高品質画像生成法の研究

1992年8月

1143049
田中 敏光

目次

図目次

表目次

第 1 章

序論

1.1 研究の目的

不適切なサンプリングが画像に及ぼす悪影響をエリアシング (aliasing) と呼ぶ。エリアシングは画像品質をいちじるしく低下させる。本研究では、エリアシングを確実に除去するため、解析的な画像生成手法を確立する。

コンピュータグラフィックス (CG) はコンピュータを用いて多値のデジタル画像を生成する技術である。デジタル画像は格子状に並んだ画素 (pixel) で構成される。画素は画像の最小単位で、それぞれが明るさと色を表示する。コンピュータグラフィックスでは画素を赤 (R), 緑 (G), 青 (B) 一組の輝度値で記述することが一般的である。

近年コンピュータグラフィックス技術はさまざまな分野で利用されている。形状デザイン, シミュレーション, ビジュアライゼーション, バーチャルリアリティなどの工学分野に限らず、医療, 娯楽, 宣伝, 芸術などの分野でも利用が進んでいる。CG 画像が日常的になるにつれ、画質に対する要求も高くなっている。

CG 画像の品質を向上するには、

(1) 表示可能な物体の形状や物理的性質を拡大し,

(2) 精密にデジタル画像を生成し,

(3) 明るさや色を正確に表示する,

画像生成技術の改善が必要である。このうち (3) はデジタル画像の表示で一般的に求められている技術で、分解能が高く発色性の良いカラープリンタや高解像度モニタの開発により改善が進んでいる。動画についても、現行のテレビジョン画像 (NTSC 方式) の 5 倍の画素数を持ち色帯域も広い HDTV の普及により、表示や記録の環境が整備されつつある。(1) では多数の研究が行われ、年々扱える形状や性質の範囲が拡大している。物体形状の記述, 物体表面での反射特性, 光源の特性, 相互反射, 動きの生成と制御, などのコンピュータグラフィックスの基本要素のそれぞれにおいて、より高度なモデルが提案されている。しかし、(2) については、十分な技術が確保されてお

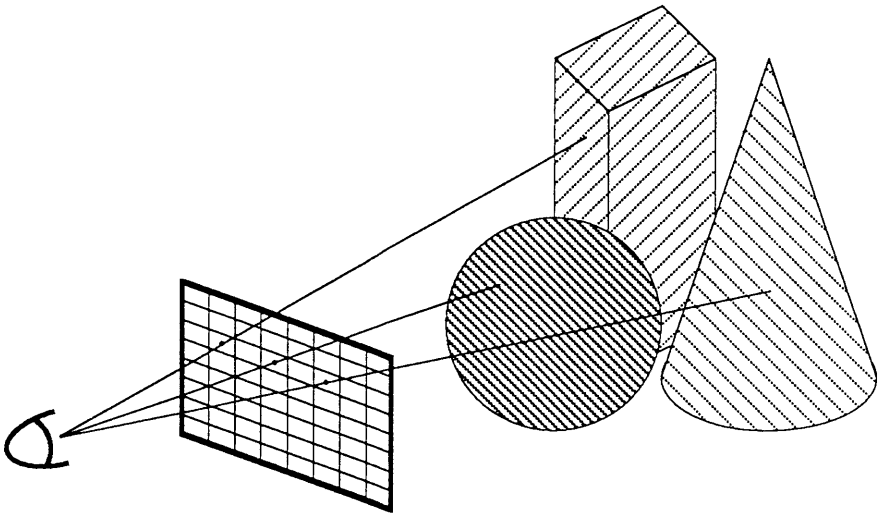


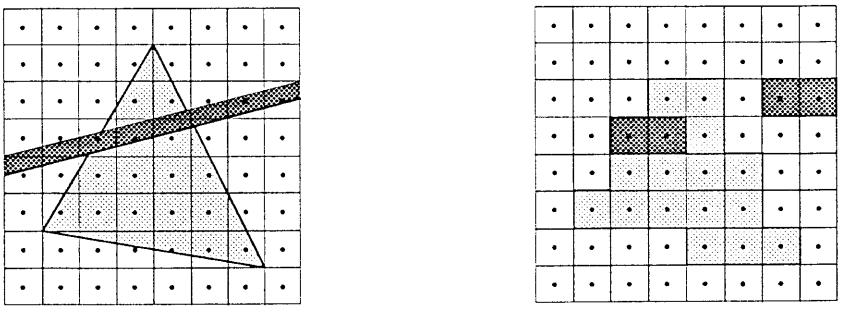
図 1-1: CG 画像の生成

らず、厳密なデジタル画像の生成には困難が伴う。以下にデジタル画像生成の問題点を示す。

コンピュータグラフィックスでは、図 1-1に示すように、視点と画面の位置を決め、各画素に投影される物体を求めることで画像を生成する。画素の明暗は R,G,B 一組の輝度値で記述されるので、図 1-1のように画素の中心点で最も視点に近い物体を求めてその明るさを計算するだけで、とりあえず画像を生成できる。画素を中心点で代表させることで、画像生成アルゴリズムが単純になり計算時間も短縮できる。このため、コンピュータグラフィックスでは画素を点または線分で近似することが一般的である。しかし、画素を 1 点で代表させると障害が発生する。

CCD(Charge Coupled Device) のように画素に入射する光の強度に比例して電位が変わる撮像素子を考えよう。簡単のため、画素は正方形で隙間なく並べられているものとする。図 1-2(a) に例示されるように、物体の輪郭では 1 つの画素に背景を含めて 2 つ以上の物体が投影される。したがって、輪郭線上の画素の電位は物体の明るさをその物体が画素内に占める面積で重み付けして合計した値になる。ところが、画素の中心点にその画素を代表させると、物体の輪郭線が通過する画素であってもどれか 1 つの物体しか選ばれない。たとえば、図 1-2(a) の物体からは図 1-2(b) に示すように選択される。

画像の例を図 1-3に示す。この画像は画素の大きさが判る程度まで拡大表示されている。画素に投影される物体の明るさをその物体が画素内に占める面積で重み付けして足し合わせると、図 1-3(a) の画像が得られる。しかし、画素の中心に投影される物体の明るさで画素の輝度値を与えると図 1-3(b) になる。この例のように、本来はなめらかな物体の輪郭線が階段状に表示されてしまう現象をジャギ (jaggy) と呼ぶ。特に



(a) 元の物体　　(b) 選択された物体

図 1-2: 画素を中心点で代表

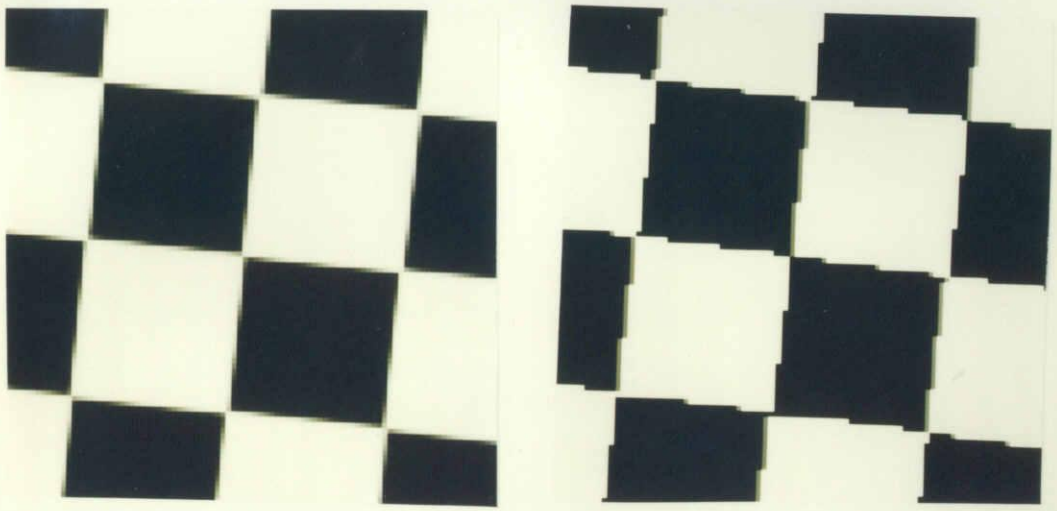
図 1-2(a) のようにポリゴンが細い場合には図 1-2(b) のように切れ切れの線になる。これもジャギの一種である。

図 1-4は扇形に広がるパターンを表示しているが、画素中心の輝度値を用いると図 1-4(b) のように定義されていない模様が現れる。この模様はモアレ (moiré) と呼ばれ、表示する物体の間隔とサンプリング周期との相乗効果により生じる。面積を重みにして物体の輝度を合計すると図 1-4(a) のように表示される。モアレが完全に除去されるわけではないが (b) に比べればはるかに改善されている。

ジャギやモアレは画素の中心点が画素全体を代表するために発生する。コンピュータグラフィックスはデジタル (離散) 画像を生成する技術であるが、最終出力が離散的だからといって途中の処理を単純に離散化してよいわけではない。画像生成の途中で不用意にサンプリング (離散化処理) を行うとジャギやモアレのような思わぬ障害が発生する。

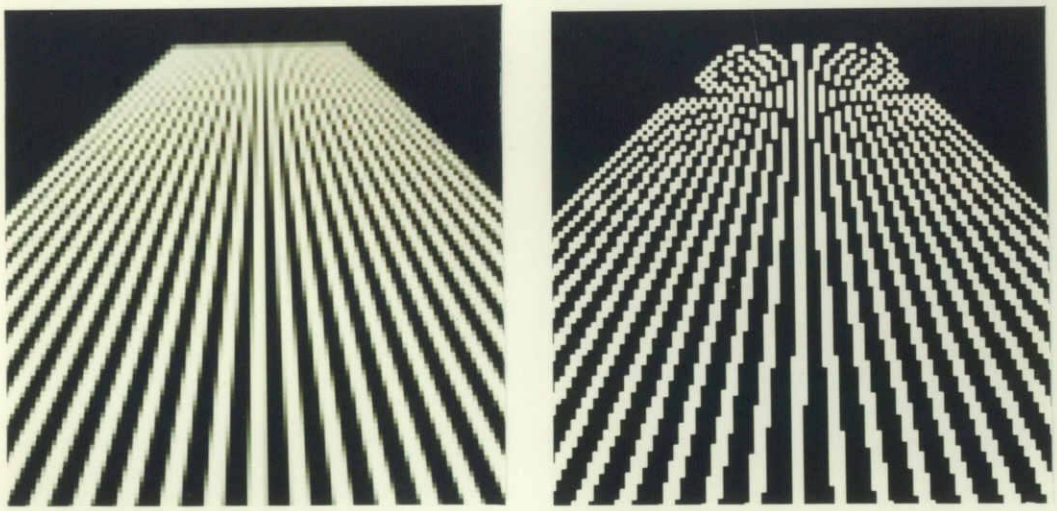
サンプリングにより発生する諸問題をエリアシングと呼ぶ。図 1-3と図 1-4に示されたようにエリアシングは CG 画像の品質を著しく低下させるので、エリアシングの厳密な除去は高品質画像を生成するために必須である。エリアシングの原因は不適切なサンプリングにあるので、エリアシングを根本的に解消するにはサンプリングを用いない画像生成手法が必要である。そこで、本研究では解析的に画像を生成する手法を確立する。

本論文では、はじめに、代表的なアンチ・エリアシング手法とそれらの問題点とを述べる (1章)。つぎに、解析的に画像を生成するため、1 画素に投影されるポリゴン領域を正確に計算できる隠れ面消去手法 (2章), 曲面の反射を精密に積分する輝度計算手法 (3章), 面光源をそのまま扱える照明手法 (4章) を報告する。さらに、積分テーブルを用いて画像生成時間を短縮する (5章)。最後に、本研究の成果をまとめる (6章)。



(a) 物体の面積に比例して輝度を計算　　(b) 画素中心の物体から輝度を計算

図 1-3: ジャギ



(a) 物体の面積に比例して輝度を計算　　(b) 画素中心の物体から輝度を計算

図 1-4: モアレ

1.2 従来のアンチ・エリアシング手法

1.2.1 アンチ・エリアシング手法の分類

画像生成におけるサンプリングの悪影響 (エリアシング) を取り除く処理を総称してアンチ・エリアシング (anti-aliasing) と呼ぶ。エリアシングは画面のごく一部にしか現れなくても画像全体の品質を著しく劣化させてしまう。アンチ・エリアシングはコンピュータグラフィックスの重要な研究課題である。

画像品質に大きな影響を与えるエリアシングは主として隠れ面消去で発生する。このため、従来手法の多くは隠れ面消去からエリアシングを除去することを目的としている。隠れ面消去手法ごとに分類した代表的なアンチ・エリアシング手法を表 1-1に示す。表 1-1の手法は 1 画素内を細かく分割する画像生成手法と後処理または前処理として行われるフィルタ手法とに分けられる。前者はさらにスーパーサンプリング (super sampling) 手法 (表 1-1の 1–3,5–11,13,15), 幾何情報を用いた領域分割手法 (表 1-1の 12,16,18,20–22) に大別することができる。

スーパーサンプリング手法は 1 画素内の複数の点をサンプル点として隠れ面消去する。画素の輝度値はサンプル点の輝度の平均で与えられる。各サンプル点の輝度は画素に投影される物体の大きさや画素内での物体表面の変化を反映するので、真の輝度値との差が小さくなる。したがって、エリアシングが減少する。効率よくアンチ・エリアシングするには、サンプル点を適切に配置する必要がある。サンプル点を配置できる場所は隠れ面消去のアルゴリズムに依存するので、隠れ面消去手法ごとに異なるスーパーサンプリング手法が提案されている。

物体の稜線の向きや頂点の位置などの幾何情報を用いると 1 画素内の物体の面積を推定できる。クリッピングを用いた手法では正確な面積が求められる。ただし、視点から見える物体の幾何情報が必要なため、適用できる隠れ面消去アルゴリズムは限定される。

エリアシングは画像の高周波成分により発生するため、フィルタで高周波成分を取り除けばエリアシングを軽減できる。隠れ面消去の後でフィルタリングするため、隠れ面消去手法ごとに適したフィルタ手法が存在する。フィルタ手法の改良は本論文では言及しないので、代表的な手法を示すだけにとどめる。

以下の節で、表 1-1に示した手法を概説する。

1.2.2 Zバッファ法のアンチ・エリアシング

Zバッファ (Z-buffer) は画像と同じ大きさの 2 次元配列で、各要素は対応する画素に投影される物体の番号と視点からその物体までの距離を記録するために使われる。Zバッファアルゴリズムは以下の手順で視点に最も近い物体を決定する [6]。はじめに、

表 1-1: 代表的なアンチ・エリアシング手法

隠れ面消去手法	アンチ・エリアシング手法
Zバッファ法	画素をさらに細かく分割する手法
	- サブ・ピクセル法 : 1
	物体の寄与率を保存する手法
	- Aバッファ法 : 2
	- 複数のZバッファを用いる手法 : 3
	デジタルフィルタ : 4
光線追跡法	格子点をサンプリングする手法
	- 適応的スーパーサンプリング法 : 5
	統計的にサンプリングする手法
	- distributed ray-tracing 法 : 6
	1 画素内を効果的にサンプリングする改良
	- Poisson sampling 法 : 7
	- jittered sampling 法 : 8
	- importance sampling 法 : 9
	サンプル点の数を削減する改良
	- 画素細分割手法 : 10
	領域をサンプリングする手法
	- cone tracing 法 : 11
	- beam tracing 法 : 12
	- pencil tracing 法 : 13
	関数フィルタ : 14
スキャンライン法	スキャンライン間を細かく分割する手法
	- マルチスキャニング法 : 15
	面積を推定する手法
	- スキャンラインに沿った面積推定法 : 16
	線積分フィルタ : 17
優先順位法	面積を推定する手法
	- 物体の輪郭に沿った面積の推定法 : 18
	輪郭線に沿ったフィルタリング : 19
領域細分法	クリッピング手法
	- 多角形どうしをクリッピングする手法 : 20
	- 画素で多角形をクリッピングする手法 : 21
	- 走査線単位のクリッピング手法 : 22
	領域積分フィルタ : 23

Zバッファの全ての要素に背景までの距離を記録する。次に、物体ごとにその物体が投影される全ての画素で視点から物体までの距離を計算し、この距離がZバッファに記録された値より小さいときに限りバッファの内容を更新する。この結果、Zバッファには常に視点にいちばん近い物体が記録される。全ての物体を処理した時点でZバッファに記録された物体が、視点に最も近い物体である。

サブ・ピクセルを用いたZバッファ法

エリアシングを削減するには画素を複数の点でサンプリングして画素に投影される物体の面積を近似的に計算する必要がある。Zバッファ法では画面を等間隔に分割すると視点から物体までの距離を効率良く計算できる。そこで、1画素の中がさらに細かい格子で分割されるように画面全体を均一に細分割し、1画素を複数の点で代表させる [6]。分割された最小単位をサブ・ピクセル (sub-pixel) と呼ぶ。画素の輝度値はサブ・ピクセルの輝度値の平均で与えられる。平均のしかたには、単純に画素内の和をとる方法や隣接する画素を含めて加重平均をとる方法がある。詳しくはフィルタリングの項で述べる。

サブ・ピクセルを通常の画素と見なせば、サブ・ピクセルを用いたアンチ・エリアシングは通常のZバッファ法で生成した高解像度画像をローパスフィルタを通して表示することと等価である。このため、Zバッファ法の隠れ面消去アルゴリズムを変更する必要はない。インプリメントも簡単である。Zバッファ法を高速実行するハードウェアも手直しなしで利用できる。高解像度画像を生成するため、隠れ面消去だけでなく物体表面の法線方向の変化やマッピングによるエリアシングも同時に改善できる。

画面全体を均一に分割する方法は一つの物体で覆われる画素も細かく分けてしまうので、無駄なサンプル点が多くなる。計算量は1画素あたりのサブ・ピクセルの数に比例して増加するので、分割が細かくなると処理時間が長くなる。また、余り細かく分割すると、Zバッファに必要な記憶領域が通常のワークステーションのメモリ容量を越えるほど大きくなる。このため、この手法は精密なアンチ・エリアシングには適さない。

Aバッファ法

処理時間と記憶領域を削減するため、物体の前後判定は画素単位で行い物体の領域はサブ・ピクセル単位で記録するAバッファ法 [5] が提案されている。この手法では、画素ごとに、その画素に投影される全ての物体について、(1) 画素内の1点で計算された視点から物体までの距離と、 (2) 物体が投影されるサブ・ピクセル, とを記録する。(2) は物体がサブ・ピクセルを含むか否かを図1-5に示すようなビットマスクで記録する。

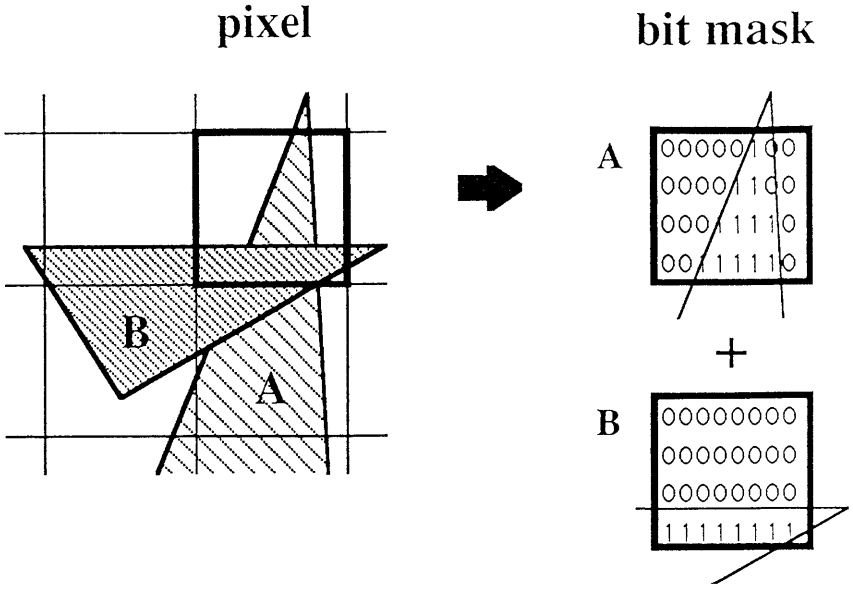


図 1-5: Aバッファ法

各画素ごとに、視点から近い順に物体を並べる。1画素の中で最も手前の物体が画素を占有する割合をビットマスクから計算し、その物体の画素値への寄与率とする。2番目の物体の寄与率は2番目の物体のビットマスクと1番目の物体に含まれないサブ・ピクセルとの論理積から計算できる。さらに物体に含まれないサブ・ピクセルが残っている場合には、すべてのサブ・ピクセルがいずれかの物体に含まれるか物体がなくなるまで同様の処理を繰り返す。こうして、画素の輝度に寄与する物体とその寄与率が決定できる。画素の輝度値は物体の輝度を寄与率で重み付けして合計することで計算できる。

Aバッファ法では画素単位でしか前後判定を行わないため、単純にサブ・ピクセルに分割する場合に比べ計算量は少ない。また、物体が覆う領域をビットマスクで保存するため記憶領域も削減できる。ただし、物体の輝度値は画素内では一定であると仮定しているため、物体表面の曲率が大きいときには正しい輝度値が求められない。物体が他の物体を貫通する場合には交線を含む画素の中でこれらの物体の前後関係が逆転する。Aバッファ法は画素内の1点だけで前後判定を行うため、交線を正しく表示できない。1画素に投影される全ての物体を記録するにはリスト形式のデータ構造を取り扱う必要がある。このため、Aバッファ法のアルゴリズムは基本的なZバッファアルゴリズムとは大きく異なる。

複数のZバッファを用いる手法

リスト形式のデータは操作が複雑でZバッファアルゴリズムの効率を悪くする。画素の輝度値に寄与する物体の数を制限すれば、Aバッファと同様のアンチ・エリアシングを複数のZバッファを用いて実現できる[37]。この場合、リストを扱う必要はない。

視点に近い順に2つの物体の寄与だけを考慮する場合は、物体番号×2, 視点から物体までの距離×2, 物体の寄与率×1, を保存する領域を画素ごとに用意する。通常のZバッファ法と同じように画素ごとに視点からの奥行きを比較するが、新しく追加する物体が記憶されている2つの物体の, (1) 前にあれば、現在最も視点に近い物体を2番目に移したあとで追加する物体の番号とその寄与率とを1番目に記録する, (2) 間にあれば、2番目の物体を新しい物体で置き換える, (3) 後ろにあれば、何もしない。3つ以上の物体の寄与を考慮したければ前述の3種類の領域を物体の数だけ用意すれば良い。

この手法は通常のZバッファアルゴリズムと類似しているため、比較的容易に実現できる。物体の輪郭を含む画素でも3つ以上の物体が輝度に関与する場合は少ないので、この手法でも十分にアンチ・エリアシングの効果がある。ただし、Aバッファ法と同様に‘画素内の輝度の変かが反映されない’,‘物体同士の交線を正しく表示できない’などの問題点を持つ。加えて、用意したバッファ数以上の物体が1画素に現れると正しく表示できないので、込み入った場面では効果が期待できない。

デジタルフィルタ

画像を格子状に均一に分割する隠れ面消去手法にはデジタルフィルタが適している。隠れ面消去後または輝度計算後にサブ・ピクセルをフィルタリングして画素の輝度値を求める。フィルタ配列の要素の値は、物体の外形線の方向により輝度値に差が生じないように、円錐関数やガウス関数などの軸対称関数を用いて与える。画像の高周波成分によるエリアシングを削減するため2×2画素や3×3画素の大きさのフィルタを用いることも多い。ただし、フィルタが大きくなると画像の解像度が低下するうえ計算量も増加する。

1.2.3 光線追跡法のアンチ・エリアシング

光線追跡(ray-tracing)法[49]は画素を通って視点に到達する光線を視点から逆に追跡する。追跡する光線が最初に交差する物体が視点から見える物体である。光線追跡法は隣接する画素の類似性を用いないので、それぞれの追跡光線は完全に独立して処理できる。したがって、画素内のどの位置をもサンプル点として選べるし、どこに選

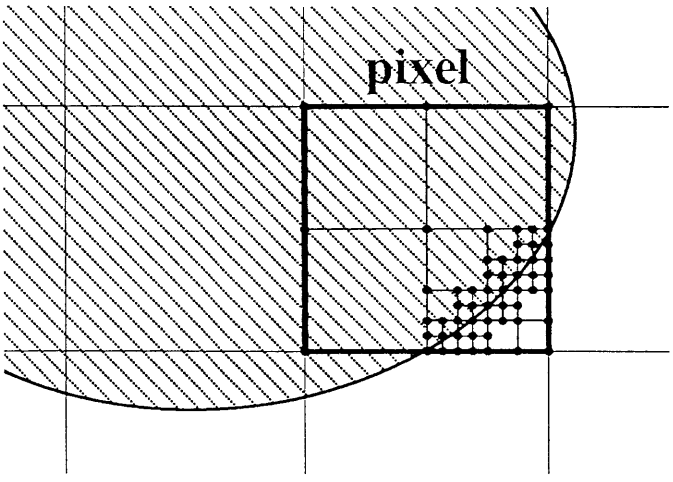


図 1-6: 適応的スーパーサンプリング法

んでも処理時間に大した差はない。そこで、この特性を生かしたスーパーサンプリング手法が提案されている。

適応的スーパーサンプリング法

光線追跡法の計算コストはZバッファ法に比べはるかに高い。このため、画像を一律に細分割する手法は画像生成時間が長大になりすぎて実用的ではない。そこでエリアシングが発生しそうな場所に重点的にサンプル点を配置する適応的スーパーサンプリング (adaptive supersampling) 法 [49] が考案されている。

エリアシングは物体の輪郭部で発生する。そこで、以下の手順により輪郭線を含む画素を細かく分割する。最初のサンプル点を画素の頂点に配置する。図 1-6のように正方形に並んだ 4 つのサンプル点を通る光線のうちの少なくとも 1 本が異なる物体と交差する場合には、その正方形を物体の輪郭線が横切ると考えられる。そこで、正方形を 4 等分して、新しく加わった頂点で光線を追跡する。4 つの小正方形に対しこの分割処理を再帰的に繰り返す。決められた細かさまで分割されたら処理を打ち切り、各サンプル点が代表する面積を重みにして輝度値の平均を求める。

適応的にスーパーサンプリングすることで等間隔にスーパーサンプリングする場合に比べサンプル点の数を削減できる。光線追跡法の特性から画素ごとのサンプル点の数や位置は自由に選べるので、部分的に細かくサンプリングしてもアルゴリズム上の不都合はない。ただし、サンプリングの結果を基に区間の分割を判定するため、細い物体や小さい物体が含まれる場合には細分割されないことがある。このため、精密に

エリアシング除去できる保証がない。

Distributed ray-tracing 法

画素を格子状のサブピクセルに分解するとモアレのような周期サンプリングの弊害が発生しやすい。そこで、distributed ray-tracing 法 [9] では乱数を用いてサンプル点を配置する。このサンプリング手法は stochastic sampling と呼ばれる。

一般的な鏡面反射では一定の方向領域から入射する光が視線方向に反射する。しかし、通常の光線追跡法は正反射方向の光線しか追跡しないので、光源の位置や表面の曲率によっては正しい輝度値が求められない。distributed ray-tracing 法では正反射方向の近傍で複数の光線を追跡することで、輝度値の精度を改善する。

この手法は隠れ面消去と輝度計算の両方のエリアシングを削減できる。ただし、計算コストの高い光線追跡処理を 1 画素あたり多数行う必要があるため、計算時間がきわめて長くなる。実用的には追跡光線の本数をそれほど多くすることはできないので、物体の複雑さ, 物体表面の曲率, 反射率, 反射の鋭さ, などによってはサンプル数が不足する。このため、確実にエリアシング除去できる保証はない。

Distributed ray-tracing のサンプル点配置手法

全くランダムにサンプル点を配置すると、サンプル点が画素内の一部分に片寄ることがある。この場合、輝度値は厳密な値から大きくずれる。このような画素は画面上にランダムに現れるので、ノイズの乗った画像になる。格子点のような周期的な点をサンプル点に選べばノイズは発生しないが、エリアシングは増大する。

ランダム配置で発生するノイズと周期配置で発生するエリアシングとは一方を減らすと他方が増加する関係にある [10]。少ないサンプル点で高品質画像を生成するため、以下のサンプリング手法が考案されている。

- Poisson sampling 法 [12]

  サンプル点相互の距離が一定値以上に保たれるようにサンプル点を配置すると、ノイズの発生を防ぎつつエリアシングを削減できる。このようなサンプルパターンは、

  1. 画素を十分に細かい格子に分割する。格子点の 1 点をサンプル点として登録する,
  2. 残った格子点の中から乱数を用いて 1 点選ぶ。その点から一定の範囲内にすでに登録されたどのサンプル点も存在しなければ、その点をサンプル点に追加する,

3. 追加できる格子点がなくなるまで 2 の処理を繰り返す,

ことで近似的に求められる。ただし、このアルゴリズムでは格子点の数がサンプル点の数よりずっと多くなるように格子の間隔を細かくする必要があるので、計算コストは極めて高い。格子点を 1 回走査するだけでサンプル点を選び出す手法も提案されているが [22]、コストの削減は十分とはいえない。

- ジッタサンプリング (jittered sampling) 法 [10]

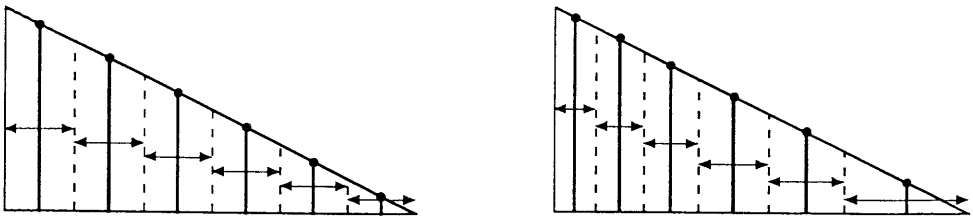
  ジッタサンプリング法はランダムでかつ画素全体に分布するサンプリングパターンを小さいコストで生成できる。この手法は格子点から微小距離だけランダムにずらした点をサンプリングする。サンプル点の相互の距離は隣会う格子点の距離からずらす距離の 2 倍を引いた値より小さくならないので、ずらす距離を適切に選べば Poisson sampling に近い良好な結果が得られる。ジッタサンプリング法ではサンプル点の数だけずらし処理を行えばよいので、 Poisson sampling 法に比べずっと小さい計算コストでサンプリングパターンを生成できる。

  ジッタサンプリングを 1 画素 1 点の光線追跡法に適用しても、サンプリングの間隔が物体の空間解像度よりも大きいため、モアレは削減されない。Z バッファを用いた隠れ面消去に適用するとサンプリングの周期性が失われるため処理効率が悪くなる。スキャンライン法は走査線単位で処理を行うためジッタサンプリングは使えない。ジッタサンプリングをマッピングの参照アドレスの決定に用いると効果的にアンチ・エリアシングできる。

- importance sampling 法 [20]

  多重反射する場合、反射を繰り返すに従い反射光の強度が画素の輝度値に与える影響は小さくなる。したがって、高次の反射光を粗くサンプリングしてもエリアシングはさほど増加しない。また、サンプル点の輝度をフィルタで加重平均する場合には、重みの小さい部分を粗くサンプリングしても誤差の影響は少ない。

  importance sampling は画素値への寄与が高い部分を細かく、低い部分を粗くサンプリングすることで近似誤差を縮小する。図 1-7に円錐フィルタで加重平均する場合の例を示す。棒の位置が標準のサンプリングの位置で矢印がジッタサンプリングでずらす範囲を示す。棒の高さは重みの大きさを示す。誤差はサンプル点の輝度とサンプル点が代表する範囲の平均の輝度との差で定義される。一般的には、代表する範囲が広いほど誤差も大きくなる。通常のジッタサンプリング (図 1-7(a)) では、誤差の大きさは同じだが左端のサンプル点の重みが最も大きいので、左端のサンプル点の誤差が支配的になる。 importance sampling(図



(a) 通常のジッタサンプリング　　(b)importance sampling

図 1-7: サンプル点の配置の比較

1-7(b)) では全てのサンプル点で誤差と重みの積が等しくなるので、どのサンプル点の誤差も等しく輝度値に影響を与える。この結果、重みの大きい箇所を細かく分割する分だけ誤差の期待値を小さくできる。

Distributed ray-tracing の画素細分割手法

光線追跡法は極めて計算コストが高いため、サンプル点をなるべく減らす必要がある。distributed ray-tracing 法でも適応的スーパーサンプリング法と同様にサンプル点を特定の画素に重点的に配置する必要がある。

詳細に分割するか否かを判断する基準としては、1 画素内のサンプル点の輝度のばらつきを用いる方法 [19] や輝度値のばらつきと輝度の平均値との比を用いる方法 [12] が提案されている。サンプリングの密度も、通常のサンプリングと精密なサンプリングの 2 通りを使う方法 [10] と適応的にサンプル点を追加していく方法 [20] が提案されている。

領域サンプリング手法

厳密には、1 つの画素を通って視点に到達する光線は視点からみてある大きさの方向領域を占める。したがって、視点から光束を追跡することでより正確に輝度を計算できる。光束の形状が異なる以下の 3 手法が提案されている。

- cone tracing 法 [1]

  視点から円錐形の光束を追跡する。円錐の広がりは視点からみた画素の立体角と同程度に選ぶ。通常の光線追跡法と同様に、光束が物体と交差するときには、光束と交差する物体表面の 1 点から正反射方向と屈折方向の光束を追跡する。これらの光束の広がりは入射光束の広がりと反射面の曲率とを考慮して決定される。光束が複数の物体と交差する場合には、交差の面積に比例した寄与率を与えて光束を分割する。

円錐形の光束と物体が交差するか否かは円錐の軸から物体までの距離を求めることで判定できるが、円錐と物体との交差領域を正確に計算することは極めて困難である。このため、実際のインプリメントではサンプリング手法を用いて近似的に交差領域を求めている。

- beam tracing 法 [17]

  この手法は視点と画素の頂点を結んでできる四角錐を追跡する。一般的な光線追跡法は2次曲面で記述される物体まで取り扱うことができるが、beam tracing 法は多面体だけを取り扱う。このように制限することで、1.2.6節で述べるクリッピング手法を用いて四角錘と物体とが交差する領域を正確に求めることができる。

  beam tracing 法では反射光と交差する物体は次のようにして求められる。四角錘の中央を通る光線 (軸光線) と物体との交点を求める。交点から正反射方向へのベクトルを視点から交点へ向かうベクトルと一致させる回転行列を求め、交点を中心に全ての物体を回転する。回転により反射光の進む向きは四角錘の進む向きと同じになる。回転された物体と四角錘との交差を求めれば、反射光が追跡できる。屈折光も同様の方法で追跡される。この手法では反射面を平面に限定しているので、光束の集中や拡散は表現できない。

- pencil tracing 法 [38]

  pencil tracing 法は光束を軸光線と1組の近軸光線とで記述する。近軸光線は軸光線との相対的な関係を記述した4次元ベクトルで定義される。4次元のうち2つの次元は軸光線に垂直な平面と近軸光線との交点座標を与え、残り2次元は近軸光線の進行方向を与える。物体表面での近軸光線の反射と屈折をおのおの 4×4 の行列で定義することにより、光束の集光や拡散現象をシミュレートできる。従って、画素内での面法線方向の変化によるエリアシングを除去できる。

  物体の輪郭部分では近軸光線による近似の誤差が大きくなるので、画素を細かく分割して光束を追跡する必要がある。分割には1.2.3節で述べた適応的スーパーサンプリング手法を用いる。

関数フィルタ

光線追跡法では適応的サンプリングや統計的サンプリングが一般的なため、Zバッファ法のように一律にデジタルフィルタを掛けることはできない。そこで、各サンプル点の輝度値を重み付きで平均する。重みは合計が1となるように正規化しておく。重みをテーブルで与えることもできるが、関数で与えたほうがより精密にアンチ・エ

リアシングできる。重み関数には方向性が表れないように回転対称の円錐関数やガウス関数などを用いる。

1.2.4 スキャンライン法のアンチ・エリアシング

スキャンライン法は走査線を単位として隠れ面消去する [50][46]。スキャンラインアルゴリズムは走査線と物体のエッジとの交点を演算精度の正確さで決定できるので、走査線方向の空間解像度はエリアシングを完全に除去できるほど高い。このため、垂直に近いエッジにはジャギが現れない。エリアシング除去に適した隠れ面消去手法といえる。しかし、走査線と直交する方向では走査線の間隔の空間分解能しかないため、水平に近いエッジにはジャギが現れる。

マルチスキャニング法

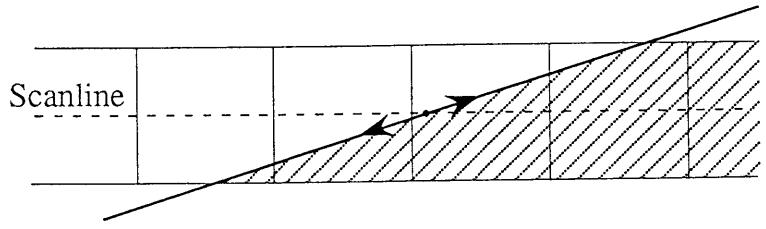
水平に近いエッジのジャギを取り除くには、元々の走査線の間に複数の走査線を挿入して垂直方向の解像度を高める必要がある。区別するため本来の走査線に付加される補助的な走査線をサブ・スキャンライン (sub-scanline) と呼ぶが、隠れ面消去では本来の走査線もサブ・スキャンラインも同等に処理される。サブ・スキャンラインを挿入して 1 画素を複数の場所で走査する手法はマルチスキャニング (multi-scanning) 法 [28] と名付けられている。

処理を簡単にするため、本来の走査線とサブ・スキャンラインを合わせた全走査線は画面全体に等間隔に配置される。画素の輝度値は画素に割り当てられた走査線の輝度値を平均して求められる。輝度の平均値を求める以外は通常のスキャンライン法とまったく同じアルゴリズムで画像を生成できる。

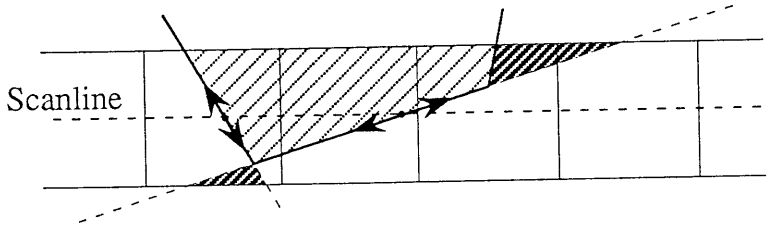
Z バッファ法で画素幅の $1/N$ の空間解像度を得るには、画素の縦横をそれぞれ N 分割しなければならない。このため、N^2 個のサンプル点が必要で、N^2 倍の計算時間がかかる。光線追跡法では適応的にスーパーサンプリングするが、最悪の場合には N^2 倍の計算時間が必要である。スキャンライン法では走査線の上での解像度は十分に高いため、1 画素あたり N 本の走査線を配置すれば縦横ともに画素幅の $1/N$ の空間解像度が得られる。従って N 倍の時間で計算できる。この点から、高精度のアンチ・エリアシングを必要とする場合には、スキャンライン法は Z バッファ法や光線追跡法より高速になる。

スキャンラインに沿った面積推定法

スキャンラインアルゴリズムではスキャンラインとポリゴンエッジとの交点座標を演算精度の正確さで決定できる。図 1-8(a) に示すように 1 画素に 1 本のエッジしか現れない場合には、交点の座標と交点でのエッジの傾きからポリゴンの領域を正しく推



(a) 面積が正しく求められる場合



(b) 誤差が生じる場合

図 1-8: スキャンラインに沿った面積の推定

定できる。ポリゴンは台形または三角形の領域に分けられるので、面積を簡単に計算できる。

物体の頂点を含む, 物体が重なり合う, 物体が細い, などにより 1 画素に複数のエッジが現れる場合には、エッジの傾きからポリゴンの領域を推定しても正しい面積は求められない。図 1-8(b) の例では、濃くハッチングした部分が近似誤差になる。図 1-8(b) の左側の誤差は、エッジ同士の交差を判定し、交わる場合は交点までをポリゴンの領域にする改良 [21] で除去できるが、右側の誤差については一方のエッジしか走査線と交わっていないので誤差を排除することはできない。

スキャンラインの間に小さいポリゴンがあってもスキャンラインと交差しないと検出されない。このため、台形近似で求められたポリゴンの領域の前に検出されないポリゴンが存在する可能性がある。したがって、近似で正しい面積が求められるわけではない。細かい物体や複雑な形状の物体を含む画像では、サブ・スキャンラインで垂直方向の分解能を高めることで面積の近似精度を上げる必要がある。

本手法は、エリアシング除去の精密さ, 処理速度, アルゴリズムの複雑さ, のいずれの点でも通常のスキャンライン法と 1.2.6節で紹介する走査線単位のクリッピング手法との中間に位置する。

線積分フィルタ

スキャンライン法ではスキャンラインと交差するポリゴンの領域は演算精度の正確さで決定できる。そこで、フィルタ関数をスキャンラインに沿って積分することで効果的にエリアシングを除去できる。フィルタ関数に方向依存性の無い回転対称関数を用いると良い結果が得られる。関数が複雑になると積分の計算時間が長くなるので、テーブルを用て高速に積分する手法が提案されている [28]。画素の輝度分解能は多くの場合 RGB 各 256 階調にすぎないので、テーブルを用いても近似誤差はさほど問題にならない。

1.2.5 優先順位法のアンチ・エリアシング

視点から遠い物体から順に画面に重ね書きすると、全ての物体を描画し終えたときには画面上に視点から見える物体だけが残る。優先順位法は、表示したい全ての物体の相互の位置関係を調べ、視点により近い物体により高い順位を与える。順位の低い物体から順に重ね書きすることで隠れ面消去する。

優先順位の決定には 2 通りの手法が考案されている。1 番目の手法は物体が定義された時点で、物体が 2 つ以上の領域に含まれないように空間を適当な平面で分割して、前後関係を木構造やテーブルで保存する。視点が決まったらテーブルや木構造を手繰って優先順位を決定する [23][15][27]。2 番目の手法は視点が与えられる度に物体相互の前後関係を調べて優先順位を決定する [24]。

物体の輪郭に沿った面積の推定法

ジャギは画面に投影された物体の輪郭に発生する。優先順位法は物体を画素や走査線でデジタル化しないので描画するときに物体の外形線の位置情報を利用できる。取り扱う物体を多面体に限定すれば、物体はその表面を構成するポリゴンを単位として描画される。ポリゴンの外形線と画素の境界線との交点から描画するポリゴンが画素を覆う面積を計算することでアンチ・エリアシングできる。ただし、外形線が交差する画素やポリゴンの頂点を含む画素のように複数の外形線が表れる画素では、ポリゴンの外形線を追跡しても正しい面積は求められない。また、物体同士が交差する場合は交線が物体の輪郭線となるが、元々の外形線ではないので追跡処理されない。このため、交線のジャギは除去されない。

簡便にポリゴンの面積を近似する手法 [33] も提案さている。図 1-9に示すように、ポリゴンの外形線を追跡して画素中心から外形線までの符号付き距離を求める。図 1-9の画素 A1 の中心からの垂直距離が求まっているとき画素 B2 からの距離は、(1) 画素を右 (2 列) に移動し、距離に外形線の傾きを足す, (2) 距離が 1/2 よりも大きけれ

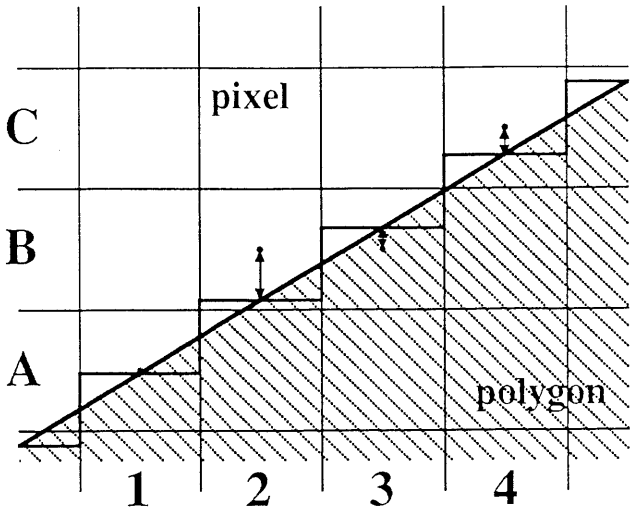


図 1-9: 境界に沿った面積の推定

ば画素を上 (B 行) に移動し距離から 1 を引く, ことで簡単に計算できる。この距離に 1/2 を足した割合だけポリゴンが画素を覆うと推定する。外形線の傾きの絶対値が 1 以上の場合は水平距離を用いて同様に面積を推定する。

この手法は物体の外形線を図 1-9に描かれた階段線で近似する。このため、画素 A1 や B3 のように外形線が画素の左右両端を通過する場合は正しい面積となるが、画素 A2 や B2 のように画素の上端または下端を通過する場合には誤差が生じる。

輪郭線に沿ったフィルタリング

物体の輪郭に沿ってローパスフィルタをかける。具体的には、物体の外形線から一定の範囲にある画素について画素中心からエッジまでの距離を求め、この距離をパラメータとする適当な関数で物体の輝度と背景 (または他の物体) の輝度とを混合する。正方形の範囲にフィルタをかける手法 [14] と画素を中心とする円形の範囲をフィルタ処理する手法 [40] とがある。円形フィルタでは円錐関数や 2 次元ガウス関数などの回転対称関数を重み関数とする。円形フィルタの効果はエッジの方向に依存しないので、正方形フィルタよりアンチ・エリアシング能力が高い。フィルタの範囲を大きくすればエッジは滑らかになるが、空間解像度が低下するため画像の細部が失われる。また、背景も含め 3 つ以上の物体が 1 画素内に存在する場合や小さい物体が存在する場合には正しく表示できない。

1.2.6 領域細分法のアンチ・エリアシング

隠れ面消去とは画面に投影されるすべての物体について前後関係を判定し、画素ごとに視点からどの物体が見えるかを決定することである。しかし、すべての物体の組

み合わせで前後関係を判定すると処理量がきわめて多くなる。また、形状が複雑だと前後判定のアルゴリズムも複雑になる。そこで、比較的単純なアルゴリズムで前後関係が判定できるようになるまで画面を再帰的に分割する。この手法を領域細分法 [47] と呼ぶ。

単に画像を生成するだけなら、画素の大きさまで分割された時点でそれ以上の分割を打ち切って最も手前の物体を選べば良い。しかし、画素に投影される物体の面積を正確に求めるには、1つの領域が1つの物体しか含まないように画面を分割する必要がある。

クリッピング手法

視点に近い物体で後ろの物体をクリッピングして、隠れた部分を取り除く。クリッピングを繰り返すことで、画面を重なりの無い領域に分割できる。クリッピング手法を適用するため、取り扱う物体を多面体に限定する。また、物体の一部が他の物体の内部に含まれることはないと仮定する。以下に多角形をクリッピングする手法を述べる。

- 多角形どうしをクリッピングする手法 [48]

  視点に最も近い多角形で2番目に近い多角形をクリッピングして共通部分を求める。2番目の多角形から共通部分を取り除くと、視点から見える領域が求められる。3番目に視点に近い多角形では、1番近い多角形に含まれる部分を取り除いた後で2番目に近い多角形に含まれる部分を取り除くことで可視領域を決定できる。このように、注目する多角形を手前にある多角形で次々にクリッピングして残りの部分を求めることで画面を分割する。この手法では、あらかじめ視点から近い順に物体をソートしておく必要がある。クリッピングには Weiler-Atherton のアルゴリズム [47] や Weiler のアルゴリズム [48] を用いる。

  この手法は分割数が少なくなるので塗りつぶしの効率が良い。しかし、一般の多角形同士の交差を判断しなければならないのでクリッピングのアルゴリズムが複雑になる。また、クリッピングを何度も繰り返すため処理時間が長くなる。

- 画素で多角形をクリッピングする手法

  元々の手法 [2] は白い紙に黒線を描くようなモノカラー画像のアンチ・エリアシングを目的とするが、物体を視点に近い順にソートすることでカラー画像にも適用できる。

  長方形を単位として画面を分割する。画素を2つに分けないように長方形の4辺を画素の境界線上にとる。領域の中に複数の物体が含まれる場合には領域を

再帰的に分割する。領域が1画素になったらクリッピングで画素を細分割する。画素を最も手前の多角形でクリッピングして面積を求め、面積を重みとしてこの多角形の輝度を画素の輝度に足し込む。画素の輝度値にはあらかじめ0を代入しておく。画素からクリッピングされた領域を取り除く。残された領域を2番目に視点に近い多角形でクリッピングし、面積を重みとして輝度を画素の輝度に足し込む。この処理を残された領域がなくなるまで繰り返す。

この手法では精密な輝度が求められるが、クリッピングを繰り返すため処理時間が極めて長くなる。

- 走査線単位のクリッピング手法 [7]

  はじめに1画素の幅を持つ走査線 (縦は1画素の高さで横は画像の横幅と同じ長方形) で多角形をクリッピングする。多角形を長方形でクリッピングするのに適した Sutherland-Hodgman のアルゴリズム [41] を用いる。次に、クリッピングされた多角形を左から右の順で画素でクリッピングする。1画素に複数の多角形が含まれる場合には、前述の画素で多角形をクリッピングする手法と同じように手前の多角形で後ろの多角形をクリッピングする。多角形を多角形でクリッピングする必要があるので、Wieler-Atherton のアルゴリズム [47] を用いる。

  走査線単位で処理することで画面分割の制御は簡単になる。走査線間の類似性を用いることで処理効率も改善する。しかし、異なるアルゴリズムを用いて2層のクリッピングを行うのでクリッピングアルゴリズムはいっそう複雑になる。また、処理時間も依然として長い。

領域積分フィルタ

クリッピングの結果をフィルタ処理することで、より効果的にアンチ・エリアシングできる。クリッピングは1画素内のポリゴンの領域を正確に求めることができるので、小さなフィルタでも十分な効果をあげられる。このため、画像の解像度をさほど低下させない。

デジタルフィルタで重み付き平均を求めることもできるが、処理量の制限からそれほど分解能の高いフィルタを使用することはできない。これでは、処理コストの高いクリッピングを用いて正確な領域を求めた意味がない。フィルタの重みを関数で定義してポリゴン領域の中で積分すれば、より精密にアンチ・エリアシングできる。

一般的に、画面上の正方形または円形の範囲にフィルタをかけることが多い。重みは計算を簡単にするためにどこでも同じにする (単純平均) か方向により差がでないように軸対称関数を用いる。この組み合わせのなかで、正方形領域の単純平均は面積を

表 1-2: 大きさを持った光源による照明手法

光源の種類	照度計算手法
点光源による近似	等間隔の点光源で近似する手法 照度計算の度に点光源近似する手法
線光源	西田の手法 Poulin の手法
面光源	西田の手法 (拡散反射のみ)

求めることと同じなので簡単に計算できる。軸対称フィルタを円形の範囲にかける場合はテーブル参照を用いて効率よく近似積分できる [13]。

1.2.7 照度計算で生じるエリアシングの除去

照度計算で発生するエリアシングは光源の取扱いを厳密にすることで削減できる。大きさをもつ光源の照明モデルを表 1-2に示す。以下に各手法を概説する。

等間隔の点光源で近似する手法

物体の明るさは、物体表面での光の反射をシミュレートすることで計算できる。コンピュータグラフィックスでは、物体の反射特性を拡散反射成分と鏡面反射成分とに分けることが一般的である。この仮定に基づいて Phong[32],Blinn[3],Cook-Torrance[8] 等の反射モデルが提案されている。どの反射モデルも点光源照明に適用できる。

面光源を点光源の集合で近似 [43] すれば、拡散反射成分と鏡面反射成分の両方を簡単に計算できる。全方向に均等に光を放射する単純な点光源だけでなく、スポットライトのように特定の方向に集中的に光を放射する光源 [45] や複雑な放射特性を持つ光源 [43][29] を使用できる利点もある。しかし、エリアシングの強さは視点・光源・物体相互の位置関係や物体の反射特性により異なるので、どの程度細かく面光源を分割すればエリアシングを実用上問題の無い程度まで削減できるかを画像を生成する前に判断することはきわめて困難である。このため、確実にアンチ・エリアシングするには光源を十分細かく分割しておかなければならなず、計算時間が増大する。近似手法のため必ずアンチ・エリアシングできる保証もない。

照度計算の度に点光源近似する手法

正反射方向からの角度のずれを引き数として反射率のテーブルを作っておく。光源を適切に分割するため、照度計算のたびに図 1-10のように光源をテーブルに投影する。

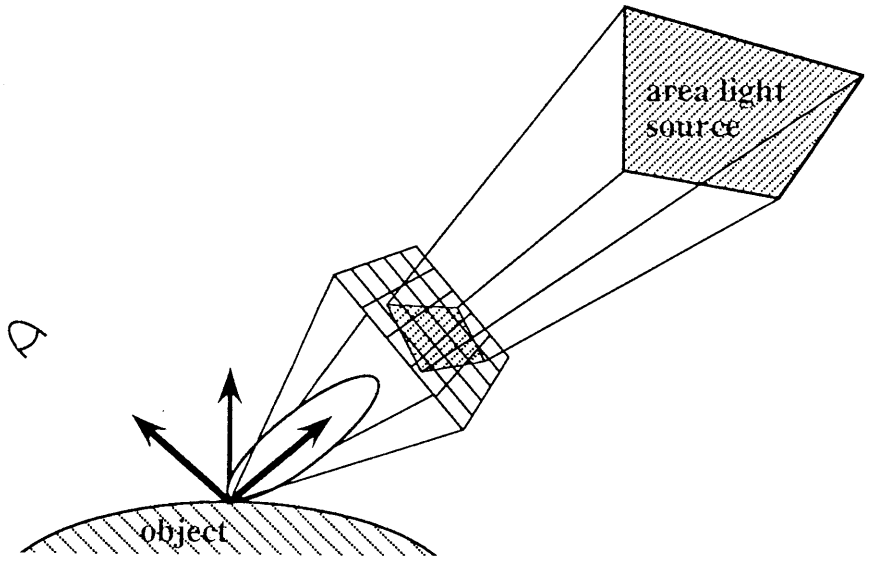


図 1-10: 面光源の適応的な点光源近似

光源に含まれるテーブル要素ごとに計算した輝度を合計して鏡面反射成分を求める [44]。光源の手前に他の物体があるときには、テーブルの場所にZバッファを置いて隠れ面消去することで光源の可視部分を検出する。拡散反射成分はラジオシティ (radiosity) 法 [16][30] で求める。

正反射方向の近傍では反射強度が大きいため輝度値への寄与も大きい。正反射方向を細かく分けるように不等分割することで近似精度を改善する手法も提案されている [39]。しかし、輝度を精密に計算するには大きなテーブルが必要で時間もかかる。

西田の線光源照明法

西田らは蛍光灯のような線光源で照明される場合の輝度計算手法を提案している [29]。この手法では、光源は Lambert の法則に従う放射特性を持つと仮定する。このような光源を完全拡散光源と呼ぶ。

拡散反射成分は、線光源と物体表面の関係を平行・垂直・その他の場合の 3 つに分けて、それぞれの場合の解析的な近似解を得ている。しかし、鏡面反射成分については線光源を点光源の列で近似して計算する。鏡面反射の強度は方向性が強いので、鏡面反射成分がもっぱら照度計算のエリアシングをひきおこす。このため、点光源近似手法に比べさほどエリアシングを削減できない。

Poulin の線光源照明法

Poulin らの手法 [34] は鏡面反射成分と拡散反射成分の両方を解析的な近似式で計算する。近似式の導出には Chebyshev 多項式を用いる。多項式の次数は近似誤差が輝度

値の量子化誤差 (一般的な CG 画像では表示する最大輝度の 1/256) 以下になるように選ぶ。こうして求めた多項式は画像生成に十分な精度を持つので、照度計算のエリアシングを厳密に防ぐことができる。

この手法は、線光源を点光源の列で近似した場合と同様の、放射光の強度が方向に依存しない光源を仮定している。このため、線光源のモデルとしては西田らが用いた完全拡散光源に比べて劣っている。

西田の面光源照明法

この手法は拡散反射成分を解析的に計算する手法である [27]。鏡面反射特性については全く考慮されていない。光源は完全拡散面光源であると仮定する。取り扱う物体を完全拡散物体に限定すれば厳密な照度を計算できる。しかし、ほとんどの物体は拡散反射特性と鏡面反射特性とを合わせ持つので、完全拡散反射物体のみで記述できるシーンは極めて限られる。このため、この手法が適用できる範囲はかなり限定される。

1.3 従来手法の問題点

従来の隠れ面消去手法の中ではクリッピングを用いた手法だけが 1 画素に投影される物体の領域を精密に求めることができる。このため、他のアンチ・エリアシング手法に比べてずっと高品質の画像を確実に生成できる。しかし、クリッピングはアルゴリズムが複雑なため処理に手間がかかる。しかも、物体表面上の視点から見える領域を決定するには繰り返しクリッピングする必要がある。このため、隠れ面消去の計算コストは極めて高い。少し物体形状が複雑になったり重なりが多くなったりしただけで実用的な時間で処理できなくなるので、クリッピングを用いた手法は余り利用されていない。

幾何情報を用いて物体の領域を推定する方法はさほど計算コストをかけずにエリアシングを弱められる。特に物体の外形線に表れるジャギの軽減に威力を発揮する。しかし、隠れ面消去したあとで付加的に面積推定を行うため、アルゴリズムが複雑になる。また、あくまで面積を推定する手法のため、小さい物体が含まれたり 1 画素に多くの物体が投影されるような複雑なシーンでは面積の推定誤りが大きくなる。この推定誤りを小さくするにはスーパーサンプリング手法と組み合わせてサンプリングの間隔を狭くする必要がある。

スーパーサンプリングは最もよく使われるアンチ・エリアシング手法である。スーパーサンプリング手法の多くは 1 画素あたり 1 点をサンプリングする通常手法の単純な拡張であるため、基本的な隠れ面消去アルゴリズムにはほとんど手を加える必要がない。点や線分を単位として処理を行うためアルゴリズムはシンプルである。したがっ

て、インプリメントは比較的簡単である。通常手法で提案されている高速化手法やグラフィックハードウェアをそのまま利用できる利点もある。

スーパーサンプリング手法では1画素あたりのサンプル数に比例してエリアシングを削減できる。しかし、確実にエリアシングを除去できる保証がない。スーパーサンプリング手法を使用するときはこの点が問題になる。これにつにては次節で詳しく論ずる。

現実の物体の稜線や頂点の多くは微小曲面で形成されている。これらの曲面の曲率は極めて大きいので、1画素に投影される領域の中でも面法線の方向が大きく変化する。この法線方向の変化は、稜線に表れるべきハイライトが消失するなどのエリアシングをひきおこす。従来手法の中でも cone tracing 法と pencil tracing 法はこの問題に対処できる。また、スーパーサンプリング手法は1画素の中を細かく分けるので、画素の輝度値に画素内の法線方向の変化が近似的に反映される。ただし、これらのサンプリング手法によるアンチ・エリアシングには確実性がない。法線方向が変化する場合の輝度を厳密に計算できる手法は報告されていない。

物体表面での反射は拡散反射成分と鏡面反射成分とに分けられるが、面光源で照明される場合には拡散反射成分しか解析的に計算できない。鏡面反射成分を計算するには面光源をサンプリングして点光源の集合で近似する必要がある。このため、サンプリング間隔が大きすぎるとエリアシングが発生する。

1.4 スーパーサンプリング手法の限界

スーパーサンプリング手法は最も一般的なアンチ・エリアシング手法であるが、確実にエリアシング除去できる保証がない。この弱点が高品質画像を生成するための手間を増大させ、効率を悪くする。以下にこの問題を詳しく述べる。

コンピュータグラフィックスで一般的に用いられている RGB 各 8bits(256 階調) の画像を仮定する。デジタル画像では、画素の輝度値を1階調以下の誤差で求められれば正しい画像といえる。黒い背景 (輝度 0) の前に白い物体 (輝度 255) が置かれたときには、1画素に投影される物体の面積を画素の面積の 1/256 以下の誤差で求めればよい。このためには、たとえば物体の外形線が垂直方向に画素を横切るときには、水平方向に 256 個のサンプル点を並べる必要がある。外形線が水平方向に横切るときには垂直方向に 256 個並べなければならない。実際には外形線は任意の位置で任意の方向に画素を横切るため、常に誤差を 1/256 以下に保つには1画素を縦横それぞれ 256 等分する必要がある。すなわち、1画素あたり 65,536 個のサンプル点が必要になる。計算時間はサンプル点の数に比例するため、このように膨大な数のサンプル点は現実的でない。したがって、すべての条件下でエリアシングを完全に除去することは事実

上不可能である。

実用的には、画像ごとに、エリアシングの悪影響が顕著に現れないサンプル点の数を適切に選ぶ必要がある。しかし、エリアシングの程度は,物体の位置・大きさ・複雑さ,物体表面の曲率や反射特性,視点の位置,光源の位置と大きさ,などの条件によりさまざまに異なる。したがって、適切なサンプル点の数を画像生成以前に論理的に決定することはきわめて困難である。実際にはサンプル点の数はユーザーが経験を基に決めなければならない。このため、サンプル点が少なすぎるとエリアシングを除去できないので、サンプル点を増やして画像生成をやり直さなければならない。他方、多すぎると無駄な計算時間が増加して効率が悪くなる。

特にアニメーションにおいては、一連のシーンの中でもっとも条件が悪い画像を基準にして1画素あたりのサンプル点の数を決めると、他のほとんどの画像ではサンプル点が過剰になる。このため、計算時間がきわめて長くなる。しかし、サンプル点を少なめに設定するとシーンの一部にエリアシングが発生する。この場合、品質不良の画像を人手で選び出し、サンプル点を追加して再度画像生成しなければならない。画像一枚一枚のサンプル点の数をあらかじめ変えておけば無駄な処理時間を減らせるが、ユーザーに極めて煩雑な作業を強いることになる。したがって、この方法も現実的ではない。スーパーサンプリングを用いる限り画像生成での試行錯誤は避けられず、結果的に作業効率が悪くなる。

光線追跡法のスーパーサンプリング手法では物体の複雑さに応じて適応的にサンプル数を増やすことができる。ただし、離散的なサンプリングの結果からそのあいだに投影される物体の複雑さを推定するので、エリアシングを削減するために必要十分な数のサンプル点を追加できる保証はない。サンプル点追加の基準や最初のサンプル数はユーザーが与えることになるので、画像生成の煩雑さは変わらない。

1.5 高品質像生成のために

エリアシングを根本的に解決するには、画素の輝度値を解析的に計算する画像生成手法が必要である。計算機の演算精度で物体の面積や反射光の強度を計算できるならば、桁落ちを考慮しても256階調の輝度値の決定には十分に正確である。画像のデジタル化によるエリアシングは依然として残るが、ディスプレイの表示限界まで画像品質を高められる可能性がある。

本研究ではエリアシングを厳密に除去するため画像を解析的に生成する手法を確立する。一般的に画像を厳密に生成すると時間がかかる。本研究では実用的な時間で画像生成することを目標にし、そのための高速化手法も検討する。ただし、研究の第一義は画像生成手法を‘常に確実に高品質画像を生成する’ように質的に転換すること

で、‘そこそこの品質の画像を短時間で生成する’ために量的に改良することではない。したがって、品質を犠牲にした生成時間の短縮は考えない。従来手法との比較でも、高品質画像を生成するために必要な時間を比べる。

本論文では以下に示す解析的手法を報告する。

1. 直交スキャンライン法

2章では直交スキャンライン法と名付けた隠れ面消去手法を報告する。隠れ面消去は画像生成に必須の課程であり、最もエリアシングが発生しやすい課程でもある。直交スキャンライン法は、1画素内に含まれるポリゴンの領域を縦横2方向の走査線を用いて解析的に求めることで、エリアシングを厳密に削除する。

ポリゴンの領域はクリッピング手法を用いても正確に求められる。ただし、ポリゴンとポリゴンのクリッピングは画像平面上の2次元処理となるため、アルゴリズムが複雑で計算時間も長くなる。直交スキャンライン法は水平方向と垂直方向の1次元の走査を組み合わせることでポリゴン領域を効率よく求めることができる。

水平走査線は画素の水平境界に配置される。通常のスキャンライン法と同様に画像の左端から右端まで水平走査して隠れ面消去する。垂直走査線は, ポリゴンの頂点, ポリゴンのエッジと水平走査線との交点, エッジ同士の交点, 画像の左右端, を通るように配置され、隣合う水平走査線の間を1画素の高さだけ走査される。垂直走査線の上でも通常のスキャンライン法と同じ手法で隠れ面消去する。この結果、隣合う垂直走査線の間ではポリゴンは重なりの無い台形領域に分割されるので、画素に投影されるポリゴンの面積を演算精度の正確さで計算できる。

効率の良いスキャンラインアルゴリズムを用いるため処理時間はさほど増加しない。直交スキャンライン法の画像生成時間は4 ～ 7本のサブ・スキャンラインを走査するマルチスキャニング法の生成時間とほぼ等しいことを実験で示す。2.7節で報告する画質評価手法を用いて比較すると、直交スキャンライン画像はマルチスキャニング画像に比べずっと高品質である。直交スキャンライン法によりエッジの方向やポリゴンの細さに依存しないアンチ・エリアシングが可能になる。

2.8節では直交スキャンライン法に適したフィルタリング手法を報告する。直交スキャンライン法は1画素内のポリゴンの領域を精密に決定できるので、小さいサイズのフィルタでも効果的にエリアシングを除去できる。

2. 反射強度積分法

3章では画素の輝度値を解析的に計算する反射強度積分法を報告する。厳密な輝度値は反射光の強度を画素内で積分することで求められるが、この積分を直接計算することはきわめて困難である。そこで、反射強度積分法では積分を極座標系に変換したのち近似計算する。極座標変換により積分範囲は画素内に存在する面法線が作る方向領域と単位球面が交差する領域に変換される。積分を簡単にするため、積分範囲を z 軸と単位球面との交点を頂点とする単位球面上の三角形に分割する。各三角領域では被積分関数を Chebyshev 多項式で近似したのち代数積分する。近似誤差を輝度の量子化誤差 (通常の CG 画像では表示する最高の輝度の 1/256) より十分に小さくすることで、精密な近似解を得る。

現実の物体の稜線や頂点には製造しやすくするためや安全のために小さい丸みがつけられることが多い。これらの稜線や頂点を、丸められた稜線・頂点と呼ぶ。丸められた稜線や頂点に代表される曲率の大きい曲面をポリゴンで近似すると、ポリゴンは画素に比べ小さく、面法線の方向は画素内で大きく変化する。このようなポリゴンでも直交スキャンライン法で隠れ面消去したのち反射強度積分法で輝度計算することで精密に表示できる。この手法を精密レンダリング法と名付ける。

精密レンダリング法は丸められた稜線や頂点に現れるハイライトや陰を正確に表示できる。画像生成実験により稜線や頂点に現れるハイライトと陰は物体の実在感を高める上で重要な役割を果たすことを示す。高度のアンチ・エリアシングと正確な輝度計算とを提供する精密レンダリング法は写実的な画像を生成する上で極めて有効な手法となる。

3. 面光源照明法

写実的な画像を生成するには、世の中で一般的な面光源による照明が不可欠である。従来手法では鏡面反射成分はサンプリング手法を用いて計算する以外に方法がないため、エリアシングが発生する恐れがあった。4章で報告する面光源照明法は拡散反射成分と鏡面反射成分の両者を解析的に計算する。

面光源照明法は多角形の完全拡散光源で照明し、入反射のエネルギーを保存するように改良した Phong の反射モデルを用いて輝度計算する。はじめに、視線方向の正反射方向が z 軸となるように極座標変換を行い、光源内での平面積分を球面積分に変換する。輝度計算する点を原点とする単位球面に光源を投影した領域が積分範囲になる。次に、積分範囲を単位球面上の三角形に分割する。分割により、2 つの積分変数の一方は代数積分できる。残りの変数は被積分関数を Chebyshev 多項式で近似したのち代数積分する。

面光源照明法は面光源によるハイライトを正確に表示できる。反射の鋭さを自

由にコントロールできるのでぼんやりとした鏡面反射も表示できる。反射が鋭い場合や光源が物体に近い場合には、従来の点光源で近似する手法よりも短時間で画像生成できる。

4. テーブル積分法

5章では反射強度積分法と面光源照明法とで共通して用いる球面積分を高速に計算するテーブル積分法を報告する。本研究は高品質画像の生成を目的としているので、近似精度を落とすような高速化手法は研究の主旨に反する。テーブル積分法では、近似誤差を輝度の量子化誤差以下に保つため、反射特性関数が急峻に変化する部分では標本点を細かく、緩やかに変化する部分では標本点を粗く配置する適応分割テーブルを用いる。標本点が非等間隔に配置されるので、入力された値と標本点の位置を格納したインデックステーブルとをバイナリサーチで比較することで参照する標本点を決定する。

誤差評価実験により 90×90 適応分割テーブルの近似精度は 10 次 Chebyshev 近似に匹敵することを示す。画像生成実験で 90×90 適応分割テーブルを用いれば十分な品質の画像が生成できることを確認する。テーブルの各要素及びインデックスは 4bytes の記憶領域を使用するので、90×90 テーブル 1 枚は約 3.3kbytes の大きさになる。ただし、鏡面反射の鋭さが同じ物体は同一のテーブルを用いて輝度を計算できるので、画面に現れる全ての物体を輝度計算してもそれほど多くのテーブルは必要ない。

適用分割テーブルを用いることで積分計算に要する時間を 1/10 に短縮できることが示される。画像生成に必要な全時間で比較しても 60 ～ 40% の短縮になる。

以下の各章では上記手法の詳細を報告する。

第 2 章

高精細隠れ面消去法

2.1 序言

本章では直交スキャンライン法(Cross Scanline Algorithm)と名付けた隠れ面消去手法を提案する。直交スキャンライン法は画像を縦横2方向に走査することで1画素に投影される物体の領域を精密に決定する。エリアシングは物体の輝度値を画素に投影される物体の面積で重み付けして足し合わせることで除去できるので、直交スキャンライン法により高度のアンチ・エリアシングが実現される。

視点から画素を通して見える物体を決定する隠れ面消去は画像生成に必須の処理である。1章で述べたように、従来の隠れ面消去手法の多くはサンプリング手法である。Zバッファ法とレイ・トレーシング法は点を単位としてサンプリングするが、単純に画素中心をサンプル点に選ぶとエリアシングが発生する。たとえば、物体の外形線が横切る画素には背景も含め複数の物体が投影される。このため、1画素あたりきわめて多くの点をサンプリングしないとポリゴンの面積を精度良く近似できない。

スキャンライン法は走査線を単位としてサンプリングすることで、物体の外形線が走査線を横切る位置を演算精度の正確さで決定できる。また、走査線間のポリゴンの面積を台形近似で推定することもできる。これらの点から、スキャンライン法はZバッファ法やレイ・トレーシング法に比べアンチ・エリアシングに適している。しかし、走査線と垂直方向の解像度は走査線の間隔で決まってしまうので、1画素あたり多数の走査線を配置することで走査線の間隔を狭めてやらないと面積の近似精度を高めることはできない。

いずれのサンプリング手法もポリゴンの面積を近似的にしか求められないので、サンプル数を増やしてもエリアシングから完全に逃れることはできない。しかも、サンプリング手法の計算量はサンプル数に比例して増加するため、面積を精密に計算するには膨大な処理時間が必要になる。

従来手法でもクリッピングを用いた手法は1画素に投影されるポリゴンの領域を正確に求めることができる。ただし、多角形を多角形でクリッピングするため、アルゴ

リズムが複雑で処理に手間がかかる。画像を生成するにはクリッピングを繰り返して視点から見える物体の領域を決定する必要がある。このため、多少形状が複雑になったり重なりが多くなったりするだけで、計算時間が極めて長くなる。

本章で提案する直交スキャンライン法は、2段階のスキャンライン走査により視点から見える物体の領域を演算精度の正確さで決定する。初めに、画像を画素の水平境界に沿って画像の幅だけ水平走査する。次に、ポリゴンの頂点, ポリゴンエッジ (ポリゴンの外形線) と水平走査線との交点, ポリゴンエッジ同士の交点, 画像の左右端, を通る位置に垂直走査線を配置して、隣接する水平走査線の間を1画素の高さだけ垂直走査する。この結果、隣合う垂直走査線の間ではポリゴンが重なりの無い台形領域に分割される。水平・垂直走査ともにスキャンラインアルゴリズムを用いて隠れ面消去する。スキャンラインアルゴリズムは走査線とポリゴンエッジとの交点座標を演算精度の正確さで計算できるので、画素に投影されるポリゴンの領域も演算精度の正確さで計算できる。

直交スキャンライン法はクリッピング手法と同等の精密なアンチ・エリアシングを提供する。クリッピング手法が画像平面上でのポリゴンの重なりを調べるのに対し、直交スキャンライン法は走査線上での重なりをスキャンラインアルゴリズムを用いて効率良く調べるので、直交スキャンライン法はクリッピング手法に比べはるかに短い時間で隠れ面消去できる。

以下の節では、従来の隠れ面消去手法の問題点、直交スキャンライン法の概要、計算機実験結果について述べる。

2.2 従来の隠れ面消去手法

代表的なアンチ・エリアシング手法を概説し、問題点を指摘する。

2.2.1 スーパーサンプリング手法

代表的な隠れ面消去手法であるZバッファ法とレイ・トレーシング法は、点ごとに物体の前後関係を調べることで隠れ面を消去する。これらの手法では、1画素に多数のサンプリング点を配置して物体の面積を近似的に求めることで、エリアシングを除去する [5][49][9]。面積の近似精度はサンプル点の密度に比例するため、エリアシングを確実に除去するには極めて多くのサンプル点が必要になる。

一般的なCG画像では赤 (R), 緑 (G), 青 (B) の輝度をそれぞれ256階調で量子化して表示する。1画素内ではポリゴンの輝度値は一定であると仮定すると、輝度の計算誤差を輝度の量子化誤差より常に小さくするには画素の大きさの1/256以下の誤差でポリゴンの面積を計算しなければならない。物体の外形線が画素を垂直に横切る場合

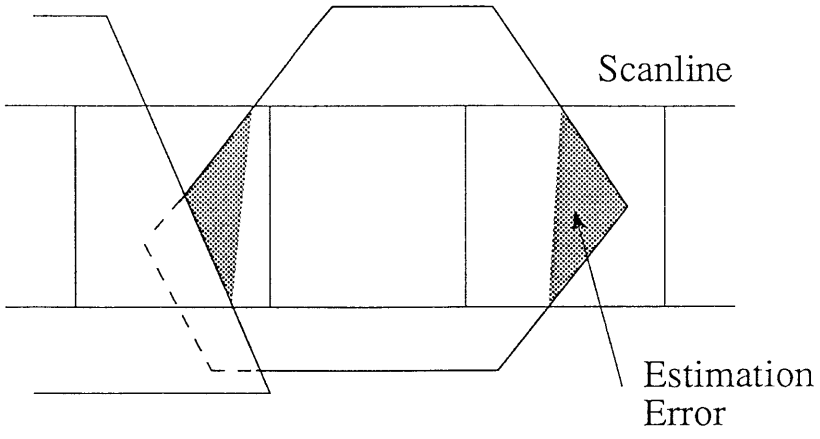


図 2-1: 通常のスキャンライン法による走査

には画素を水平方向に 256 等分しないと要求精度が満たせない。外形線が水平に横切る場合には垂直方向に 256 等分する必要がある。任意の方向の外形線に対して誤差を保証するには、縦横それぞれ 256 に分割しなければならない。すなわち $256 \times 256 = 65,536$ 個のサンプル点が必要になる。このため、スーパーサンプリングで確実にエリアシング除去することは事実上不可能である。

物体の輪郭線の近傍を適応的にサンプリングすることで、サンプル点の数を減らすことができる。ただし、輪郭部であるか否かはサンプリングの結果から判断されるので、物体が小さい場合や細い場合には細かく分割されないことがある。このため、確実にアンチ・エリアシングできる保証がない。

多くの画像では、それほど多数のサンプル点を用いなくても、視覚的に問題にならない程度までエリアシングを弱められる。ただし、エリアシングの程度は物体の複雑さや視点からの距離などの画像生成条件により大きく異なる。このため、常に画像品質を著しく低下させるエリアシングが発生する心配がつきまとう。

2.2.2 マルチスキャニング法

スキャンライン法では物体をポリゴンで記述する。スキャンライン法は、スキャンライン (走査線) とポリゴンエッジ (ポリゴンの輪郭線) との交点を数値計算することで、スキャンライン上の物体の領域を演算精度の正確さで決定できる。2 本のスキャンラインの間を台形近似することで、画素中のポリゴンの面積を推定できる。特に、画素内にポリゴンエッジが 1 本しか現れないときには、ポリゴンの面積を正確に計算できる。これらの理由から、スキャンライン法は Z バッファ法やレイ・トレーシング法に比べてアンチ・エリアシングの点で有利である。しかし、図 2-1のように、スキャンラインの間にエッジの端点が存在する場合やエッジが互いに交差する場合には近似

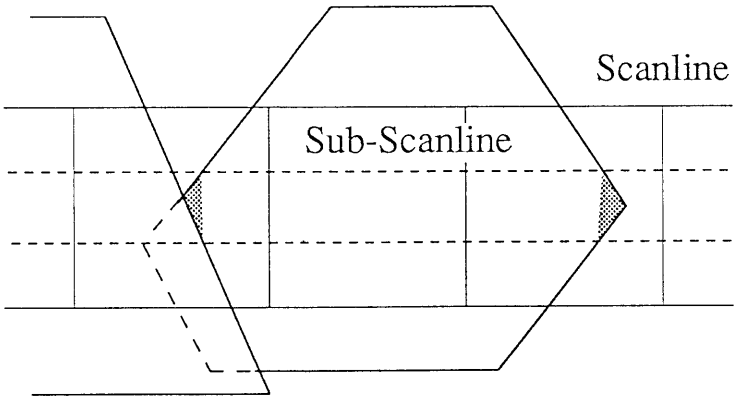


図 2-2: マルチスキャニング法による走査

誤差が大きくなる。

スキャンライン法でのスーパーサンプリング手法をマルチスキャニング法 [28] と呼ぶ。マルチスキャニング法は、エリアシング除去に必要な近似精度を得るため、元々のスキャンラインの間に仮想的なサブ・スキャンラインを挿入して垂直方向のサンプリング間隔を細かくする。サブ・スキャンラインにより 1 画素は複数の場所で走査される。例えば、図 2-2に示すように、1 スキャンラインあたり 2 本のサブスキャンラインを挿入する。これに画素境界に配置される元々のスキャンライン 2 本を加えた 4 本のスキャンラインとポリゴンとの交線を求め、ポリゴンを台形近似する。図 2-2の例では近似誤差の面積を 1/9 にできた。

実際の画像生成においては、以下の 3 点が問題となる。

1. 図 2-2に示されるように、依然として面積の近似誤差が残る。たとえ近似誤差が画素の面積の 5% であっても、256 階調を表示できる標準的なグラフィックディスプレイでは 12 階調の差となる。

2. 垂直分解能が低い。スキャンラインとエッジとの交点座標は演算精度の正確さで求められるので、水平方向の分解能は極めて高い。しかし、垂直方向はサブ・スキャンラインの間隔で決まる分解能しかない。このため、ジャギは水平に近いエッジに選択的に発生する。

3. 計算時間が挿入するサブ・スキャンライン数に比例して増大する。効率よくエリアシングを除去するには、サブ・スキャンライン数を必要最小限にとどめなければならない。しかし、エリアシングの程度は画面の複雑さやエッジの方向により異なるため、あらかじめ最適数を求めておくことは容易ではない。

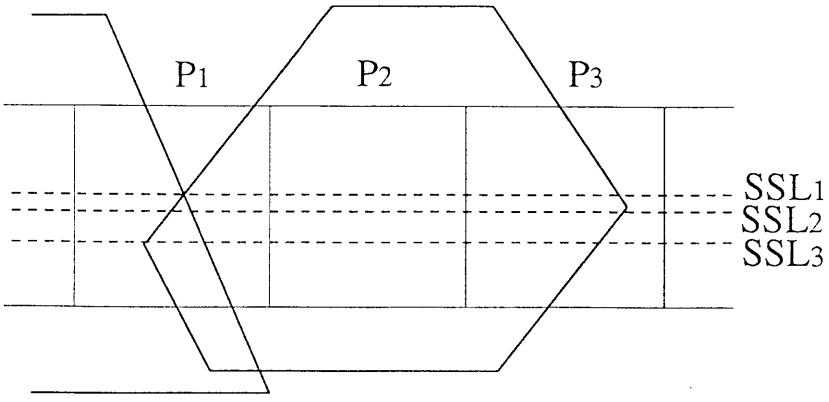


図 2-3: 適応型マルチスキャニング法による走査

2.2.3 クリッピング手法

各画素ごとに、ポリゴンを視点に近い順に並べ替えてからクリッピングを行えば、ポリゴンの占有する領域を精密に決定できる。はじめに画素を最も手前のポリゴンでクリップする。画素の一部でもポリゴンに覆われない場合は、残った部分を2番目のポリゴンでクリップする。この処理を画素の全体がポリゴンで覆われるまで、またはポリゴンが無くなるまで繰り返す。

クリッピング手法は1画素内に投影されるポリゴンの面積を正確に計算できる。このため、確実にアンチ・エリアシングできる。ただし、クリッピングアルゴリズムはインプリメントが複雑で、各画素ごとに独立して処理しなければならないので、処理時間が長くなる。

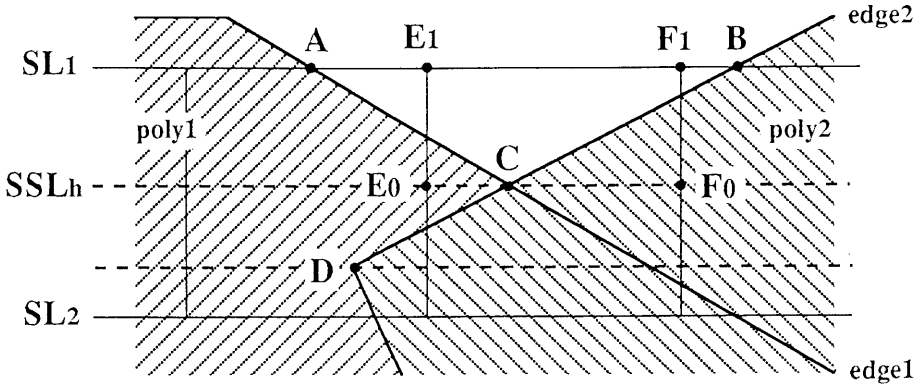
水平方向の画素並びを単位とする高速化手法 [2][7] も考案されてはいるが、処理が複雑になる割にはそれほど計算時間は短縮されない。このため、現実的なシーンではサンプリング手法と比較にならないほど時間がかかり、実用的ではない。

2.3 スキャンライン法の改良

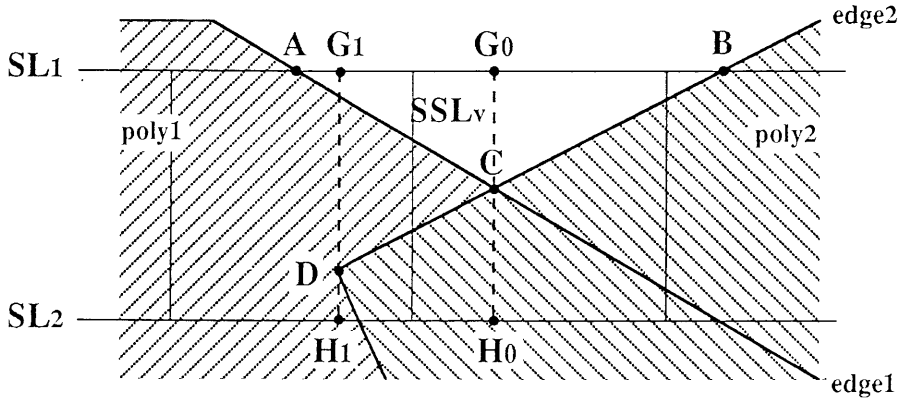
スキャンライン法はZバッファ法やレイトレーシング法に比べてアンチ・エリアシング能力が高い。また、クリッピング手法と比べれば高速である。そこで、スキャンラインアルゴリズムを基に1画素内でポリゴンが占める領域を正確に計算する手法を検討する。

2.3.1 サブ・スキャンラインの適応配置

面積の近似誤差はポリゴンの頂点やエッジ同士の交点で発生する。これらの点を通るようにサブ・スキャンラインを配置してポリゴンを分割すれば、台形則でポリゴン



(a) 水平走査



(b) 垂直走査

図 2-4: 1 画素に限定したサブ・スキャンライン走査

の面積を正確に計算できる。例を図 2-3に示す。この手法を適応型マルチスキャニング法と呼ぶ。

図形が複雑になると分割に必要なサブ・スキャンライン数が膨大になるため、計算量が極めて大きくなる。しかも、サブ・スキャンラインによる分割は、ポリゴンの頂点やエッジの交点を含む画素では必要であるが、他の画素ではかえって邪魔になる。

2.3.2 1 画素に限定したサブ・スキャンライン走査

適応型マルチスキャニング法では無駄な分割が多い。たとえば、図 2-3の中で最も上にあるサブ・スキャンライン SSL_1 による分割は、画素 P_1 内では必要であるが、画素 P_2, P_3 内では不要である。そこで、ポリゴンの頂点やエッジ同士の交点を含む画

素内だけサブ・スキャンラインを走査する改良で、不要な処理を削除できる。

図 2-4の $edge1$ と $edge2$ の交点 C を通るサブ・スキャンラインを例に 1 画素の内部に限定した走査手法を検討すると、

- 図 2-4(a) に示すように、水平方向のサブ・スキャンライン SSL_h を点 E_0 から点 F_0 までの間だけ走査する,
- 図 2-4(b) に示すように、垂直方向にサブ・スキャンライン SSL_v を点 G_0 から点 H_0 まで走査する,

の 2 つの方法が可能である。まず垂直走査から検討する。

2.3.3 垂直走査

図 2-4(b) のように G_0 から H_0 まで SSL_v をスキャンラインアルゴリズムを用いて走査するには、

(1) 点 G_0 で存在するすべてのポリゴン,

(2) 線分 G_0H_0 と交差するポリゴンのエッジ,

を求めておく必要がある。ここで‘点 G_0 で存在するすべてのポリゴン’とは画面上で点 G_0 を含む全てのポリゴンをさす。この中には視点に最も近いポリゴンだけでなく他のポリゴンに覆い隠されるポリゴンも含まれる。図 2-4(b) の点 G_0 はどのポリゴンにも含まれていないので空集合となる。

SL_1 を走査した時点で点 A と点 B との間にはどのポリゴンも存在しないことが求められている。また、点 G_0 が点 A と点 B の間にあることは x 座標値の大小を比較することで容易に判定できる。したがって、(1) は SL_1 の走査結果を保存しておきさえすれば簡単に求めることができる。

(2) は直前の SSL_v 上のエッジから高速に求めることができる。サブ・スキャンラインがポリゴンの頂点とエッジ同士の交点を通る位置に配置される場合には、図 2-4(b) の頂点 D を通る G_1H_1 が直前の SSL_v である。線分 G_0H_0 と交差するエッジのリストは線分 G_1H_1 と交差するエッジのリストから

(v1) G_1G_0 間で SL_1 と交差するエッジ,

(v2) H_1H_0 間で SL_2 と交差するエッジ,

(v3) 長方形 $G_0H_0H_1G_1$ 内に存在するポリゴンの頂点を通るエッジ,

を追加・削除するだけで求めることができる。SL_1 を走査した際に、SL_1 と交差するエッジをその交点の x 座標値でソートしたリストが作られている。このリストを、

SL_1 と SL_2 の間で走査すべき SSL_v の x 座標値で分割することにより、すべての SSL_v について上記 (v1) を求める事ができる。このため、必要となる計算コストはごくわずかである。同様に、SL_2 のエッジリストから (v2) を求める事ができる。(v3) は、SL_1 と SL_2 の間に存在するポリゴンの頂点をその x 座標でソートしたリストを作れば、(v1) と同じ処理で求められる。したがって、(v1)(v2)(v3) を求めるための処理時間は極めて小さい。

2.3.4 水平走査

図 2-4(a) のように SSL_h を E_0 から F_0 まで走査するには、上記 (1)(2) に代わり、

(3) 点 E_0 で存在するすべてのポリゴン,

(4) 線分 E_0F_0 と交差するポリゴンのエッジ,

を求める必要がある。水平走査では、走査する画素がサブ・スキャンライン SSL_h ごとに異なるので、多くの場合、直前のスキャンライン (この場合は SL_1) を参照することになる。一般に SL_1 上の点 E_1 で存在するポリゴンと E_0 で存在するポリゴンとは同じではない。図 2-4(a) でも点 E_1 ではポリゴンが存在しないが E_0 では $poly1$ が存在する。このため、線分 E_1E_0 と交差するエッジを求め、E_1 でのポリゴンから E_0 でのポリゴンを導く必要がある。

また、(2) と同様の手法で (4) を求めるには、上記 (v1)(v2)(v3) に代わり、

(h1) 線分 E_1E_0 と交差するエッジ,

(h2) 線分 F_1F_0 と交差するエッジ,

(h3) 長方形 $E_1F_1F_0E_0$ 内に存在するポリゴンの頂点を通るエッジ

を求めなければならない。(v1)(v2) の場合と異なり、E_1E_0, F_1F_0 は SL_1, SL_2 上にはないので、SL_1 と SL_2 の間に存在するエッジについてあらたに E_1E_0, F_1F_0 との交差を調べ、(h1)(h2) を求める必要がある。このため、(v1)(v2) に比べ多くの処理を必要とする。

2.4 直交スキャンラインアルゴリズム

2.3.3節と 2.3.4節の考察から、垂直走査は水平走査に比べはるかに計算量を削減できる。そこで、縦方向にサブ・スキャンラインを配置する手法を採用する。直交する向きに画像を走査することから、この手法を直交スキャンライン法 (Cross Scanline Algorithm) と名付ける。混同を避けるため、画素の水平境界に配置された従来通りのスキャンラインをHスキャンライン、2本のHスキャンラインの間を、これと垂直に

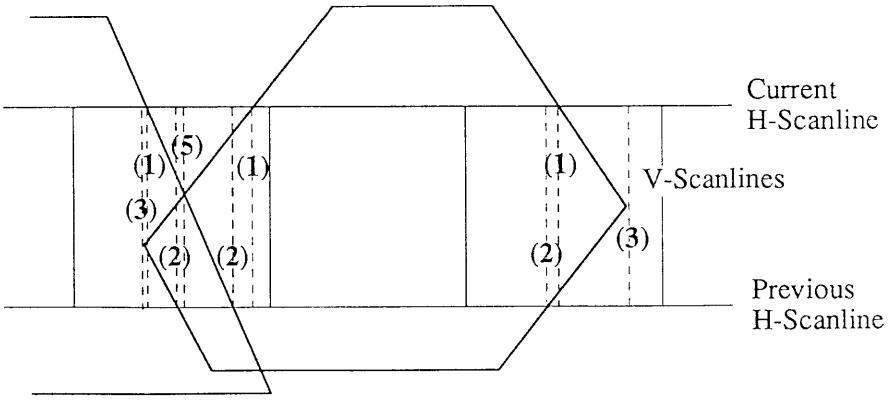


図 2-5: 直交スキャンライン法による走査

1画素の高さだけ走査するサブ・スキャンラインをVスキャンラインと呼ぶことにする。

2.3.3節で示したようにポリゴンの頂点とエッジ同士の交点を通る位置だけに垂直走査線を配置する場合には、画素ごとに含まれるポリゴンの面積を計算するためにエッジとHスキャンライン及びVスキャンラインとの交点からエッジを追跡してポリゴンの領域を決定しなければならない。このため、面積を計算するための処理が複雑になる。

各画素内でのポリゴンの面積を比例配分で簡単に計算するため、直交スキャンラインアルゴリズムではポリゴンの頂点とエッジの交点に加え上下のHスキャンラインとエッジとの交点でもVスキャンラインを走査する。この結果を図2-5に示す。Vスキャンラインの本数は増加するがポリゴンはVスキャンラインにより重なりのない台形または三角形に分割されるので、ポリゴンとVスキャンラインとの交線の長さから面積をたやすく計算できる。画素の垂直境界が隣合うVスキャンラインの間にあっても、画素ごとのポリゴンの面積は比例配分により簡単に計算できる。

以下に直交スキャンライン法の処理の流れを示す。

- [開始]

  (1-1) 透視変換, 裏面の削除, 画面でのクリッピング, 等の前処理を行う。

  (1-2) ポリゴンのエッジを端点の y 座標値でソートしたリスト (Y リスト) を作成する。

  (1-3) Hスキャンラインの位置を画面下端に設定し、交差するエッジのリスト (HE リスト) を作る。

- [Hループ]

(2-1) Hスキャンラインと HE リストに含まれるエッジとの交点座標を計算する。エッジを交点の x 座標値でソートして HC リストを作る。

(2-2) HC リストを用いてHスキャンラインとポリゴンとの交線を求め、視点から見えるポリゴンとそのポリゴンがHスキャンライン上に占める領域を決定する。

(2-3) Hスキャンラインが画面下端なら、［Hスキャンラインの更新］に進む。

(2-4) Vスキャンラインの走査位置を決定する。図2-5に示されるように、

(1) 現在の HC リストに含まれる点、

(2) 1本前の HC リストに含まれる点、

(3) 現在のHスキャンラインと1本前のHスキャンラインとの間に存在するポリゴンの頂点、

(4) 画面の左右端、

を通るVスキャンラインを走査する。

(2-5) Vスキャンラインの位置を画面左端に設定し、Vスキャンラインと交差するエッジのリスト (VE リスト) を作る。

- ［Vループ］

(3-1) VE リストに含まれるエッジとVスキャンラインとの交点を求め、エッジを交点の y 座標値でソートして VC リストを作る。

(3-2) Vスキャンラインを走査し、ポリゴンと交差する領域を求める。

(3-3) Vスキャンラインが画面左端ならば［Vスキャンラインの更新］に進む。

(3-4) 左隣のVスキャンラインとの間で画面上に投影されたエッジが互いに交差すれば、その交点を通るように新たにVスキャンラインを追加する。新たに追加されたVスキャンラインの中で最も左のものを選び、(3-1) にもどる。(図2-5では (5) で示されるVスキャンラインが追加される。)

(3-5) 隣合う2本のVスキャンラインの間のポリゴンの領域を求める。この間ではエッジが交差しないので、領域は台形か三角形になる。各画素に含まれるポリゴンの面積を比例分配により計算する。

(3-6) Vスキャンラインが画面右端ならば［Hスキャンラインの更新］に進む。

- ［Vスキャンラインの更新］

(4-1) Vスキャンラインを右隣の走査位置に移動する。

(4-2) この位置でVスキャンライン上に端点を持つエッジやHスキャンラインと交差するエッジをVEリストに追加または削除する。

(4-3) [Vループ] にもどる。

- [Hスキャンラインの更新]

(5-1) Hスキャンラインが画面上端ならば [終了] 。

(5-2) Hスキャンラインの位置を1つ上の画素境界に移動。

(5-3) Yリストを用いてHEリストにエッジを追加または削除する。

(5-4) [Hループに] もどる。

- [終了]

Hループは、スキャンラインを画素境界に配置する以外、通常のスキャンラインアルゴリズムと全く同じである。Vループでは、Vスキャンラインが(2-4)の条件を満たす場所に配置されるため、走査間隔が不定になる。また、(3-4)においてVスキャンラインが追加されることがあるので、Vスキャンラインの走査位置はxが増大する方向にのみ移動するとは限らない。しかし、これらの点をのぞけば、Vスキャンライン走査も通常のスキャンライン走査と同じである。したがって、Hスキャンライン走査とVスキャンライン走査に共通する部分をサブルーチン化しておけば、プログラム量はそれほど増大しない。

2.5 ポリゴン同士の交線のアンチ・エリアシング

2.4節で示したアルゴリズムは画素がエッジのみで領域分割されることを前提としている。物体がポリゴンで記述されているときには、ポリゴンが他のポリゴンの一部を隠す場合も含め、この前提条件が成り立つ。しかし、ポリゴンが他のポリゴンを貫通する場合には、ポリゴン同士の交線で画素が分割されることがあるので、上記の前提条件が成立しない。

ポリゴン同士が交差しないように前処理でポリゴンを分割することは、計算量を増大させるため現実的ではない。そこで、次のようにアルゴリズムを改良する。2.4節のアルゴリズムの(2-2)と(3-2)でポリゴン同士の交差が見つかった場合にのみ、交線でポリゴンを分割するとともに、交線を仮想的なエッジとしてエッジリストに追加する。仮想エッジを加えることで、画素は常にエッジで分割されることになる。

図2-6に示すように、各1組のH, VスキャンラインHS_1, HS_2, VS_1, VS_2で区切られる矩形領域を考える。実線で示される真のポリゴンエッジの場合、ある点でエッジが領域の境界と交差するならばそのエッジは他の点でも領域の境界と交差する。しか

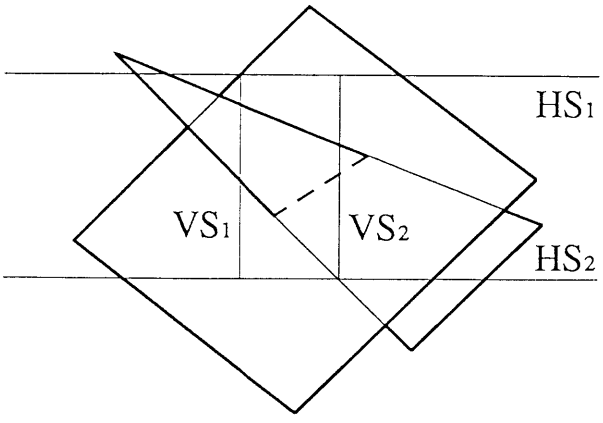


図 2-6: ポリゴン交線の作る仮想エッジ

し、破線で示される仮想エッジでは、図2-6のようにこの性質が成立しない場合がある。そこで、仮想エッジと矩形との交点が1つしか見つからない場合には、仮想エッジを延長してもう1つの交点を求め、その位置もリストに追加する。このような処理を導入すれば、仮想エッジも通常のエッジと同様に扱うことができる。ただし、仮想エッジはポリゴンの輪郭線ではないので、スキャンラインが横切ってもポリゴンの追加・削除を行なわない。

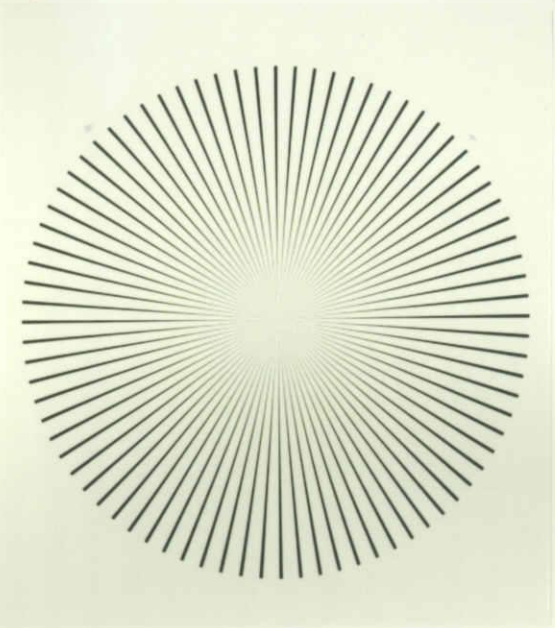
2.6 画像生成実験

本手法の有効性を検証するために、種々の画像を生成した。実験には浮動小数点演算機構付きの VAX8840 を用いた。本節の実験では、隠れ面消去アルゴリズム自体のアンチ・エリアシング能力を比較するため、フィルタ処理を行わない。フィルタリング手法は 2.8節で議論する。

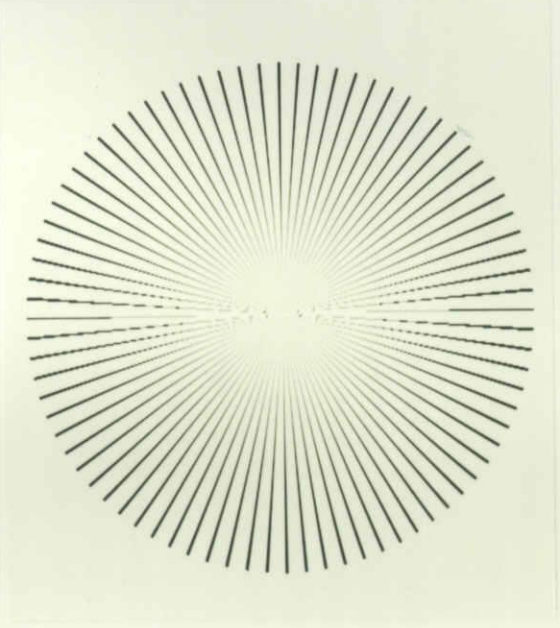
2.6.1 方向特性

エッジの方向によるジャギのちがいを調べるために、白い背景の前に黒く細長い三角形を放射状に並べて図2-7に示すパターンを表示した。図2-7(a) は直交スキャンライン法により生成した画像で、図2-7(b) は通常のスキャンライン法 (サブ・スキャンライン無し) で作成した画像である。図2-7(a) の生成には (b) の約2.5倍の時間がかかるが、従来のマルチスキャニング法で同じ時間をかけてもこの品質は得られない。直交スキャンライン法とマルチスキャニング法とを比較するため、2.6.4節で画像生成時間を測定し、2.7節で画質を評価する。

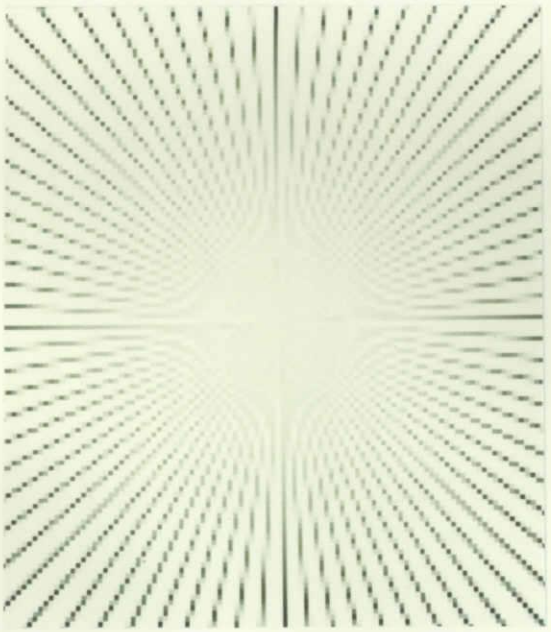
図2-7(c) と (d) は (a) と (b) の中心部を縦横各4倍に拡大したものである。通常のスキャンライン走査では水平エッジでジャギが著しく目立つのに対し、直交スキャンライン法ではどの方向のエッジでも目だったジャギが発生しない。この実験結果は、



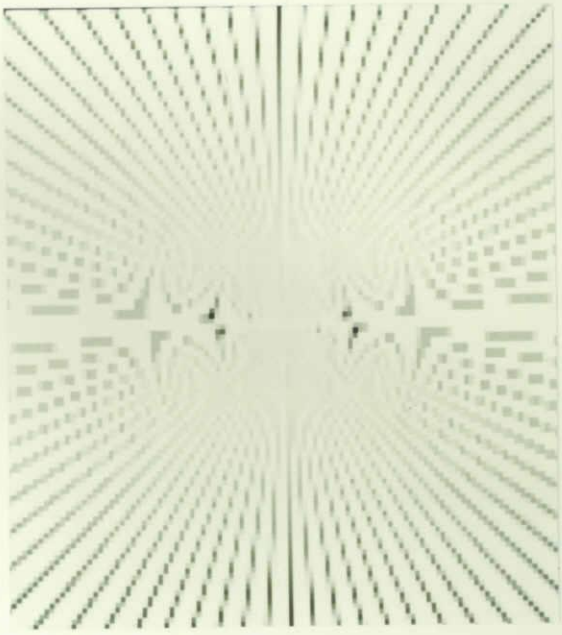
(a) 直交スキャンライン法



(b) 通常のスキャンライン法

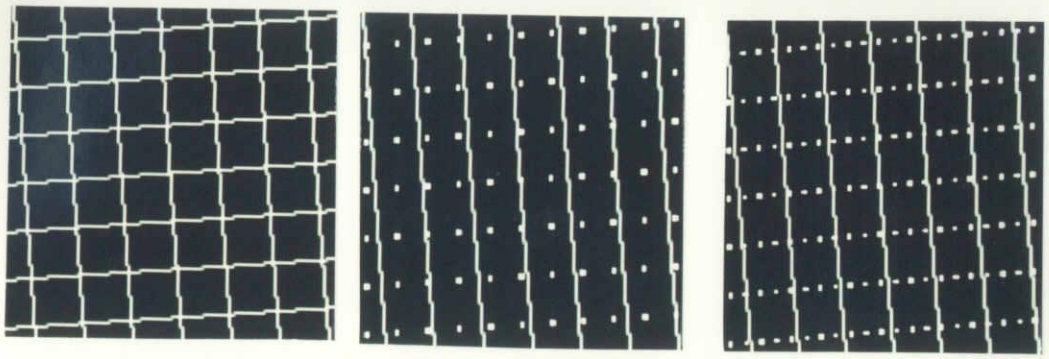


(c) (a) の中央部を 4 倍に拡大

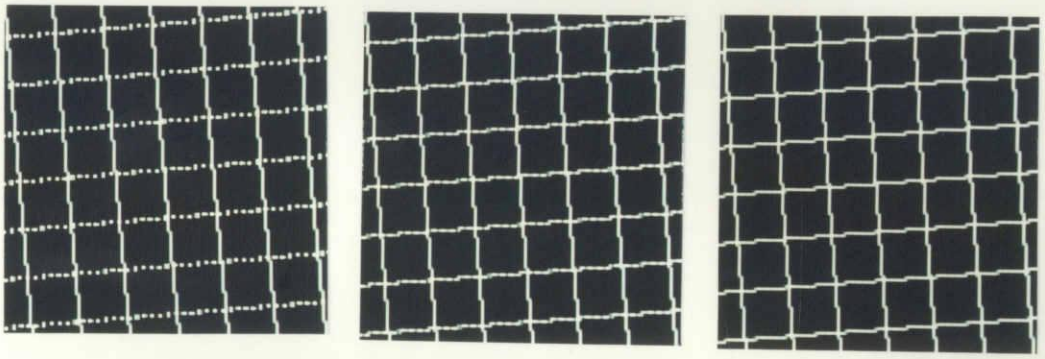


(d) (b) の中央部を 4 倍に拡大

図 2-7: 放射状パターン



(a)Cross Scanline　　(b)S/H=1　　(c)S/H=2



(d)S/H=4　　(e)S/H=8　　(f)S/H=16

S/H: 1画素あたりのスキャンライン数

図 2-8: 傾き5度のメッシュパターン

直交スキャンライン法がエッジの方向に依存しないアンチ・エリアシングを実現することを示している。

2.6.2 細線表示

直交スキャンライン法を用いるときわめて細い線分も適正に表示できる。この特性を確認するため、細い線分で構成されたメッシュを隠面消去した。画像上に投影される線分の幅と間隔はそれぞれ 0.05 画素, 20 画素程度である。

デジタル画像は画素を最小単位とするため、0.05 画素幅の線分も 1 画素の太さで表示される。これは原理的に避けられないので、線分が連続で、かつ、線分の明るさが方向によらないとき適正に表示されていると判断する。

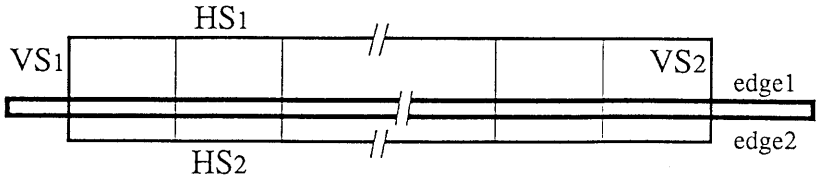


図 2-9: 水平方向に細長い矩形の走査

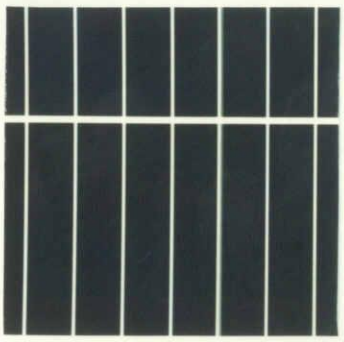
図 2-8はメッシュが x 軸から 5 度傾いた場合である。図 2-8(a) は直交スキャンライン法による結果で、縦線と横線が共に連続で面積に比例した明るさで表示される。したがって、適正な表示と言える。一方、図 2-8(b) から (f) は 1 画素あたりのスキャンライン (元々のスキャンライン＋サブ・スキャンライン) の本数を 1, 2, 4, 8, 16 と変えて画像生成実験を行った結果である。スキャンライン数が少ないと横線が切れ切れに表示される。これは、スキャンラインの間隔で決まる垂直分解能が横線を連続的に表示するには低すぎるからである。実験結果から、この条件で横線が連続して表示されるには 1 画素あたり 8 本以上のスキャンラインが必要であった。この場合でも縦線と横線の輝度が明らかに違っており、適正な表示とはいえない。

次に、メッシュが完全に xy 軸と一致する場合を調べた。メッシュの縦線は通常のスキャンライン法でも問題なく表示できる。メッシュの横線を表示する場合の問題点を図 2-9を用いて説明する。この図に示されるように横線は極めて細長い矩形として定義される。その高さは 1 画素の高さに比べ十分に小さいので、多くの場合は水平走査線 (図 2-9では HS_1 と HS_2) の間に入ってしまう。このため、通常のスキャンライン法では横線が表示されない。

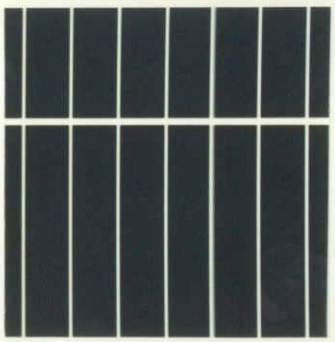
直交スキャンライン法でもHスキャンラインは矩形と交差しないので、P.38に示したアルゴリズムの (2-1) で作成する HC リストは HS_1 でも HS_2 でも空集合となる。また、HS_1 と HS_2 の間にはポリゴンの頂点も含まれない。したがって、アルゴリズム (2-4)(P.38) から、画面の左右端に配置されるVスキャンライン VS_1 と VS_2 だけが走査される。しかし、矩形の外形線はYリスト (P.37, アルゴリズム (1-2)) に含まれるので、HS_1 と HS_2 の間で作成する VE リスト (P.38, アルゴリズム (2-5)) には図 2-9に $edge1$ と $edge2$ で示されるポリゴンエッジが含まれている。したがって、画面の左右端でVスキャンラインを走査するとVスキャンラインと $edge1$, $edge2$ との交差から矩形の領域が正確に求められる。このように、直交スキャンライン法はVスキャンライン走査により横線を正しく表示できる。

xy 軸と一致するメッシュの表示例を図 2-10に示す。図 2-10(a) は直交スキャンライン法による画像で、縦線と横線が同じ輝度で正しく表示される。図 2-10(b) から (g) は

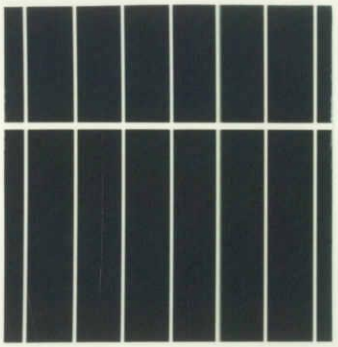
(a)Cross Scanline



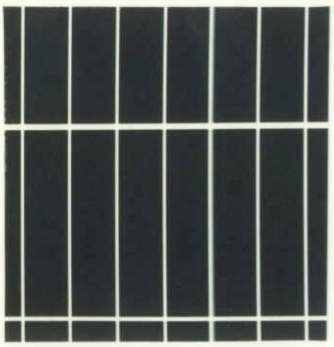
(b)S/H=1



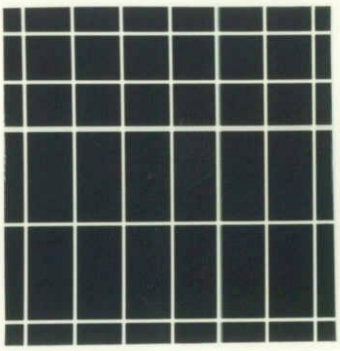
(c)S/H=2



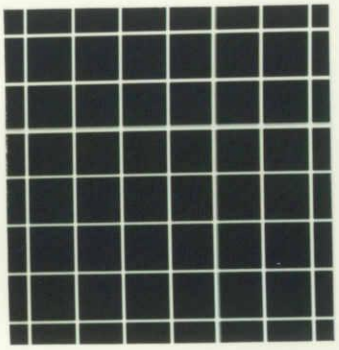
(d)S/H=4



(e)S/H=8



(f)S/H=16



(g)S/H=32

S/H: 1 画素あたりのスキャンライン数

図 2-10: xy 軸と平行なメッシュパターン

1画素あたりのスキャンライン数を変えて画像生成実験を行った結果である。1画素あたり8本のスキャンラインを用いてもサンプリング間隔はメッシュを構成する線分より広い。このため、スキャンラインの間に入った横線は表示されない。横線が表示される場合は縦線より明るくなる。これは、垂直方向の分解能が1/8画素であるため、横線がスキャンラインと交差する場合には1/8画素の面積を持つと判断されるからである。他方、縦線では正しく1/20画素の面積が求まる。この結果、縦線と横線で輝度が異なる。これがマルチスキャニング法の原理的な限界である。

2.6.3 精細な建造物の表示

細い線分を含む建造物の例として、図2-11に示す吊り橋の画像を直交スキャンライン法で生成した。図2-11(a)は全景で(b)はその一部を縦横各4倍に拡大表示したものである。橋を吊るワイヤーは階段状ではあるが連続した線として表示される。2.8節で述べるフィルタ処理を行えば滑らかに表示できる。図2-12(a)と(b)は図2-11(b)と同じ部分をマルチスキャニング法で生成した画像である。図2-12(a)はサブ・スキャンラインを挿入していないため、ワイヤーが点列として表示されてしまう。図2-12(b)は8倍マルチスキャニング画像であるが、依然として一部のワイヤーは点列として表示される。このように、直交スキャンライン法は細いポリゴンを含む物体の表示に効果的である。

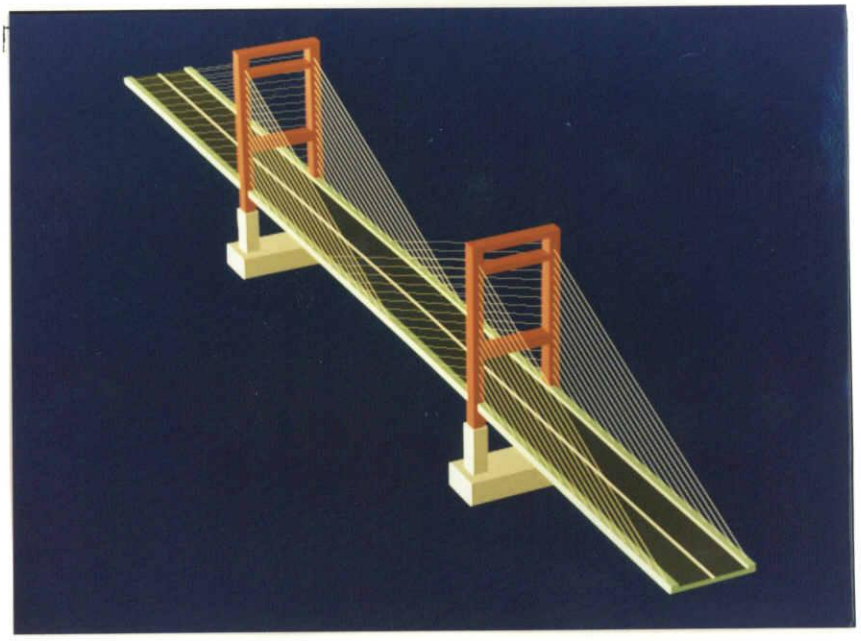
実際の吊り橋では、全長数100メートルの橋でも10センチ程度のワイヤーが張られている。また、数10メートルのビルディングでも窓枠はせいぜい数センチしかない。ワイヤーや窓枠は全体の大きさに比べ十分に小さいが、これらを省略した画像は実物とは大きく異なる。建築物では大きさが数千、数万倍も異なる部品がごく普通に使われる。景観シミュレーションでCG画像を活用するには細かい物体を表示する技術が必須である。

図2-13, 図2-14, 図2-15も直交スキャンライン法で生成した画像である。これらの例も直交スキャンライン法の画像品質の高さを示している。

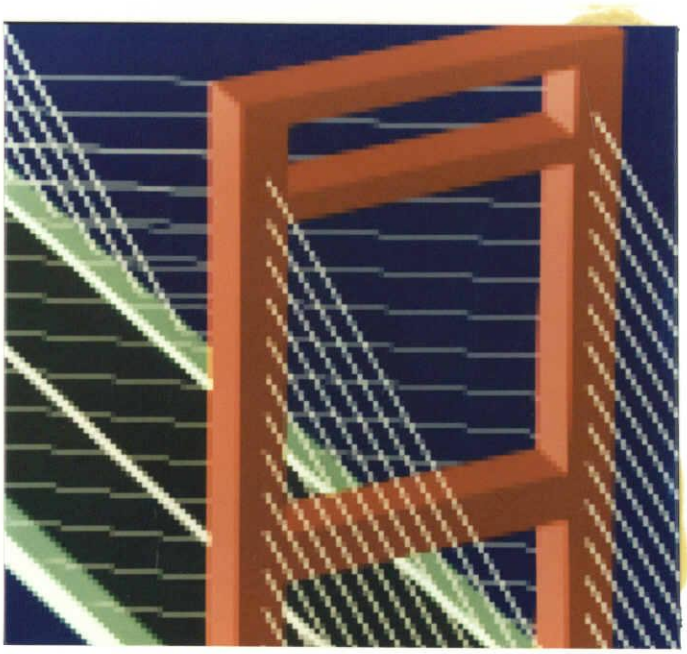
2.6.4 画像生成時間の比較

直交スキャンライン法とマルチスキャニング法との画像生成時間を比較して、直交スキャンライン法と同じ計算時間で走査できるサブ・スキャンライン数(N_{ET}：時間等価サブ・スキャンライン数)を実験的に求める。実験に用いた画像は図2-7(*Radial pattern*：2.6.1節で用いた放射状パターン)と図2-13(*Buildings*), 図2-14(*Three balls*), 図2-15(*Seven balls*)である。各画像に含まれるポリゴンの数を表2-1に示す。生成する画像は大きさ512 × 512画素で輝度はRGBそれぞれ0 ～ 255の値をとる。

4種のデータについて直交スキャンライン法の画像生成時間を測定した。また、サ

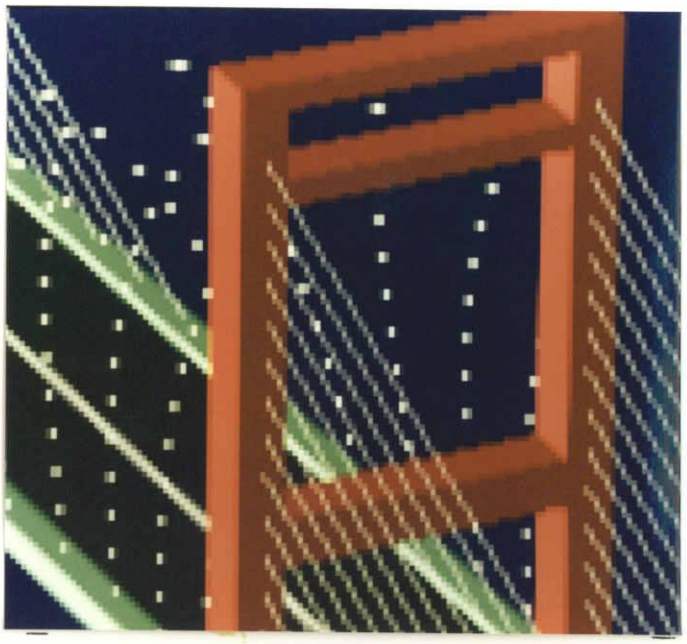


(a) 全景

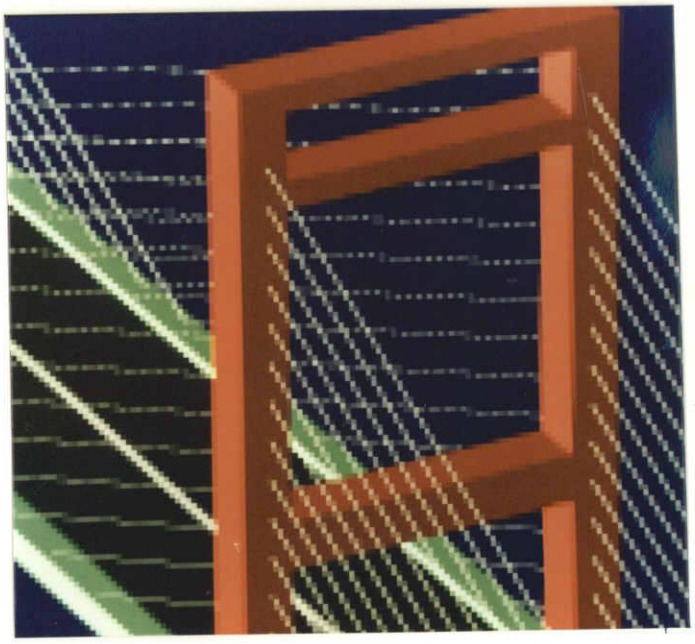


(b) (a) の一部を 4 倍に拡大

図 2-11: 直交スキャンライン法で生成した吊り橋



(a) サブ・スキャンライン無し (通常のスキャンライン法)



(b) 8倍マルチスキャニング

図 2-12: マルチスキャニング法で生成した吊り橋の一部 (4 倍に拡大表示)

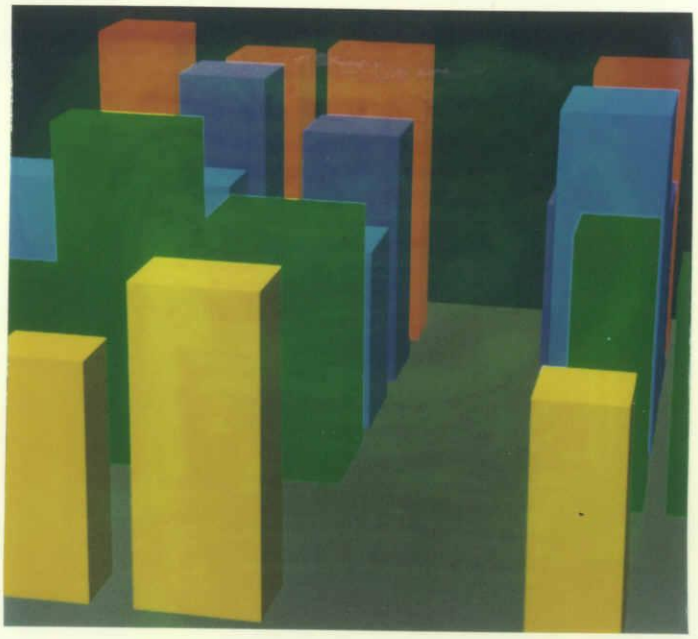


図 2-13: 実験画像： *Buildings*

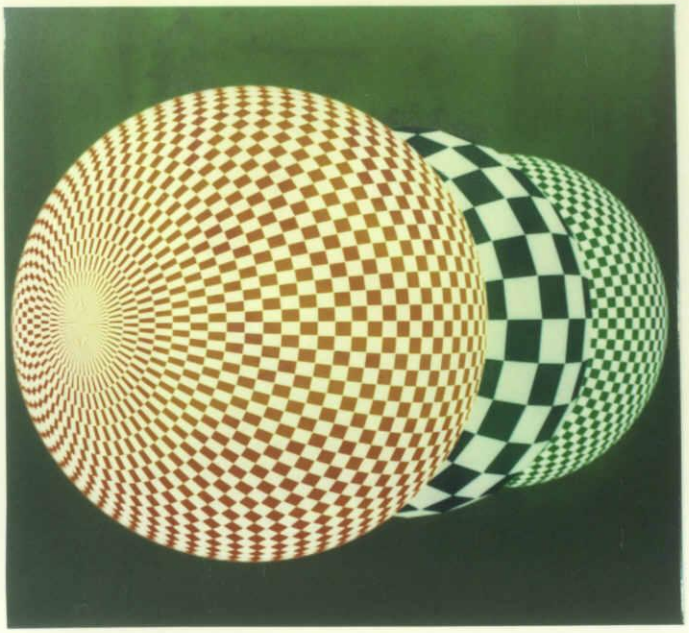


図 2-14: 実験画像： *Three balls*

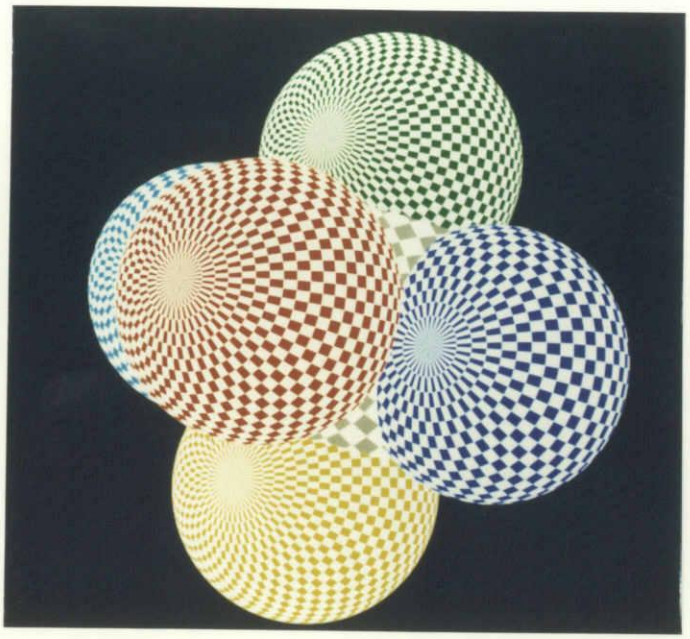


図 2-15: 実験画像： *Seven balls*

表 2-1: 実験画像のポリゴン数

実験画像	データ名	ポリゴン数	N_{ET}
図 2-7 (P.41)	*Radial pattern*	80	4
図 2-13 (P.48)	*Buildings*	151	4
図 2-14 (P.48)	*Three balls*	10450	6
図 2-15 (P.49)	*Seven balls*	16150	7

N_{ET}：時間等価サブ・スキャンライン数

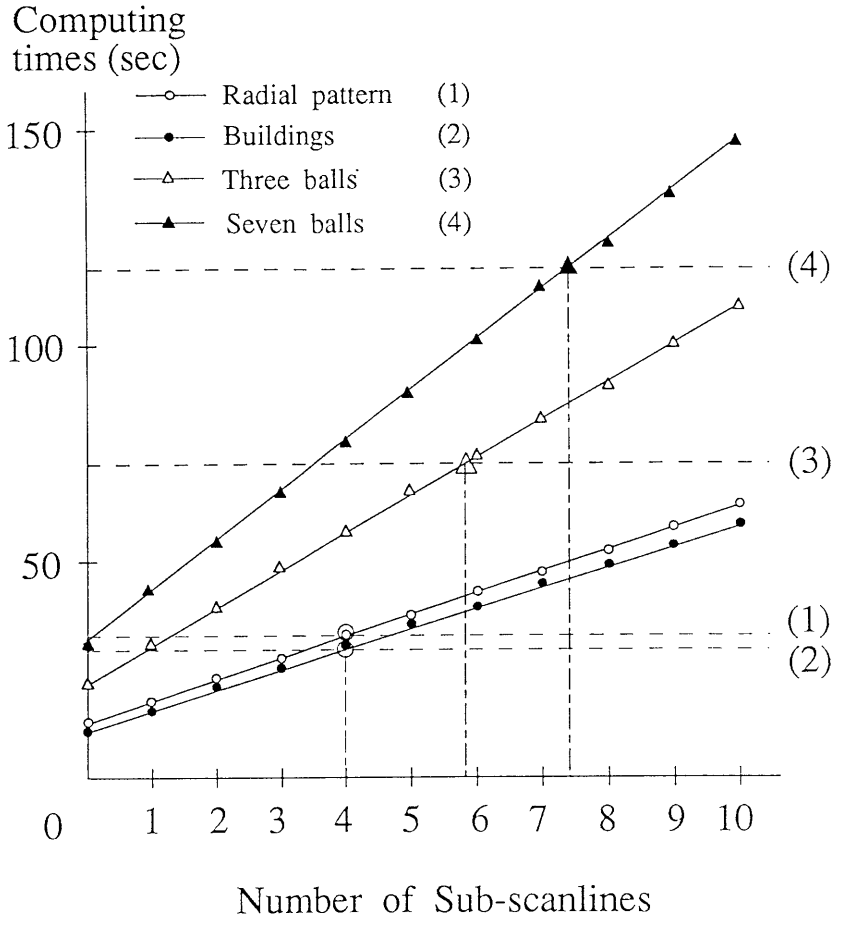

Computing
times (sec)
Radial pattern (1)
Buildings (2)
Three balls (3)
Seven balls (4)
150
100
50
0
(4)
(3)
(1)
(2)
1 2 3 4 5 6 7 8 9 10
Number of Sub-scanlines

図 2-16: 画像生成時間

ブ・スキャンライン数を0から10まで変えてマルチスキャニング法の生成時間を測定した。図2-16の斜め線がマルチスキャニング法による生成時間を、水平線が直交スキャンライン法による生成時間を示す。これらの交点から定まる時間等価サブ・スキャンライン数N_{ET}を表2-1に示す。この結果から、直交スキャンライン法は4～7本のサブ・スキャンラインを挿入するマルチスキャニング法と同程度の計算時間で画像生成できることがわかる。

マルチ・スキャニング法では、サブ・スキャンライン数に比例して品質が向上するが、計算時間も同様に増大する。直交スキャンライン法は通常のスキャンライン法の数倍の時間を要するが、各画素はそこに含まれるポリゴンの面積に比例した精密な輝度値を持つ。直交スキャンライン法の画像はサブ・スキャンライン数を極限まで増やしたときのマルチスキャニング画像と一致する。したがって、どちらの手法が短時間で画像生成できるかはどの程度の画質で満足するかにより異なる。すなわち、多少のエリアシングを問題としないならばマルチ・スキャニング法が速いが、厳密なエリアシング除去を必要とする場合には直交スキャンライン法がはるかに高速である。

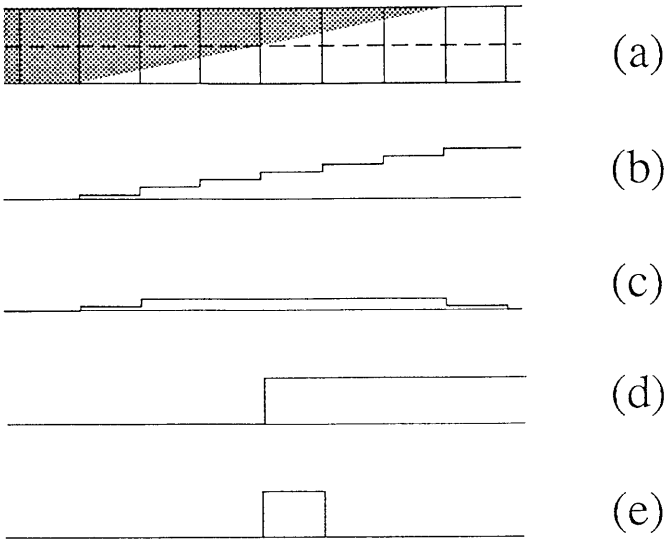
マルチ・スキャニング法の問題点は、要求される画像品質を満足するサブ・スキャンライン数が物体の複雑さや視点の位置により異なる点にある。最適なサブ・スキャンライン数をあらかじめ決めておくことができないので、サブ・スキャンラインを必要以上に挿入するか、本数不足のため再度画像生成しなければならないかのどちらかになることが多い。直交スキャンライン法では常に最良の画質が得られるため、このような無駄な処理時間や判断のわずらわしさは生じない。この点でも直交スキャンライン法が優れている。

2.7 画質評価

ジャギの発生メカニズムに基づき画質の評価を行う。初めに、ジャギの発生メカニズムを考察し、評価尺度を定める。次に、この尺度を用いて、直交スキャンライン法で生成した画像をN_{ET}本のサブ・スキャンラインを挿入したマルチスキャニング法の画像と比較する。N_{ET}は2.6.4節で実験的に求めた直交スキャンライン法と同じ計算時間で走査できるサブ・スキャンライン数である。

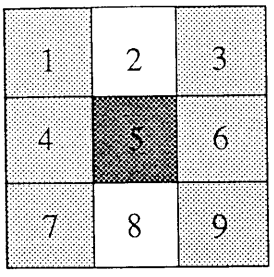
2.7.1 ジャギ発生のメカニズム

画素は縦横の格子状に並んでいるため、ジャギは格子と同じ方向のエッジに発生しやすい。この仕組みを図2-17で説明する。図2-17(a)のようにポリゴンを配置したとき、各画素に投影されるポリゴンの面積を正しく反映した輝度値は図2-17(b)で示され、その差分は図2-17(c)となる。しかし、画素中央を通るスキャンラインだけから

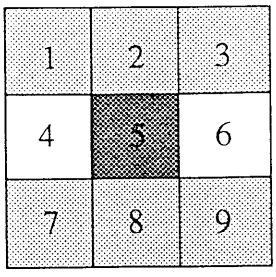


(a) ポリゴンの配置,
(b) 面積に比例した輝度値,
(c) (b) の差分,
(d) 点線の位置を走査した場合の輝度,
(e) (d) の差分

図 2-17: ジャギの検出



(a) 左右連続



(b) 上下連続

図 2-18: 連続性の定義

求めた輝度値とその差分は、図 2-17(d) と (e) である。図 2-17(e) で値が大きく変化する画素にジャギが現れる。

2.7.2 画質評価尺度

以上の考察を元に、評価式を決定し、評価用画像を生成する。元の画像を $I(i,x,y)$ と記す。パラメータ i は R(赤), G(緑), B(青) のいずれかを、x,y は画像上の画素の位置を示す。水平方向の差分を

$$D_x(x,y) = \sum_{i=R,G,B} |I(i,x,y) - I(i,x-1,y)| \tag{2.1}$$

で定義し、垂直方向の差分を

$$D_y(x,y) = \sum_{i=R,G,B} |I(i,x,y) - I(i,x,y-1)| \tag{2.2}$$

で定義する。差分画像からジャギを検出するには、水平方向でも垂直方向でも同じ式を用いる。そこで、以下では、$D_d(x,y)$ と書いて $D_x(x,y)$ または $D_y(x,y)$ を表す。

孤立した大きな輝度変化がジャギの原因となる。そこで、差分画像 $D_d(x,y)$ から孤立点を取り出す。差分画像 $D_d(x,y)$ の中の注目する画素とその周囲の画素に図 2-18に示す番号をつける。はじめに、注目する画素 (図 2-18の 5 番の画素) での輝度変化の連続性を定義する。輝度変化が左右に連続ならば、画素 3, 6, 9 の値の少なくとも 1 つは画素 5 の値に近くなければならない。また、画素 1, 4, 7 の値の少なくとも 1 つも画素 5 の値に近くなければならない。同様に、上下方向に連続ならば、画素 1, 2, 3 の 1 つと画素 7, 8, 9 の 1 つは画素 5 に近い値を持つ。したがって、$D_d(x,y)$ の点 (x,y) における連続性 $C_d(x,y)$ は次式で定義できる。

$C_d(x,y) =$

$$
\begin{aligned}
\text{TRUE} \quad : if \quad & ((\ max[D_d(x+j,y-1)]_{j=-1,1} \geq D_d(x,y) \times \alpha \\
& \quad and \ \ max[D_d(x+j,y+1)]_{j=-1,1} \geq D_d(x,y) \times \alpha) \quad \text{(vertical)} \\
& \quad or \\
& \quad (\ max[D_d(x-1,y+j)]_{j=-1,1} \geq D_d(x,y) \times \alpha \\
& \quad and \ \ max[D_d(x+1,y+j)]_{j=-1,1} \geq D_d(x,y) \times \alpha) \quad \text{(horizontal)} \\
\text{FALSE} \quad & : \text{otherwise}
\end{aligned}
\tag{2.3}
$$

式中の添え字 d は x と y を取り得るが、式中の全ての d は同一の記号を選ぶ。また、$C_x(x,y)$ は水平方向の差分画像に対する連続性を、$C_y(x,y)$ は垂直方向の差分画像に対する連続性を与える。

次に、画像 $D_d(x,y)$ の中の孤立点を取り出してジャギポイント画像 $JP_d(x,y)$ を作る。孤立点では式2.3は偽となるので、$JP_d(x,y)$ は

$$
\begin{aligned}
if \ \ & D_d(x,y) \geq \beta \ \ and \ \ C_d(x,y) = \text{FALSE} \\
& \quad then \ \ JP_d(x,y) = white \quad \text{(jaggy point)} \\
& \quad else \ \ JP_d(x,y) = black
\end{aligned}
\tag{2.4}
$$

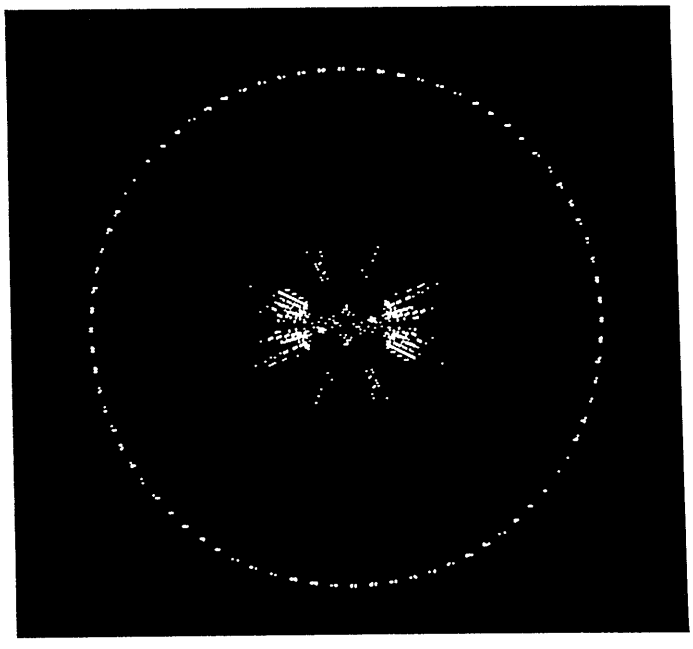
で定義できる。画像 $JP_d(x,y)$ 中の白点がジャギの発生を示す。式2.3、式2.4中の α と β は、$0 < \alpha < 1, 0 < \beta$ を定義域に持つパラメータである。

2.7.3 画質評価実験

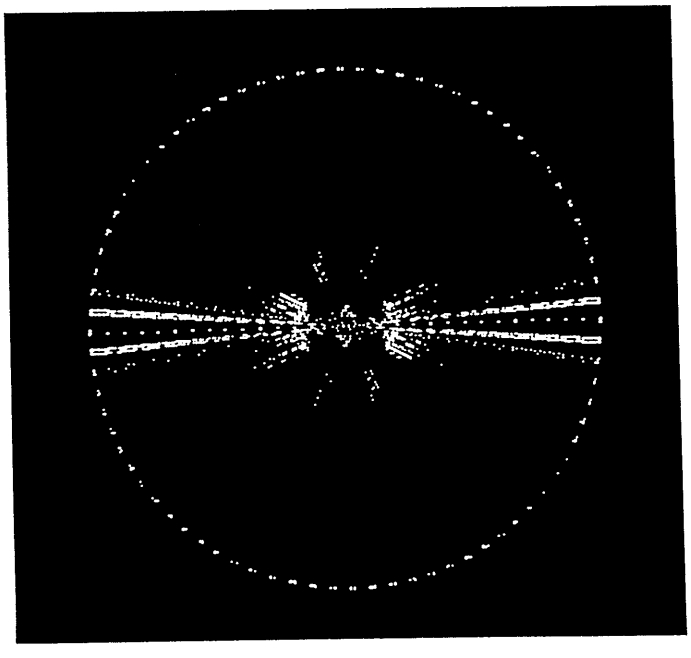
直交スキャンライン法で生成された画像と2.6.4節の実験で求められた N_{ET} マルチスキャニング画像との間で画質を比較する。N_{ET} は直交スキャンライン法の画像生成時間に最も近い時間で走査できるサブ・スキャンライン数を示す。実験には2.6.4節で用いた4種類の画像を用いた。

マルチスキャニング法で隠れ面消去する場合には、水平方向のエッジに特徴的にジャギが現れる。これは、サブ・スキャンラインの間隔で定まる垂直解像度が、演算精度で定まる水平解像度に比べ、極めて低いことが原因である。そこで、式2.1を用いて差分画像を作成し、水平エッジのジャギを比較する。JP_x 画像を図2-19、2-20、2-21、2-22に示す。いずれの実験画像でも、JP_x 画像中の白点の数は、マルチスキャニング法に比べ直交スキャンライン法が少ない。この結果は、直交スキャンライン法はマルチスキャニング法に比べ、同じ時間内でより高精度にエリアシングを除去できることを示している。

次に、放射状パターン(図2-7)を題材にして、スキャンライン法とZバッファ法でサンプリング間隔を変化させて JP_x 画像と JP_y 画像を作成した。図2-23は画素の水平・垂直方向の分割数を横軸に、JP_x, JP_y 画像中の白点の数を縦軸にプロットしている。Zバッファ法では水平・垂直方向ともほとんど同じグラフになる。スキャンライン法では、水平方向はZバッファ法と類似したグラフになるが、垂直方向はほぼ一定にな

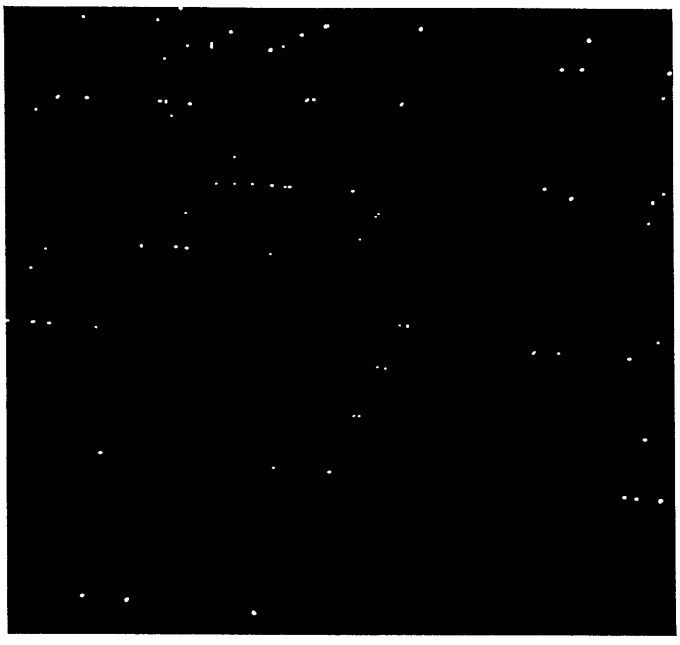


(a) 直交スキャンライン法

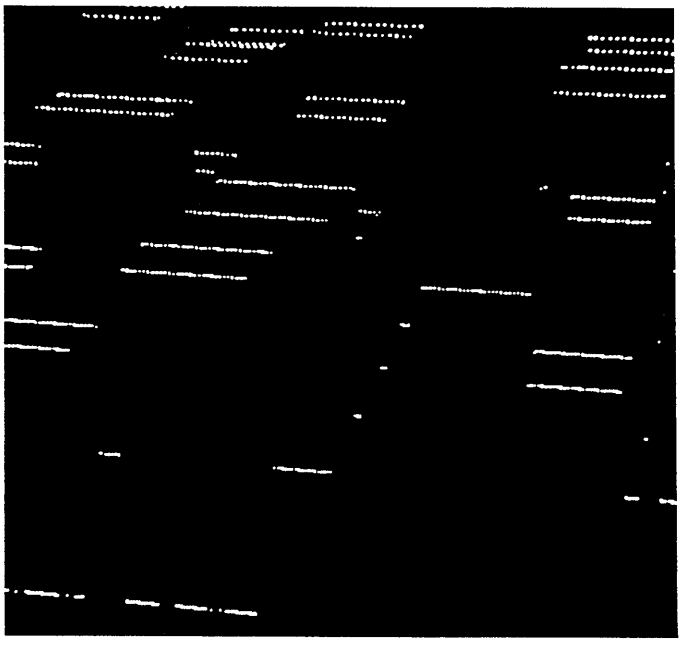


(b)N_{ET} マルチスキャニング法

図 2-19: 放射状パターンの JP_x 画像

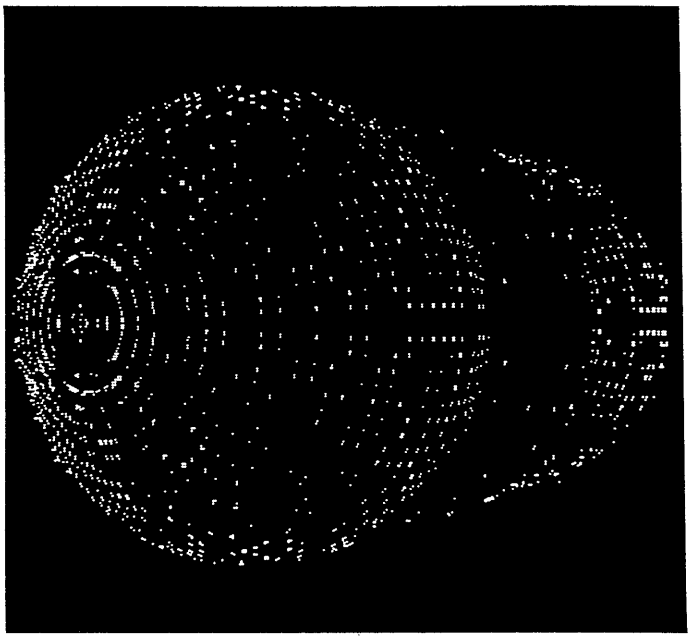


(a) 直交スキャンライン法

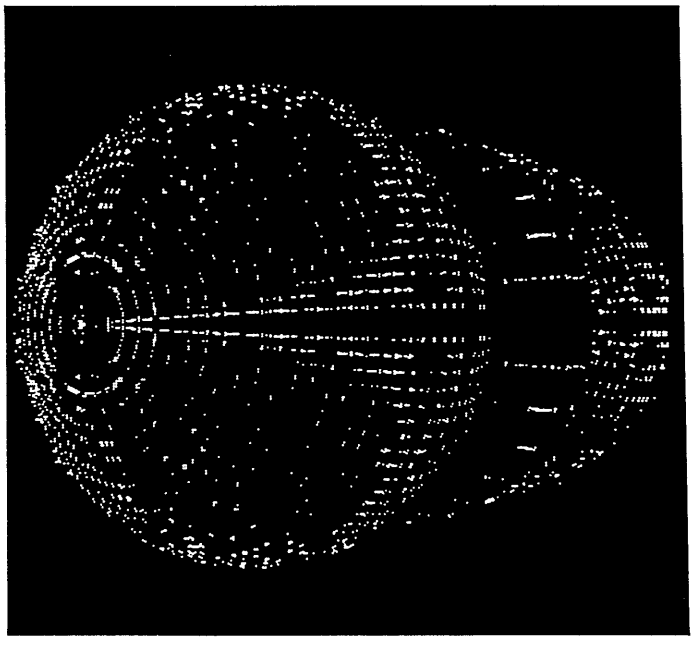


(b)N_{ET} マルチスキャニング法

図 2-20: *Buildings* の $\mathcal{JP}_x$ 画像

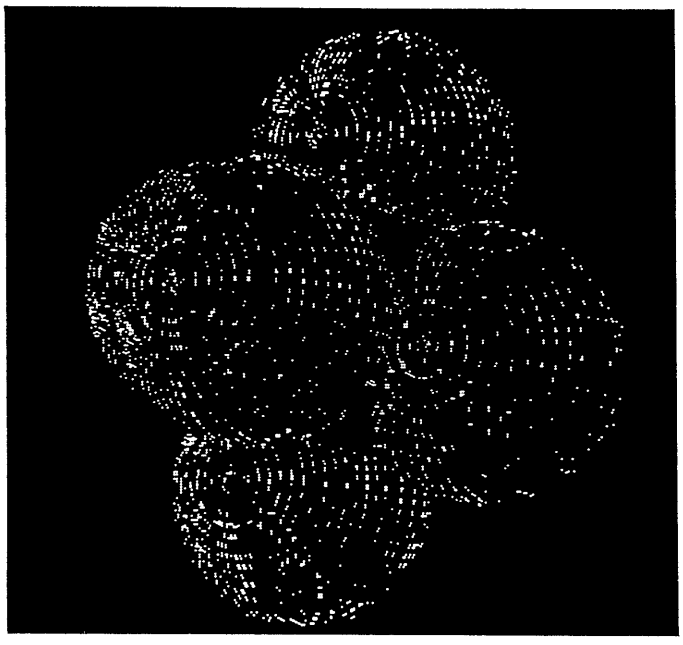


(a) 直交スキャンライン法

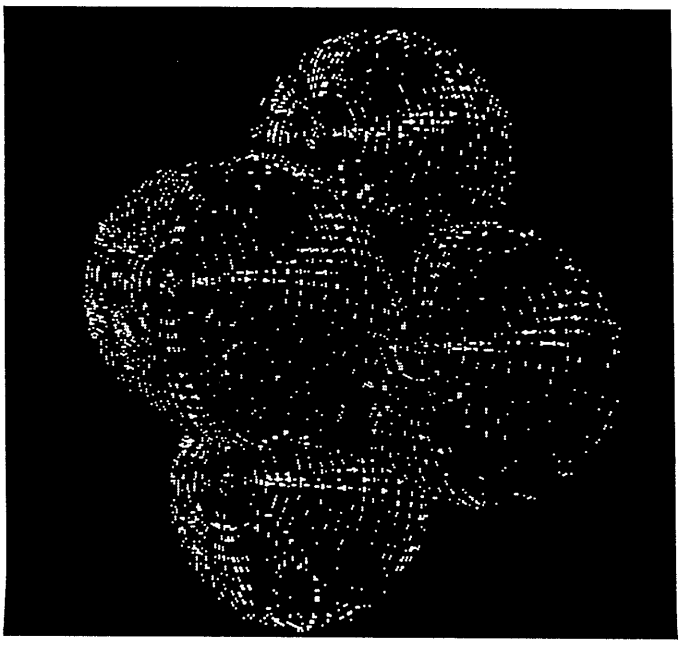


(b)N_{ET} マルチスキャニング法

図 2-21: *Three balls* の $\mathcal{JP}_x$ 画像



(a) 直交スキャンライン法



(b)N_{ET} マルチスキャニング法

図 2-22: *Seven balls* の JP_x 画像

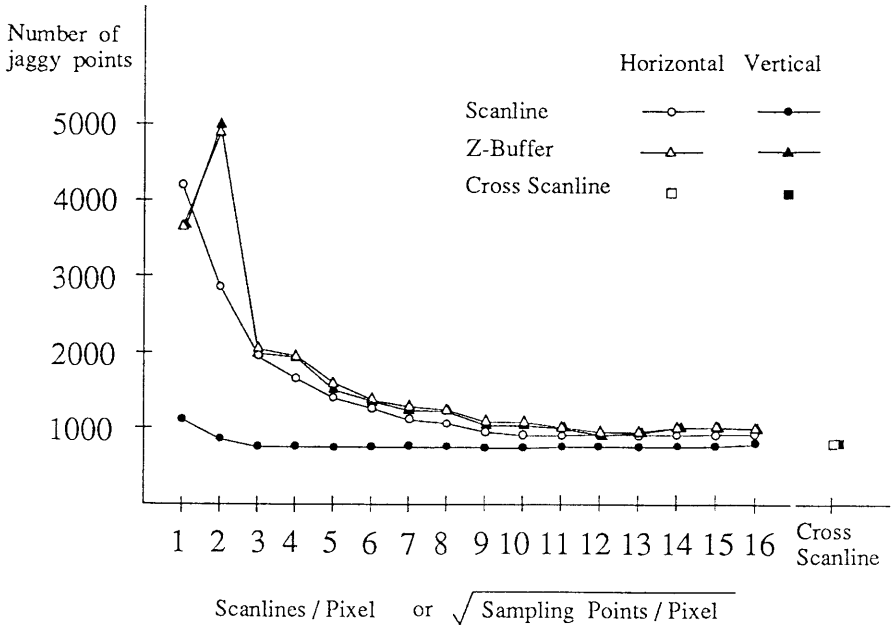


図 2-23: 画素あたりの走査線の本数と *JP* 画像中の白点の数

る。これはスキャンラインの水平分解能が演算精度の正確さを持つことに起因する。このグラフはサンプリング間隔を細かくすると画質が向上することを示している。

比較のため直交スキャンライン法の白点数を図 2-23の右端の Cross Scanline の欄に示した。直交スキャンライン法の白点数は、Zバッファ法やスキャンライン法で画素を極めて細かく分割する場合の白点数に相当する。実験から、直交スキャンライン法の画像は極めて高品質であるといえる。

2.8 フィルタ処理

フィルタ処理は細い線分を滑らかに表示したりモアレを除去するのに有力な手法である。本節では直交スキャンライン法に適したフィルタ手法を述べる。

2.8.1 フィルタの形式

直交スキャンライン法は図 2-24に示すように、水平 (H)・垂直 (V) スキャンラインを用いてポリゴンを三角形または台形に分割する。このため、クリッピングによる隠れ面消去と同様に、1 画素内に含まれるポリゴンの領域を精密に決定できる。そこで、フィルタ関数をポリゴン領域内で積分すると効果的にエリアシング除去できる。

フィルタ関数には、方向に依存しないことと微分連続性を持つことを考慮して、2 次元の Gauss 関数を採用した。フィルタサイズを大きくするとエリアシング除去はたやすくなるが、空間解像度の低下を招く。そこで、フィルタのサイズを 2×2 画素に選

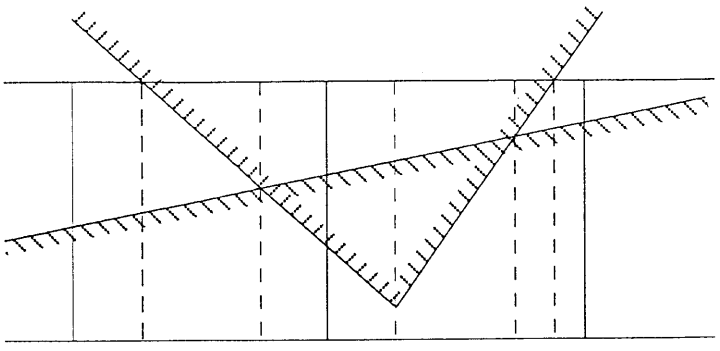


図 2-24: 直交スキャンラインによるポリゴンの分割

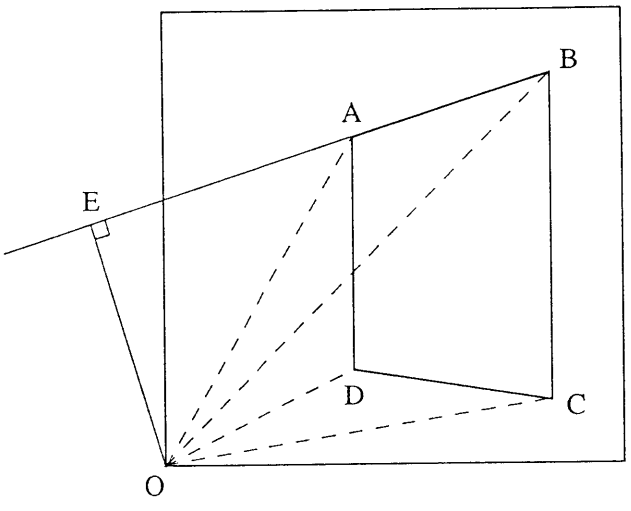


図 2-25: Feibush のフィルタリング手法

んだ。直交スキャンライン法では精密に求められたポリゴン領域内でフィルタ関数を積分するため、小さいフィルタでも十分効果的にエリアシングを除去できる。Gauss 関数を台形領域で代数積分することは困難なため、テーブルを用いて近似積分する。

2.8.2 Feibush の手法

回転対称関数を 2 次元領域で積分する方法として、Feibush の手法 [13] が知られている。図 2-25の領域 $ABCD$ 内での積分を例に手法を解説する。

点列 P_1, P_2, ..., P_n を頂点とする多角形内部での積分を $I(P_1, P_2, ..., P_n)$ で表記する。図 2-25に示すように、台形領域 $ABCD$ での積分は O を頂点に含む三角形での積分を用いて、

$$I(A,B,C,D) = I(O,A,B) + I(O,B,C) - I(O,C,D) - I(O,D,A) \quad (2.5)$$

で記述できる。従って、各三角領域での積分値を計算すれば $ABCD$ での積分値が与えられる。

点 O から直線 AB におろした垂線と AB との交点を E とする。E を用いて $I(O,A,B)$ は

$$I(O,A,B) = I(O,E,B) - I(O,E,A) \tag{2.6}$$

のように分割できる。$I(O,E,A)$ を積分するため、ベクトル EO が y 軸となるように座標系を回転する。この座標変換では回転対称関数の積分値は変化しない。変換後の A の座標を (x_A, y_A) であらわすと、$I(O,E,A)$ は Gauss 関数を $(0,0), (0,y_A), (x_A,y_A)$ を頂点とする直角三角形内で積分して求められる。Gauss 関数の対称性から、この積分は $(0,0), (0,|y_A|), (|x_A|,|y_A|)$ を頂点とする三角形の内部での積分と一致する。

積分を解析的に計算することは困難なため、テーブル参照により近似計算する。2×2 画素のフィルタでは、x と y の定義域は画素幅を単位として $[0,\sqrt{2}]$ となる。定義域内を格子状に分割し、格子点 P_{ij} の座標値を (x_i, y_j) で記述する。テーブル要素 $T(i,j)$ には Gauss 関数を三角形 $(0,0), (0,y_j), (x_i,y_j)$ で積分した値を格納する。

このテーブルを用いることで、台形 $ABCD$ での積分は座標変換と 8 回のテーブル参照、及びテーブル値の加減算で与えられる。フィルタの大きさが 2 × 2 画素の場合は、台形領域は周囲 4 画素の輝度値に影響を与えるので、原点を移動して都合 4 回積分計算を行う必要がある。

2.8.3 y 積分テーブル法

Feibush の手法では座標系を回転するために多数の乗除算が必要になる。また、1 つの台形を 8 つの三角形で置き換えるので処理量が増加する。そこで、直交スキャンラインアルゴリズムで分割された台形領域の上底と下底は y 軸と平行になる特徴を利用して、座標系を回転しないテーブル積分法を考案した。1 画素を格子状に分割し、格子点 P_{ij} の座標系を (x_i, y_j) で記述する。P_{ij} に対応するテーブルの要素 $T(i,j)$ に長方形 $(x_{i-1},0)$, $(x_i,0)$, (x_i,y_j), (x_{i-1},y_j) で Gauss 関数を積分した値を格納する。領域 $ABCD$ の積分のうち y 方向はテーブルの値の差で計算し、x 方向はループで総和を求める。

Feibush の手法と y 積分テーブル法とで処理時間を比較した。テーブルサイズと処理時間の関係を図 2-26に示す。横軸の数字はテーブルの 1 辺の要素数を示している。縦軸は、乱数を用いて発生した 100 万個の台形領域をフィルタリングするために必要な時間を、秒単位で示している。Feibush の手法の処理時間はテーブルの大きさにかかわらず一定になる。y 積分法では、x 方向の分割数が多くなるほどテーブルで値を足し込む回数が増えるため、テーブル 1 辺の要素数に比例して計算時間が増大する。この実験では 150 × 150 のテーブルで処理時間の優劣が入れ替わった。実際の画像生

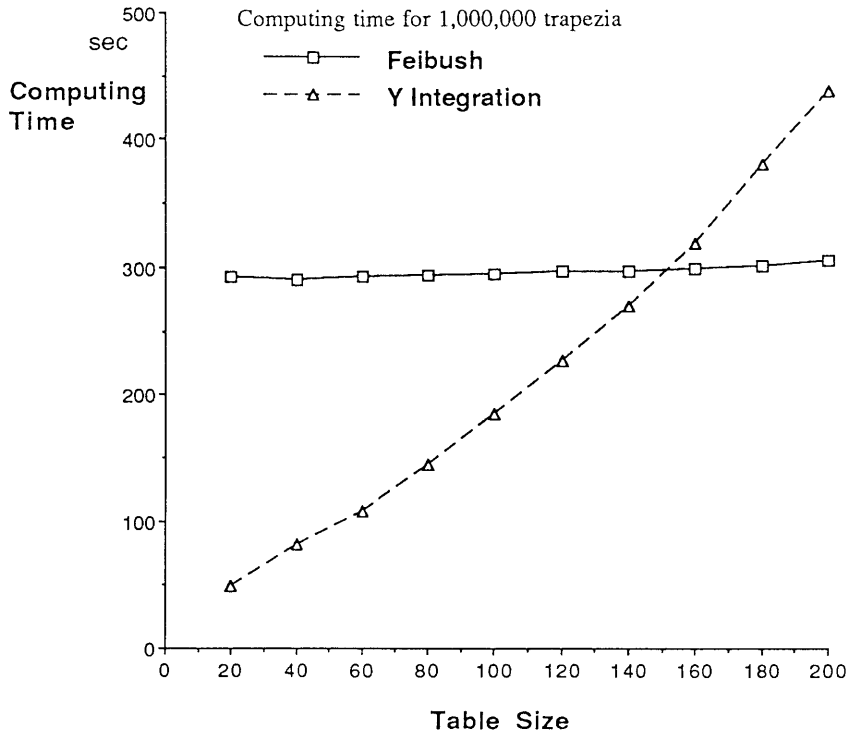


図 2-26: 100 万回のフィルタリングに要する時間

成で使うテーブルはせいぜい 100 程度の大きさのため、本研究では y 積分テーブルを用いた。

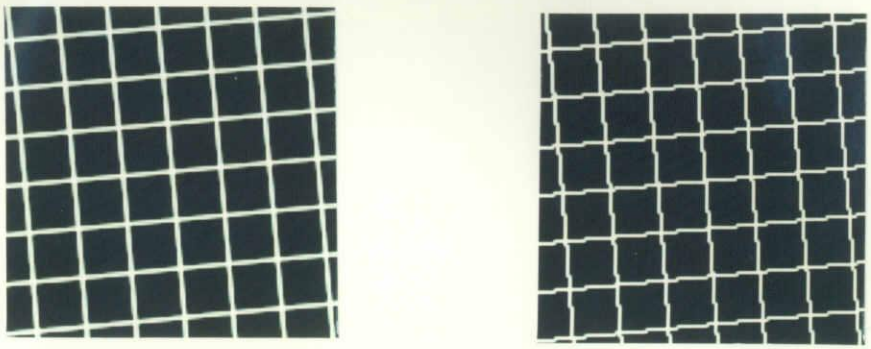
2.8.4 実施例

フィルタ処理の効果は細線表示において特に顕著に現れる。フィルタ処理した場合としない場合の差を図 2-27に示す。画像は見やすくするため 4 倍に拡大されている。この実験では 2×2 画素 Gauss フィルタ処理を大きさ 100×100 の y 積分テーブルで実現している。フィルタ処理により明らかにエリアシングが減少した。

2.9 結言

隠れ面消去に起因するエリアシングを確実に防ぐため、Hスキャンラインと Vスキャンラインを用いて画像を縦横 2 方向に走査する直交スキャンライン法を提案した。

Hスキャンラインは、画素の水平境界線に配置され、通常のスキャンラインアルゴリズムを用いて画像の左端から右端まで走査される。Vスキャンラインは、ポリゴンの頂点, エッジ同士の交点, エッジとHスキャンラインとの交点, 画面の左右端, を通るように配置され、隣合う 2 本のHスキャンラインの間を 1 画素の高さだけ走査される。Vスキャンラインもスキャンラインアルゴリズムで隠れ面消去する。Hスキャンライ



(a) フィルタ処理有り　　(b) フィルタ処理無し

図 2-27: フィルタの効果

ンとVスキャンラインによりポリゴンが重なりの無い台形または三角形に分割されるので、1画素内に投影されるポリゴンの面積を簡単な処理で精密に計算できる。したがって、高精度のアンチ・エリアシングが実現できる。2.6節の実験では、直交スキャンライン法がエッジの方向に依存せずエリアシングを除去できることと極めて細い線分でも適正に表示できることが示された。

マルチスキャニング法のエリアシングは挿入するサブ・スキャンライン数に比例して減少するが、計算量はサブ・スキャンライン数に比例して増大する。画像を効率良く生成するには、画像に応じて適切なサブ・スキャンライン数を選ぶ必要がある。最適数を論理的に決定することは極めて難しいので、実際には経験的にサブ・スキャンライン数を決めて画像生成している。このため、エリアシングが発生すればサブ・スキャンラインを追加して画像生成をやり直さなければならない。特にアニメーションでは視点と物体との位置関係がダイナミックに変化するため、場面によりエリアシング除去に必要なサブ・スキャンライン数が大きく異なる。画像品質の確認と画像生成のやり直しはユーザーにとってきわめて煩わしい作業である。直交スキャンライン法は正確なポリゴンの面積を計算できるので、エリアシングを確実に除去できる。したがって、画像生成をやり直す必要がない。作業効率が良くなるので全体の処理時間は減少する。

2.6.4節の実験では、直交スキャンライン法は4～7本のサブ・スキャンラインを挿入するマルチスキャニング法と等しい時間で画像生成できることを示した。2.7節の画質評価実験では、同じ処理時間ならば、直交スキャンライン法の画像がはるかに高品質であることを確認した。以上の結果から、精密なアンチ・エリアシングを必要とする場合には直交スキャンライン法が時間・画質共に有利である。

フィルタ処理と組み合わせることで、いっそう強力にエリアシング除去できる。本章では直交スキャンライン法に適したy積分テーブル法でフィルタリングした。直交スキャンライン法は1画素の中でポリゴンが占める領域を精密に検出できるので、2×2画素のGaussフィルタでも十分効果的にエリアシングを除去できた。

第 3 章

精密輝度計算法

3.1 序言

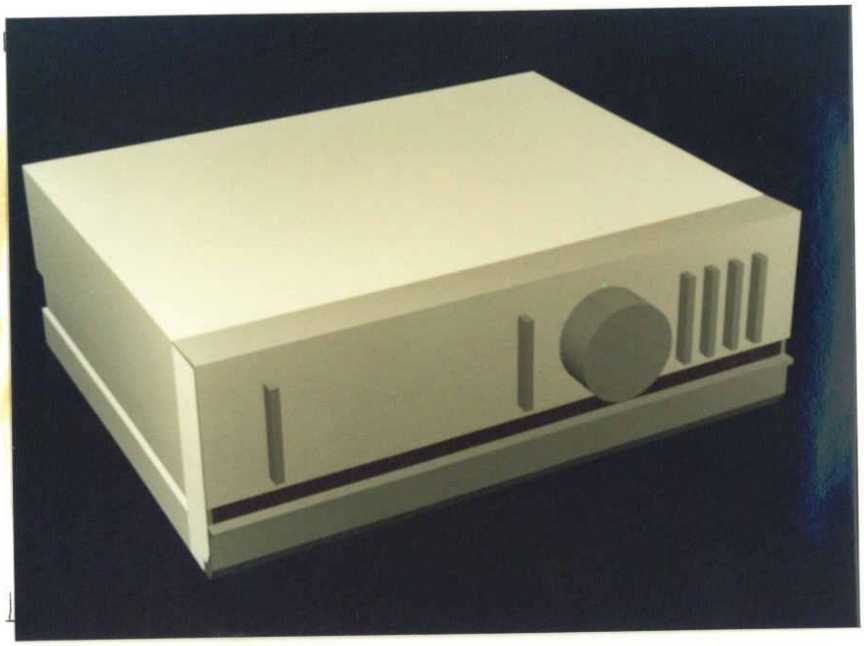
本章では精密レンダリング法 (Precise Rendering Method) と名付けた画像生成手法を提案する。本手法は、2章で報告した直交スキャンライン法で 1 画素に投影されるポリゴンの領域を精密に求めた後、本章で報告する反射強度積分法 (Reflection Integration Method) でその領域が反射する光の強度を解析的に積分する。精密レンダリング法は画像中に大曲率の曲面が含まれていても画素の輝度値を精密に計算できる。

実際の物体では、一見すると平面同士の交線のように見える稜線も微小な半径の円筒面で構成されている。同様に、頂点も数学的な点ではなく微小な球面でできている。これらの稜線や頂点を、丸められた稜線, 丸められた頂点と呼ぶ。多くの日用品は安全のため稜線と頂点が丸められている。プラスチック製品や鋳物製品の形状は型抜きしやすいように稜線と頂点を丸めたデザインになっている。丸めには見た目を美しくする効果もある。このように、現実の稜線や頂点は曲面で定義されるほうが一般的である。

丸められた稜線や頂点のような曲率の大きな曲面には広い範囲の光源が映り込むので、ハイライトが現れる確率が高い。ハイライトは物体形状を認識する手がかりになるので、ハイライトを含む画像は見る者に実在感を与える。ハイライトの効果を図 3-1に示す。図 3-1(a) と (b) は稜線と頂点のハイライトを除き全く同じ画像である。この例からも、ハイライトが画像の写実性を高めることは明かである。

曲率の大きな曲面上では短い距離の間に面法線ベクトルの方向が連続的に、かつ、大きく変化する。1 画素の中でもサンプル点の位置により面法線の方向が異なるので、反射光の強度も異なる。このため、輝度計算でエリアシングが発生する。曲率の大きな曲面をポリゴンで近似するとポリゴンは極めて細かくなる。このため、隠れ面消去でもエリアシングが発生する。これらのエリアシングにより、1 画素を少数の点でサンプリングしたのではハイライトが失われたり過度のハイライトが現れたりする。

精密レンダリング法による画像生成では、隠れ面消去のエリアシングは直交スキャ



(a) ハイライト無し

(b) ハイライト付き

図 3-1: ハイライトの効果

ンライン法で解消され、輝度計算のエリアシングは反射強度積分法で解消される。直交スキャンライン法は、画像を画素の水平境界に沿って画像の幅だけ走査したのち、ポリゴンの頂点, ポリゴンエッジと水平走査線との交点, ポリゴンエッジ同士の交点, 画像の左右端, を通る位置を隣接する水平走査線の間だけ垂直走査する。2 方向の走査によりポリゴンは重なりの無い台形領域に分割されるので、1 画素に投影されるポリゴンの領域を演算精度の正確さで決定できる。

反射強度積分法は、Blinn のモデル [3] を用いて反射光の強度を計算する。視線方向ベクトルと光源方向ベクトルの二等分ベクトルを z 軸に選んで極座標変換すると、輝度は球面積分で与えられる。直交スキャンライン法で求められたポリゴン領域内で面法線の方向が滑らかに変化する場合、領域内の面法線は領域の頂点の面法線が張る方向領域内に分布する。この方向領域と原点を中心とする単位球面との交差領域が積分範囲となる。積分を簡単にするために、積分範囲を z 軸と単位球面との交点を頂点とする単位球面上の三角形に分割する。この変換により、積分変数の一方を代数積分できる。他方は直接積分することが困難なため、被積分関数を Chebyshev 多項式で近似したのち代数積分する。

一般的な CG 画像は赤緑青をそれぞれ 256 階調で表現するため、表示する輝度の最大値の 1/256 の量子化誤差を含む。輝度の計算誤差が輝度の量子化誤差よりも小さければ、正確な輝度値といえる。計算機の演算誤差は量子化誤差に比べ十分に小さいので、近似誤差が量子化誤差よりも小さくなるように Chebyshev 多項式の次数を選ぶことで正確な輝度値を計算できる。

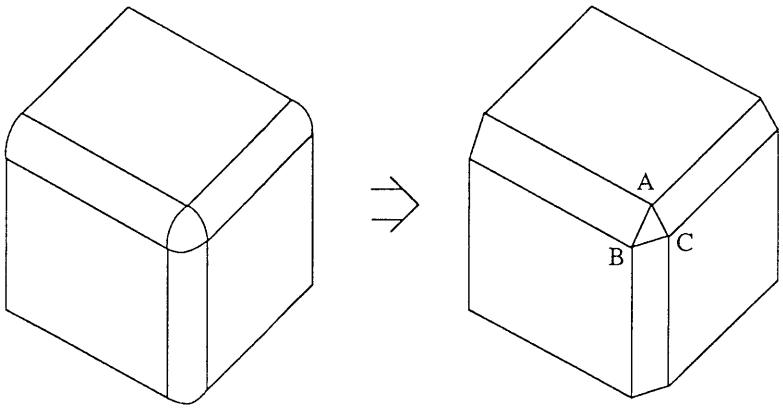
精密レンダリング法は、直交スキャンライン法と反射強度積分法とを組み合わせることで、高度のアンチ・エリアシングと正確なハイライト表示の両方を同時に実現する。このうち、直交スキャンライン法の詳細は2章で報告した。本章では反射強度積分法を報告する。以下、3.2節で従来手法の問題点、3.3節と 3.4節で反射強度積分法の詳細、3.5節で計算機実験結果を報告する。

3.2 従来手法

丸められた稜線や頂点を曲面パッチで与えることは、計算コストが高く、現実的ではない。そこで、次の 2 種類のアプローチがとられる。

3.2.1 ポリゴン分割法

丸められた稜線や頂点をポリゴンで近似する。例えば、図 3-2(a) の丸められた稜線と頂点は (b) のように近似できる。一般に、丸めの半径は物体の大きさに比べ小さいため、これらのポリゴンは図 3-3に示すように画素より細く、小さくなる。図 3-3を通



(a) 曲面による記述　　(b) ポリゴン近似

図 3-2: 丸められた稜線と頂点

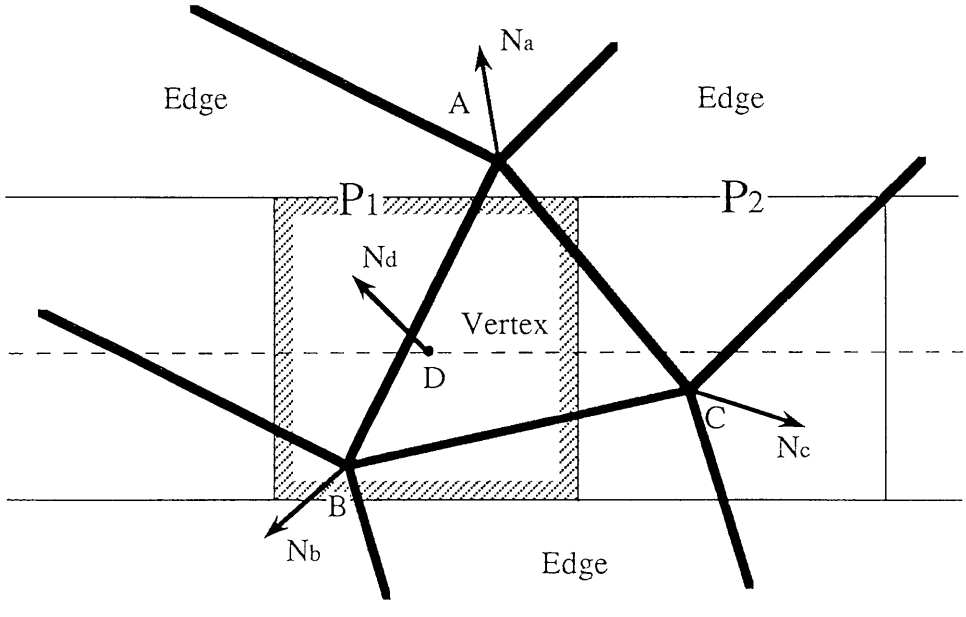


図 3-3: 画素中心をサンプリング

常のZバッファ法で隠れ面消去すると、三角形 ABC が画素 P_1 全体を覆っていると判断される。このため、画素 P_1 の輝度値が三角形 ABC の輝度値だけから決まってしまう。

図3-3のように頂点 A, B, C に与えられる面法線ベクトル N_a, N_b, N_c の方向が大きく異なる場合は、三角形 ABC には広い範囲の光源が映り込む。しかし、画素中心 D の面法線 N_d だけを用いて輝度計算すると、ごく限られた範囲に光源が存在する場合を除きハイライトは生じない。このように、点 D での輝度は三角形 ABC の輝度とは異なる。

以上のことから、画素 P_1 の輝度は正しく求められないので、エリアシングが発生する。丸められた稜線や頂点をポリゴンで近似する場合の問題点をまとめると、

(1) ポリゴンが画素に比べ小さいため隠れ面消去でエリアシングが生じる,

(2) ポリゴンの各頂点に与えられる面法線の向きが大きく異なるため、サンプル点の位置により輝度が異なる,

の2つである。

スーパーサンプリング手法は両方の問題点を解決できるが、輝度を精密に計算するには1画素あたり極めて多くのサンプル点が必要になる。サンプル点を減らすには、物体の複雑さ, 視点・光源・反射点の配置, 反射の鋭さなどに応じて1画素あたりのサンプル数を動的に変える必要がある。しかし、サンプリングの結果をもとに細かく分割するか否かを判断するため、エリアシングの発生する箇所がもれなく詳細にサンプリングされる保証はない。これは適応サンプリング手法の重大な欠点である。

問題点の (1) はクリッピングを用いた隠れ面手法 (1.2.6節) や直交スキャンライン法 (2章) を用いて解決できる。これらの手法は1画素中のポリゴン領域を正確に決定できるので、隠れ面消去のエリアシングは精密に除去される。

問題点の (2) は、1画素に投影されるポリゴン領域を鏡面反射強度の変化が無視できるほどの小領域に細分割して輝度計算することで解決できる。ただし、反射が鋭い場合には反射強度は面法線方向の変化に敏感なため、きわめて詳細に分割する必要がある。ポリゴン内を一律に分割する場合は、ポリゴンを極めて多数の微小領域に分けなければならないので、計算時間が長大になる。計算時間を減らすには動的に分割する必要があるが、サンプリングの結果を用いて分割を制御するため、誤りなく分割できる保証はない。

直交スキャンライン法で隠れ面消去したあとスーパーサンプリングで輝度計算する手法は初めからスーパーサンプリングを用いる手法よりは効果的だが、輝度値の精度の点でもエリアシング除去の確実性の点でも不十分である。

3.2.2 ハイライト合成法

ハイライト合成法 [35] は丸められた稜線と頂点のハイライトを表示する手法である。丸められた稜線と頂点をそれぞれ円筒面と球面で近似して、精密にハイライトを計算する。他の部分の陰影画像は丸めがない場合の形状をZバッファ法でレンダリングして生成する。最後に各画素のZ値を比較して2枚の画像を重ね合わせることでハイライト画像を生成する。この手法は稜線と頂点以外の精密なハイライト計算を行わないので、計算時間はそれほど増大しない。ポリゴン近似を行わないため、データ量も増加しない。しかし、以下の点が問題となる。

陰影画像とハイライトの合成に視点からの奥行きを用いるが、奥行きはサンプリングで求めるため厳密ではない。このため、前後判定に誤りが生じ、ハイライトが失われたり見えないハイライトが現れたりする。ハイライト以外の部分をZバッファ法で生成するため、画像にエリアシングが表れる。スーパーサンプリングで陰影画像を作ればエリアシングを取り除くことはできるが、計算量は増加する。

平面部分にハイライトが生じる場合には、丸められた稜線や頂点の輝度は平面部分に比べて低くなる。稜線の陰は物体の輪郭を際だたせる効果があるので、この現象は画像の写実性を高める上で重要である。しかし、ハイライト合成法は平面部分の輝度にハイライトの輝度を足すため、陰を表示できない。

ハイライト合成法では稜線のハイライトは輝度を持つ線分として計算されるため、丸めの半径が画素より小さい場合にしか正しく表示できない。このため、アニメーションのように視点がダイナミックに動く場合には適用が難しい。また、稜線と頂点という物体形状の特徴的部分だけを円筒面または球面で近似してハイライトを計算するので、一般的な曲面に現れるハイライトは正しく表示できない。

ポリゴン近似手法では、画素の輝度値が正しく求められるならば上記の問題は解決できる。そこで、本論文は、丸められた稜線と頂点をポリゴンで近似する。

3.3 精密レンダリング法

3.2節でも述べたように、曲率の大きな曲面をポリゴンで近似する場合、次の点で高品質画像の生成が困難になる。

(1) ポリゴンが画素に比べ小さいため、隠れ面消去でエリアシングが発生する。

(2) ポリゴンの各頂点に与えられる面法線ベクトルの向きが大きく異なるため、画素内の位置により輝度値が異なる。

この問題を解決するために、我々は精密レンダリング法を提案する。精密レンダリング法は、2章で報告した直交スキャンライン法を用いて1画素内に投影されるポリゴ

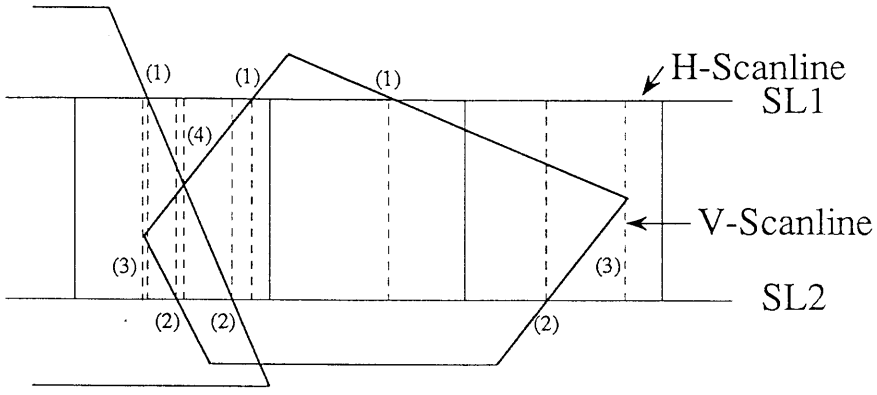


図 3-4: 直交スキャンライン法による分割

ンの領域を正確に求め、その領域の輝度を反射強度積分法と名付けた手法を用いて精密に計算する。

直交スキャンライン法は画像を水平と垂直の 2 方向に走査する。水平スキャンラインは画素の水平境界に配置され、画面の左端から右端まで走査される。垂直スキャンラインはポリゴンの頂点, ポリゴンエッジ同士の交点, ポリゴンエッジと水平スキャンラインとの交点, 画面の左右端, のいずれかを通るように配置され、隣あう 2 本の水平スキャンラインの間だけで走査される。図 3-4に示すように、水平・垂直スキャンラインによりポリゴンは台形または三角形に分割されるので、1 画素に投影されるポリゴンの領域が正確に求められる。この結果、たとえポリゴンが小さくとも、隠れ面消去から厳密にエリアシングを除去できる。

1 画素に投影される領域の各頂点の面法線ベクトルは、元々のポリゴンの頂点に与えられた面法線ベクトルの内挿で与えられる。領域内の面法線ベクトルは、これらの面法線ベクトルが張る角錐型の方向領域の内で変化する。したがって、この方向領域を積分範囲として反射光の強度を積分すれば、ポリゴン領域の正しい輝度が求められる。反射強度積分法は、この積分を解析的に計算することで、ポリゴン領域の中で面法線の方向が大きく変化しても精密な輝度値を得る。

精密レンダリング法を構成する手法のうち、直交スキャンライン法は 2章で詳しく報告した。本章では反射強度積分法を報告する。

3.4 反射強度積分法

3.4.1 Blinn の反射モデル

コンピュータグラフィックスでは、物体表面の明るさを計算するため、Phong[32], Blinn[3], Cook-Torrance[8] などの反射モデルが用いられる。このうち、計算の容易さ

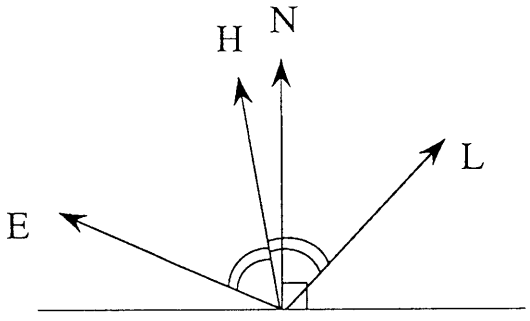


図 3-5: Blinn の反射モデル

とモデルの正確さを考慮して、Blinn のモデルを採用する。このモデルでは、反射光の強度 I_r を拡散反射成分 I_d と鏡面反射成分 I_s に分ける。各成分は、

$$I_r = k_d I_d + k_s I_s, \tag{3.1}$$

$$I_d = R_d (N \cdot L) I_i, \tag{3.2}$$

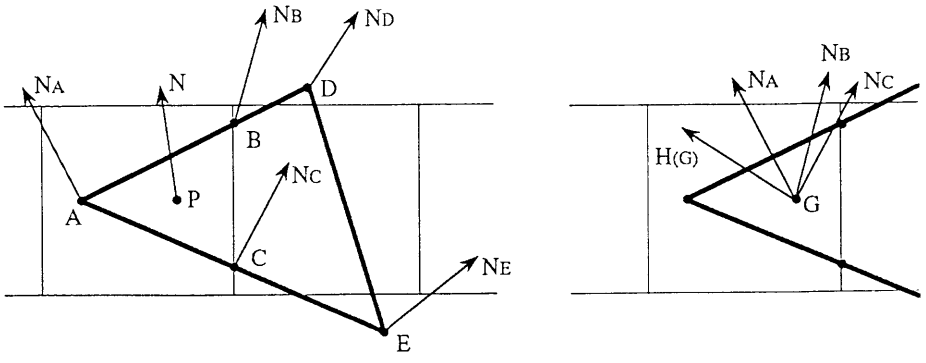
$$I_s = R_s (N \cdot H)^n I_i \tag{3.3}$$

で与えられる。各ベクトルの関係を図 3-5に示す。また、式 3.1–3.3, 及び図 3-5中の記号は次のように定義される。

I_i : 入射光の強度,
I_r : 反射光の強度,
I_d : 拡散反射光の強度,
I_s : 鏡面反射光の強度,
L : 光源の方向を示す単位ベクトル,
E : 視点の方向を示す単位ベクトル,
H : L と E の二等分線方向の単位ベクトル,
N : 面法線方向の単位ベクトル,
n : 鏡面反射指数 (鏡面反射の鋭さを表す指数),
k_d : 拡散反射の寄与率,
k_s : 鏡面反射の寄与率 ($k_d + k_s = 1$),
R_d : 拡散反射率,
R_s : 鏡面反射率。

3.4.2 鏡面反射強度の積分

図 3-6の例について三角形 ABC の鏡面反射成分 I_s の計算を考える。面法線ベクトルは元々のポリゴンの頂点 A, D, E だけに与えられている。図 3-6(a) の N_A, N_D, N_E が与えられた面法線ベクトルである。点 B, C での面法線ベクトル N_B, N_C は Phong のスムーズシェーディング手法 [32] と同様に N_A と N_D, N_A と N_E をそれぞれ線型に



(a) 1ピクセル内の面法線ベクトル　　(b) 近似計算

図 3-6: 鏡面反射成分の積分

内挿して求める。また、三角形 ABC 内の任意の点 P での面法線ベクトルを N とすると、N は三角形 ABC の頂点の法線ベクトル N_A, N_B, N_C の線型結合で与えられる。点 P での鏡面反射強度は式3.3で与えられるので、三角形 ABC の鏡面反射鏡度は

$$I_s = \int_{\triangle ABC} R_s (N \cdot H)^n I_i \, dS \tag{3.4}$$

で計算できる。

式3.4の H と I_i は厳密には点 P の位置により異なる。しかし、光源と視点の位置が三角形 ABC から十分に遠い場合には、三角形 ABC 内では一定と仮定できる。そこで、H, I_i を三角形 ABC の重心 G での値 H_G, I_{iG} で代表させると、式3.4は

$$I_s = R_s I_{iG} \int_{\triangle ABC} (N \cdot H_G)^n \, dS \tag{3.5}$$

となる。この近似を図で表すと、図3-6(b) となる。

G を原点, H_G を z 軸に選んで極座標変換する。z 軸から測った角度を θ, z 軸回りの回転角を ϕ で表すと、$N \cdot H_G = \cos\theta$ となる。極座標変換により被積分関数の変数が θ だけになるので、積分を簡略化できる。

N_A, N_B, N_C は単位ベクトルなので、始点を原点に選ぶと終点は単位球面上の点になる。そして、これらを内挿して得られる面法線ベクトルの終点は、図3-7の斜線で示した単位球面上の三角形内に存在する。そこで、積分範囲を N_A, N_B, N_C で定義される方向領域 $\Omega_{N_A N_B N_C}$ に変換する。また、三角形 ABC 内の微小面積 dS は単位球面上の微小面積 $d\omega$ で置き換えられる。この結果、鏡面反射鏡度 I_s は

$$I_s = R_s I_{iG} \int_{\Omega_{N_A N_B N_C}} \frac{\partial(x,y)}{\partial(\theta,\phi)} \cos^n\theta \, d\omega \tag{3.6}$$

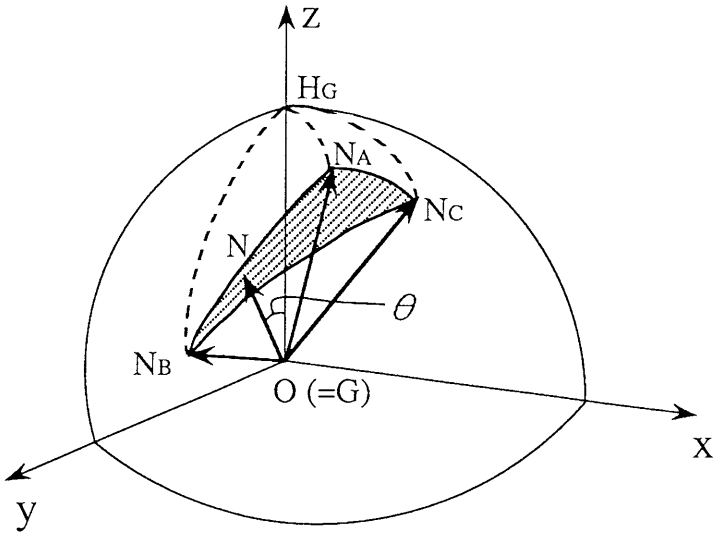


図 3-7: 単位球面上の積分領域の分割

で表せる。式3.6のヤコビヤンを

$$\frac{\partial(x,y)}{\partial(\theta,\phi)} \approx \frac{S_{ABC}}{S_{N_AN_BN_C}} = K_{ABC} \tag{3.7}$$

で近似して、

$$I_s = R_s I_{iG} K_{ABC} \int_{\Omega_{N_AN_BN_C}} \cos^n\theta \, d\omega \tag{3.8}$$

を得る。式3.7中の S_{ABC} は三角形 ABC の面積を、$S_{N_AN_BN_C}$ は方向領域 $\Omega_{N_AN_BN_C}$ に切り取られる単位球面の面積 (すなわち $\Omega_{N_AN_BN_C}$ の立体角) を表す。式3.7は方向領域 $\Omega_{N_AN_BN_C}$ 内の面法線ベクトルが三角形 ABC 内に均等に現れることを仮定している。この分布は面法線ベクトルを三角形 ABC 内で線形に内挿する場合とは異なるが、内挿の目的は面法線をなめらかに変化させることにあるので、式3.7は妥当な仮定と言える。

3.4.3 積分領域の分割

式3.8の積分を簡単にするため、積分範囲を分割する。先ず、関数 G を

$$G(R,S,T) = \int_{\Omega_{RST}} \cos^n\theta \, d\omega \tag{3.9}$$

で定義する。Ω_{RST} は単位球面の上の任意の3点 R, S, T で定義される方向領域である。式3.8は、関数 G を用いて、

$$I_s = R_s \, I_{iG} \, K_{ABC} \, G(N_A, N_B, N_C) \tag{3.10}$$

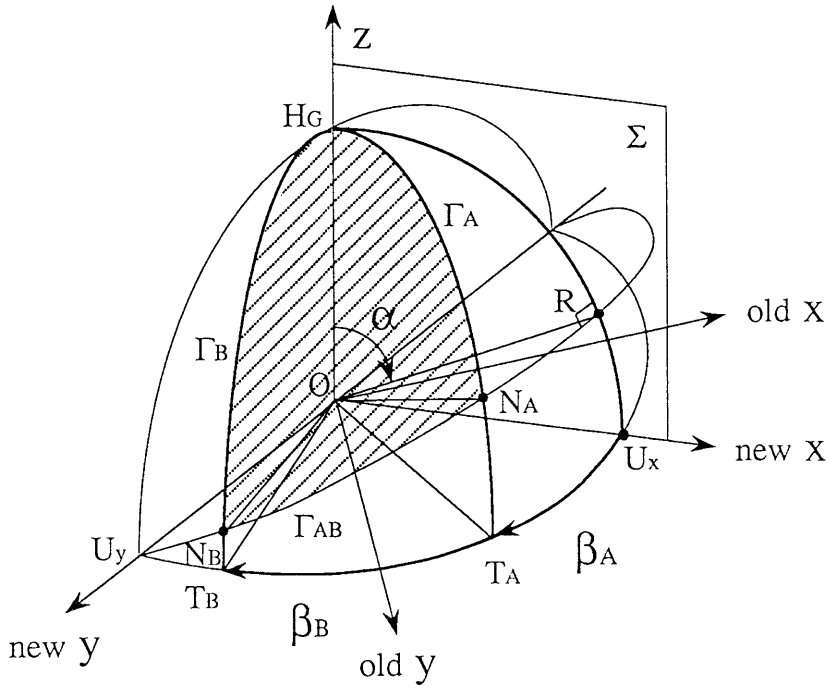


図 3-8: 極座標系における積分

で記述される。次に、$G(N_A,\ N_B,\ N_C)$ を z 軸と単位球面との交点 H_G を用いて分割する。図 3-7の例では、

$$G(N_A, N_B, N_C) = G(H_G, N_B, N_C) - G(H_G, N_A, N_B) - G(H_G, N_A, N_C) \qquad (3.11)$$

のように分割できる。

式 3.11の右辺の各項を積分すれば、式 3.10から I_s が求められる。$G(H_G,\ N_A,\ N_B)$ を例に計算手法を示す。図 3-8に示す点、単位円、平面を以下のように定義する。すなわち、

Γ_{AB} : N_A と N_B を通る大円,
Γ_A : H_G と N_A を通る大円,
Γ_B : H_G と N_B を通る大円,
U_y : Γ_{AB} と xy 面との交点,
T_A : Γ_A と xy 面との交点,
T_B : Γ_B と xy 面との交点,
Σ : 原点を通りベクトル OU_y と垂直な平面,
R : Γ_{AB} と Σ との交点。

$\Gamma_A, \Gamma_B, \Gamma_{AB}$ で囲まれる単位球面上の三角形が積分範囲になる。

ベクトル OU_y が y 軸と一致するまで、z 軸の周りに座標系を回転する。式 3.11の被積分関数は z 軸の回りに回転対称であるため、座標系を回転しても積分値は変わら

ない。z 軸とベクトル OR のなす角度を α、新しい x 軸とベクトル OT_A, OT_B のなす角度をそれぞれ β_A, β_B とする。Γ_{AB} 上の点は ϕ をパラメータとして $(1, \theta_{AB}(\phi), \phi)$ で表せる。$\theta_{AB}(\phi)$ は

$$\cos\theta_{AB}(\phi) = \frac{\cos\phi}{\sqrt{\tan^2\alpha + \cos^2\phi}} \tag{3.12}$$

で与えられる。また、極座標系では

$$d\omega = \sin\theta d\theta d\phi \tag{3.13}$$

と記述できる。従って、

$$G(H_G, N_A, N_B) = \int_{\beta_A}^{\beta_B}\int_0^{\theta_{AB}(\phi)} \cos^n\theta \ \sin\theta \ d\theta \ d\phi \tag{3.14}$$

となる。式 3.14を θ で代数積分して、

$$\begin{aligned} G(H_G, N_A, N_B) &= \int_{\beta_A}^{\beta_B}\left[-\frac{1}{n+1}\cos^{n+1}\theta\right]_0^{\theta_{AB}(\phi)} d\phi \\ &= \frac{1}{n+1}\{(\beta_B - \beta_A) - (H(n,\alpha,\beta_B) - H(n,\alpha,\beta_A))\} \end{aligned} \tag{3.15}$$

を得る。ここで、関数 H は

$$H(n,\alpha,\beta) = \int_0^{\beta}\left(\frac{\cos^2\phi}{\tan^2\alpha + \cos^2\phi}\right)^{\frac{n+1}{2}} d\phi \tag{3.16}$$

で定義される。関数 H を積分すると関数 G が求まり、鏡面反射強度 I_s が計算できる。

3.4.4 多項式近似による積分

上記式 3.16をそのまま代数積分することは困難なため、被積分関数を多項式で近似してから代数積分する。計算量を減らすには、なるべく低い次数で近似を打ち切ることが望ましい。本研究では誤差評価が簡単な Chebyshev 多項式近似を用いることにする。式 3.16の被積分関数を区間 $[0, \pi/2]$ で m 次 Chebyshev 近似 [42] すると、

$$\left(\frac{\cos^2\phi}{\tan^2\alpha + \cos^2\phi}\right)^{\frac{n+1}{2}} \approx \sum_{i=0}^{m} C_i(n,\alpha)T_i(\frac{4\phi}{\pi} - 1) \tag{3.17}$$

で記述できる。ここで、$T_i(t)$ は i 次 Chebyshev 多項式で、

$$T_i(t) = \cos(i\cos^{-1}(t)) \tag{3.18}$$

で定義される。具体的には、

$$\begin{aligned} T_0(t) &= 1, \\ T_1(t) &= t, \\ T_{i+1}(t) &= 2tT_i(t) - T_{i-1}(t) \quad (i \geq 1) \end{aligned} \tag{3.19}$$

の漸化式で与えられる [42]。係数 $C_i(n,\alpha)$ は n と α のみの関数である。$T_i(4\phi/\pi-1)$ は ϕ の i 次の多項式であるから、たやすく積分できる。積分結果を

$$\int_0^{\beta} T_i(\frac{4\phi}{\pi}-1)d\phi = S_i(\beta) \tag{3.20}$$

で表すと、式 3.16は

$$H(n,\alpha,\beta) \approx \sum_{i=0}^{m} C_i(n,\alpha)S_i(\beta) \tag{3.21}$$

となる。

一般的な CG 画像は RGB の値をそれぞれを 256 階調で表現するため、表示する最大輝度の 1/256 の量子化誤差を含む。したがって、計算誤差が量子化誤差より小さければ正確な輝度値といえる。反射強度積分法では積分結果を数個足し合わせて画素の輝度値を計算するので、安全のために積分の近似誤差を輝度値の量子化誤差の 1/10 (4×10^{-4}) 以下にする。この精度は 10 次の Chebyshev 多項式を用いることで達成できる。近似式に数値を代入して輝度を計算すると数値計算の丸め誤差が発生するが、計算機の演算誤差は 10^{-6} 以下なので、近似誤差に比べれば問題にならない。したがって、近似式を用いて輝度を計算することで正確な値が求められる。

3.4.5 円周積分

多角形領域であっても、頂点の面法線ベクトルが一致すれば、球面積分 (2 次元積分) が円周積分 (1 次元積分) になる。また、丸められた稜線では、面法線は稜線と垂直な平面内で変化するため、円周積分となる。そこで、図 3-6で N_B と N_C が一致する場合を例に、近似解を示す。

鏡面反射強度は 1 変数の積分で与えられる。$G(H_G,\ N_A,\ N_B)$ の積分と同様の座標変換を施すと、鏡面反射強度 I_s は

$$I_s = R_s I_{iG} K_{AB} \cos^n\alpha \int_{\gamma_A}^{\gamma_B} \cos^n\omega \ d\omega \tag{3.22}$$

で与えられる。ここで、α は図 3-8に示される OR と z 軸とのなす角度で、γ_A, γ_B は OR と ON_A, ON_B のなす角度である。K_{AB} は座標変換に伴うヤコビアンで、

$$\frac{\partial(x,y)}{\partial(\theta,\phi)} \approx \frac{S_{ABC}}{\gamma_B-\gamma_A} = K_{AB} \tag{3.23}$$

で近似する。式 3.22も被積分関数を Chebyshev 近似してから代数積分する。

3.4.6 拡散反射成分の積分

これまでは、鏡面反射成分の積分法について述べてきたが、拡散反射成分も同様に計算できる。拡散反射成分の強度は式 3.2で与えられる。式 3.3と式 3.2とを比べると、

ベクトル H をベクトル L で置き換え、$n = 1$ とすれば、鏡面反射成分と同様に積分できる。この積分は、L を z 軸とする極座標系において、

$$I_d = R_d I_{iG} K_{ABC} \int_{\Omega_{N_A N_B N_C}} \cos\theta \, d\omega \tag{3.24}$$

で与えられる。したがって、鏡面反射成分と同様に近似積分できる。

3.5 計算機実験

3.5.1 スキャンライン法との比較

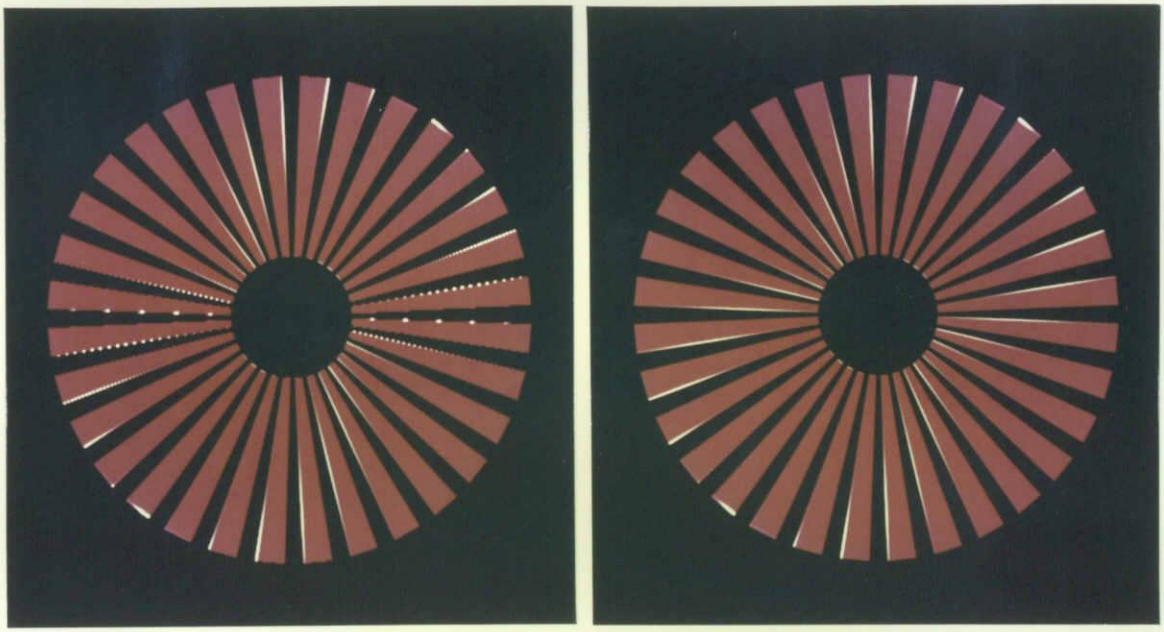
通常のスキャンライン法に Phong のスムーズシェーディング手法 [32] を組み合わせた手法と精密レンダリング法とを比較するため、画像生成実験を行った。はじめに、各稜線と頂点が丸められた放射状パターンを生成した。結果を図 3-9に示す。この図では丸めの半径は画素より小さい。スキャンライン法で生成した図 3-9(a) では水平に近い稜線にジャギが生じた。ハイライトがジャギを強調するため、画像品質が著しく低下する。精密レンダリング法で生成した図 3-9(b) ではどの方向でも正確にハイライトが表示された。

次に、稜線と頂点が丸められた直方体を少しずつずらして配置して、図 3-10を生成した。どの位置でも頂点にハイライトが生じるように平行光線で照明したが、スキャンライン法で生成した図 3-10(a) では場所により頂点の明るさがばらついた。これは、丸めの半径がスキャンラインの間隔と同程度に小さいので、スキャンラインと交差する位置により面法線の方向が大きく異なるために発生する。この動きを動画にすると、頂点が点滅しながら移動することになり、極めて見苦しい。精密レンダリング法は、図 3-10(b) に示すように、どの位置でも正しいハイライトを生成した。

3.5.2 稜線の陰

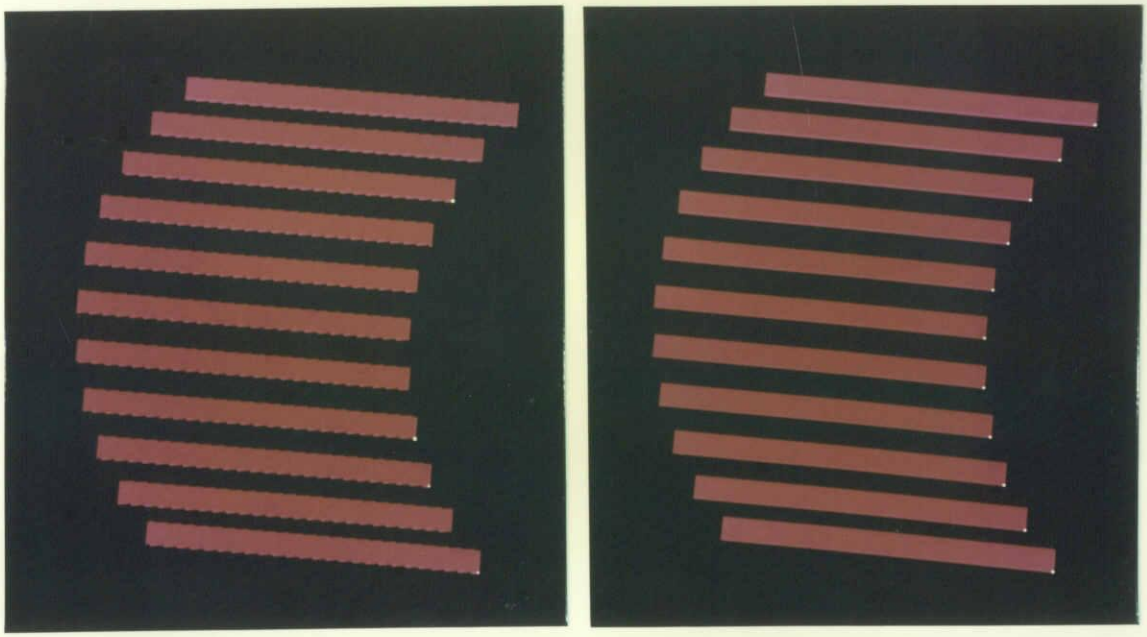
図 3-11は丸められた稜線が暗くなる例を示している。図 3-11に示される多面体の上面は明るく照らされていて、それぞれの輝度差は小さい。面の稜線を丸めると、そこが陰となるため、図 3-11(b) に示されるように境界線が現れる。稜線を丸めないと、図 3-11(a) のように面がつながって見える。

この例に示されるように、陰稜線により物体形状の理解がずっと容易になる。正しい陰の生成もハイライトと同様に写実的な画像を作る上で大切である。陰はハイライトと同じ原因で発生するため、精密レンダリング法は陰稜線を正しく表示できる。しかし、ハイライト合成法はハイライトの輝度を単純に足し合わせるため、陰稜線を表示できない。



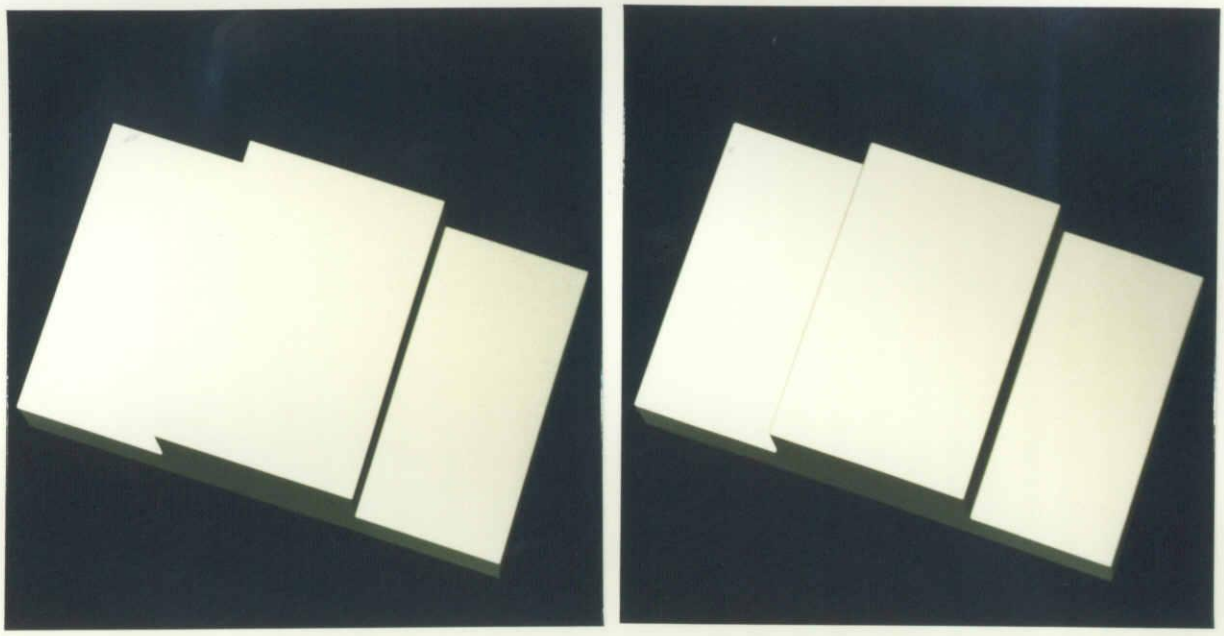
(a) 通常のスキャンライン法　　(b) 精密レンダリング法

図 3-9: 稜線が丸められた放射状パターン



(a) 通常のスキャンライン法　　(b) 精密レンダリング法

図 3-10: 頂点が丸められた物体の移動



(a) 鋭い稜線　　(b) 丸められた稜線

図 3-11: 稜線に現れる陰

3.5.3　丸めの効果

図 3-12は電話機の各ボタンの稜線と頂点に現れるハイライトを表示したものである。図 3-12(a) に稜線を丸めない場合の画像を、(b) に丸めた場合の画像を示す。図 3-13は図 3-12のボタン部分を拡大した画像である。稜線のハイライトと陰は物体の輪郭を強調するので、物体形状を認識する助けになる。このため、画像の立体感が高まる。

図 3-14と図 3-15に稜線を丸めた三次元ロゴの画像を示す。本実験は稜線の作るハイライトが画像の写実性を高める上で重要であることを示している。

3.5.4　画像生成時間

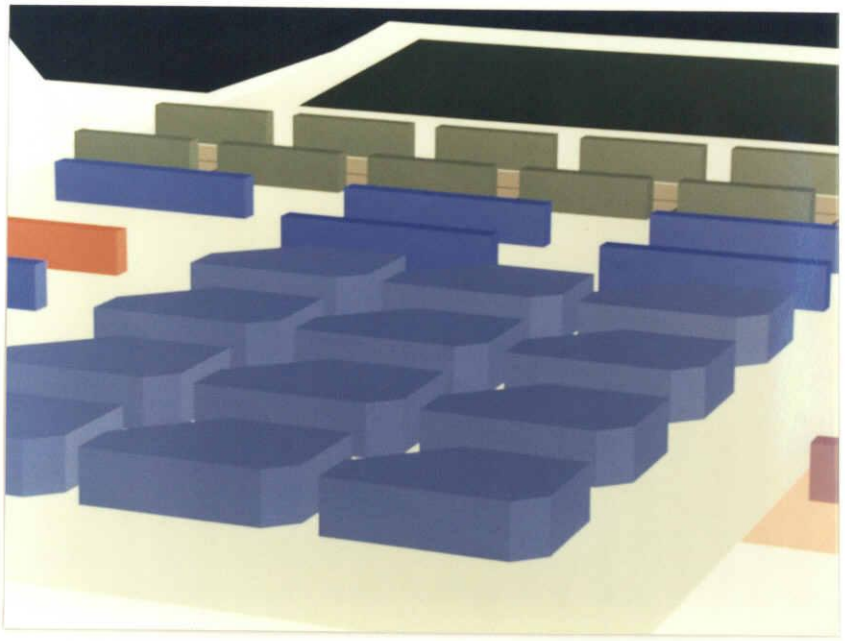
実験で用いた画像の生成時間を測定した。反射強度を積分する場合 (積分あり) と通常どおり 1 法線を用いて輝度計算する場合 (積分なし) の生成時間を表 3-1に示す。単位は秒である。実験には約 6MIPS の処理能力を持つ VAX8840 を用いた。表中の生成時間は 2.8節で報告した 2 次元 Gauss フィルタの処理時間も含む。比較のため、スムーズシェーディングを伴う通常のスキャンライン法の生成時間も併せて示した。

‘時間差’の項に示された秒数が反射強度積分に要する時間である。実験では、丸められたエッジや頂点の画面上の面積が広い場合に、輝度積分の時間が長くなった。最

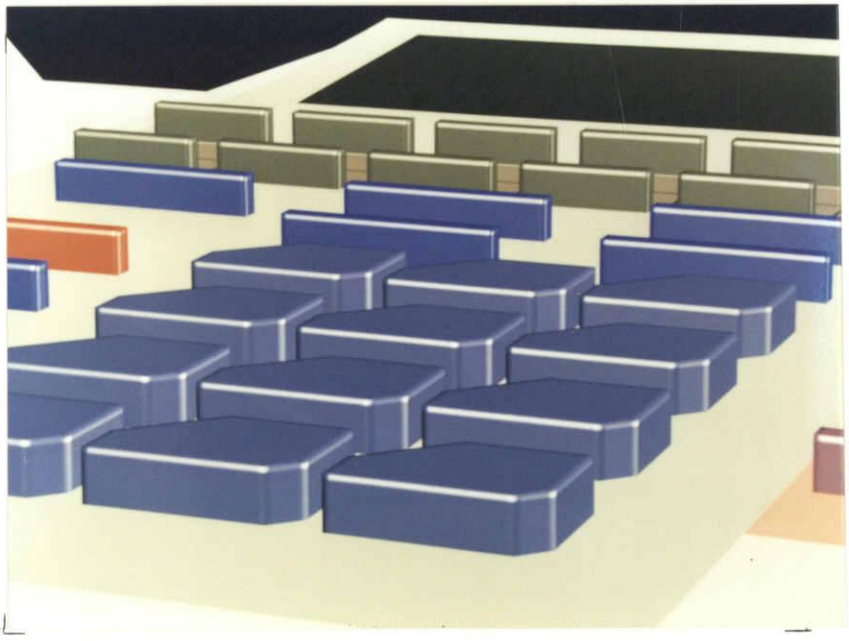
(a) 鋭い稜線

(b) 丸められた稜線

図 3-12: 電話機



(a) 鋭い稜線



(b) 丸められた稜線

図 3-13: ボタン部分の拡大図

図 3-14: 丸められた稜線を持つ３Ｄロゴ

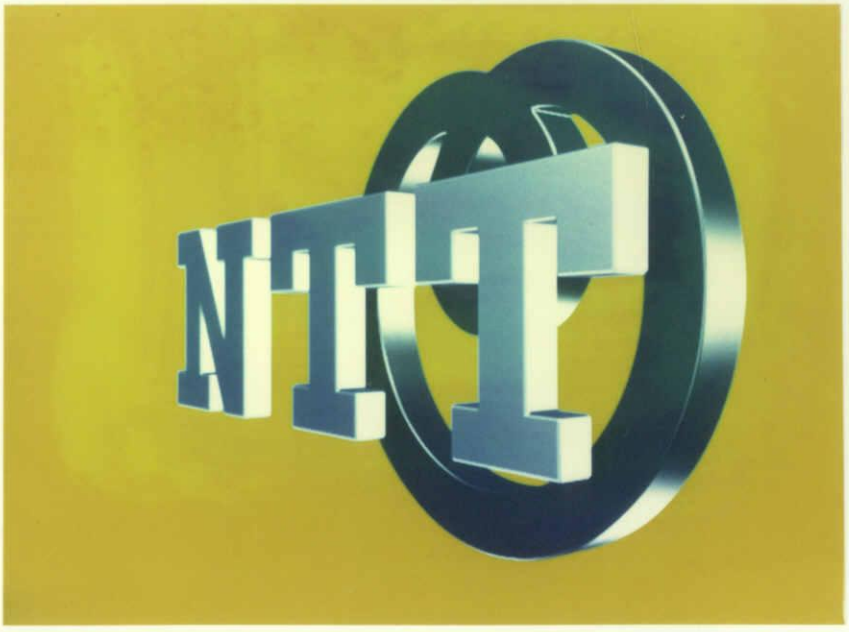


図 3-15: 丸められた稜線を持つ３Ｄシンボル

表 3-1: 実験画像の生成時間

実験画像	サイズ	画像生成時間 (sec)			
		精密レンダリング法			スキャンライン法
		積分あり	積分なし	時間差	
図 3-1 (P.66)	640 × 480	180	142	38	85
図 3-9 (P.79)	512 × 512	146	116	30	41
図 3-10 (P.79)	512 × 512	109	96	13	28
図 3-11 (P.80)	512 × 512	66	65	1	45
図 3-12 (P.81)	640 × 480	241	174	67	80
図 3-13 (P.82)	640 × 480	356	226	130	117

も長いものでは、画像生成時間の 1/3 が積分計算に費やされている。したがって、画像生成を高速化するには積分計算を短縮する必要がある。

本実験では、精密レンダリング法は通常のスキャンライン法の 4 倍程度の時間で画像を生成できた。したがって、スーパーサンプリングで輝度を精密に求めるよりはずっと高速である。

3.6 結言

高度のアンチ・エリアシングと正確なハイライト表示とを同時に実現する精密レンダリング法を報告した。精密レンダリング法は、直交スキャンライン法で 1 画素内のポリゴン領域を精密に求め、反射強度積分法でその輝度を解析的に計算する。直交スキャンライン法の詳細は 2章で報告したので、本章では反射強度積分法を報告した。

精密レンダリング法は曲面をポリゴンで近似する。丸められた稜線や頂点は一般に画素より細かいポリゴンで近似される。このため、エリアシングを防ぐには、各画素に投影されるポリゴンの領域を正確に求める必要がある。直交スキャンライン法は、画像を縦横 2 方向に走査することで、1 画素内のポリゴンの領域を精密に、かつ、効率よく決定できる。この結果、隠れ面消去におけるエリアシングを厳密に除去できる。

曲率の大きな曲面をポリゴンで近似すると、面法線の方向はポリゴンの頂点ごとに大きく異なる。したがって、正確な輝度を求めるには、ポリゴン内で反射光の強度を積分する必要がある。反射強度積分法はこの積分を極座標変換して計算する。ポリゴンの領域内で面法線が変化する方向領域と単位球面との交差領域が積分範囲になる。積分を簡略化するため、積分範囲を z 軸と単位球面との交点を含む単位球面上の三角形に分割する。積分を直接計算することは困難なため、被積分関数を Chebyshev 近似した後で代数積分する。近似誤差を画素の輝度値の量子化誤差より十分に小さくする

ことで正確な輝度が計算できる。

計算機実験により、精密レンダリング法は稜線の方向や位置によらず正しいハイライトを生成できることを示した。丸めによる陰も表示できた。画像を比較することで、丸められた稜線と頂点に表れるハイライトや陰が物体の実在感を高める上で重要であることを示した。ハイライトや陰を常に正確に計算する精密レンダリング法は、写実的な画像を生成する有効な手法である。

輝度の積分には多くの計算時間を要する。実験では画像生成時間の 1/3 以上を占める場合もあった。積分の高速化手法は 5章で報告する。

第 4 章

高精度面光源照明法

4.1 序言

本章では面光源照明法 (Area Light Illumination Method) を報告する。本手法は面光源で照明された物体の輝度を解析的に計算することにより、精密にアンチ・エリアシングされた写実的な画像を生成する。

日常生活ではさまざまな種類の光源が使われているが、その多くは固有の大きさを持っている。これらの光源は面光源で近似できる。したがって、コンピュータグラフィックスの重要な課題の一つである写実的な画像の生成には、面光源による照明が不可欠である。

物体の明るさは、物体表面での光の反射をシミュレートすることで計算できる。コンピュータグラフィックスでは、物体表面の反射を拡散反射成分と鏡面反射成分とに分けることが一般的である。このうち、拡散反射成分は光源の方向だけに依存するので、比較的簡単に計算することができる。面光源で照明される場合でも拡散反射成分は解析的に計算できる [27]。また、完全鏡面反射という特殊な場合に限ってはマッピングで表現できる。しかし、一般的な鏡面反射成分を計算するには面光源を点光源の集合で近似する必要があった。

面光源を点光源に分割すると、鏡面反射成分も簡単に計算できる [43]。点光源用に開発された多くの反射モデルも利用できる。ただし、エリアシングが発生しないように光源を十分細かく分割しなければならないので、計算時間が長大になる。エリアシングの程度は鏡面反射の鋭さ, 視点の位置, 光源と物体との距離などによりさまざまに異なるので、エリアシングを厳密に除去することは困難である。

エリアシングを確実に解消するには鏡面反射成分も解析的に計算する必要がある。しかし、一般的な鏡面反射強度は光源方向と視点方向の両者に依存した複雑な積分式で与えられるので、その解析的な計算手法はこれまで求められていなかった。本論文で報告する面光源照明法は一般的な鏡面反射を正確に計算できる初めての手法である。

面光源照明法は、完全拡散放射特性を持つ多角形光源で照明し、入反射のエネルギー

を保存するように改良したPhongのモデルを用いて輝度計算する。面光源照明下での物体の輝度値を正確に求めるには光源上の微小領域が作る輝度を光源内で積分する必要がある。しかし、鏡面反射の強度は光源・視点・反射点(輝度計算をおこなう点)の位置関係で定まるので、デカルト座標系では被積分関数と積分範囲が共に複雑な関数になる。このため、このまま代数積分することは困難である。

面光源照明法では、はじめに、輝度を計算する点を原点として面光源内の積分を極座標系の二次元積分に変換する。原点を中心とする単位球面上に光源を投影した領域が積分範囲になる。次に、積分範囲をz軸と単位球面との交点を頂点とする球面上の三角形に分割する。この分割により、各三角領域では2変数の一方を代数的に積分できる。他方はそのままでは代数積分することが困難なため、被積分関数をChebyshev多項式で近似してから積分する。近似誤差が画素の輝度値の量子化誤差より十分に小さくなるように多項式の次数を選べば、正確な輝度値が計算できる。

以下、4.2節で従来の輝度計算手法を、4.3節で新しく提案する手法の詳細を、4.4節で計算機実験の結果を示す。

4.2 従来の領域光源による照明手法

本節では、従来より提案されている、線光源および面光源で照明された物体の輝度計算手法を概説する。

4.2.1 線光源

西田らは蛍光灯のような線光源で照明された場合の輝度計算手法を提案した[29]。この手法ではLambertの法則に従う放射特性をもつ光源(これを完全拡散光源と呼ぶ)を仮定している。拡散反射成分は、線光源と物体表面の関係を平行・垂直・その他の場合の3つに分けて、それぞれの場合で解析解を得ている。しかし、鏡面反射成分については、サンプリングにより線光源を点光源の列で近似して計算している。

PoulinらはChebyshev多項式近似を用いて拡散反射成分と鏡面反射成分の両方を解析的に計算する手法を提案した[34]。ただしこの手法は、線光源を点光源の列で近似した場合と同様の、放射光の強度が方向に依存しない光源(これを均等放射光源と呼ぶ)を仮定している。このため、線光源のモデルとしては完全拡散光源に比べて劣っている。

4.2.2 面光源

西田らは完全拡散反射物体に限定した輝度の解析解を提案している[27]。この手法は拡散反射を厳密に計算できるが、鏡面反射については考慮していない。ほとんどの

物体は拡散反射特性と鏡面反射特性とを合わせ持つので、完全拡散反射物体のみで記述できるシーンは極めて限定される。したがって、この手法を適用できる範囲はかなり限られる。

面光源は点光源の集合で近似できる。実際、 Verbeck らにより、面光源を格子状に分割して点光源で近似する画像生成システム [43] が提案されている。このシステムでは面光源は点光源のアレイでシミュレートされる。彼らのシステムには、点光源用に提案されてきた複雑な反射モデルを利用できる利点がある。しかし、光源のサンプリング間隔が広いと、照度が不連続になるなどのエリアシングが生じる。エリアシングの程度は、光源の大きさや物体と光源との距離, 反射の鋭さなどによりさまざまに異なる。このため、エリアシングを完全に除去するには極めて多数の点光源が必要で、計算量が増大する。

radiosity 法 [16][30] に光線追跡法 [49] を組み合わせた手法 [44] や distributed ray-tracing 法 [9] などの手法は光源方向を複数の点でサンプリングするため、面光源の輝度を計算できる。また、 radiosity 法に hemi-cube[18] を付加した手法 [36][39] のように反射の方向を格子状に分割して反射強度を計算する手法も面光源を扱うことができる。しかし、現実的な計算時間で画像を生成しようとすると、それほど細かくサンプリングすることはできない。このため、鋭い反射特性を持つ物体ではエリアシングが発生する。

4.3 解析的輝度計算手法

輝度計算で発生するエリアシングを防ぐには、拡散反射成分と鏡面反射成分を共に解析的に計算する必要がある。本節では、先ず、面光源で照らされた物体の輝度の解析解を導出する。次に、その具体的な計算手法を提案する。

4.3.1 面光源

図 4-1の面光源を単純な点光源に分割する。図 4-1の点 A と B は面光源の中心から等距離にある。点光源は全方向に均等に光を放射するため、光源から到達する光のエネルギーは点 A と B でほぼ等しくなる。しかし、光源を斜めから見る点 B では正面からみる点 A に比べ光源の見込み角度が小さい。点 B では点 A に比べて狭い領域から同じ量のエネルギーが入射するので、点 B から見た光源は点 A から見た光源より明るくなる。これは実際の面光源の見え方とまったく異なる。

そこで、光源を多角形の完全拡散面光源 (Lambertian distributed light source) であると仮定する。図 4-1に示すように、光源の面法線を N、物体の方向を L で表す。完全拡散光源は Lambert の法則に従って光を放射する。すなわち、 L 方向に放射さ

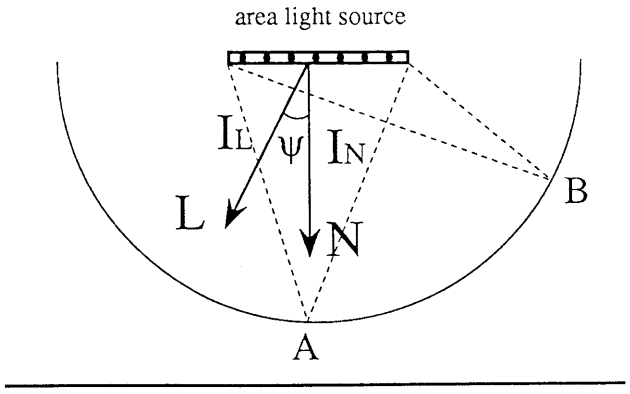


図 4-1: 完全拡散光源

れる光束の強度 I_L は N 方向に放射される光束の強度 I_N を用いて

$$I_L = (L \cdot N) I_N \tag{4.1}$$

で与えられる。この仮定では、斜めから見ても光源が明るくなることはない。完全拡散面光源による近似は点光源のアレイによる近似より優れている。

面光源・視点・反射面を図 4-2のように配置する。完全拡散光源であれば、光源上の微小領域 dS から輝度を計算したい点 P(以下、反射点と呼ぶ) に入射する光の強度 dI_i は

$$dI_i = I_o \frac{(N_Q \cdot L_Q)}{r^2} dS = I_o \frac{\cos\psi}{r^2} dS \tag{4.2}$$

で与えられる。式中の I_0 は単位面積の光源がその面法線の方向に放射する光の強度 (光束の密度) である。図 4-2中の記号は表 4-1で定義される。

点 P から dS を見た立体角を $d\omega$ とすると、$d\omega$ と dS との関係は

$$\cos\psi \, dS = r^2 d\omega \tag{4.3}$$

で与えられる。従って式 4.2は

$$dI_i = I_o d\omega \tag{4.4}$$

で置き換えられる。式 4.4は、入射光の強度は反射点 P から見た立体角に比例することを示している。

4.3.2 反射モデル

Phong の反射モデル

鏡面反射を表現するために Phong[32], Blinn[3], Cook-Torrance[9] などの反射モデルが用いられる。なかでも Phong のモデルは鏡面反射強度が視線の正反射方向ベク

表 4-1: 記号の定義

E	:	視点の方向を示す単位ベクトル
E_r	:	E の鏡面反射方向を示す単位ベクトル
L_P	:	光源の方向を示す単位ベクトル
L_Q	:	点 Q から点 P へ向かう単位ベクトル $(= -L_p)$
N_P	:	点 P での反射面の面法線ベクトル
N_Q	:	点 Q での光源の面法線ベクトル
ψ	:	N_Q と L_Q のなす角度
r	:	点 P と点 Q の間の距離
S	:	多角形光源
$V_1 \sim V_4$	:	多角形光源の頂点
dS	:	S 上の微小領域
$d\omega$	:	点 P から dS を見た立体角
dI_i	:	微小領域 dS から入射する光の強度
I_o	:	S の単位面積から面法線方向に放射される光の強度
I_i	:	入射光の強度
I_r	:	反射光の強度
I_d	:	拡散反射光の強度
I_s	:	鏡面反射光の強度
k_d	:	拡散反射の寄与率
k_s	:	鏡面反射の寄与率 $(k_d + k_s = 1)$
R_d	:	拡散反射率
R_s	:	鏡面反射率
n	:	鏡面反射指数 (鏡面反射の鋭さを表す指数)

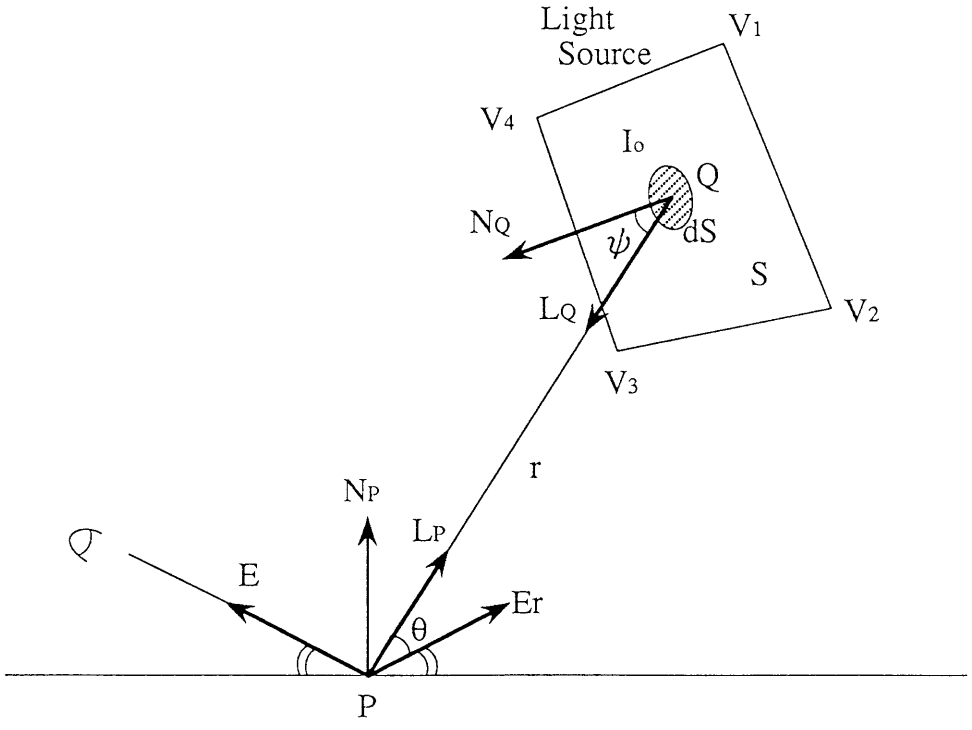


図 4-2: 照明モデル

トル (図 4-2の E_r) と光源方向ベクトル (図 4-2の L_p) とのなす角度だけで決まり、反射モデルとして比較的簡単であるにもかかわらず、良好な結果を与える。そこで、ここでは Phong のモデルを採用することにした。Phong のモデルを数式で表すと、

$$I_r = k_d I_d + k_s I_s, \tag{4.5}$$

$$I_d = R_d (N_P \cdot L_P) I_i, \tag{4.6}$$

$$I_s = R_s (E_r \cdot L_P)^n I_i \tag{4.7}$$

となる。ここで、式中の記号は表 4-1で定義される。k_d, k_s は拡散・鏡面反射の寄与率を与え、

$$k_d + k_s = 1, \quad 0 \leq k_d \leq 1, \; 0 \leq k_s \leq 1 \tag{4.8}$$

の関係がある。

図 4-3に示すように、面法線とのなす角度が ε である直線を面法線の回りに回転してできる円錐の底面 S から、式 4.4で定義される光線が反射点 P へ入射する場合を考える。面法線方向への反射強度は式 4.7を用いて、

$$\begin{aligned} I_s &= \int_0^{2\pi} \int_0^{\varepsilon} R_s I_o \cos^n\theta \; \sin\theta \, d\theta \, d\phi \\ &= R_s I_o \frac{2\pi}{n+1} (1 - \cos^{n+1}\varepsilon), \end{aligned} \tag{4.9}$$

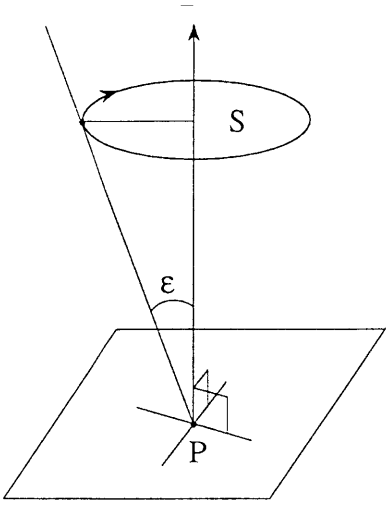


図 4-3: 鉛直方向からの入射光

で与えられる。$k_s = R_s = 1$ の場合、Phong のモデルでは n が大きくなるに従って反射が鋭くなり、n が無限大で完全鏡面となるはずである。しかし、式4.9は、n とともに反射強度が減少し、ついには 0 となることを示している。これは明らかに事実と矛盾している。原因は Phong のモデルが点光源 (離散系) で定義されていることにある。面光源 (連続系) で使用するには以下の補正が必要になる。

Phong のモデルの改良

法線方向から強度 I_i の平行光線が入射する場合を考える。鏡面反射光の総エネルギーは式4.7を全ての反射方向で積分して求められる。P での面法線 N_p を z 軸に選んだ極座標系で I_s を積分すると、

$$\begin{aligned} \int_0^{2\pi}\int_0^{\pi/2} I_s \sin\theta \, d\theta \, d\phi &= R_s I_i \int_0^{\pi/2} \cos^n\theta \, \sin\theta \, d\theta \int_0^{2\pi} d\phi \\ &= \frac{2\pi}{n+1} R_s I_i, \end{aligned} \tag{4.10}$$

となる。鏡面反射のみ ($k_s = 1$) で完全反射 ($R_s = 1$) の理想物体では、エネルギー保存則により反射エネルギーの総和は入射エネルギー I_i と等しくなければならない。この関係を満たすには、式4.7の右辺を $(n+1)/2\pi$ 倍した

$$I_s = \frac{n+1}{2\pi} R_s \left(E_r \cdot L_P\right)^n I_i \tag{4.11}$$

を用いて鏡面反射強度を積分する必要がある。式4.11を用いると式4.9は

$$I_s = R_s \, I_o(1 - \cos^{n+1}\varepsilon) \tag{4.12}$$

となり、前述の矛盾は生じない。

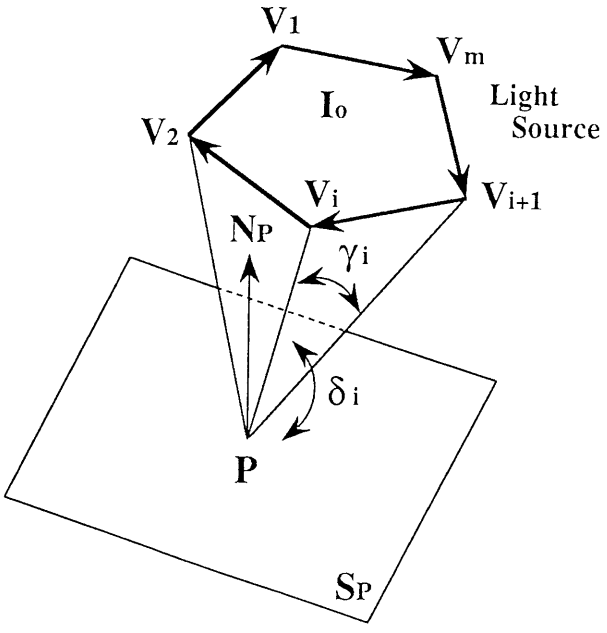


図 4-4: 拡散反射成分の積分

同様に、面法線方向を z 軸として拡散反射強度を半空間で積分すると、

$$\begin{aligned}\int_0^{2\pi}\int_0^{\pi/2} I_d \sin\theta\, d\theta\, d\phi &= R_d I_i \int_0^{\pi/2} \cos\theta\ \sin\theta\, d\theta \int_0^{2\pi} d\phi \\ &= \pi R_d I_i, \end{aligned} \tag{4.13}$$

となる。$k_d = R_d = 1$ の理想的な拡散反射物体のときにエネルギーが保存されるように式 4.6を改良して、

$$I_d = \frac{1}{\pi} R_d (N_P \cdot L_P) I_i \tag{4.14}$$

を得る。

4.3.3 拡散反射成分の計算

式 4.14で与えられる拡散反射成分の計算には西田らの手法 [27] を用いればよい。図 4-4に示すように多角形の面光源が与えられるものとする。面光源の頂点 $\{V_i\}_{i=1,\ldots,m}$ を表からみて時計回りに定義する。点 P での拡散反射強度 I_d は、ベクトル PV_i と PV_{i+1} のなす角度 γ_i、平面 S_P と三角形 PV_iV_{i+1} のなす角度 δ_i を用いて、

$$I_d = \frac{R_d I_o}{2\pi} \sum_{i=1}^{m} \gamma_i \cos\delta_i \tag{4.15}$$

で与えられる。

4.3.4 鏡面反射成分の計算

本節で拡散反射強度を解析的に求める手順を示す。

積分式の導出

微小立体角 $d\omega$ から入射する光線による鏡面反射強度 dI_s は、式4.11に式4.4を代入することで求められる。面光源全体による強度 I_s は、dI_s を面光源内で積分する

$$I_s = \frac{n+1}{2\pi} R_s I_o \iint_{\{V_i\}_{i=1,\ldots,m}} (E_r \cdot L_P)^n \, d\omega \tag{4.16}$$

で計算できる。

式4.16の積分を実行すれば解析解が求まる。しかし、デカルト座標系においては、

(1) 内積項 $E_r \cdot L_p$ が x, y, z の関数となるので被積分関数が複雑になる,

(2) 積分範囲が x, y, z の関数のため、変数分離が難しい,

ため、式4.16を直接積分することは困難である。

4.3.5 球面積分

Phong の反射モデルでは、鏡面反射強度は式4.11に示されるようにベクトル E_r に対して回転対称である。この点に着目して、E_r を z 軸に選んで極座標変換を行う。z 軸からの角度を θ で表記すると、$(E_r \cdot L_p)$ は $\cos\theta$ と書ける。

図4-5に示すように、多角形光源 $\{V_i\}_{i=1,\ldots,m}$ を反射点 P を中心とする半径1の単位球面上に投影して領域 Ω を求める。Ω は P から V_i に向かう半直線と単位球との交点 W_i で囲まれる領域である。境界線 W_iW_{i+1} は大円 (点 P を含む平面と単位球面との交線) の一部である。完全拡散光源では、被積分関数が距離 r の項を含まないため、多角形光源内での積分は Ω 内での積分と等価になる。Ω を積分領域に選ぶと、多角形光源による鏡面反射の強度は

$$I_s = \frac{n+1}{2\pi} R_s I_o \iint_{\Omega} \cos^n\theta \, \sin\theta \, d\theta \, d\phi \tag{4.17}$$

で与えられる。式中の ϕ は z 軸回りの回転角を表す。

積分領域の三角形分割

積分を簡単にするため、多角形で与えられた積分領域を単位球面上の三角形に分割する。z 軸と単位球面の交点を W_z とする。$i = 1, \ldots, m$ で W_z, W_i, W_{i+1} を頂点とする単位球面上の三角領域を $\Delta(W_z, W_i, W_{i+1})$ で記述する。ただし、$W_{m+1} = W_1$ である。符号関数 $F(W_z, W_i, W_{i+1})$ を

$$F(W_z, W_i, W_{i+1}) = \begin{cases} 1 & : W_z, W_i, W_{i+1} \text{ が } P \text{ から見て時計回りの場合} \\ -1 & : \text{その他の場合} \end{cases} \tag{4.18}$$

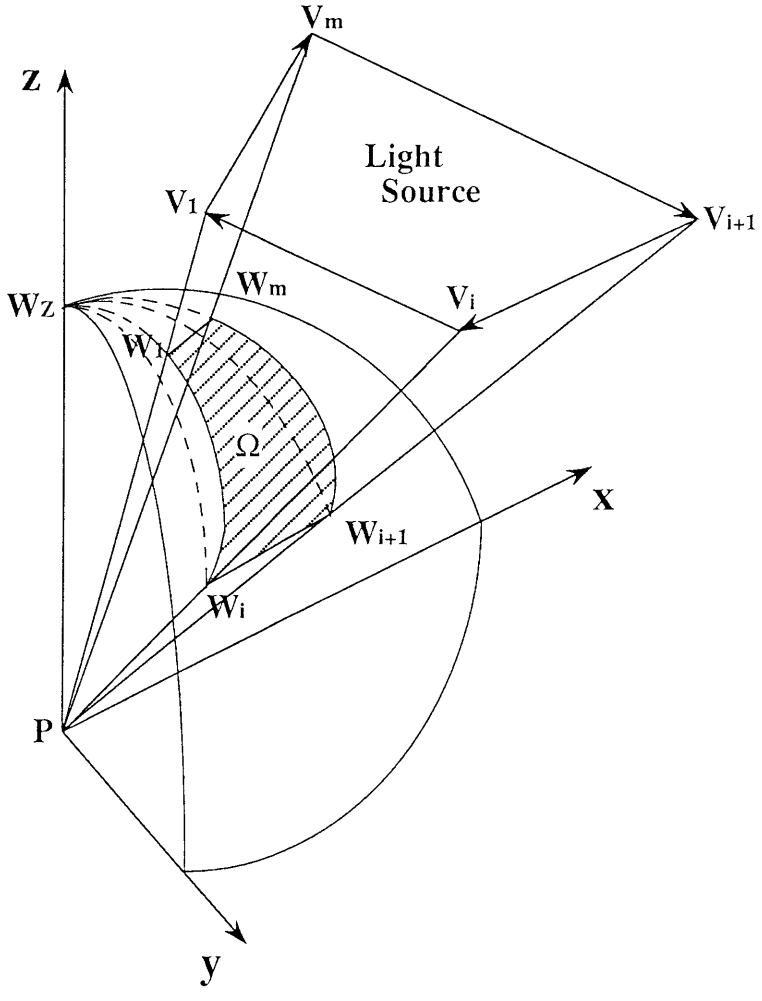


図 4-5: 単位球面への投影

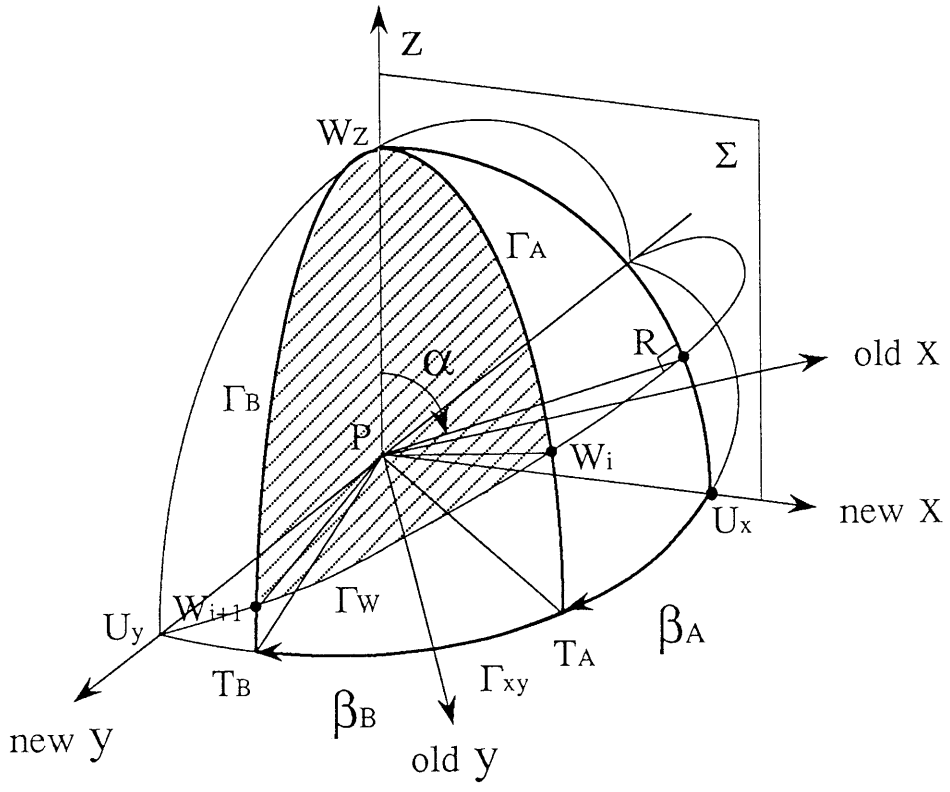


図 4-6: 三角形内での鏡面反射成分の積分

で定義すると、領域 Ω は

$$\Omega = \sum_{i=1}^{m} F(W_z, W_i, W_{i+1}) \Delta(W_z, W_i, W_{i+1}) \tag{4.19}$$

のように分割できる。

三角形領域 $\Delta(W_z, W_i, W_{i+1})$ における式 4.17の積分を

$$G(W_z, W_i, W_{i+1}) = \frac{n+1}{2\pi} \iint_{\Delta(W_z, W_i, W_{i+1})} \cos^n\theta \sin\theta \, d\theta \, d\phi \tag{4.20}$$

で定義する。鏡面反射強度は関数 F と関数 G を用いて

$$I_s = R_s I_o \sum_{i=1}^{m} F(W_z, W_i, W_{i+1}) G(W_z, W_i, W_{i+1}) \tag{4.21}$$

で記述できる。従って、式 4.20を計算すれば鏡面反射強度が求められる。

三角形領域での積分

式 4.20を積分するため、図 4-6 に示すように座標系をとる。先ず、3 つの大円, Γ_W, Γ_A, Γ_B を定義する。ここで、大円とは単位球の原点 P を通る平面と単位球面との交

線である。Γ_W は多角形光源の頂点 W_i と W_{i+1} を、Γ_A は W_z と W_i を、Γ_B は W_z と W_{i+1} を通る大円である。積分域は大円 $\Gamma_W, \Gamma_A, \Gamma_B$ で囲まれた領域となる。

xy 平面と $\Gamma_W, \Gamma_A, \Gamma_B$ との交点をそれぞれ U_y, T_A, T_B とする。点 P を通りベクトル PU_y に垂直な平面を Σ、Σ と Γ_W との交点を R とする。y 軸が PU_y と一致するまで座標系を z 軸の回りに回転する。Σ が新しい xz 面になる。

z 軸とベクトル PR のなす角度を α、新しい x 軸とベクトル PT_A, PT_B のなす角度をそれぞれ β_A, β_B とする。Γ_W 上の点は ϕ をパラメータとして $(1, \theta_W(\phi), \phi)$ で表せる。$\theta_W(\phi)$ は

$$\cos\theta_W(\phi) = \frac{\cos\phi}{\sqrt{\tan^2\alpha + \cos^2\phi}} \tag{4.22}$$

で与えられる。従って、式4.20は

$$G(W_z, W_i, W_{i+1}) = \frac{n+1}{2\pi}\int_{\beta_A}^{\beta_B}\int_0^{\theta_W(\phi)} \cos^n\theta \, \sin\theta \, d\theta \, d\phi \tag{4.23}$$

に変換される。式4.23を θ で代数積分して

$$\begin{aligned} G(W_z, W_i, W_{i+1}) &= \frac{n+1}{2\pi}\int_{\beta_A}^{\beta_B}\left[\frac{-1}{n+1}\cos^{n+1}\theta\right]_0^{\theta_W(\phi)} d\phi \\ &= \frac{1}{2\pi}\{(\beta_B - \beta_A) - (H(n,\alpha,\beta_B) - H(n,\alpha,\beta_A))\} \end{aligned} \tag{4.24}$$

を求める。ここで、関数 H は

$$H(n,\alpha,\beta) = \int_0^{\beta}\left(\frac{\cos^2\phi}{\tan^2\alpha + \cos^2\phi}\right)^{\frac{n+1}{2}} d\phi \tag{4.25}$$

で定義される。

多項式近似による積分

上記式4.25は、3.4.4節と全く同様に、被積分関数を Chebyshev 多項式で近似したあと代数積分される [34]。式4.25の被積分関数の $[0, \pi/2]$ における m 次 Chebyshev 近似は

$$\left(\frac{\cos^2\phi}{\tan^2\alpha + \cos^2\phi}\right)^{\frac{n+1}{2}} \approx \sum_{i=0}^{m} C_i(n,\alpha)\, T_i(\frac{4\phi}{\pi} - 1) \tag{4.26}$$

で記述できる [42]。ここで、$T_i(t)$ は3.4.4節の式3.18で定義される i 次 Chebyshev 多項式、係数 $C_i(n,\alpha)$ は n と α のみの関数である。$T_i(4\phi/\pi - 1)$ は ϕ の i 次の多項式であるから、たやすく代数積分できる。積分結果を

$$\int_0^{\beta} T_i(\frac{4\phi}{\pi} - 1)\, d\phi = S_i(\beta) \tag{4.27}$$

で表すと、式4.25は

$$H(n,\alpha,\beta) \approx \sum_{i=0}^{m} C_i(n,\alpha) S_i(\beta) \tag{4.28}$$

となる。

3.4.4節と同様に、面光源照明法でも10次のChebyshev多項式で近似する。この場合、近似誤差は4×10^{-4}以下になるので、輝度を計算するために数回の加減算を行っても計算誤差を1/256以下にできる。したがって、一般的な256階調のCG画像を表示するには十分な精度である。

4.4 実験と考察

4.4.1 鏡面反射の効果

鏡面反射の効果を示すため、本手法と拡散反射成分のみを計算する手法とを比較した。比較には天井に長方形の光源を配置した室内シーンを用いた。図4-7(a)は本手法で生成した画像で、拡散反射成分と鏡面反射成分の両方を含む。図4-7(b)には拡散反射成分のみを示した。図4-7(a)では壁面や床面に鏡面反射によるハイライトが現れ、写実性に富んだ画像となっている。しかし、図4-7(b)ではハイライトが現れないため、のっぺりした画像となる。

視点を左に移動して図4-8を生成した。鏡面反射強度は視点・反射点・光源の三者の位置関係で定まる。このため、図4-7(a)と図4-8(a)を比べると、視点の移動につれ床面上のハイライトの位置も移動する。しかし、拡散反射強度は光源と物体の位置関係のみで定まるため、図4-7(b)と図4-8(b)で床面の上の明るく照らされる位置は変化しない。

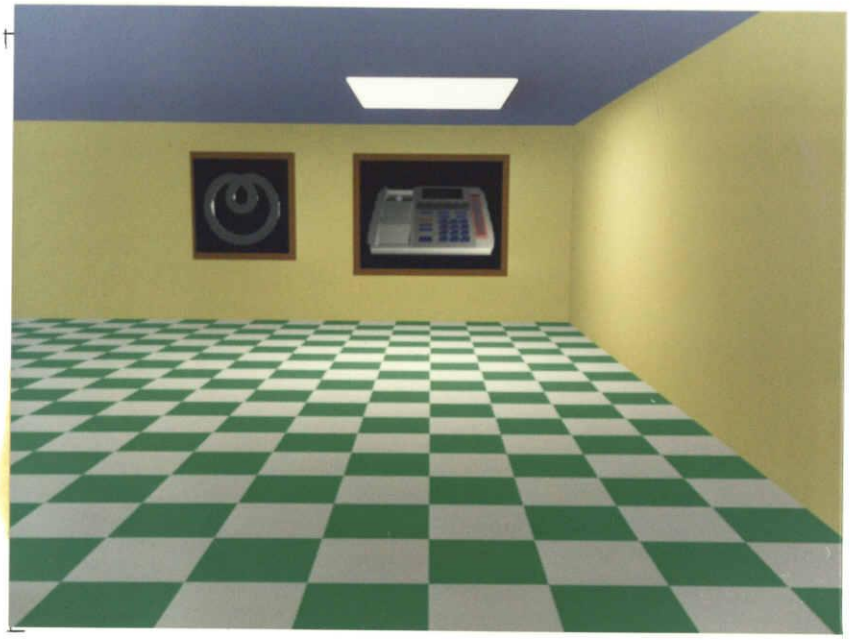
この実験から明らかなように、ハイライトは画像のリアリティを高める上で重要な働きをする。ハイライトは鏡面反射により発生する。鏡面反射強度は光源の方向に敏感に依存するが、本手法は鏡面反射強度を解析的に計算するため、常に正確なハイライト表示が可能である。

4.4.2 鏡面反射の鋭さ

図4-9に示すように、星型光源で傾斜した床を照らす。床面の鏡面反射指数nを変えて反射パターンを比較した。$n = 4, 16, 64$の場合の結果を図4-10に示す。nが大きくなるにつれて反射が鋭くなるので、映り込む光源の輪郭がくっきりする。鏡面反射指数を変えることで、完全鏡面反射から完全拡散に近いぼんやりした反射まで自由に表現できる。

1枚の画像の中でも、上側と下側で映り込んだ光源の輪郭の鋭さが異なる。たとえば、図4-10(c)では、星型光源の上向きの2つの角が鋭く映り込むのに比べ、下向きの角は鈍く映り込む。これは、図4-9に示されるように床と光源までの距離の違いによる。この効果は単純なマッピングでは表現できない。

(a) 鏡面反射成分と拡散反射成分



(b) 拡散反射成分のみ

図 4-7: 長方形光源による照明

(a) 鏡面反射成分と拡散反射成分

(b) 拡散反射成分のみ

図 4-8: 視点を左に移動

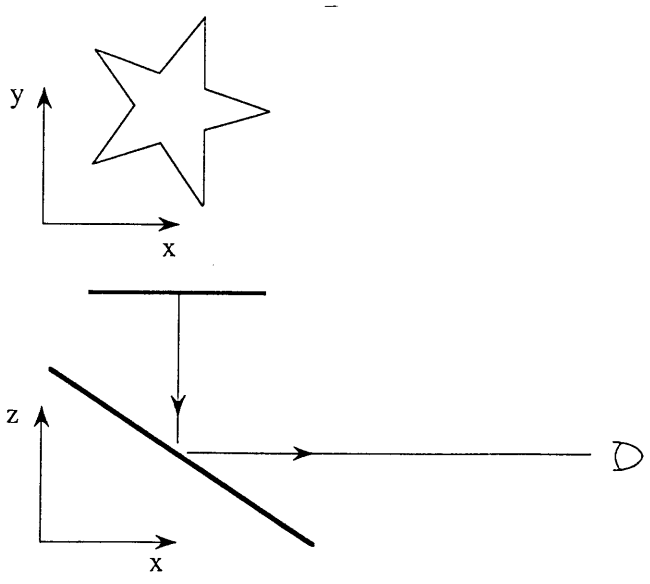


図 4-9: 星型光源と反射面

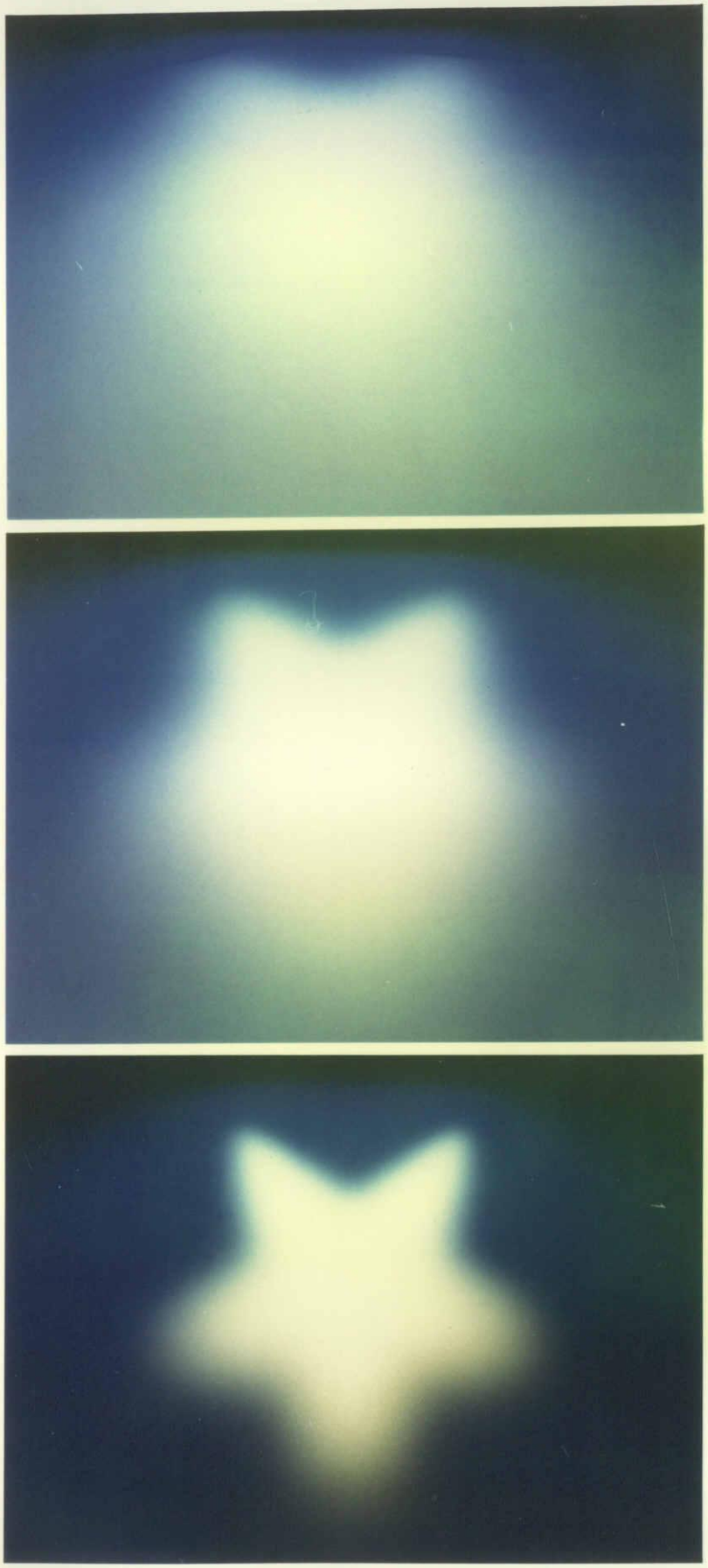
4.4.3 Dull reflection

画像のリアリティを高める上で重要な働きをする dull reflection の例を図 4-11に示す。この図はティーポットが青・白・赤の帯状光源で照明されている。ティーポットと背後の壁面に光源が映り込んでいるが、反射が鈍いため輪郭がぼけている。鏡面反射指数と光源からの距離の違いにより、ポットと背後の壁面とでは異なる反射パターンが表示される。特に壁面では色のにじみの効果が見られる。これらの dull reflection による効果は単純なマッピングでは表現できない。

4.4.4 点光源との比較

図 4-11のティーポットを白色の正方形光源で照明して、面光源照明法と面光源を点光源のアレイで近似する従来手法とを比較した。近似に用いる点光源の数を変えて画像生成実験を行い、計算時間を測定した。結果を図 4-12に示す。図 4-12の横軸は正方形光源の 1 辺の分割数を示す。横軸の数字の 2 乗が近似に用いた点光源の数になる。縦軸は画像生成に要する時間を分単位で示す。

図 4-12の点線は面光源照明法を用いた場合の画像生成時間を示す。この実験では、面光源照明法は正方形光源を 7×7 点光源で近似する場合と同じ時間で画像生成できた。比較のため、面光源照明法で生成した実験画像を図 4-13に、7×7 点で近似した場合の画像を図 4-14に示す。7×7 分割では近似が粗いため、ハイライトの中に光源のサンプリングパターンが現れている。



(a) $n = 4$

(b) $n = 16$

(c) $n = 64$

図 4-10: 鏡面反射の鋭さ

図 4-11: Dull reflection(ぼんやりした鏡面反射)

画像 $I_1(i,x,y)$ と $I_2(i,x,y)$ の差画像 $D(x,y)$ を

$$D(x,y) = \max[|\ I_1(i,x,y) - I_2(i,x,y)\ |]_{i=R,G,B} \tag{4.29}$$

で定義する。面光源照明法で生成した図 4-13と面光源を 7×7 点光源で近似して生成した図 4-14の差画像を図 4-15に疑似カラー表示する。ハイライト部分では輝度値の差が 16 階調以上もある。目視評価では 13×13 点でサンプリングパターンがほとんど見分けられなくなった。この画像と面光源照明法で生成した画像との差画像を図 4-16に示す。図 4-15と比べると大幅に差が小さくなっている。実際には、まだサンプリングによる照度パターンが残っているが、差は 4 階調よりも小さい。実用上は十分に小さい差である。図 4-12のグラフに示されるように、13×13 点に分割する場合の画像生成時間は面光源照明法の 3 倍長くなる。

物体の反射特性が鋭い場合 (n が大きい場合) には、反射強度は光源方向の変化につれて急峻に変化する。このため、光源を細かくサンプリングしないとエリアシングが発生する。面光源照明法では反射強度の積分時間は反射面の特性によらず一定である。従って、反射が鋭い場合には、面光源照明法がより有利になる。

4.4.5 立体光源への拡張

完全拡散光源の場合には、照度は反射点における光源を見込む方向領域から計算できる。方向領域は反射点から見た光源の輪郭線だけを用いて決定できるので、立体光

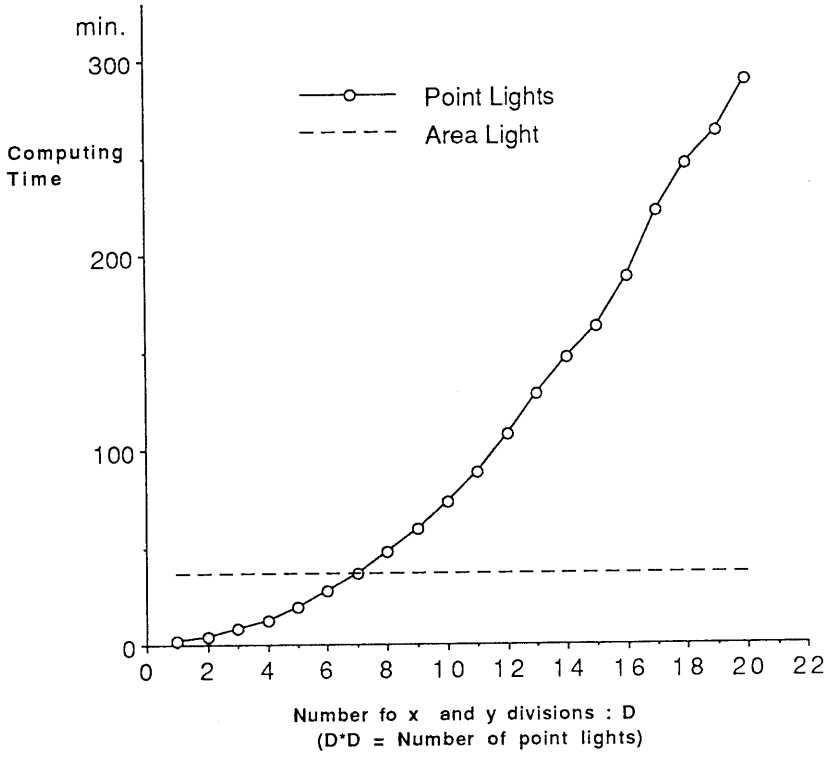


図 4-12: 点光源近似の画像生成時間

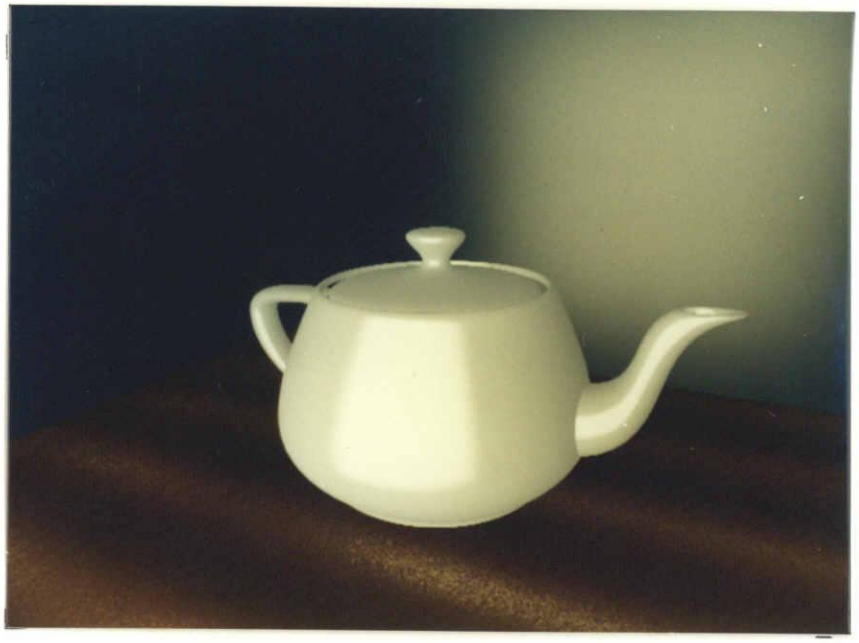


図 4-13: 正方形の面光源による照明

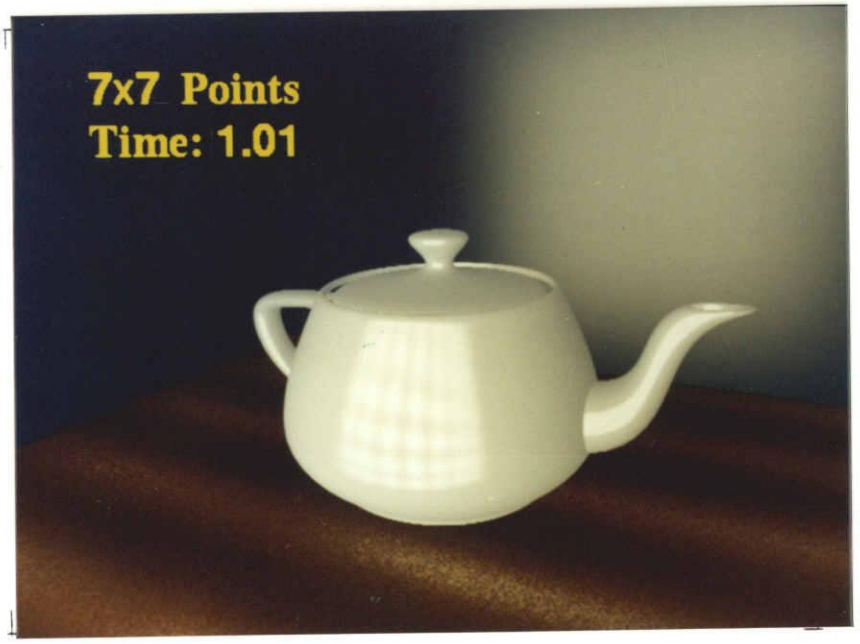


図 4-14: 7×7 点光源近似

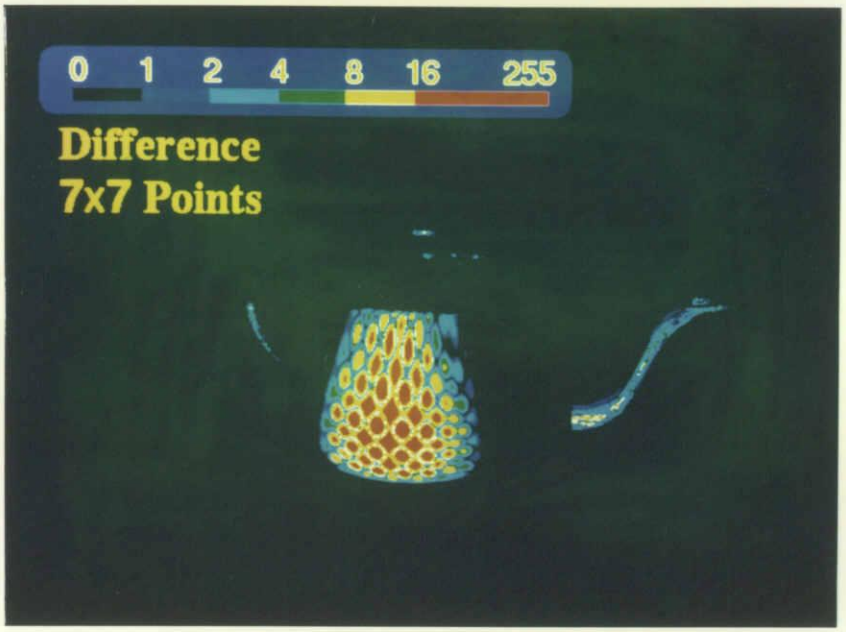


図 4-15: 面光源照明と 7 × 7 点光源近似との差画像

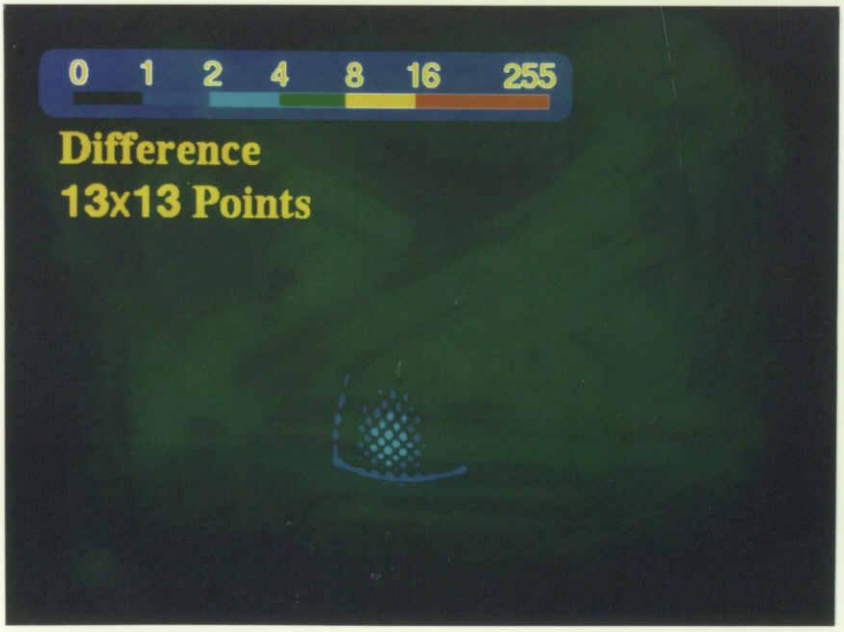


図 4-16: 面光源照明と 13 × 13 点光源近似との差画像

源の輪郭線が多角形であれば、本論文で提案した手法が適用できる。すなわち、多面体光源までは問題なく拡張できる。

凸多面体の場合には輪郭線は簡単に求まる。先ず、視点から見える多角形を選び出す。これらの外形線のうちで一度しか表れないものを取り出し時計回りにつなぐと、光源の輪郭線となる。凹多面体の場合は光源が自らを隠すことがあるので、隠れ面消去して可視面を求める必要がある。

曲面の光源は多面体近似したあと輪郭線を求めることで対処できるが、多面体近似による誤差が発生する。また、光源を構成する面の数が多くなるので、輪郭線は多くの頂点を有する多角形となる。このため、計算量が増大する。

4.4.6 付影処理

今後、利用分野を広げるには付影処理が必要である。原理的には、反射点から見える光源の領域を求めれば良いので、光源を手前の物体でクリッピングする前処理で実現できる。しかし、単純なクリッピングによる付影処理は極めて多くの処理時間を必要とする。処理の高速化も含め、今後の研究課題である。

4.5 結言

面光源で照明される条件のもとでの解析的な輝度計算手法を提案した。物体表面での反射は鏡面反射成分と拡散反射成分とに分けられるが、従来の輝度計算手法では拡散反射成分しか解析的に計算できなかった。鏡面反射成分は面光源をサンプリングして点光源のアレイで近似して計算していたため、エリアシングが発生した。このエリアシングを完全に防ぐには、面光源による輝度を解析的に計算する必要がある。本研究では鏡面反射成分と拡散反射成分の両方を解析的に計算する手法を提案した。本手法により、一般的な反射特性を持つ物体を精密に表示できるようになった。

面光源照明法では、多角形の完全拡散光源で照明し、入反射のエネルギーが保存されるように改良したPhongの反射モデルを用いて輝度を計算する。Phongのモデルが回転対称であることに着目して、視線方向の正反射方向がz軸となるように極座標変換を行い、光源内での平面積分を球面積分に変換する。積分範囲は輝度を計算する点を原点とする単位球面に光源を投影した領域になる。積分範囲を単位球面上の三角形に分割して積分を簡単にした後で、二つの積分変数の一方を代数積分する。残りの変数は被積分関数を10次Chebyshev多項式で近似してから代数積分する。こうして求めた近似解の誤差は256階調の画像の量子化誤差より十分に小さいので、正確な輝度値が計算できる。

面光源照明法の有効性を示すため計算機実験を行った。本手法を用いることで鏡面

反射により生ずるハイライトを表示できた。また、鏡面反射指数を変えることで反射の鋭さを自由にコントロールできた。本手法は dull reflection の表示にも有効であった。これらの効果は写実的な画像を生成するために重要であるが、完全拡散反射しか扱えない従来の解析手法では表現できなかった。

点光源近似手法との比較実験では、計算速度と画質の両面で本手法が有利であった。同じ計算時間をかけてもより高品質の画像が生成できた。また、同じ品質の画像で比較すると生成時間を 1/3 に短縮できた。

実験結果から、本手法は写実的な画像を生成する上できわめて有効であると結論される。ただし、多項式近似により輝度を計算するため、画像生成に長い時間を必要とする。生成時間のほとんどは輝度計算のために費やされるので、生成時間を短縮するには積分を高速化する必要がある。次章では、テーブル参照を用いた近似積分手法を報告する。

第 5 章

高精度テーブル積分法

5.1 序言

3章で報告した反射強度積分法でも 4章で報告した面光源照明法でも単位球面上の三角形を積分範囲とする極座標積分が輝度計算の基本処理となっている。3章と 4章ではこの積分を Chebyshev 多項式で近似したため、計算に長い時間がかかった。本章では適応分割テーブルを参照することで積分を高精度かつ高速に計算する。

コンピュータグラフィックスでは、一般的に R(赤),G(緑),B(青) 各 8bit のデジタル画像を用いる。この場合、R,G,B のおのおのは 256 階調の異なる明るさしか表現できない。したがって、輝度 k が $0 \leq k \leq K$ の範囲にある画像を表示する場合、輝度の計算誤差が $K/256$ 以下ならば、計算誤差が量子化誤差より小さくなるので、正確な画像といえる。

この精度を確保するため、反射強度積分法の輝度計算式 (式 3.16) も面光源照明法の輝度計算式 (式 4.25) も、被積分関数を Chebyshev 多項式で近似したあと代数積分して近似解を求めた。Chebyshev 近似は誤差評価が容易なため、精度を保証した近似が簡単にできる。この近似式は鏡面反射指数 n と 2 つの角度 α, β をパラメータとして含むが、このうち n と α は Chebyshev 係数に含まれる。このため、積分の度に係数値を計算しなければならないので、輝度計算には長い時間が必要であった。

誤差を $K/256$ 以下にする近似はテーブル参照を用いても実現可能である。ただし、鏡面反射が鋭い場合には正反射方向から離れるにしたがい反射強度が急速に小さくなるので、パラメータの定義域を均等に分割するテーブルではテーブルサイズを極めて大きくしないと必要な精度が確保できない。そこで、本研究では、小さなテーブルで正確に輝度計算するため適応分割テーブルを用いる。このテーブルは、輝度の近似式の変化が大きいところでは標本点を細かく、小さいところでは粗く配置して、近似誤差を縮小する。

3 つのパラメータのうち、α と β は実数値をとるが n は自然数しかとらない。したがって、n に関する標本化誤差は発生しない。また、n は物体の材質ごとに与えられ

る定数なので、異なる物体でも材質が同じならば同一のテーブルで輝度計算できる。そこで、n の値ごとに α と β の 2 次元テーブルを作成し、画像生成前に画像中に現れる材質のテーブルだけを読み込む。これらの工夫により、テーブルが使用するメモリ量を大幅に削減し、実用的な大きさにできる。

積分値をテーブル参照で求めることで処理時間の大幅な短縮が可能となった。実験により、同じ精度の Chebyshev 近似積分と比べて 1/10 以下の時間で積分値を計算できることが示される。画像生成全体の時間も 40 ～ 60% に短縮される。

以下の節では、Chebyshev 近似による積分の問題点、適応分割テーブルを用いた積分法の詳細、精度と計算時間の評価について報告する。

5.2 輝度積分

面光源照明法 (4章) では画像生成時間のほとんどが、精密レンダリング法 (3章) でも 30% 程度の時間が輝度積分に費やされている。本節では前章で用いた Chebyshev 近似積分の問題点とテーブル参照による積分法を述べる。

5.2.1 Chebyshev 近似

反射強度積分法では式 3.15、面光源照明法では式 4.24で輝度を計算していた。これら 2 つの積分式は $(n+1)/2\pi$ 倍されていることを除いて全く同じである。そこで、以下では、式 4.24の積分について議論する。式 4.24をあらためて次のように記述する。

$$\begin{aligned} G(n,\alpha,\beta_A,\beta_B) &= \frac{n+1}{2\pi}\int_{\beta_A}^{\beta_B}\left[\frac{-1}{n+1}\cos^{n+1}\theta\right]_0^{\theta_W(\phi)} d\phi \\ &= \frac{1}{2\pi}\left\{(\beta_B-\beta_A)-\int_{\beta_A}^{\beta_B}\left(\frac{\cos^2\phi}{\tan^2\alpha+\cos^2\phi}\right)^{\frac{n+1}{2}} d\phi\right\}. \end{aligned} \quad (5.1)$$

4章では、輝度積分を

$$H(n,\alpha,\beta)=\int_0^{\beta}\left(\frac{\cos^2\phi}{\tan^2\alpha+\cos^2\phi}\right)^{\frac{n+1}{2}} d\phi \approx \sum_{i=0}^{m} C_i(n,\alpha)S_i(\beta) \quad (5.2)$$

のように Chebyshev 多項式で近似してから代数積分していた。ここで $C_i(n,\alpha)$ は i 次 Chebyshev 係数で、$S_i(\beta)$ は 3.4.4節の式 3.18で定義される i 次 Chebyshev 多項式を $[0,\beta]$ で積分して得られる多項式である。係数 $C_i(n,\alpha)$ のパラメータ α は $[0,\pi/2]$ の範囲の任意の実数をとるため、すべての場合の係数値をあらかじめ計算しておくことは現実的ではない。したがって、積分値を求める度に Chebyshev 係数を計算しなければならず、多くの処理時間が必要であった。

一般的な 256 階調の画像では、輝度の計算誤差が表示する最大輝度の 1/256 以下であれば画素の輝度値の誤差は 1 階調以下になる。1 階調以下の誤差は量子化誤差より

小さいので、求められた輝度は正しい値といえる。したがって、一般にはそれほど高次の近似式は必要ない。しかし、式5.1の第2項の被積分関数を急峻に変化させるnについては、被積分関数を高次のChebyshev多項式で近似しないと要求される精度が確保できない。Chebyshev係数を求める演算量は次数の2乗に比例して増加するので、積分時間が増大する。

Chebyshev近似は誤差を簡単に評価できるため、必要とされる近似精度を満たす多項式を容易に得ることができる。これはChebyshev多項式を用いる大きな利点である。しかし、輝度積分は画像生成時間のかなりの部分を占めるため、積分の高速化が必要である。

5.2.2 テーブル参照

一般的な画像生成では、輝度計算の誤差が表示する最大輝度の1/256以下であれば十分に正確である。この点に着目して、式5.1をテーブル参照を用いて近似計算することで、積分を高速化する。関数H_tを

$$H_t(n,\alpha,\beta) = \frac{1}{2\pi}\left\{\beta - \int_0^{\beta}\left(\frac{\cos^2\phi}{\tan^2\alpha+\cos^2\phi}\right)^{\frac{n+1}{2}} d\phi\right\} \tag{5.3}$$

と定義すると、式5.1の関数Gは、

$$G(n,\alpha,\beta_A,\beta_B) = H_t(n,\alpha,\beta_B) - H_t(n,\alpha,\beta_A) \tag{5.4}$$

で記述できる。したがって、関数$H_t(n,\alpha,\beta)$の値をn,α,βを引き数とするテーブルに格納すれば、関数Gの値は2回のテーブル参照と1回の減算で求められる。

関数H_tの引き数のうち、鏡面反射指数nは自然数に限られるため、テーブル化してもnに起因する標本化誤差は生じない。また、nは物体の材質ごとに与えられるので、nの連続性を考慮する必要はない。したがって、関数H_tはnの値ごとに独立した二次元テーブルの集合で与えてもさしつかえない。このテーブルを$T_n(\alpha,\beta)$と記述する。

αとβは$[0,\pi/2]$の任意の実数値をとるため、テーブルの引き数を決定する際に標本化誤差が発生する。誤差を縮小するため、関数値$H_t(n,\alpha,\beta)$の値を(α,β)を囲む4つの標本点のテーブル値を内挿して与える。具体的には、パラメータα,βが連続する標本値α_i,α_{i+1}とβ_j,β_{j+1}に対して、

$$\alpha_i \le \alpha \le \alpha_{i+1}, \quad \beta_j \le \beta \le \beta_{j+1} \tag{5.5}$$

の関係にあるとき、関数H_tを

$$H_t(n,\alpha,\beta) = \sum_{k=0}^{1}\sum_{l=0}^{1} \frac{(-1)^{k+l}(\alpha_{i+k}-\alpha)(\beta_{j+l}-\beta)}{(\alpha_{i+1}-\alpha_i)(\beta_{j+1}-\beta_j)} T_n(\alpha_{i+1-k},\beta_{j+1-l}) \tag{5.6}$$

で与える。

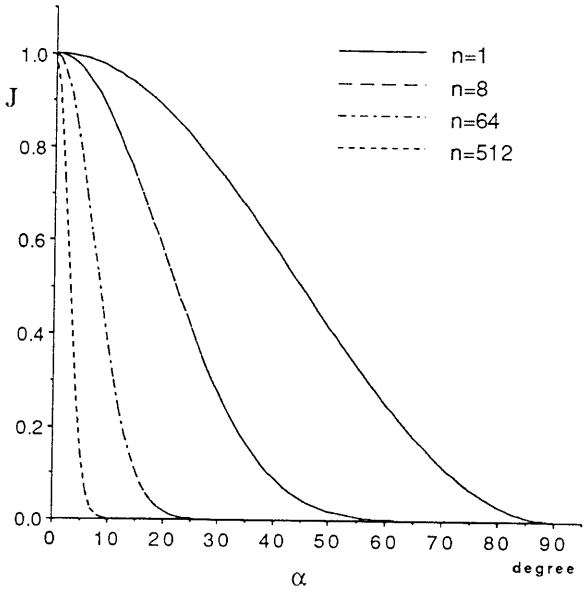


図 5-1: 関数 J

5.3 適応分割テーブルを用いた高速積分

テーブル参照を用いる場合、十分な精度が得られないことと使用するメモリ量が増大することが問題になる。本研究ではこれらの問題を適応分割テーブルとテーブルの選択読み込みで解決する。

5.3.1 均等分割テーブル

テーブルを作るには、n の値ごとにパラメータ α と β の値を決めて関数値を計算する必要がある。このパラメータの組を標本点と呼ぶ。定義域 $[0, \pi/2]$ を等間隔に分割した点を標本点とするテーブルを均等分割テーブルと名付ける。このテーブルは1次関数で参照アドレスを決定できるので参照が簡単である。均等分割テーブルによる近似積分は2次元フィルタリングにおける Feibush の手法 [13](2.8節参照) と等価である。

式 5.3の第2項の被積分関数は $\phi = 0$ で最大値を持つ。そこで、ϕ に 0 を代入して関数 J を得る。関数 J は

$$J(n, \alpha) = \left(\frac{1}{\tan^2 \alpha + 1} \right)^{\frac{n+1}{2}} \tag{5.7}$$

で定義される。

図 5-1に示すように、n が大きいときは、α が 0 から離れると関数 J は急速に 0 に近づく。したがって、関数 H_t の値は $\alpha = 0$ の近傍で大きく変化する。α の定義域を均等に分割すると隣合うテーブル要素の値が大きく異なるため、線型近似の誤差が増

大する。一方、αが相対的に大きいところではテーブルに格納される値がほとんど同じになるので、配列要素が有効に使われない。

5.3.2 適応分割テーブル

被積分関数の変化が大きいところではαを細かく、小さいところではαを荒く分割するように標本点を配置すれば、近似精度は向上する。このテーブルを適応分割テーブルと名付ける。

標本点を非均等に配置する場合、パラメータの値からテーブルの参照アドレスへの写像をどのように与えるかが問題になる。参照アドレスをlogやarccosなどの関数を用いて決定することは可能だが、アドレス計算に時間がかかる。また、標本点を最適に配置できないため関数の近似精度が低下する。

照度積分法と面光源照明法で用いた物体表面の反射特性関数には、回転対称以外の制限は付けられていない。プログラムを変更することなく全ての回転対称関数をテーブル積分できれば、将来の反射モデルの高度化に対しても有利である。

以上を考慮して、インデックステーブルを用いたテーブル参照法を提案する。インデックステーブルは1次元のテーブルで、パラメータ毎に用意する。インデックステーブルには標本点のパラメータ値を記録する。区別のため、積分値を保存するテーブルを関数値テーブルと呼ぶ。

与えられたパラメータの値とインデックステーブルに保存された値とをバイナリーサーチで比較して参照アドレスを決定する。標本点が100点あっても7回の比較で参照アドレスを決定できるので、アドレス計算はそれほど大きな負担にはならない。このアドレス決定機構を用いれば、いっさいの制限を付けずに標本点を配置できる。したがって、テーブルを作りさえすれば、プログラムを全く変更することなく積分値を計算できる。

5.3.3 標本点の配置

以上の考察から、パラメータαの定義域を非均等に分割する。分割点でのパラメータの値はインデックステーブルに保存する。式5.3の計算ではβを均等に分割しても誤差はそれほど大きくならない。そこで、処理時間を短縮するためβの標本点は等間隔に配置する。したがって、本手法ではαだけが非均等に分割される。本節ではαの標本点を決める1手法を述べる。

式5.7を用いて標本点のパラメータ値を決定する。分割数がmならば、関数$J(n,\alpha)$は$\alpha=0,\pi/2$を両端とする$m+1$個の点で折線近似される。近似誤差を最小にする標本点の配置が最適であるが簡単には求められない。そこで、以下のアルゴリズムで標本点を配置する。最適配置ではないが実用上は十分である。

(1) α の定義域 $[0, \pi/2]$ を $r \times m$ 個の区間に分割する。r は 1 より十分大きい整数で、本報告では r を 100 に選んでいる。

(2) 両端点を含む区間の切れ目を分割点と呼ぶ。α が小さい順に分割点に番号をつける。分割点で関数 J を計算する。

(3) 両端を除く偶数番目の分割点で、両側の分割点の関数値を線型補間して求められる値と (2) で計算した関数値との誤差を計算する。

(4) 誤差の平均値を求める。

(5) 誤差が平均値以下の分割点を取り除く。

(6) 残りの分割点が $m+1$ 個よりも多ければ、番号を振り直し (3) にもどる。

(7) 残りの分割点が $m+1$ 個よりも少なければ、直前に取り除いた分割点の中から誤差が大きい順に点を復活し、$m+1$ 個の点列を得る。

(8) 最後に残った分割点のうち、$\alpha = 0$ 除いた m 点を標本点とする。

上記のアルゴリズムでは、(3) の処理で両端点を除くため、両端点は必ず最後まで残る。したがって、パラメータの定義域は m 個の区間でもれなく覆われる。n の値ごとに関数 J の値は異なるので、n の値ごとに標本点の配置も異なる。ただし、インデックステーブルを用いて参照アドレスを決めるので、参照の手法はどの n についても同じで良い。

5.3.4 メモリ使用量の削減

テーブル参照積分ではテーブルを保持するための記憶領域が必要になる。輝度は画素の並びの順に計算されるので、参照するテーブルも頻繁に替わる。このため、テーブルが替わる度にアクセス速度が遅いディスク等の補助記憶装置から必要なテーブルを読み込むと、テーブル参照による速度改善の効果が失われてしまう。仮想記憶オペレーションシステムでもテーブルが使用する記憶領域が大きすぎるとページフォルトが多発して処理速度が低下する。したがって、テーブルは主記憶上に置ける大きさである必要がある。

鏡面反射の鋭さを制御する係数 n は自然数に限られる。また、n には物体の材質ごとに固有の値が与えられる。このため、n については画像中に現れる材質を表現するために必要なテーブルさえ用意すればよい。実際には、1 枚の画像中にそれほど多種類の材質が現れるわけではない。また、n が大きい場合には、2 の整数乗といった代表的な値だけを用いても物体の鏡面反射特性を十分に表現できる。

n を連続に変化させる必要がないので、テーブル数十枚を記憶する領域を用意し、必要なテーブルだけを読み込む。この結果、使用するメモリー量を飛躍的に削減でき

る。例えば、α と β をそれぞれ 100 分割して同時に 100 種類の n を使用するには、$100 \times 100 \times 100 \times 4$bytes= 4Mbytes の記憶領域を用意すれば良い。これは、画像を保持するための領域 ($640 \times 480 \times 3$bytes$\approx$ 922kbytes) の約 4 倍で、現在の計算機のメモリー実装量から考えると問題になる大きさではない。

5.4 積分手法の比較実験

本節では均等分割テーブル積分, 適応分割テーブル積分, Chebyshev 近似積分の計算精度と処理速度を比較する。テーブル積分では式 5.3の値をテーブル参照で求め式 5.4に代入して関数 G を計算する。Chebyshev 近似では式 5.1を区間 $[\beta_A, \beta_B]$ で直接近似積分して G を計算する。こちらのほうが近似が 1 回で済むため、計算精度が向上し処理時間も削減できる。実験には浮動小数点演算器付きの VAX8840 を用い、単精度で計算する。

5.4.1 テーブル参照積分の誤差特性

3 つの角度 $(\alpha, \beta_A, \beta_B)$ で定義される積分領域を乱数を用いて 100,000 個発生した。それぞれの角度は $[0, \pi/2]$ で定義され、$\beta_A < \beta_B$ の関係がある。10 種類の鏡面反射指数 (n = 1, 2, 4, 8, 16, 32, 64, 128, 256, 512) について、6 とおりの大きさのテーブル ($\alpha \times \beta = 30 \times 30$, 60×60, 90×90, 120×120, 150×150, 180×180) で $G(n, \alpha, \beta_A, \beta_B)$ を計算した。鏡面反射指数とテーブルサイズのすべての組み合わせについて、それぞれ 100,000 サンプルの近似誤差を計算し、その最大値を求めた。誤差評価に用いた真の値は台形近似積分で求めたが、関数の定義域を十分に細かく分割したうえ倍精度で演算したので、十分に高い近似精度を持つ。このため、本実験で用いる限り真の値とみなして良い。

n = 1, 64, 512 の場合の、テーブルサイズと最大誤差との関係を図 5-2に示す。均等分割でも適応分割でもテーブルを大きくするほど誤差が減少する。どの n についても、適応分割テーブルの誤差は均等分割テーブルの誤差より小さい。

テーブルサイズが 30×30, 90×90, 180×180 の場合の鏡面反射指数 n と最大誤差との関係を図 5-3 に示す。実験では均等分割テーブルの誤差は n に比例して増加した。これは n が大きくなるにつれて反射特性が急峻になるためと考えられる。適応分割テーブルでは誤差はほとんど変化しなかった。このため、n が大きくなるほど適応分割による精度改善の効果が顕著になる。以上の結果から、適応分割により大幅に近似精度が向上する。

輝度値は関数 G の値を数個足したり引いたりして計算する。このため、関数 G の計算では 4×10^{-4}(最終的に必要な精度の 10 倍) 程度の精度が必要になる。本実験の

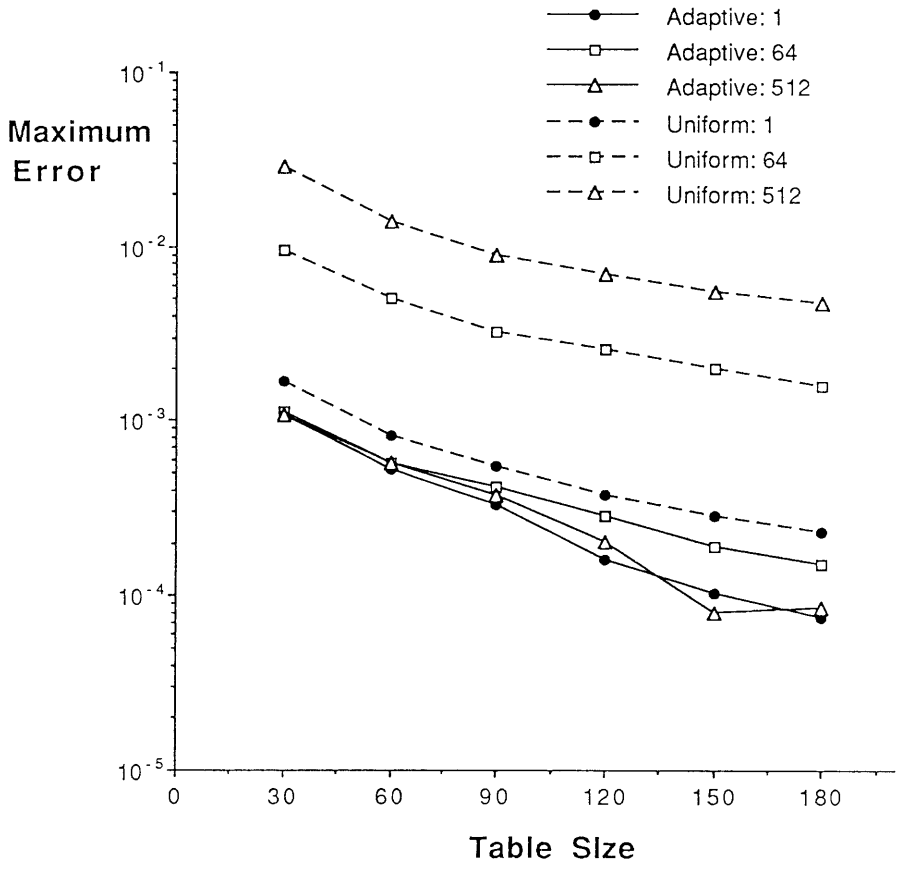


図 5-2: テーブルサイズと最大誤差

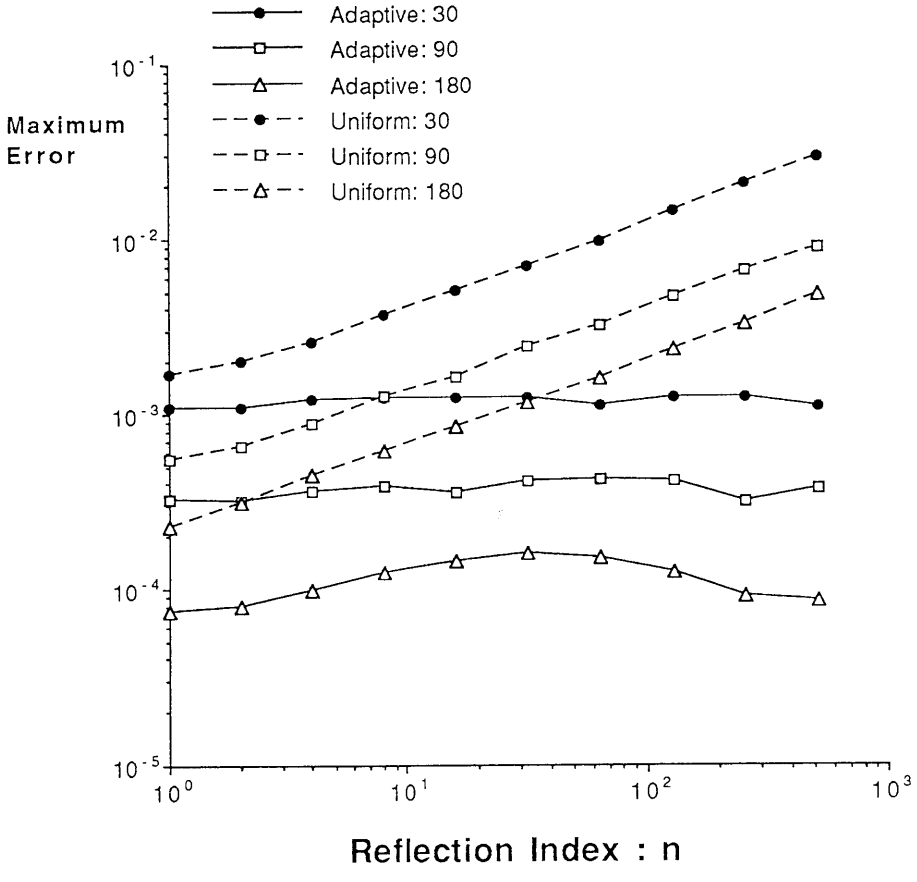


図 5-3: 鏡面反射指数 n とテーブル積分の最大誤差

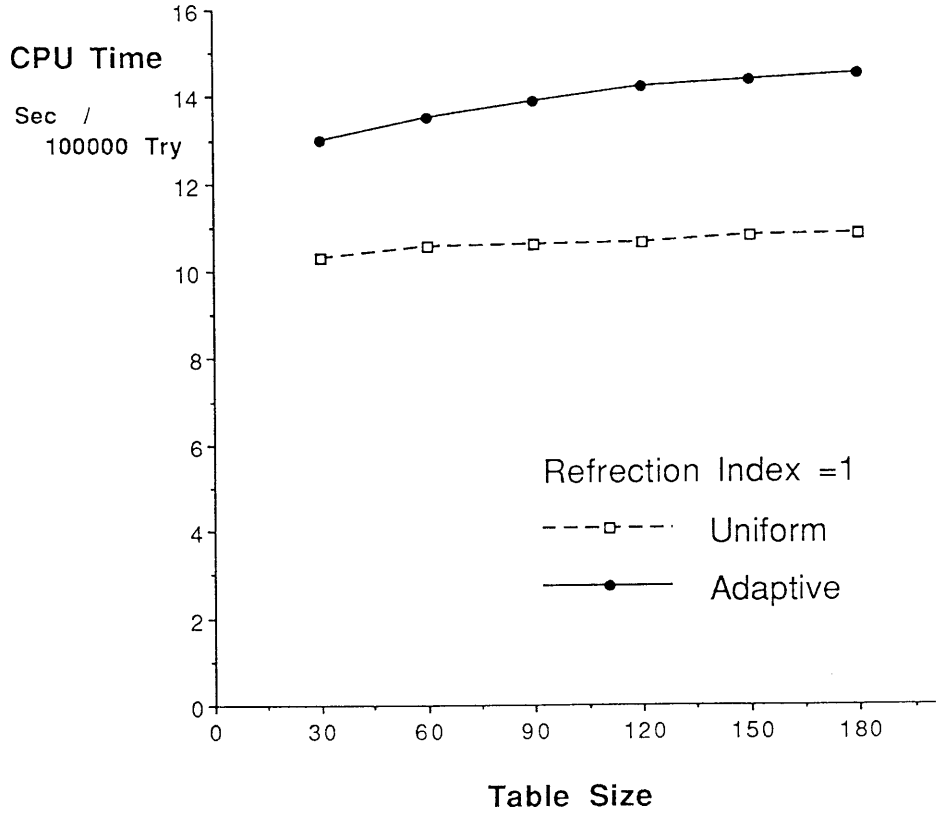


図 5-4: テーブルサイズと積分時間

結果は、90 × 90 の適応分割テーブルで要求精度を満たせることを示している。積分誤差が画像に与える影響は 5.5.1節で議論される。

5.4.2 テーブル参照積分の計算時間

5.4.1節の実験の処理時間を計測した。テーブルサイズと処理時間との関係を図 5-4に示す。テーブル参照に要する時間は鏡面反射指数 n とは無関係なので、$n = 1$ の場合の結果のみを示す。

理論どおり均等分割テーブルの積分時間はテーブルサイズとは無関係であった。適応分割テーブルの積分時間はテーブルサイズに比例して若干増大した。これは α のインデックステーブルを検索する回数がテーブルサイズに比例して増加することによる。均等分割テーブルと適応分割テーブルとの時間差がインデックスの検索時間である。実験では 30% ほどの処理時間が検索に使われた。

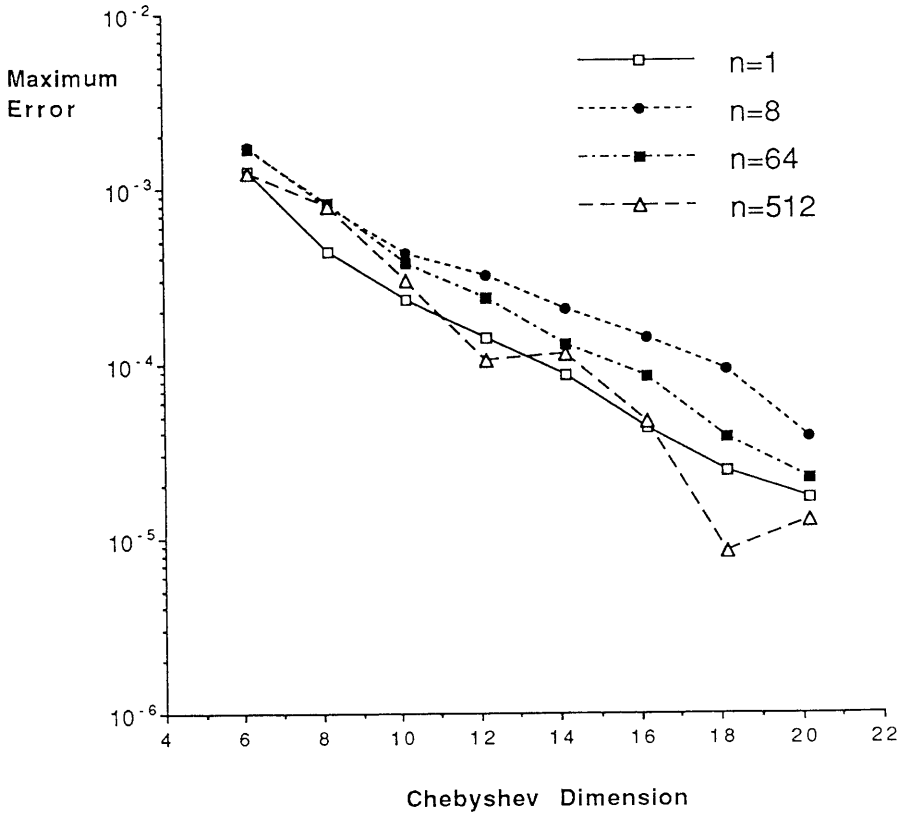


図 5-5: Chebyshev 近似次数と最大誤差

5.4.3 Chebyshev 近似積分の誤差特性

5.4.1節と同じ 100,000 組のサンプルを用いて実験を行った。Chebyshev 近似の次数を 6 次から 20 次まで 2 次おきに変化させ、5.4.1節の 10 種類の n について最大誤差を求めた。

近似次数と最大誤差との関係を図 5-5に示す。$n = 512$ では被積分関数が急峻すぎるため数値演算の誤差により最大誤差がばらつくが、一般的な傾向として、次数を増すにつれて誤差が減少する。

n と最大誤差との関係を図 5-6に示す。Chebyshev 近似の次数が高くなるほど演算誤差の影響で最大誤差がばらつくが、鏡面反射指数 n と最大誤差の間に顕著な関連は認められなかった。

この実験結果から、4×10^{-4} の計算精度を得るには 10 次の Chebyshev 多項式で近似する必要があることが明かとなった。実際に、反射強度積分法 (3章) と面光源照明法 (4章) では 10 次 Chebyshev 多項式を用いた。

5.4.4 Chebyshev 近似積分の計算時間

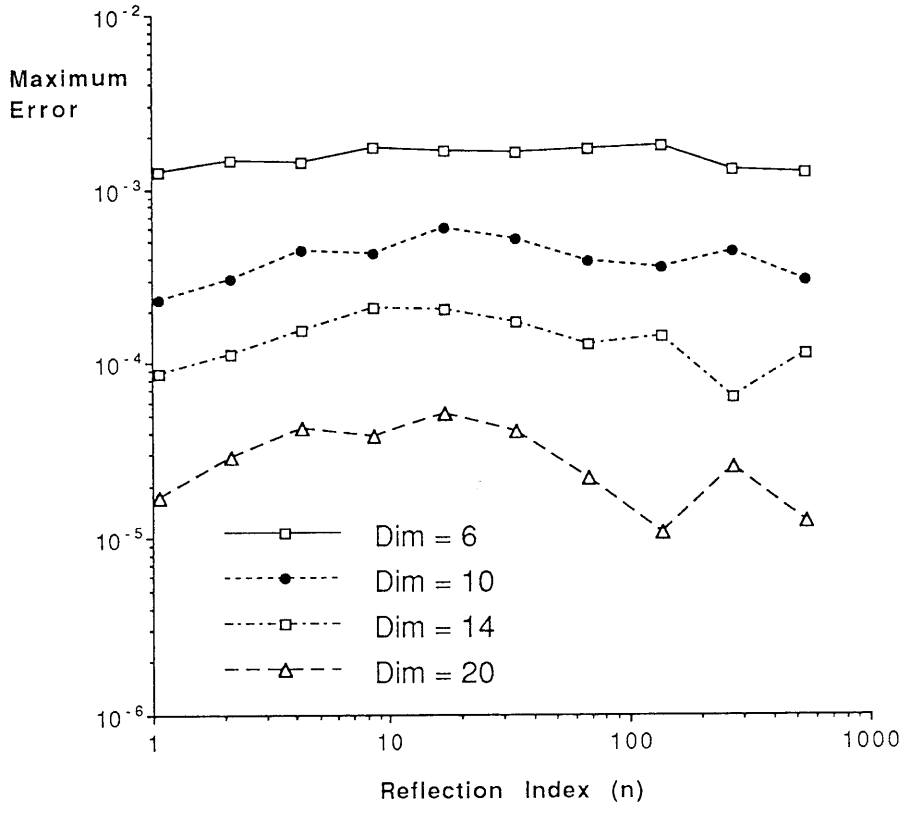


図 5-6: 鏡面反射指数 n と Chebyshev 近似積分の最大誤差

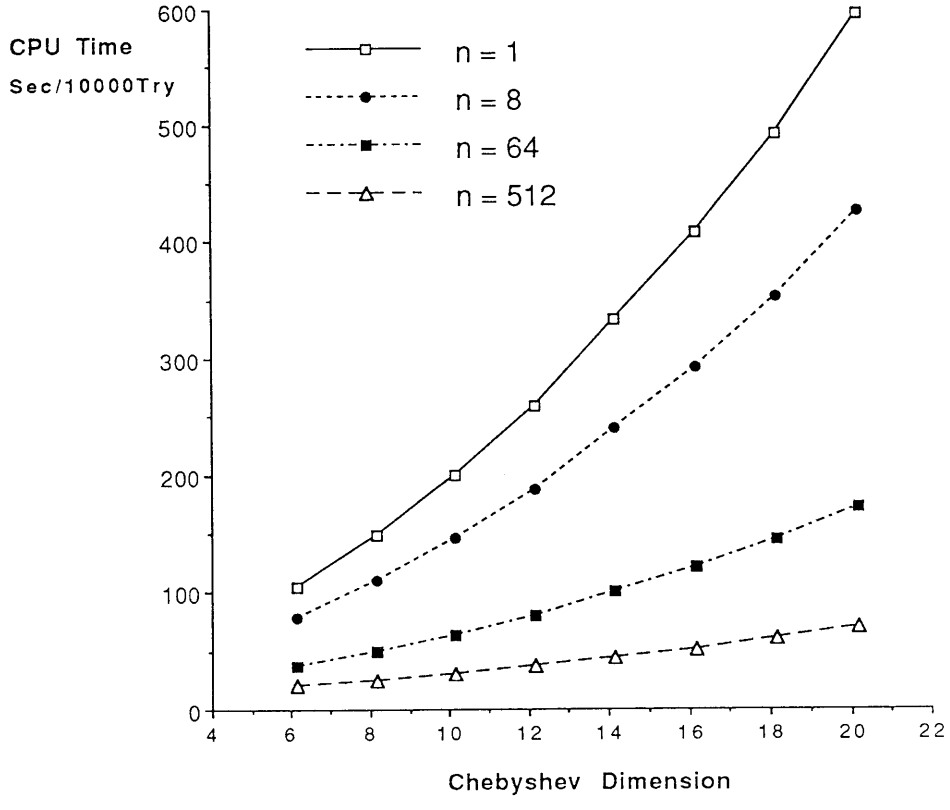


図 5-7: Chebyshev 近似次数と積分時間

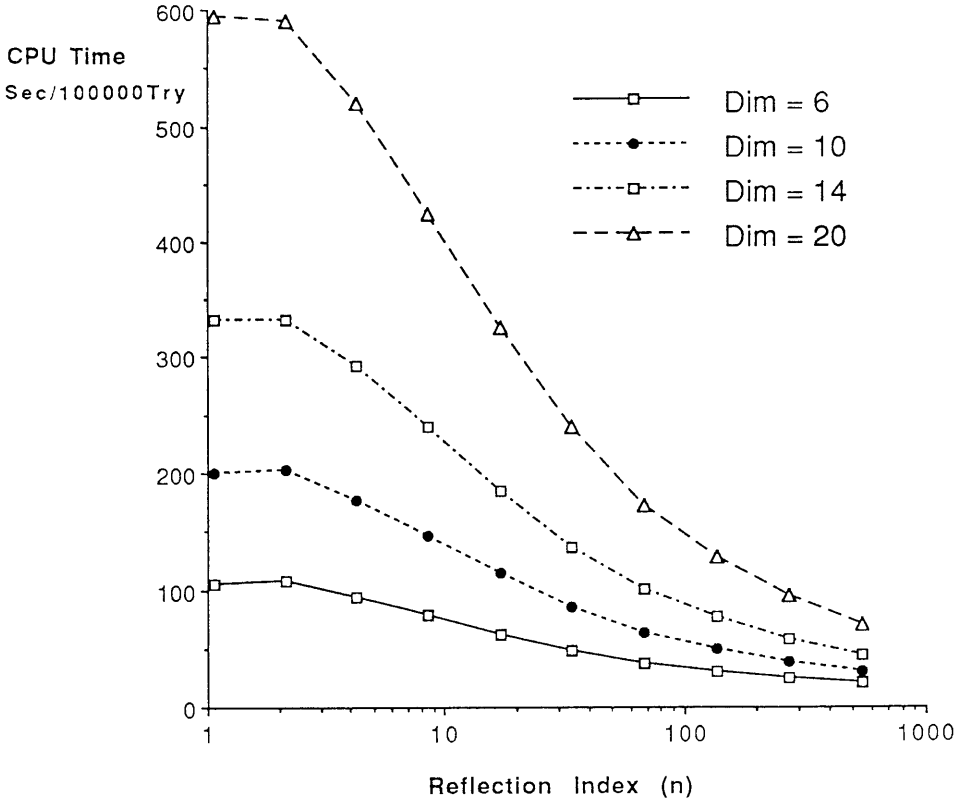


図 5-8: 鏡面反射指数 n と Chebyshev 近似の積分時間

5.4.3節の処理時間を計測した。Chebyshev 近似次数と計算時間との関係を図 5-7に、n と計算時間との関係を図 5-8に示す。実験結果は、近似次数が大きくなるにつれ計算時間が増加すること、n が小さい場合ほど処理時間がかかることを示している。

理論的には、Chebyshev 近似積分の計算時間は近似次数の 2 乗に比例して増加し、n の値には無関係になる。しかし、本研究では、計算時間を短縮するため、多項式近似する前にその必要性を判定している。具体的には、式 5.7を量子化誤差より十分に小さくする α については、式 5.1の第 2 項の積分値を 0 とみなせるので近似を省略する。n が大きくなると近似を省略できる α の範囲が広くなるので、近似積分の回数が減る。このため、合計の処理時間は減少する。図 5-8に示される実験結果は前処理による計算時間削減の効果を示している。

5.4.5 テーブル参照と Chebyshev 近似の比較

適応分割テーブルを用いた積分と Chebyshev 近似積分との精度を比較する。図 5-9の実線は Chebyshev 次数と 5.4.3節で測定した最大誤差の平均値との関係を示している。図 5-9の点線は 5.4.1節で測定した適応分割テーブルの最大誤差の平均を示してい

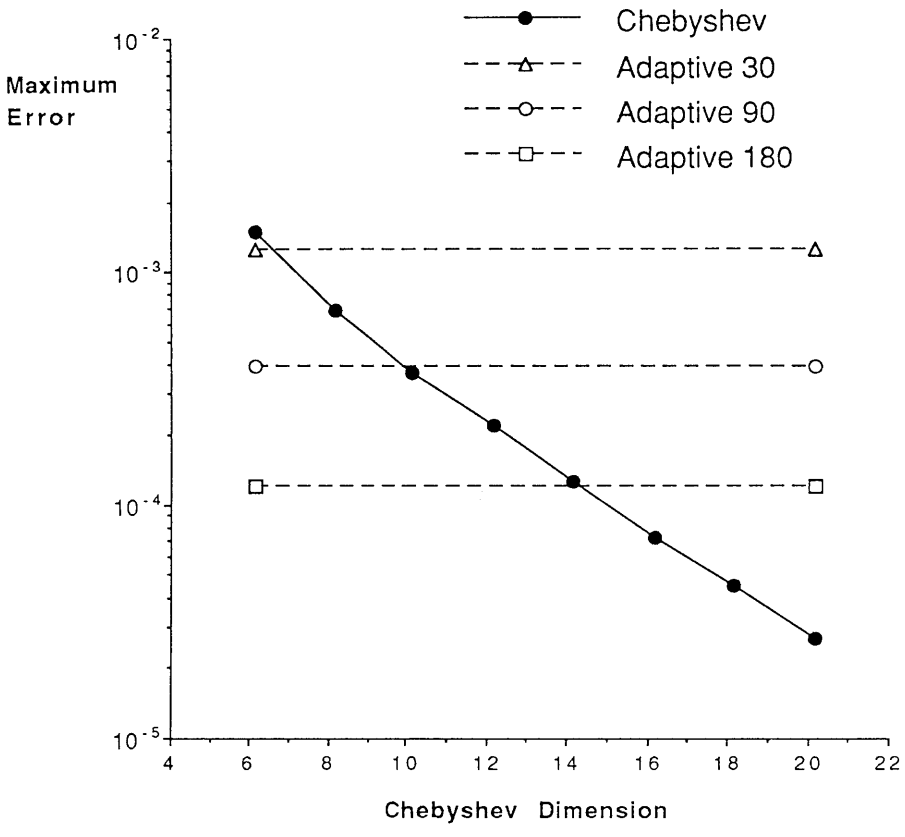

Chebyshev
Adaptive 30
Adaptive 90
Adaptive 180
Maximum
Error
10-2
10-3
10-4
10-5
4
6
8
10
12
14
16
18
20
22
Chebyshev Dimension

図 5-9: 近似精度の比較

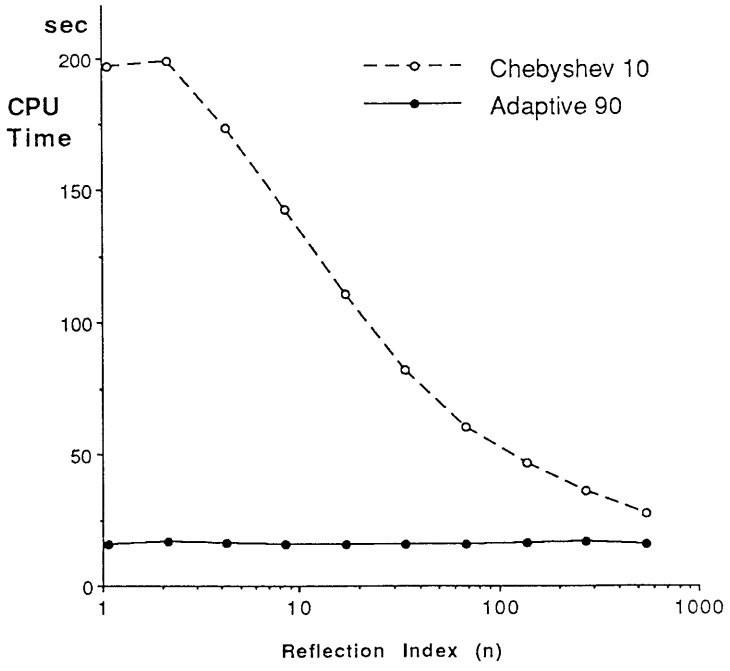


図 5-10: 処理時間の比較

る。点線と実線の交点から、 30×30, 90×90, 180×180 適応分割テーブル積分はそれぞれ 6,10,14 次 Chebyshev 近似積分と同等の精度を持つといえる。

積分計算で誤差が 4×10^{-4} 以下であれば、輝度値の誤差を 1 階調以下に抑えることができる。この場合、計算誤差は量子化誤差より小さいので、正確な輝度値といえる。実験結果は、90×90 適応分割テーブルと 10 次 Chebyshev 近似は精密な画像を生成するために必要な近似精度を満たすことを示している。

鏡面反射指数 n を横軸にとって、90×90 適応分割テーブル積分と 10 次 Chebyshev 近似積分との計算時間を比較する。結果を図 5-10に示す。どの場合でもテーブル積分が Chebyshev 近似積分より高速であった。$n = 1$ では 10 倍以上の高速化が実現できた。

5.5 画像生成実験

5.5.1 誤差が画像に与える影響

均等分割テーブルと適応分割テーブルとで、近似誤差が生成画像に与える影響を比較した。平面上に置かれた立方体が星型光源により照明されるシーンを実験画像に選んだ。立方体の n(鏡面反射指数) を 512、背景の平面の n を 64 に設定している。

図 5-11に 90×90 適応分割テーブルで生成した画像を示す。5.4.1節と 5.4.5節の実験結果が示すように、90×90 適応分割テーブル積分の誤差は 4×10^{-4} 以下のため、

輝度値の誤差を1階調以下に抑えることができる。したがって、画像には近似誤差の影響は現れない。

図5-12に30 × 30均等分割テーブル積分による画像を示す。5.4.1節の実験では、30 × 30均等分割テーブルは$n = 512$のとき3×10^{-2}の誤差を含む。計算誤差が大きいので、星型光源のエッジの延長線上に輝線が発生した。また、ハイライトの中に本来は見えない光源のシルエットが現れた。比較のため、図5-13に30 × 30適応分割テーブル積分の画像を示す。この場合にも画像上にわずかに誤差の影響が現れる。しかし、30×30適応分割テーブルの誤差は1×10^{-3}なので、誤差の影響は均等分割テーブルに比べて極めて小さく、注意しないとわからない程度である。

図5-14に180×180の均等分割テーブルによる画像を示す。5.4.1節の実験結果が示すように、均等分割では180 × 180のテーブルを用いても$n = 512$で5×10^{-3}の誤差を含む。このため、近似誤差による輝線を完全に除去することはできなかった。適応分割では1/4のメモリ量で全く問題の無い画像を生成できた。この結果は、適応分割による近似精度の向上を端的に示している。

さらに悪い条件で画像生成実験を繰り返したが、90 × 90の適応分割テーブルを用いれば計算誤差の影響は表れなかった。90×90テーブル1枚は90×90×4 = 32.4kbytesのメモリを必要とするにすぎない。インデックステーブルは$90 \times 1 \times 4 = 360$bytesである。従って100枚のテーブル領域を用意しても約3.3Mbytesで現在のコンピュータ環境では全く問題にならない大きさである。一般的な画像の生成には100枚のテーブル領域で十分である。

5.5.2 画像生成速度の比較

同じ近似精度を持つ10次のChebyshev近似と90 × 90適応分割テーブルとで画像生成時間を比較した。実験には、本章の図5-11 (5.5.1節で用いた星型光源)と図5-15 (チェスの駒), 4章の図4-7(a) (面光源で照明された室内シーン)と図4-11 (dull reflectionを持つティーポット)を用い、約6MIPSの処理能力を持つ浮動小数点演算器付きのVAX8840で画像生成した。結果を表5-1に示す。この実験では、計算時間が60 ～ 40%に減少した。目視による評価では画像品質に差は認められなかった。

5.6 結言

反射強度積分法と面光源照明法に共通する基本演算である球面積分を適応分割テーブルを用いて高速化した。

鏡面反射指数が大きい場合、反射特性関数は正反射方向から離れるにつれて急速に0に近づく。このため、標本点を等間隔に配置する均等分割テーブルでは、配列が有

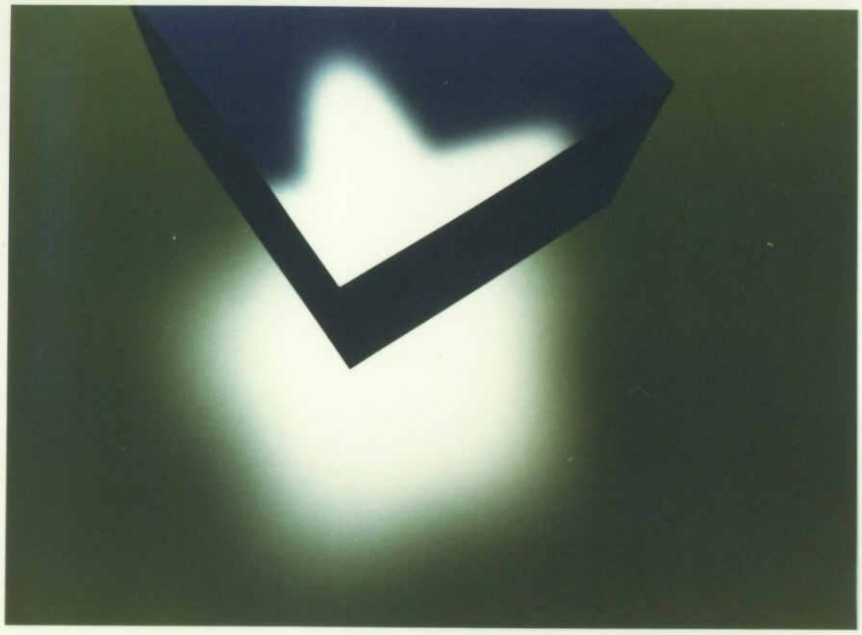


図 5-11: 90 × 90 適応分割テーブル

図 5-12: 30 × 30 均等分割テーブル

図 5-13: 30 × 30 適応分割テーブル

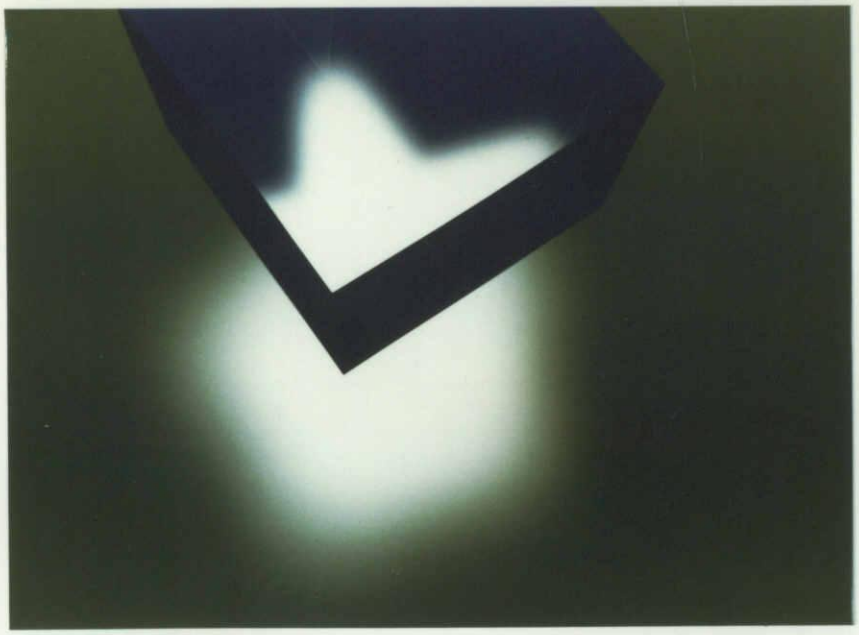


図 5-14: 180 × 180 均等分割テーブル

図 5-15: 実験画像;*Chess*

表 5-1: 画像生成時間

実験画像	生成時間		短縮率
	Chebyshev 近似	適応分割テーブル	
図 4-7(a) P.99	38'37"	13'20"	0.35
図 4-11 P.103	1:35'10"	36'45"	0.39
図 5-11 P.126	1:04'21"	31'50"	0.49
図 5-15 P.128	1:57'59"	1:06'28"	0.56

効に使用されないので、近似精度が低下する。そこで、反射特性関数が急峻に変化する部分では標本点を細かく、緩やかに変化する部分では標本点を粗く配置する適応分割テーブルで近似精度の向上をはかった。本論文では、適応分割テーブルを作成する1手法を述べた。均等分割テーブル積分, 適応分割テーブル積分, Chebyshev 近似積分の計算精度と処理速度を比較し、適応分割テーブルの優位性を示した。

標本点を非均等に配置するときに問題となる参照アドレスの決定は、インデックステーブルを用いて解決した。標本点のパラメータの値をインデックステーブルに保存し、インデックステーブルの値と入力されたパラメータの値をバイナリサーチで比較してテーブルの参照アドレスを決定する。この手法は参照アドレスを高速決定でき、アルゴリズムを変えることなくどのような標本点の配置にも対応できる。

鏡面反射指数は物体の材質ごとに与えられる。多くの場合、1枚の画像にはそれほど多くの異なる材質は含まれない。そこで、画像生成に必要なテーブルのみをシステムに読み込むことで、テーブルが占有するメモリを大幅に削減した。実験により 90 × 90 適応分割テーブルで十分な精度が得られることが示された。この場合、100 枚のテーブル領域を用意しても 3.3Mbytes のメモリしか必要としない。

適応分割テーブルと均等分割テーブルとの比較実験では、適応分割により処理時間は3割程度増加するものの近似精度は大幅に向上した。誤差評価実験により 90×90 適応分割テーブルの近似誤差は 4×10^{-4} 以下であることを確認した。これは 10 次 Chebyshev 近似に匹敵する精度で、256 階調の画像を生成するには十分である。適応分割テーブルを用いることで積分計算に要する時間は 1/10 に短縮された。

画像生成実験でも 90 × 90 適応分割テーブルを用いれば十分な画像品質が得られることが確かめられた。この場合の画像生成時間は同精度の Chebyshev 近似積分の 60 ～ 40% に削減された。適応分割テーブルは面光源照明法を高速化する上できわめて有効な手法である。

第 6 章

結論

本研究では解析的な画像生成手法を確立した。これにより、精密で確実なアンチ・エリアシングが実現された。

本論文で報告した直交スキャンライン法は物体の外形線の方向によらず精密にアンチ・エリアシングする。また、どのように細かい物体でも適正に表示する。精密レンダリング法は物体表面の曲率が大きくとも画素の輝度値を正確に計算できるので、丸められたエッジと頂点に現れるハイライトや陰を正しく表示する。面光源照明法は鏡面反射成分と拡散反射成分の両方を解析的に計算することで、写実的な画像を正確に生成する。これらの解析的手法はエリアシングを確実に除去できる。スーパーサンプリング手法のように試行錯誤を心配しなくてよいので、ユーザーの負担は大幅に軽減される。

かつてはコンピュータの処理能力が低かったため、多少のエリアシングには目をつむっても画像生成速度を改善する努力が行われた。しかし、たとえエリアシングが画像のごくわずかの部分だけに表れるとしても、画像全体の品質は著しく劣化する。このため、計算機の処理速度が向上した今日では、エリアシングを厳密に除去して高品質画像を生成することが求められている。

実際には画素は画像平面上の正方形領域を代表している。このため、物体の輪郭線が横切る画素には背景を含め複数の物体が投影される。画素の中心点で隠れ面消去すると、これらの物体の１つだけから画素の輝度値が決まってしまうので、正しい値にはならない。また、物体表面の曲率が大きいとわずかの距離で面法線の方向が大きく変化するので、１画素の中でも場所により物体の輝度が異なる。したがって、画素内の１点だけで輝度を計算したのでは、真の値と大きく異なることがある。このように、エリアシングは画像生成の過程で不適切にサンプリングすることで発生する。

従来のアンチ・エリアシング手法の多くは、1章で報告したように、スーパーサンプリング手法かサンプリング結果を後処理する手法である。サンプリングを用いた手法では、１画素を複数の点で代表させることで画素内の微細構造を近似的に画素値に反映することができる。しかし、精密にエリアシングを除去するには極めて多くのサ

ンプル点が必要になる。

隠れ面消去だけを考えても、256 階調の画像で輝度値の誤差を確実に 1 階調以下に抑えるには、面積の近似誤差を画素の大きさの 1/256 以下にする必要がある。物体の輪郭線がどちらを向いても常にこの精度を確保するには、画素を縦横それぞれ 256 等分しなければならない。均等分割で実現すると 1 画素あたり $256 \times 256 = 65,536$ 個のサンプル点が必要となるので、現在のコンピュータの処理速度をもってしても実現は難しい。

実際の画像生成では生成時間も考慮しなければならない。このため、エリアシングを実用上問題の無い程度まで弱められるサンプル点数で妥協している。ただし、エリアシングの程度は画像生成条件によりさまざまに異なるので、適正なサンプル数を論理的に決めることは極めて困難である。事実、サンプル数は経験的に決められている。適応的にサンプリングする場合にも、分割の間隔や打ち切りの基準を経験的に決める必要がある。

サンプル点が少なすぎるとエリアシングが目立つので、サンプル点を追加して画像生成をやり直さなければならない。スーパーサンプリング手法では確実にアンチ・エリアシングできる保証がないので、このような試行錯誤は避けられない。画像生成条件がダイナミックに変わるアニメーションでは特にサンプル数の決定が困難で、画像品質の確認と画像生成のやり直しのため作業が煩雑になる。このようなわずらわしさを解消するためにも、確実にアンチ・エリアシングできる画像生成手法が必要である。

本研究では、解析的に画像生成することでサンプリングの悪影響を根本的に排除する。本論文では画像生成の主要技術である隠れ面消去と輝度計算の解析的アルゴリズムを報告した。本研究の成果を以下にまとめる。

1. 直交スキャンライン法 (2 章)

従来の隠れ面消去手法の中でも、クリッピングを用いた手法は 1 画素内に投影されるポリゴンの領域を正確に決定できる。このような解析的手法では精密なアンチ・エリアシングが可能である。ただし、クリッピング手法は処理コストが高いので、一般的なシーンに適用すると画像生成時間がきわめて長くなる。

2章で報告した直交スキャンライン法は、水平走査と垂直走査を組み合わせることで、1 画素内のポリゴン領域を効率よく決定する。水平走査線は画素の水平境界に配置され、通常のスキャンラインと同様に画像の左端から右端まで走査される。垂直走査線は、ポリゴンの頂点, ポリゴンエッジと水平走査線との交点, ポリゴンエッジ同士の交点, 画面の左右端を通るように配置され、隣合う水平走査線の間を 1 画素の高さだけ走査される。走査線上では通常のスキャンライン法と同様に隠れ面消去される。2 方向の走査線によりポリゴンは重なりの無い台形領域に分割されるので、1 画素内

のポリゴンの領域を演算精度の正確さで決定できる。

2.6節の実験では、直交スキャンライン法はどの方向のポリゴンエッジのエリアシングも精密に除去できることが示された。また、極めて細いポリゴンでも適正に表示できることが示された。2.6.4節に示した画像生成速度の比較では、直交スキャンライン法の生成速度は1画素あたり4 ～ 7本のサブ・スキャンラインを走査するマルチスキャニング法の生成速度と同じであった。

2.7節では画質評価手法を報告した。実験では直交スキャンライン画像は同じ時間で生成できるマルチスキャニング画像に比べはるかに高品質であることが確認された。実験結果が示すように、直交スキャンライン法はスーパーサンプリング手法に比べずっと効率良くアンチ・エリアシングできる。

2.8節では直交スキャンライン法に適したフィルタリング手法を報告した。フィルタ処理を行うことで、よりいっそう画像品質を改善できる。直交スキャンライン法は1画素内のポリゴンの領域を精密に決定できるので、小さいサイズのフィルタでもエリアシングを効果的に除去できる。

2. 反射強度積分法 / 精密レンダリング法 (3 章)

3章では物体表面の曲率が大きい場合でも輝度を正確に計算できる反射強度積分法を考案した。曲率が大きいと1画素の中でも場所により面法線の方向が変化するため、画素の輝度値を正確に計算するには画素内で反射光の強度を積分しなければならない。この積分を直接計算することは困難なため、反射強度積分法では極座標系に変換して計算する。画素内に存在する面法線が張る方向領域と単位球面とが交差する部分が積分範囲になる。積分を簡単にするため、積分範囲を z 軸と単位球面との交点を頂点とする単位球面上の三角形に分割する。この分割で2変数の一方を代数的に積分できる。他方は被積分関数を Chebyshev 多項式で近似したのち代数積分する。画像生成に十分な精度の多項式で近似することで精密な輝度値が計算できる。

丸められた稜線や頂点では短い距離で面法線の方向が大きく変化するので、広い範囲の光源が映り込む。これらの稜線や頂点に現れるハイライトや陰は物体形状を認識する有力な手がかりとなる。このため、正確なハイライトや陰の生成は画像の現実感を高める効果的な手段となる。

丸められた稜線や頂点のような曲率の大きい曲面をポリゴンで近似すると、ポリゴンは画素に比べ小さく、面法線の方向は画素内で大きく変化する。このようなポリゴンの輝度を正確に計算するため、直交スキャンライン法で1画素内に投影されるポリゴンの領域を正確に求め、得られた領域の輝度を反射強度積分法で厳密に計算する手法を開発した。この精密レンダリング法と名付けられた手法はスーパーサンプリング手法に比べはるかに正確に画像を生成できる。

3.5節の計算機実験では、精密レンダリング法は稜線の方向や位置によらず正しいハイライトを生成できることが示された。また、丸められたエッジに現れる陰も表示できることが示された。画像を比較した結果、稜線のハイライトや陰が物体の実在感を高める上で重要であることが確認できた。高度のアンチ・エリアシングと正確な輝度計算を提供する精密レンダリング法は写実的な画像を生成する上で極めて有効な手法である。

3. 面光源照明法 (4 章)

世の中一般の光源はそれぞれ固有の大きさを持つ。これらの光源は面光源で近似できる。したがって、現実的な画像を生成するには面光源による照明を取り扱うことが不可欠である。面光源は点光源に分割できるが、サンプリング間隔が粗いと明るさが不連続になるため疑似的な照度パターンが発生する。4章で報告した面光源照明法は、物体表面での鏡面反射と拡散反射の両方を解析的に計算することで、照明によるエリアシングを完全に除去した。

面光源照明法は多角形の完全拡散光源で照明し、入反射のエネルギーを保存するように改良した Phong の反射モデルを用いて輝度計算する。はじめに、視線方向の正反射方向が z 軸となるように極座標変換を行い、光源内での平面積分を球面積分に変換する。輝度計算する点を原点とする単位球面に光源を投影した領域が積分範囲になる。次に、積分範囲を z 軸と単位球面との交点を頂点に含む単位球面上の三角形に分割する。分割により一方の変数を代数積分できる。最後に、被積分関数を Chebyshev 多項式で近似してから残りの変数を代数積分し、解析的な近似解を得る。近似誤差が画素の輝度値の量子化誤差より十分に小さくなるように多項式の次数を選ぶことで、正確な輝度値が計算できる。

4.4節の計算機実験では、面光源照明法は鏡面反射により生ずるハイライトを正確に表示できた。また、鏡面反射係数を変えることで反射の鋭さを自由にコントロールできた。dull reflection の表示にも有効であった。これらの効果は写実的な画像を生成するために重要であるが、完全拡散反射物体しか扱えない従来手法では表現できなかった。画像生成速度で比較すると、面光源照明法の画像は同じ品質の点光源近似法の画像の 1/3 の時間で生成できた。これらの実験結果から、面光源照明法の優位性が確認された。

4. 高精度テーブル積分法 (5 章)

精密レンダリング法 (3章) と面光源照明法 (4章) の実験では、画像生成時間の多くを座標変換と球面積分が占めている。画像生成時間を短縮するため、5章では適応分割テーブルを用いて球面積分を高速かつ高精度で計算する手法を報告した。

反射強度の積分式は3つのパラメータを含む。このうち鏡面反射指数 n は物体の材質ごとに固有の値をもつので、連続性を考える必要がない。このため、積分値は n の値ごとに独立した2次元テーブルで与えられる。

一般的なCG画像を正しく表示できる近似精度を確保するため、反射特性関数が急峻に変化する部分では標本点を細かく、緩やかに変化する部分では標本点を粗く配置する適応分割テーブルを用いた。テーブルの参照アドレスはインデックステーブルをバイナリサーチして決定する。インデックステーブルを用いることで標本点を自由に配置できる。

誤差評価実験により 90×90 の適応分割テーブルの近似誤差は 4×10^{-4} 以下であることを確認した。この精度は10次Chebyshev近似に相当し、256階調の画像を生成するには十分に高い。画像生成実験でも 90×90 適応分割テーブルを用いれば十分高い品質の画像が得られることを確かめた。

テーブルが占有する記憶領域を縮小するため、画像生成に必要な n の値を持つテーブルのみをシステムに読み込む。90×90 テーブルでは100枚分の領域を用意しても3.3Mbytesの記憶領域しか必要としないので、テーブルが占有する記憶領域はさほど問題にならない。

適応分割テーブルは積分計算に要する時間を1/10に短縮した。画像生成全体で比較しても、同等の計算精度を持つ10次Chebyshev近似積分を用いた場合の60～40%の計算時間で画像生成できた。適応分割テーブルは面光源照明法を高速化する上できわめて有効な手法である。

以上に示した手法を用いて画像を解析的に計算することで、エリアシングの心配がない画像生成が可能となった。スーパーサンプリング手法に必須のサンプル数の決定や画像生成のやり直しといった手間を一切省くことができるため、ユーザーの負担はずっと小さくなる。精密に画像生成するため処理時間は増加するが、通常手法の数倍程度に収まる。したがって、スーパーサンプリング手法よりはずっと高速である。

エリアシングは不適切なサンプリングにより発生する。厳密にエリアシングを除去するには画像生成の全ての過程を解析的に計算する必要がある。例えばマッピングでは、1点で計測したマップの値を用いて画素の輝度値を計算するとエリアシングが発生する。スーパーサンプリングでエリアシングを削減できるが、根本的に解消するには画素内に射影されるマップ画像の領域を精密に求める必要がある。

アニメーションで物体が高速移動する場合には動きがぎくしゃくする。これは1フレーム間の物体の移動距離が大きすぎるために生じる時間軸上のエリアシングである。フレーム間に疑似的なフレームを挿入することで改善できるが、厳密にエリアシング

を除去するにはフレームの間で物体の軌跡を積分する必要がある。

このように解決されなければならない問題は多数残っている。さらに画像品質を高めるため、画像生成の過程で使われるサンプリング手法を解析手法で置き換える研究を今後も進めていく必要がある。

謝辞

本研究を遂行するにあたっては多くの方々の御助力を戴いた。深く感謝します。

名古屋大学工学部電気工学科 杉江昇教授には著者が名古屋大学に在籍していた当時より継続して御指導と御助力を戴いている。本研究をまとめるにあたっても、論文の内容を高めるための数多くの御助言を戴くとともに終始御激励を賜った。名古屋大学工学部情報工学科 鳥脇純一郎教授ならびに横井茂樹助教授には学会活動を通じて日頃から御指導御鞭撻戴いている。このたびは貴重な時間を割いて本論文の査読をして戴き、数多くの御助言を賜った。著者在学中の御教授も合わせて深く感謝します。

東京大学工学部計数工学科 杉原厚吉教授には名古屋大学在任中に3年にわたり直接ご指導戴くとともに、以後も御指導と御激励を賜っている。本論文についても論理的な問題点から文章の書き方に至るまでの幅広い範囲で数多くの御助言を戴いた。

本論文はNTTでのコンピュータグラフィックス研究の成果をまとめたものである。研究の機会を与えてくださったNTTヒューマンインタフェース(HI)研究所長 釜江尚彦博士、HI研究所知能ロボット研究部長 高野陸男博士に感謝します。また、論文をまとめる機会をくださった知能ロボット部主席研究員 酒井高志氏，同部主幹研究員 奥平雅士博士に感謝します。

HI研究所知能ロボット部主幹研究員 高橋時市郎氏にはNTT入社以来継続して御指導戴くとともに、共同研究者として研究全般にわたり御助言と御激励を賜った。同部主任研究員 新谷幹夫博士にも研究全般で多数の御助言を戴いた。なかでも4章の面光源照明法の原理については詳細に御議論戴いた。また、同部主任研究員 斎藤隆文博士にはエッジハイライトについて御議論戴いた。同部主幹研究員 成瀬正博士，同部主任研究員 吉田雅治氏，はじめ知能ロボット研究部諸氏には日頃から有益な討論をして戴いている。セコムIS研究所パターン情報処理研究部長 増田功博士，NTT基礎研究所情報科学研究部主幹研究員 内藤誠一郎博士には研究の心構えを御教授戴いた。皆様方の御助力に深謝します。

ローレンスリバモア研究所 Nelson Max 博士には3章の反射強度積分法についた貴重な御助言を戴いた。また、ブリティッシュコロンビア大学の Pierre Poulin 氏には氏がＮＴＴ滞在中に頻繁に御討論戴いた。面光源照明法は氏の線光源照明手法に触発された研究である。御両名に感謝します。

参考文献

[1] J. Amanatides, "Ray tracing with cones", Computer Graphics, 18(3) (Proc. *SIGGRAPH'84*), pp. 129–135 (1984).

[2] J. Barros and H. Funch, "Generating smooth 2-D monocolor line drawings on video display", Computer Graphics, 13(2) (Proc. *SIGGRAPH'79*), pp. 260–269 (1979).

[3] J.F. Blinn and M. E. Newell, "Texture and reflection in computer generated images", Comm. ACM, 19(10), pp. 542–547 (1976).

[4] J.F. Blinn, "Methods of light reflection for computer synthesized pictures", Computer Graphics, 11(2) (Proc. *SIGGRAPH'77*), pp. 192–198 (1977).

[5] L. Carpenter, "The A-buffer, an antialiased hidden surface method", Computer Graphics, 18(3) (Proc. *SIGGRAPH'84*), pp. 103–108 (1984).

[6] E. Catmull, "A subdivision algorithm for computer display of curved surfaces", Ph. D. Thesis, Report UTEC-CSc-74-133, Computer Science Department, University of Utha. Salt Lake City, UT (1974).

[7] E. Catmull, "A hidden-surface algorithm with anti-aliasing", Computer Graphics, 12(3) (Proc. *SIGGRAPH'78*), pp. 6–11 (1978).

[8] R.L. Cook and K. Torrance, "A reflectance model for computer graphics", Computer Graphics, 15(3) (Proc. *SIGGRAPH'81*), pp. 307–316 (1981).

[9] R.L. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing", Computer Graphics, 18(3) (Proc. *SIGGRAPH'84*), pp. 137–145 (1984).

[10] R.L. Cook, "Stochastic sampling in computer graphics", ACM Trans. Graphics, 5(1), pp. 51–72 (1986).

[11] F.C. Crow, "The use of grayscale for improved raster display of vectors and characters", Computer Graphics, 12(3) (Proc. *SIGGRAPH'78*), pp. 1–5 (1978).

[12] M. Dippé and E.H. Wold, "Antialiasing through stochastic sampling", Computer Graphics, 19(3) (Proc. *SIGGRAPH'85*), pp. 69–78 (1985).

[13] E.A. Feibush, M. Levoy, and R.L. Cook, "Synthetic texturing using digital filters", Computer Graphics, 14(3) (Proc. *SIGGRAPH'80*), pp. 294–301 (1980).

[14] A. Fujimoto and K. Iwata, "Jag free images on raster display", IEEE CG & A, 3(9), pp. 26–34 (1983).

[15] H. Fuchs and Z.M. Kedem, "On surface generating by priority tree structures", Computer Graphics, 14(3) (Proc. *SIGGRAPH'80*), pp. 124–133 (1980).

[16] C.M. Goral, K.E. Torrance, D.P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces", Computer Graphics, 18(3) (Proc. *SIGGRAPH'84*), pp. 213–222 (1984).

[17] P.S. Heckbert and P. Hanrahan, "Beam tracing polygonal objects", Computer Graphics, 18(3) (Proc. *SIGGRAPH'84*), pp. 119–127 (1984).

[18] D.S. Immel, M.F. Cohen, and D.P. Greenberg, "A radiosity method for non-diffuse environments", Computer Graphics, 20(4) (Proc. *SIGGRAPH'86*), pp. 133–142 (1986).

[19] M.E. Lee, R.A. Redner, and S.P. Uselton, "Statistically optimized sampling for distributed ray tracing", Computer Graphics, 19(3) (Proc. *SIGGRAPH'78*), pp. 61–67 (1985).

[20] J.T. Kajiya, "The rendering equation", Computer Graphics, 20(4) (Proc. *SIGGRAPH'86*), pp. 143–150 (1986).

[21] N. Max, "Antialiasing scan-line data", IEEE CG & A, 10(1), pp. 18–30 (1990).

[22] D.P. Mitchell, "Generating antialiased images at sampling densities", Computer Graphics, 21(4) (Proc. *SIGGRAPH'87*), pp. 65–72 (1987).

[23] 中前, 西田 " 多面体の隠れ線消去の一手法 ", 情処論, 13(4), pp. 239–246 (1972).

[24] E.M. Newell, R.G. Newell, and T.L. Sancha, "A solution to the hidden surface problem", Proc. ACM National Conference, pp. 443–450 (1972).

[25] 西田, 中前, " カラーディスプレイにおけるスムーズな線分の発生方法 ", 情処論, 22(6), pp. 505–511 (1981).

[26] 西田, 藤井, 中前, " 優先順位テーブルを用いた三次元物体の陰影表示の一手法 ", 情処論, 24(4), pp. 429–435 (1983).

[27] T. Nishita and E. Nakamae, "Half-tone representation of 3-D objects illuminated by area sources of polyhedron sources", Proc. IEEE Computer Software and Application Conference, pp. 237–242 (1983).

[28] 西田, 中前, “マルチスキャニング法によるスムーズエッジ処理を施した三次元物体の陰影表示”, 情処論, 25(5), pp. 703–711 (1984).

[29] T. Nishita and E. Nakamae, “Shading models for point and linear sources”, ACM Trans. Graphics, 4(2), pp. 124–146 (1985).

[30] T. Nishita and E. Nakamae, “Continuous tone representation of three-dimensional objects taking account of shadows and interreflection”, Computer Graphics, 19(3) (Proc. *SIGGRAPH'85*), pp. 23–30 (1985).

[31] T. Nishita and E. Nakamae, “Continuous tone representation of three-dimensional objects illuminated by sky light”, Computer Graphics, 20(4) (Proc. *SIGGRAPH'86*), pp. 125–132 (1986).

[32] B. Phong, “Illumination for computer generated pictures”, Comm. ACM, 18(6), pp. 311–317 (1975).

[33] M.L.V. Pitteway and D.J. Watkinson, “Bresenham's algorithm with gray scale”, Comm. ACM, 23(11), pp. 625–626 (1980).

[34] P. Poulin and J. Amanatides, “Shading and shadowing with linear light sources”, Proc. *Eurographics'90*, pp. 377–386 (1990).

[35] T. Saito and T. Takahashi, “Highlighting rounded edges”, Proc. *CG International'89* pp. 613–629 (1989).

[36] M.Z. Shao, Q.S. Peng, and Y.D. Liang, “A new radiosity approach by procedural refinements for realistic image synthesis”, Computer Graphics, 22(4) (Proc. *SIGGRAPH'88*), pp. 93–101 (1988).

[37] 芝本, “モーショングラフィックスにおけるエリアシングの除去”, 情処学会グラフィックスとＣＡＤ研究会, 9-1 (1983).

[38] M. Shinya, T. Takahashi, and S. Naito, “Principles and applications of pencil tracing”, Computer Graphics, 21(4) (Proc. *SIGGRAPH'87*), pp. 45–54 (1987).

[39] F. Sillion and C. Puech, “A general two-pass method integrating specular and diffuse reflection”, Computer Graphics, 23(3) (Proc. *SIGGRAPH'89*), pp. 335–344 (1989).

[40] R.F. Sproll and S. Gupta, “Filtering edges for gray-scale display”, Computer Graphics, 15(3) (Proc. *SIGGRAPH'81*), pp. 1–15 (1981).

[41] I.E. Sutherland and G.W. Hodgman, “Reentrant polygon clipping”, Comm. ACM, 17(1), pp. 32–42 (1974).

[42] 宇野, “計算機のための数値計算 (応用数学力学口座 14)”, 朝倉書店, pp. 192–197 (1964).

[43] C.P. Verbeck and D.P. Greenberg, “A comprehensive light-source description for computer graphics”, IEEE CG & A, pp. 66–75 (1984).

[44] J.R. Wallace, M.F. Cohen, and D.P. Greenberg, “A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity method”, Computer Graphics, 21(4) (Proc. *SIGGRAPH'87*), pp. 311–320 (1987).

[45] D.R. Warn, “Light controls for synthetic images”, Computer Graphics, 17(3) (Proc. *SIGGRAPH'83*), pp. 13–21 (1983).

[46] G.S. Watkins, “A real time visible surface algorithm”, Ph.D. Thesis, Technical Report UTEC-CSc-70-101, NTIS AD-762 004, Computer Science Department, University of Utha, Salt Lake City, UT (1970).

[47] K. Weiler and P. Atherton, “Hidden surface removal using polygon area solution”, Computer Graphics, 11(3) (Proc. *SIGGRAPH'77*), pp. 214–222 (1977).

[48] K. Weiler, “Polygon comparison using a graphic representation”, Computer Graphics, 14(3) (Proc. *SIGGRAPH'80*), pp. 10–18 (1980).

[49] T. Whitted, “An improved illumination model for shaded display”, Comm. ACM, 23(6), pp. 343–349 (1980).

[50] C. Wylie, G.W. Romney, D.C. Evans, and A.C. Erdahl, “Halftone perspective drawings by computer”, Proc. Fall Joint Comp. Conf. 67, Thompson Books, Washington, DC, pp. 49–58 (1976).

[51] J.D. Foley and A.Van Dam, “Fundamentals of interactive computer graphics”, Addison-Wesley Publishing Company (1982).

[52] J.D. Foley, A.Van Dam, S.K. Feiner, and J.F. Hughes, “Computer graphics; principles and practice (second edition)”, Addison-Wesley Publishing Company (1990).

[53] 中前, “コンピュータグラフィックス (ニューメディア技術シリーズ)”, オーム社 (1987).

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

著者研究業績

1. 論文

1. 田中, 杉原, 杉江, “多面体に対する寸法指定情報の適正さとその判定法”, 情報処理学会論文誌, Vol. 26, No. 1, pp. 104–111 (1985).
2. 田中, 内藤, 高橋, 増田, “Generalized Symmetry に基づく 3 次元形状 の復元”, 電子情報通信学会論文誌, Vol. J71-D, No. 1, pp. 84–91 (1988).
3. T.Tanaka, S.Naito, and T.Takahashi, “Generalized symmetry and its application to 3D shape generation”, The Visual Computer, Vol. 5, No. 5, pp. 83–94 (1989).
4. 田中, 高橋, “アンチ・エリアシングのための直交スキャンライン法”, 情報処理学会論文誌, Vol. 32, N0. 2, pp. 197–205 (1991).
5. 田中, 高橋, “面光源で照らされた物体の照度計算法”, 情報処理学会論文誌, Vol. 32, No. 11, pp. 1383–1391 (1991).
6. 田中, 高橋, “精密レンダリング法とそのハイライト生成への応用”, 情報処理学会論文誌, Vol. 33, No. 4, pp. 471–480 (1992).
7. T.Tanaka and T.Takahashi, “Precise rendering method for exact anti-aliasing and highlighting”, The Visual Computer, (Vol. 8, No. 4 に掲載予定).

2. 国際会議

1. T.Tanaka, S.Naito, and T.Takahashi, “Three-dimensional shape generation based on generalized symmetry”, Proc. *CG International'88* (New Trend in Computer Graphics, N. Thalmann and D. Thalmann Eds.), pp. 364–378 (1988).
2. T.Tanaka and T.Takahashi, “Cross scanline algorithm”, Proc. *Eurographics'90*, pp. 63–74 (1990).

3. T.Tanaka and T.Takahashi, "Precise rendering method for edge highlighting", Proc. *CG International'91* (Scientific visualization of physical phenomena, N. Patrikalakis Ed.), pp. 283–298 (1991).

4. T.Tanaka and T.Takahashi, "Shading with area light sources", Proc. *Eurographics'91*, pp. 235–246 (1991).

3. 研究会，シンポジウム

1. 田中, 杉原, 杉江, " 工業用図面における寸法指定情報の計算機処理 ", 情報処理学会技術報告, グラフィックスと CAD, 10-1 (1983).

2. 田中, 内藤, 高橋, 増田, "Generalized Symmetry に基づく三次元曲面形状の復元 ", 電子情報通信学会技術報告, PRU86-5, pp. 35–44 (1986).

3. 田中, 高橋, " 直交スキャンライン法によるアンチ・エリアシング ", 情報処理学会平成元年度グラフィックスと CAD シンポジウム, pp. 151–160 (1989).

4. 全国大会

1. 田中, 杉原, 杉江, " 平面図形における寸法指定情報の計算機処理 ", 昭和 57 年度電気関連学会東海支部連合大会 (1982).

2. 田中, 杉原, 杉江, " 平面図形における寸法指定情報の計算機処理 ", 情報処理学会第 27 回 (昭和 58 年後期) 全国大会, pp. 1575–1576 (1983).

3. 田中, 内藤, 増田, "Generalized Symmetry の仮定に基づく三次元形状の 合成 ", 昭和 61 年度電子通信学会総合全国大会, p.1645 (1986).

4. 田中, 高橋, 増田, "Generalized Symmetry に基づく立体形状の復元 ", 情報処理学会第 33 回 (昭和 61 年後期) 全国大会, pp. 1621-1622 (1986).

5. 田中, 内藤, 増田, "Generalized Symmetry 物体の形状復元 ", 電子情報通信学会創立 70 周年記念総合全国大会 (昭和 62 年), p.7-287 (1987).

6. 田中, 高橋, " ジャギ発生のメカニズムに着目した画質評価尺度 ", 1990 年電子情報通信学会秋季全国大会, p.6-427 (1990).

7. 田中, 高橋, “直交スキャンライン法によるハイライト稜線の表示”,
1990 年電子情報通信学会秋季全国大会, p.6-428 (1990).

8. 田中, 高橋, “面光源で照らされた物体のシェーディング”,
1991 年電子情報通信学会春季全国大会, p.7-378 (1991).

9. 田中, 高橋, “非線型テーブルを用いた照度計算の高速化”,
情報処理学会第 43 回 (平成 3 年後期) 全国大会, Vol.2, pp. 2-509 – 2-510 (1991).

5. その他

1. 田中, 高橋, “ＣＧ作品：「ビデオアンプ」と「ティーポット」”,
グラフィックスと CAD 研究会設立 10 周年記念 CG 作品集, p.30 (1991).

2. 高橋, 田中, “対称性を利用した立体生成”,
日本コンピュータグラフィックス協会, ニコグラフ デザイニクスシンポジウム,
pp.16–17 (1992).

6. 賞罰

1. 平成 3 年 3 月 27 日
平成 2 年度電子情報通信学会篠原記念学術奨励賞
授賞論文：“直交スキャンライン法によるハイライト稜線の表示”

2. 平成 4 年 5 月 21 日
平成 3 年度情報処理学会論文賞
授賞論文：“アンチ・エリアシングのための直交スキャンライン法”