芸術科学会論文誌 Vol. 10, No. 4, pp. 251 – 262

リアルタイム3DCGにおける物体の形状を考慮した輪郭線の誇張表現手法の提案

松尾 隆志 † 三上 浩司 ‡ 渡辺 大地 ‡ 近藤 邦雄 ‡

東京工科大学大学院 バイオ・情報メディア研究科 メディアサイエンス専攻 †
東京工科大学 メディア学部 ‡

Shape Oriented Line Drawing in Real-Time 3DCG

Takashi Matsuo † Koji Mikami ‡ Taichi Watanabe ‡ Kunio Kondo ‡

Graduate School of Bionics, Computer and Media Sciences, Tokyo University of Technology †
School of Media Science, Tokyo University of Technology ‡

t_matsuo@mf.teu.ac.jp † {mikami, earth, kondo}@media.teu.ac.jp ‡

概要

2D アニメーションや漫画では，物体の形状をより効果的に表現するために，輪郭線の線質を変えるなどの誇張した表現で描くことがある．3DCG 上で 2D アニメーションや漫画の表現を行うトゥーンレンダリングでは，輪郭線は容易な手法である均一な線で表現することが多い．本研究は，ゲームなどのリアルタイムコンテンツ内でのトゥーンレンダリングにおける物体の形状を考慮した輪郭線の誇張表現手法を提案する．本手法では，2D アニメーションの作画に用いるデザインやデッサンの手法である輪郭線の強弱の変化による形状の誇張表現に着目した．提案手法では，対象となる 3 次元モデルのポリゴン形状が曲線を描く部分を特徴とし，その特徴に従って誇張表現を行った．また，モデルを構成するポリゴンをすべて裏返したモデルである，裏ポリゴンモデルを用いることで，形状に沿う連続した線の表現を行った．このことより，リアルタイム 3DCG でのトゥーンレンダリングにおいて形状を効果的に表現する輪郭線の誇張表現を実現した．

Abstract

In 2D animation and comics, the thickness of object outlines and contour lines are often varied to emphasize or exaggerate the shape of the object. Adding a 2D touch to 3DCG animation using toon-rendering is common, but existing techniques can not achieve varying line thickness. This paper proposes a method of recreating the effect of varying outline and contour line thickness based on object shape in real-time 3D environments, such as video games. Our method focuses on exaggerating thickness of lines where the target polygon is curved. Furthermore, by using an backface polygon of the target polygon, we are able to create a continuous outline of the object, thus recreating a closed outline of object with varying line thickness.

1 はじめに

近年，鉛筆画や水彩画などの手で描くような質感の画像を，3 次元コンピュータグラフィックス (以下「3DCG」) を用いて再現するノンフォトリアリスティックレンダリング (Non-Photorealistic Rendering:NPR) [1][2] の研究が盛んに行われている [3][4][5][6][7][8]．その中に，漫画や 2D アニメーション調の表現を行うトゥーンレンダリング (セルレンダリング)[9] という手法がある．トゥーンレンダリングは，数階調の陰影であるシェーディングと細い輪郭線で表現する手法である．特に日本では手描きアニメーションと 3DCG を組み合わせる際に，互いを調和する目的でトゥーンレンダリングを用いている [10]．本研究ではこのトゥーンレンダリングに用いる輪郭線に着目した．

本来，人は物体を見るとき，明度や彩度，色彩によって形を認識している．これに加え，元々物体にはない輪郭線を描くことで，より簡単に認識でき，意図的に形状や差を強調することができる．この輪郭線は漫画や 2D アニメーションに利用されている．特にディジタル 2D アニメーションでは，均一な輪郭線が多く用いられている．これは，紙に作画した線をコンピュータ上にスキャンするときに，コンピュータ上で均一な線として処理しているためである．また，3DCG 上でディジタル 2D アニメーション調の表現を行うトゥーンレンダリングでも，処理が最も単純である均一な線を多く用いている．

しかし，実際の漫画や 2D アニメーション作品では，イラストや水彩などの様々な線の表現を再現するために，線の強弱や濃淡などで輪郭を誇張表現することがある．この誇張する表現によって，輪郭や形状をより豊かに表現することが可能となる．ただし，こうした輪郭線表現を 2D アニメーション上で行うためには，フィルタワークや手作業によるレタッチなどの手間を加える必要がある．3DCG 上においても同様に，輪郭線の太さが変化するような処理や描画 [11][12] を行う必要がある．このような処理のためにプリレンダリングという手法を用いているが，計算や処理に時間がかかるという課題がある．

一方，リアルタイムレンダリングは，1 秒間に 30〜60 フレームの描画速度を維持するという制約の中で表現を行う手法である．リアルタイムレンダリングでは高速処理が必要であり，プリレンダリングと同様の手法や表現を行うことは難しい．このため，リアルタイムレンダリングでのトゥーンレンダリングの多くは，描画に影響が少なく容易な手法である，均一な線で表現することが多い．このために，プリレンダリングで行っている高品質な太さの変化がある輪郭線表現が求められている．

この課題を解決するために，本研究では，リアルタイム 3DCG でのトゥーンレンダリングにおいて物体の形状をより効果的に表現する輪郭線の誇張表現手法を提案を目的とする．本研究で描画目標とする輪郭線の誇張表現は，2D アニメーションにおいて作画する際に用いる，デッサンやデザインによる線の強弱の表現である．このために，ポリゴンモデルの裏ポリゴンを用いた輪郭線の抽出と描画，および線の強弱が出やすい，曲線となる特徴を利用した裏ポリゴンモデルの変形による線の太さ変化を描画する手法を提案する．さらに，提案手法を用いた評価実験を行い，提案手法の有用性を評価した．実験結果から，リアルタイム 3DCG でのトゥーンレンダリングで表現を行うこと，輪郭線の太さからの変化で 3D モデルの形状をより効果的に表現すること，連続した強弱の変化で手描きに近い輪郭線の表現を行うことができた．なお，本稿は第 26 回 NICOGRAPH 秋季大会で講演した内容を含む [13]．

以下，第 2 章では NPR 分野における輪郭線描画の従来手法，第 3 章でリアルタイム処理のための輪郭線誇張表現手法について述べる．そして第 4 章では提案手法による描画実験とその評価，特に輪郭線の誇張表現とリアルタイム性の評価について述べる．

2 従来手法

トゥーンレンダリングで用いるような 3 次元モデルを利用する輪郭線の誇張表現について次の 3 点に分けて述べる．

(1) 3 次元モデルを用いた輪郭線の描画

(2) 3 次元モデルを用いた変化のある輪郭線の描画

(3) NPR における絵画風の輪郭線描画

特に，従来手法の特徴と本研究で取り扱う課題について述べる．

(1) 3 次元モデルを用いた輪郭線の描画

3 次元モデルにおける輪郭線描画に関する研究は多くあり [14][15]，物体の形状をより明確に認識できるような輪郭線の表現 [16] もある．また，トゥーンレンダリングに関する研究の一部として，輪郭線の描画手法を提案する研究 [9][17] もある．ただし，これらの研究においての輪郭線の太さは，連続的な強弱の変化による表現ではなく，均一な線で表現している．しかし，実際の 2D アニメーションなどでは，様々な線の表現を再現するために，線の強弱や濃淡などの変化で輪郭を誇張表現することがある．

(2) 3 次元モデルを用いた変化のある輪郭線の描画

変化のある輪郭線表現を 2D アニメーション上で行うためには，手作業によるレタッチなどの手間を加える必要がある．3DCG 上でも同様に，輪郭線の太さが変化するような処理や描画の手間を加える必要がある．「大神 [18]」のように処理を加えることで変化のある輪郭線の表現を行い，誇張した輪郭線の表現を行っているゲーム作品の例もある．ただし現状の手法は，輪郭線を無作為に変化する表現 [19] であり，対象となる物体を効果的には表現するものとはなっていない．このように，輪郭線の太さの変化を用いて 3D モデルの形状をより効果的に表現することが望まれている．

(3) NPR における絵画風の輪郭線描画

また，手で描いたような質感の画像を 3DCG で再現する NPR の分野では，リアルタイムに絵画調の表現をする研究があり，輪郭線を誇張する表現を実現している．しかし，トゥーンレンダリングにおいてそれらの表現を用いた場合，輪郭線が絵画調や絵画手法に則すもの [20][21][22] や，連続した線ではなく途切れた線の表現 [23] となる．このため，連続した輪郭線を作画し表現する，2D アニメーション調の表現とは異なるものとなる．したがって，連続した線の強弱変化で，手描きに近い輪郭線の表現を行うことが必要である．

以上の従来手法の成果と課題から，本研究では次の 3 項目を満たす輪郭線の誇張表現手法を提案する．

- リアルタイム 3DCG でのトゥーンレンダリングで輪郭線の表現を行うこと
- 輪郭線の太さの変化で 3D モデルの形状をより効果的に表現すること
- 連続した強弱の変化で，手描きに近い輪郭線の表現を行うこと

3 輪郭線の誇張表現手法

本研究では輪郭線の誇張表現に関し，2D アニメーションにおいて作画する際に用いる，デッサンやデザインによる線の強弱の表現に注目した．線の強弱による表現は，線の太さを変えるというシンプルな手法であり，描き手のストロークや筆圧が直接線として出る手法である [24]．また，曲線を描く形状や奥行きを表現する際に用い，丸みや立体感，強調といった形状を誇張する効果を与える [25]．この線の強弱による表現は，形状を効果的に表現する手描きならではの手法である．

本章では，このような線の強弱による表現をリアルタイム 3DCG 上で行う手順を解説する．

図 1 は，本手法の概略を示した図である．

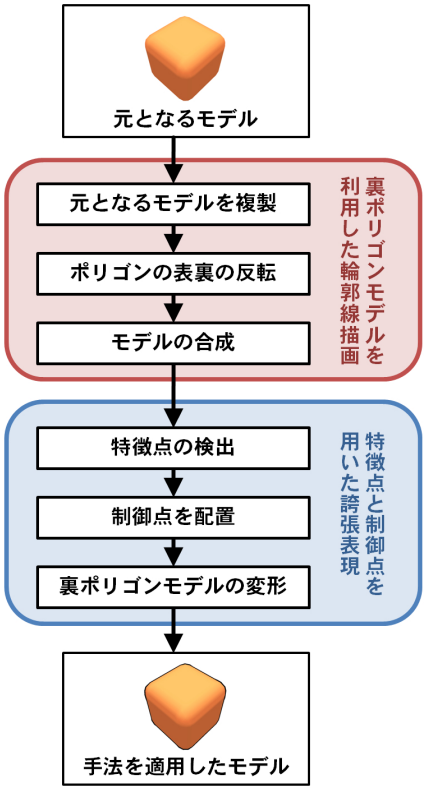


図 1 提案手法の流れ

本手法は次の 2 つのステップに大きく分かれている．

1. 裏ポリゴンモデルを利用した輪郭線の描画
2. 裏ポリゴンモデルの変形による輪郭線の誇張表現

まず，裏ポリゴンで構成するモデルである，裏ポリゴンモデルを利用した輪郭線の描画手法について，3.1 節で述べる．次に，モデルの特徴である特徴点と変形を制御する制御点を用いた，裏ポリゴンモデルの変形による輪郭線の誇張表現について，3.2 節で述べる．

また，3.3 節では本手法を利用するにあたり，形状を効果的に誇張表現するための調整方法について述べる．

3.1 裏ポリゴンモデルを利用した輪郭線描画

リアルタイム 3DCG にて輪郭線の誇張表現を行う本研究では，次の 2 点を考慮することとした．

- 形状を効果的に表現するための形状に沿う正確な輪郭線

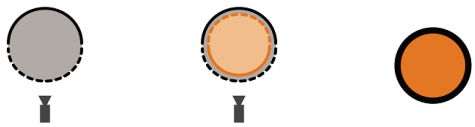
- リアルタイムレンダリングを維持可能な実行速度

これらを踏まえ本研究では，3D モデルを複製した後に引き伸ばし，ポリゴンの表裏を反転する手法を用いた．その概略として鈴木らの研究 [26] における手法を次に述べる．

1. モデルの複製
   元となる 3D モデルを複製した後，頂点の法線方向に拡大し，面の色を黒くする．頂点の法線は，その頂点を構成要素に持つポリゴンの法線ベクトルの平均とする．
2. ポリゴンの表裏の反転
   拡大したモデルのポリゴンの表裏を反転し，視点から見て手前になるモデルの面は表示せず，奥となる面を表示するようにする．
3. モデルの合成
   拡大し，ポリゴンの表裏を反転したモデルに，元となるモデルを重ね合わせる．

図 2 は球体のモデルを利用して，3D モデルを拡大し裏ポリゴンにして輪郭線にする処理を図示したものである．

(a) 基本のモデル　(b) 上空から見た断面　(c) 手順 1



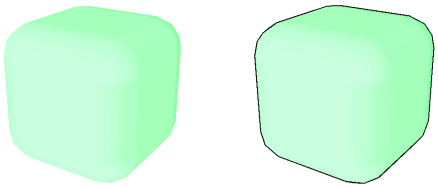
(d) 手順 2　(e) 手順 3　(f) 最終的なモデル

図 2 輪郭線描画の過程

図 2(a) と図 2(f) は対象となる球体モデルを正面から見た図である．図 2(b)〜図 2(e) は，モデルを上空から見下ろした時の断面図であり，実際の視点は図の下側にあるものとする．また，実線部は視点から見て表示している面，点線部は表示していない面を表す．

図 2(b) は元のモデルを上空から見た時の断面図であり，図 2(c) は複製して拡大し，面を黒く塗ったモデル，図 2(d) はポリゴンの表裏を反転したモデル，図 2(e) は元々のモデルを重ね合わせた図となっている．この処理を用いた結果，元のモデルからはみ出した部分が表示される．図 2(f) が実際の視点から見た図であり，球体のモデルに輪郭線を描いたように見える．

図 3 は，以上の輪郭線の描画処理を 3D モデルに適用した結果を示した図である．



(a) 元となるモデル　(b) 輪郭線の描画後

図 3 裏ポリゴンを利用した輪郭線

元のモデルである図 3(a) に，処理を行った結果が図 3(b) であり，十分に輪郭線が描画できていることが分かる．

この手法の特徴として，単純な処理で均一な輪郭線が描画できる点と，輪郭線となるモデル（以下「裏ポリゴンモデル」）を変形することで輪郭線を容易に変化できる点がある．また，単純に同じモデルを 2 つ重ねて用いるため，形状に沿った正確な輪郭線が描画でき，かつ処理速度も維持できる．本研究ではこの手法を用い，リアルタイム 3DCG 上で形状を考慮した輪郭線の誇張表現手法を提案する．

3.2 特徴点と制御点を用いた誇張表現

3.2.1 特徴点の検出

本研究では線の強弱が出やすい，曲線となる部分を特徴とした．3DCG 上で曲線となる部分は，3D モデルの凹形部分と凸形部分である．図 4 はその例を示したものであり，青い丸で囲んだ部分がモデルの凹形部分，赤い丸で囲んだ部分がモデルの凸形部分である．

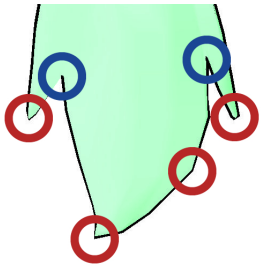


図 4 モデルの凹凸部分

図 4 の青い丸，赤い丸で示す箇所のような形状を検出するために，地神らの研究 [27] における，各頂

点の法線を利用した 3D モデルの凹凸部分を検出する手法を拡張して用いる．

図 5 は，3 次元空間上でポリゴンに接続している頂点とその法線ベクトルを表したものである．

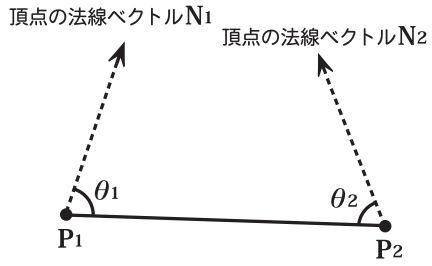


図 5 接続している頂点とその法線ベクトル

モデル上のある頂点座標を $\mathbf{P_1}$ とし，その頂点に隣接し接続する頂点座標を $\mathbf{P_2}$ とする．また，この頂点の法線ベクトルをそれぞれ $\mathbf{N_1}$，$\mathbf{N_2}$ とする．頂点の法線ベクトルとは，その頂点を構成要素に持つポリゴン面の法線ベクトルの平均とする．また，線分 $\mathbf{P}_1\mathbf{P}_2$ と $\mathbf{N_1}$ がなす角度を θ_1，線分 $\mathbf{P}_2\mathbf{P}_1$ と $\mathbf{N_2}$ がなす角度を θ_2 とする．

θ_1 と θ_2 が式 (1) を満たす場合，凹形状のポリゴンの接続である．

$$0^\circ \leq \theta_1 + \theta_2 < 180^\circ \tag{1}$$

図 6 は，頂点の法線ベクトルが向かい合っている，凹形状のポリゴンの接続を示した図である．

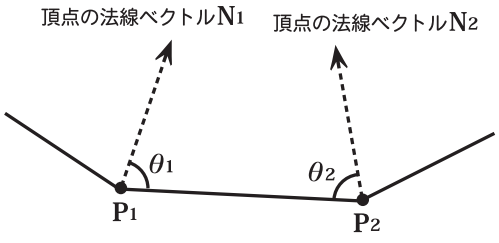


図 6 凹形状のポリゴンの接続

この場合，θ_1 と θ_2 の合計が小さいほど，頂点の法線ベクトルがより向かい合っており，より凹形の状態である．全ての頂点において，各頂点の法線ベクトルとその頂点に隣接する頂点の法線ベクトルを比較し，式 (2) を満たす任意の角度 A 以下である，隣接する頂点の数を求める．

$$A > \theta_1 + \theta_2 \tag{2}$$

一方，θ_1 と θ_2 が式 (3) を満たす場合，凸形状のポリゴンの接続となる．

$$180^\circ \leq \theta_1 + \theta_2 < 360^\circ \tag{3}$$

図 7 は，頂点の法線ベクトルが反対を向いている，凸形状のポリゴンの接続を示した図である．

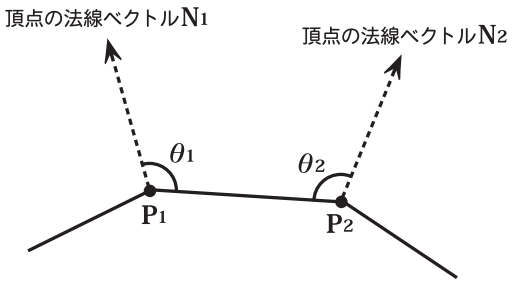


図 7 凸形状のポリゴンの接続

この場合，θ_1 と θ_2 の合計が大きいほど，頂点の法線ベクトルがより反対を向いており，より凸形の状態である．こちらも同様に，全ての頂点において各頂点の法線ベクトルと隣接する頂点の法線ベクトルを比較し，式 (4) を満たす任意の角度 B 以上である，隣接する頂点の数を求める．

$$B < \theta_1 + \theta_2 \tag{4}$$

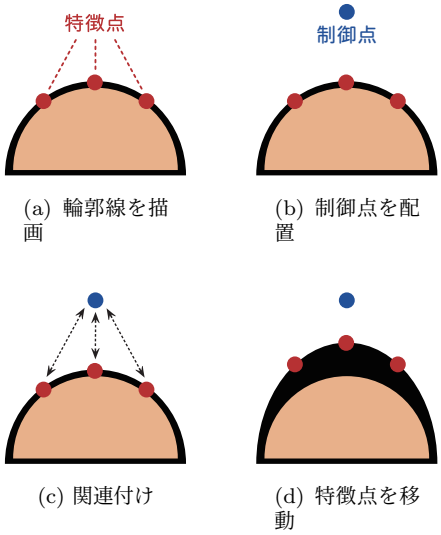
各頂点において，式 (2) を満たす隣接する頂点の数が任意の値よりも多い場合，図 4 の青い丸で囲んだ部分であるとして，その頂点の情報を凹形状の特徴部分として保存しておく．同様に，式 (4) を満たす隣接する頂点の数が任意の値よりも多い場合，図 4 の赤い丸で囲んだ部分であるとして，その頂点の情報を凸形状の特徴部分として保存しておく．以降，検出した凹凸形状の特徴部分である頂点を「特徴点」と呼ぶこととする．

またこのとき，隣接する頂点の数と比較する任意の値（以下「比較値」）により，特徴点となる頂点の検出に変化が生じる．仮に，比較値を大きく設定した場合，隣接する頂点の多くに凹凸形状の特徴がある場合に特徴点となり，より凹凸の変化が顕著な部分が特徴点となる．反対に，比較値を小さく設定した場合，隣接する頂点のいずれかに凹凸形状の特徴がある場合に特徴点となり，凹凸の変化が少なくとも特徴点となる．

よって，より凹凸の変化が大きい箇所のみを特徴点とする場合は，比較値を大きく設定し，全体的に凹凸の変化がある箇所を特徴点とする場合は，比較値を小さく設定する．

3.2.2 制御点を用いた裏ポリゴンモデルの変形

3.2.1 節で検出した特徴点を用いて，裏ポリゴンモデルに処理を行う．この裏ポリゴンモデルの変化によって，誇張した輪郭線の表現を行う．次の図 8 は裏ポリゴンモデルに行う処理の過程を示した図である．なお，図中の赤い点は特徴点であり，青い点は制御点である．



(a) 輪郭線を描画　(b) 制御点を配置

(c) 関連付け　(d) 特徴点を移動

図 8 制御点を利用した裏ポリゴンモデルの変形

図 8 で示す，特徴点と制御点を用いた裏ポリゴンモデルの変形処理を次の順に行う．

1. 輪郭線を描画

まず，3.1 節の手法を利用して，輪郭線となる裏ポリゴンモデルを描画する．また，3.2.1 節の手法を利用し，裏ポリゴンモデルの特徴点を検出する．対象のモデルの裏ポリゴンモデルの輪郭線を描画した状態が図 8(a) である．

2. 制御点を配置

次に，モデルの周囲に変形を制御するための点 (以下「制御点」) を配置する．図 8(b) は，対象のモデル付近に制御点を配置した図である．制御点は，3 次元空間上において裏ポリゴンモデルの特徴点を移動する方向に手動で配置する．この配置した制御点に対して特徴点となる頂点を移動することで，裏ポリゴンモデルを変形する．

3. 特徴点と制御点の関連付け

また，特徴点から最も近い座標にある制御点と特徴点を関連付ける．図 8(c) は，制御点と特徴点を関連付けを表した図である．頂点数が多く特徴点が多い場合や，配置する制御点が多い場合は，特徴点と制御点の関連付けが煩雑になる．このため，本手法では特徴点から最も近い制御点に対して関連付けることとした．この特徴点と制御点の関連付けを保存し，次の手順で利用する．

4. 特徴点を移動

最後に，特徴点と制御点の関連付けから，頂点の法線方向に特徴点を動かすことで輪郭線の誇張表現を行う．図 8(d) は，特徴点を移動することでモデルの変形を行った図である．このモデルの変形にあたり，特徴点となる m 個の頂点の位置ベクトルを $\mathbf{P}_i(i = 0, 1, 2, \ldots, m-1)$ とし，制御点となる n 個の点の位置ベクトルを $\mathbf{Q}_j(j = 0, 1, 2, \ldots, n-1)$ とする．このとき，特徴点の移動距離 $\mathbf{D}_{i,j}$ を次の式 (5) で求める．なお，t は任意の定数である．

$$\mathbf{D}_{i,j} = \frac{t}{(\mathbf{P}_i - \mathbf{Q}_j)^2} \tag{5}$$

特徴点 $\mathbf{P}_i$ を式 (5) で求めた $\mathbf{D}_{i,j}$ 分，法線方向に移動することで変形を行う．また，式 (5) は関連付けた制御点と特徴点間の距離 $\mathbf{P}_i - \mathbf{Q}_j$ の 2 乗による反比例式である．このため特徴点は，制御点との距離が近いほどより大きく制御点の方に移動し，距離が遠くなるほど移動距離は短くなる．これにより，制御点に近い特徴点部分を強調し，徐々に強調の度合いが小さくなるような滑らかな表現を行う．

以上の処理を行った結果を図 9 に示す．なお，図 9(b) と図 9(c) の右下の図は同一部分を拡大した図である．

(a) 元のモデル　(b) 均一な輪郭線のモデル

(c) 本手法を用いたモデル　(d) 制御点を表示した状態

図 9 本手法を用いた輪郭線の誇張表現

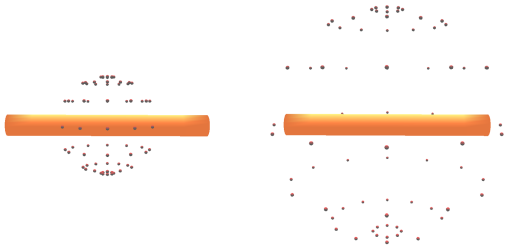
図 9(a) は元となるモデルであり，図 9(b) は均一な輪郭線を表示したモデルである．図 9(c) は本手法を用いた結果であり，図 9(d) は図 9(c) で用いた制御点を球体のモデルとして可視化したものである．なお，図 9(c) における制御点の配置は図 9(d) に示す通り，モデルの周囲を球状に囲むように配置した．

図 9(b) に示す均一な輪郭線を描画したモデルに対して，本手法を用いたモデルである図 9(c) の輪郭線には強弱がついていることが分かる．

3.3 モデルの形状に対応した制御点の調整

図 9(c) や図 9(d) で配置した制御点は，簡易的に一定の条件下で配置した．しかしこの場合，縦長や横長といった極端な形状の 3D モデルを使うと大きくはみ出すことがある．

図 10 は，一定の条件下で制御点を配置した場合に，モデルがはみ出す場合を図示したものである．図 10(a) は縦幅を，図 10(b) は，横幅を基準として配置した場合である．



(a) モデルがはみ出す例　(b) 大きく囲んだ例

図 10 制御点の配置例

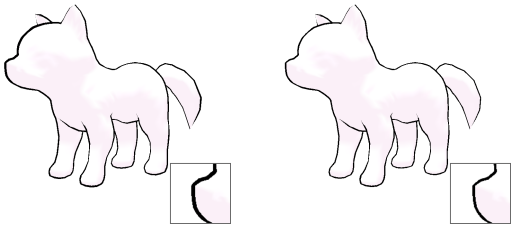
図 10(a) は，横幅が足りずにはみ出した状態となっていることが分かる．図 10(b) は，モデルの中央付近と制御点の距離が大幅に離れていることが分かる．

図 10(a) に示すように，モデルがはみ出すように制御点を配置した場合，制御点の内側にある特徴点も外側にある特徴点も変形する．しかし，特徴点が制御点の外側にある場合，制御点との距離が極端に近くなることもあり，予期しない変形が生じる可能性がある．このため，特徴点を意図的に変化できるよう，常に制御点の内側にモデルが存在するような制御点の配置を行う必要がある．

また，図 10(b) に示すように，3D モデルを大きく包み込むように制御点を配置した場合，特徴点から最も近い制御点が遠い位置にある可能性がある．3.2.2 節の式 (5) で示した通り，特徴点を動かす要素に特徴点と制御点間の距離がある．大きく球状に包み込んだ場合，特徴点と制御点間の距離が増加するため，特徴点の移動量は減少し，輪郭線の変化は小さくなる．

なお，図中の輪郭線は変化をより明確とするために，式 (5) における裏ポリゴンモデルの移動量を通常よりも大きくした．

図 11 は特徴点と制御点間の距離による変化を比較した図である．



(a) 制御点を球状に配置　(b) 2 倍の距離に配置

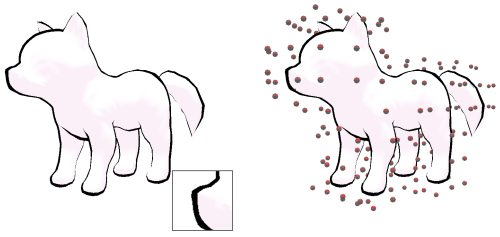
図 11 特徴点と制御点間の距離による変化

図 11(a) と図 11(b) はともに，モデルの周囲を球状に包むような図 9(d) と同様の制御点の配置をしたものである．ただし，図 11(b) の制御点は，図 11(a) での配置の 2 倍の距離に配置している．

通常の制御点の配置である図 11(a) は図 11(b) に比べ，裏ポリゴンモデルの変形が大きくなっていることが分かる．一方で，制御点を通常の 2 倍の距離に配置した図 11(b) の場合，図 11(a) の裏ポリゴンモデルの変形よりも，変化が小さい．

このように，制御点と特徴点の距離が遠くなる場合，本手法を効果的に利用できない．よりバランス良く効果的に表現するためには，モデルと制御点の距離が近すぎることがなく，また遠すぎることもない一定の距離であることが望ましい．よって，全体的にモデルと制御点の距離が一定となる配置である，モデルの形状に近い配置をすることで，より効果的な表現が可能となる．

図 12 は，モデルの形状に沿って制御点を配置した状態を図示したものである．



(a) 手動で配置した結果　(b) 制御点を表示した状態

図 12 形状に沿って制御点を配置した場合

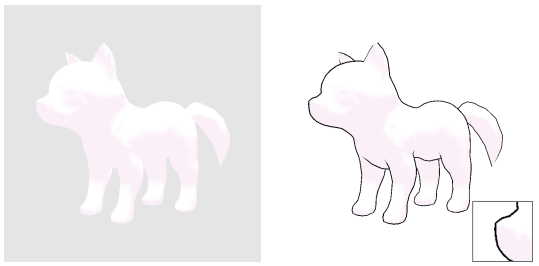
図 12(a) は形状に沿って制御点を配置した場合の結果であり，図 12(b) は図 12(a) で配置した制御点を可視化した図である．

制御点を球状に配置した図 9(c) は全体的に膨らんでいる様子に近いが，形状に沿って制御点を配置した場合の図 12(a) は，図 9(c) よりも線に強弱が出ている．このことより，制御点は球状に配置するなどの簡易的な配置よりも，形状に沿った配置の方がより効果が高いことが分かる．

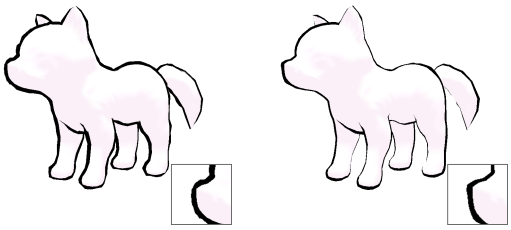
表 1 実行環境

OS	Windows 7 Enterprise 64bit
CPU	Intel Core2 Duo 3.16GHz
メモリ	4.00GB

図 13 と図 14 は複数の輪郭線表現を行ったモデルの画像である．図 13(a)，図 14(a) は輪郭線表現をしていないモデル，図 13(b)，図 14(b) が裏ポリゴンモデルを利用した輪郭線表現を行ったモデルである．図 13(c)，図 14(c) が無作為に揺らすことで誇張する既存の輪郭線表現をしたモデル，図 13(d)，図 14(d) が本手法を用いた輪郭線の誇張表現をしたモデルである．なお，既存の輪郭線表現は，無作為な変化による輪郭線の例 [19] を参考に誇張表現を行った結果を示す．



(a) 元のモデル　(b) 通常の輪郭線表現



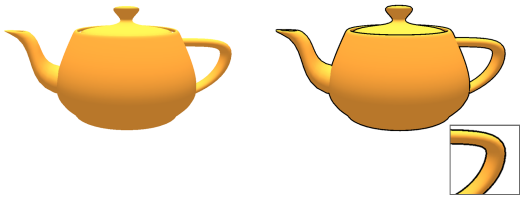
(c) 無作為な線での表現　(d) 本手法を用いた表現

図 13 輪郭線表現の比較 (1)

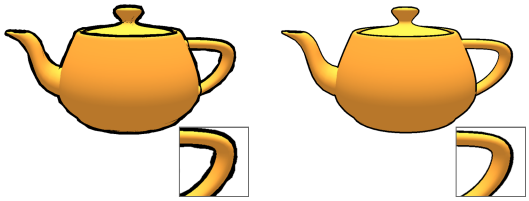
4 検証

4.1 本手法の効果を検証

本手法を用いた輪郭線の誇張表現について検証する．検証に使用した環境を次の表 1 に示す．



(a) 元のモデル　(b) 通常の輪郭線表現



(c) 無作為な線での表現　(d) 本手法を用いた表現

図 14 輪郭線表現の比較 (2)

図 13(b) や図 14(b) の均一な輪郭線表現に対し，図 13(d) や図 14(d) の本手法の輪郭線表現は，線に連続した強弱の変化が出ていることが分かる．

また，無作為に線を揺らす表現である図 13(c) や図 14(c) のような誇張表現は，形状を誇張するのではなく無作為に太さが変化した線で誇張している．このため，モデルの形状を効果的に表現できていない．これに対し本手法では，図 13(d) や図 14(d) に示すように，誇張した輪郭線によって，よりモデルの形状の凹凸を強調するように表現している．

以上のことより，本手法は既存の手法よりも輪郭線をより豊かに表現し，絵を描いた時のような輪郭線の表現をしているといえる．

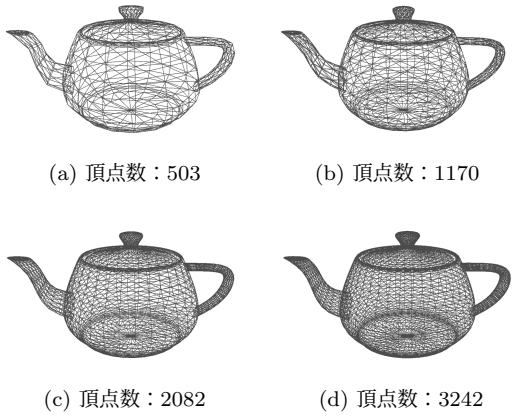
4.2 リアルタイム性の検証

本手法のリアルタイム性を検証する．検証には 4.1 節と同様に，表 1 の環境を用いた．

提案手法では，モデルの頂点を特徴点とし，制御点を用いて頂点を移動することで輪郭線の誇張表現を行っている．このため，頂点数の違う複数のモデルを用意し，次の 3 手法において比較し，検証を行うこととした．

1. 3.1 節での均一な輪郭線の描画手法
2. 事前計算の 1 回のみで輪郭線誇張のための裏ポリゴンモデルの変形処理を行う手法
3. 毎フレームごとに輪郭線誇張のための裏ポリゴンモデルの変形処理を行う手法

また検証用のモデルとして，頂点数が異なる同形状のモデルを用意した．検証に利用したモデルをワイヤーフレーム化したものを次の図 15 に示す．



(a) 頂点数：503　(b) 頂点数：1170

(c) 頂点数：2082　(d) 頂点数：3242

図 15 頂点数が違う同じ形状のモデル

それぞれのモデルの頂点数は，図 15(a) は 503 個，図 15(b) は 1170 個，図 15(c) は 2082 個，図 15(d) は 3242 個である．

4.2.1 事前計算による誇張表現

事前計算による誇張表現は，事前に本手法を 1 回適用し，以後本手法を適用せずに描画する方法である．この場合，本手法を適用するのは事前計算の 1 回のみのため，それ以後に新たな特徴点が生じたり，制御点を移動する場合でも，輪郭線の太さが変わることはない．

表 2 は，通常の均一な輪郭線表現と事前計算による輪郭線の誇張表現の 1 秒間当たりの描画回数を表したものである．なお，描画速度の単位は fps(Frames Per Second) である．

表 2 描画速度の測定結果

頂点数	描画速度 (fps)	
	均一な輪郭線	1 回のみの変形
530	653.2	647.0
1170	351.4	340.8
2082	197.8	193.0
3242	74.0	72.6

事前計算による変形の場合，事前に変形処理を行い，以降は変形処理を行わない．このため，描画の際には均一な輪郭線表現と同様の状態であり，通常の輪郭線処理と同様の描画速度で異なる表現が可能である．

4.2.2 毎フレームごとの計算による誇張表現

ゲームなどのコンテンツでは，モーションの変化などによってモデル形状が変化することがある．モデルの形状が変化した場合，特徴点となる頂点が変化したり，特徴点と制御点の位置関係が変化する可能性があり，輪郭線の誇張表現を効果的に表現できない．このため本研究では，それらの形状変化を想定して毎フレームごとに形状変化する場合の検証を行った．

表 3 は，事前計算による輪郭線の誇張表現と毎フレームごとの計算による輪郭線の誇張表現の 1 秒間当たりの描画回数を表したものである．なお，描画速度の単位は表 2 と同様に fps(Frames Per Second) である．

表 3 描画速度の測定結果

	描画速度 (fps)	
頂点数	1 回のみの変形	毎フレームの変形
530	647.0	133.0
1170	340.8	54.2
2082	193.0	28.8
3242	72.6	8.0

毎フレームごとの計算による変形の場合，事前計算による変形よりも描画速度は低下する．しかし，表 3 に示すように，頂点数が 2000 個程度までであれば，30fps 程度の描画速度を維持できている．このため，頂点数が一定の範囲内であれば，動的変形による表現もリアルタイムで実行できる．よって，アニメーションなどによって特徴点や制御点が変化する場合でも，本手法は適用可能であるといえる．

5 まとめ

本研究では，リアルタイム 3DCG におけるトゥーンレンダリングにて輪郭線の誇張表現を行うための手法を提案し，評価実験を行った．この結果，次のことが可能となった．

- 物体の形状を効果的に誇張する表現

イラストやデッサンに用いる，線の強弱による凹凸の表現に注目し，曲線となる部分を特徴として検出，誇張することにより，対象とする物体の形状に対して，より効果的に表現することが可能となった．

- リアルタイムで実行できる処理速度

モデルの形状に沿っており輪郭線が正確であり，かつ処理が単純である裏ポリゴンモデルを利用する輪郭表現手法に着目した．これにより，無作為に線を動かさない連続した線の強弱による輪郭線の誇張表現が，リアルタイム 3DCG でのトゥーンレンダリングで実現でき，手描きに近い輪郭線の表現が可能となった．

今後の課題は，次の通りである．

- 意図しない変形について

本手法では，モデルの頂点の構成から特徴を検出するため，形状によっては意図しない部分を特徴とする可能性がある．このため，今後はモデルの特徴検出をより正確に行えるよう，モデルの曲率から特徴を検出する手法などの新たな手法を検討する必要がある．

- 毎フレームごとの計算による誇張表現でのリアルタイム性について

毎フレームごとの計算による誇張表現に関して，現状では頂点数によるリアルタイム性の制約がある．これに対し，グラフィクスハードウェアである GPU(Graphics Processing Unit) を用いることで，より多くの頂点を処理できるようにすることが望まれる．

- 制御点の配置について

現状の手法で最適な結果を得るためには，制御点を配置する座標を手動で指定していく必要があり，煩雑な作業である．これに対し，自動的にモデルの周囲に制御点を配置するなどのより容易に手法を適用できる手法を検討する必要がある．

このような課題を解決することで，ゲームなどのコンテンツで用いることができるようにしたい．また，イラストやデッサンで用いる光源や奥行きによる輪郭線の太さの変化にも対応することで，より手描きに近いトゥーンレンダリングの表現として高めていきたい．

参考文献

[1] Bruce Gooch, Amy Gooch,「Non-Photorealistic Rendering」, AK Peters Ltd, 2001.

[2] Thomas Strothotte, Stefan Schlechtweg,「Non-Photorealistic Computer Graphics: Modeling, Rendering, and Animation」, Morgan Kaufmann, 2002.

[3] Mario Costa Sousa, John W. Buchanan, “Computer-Generated Graphite Pencil Rendering of 3D Polygonal Models”, Computer Graphics Forum, Vol.18, pp.195-208, 1999.

[4] Praun E, Hoppe H, Webb M, Finkelstein A, “Real-time hatching”, Proc. SIGGRAPH 2001, pp.581-586, 2001.

[5] Thomas Luft, Oliver Deussen, “Interactive Watercolor Animations”, Proc. PacificGraphics 2005, pp.7-5, 2005.

[6] Thomas Luft, Oliver Deussen, “Real-Time Watercolor Illustrations of Plants Using a Blurred Depth Test”, Proc. NPAR 2006, pp.11-22, 2006.

[7] Gooch Amy, Gooch Bruce, Shirley Peter, Cohen Elaine, “A non-photorealistic lighting model for automatic technical illustration”, Proc. SIGGRAPH 1998, pp.447-452, 1998.

[8] Aaron Hertzmann, “Non-Photorealistic Rendering and the Science of Art”, Proc. NPAR 2010, pp.147-157, 2010.

[9] Philippe Decaudin, “Cartoon-looking rendering of 3D-scenes”, Research Report, 1996.

[10] デジタルアニメ制作技術研究会, 東京工科大学, 「プロフェッショナルのためのデジタルアニメマニュアル 2009 ～工程・知識・用語～」, デジタルアニメ制作技術研究会, pp.49-74, 2009.

[11] Todd Goodwin, Ian Vollick, Aaron Hertzmann,“Isophote Distance: A Shading Approach to Artistic Stroke Thickness”, Proc. NPAR 2007, pp.53-62, 2007.

[12] Pierre Benard, Forrester Cole, Aleksey Golovinskiy, Adam Finkelstein, “Self-Similar Texture for Coherent Line Stylization”, Proc. NPAR 2010, pp.91-97, 2010.

[13] 松尾隆志, 三上浩司, 渡辺大地, 近藤邦雄, “リアルタイム 3DCG における物体の形状を考慮した誇張表現手法の提案”, 第 26 回 NICOGRAPH 秋季大会 論文集, II-2, 2010.

[14] Takafumi Saito, Tokiichiro Takahashi, “Comprehensible Rendering of 3-D Shapes”, Proc. SIGGRAPH 1990, pp.197-206, 1990.

[15] 望月義典, 近藤邦雄, “再分割曲面の輪郭線抽出描画手法”, 情報処理学会論文誌, Vol.41, No.3, pp.601-607, 2000.

[16] 望月義典, 近藤邦雄, “形状特徴表現のためのエッジ強調描画手法”, 情報処理学会論文誌, Vol.40, No.3, pp.1148-1155, 1999.

[17] 金子満, “次世代アニメーション制作システムに関する研究”, 学位論文, 東京工業大学, 1996.

[18] “大神”, CAPCOM CO.,LED., カプコン, 2006, 2008.

[19] Works Corporation, 「アミューズメント映像探検隊 69 大神」, CGWORLD (2006 年 6 月号), Vol.94, pp.85-89, 2006.

[20] H Lee, S Kwon, S Lee, “Real-time pencil rendering”, Proc. NPAR 2006, pp.37-45, 2006.

[21] 村上恭子, 鶴野玲治, “顔料及び支持体の特性を考慮したパステル画風レンダリング”, 芸術科学会論文誌, Vol.1, No.2, pp.89-96, 2002.

[22] 川嵜敬二, 中丸幸治, 大野義夫, “NPR におけるストローク方向の決定と水墨画調レンダリングへの適用”, 芸術科学会論文誌, Vol.3, No.4, pp.235-243, 2004.

[23] Robert D. Kalnins, Lee Markosian, Barbara J. Meier, Michael A. Kowalski, Joseph C. Lee, Philip L. Davidson, Matthew Webb, John F. Hughes, Adam Finkelstein, “WYSIWYG: NPR Drawing Strokes Directly on 3D Models”, Proc. SIGGRAPH 2002, pp.755-762, 2002.

[24] AL Guptill, SE Meyer, 「鉛筆で描く」, マール社, pp.33-43, 1977.

[25] AL Guptill, SE Meyer, 「ペンで描く」, マール社, pp.48-56, 1976.

[26] 鈴木隼人, 渡辺大地, “リアルタイム 3DCG における米国漫画調レンダリングの開発”, 情報処理学会 第 66 回 全国大会 講演論文集, 4N-9, pp.185-186, 2004.

[27] 地神知哉, 渡辺大地, “簡略化表現を用いた、アニメ調 3DCG の視認性向上手法”, 第 7 回 情報科学技術フォーラム 講演論文集, I-054, pp.307-308, 2008.

松尾 隆志

2010 年東京工科大学卒業．現在は東京工科大学大学院バイオ・情報メディア研究科メディアサイエンス専攻修士課程在籍．コンピュータグラフィックスに関する研究に従事．ACM SIGGRAPH，芸術科学会会員．

三上 浩司

1995 年慶應義塾大学環境情報学部卒業，博士（政策・メディア：2008 年慶応義塾大学）．1999 年より東京工科大学片柳研究所クリエイティブ・ラボに従事し，現在はメディア学部講師．主に 3DCG を利用したアニメ，ゲームの制作技術と管理手法に関する研究開発に従事．ACM SIGGRAPH，芸術科学会，情報処理学会，日本デジタルゲーム学会ほか所属．

渡辺 大地

1994 年慶應義塾大学環境情報学部卒業．1996 年慶應義塾大学政策・メディア研究科修士課程修了．修士 (政策・メディア)．1999 年より東京工科大学メディア学部講師．コンピュータグラフィックスやゲーム制作に関する研究に従事．情報処理学会，芸術科学会会員．

近藤 邦雄

名古屋工業大学卒業，工学博士 (東京大学)，名古屋大学，東京工芸大学，埼玉大学工学部情報システム工学科を経て，現在は東京工科大学メディア学部教授．情報処理学会グラフィクスと CAD 研究会主査，画像電子学会副会長，ヴィジュアルコンピューティング研究委員会委員長，日本図学会副会長，図学教育研究会委員長など歴任．現在，芸術科学会会長．