社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－CG－121 (18)
2005／11／19

# 高速描画のためのハーフエッジ階層を利用した視点および照明依存メッシュ簡略化

宮村 史朗* 今井 桂子†

**概 要**

本研究では, 対話的なレンダリング手法を提案する. 近年, 3 次元スキャナの技術や CAD 技術の発展に伴い, 描画対象となるポリゴンモデル (三角形メッシュモデル) の複雑化が進み, モデルの描画処理に要する時間も増大している. そこで本研究では, 三角形メッシュで表現されるモデルに, ハーフエッジ階層を用いた多重解像度表現を導入し, 視点や照明に着目することでデータ量を軽減するメッシュ簡略化手法を提案する. また, 簡略化前後のモデルをディスプレイ上に描画された画像で比較することで, モデルの画像品質について考察する.

# Mesh Simplification Depending on Viewpoints and Illumination with a Half-edge Hierarchy for Fast Drawing

SHIRO MIYAMURA* KEIKO IMAI†

**Abstract**

In this paper, we consider the problem of fast drawing for complex polygonal models. Recently, three dimension scanning technology and CAD technology are developed rapidly and the polygonal models complexify more. Therefore, the processing time for drawing the model goes on increasing. We propose a mesh simplification using a multiresolution representation in order to reduce the volume of data in the model. Our method uses a half-edge hierarchy for triangular mesh models, and depends on viewpoints and illumination. Moreover, we also show experimental results and discuss the quality of the images.

## 1 序論

近年, 3 次元スキャナの技術や CAD 技術の発展に伴い, 描画対象となるポリゴンモデル (三角形メッシュモデル) の複雑化が進んでいる. CPU やグラフィクボードの革新にもかかわらず, 複雑化するモデルを画面上に対話的に描画することは難しい. 本研究では, 三角形メッシュモデルを対象に, 視界に入っていない面や視点から遠くにある面, および照明が強く当たっていない面など, 視覚的に冗長と思われる部分を簡略化し, これにより, データ量を軽減しモデルを高速に描画する手法を提案する.

**【過去の研究】** メッシュ簡略化に対する数々の方法がここ 10 年間で開発されてきている. 例えば, 三角形メッシュのプログレッシブ簡略化におけるの研究 [2] や, 効果的な幾何学的評価における研究 [1] がある.

[2] では, エッジ消去と頂点分割の 2 つの操作を用いて, モデル全体の位相を変えずに LOD 近似メッシュを構築している. さらに, [4] では, [2] の 2 つの操作をハーフエッジ構造に応用し, 視点に依存した多重解像度メッシュの効果的実装を行なっている.

*中央大学大学院 理工学研究科 情報工学専攻
Information and System Engineering Course, Graduate School of Science and Engineering, CHUO University
†中央大学 理工学部 情報工学科
Department of Information and System Engineering, Faculty of Science and Engineering, CHUO University

多重解像度メッシュを作成するには一度メッシュ全体を簡略化する必要がある. その際に, [1] では Quadric Error Metric (QEM) を導入している. 法線ベクトル $\boldsymbol{n}$ と $\boldsymbol{n}^T\boldsymbol{p}+d=0$ ($\boldsymbol{p}\in\mathbb{R}^3$ は平面上の任意の点) で表される面 $f$ に対して, 任意の点 $\boldsymbol{x}\in\mathbb{R}^3$ から平面 $f$ までの 2 乗距離誤差の関数は,

$$Q^f(\boldsymbol{x})=\boldsymbol{x}^TA\boldsymbol{x}+2\boldsymbol{b}^T\boldsymbol{x}+c$$

と定義している. ただし, $(A,\boldsymbol{b},c)=(\boldsymbol{n}\boldsymbol{n}^T,d\boldsymbol{n},d^2)$ である. さらに, メッシュの各頂点 $v$ に対して, 任意の点 $\boldsymbol{x}$ から頂点 $v$ までの 2 乗距離誤差の関数は,

$$Q^v(\boldsymbol{x})=\sum_{f\in F_v}Q^f(\boldsymbol{x})$$

と定義している. ただし, $F_v$ は $v$ に隣接する面の集合である. さらに, [5] では QEM をハーフエッジ構造に応用した One-sided Quadric Error Metric (1-QEM) を提案している.

[3] では, 照明依存のメッシュ簡略化を提案している. 頂点に隣接している面の法線ベクトルの和を頂点の法線ベクトルとし, 簡略化前後の頂点の法線ベクトルの偏差を利用して照明を考慮することで, モデルの最適な解像度を決定している.

**【研究目的】** 本研究では, 多重解像度メッシュの効果的実装を行なっている [4] の手法をベースに, 一度メッシュ全体を簡略化するためにハーフエッジ構造に適した 1-QEM [5] を利用する. そして, [3] の

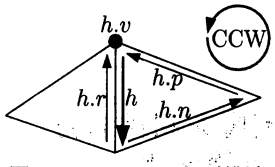



図 1.　ハーフエッジ構造

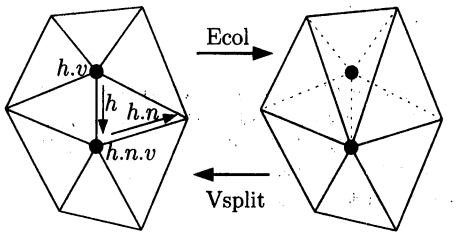



図 2.　エッジ消去 (Ecol) と頂点分割 (Vsplit)

照明を利用して解像度を選択する手法を考慮することで, 視点および照明依存メッシュ簡略化手法を提案する.

本研究の目的は,

- ユーザが視点を変化させてからモデルが再描画されるまでの待ち時間を短くすること,
- 描画された (2 次元の) 画像が, 簡略化していないモデルの画像との "見た目の違い" が少ないこと,

の 2 点である. ただし, この "見た目の違い" は画像品質として 5 節で定義する.

**【本稿の構成】** まず, 2 節では提案する描画手法の概要について述べる. 次に, 3 節 (前処理) と 4 節 (ノードテスト) では描画手法の詳細について述べる. 5 節では, 描画手法を計算機に実装し, 描画速度や画像品質で提案手法の優位性を示す. 最後に 6 節で結論を述べる.

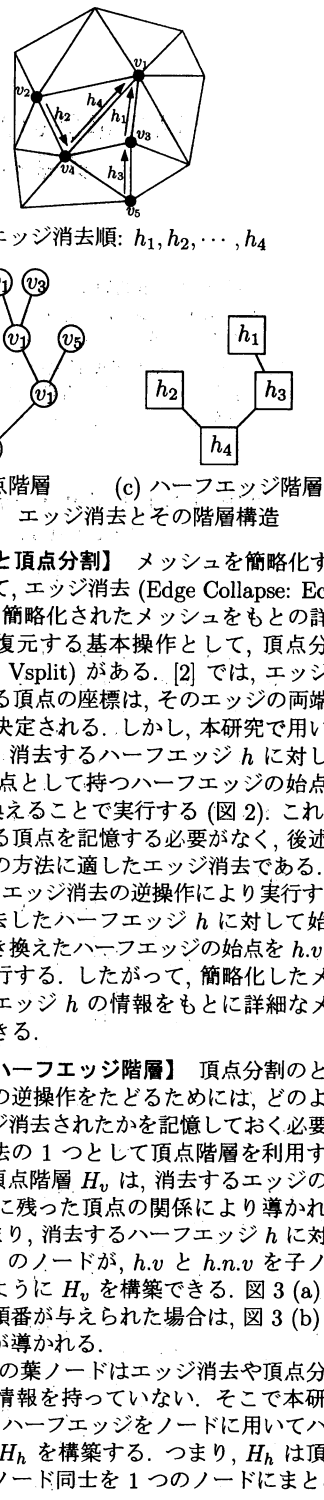
## 2　提案する描画手法

本節では, まず基本概念について説明する. その後に, 本研究で提案する描画手法の概略について述べる.

### 2.1　基本概念

ハーフエッジ構造やエッジ消去と頂点分割など, 前処理やノードテストで必要となる基本概念について説明する.

**【ハーフエッジ構造】** 本研究で用いるメッシュの各三角形は, 3 本の方向つきのエッジによって定義する. この方向つきのエッジのことをハーフエッジと呼ぶ. 図 1 に示すように, ハーフエッジ $h$ は, 1 つの三角形内で反時計回り (Counter Clock Wise: CCW) に次のハーフエッジ $h.n$, 前のハーフエッジ $h.p$, 逆向きのハーフエッジ $h.r$, および始点となる頂点 $h.v$ の情報を保持する.




(a) エッジ消去順: $h_1, h_2, \cdots, h_4$

(b) 頂点階層　　(c) ハーフエッジ階層

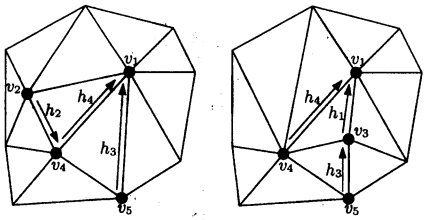
図 3.　エッジ消去とその階層構造

**【エッジ消去と頂点分割】** メッシュを簡略化する基本操作として, エッジ消去 (Edge Collapse: Ecol) がある. また, 簡略化されたメッシュをもとの詳細なメッシュに復元する基本操作として, 頂点分割 (Vertex Split: Vsplit) がある. [2] では, エッジ消去後に挿入する頂点の座標は, そのエッジの両端点の座標により決定される. しかし, 本研究で用いるエッジ消去は, 消去するハーフエッジ $h$ に対して頂点 $h.v$ を始点として持つハーフエッジの始点を $h.n.v$ に置き換えることで実行する (図 2). これは, 新たに挿入する頂点を記憶する必要がなく, 後述するコスト計算の方法に適したエッジ消去である.

頂点分割は, エッジ消去の逆操作により実行する. すなわち, 消去したハーフエッジ $h$ に対して始点を $h.n.v$ に置き換えたハーフエッジの始点を $h.v$ に戻すことで実行する. したがって, 簡略化したメッシュはハーフエッジ $h$ の情報をもとに詳細なメッシュに復元できる.

**【頂点階層とハーフエッジ階層】** 頂点分割のときにエッジ消去の逆操作をたどるためには, どのような順番でエッジ消去されたかを記憶しておく必要がある. その方法の 1 つとして頂点階層を利用する方法がある. 頂点階層 $H_v$ は, 消去するエッジの両端点と消去後に残った頂点の関係により導かれる森である. つまり, 消去するハーフエッジ $h$ に対して, 頂点 $h.n.v$ のノードが, $h.v$ と $h.n.v$ を子ノードとして持つように $H_v$ を構築できる. 図 3 (a) のエッジ消去の順番が与えられた場合は, 図 3 (b) に示す頂点階層が導かれる.

しかし, $H_v$ の葉ノードはエッジ消去や頂点分割に必要となる情報を持っていない. そこで本研究では, 消去するハーフエッジをノードに用いてハーフエッジ階層 $H_h$ を構築する. つまり, $H_h$ は頂点階層 $H_v$ の子ノード同士を 1 つのノードにまとめ,




(a) $h_1$ をエッジ消去　(b) $h_2$ をエッジ消去

図 4.　図 3 (a) の選択的メッシュ簡略化例

$H_v$ の根ノードを除いた構造をしている．図 3 (a) のエッジ消去の順番が与えられた場合は，図 3 (c) に示すハーフエッジ階層が導かれる．

**【選択的メッシュ簡略化】**　エッジ消去の階層構造を利用することで，部分的に簡略化または部分的に詳細なメッシュを作成することができる．これを選択的メッシュ簡略化と呼ぶ．ただし，ハーフエッジ階層 $H_h$ に対して，親ノードよりも子ノードのエッジ消去を優先したメッシュを選択する必要がある．図 3 (a) を選択的に簡略化した例を図 4 に示す．図 3 では，$h_4$ は $h_2$ や $h_3$ を先にエッジ消去しておく必要があるため，$h_4$ のみをエッジ消去することはできない．

### 2.2　提案手法の概略

提案する描画手法は，まずデータ入力後，前処理でハーフエッジ階層を構築する．その後，ユーザが視点を変化させるたびにノードテスト，レンダリングの順で行なう (図 5)．

本研究で対象とするデータは，中規模の三角形メッシュデータである．ただし中規模データとは，サーバクライアント型で描画するほど大規模なデータではなく，要素 (三角形) 数が 10,000 から 100,000 程度 (ファイルサイズ 1 から 4 MB) のデータとする．また，入力するデータは，頂点の座標および面の頂点インデックスのみを持つデータとする．そこで，前処理よりも前に頂点や面の隣接関係を取得しておく．

**前処理**　前処理では，モデルを描画するための準備としてハーフエッジ階層を構築する．ハーフエッジ階層は，メッシュ全体を簡略化する過程で構築する．前処理について詳しくは 3 節で説明する．

**ノードテスト–レンダリング**　前処理の後やユーザが視点を変化させた後に，ノードテストでメッシュを選択的に簡略化し，レンダリングでメッシュを描画する．ノードテストについて詳しくは 4 節で説明する．

ユーザが視点を変化させてからモデルが再描画されるまでの待ち時間は，ノードテストにかかる時間とレンダリングにかかる時間の合計となる．

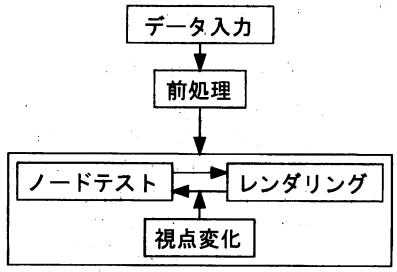



図 5.　提案する描画手法の概略

## 3　前処理

本節では，前処理で行なうハーフエッジ階層の構築方法を説明する．ハーフエッジ階層を構築するためには，一度メッシュ全体を簡略化する必要がある．$n$ 個の頂点を減らすために，オリジナルメッシュ $M^n$ を，$n$ 回のエッジ消去によってモデル全体の位相を変えずにより要素数の少ないメッシュ $M^0$ に簡略化する．この $M^0$ を基底メッシュと呼ぶ．

基底メッシュを作成する過程で，ハーフエッジ階層においてどのノードがどのノードの親になるかを決定する．基底メッシュを作成するために，すべてのハーフエッジに対して，エッジ消去を実行したときにモデル全体へ与える形状の影響度合いを決定する．これをハーフエッジのコストと呼ぶ．

**【ハーフエッジのコスト計算】**　ハーフエッジのコスト計算には，[5] の 1-QEM と Flatness Criterion を用いる．まず，1 節で紹介した QEM をハーフエッジ構造に適用させる方法を説明する．消去するハーフエッジ $h$ に対して，頂点 $h.n.v$ の座標を $\boldsymbol{x}_{h.n.v}$ とする．このとき，ハーフエッジ $h$ の誤差は，

$$Q^h(\boldsymbol{x}_{h.n.v}) = \frac{Q^v(\boldsymbol{x}_{h.n.v})}{\sqrt{\deg(h.n.v) - 2}}$$

として計算する．ただし，$\deg(h.n.v)$ は $h.n.v$ に隣接する面の数とする．

次に，Flatness Criterion について説明する．頂点 $h.v$ が，$k$ 個の面 $f_i$ $(i = 1, \ldots, k)$ と隣接しているとする．また，ハーフエッジ $h$ を消去すると $f_i$ が $f_i'$ によって置き換えられるとする．$f_i$ と $f_i'$ がなす角度を $\alpha_i$ とする．$\alpha_i$ が大きいとエッジ消去後にモデルの形状を大きく変える可能性がある．そこで，$h$ に対する Flatness Criterion $w(h)$ を

$$w(h) = \begin{cases} \frac{2}{n}\sum_{i=1}^{k}(1 - \cos\alpha_i) & \alpha_i < 90 \\ \infty & \alpha_i \geq 90, \end{cases}$$

として定義する．ただし $\cos\alpha_i$ は，2 つの面の法線ベクトルから計算する．ハーフエッジ $h$ のコスト $C^h(h)$ は，

$$C^h(h) = Q^h(\boldsymbol{x}_{h.n.v}) \cdot w(h)$$

として計算する．

**【頂点のコスト計算】** 次に, 頂点のコストを考える. 頂点 $v$ のコスト $C^v(v)$ は,

$$C^v(v) = \min_{h \in E_v} C^h(h)$$

として計算する. ただし, $E_v$ は, $v$ を始点として持つハーフエッジの集合とする. すべての頂点のコストを計算した後に, コストの小さい順 (以下, コスト順) でヒープを作成し, コスト順にエッジ消去を行なうことで基底メッシュを作成する.

また, エッジ消去の後には影響を受けた頂点のコストを更新し, ヒープも更新する操作が必要となる.

前処理のアルゴリズムは以下のようになる.

Step 1 すべてのハーフエッジ $h$ に対して $C^h(h)$ を計算する.

Step 2 すべての頂点 $v$ に対して, $C^v(v)$ を計算する.

Step 3 頂点のコストを用いてヒープを作成する.

Step 4 基底メッシュが得られるまで以下を繰り返す.

(a) ヒープから最小の頂点のコストを持つ頂点を取り除く.

(b) 取り除いた頂点のコストを最小にしていたハーフエッジに対して, エッジ消去可能ならば実行し, 不可能ならばヒープを更新し (a) へ.

(c) エッジ消去で影響を受けた頂点のコストを再計算し, ヒープも更新する.

ただし, Step 4 の反復は簡略化前後の頂点数の比が $\alpha$ になるまで繰り返す ($\alpha = 1$ でオリジナルメッシュ, $\alpha = 10$ でオリジナルメッシュの 1/10 の頂点数). ユーザは, $\alpha$ の値を入力することでハーフエッジ階層の深さを制御できる. Step 4 (b) でモデルの位相を変化させるエッジ消去は実行不可能とする. また, 4 節で述べる bounding sphere の半径 $t.r$ や normal cone の頂角の正弦 $t.\sin\theta$ も Step 4 (b) で計算する.

Step 1 の後にハーフエッジのコスト順でヒープを作成して, 基底メッシュの作成もできる. しかし, 消去するハーフエッジ $h$ に対して, 頂点 $h.v$ を始点として持つ他のハーフエッジのコストは, ヒープから削除されるため計算効率が悪い. なぜならば, 頂点のコスト順でヒープを作成した場合は, 1 回のエッジ消去につき消去される頂点は 1 つであり, 他の頂点をヒープから削除することはないからである.

## 4 ノードテスト

前処理の直後やユーザが視点を変化させた後に, 視点および照明を考慮して適切なメッシュを選択する. このときに, 前処理で構築したハーフエッジ階層 $H_h$ に対して, ノード $t \in H_h$ のハーフエッジをエッジ消去するかをノードテストにより決定する.

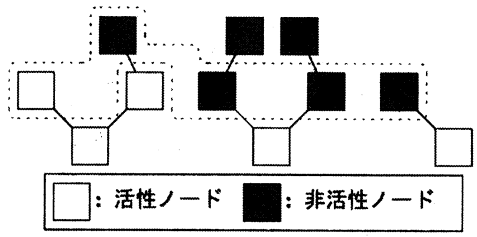



図 6. 活性ノード, 非活性ノードおよびフロントノード (点線内)

[4] では視点依存テストのみを実行していた. しかし, 実際にコンピュータグラフィクスや CAD ビューアーでモデルを描画するときに照明を考慮する場合がある. そこで本研究では, 照明を考慮することで詳細なメッシュに復元するかを決定する照明依存テストを視点依存テストに加えて実行する.

本節ではまず, ノードテストの対象となるフロントノードについて定義する. 次に, ノードテストに必要となる bounding sphere と normal cone について述べる. これらは 3 節の最後で述べたとおり前処理中に計算される. 本節の最後に視点依存テストと照明依存テストについて簡単に述べる.

### 4.1 フロントノード

ハーフエッジ階層において, 現在, 描画している (選択している) メッシュでエッジ消去を実行していないノードを活性ノード, 実行しているノードを非活性ノードとする (図 6). 例えば, 図 4 (a) では, $h_1$ が非活性ノード, $h_2, h_3, h_4$ が活性ノードとなる. さらに, $F_1$ を親ノードが活性ノードである非活性ノードの集合とし, $F_2$ を親ノードが活性ノードである活性な葉ノードの集合とする. このときフロント $F$ を $F = F_1 \cup F_2$ と定義し, そのノード $h \in F$ をフロントノードと呼ぶ.

すべてのノードテストは, フロントノードに対して行なわれる. また, フロントは視点の変化後および 1 つノードに対するノードテストが終了した後にも変化する.

### 4.2 bounding sphere と normal cone

ノードテストに必要な情報として, 以下に示す bounding sphere の半径と normal cone の角度がある. これらは共に, ハーフエッジ階層の各ノードに記憶される.

ハーフエッジ階層 $H_h$ のノード $t \in H_h$ に対して, $t$ とその子孫のエッジ消去によって影響を受ける三角形をすべて囲む最小の球を考える. この球を bounding sphere と呼ぶ (図 7). $t$ は, bounding sphere の半径を記憶する.

ノード $t \in H_h$ に対して, $t$ とその子孫のエッジ消去によって影響を受ける頂点の法線ベクトルをすべて囲む最小の円錐を考える. この円錐を normal cone と呼ぶ (図 8). $t$ は, normal cone の母線と頂

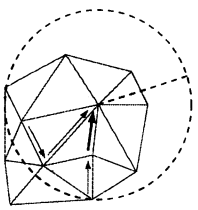


図 7. bounding sphere

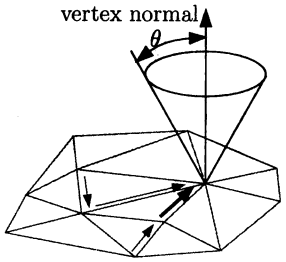



図 8. normal cone

点の法線ベクトルがなす角 $\theta$ の正弦 ($\sin\theta$) を記憶する.

### 4.3 視点依存テストと照明依存テスト

視点依存テストとして, 次の 5 つを考慮する.

- 視錐外部分の簡略化: 視錐 (視界) の外にある部分は簡略化する.
- バックフェイスの簡略化: 視点方向を向いていない面は簡略化する.
- 輪郭の保存: 視点から見てモデルの輪郭となる部分は保存する.
- スクリーン投射面積による簡略化: 視点から遠くにある三角形は, スクリーン上では小さな三角形で表現される. ユーザは面積の下限 $\tau$ を指定し, これより小さな面積を持つ三角形は簡略化する.
- 非陰影部分の簡略化: ほぼ等しい法線ベクトルを持つ面は, 1 つの大きな三角形で表現できる可能性がある. ユーザは陰影の下限 $\varphi$ を指定し, これより小さな表面法線ベクトルの偏差を持つ三角形は簡略化する.

照明依存テストとして, 次の 2 つを考慮する.

- 光源からみた輪郭の保存: 光源から見たときにモデルの輪郭となる部分は, 明暗がはっきりと分かれる部分であり保存する.
- ハイライト部分の保存: 光が強く当たっているハイライトの部分は, ユーザが照明を強く当てており重要な部分の可能性がある. ユーザはハイライトの反射角の上限 $\rho$ を指定し, これより小さなハイライトの反射角をもつ三角形は保存する.

ユーザは, $\tau$, $\varphi$ および $\rho$ を入力することで 5 節で述べる再描画速度と画像品質を制御することができる.

## 5 実験

本節では, 提案する手法の計算機実験の結果について述べる. 表 1 に示す実験環境で実験を行なった. また, グラフィクライブラリとして OpenGL を用いた.

スタンフォード兎モデル (頂点数 35,947, 面数 69,451) を対象として実験を行なった. また, 3 節で説明した頂点数の比 $\alpha$ と 4 節で説明したスクリーン投射面積の下限 $\tau$, 陰影の下限 $\varphi$, ハイライトの反射角の上限 $\rho$ を指定し以下 4 つの条件で実験を行なった.

1. オリジナルモデル: ($\alpha = 1$).
2. 視点依存モデル:
   ($\alpha = 10$, $\tau = 2.0 \times 10^{-9}$, $\varphi = 3$, $\rho = 0$).
3. 照明依存モデル:
   ($\alpha = 10$, $\tau = 1.0$, $\varphi = 0$, $\rho = 30$).
4. 視点および照明依存モデル:
   ($\alpha = 10$, $\tau = 2.0 \times 10^{-9}$, $\varphi = 3$, $\rho = 30$).

ここで, すべての条件において 4 節で述べた 7 つのノードテストのすべてを行なっている点に注意されたい.

これらの実験環境, モデルおよび条件で描画速度を検証する実験とディスプレイ上に描画された画像比較による品質評価実験の結果を示す. また, 表 2 に実験結果をまとめたものを示す.

### 5.1 描画速度実験

モデルの中心から視点までの初期距離を 1 とし, 拡大率 $z$ を距離の逆数として定義する. 描画速度実験として, 拡大率 $z$ を 1.0 から 3.0 まで 0.1 づつ増加させ, 視点をモデルに近づける操作を 40 回繰り返した. 各シーンを再描画するまでの平均時間を表 2 に示す. 再描画にかかる時間は, ノードテストにかかる時間とレンダリングにかかる時間にわけられる. また $\alpha = 10$ では, 前処理に 8.96 秒を要した. 表 2 の再描画時間を見ると, 照明依存テストを考慮しても再描画までの時間に大きく影響を与えないことがわかる.

### 5.2 品質評価実験

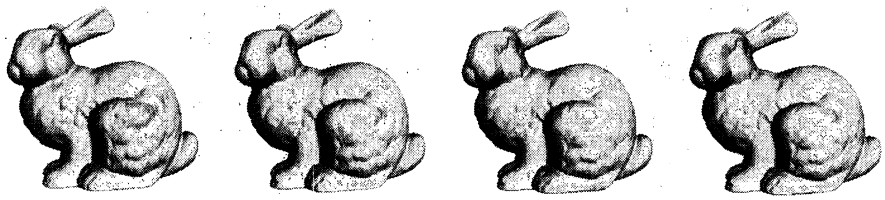
まず, 画像の品質について定義する. 画像の品質は, ディスプレイ上に描画された画像をオリジナルのモデルと比較することで評価する. つまり, オリジナルモデルの画像と簡略化したモデルの画像における各ピクセルの RGB 値の差分を 2 乗し, ピクセル数で平均化した平均 2 乗誤差 (MSE) で評価する. 簡略化したモデルの画像は, MSE 値が小さいほどオリジナルモデルの画像に視覚的に近いことを示す.

表 1. 実験環境

| | |
|---|---|
| **OS** | Windows XP Pro (SP2) |
| **CPU** | Pentium 4 (2.8GHz) |
| **Memory** | 512MB RAM |
| **Video Card** | NVIDIA GeForce2 MX/MX 400 |

表 2.　実験結果

| 条件 | 描画速度実験結果 [ms] テスト | レンダリング | 再描画合計 | 品質評価実験結果 MSE |
|---|---|---|---|---|
| オリジナル | - | - | 216.46 | - |
| 視点依存 | 41.05 | 68.75 | 109.80 | 149.81 |
| 照明依存 | 40.19 | 72.32 | 112.50 | 181.29 |
| 視点・照明依存 | 43.90 | 75.96 | 119.86 | 135.52 |



(a)　(b)　(c)　(d)

(e)　(f)　(g)

図 9.　スタンフォード兎モデルの描画例, (a) オリジナルモデル, (b) 視点依存モデル, (c) 照明依存モデル, (d) 視点および照明依存モデル, (e) 視点依存モデルの差分画像, (f) 照明依存モデルの差分画像, (g) 視点および照明依存モデルの差分画像

前述の 4 つの条件に対して, 拡大率 $z = 2.4$ における $443 \times 443$ ピクセルの画像を 1/5 に縮小した画像を図 9 の上段に示す. また, 各ピクセルでの差分の 2 乗を反映させた画像を差分画像として図 9 の下段に示す.

図 9 (f) を見ると, (e) に比べ光が強く当たっている部分が白くなっていることがわかる. これは, オリジナルモデルの画像との差分が小さいことを示す. しかし, MSE 値の総和は (e) よりも (f) は大きい. 視点と照明の両方を考慮したモデル (g) については MSE 値の総和が小さく, かつ光が強く当たっている部分の差分が小さい.

## 6　結論

本研究では, 視点や照明に注目した高速レンダリング手法を提案した. 三角形メッシュで表現されるモデルにハーフエッジ階層 [4] を用いた多重解像度表現を導入した. ハーフエッジ階層構築には, [5] の 1-QEM と Flatness Criterion を用いた. 視点依存テストを用いて視覚的に冗長な三角形を簡略化し, 照明依存テストを用いて光源から見て輪郭となる部分や光が強く当たっている部分を保存した. また, 画像品質をオリジナルモデルの画像との MSE 値として定義し, 視点と照明を考慮することの有効性を示した.

今後の課題として, シェーディング効果を考慮して MSE 値を小さくすることや, ユーザ操作の待ち時間にノードテストを実行し, 再描画時間を短縮することが考えられる.

## 参考文献


[1] M. Garland, P. Heckbert, "Surface simplification using quadric error metrics," *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 209–216, 1997.

[2] H. Hoppe, "Progressive meshes," *In Computer Graphics Proceedings*, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings), pp. 99–108, 1996.

[3] R. Klein, A. Schilling, "Efficient rendering of multiresolution meshes with guaranteed image quality," *The Visual Computer*, Vol.15, No.9, pp. 443–452, 1999.

[4] R. Pajarola, C. DeCoro, "Efficient Implementation of Real-Time View-Dependent Multiresolution Meshing," *IEEE Transactions on Visualization and Computer Graphics*, Vol.10, No.3, pp. 353–368, 2004.

[5] W. Dong, J. Li, and J. Kuo, "Fast Mesh Simplification for Progressive Transmission," *IEEE International Conference on Multimedia and Expo (III)*, pp. 1731–1734, 2000.