グラフィクスとＣＡＤ 103−4
（２００１．　４．２０）

三角形メッシュの効率的表示のための階層的詳細度表現

神谷周　鈴木宏正　川地克明
東京大学大学院工学系研究科 精密機械工学専攻
E-mail: {*kamiya, suzuki, kawachi*}*@cim.pe.u-tokyo.ac.jp*

本論文では、メッシュモデルの面を評価関数によって階層的にグループ化し、効果的に描画時の計算量を減らす手法を提案する。またグループ形状を段階的に簡略化し、高速な描画とユーザが望む詳細度でのモデル表示を実現する。さらにデータを逐次ファイルから読み出すことによって必要なメモリ量を減らし、小型のマシンで複雑なメッシュを扱うことを可能とした。そして、この手法を実装したシステムを用いて手法の有効性を確認した。

LODs for efficient displaying triangle meshes

MAKOTO KAMIYA　HIROMASA SUZUKI　KATSUAKI KAWACHI
Dept. Precision Engineering, University of Tokyo.
E-mail: {*kamiya, suzuki, kawachi*}*@cim.pe.u-tokyo.ac.jp*

We propose the method that reduces the amount of calculation at the time of drawing by grouping faces of a dense triangle mesh with an evaluation function. And by simplifying the groups form gradually, we realize high-speed drawing and detail view that users need. Furthermore, we reduce the required amount of memories by reading data from a file layer by layer, so we can treat a lot of faces on a small computer. We implemented a prototype system and demonstrate some examples.

1　はじめに

複雑な形状を計算機内で表現するための手法として、3 次元スキャナによって測定された点群をもとに形状復元手法を用いてメッシュモデルを生成することが一般的に行われている。この方法では、測定装置の性能の向上により高解像度の測定が可能になってきているが、同時に生成されるメッシュの面数は非常に多くなり、従来では考えられないほど大きな負荷を計算機にかけるようになってきた。また、求められる CG の画質の向上により、より複雑なモデルが扱われる傾向になってきている。

本研究では、大量の面で構成された三角形メッシュを小型の計算機でも扱うことができるようにすることを目的とする。そのためには、以下のような要件を満たす必要がある。

- 高速な描画が可能であること。
- 高速な計算を必要としないこと。
- 大量のメモリを必要としないこと。

本研究では、効果的に Culling を行い物体の描画を省略する処理するためのグループ化を行い、階層的な詳細度表現を実現する手法を提案する。

2　手法の概要

本研究の基本的な考え方は、裏側を向いている面の描画を省略する Backface Culling である。しかし単純に裏面の描画を省略するだけでは、全ての面を探索する必要があるため効果が小さい。

Hoff は物体の法線によって面をグループ分けし、面の判定をまとめて行う手法を提案した [1]。この手法は、前処理として立方体の中心とそれぞれの面から成る錐台を作り、全ての面を法線の方向によって各錐台に分類しておく。描画時は、まず錐台の可視・不可視チェックを行い、チェックが通ったものだけをさらに判定を行う。法線をある程度まとめることによって、一度に複数の面の Culling が可能になるのである。

本研究でも同じような方向を向いている面のグループ化を行い、グループごとに可視・不可視を判断することで裏面判定にかかるコストを抑え、計算量を軽減する。グループの生成は評価関数を用いて行い、パラメータを変更することで様々な要求に対応したグループ化を行うことができるようにした。また、グループを階層的に表現することで効率的な探索を可能にする。さらに、グループ形状を段階的に簡略化することにより面の数を階層的

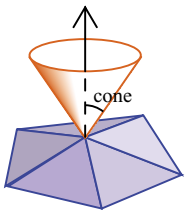


図 1: 法線と法線の広がり

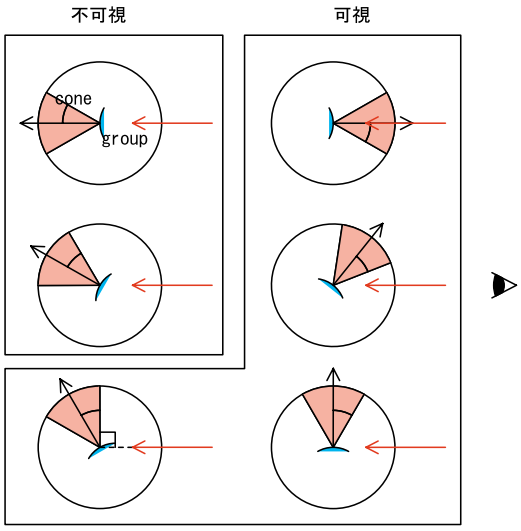


図 2: 可視・不可視の例。視線は全て右から左の方向である。

に減らし、メッシュモデルを素早く視覚化できるようにした。

3　法線と cone

同じような方向を向いている面をグループ化していく上で必要な情報は、グループの法線と法線の広がり (cone と呼ぶ) である (図 1)。 cone はグループ内に存在する全ての面の法線の広がりを表し、グループの法線とグループ内の各面の法線がなす最大角を意味する。

グループの法線と cone により、図 2のようにある視線におけるグループの可視・不可視領域が決定される。

ここで、グループの法線ベクトルと視線のベクトルの逆方向とのなす角を α,cone の角度を c とすると、可視・不可視の判定は以下のように表すことができる。

$$\alpha - c \begin{cases} \leq \pi/2 & \cdots \text{可視} \\ > \pi/2 & \cdots \text{不可視} \end{cases} \qquad (1)$$

このように、グループの法線と cone の情報が分かっていれば、式 1を用いることで全ての面を探索しなくてもグループごとに可視・不可視が判断できるようになる。

4　評価値を用いたグループ化

本研究では、評価値を用いてグループ化を行う手法を提案する。この評価値はグループの法線、 cone、形状など、グループ化に関係するパラメータを評価関数を用いて計算する。評価値をソートし、優先順位の高いものからグループ化を行うことで、グループ形状の制御が可能になる。

4.1　評価関数

評価関数のパラメータには、以下のようなものを採用した。各項目の評価値は $0 \leq v \leq 1$ に正規化しており、より大きな値の方が評価が高いものとした。

- cone による評価
  cone の大きさによる評価であり、グループの cone がなるべく小さくなるようにするための評価である。

$$v_c = (\pi - \alpha)/\pi \quad (\alpha : \text{coneの角度})$$

- 形状による評価
  グループ全体の輪郭をユーザーが意図する形状に近づけるための評価である。今回は円形に近づける事を目標とし、グループの法線を法線とするような平面に投影した図形と外接円の面積の比で判定した。

$$v_s = S_p/S_c \quad (S_p : \text{投影面の面積}, \; S_c : \text{投影形状の外接円の面積})$$

- グループの面数による評価
  各グループの面数をなるべく減らすことによって、他のグループと面数ができるだけ均等になるようにするための評価である。

$$v_n = (N - n)/N \quad (N : \text{物体の面数}, n : \text{グループの面数})$$

これらの評価の重み付き線形和を評価関数とする。グループ形状や、各グループの cone や面数のばらつきは、重み k を変化させることで制御することができる。

$$v = k_c v_c + k_s v_s + k_n v_n \qquad (2)$$

4.2　cone の閾値とグループ化のアルゴリズム

式 2によって面と面の組合せが数値的に評価できる。さらに、 cone の値に閾値を設け、ある一定以上の大きさの cone ができないようなグループ化の条件を使用することができるようにした。これを利用したグループ化のアルゴリズムは以下のようになる。

1. 隣接する面どうしの組合わせを全て評価する。全ての面を調べることも可能ではあるが、組合せ数の増大が起きてしまう。

2. 組合わせたときに cone がある閾値以上になる組合わせを評価対象から外す (cone の閾値条件を使うとき)。

3. 評価値をソートして、最も優先度の高い組合せ (f_1, f_2) を取り出す。

4. f_1, f_2 をグループ G としてグループ化。

5. G に所属する面に関連した組合せの評価値を評価対象から外す。

6. G に隣接する面の評価値を再計算。

7. グループ化対象の組合わせがなくなるまで 2 へ戻って処理を繰り返す。

この処理を繰り返すことで常に優先度の高いものをグループ化していくことが保証される。

本研究では、各グループをさらにグループ化し、階層的に表現することで効率を高めるようにした。階層化の処理は、面ではなくグループを対象に考え、元の面を第 1 番目の階層のグループと見なすことで前節と同じアルゴリズムを用いることができる。ある階層に対して一つもグループ化対象の組合せが無いときや、最上位の階層のグループ数がある閾値以下になったときは処理を終了する。

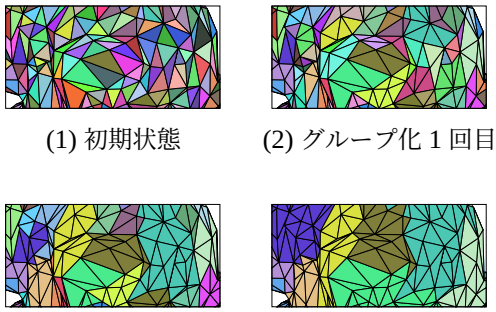
なお、グループの法線と cone の計算では平均ではなく正確な値を計算しているが、通常この処理はグループ内のある 3 つの面の法線によって作られる cone が、それ以外の面の法線を全て内包するかどうかを調べなければいけないため、組合せの数は ${}_4\mathrm{C}_4$ であり、計算量は $O(n^4)$ となってしまう。本研究では法線と cone を求める際、まず平均法線を求め、この平均法線となす角度によって全ての法線をソートし、角度の大きなものから cone の候補とすることにした。これにより、正確性を維持しながら大幅な高速化を実現した。

4.3 グループ化の様子

ここまでで述べた方法により、メッシュを階層的にグループ化したデータを生成することができる。階層の違うグループどうしには親子関係が存在し、一つの親に対して複数の子供グループが所属している。階層が上るにつれてグループ化の条件が緩められることになるので、グループ数は少なくなり、所属する面数は多くなっている。図 3はグループ化の様子を示した例である。

5 実験

評価値によるグループ化が効果的に行われているかを検証するため、式 2の各係数を変化させてグループ化の実験を行った。評価は、生成したモデルを通常の Backface Culling における判定処理の計算量と比較して行った。計算量は判定処理の数と比例するものとする。 Backface Culling の判定処理の数はメッシュの面数と等しい。一方グループ化を行ったモデルでは、グループの数と描画が必要なグループ内の面数の和が、必要な判定処理の数である。以降では、 Backface Culling で判定をしたときの計算量を 1 としている。



(1) 初期状態　(2) グループ化 1 回目

(3) グループ化 2 回目　(4) グループ化 3 回目

図 3: グループ化の様子

5.1 実験結果

結果を図 4,5,6に示す。階層的なグループ化では、グループの数は離散的にしか得られないので非常に確認しにくいものになってしまうため、この図では補間して面群に近似し、確認しやすいようしている。 Group 数が多いほどグループ化がされていない場合を意味する。赤い半透明の平面は Backface Culling を行なった場合の計算量である。

cone の評価値である k_c は効果が認められたが、 k_s, k_n については効果が小さかった。表示の効率化という観点では、形状やグループ面数は影響が小さいと言える。

6 グループ形状の簡略化

本研究ではグループ化を行った階層構造を階層に合わせて簡略化をし、表示時には逐次的に詳細化をすることで素早い可視化を実現する。

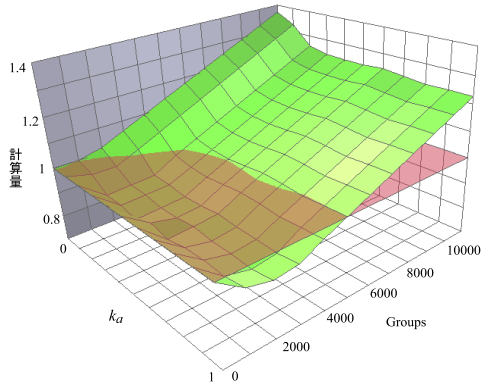
6.1 QEM

簡略化は QEM (Quality Error Metric) [2] という手法を用いて行う。この手法は Edge collapse と呼ばれる操作を繰り返すことによって実現される (図 7)。

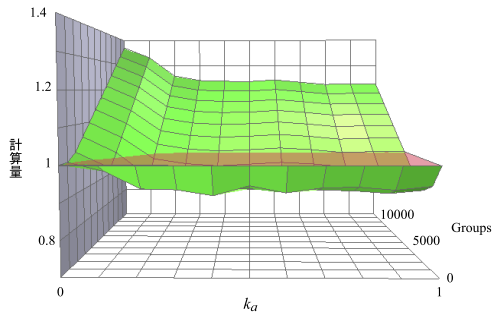
この Edge collapse 操作は、エッジの両端の頂点 $\mathbf{v}_1, \mathbf{v}_2$ を頂点 $\bar{\mathbf{v}}$ に統合する事で 1 つのエッジを消去 (それに伴って 1 頂点,2 面が消去される) する操作である。

6.2 QEM によるグループ形状の簡略化

グループ形状の簡略化時においては、グループの輪郭を崩さないように注意する必要がある。グループに関し



(1) 俯瞰図



(2) 正面図

図 4: cone 係数の変化による計算量の違い

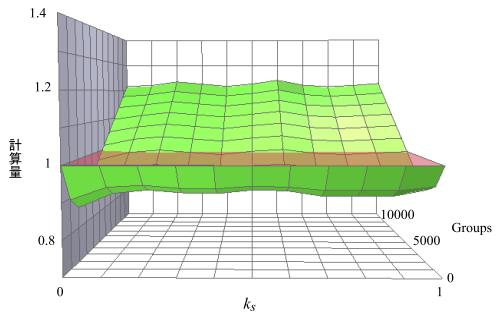


図 5: 形状係数の変化による計算量の違い

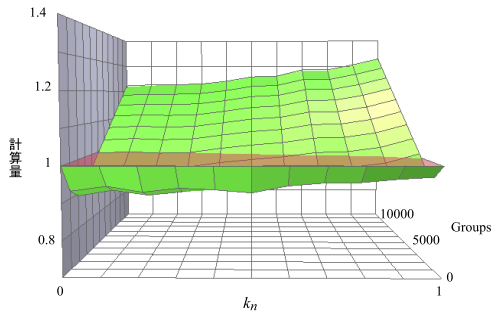


図 6: グループ面数係数の変化による計算量の違い

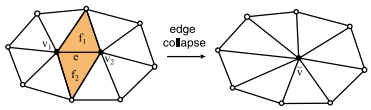


図 7: Edge collapse 操作

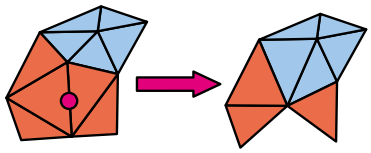


図 8: グループの分割が起きる例

て何も考慮せずにモデルの簡略化を行うと、グループが分割されたり消滅するなどして、グループとして成り立たなくなってしまう可能性がある (図 8)。

これを防ぐために、図 7のような辺 e の Edge collapse 時に以下のような条件を設ける。

条件 1 v_1, v_2 のいずれかがグループの境界上に無く、 f_1 の隣接面のうち f_2 以外の面群に同一グループの面が一つ以上存在すること。同様の条件が f_2 についても成り立つこと。

条件 2 v_1, v_2 いずれかの端点周りのグループ数が 2 以下であること。

条件 1 は Edge collapse により、グループが分割・消滅しないためのものであり、条件 2 は 3 つ以上のグループの境界となる点どうしを統合して多くのグループが隣接する頂点が生成されるのを防ぐためのものである。

これらの条件を満たすように簡略化を行うことで、グループの整合性を保ちながらグループ形状の簡略化を行うことが可能になった。

ただし、階層化した形状を逐次詳細化して表示することを考慮すると、グループの輪郭を正確に保ちながら簡略化を行う必要がある。というのは、全ての表示グループが同一階層のとき、ある一つのグループだけを詳細化するとグループ間にギャップ等の不整合が起きる可能性があるからである (図 9)。これを防ぐためには、 Edge collapse の対象となるエッジがグループどうしの境界エッジにならないように注意すればよい。

しかし、輪郭を簡略化しないとモデルの面数はあまり減らせず、簡略化の効果が著しく低くなってしまう。また、完全に詳細化が完了した時点では全ての可視面の階層レベルは統一され、グループ間の整合性は保たれるので、本研究では詳細化表示の中間段階でのグループ間の不整合は無視することにした。

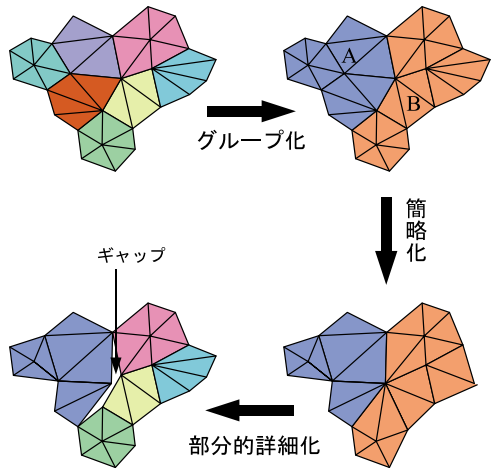


図9: ギャップの例。輪郭を簡略化するとグループ間ギャップが生成される

7　表示の詳細化

このようにして生成されたモデルの上位層では非常に簡単なメッシュになっているため、最初に最上位層を表示することで高速な視覚化が可能になる。もちろん、より詳細なメッシュを表示しようとすると計算量が多くなるが、詳細化は計算機の負荷が小さくなったときに行うことにより、ユーザは対話的な操作をすることができる。すなわち、モデルの回転や移動などが行われ、画面の更新が頻繁なときは一番簡単なモデルを表示し、ユーザが操作を止めている間に詳細化を進めるのである。

ここで、詳細化にあたっては視線と平行に近い面より垂直に近い面の方をモデルの特徴をよく表していると考え、優先的に詳細化することにした。また、 cone の大きなグループの方がばらつきが多く、もともと滑らかでは無い領域なので同様に優先度を高くした。

本研究では、次のような評価関数により詳細化を行うグループの優先順位を決定する。

$$f(g) = \begin{cases} \left(1 - \cos\left(\mathrm{angle}(\mathbf{e},\mathbf{n}) + \mathbf{c}\right)\right)/2 \\ \quad \cdots \left(\mathrm{angle}(\mathbf{e},\mathbf{n}) + \mathbf{c} < \pi \text{のとき}\right) \\ 1 \cdots \left(\mathrm{angle}(\mathbf{e},\mathbf{n}) + \mathbf{c} \geq \pi \text{のとき}\right) \end{cases}$$

g は対象のグループ、 $\mathbf{e}$ はグループ上の点から視点へのベクトル、 $\mathbf{n}$ はグループの法線であり、 c はグループの cone の大きさを表す。 $0 \leq f(g) \leq 1$ であり、より大きな値を持つグループの方が優先的に詳細化されるべきグループであることを意味する。

ただし、前述したように隣接した異なる階層のグループ間ではギャップができる恐れがあるため、特徴部分だけを優先した深さ優先探索により詳細化を行うと、多数のギャップが生成されてしまう可能性がある。これを少なくするため、階層の差を最大でも 1 とする制限を設けた広さ優先探索で詳細化を行うことにした。

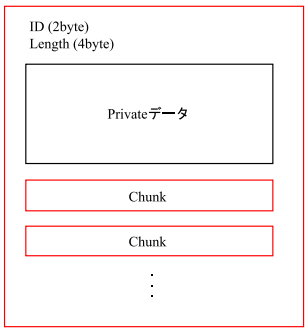


図 10: Chunk 構造。 Chunk が集まってファイルを形成している。

8　ファイル構造

本研究では、詳細なメッシュでも素早く視認できるようにするため、表示の詳細度を徐々に上げていく方法を使用した。このとき、元のデータ以外に様々な情報が付加されながら階層化を行うため、一つのモデルに対するデータ量は元の状態に比べてかなり大きくなってしまい、特に面数が多いメッシュデータを使用した場合は、計算機のメモリ上に展開できないほどのサイズになる恐れが出てきた。そこで、本研究ではメモリ上にはあまりデータを格納せず、必要に応じてデータをファイルから読み出すことにした。

本研究で使用するファイルは “Chunk” と呼ぶ複数のデータの塊によって構成されるバイナリデータとした。Chunk の基本構成は ID と Chunk 自身のサイズ、データ部分である。データ部分はその Chunk の種類に応じたデータと複数の Sub Chunk から成り、階層構造を形成する (図 10)。 Chunk の種類は、必要最低限のメッシュ表現に加え、本研究で必要となる階層的なグループ表現ができるように配慮した。このようなデータ構造により、一度に必要なメモリ量を削減しつつ、効率的にデータを読み込むことが可能になる。

9　適用例

適用例として、図 11のようなモデルについて簡略化を伴った階層化データを作成した。元のモデルの面数は 69,451、頂点数は 35,947 である。階層的な簡略化後は、図 12のように 8 つの階層からなるデータとなった。

このデータを様々な方向から視覚化し、表示速度の比較を行った。実験に使用した計算機は SHARP の PC-PJ2 (PentiumII 233MHz) というノートパソコンである。グラフィクスアクセラレータは搭載していない。描画速度を測定した結果を表 1に示す。

元モデルの初期化時は、膨大な量のデータを読み込むことによりメモリのスワップが起き、非常に時間がかかっている。一方、階層化したデータの方は初期化速度に向上が見られた。一方、詳細な形状を表示する場合はメモ

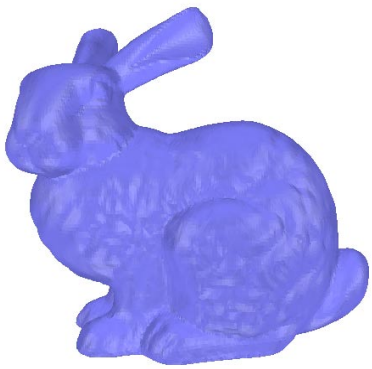


図 11: 使用したうさぎのモデル

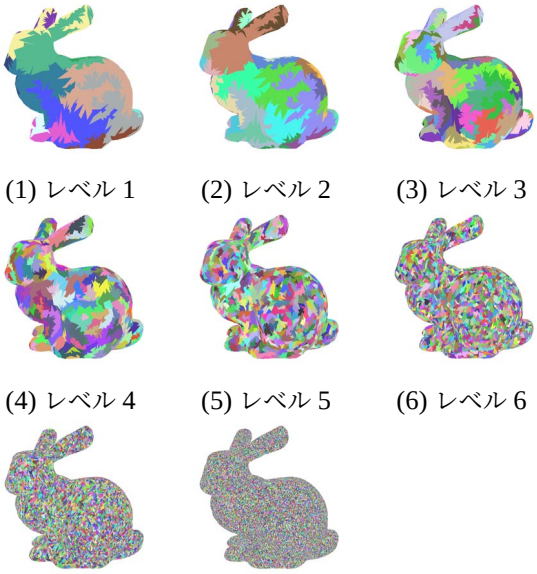


図 12: うさぎモデルの階層構造

リに展開しておく元モデルの方が高速であった。しかし、それでも一回の描画に 4 秒もかかるので、対話的な操作ができるとは言い難い。階層化データの一番簡略化したモデル (図 12 (1)) であれば一瞬で表示できるので、膨大なデータを扱う場合、対話的な操作には階層的なデータの方が向いていると言える。

メモリの使用量は、元モデルでは 64,992KB、階層化データでは 2,840KB であった。従来の手法のようにメモリ上に展開する場合はモデルの詳細度に応じたメモリ量が必要になり、膨大な面数を扱う場合は現実的ではない。一方、本手法を適用した階層化データはデータをファイルから逐次読み込んでいるため、詳細度に関わらずメモリの使用量が少なくて済むという利点が見られた。

図 13は、 20 万面の仏像のモデルに適用した例である。このモデルは面数が多すぎて通常の手法で表示するのは

	初期化時間 [sec]	詳細形状 描画時間 [sec]
元モデル	約 200 秒	4
階層化データ	11	15

表 1: 描画速度の比較

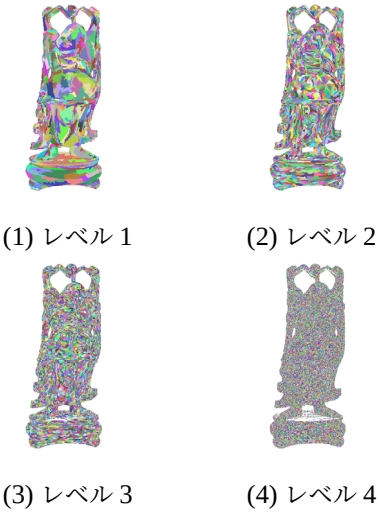


図 13: 仏像モデルの階層構造

不可能であったが、本手法を用いたシステムでは小型の計算機で表示する事が可能であった。

10 結論

本研究では、評価関数を利用したグループ化手法を提案した。 Culling に関しては、生成されるグループの cone を評価する関数と、 cone の上限を閾値で設定する方法が有効であることを確認した。また、生成されたグループを階層的にし、段階的に簡略化をすることで高速かつ必要メモリ量が少なくなる手法を提案した。さらに、実験によって描画時間の測定を行い、手法の有効性を実証した。

参考文献

[1] Kenny Hoff. Backface cluster culling using normal-space partitioning. Technical report, University of North Carolina at Chapel Hill, August 1996. http://www.cs.edu/˜hoff/techrep/quickbfc.html.

[2] Michael Garland and Paul S.Heckbart. Surface simplification using quadric error metrics. *In Computer Graphics(Proc. SIGGRAPH 97)*, pp. 209–216, 1997.