloor +1$$

ここで，式(A-32)より，$L_a=\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1, L_z=m-\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor$ であるから，

ゆえに，$L_a-1 \le L_z \le L_a \cdots\cdots\cdots (A-40)$

となる．式(A-32)，(A-37)，(A-39)及び(A-40)より，

$$L_a=\left\lfloor \frac{m}{2(m-n)} \right\rfloor +1, L_k=\left\lfloor \frac{(2k+1)m}{2(m-n)} \right\rfloor -\left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor$$

$$L_s=\left\lfloor \frac{m}{2(m-n)} \right\rfloor -\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor +m, L_z=m-\left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor$$

$2L_a-2 \le L_k \le 2L_a, 2L_a-2 \le L_s \le 2L_a-1, L_a-1 \le L_z \le L_a \cdots\cdots\cdots (A-41)$

となる．

以上をまとめると式(A-5)，(A-24)及び(A-41)より，ディジタル直線の性質は次の様になる．

＜ディジタル直線の性質＞

Ⅰ) $0 < a = 1/h < 1/2$ **（**a **は直線の傾き，**h **は正の整数）の場合**

$$L_a = \lceil h/2 \rceil, L_s = h, L_z = (h+1) - \lceil h/2 \rceil$$
$$2L_a - 1 \le L_s \le 2L_a, L_a \le L_z \le L_a + 1$$

Ⅱ) $0 < a = n/m < 1/2$ **（**a **は直線の傾き，**n, m **は互いに素な正の整数）の場合**

$$L_a = \left\lceil \frac{m}{2n} \right\rceil, L_k = \left\lceil \frac{(2k+1)m}{2n} \right\rceil - \left\lceil \frac{(2k-1)m}{2n} \right\rceil$$

$$L_s = \left\lceil \frac{m}{2n} \right\rceil - \left\lceil \frac{(2n-1)m}{2n} \right\rceil + m, L_z = m + 1 - \left\lceil \frac{(2n-1)m}{2n} \right\rceil$$

$$2L_a - 2 \le L_k \le 2L_a, 2L_a - 2 \le L_s \le 2L_a, L_z = L_a$$

Ⅲ) $1/2 \le a = n/m < 1$ **（**a **は直線の傾き，**n, m **は互いに素な正の整数）の場合**

$$L_a = \left\lfloor \frac{m}{2(m-n)} \right\rfloor + 1, L_k = \left\lfloor \frac{(2k+1)m}{2(m-n)} \right\rfloor - \left\lfloor \frac{(2k-1)m}{2(m-n)} \right\rfloor$$

$$L_s = \left\lfloor \frac{m}{2(m-n)} \right\rfloor - \left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor + m, L_z = m - \left\lfloor \frac{\{2(m-n)-1\}m}{2(m-n)} \right\rfloor$$

$$2L_a - 2 \le L_k \le 2L_a, 2L_a - 2 \le L_s \le 2L_a - 1, L_a - 1 \le L_z \le L_a$$