情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2011-CG-142 No.18
2011/2/9

制御点を用いたメッシュ変形手法

工 藤 成 司[†1]　今 井 桂 子[†2]

本研究では，3 次元表面メッシュデータに対する対話的変形アルゴリズムを提案する．メッシュの形状を対話的に変形させる手法は，3 次元コンピュータグラフィックスのモデリングやアニメーション作成において用いられている．近年，スケッチや制御点などを用いた，直感的なメッシュ変形操作を行なう手法が注目されている．制御点を用いた変形手法の一つに McDonnell らの手法がある[1)]．その手法は，ねじれを伴う変形を行なうことができていない．そこで，本研究ではその手法を基に，ユーザーが入力した回転軸を用いることによって，メッシュの一部に対してねじる変形を行なう手法を提案し，実装する．

Mesh Deformation Methods Using Control Points

Seiji Kudo[†1] and Keiko Imai[†2]

In this paper, we propose a new interactive twist deformation algorithm for 3D surface meshes. Interactive mesh deformations methods are used for modeling and animation in computer graphics. Recently, instinctive mesh deformation methods such as sketch-based and point-based attract attention. McDonnell et al.[1)] presented a point-based deformation methods. However, the Methods cannot perform twist deformation. Therefore, we add twist deformation to their method. We also show some experimental results.

†1 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
†2 中央大学 理工学部 情報工学科
Department of Information System and Engineering, Chuo University

1. 序　　論

メッシュとは三角形や四角形などの多角形を組み合わせることによって物体を表現したものである．メッシュは有限要素解析や 3 次元コンピュータグラフィックスなど様々な分野において応用されている．3 次元コンピュータグラフィックスのモデリングやアニメーション作成において，対話的にメッシュの形状を変形させる技術が用いられている．既存のモデルを変形することにより新たなモデルを作成することができたり，また，メッシュを変形したものを連続して表示することによりアニメーションを作成したりすることができる．本研究では，三角形表面メッシュに対する変形を扱う．

メッシュを変形する手法に関して様々な研究[2),3)] が行なわれてきた．近年，制御点やスケッチなどを用いた直感的にメッシュを変形する手法について多くの研究が行なわれている．制御点を用いた変形手法の一つに McDonnell らが提案した Point-Based Free-Form Deformation 法（PB-FFD）と呼ばれるものがある．この手法は，3 次元表面メッシュの周囲に複数の制御点を配置し，ユーザーが制御点を移動させることによってメッシュの形状を変形する手法である．PB-FFD 法はユーザーが容易な操作で変形を行なうことができ，きれいな変形結果を得ることができる．

しかし，PB-FFD 法はねじれを伴う変形を行なうことができていなかった．そこで，本研究では PB-FFD 法に基づいたねじる変形を行なうアルゴリズムを提案する．提案手法は，ユーザーが入力した回転軸と回転角を基にねじる変形を行なう．

2. 制御点を用いたメッシュ形状の変形

本節では，PB-FFD 法に基づく制御点を用いたメッシュ形状の変形アルゴリズムについて述べる．メッシュの周囲に配置される複数の制御点は，各々の影響領域を持つ．影響領域とは，その制御点を移動したとき，それに沿って移動するメッシュ頂点集合が含まれる空間である．また，すべてのメッシュ頂点が 1 つ以上の制御点の影響領域に含まれるように制御点は生成される．そして，メッシュ頂点はその点が含まれる影響領域毎に，その影響領域の制御点を原点とする局所座標系にパラメータ化される．このパラメータ値と，楕円放射基底関数を用いて算出した各制御点のメッシュ頂点に対する影響力，制御点の座標の 3 つの要素から成る式によって，メッシュ頂点座標を表現することが可能となる．ユーザーが制御点を移動させたとき，変更した制御点の座標を用いてその式を計算することにより，変形後のメッシュ頂点座標を求めることができる．以下では，メッシュ頂点のパラメータ化，制御点

の自動生成について詳細を述べる．

2.1 メッシュ頂点のパラメータ化

制御点 $\boldsymbol{p}_j = (p_{jx}, p_{jy}, p_{jz})$ に関する局所座標系の 3 つの軸を表す直交基底ベクトルを，$\boldsymbol{a}_j, \boldsymbol{b}_j, \boldsymbol{c}_j (\boldsymbol{a}_j = (a_{jx}, a_{jy}, a_{jz}), \boldsymbol{b}_j, \boldsymbol{c}_j$ も同様) とする．このとき M 個のメッシュ頂点のうちの一つ $\boldsymbol{v}_i$ の $\boldsymbol{p}_j$ に関するパラメータ $\boldsymbol{u}_{ij} = (\alpha_{ij}, \beta_{ij}, \gamma_{ij})$ は，局所座標系の上に頂点を射影することによって求めることができ，以下の式で求められる．

$$\boldsymbol{u}_{ij} = \left(\frac{(\boldsymbol{v}_i - \boldsymbol{p}_j)\cdot \boldsymbol{a}_j}{|\boldsymbol{a}_j|^2}, \frac{(\boldsymbol{v}_i - \boldsymbol{p}_j)\cdot \boldsymbol{b}_j}{|\boldsymbol{b}_j|^2}, \frac{(\boldsymbol{v}_i - \boldsymbol{p}_j)\cdot \boldsymbol{c}_j}{|\boldsymbol{c}_j|^2} \right) \tag{1}$$

また，メッシュ頂点に対する制御点 $\boldsymbol{p}_j$ の影響力は，次式の楕円放射基底関数によって計算される．

$$\phi_j(\boldsymbol{x}) = \frac{1}{\tau\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} (\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{R}\boldsymbol{S}^{-1}\boldsymbol{S}^{-1}\boldsymbol{R}^{\mathrm{T}} (\boldsymbol{x} - \boldsymbol{\mu}) \right) \tag{2}$$

$$\tau = \frac{\sqrt{2}}{2\sigma\sqrt{\pi}}$$

ここで，σ は標準偏差，行ベクトル $\boldsymbol{x} = [x\ y\ z\ 1]$ は同次座標における 3 次元座標，$\boldsymbol{\mu}$ は楕円の中心座標，τ は正規化因子，$\boldsymbol{S}$ は x, y, z 方向における楕円の倍率を特徴づける拡大縮小行列を表す．また，$\boldsymbol{R}$ は $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ を単位として楕円の方向を表現する回転行列である．

本研究では，すべての局所座標系の 3 つの軸を表す直交基底ベクトルをそれぞれ，$\boldsymbol{a}_j = (c, 0, 0), \boldsymbol{b}_j = (0, c, 0), \boldsymbol{c}_j = (0, 0, c)$（$c$: 局所座標系の軸の長さ）と定めた．よって，$\boldsymbol{S}$ と $\boldsymbol{R}$ は単位行列となる．したがって，式（2）は以下のように簡略化される．

$$\phi_j(\boldsymbol{x}) = \frac{1}{\tau\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2} (\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}} (\boldsymbol{x} - \boldsymbol{\mu}) \right)$$

標準偏差の値によって変形結果が大きく左右されることが本研究の実験より分かり，入力メッシュデータの座標の大きさによっては標準偏差の値を調整する必要がある．

今，メッシュ頂点 $\boldsymbol{v}_i = (v_{ix}, v_{iy}, v_{iz})$ の x 座標 v_{ix} を制御点に関して次のように定義することができる．

$$v_{ix} = \sum_{j=1}^{M} \widehat{\phi}_j(\boldsymbol{u}_{ij}) (p_{jx} + \boldsymbol{u}_{ij} \cdot (a_{jx}, b_{jx}, c_{jx})) \tag{3}$$

$$\widehat{\phi}_j(\boldsymbol{u}_{ij}) = \phi_j(\boldsymbol{u}_{ij}) / \sum_{k=1}^{M} \phi_k(\boldsymbol{u}_{ik})$$

v_{iy} と v_{iz} に対する式は同じように求めることができる．$\widehat{\phi}_j(\boldsymbol{u}_{ij})$ は頂点 i に影響を与えるすべての制御点の影響力の合計を 1 としたときの制御点 j の影響力を表す．

$\boldsymbol{u}_{ij}$ と $\widehat{\phi}_j(\boldsymbol{u}_{ij})$ は制御点が移動されたとき不変である．したがって，式（3）は次のように変形することができる．

$$v_{ix} = \sum_{j=1}^{M} \widehat{\phi}_j(\boldsymbol{u}_{ij}) p_{jx} + \sum_{j=1}^{M} \left[(\widehat{\phi}_j(\boldsymbol{u}_{ij}) \boldsymbol{u}_{ij}) \cdot (a_{jx}, b_{jx}, c_{jx}) \right] \tag{4}$$

v_{iy} と v_{iz} に対する式も同様に変形することができる．

制御点がユーザーによって移動させられたとき，その変化した座標を用いてすべてのメッシュ頂点に対して式（4）を再計算すること（v_{iy} と v_{iz} も同様）により，変形後のメッシュ頂点の座標を求めることができる．

2.2 制御点の自動配置

メッシュ頂点は制御点の移動によって間接的に移動され，その移動がメッシュの変形を引き起こす．制御点を自動的に配置し，PB-FFD に適した変形空間を構成するアルゴリズムについて説明を述べる．アルゴリズムの大まかな流れを以下に示し，次の段落でより詳細な説明を述べる．

(1) 入力メッシュの頂点から 3 次元ボロノイ図を計算する．
(2) メッシュの外側に位置するボロノイ点を削除する．
(3) ユーザーが指定した数になるまで，8 分木を用いて再帰的に点を削除する．
(4) 各制御点に対して 1 つの楕円放射基底関数を割り当てる．
(5) メッシュの頂点をコピーした点集合を作り，その点集合の点を頂点法線方向に少し移動させる．
(6) ユーザーが指定した数になるまで点を削除する．
(7) 各制御点に対して 1 つの楕円放射基底関数を割り当てる．
(8) 2 つの点集合の和集合をとり，それを最終的な制御点の集合と定義する．

2.2.1 制御点の自動配置アルゴリズム

アルゴリズムのステップ 1 – 4 は，メッシュの内側に位置する制御点（**図 1**）を生成している．メッシュの頂点から 3 次元ボロノイ図を計算し，メッシュの外側に位置するすべてのボロノイ点を削除する．その削除は kd 木を用いて次のように行なわれる．ボロノイ点 $\boldsymbol{q}$ が与えられたとき，その点に最も近いメッシュ頂点を $\boldsymbol{r}$ とし，$\boldsymbol{r}$ の法線ベクトルを $\boldsymbol{n}$ とす

Vol.2011-CG-142 No.18
2011/2/9

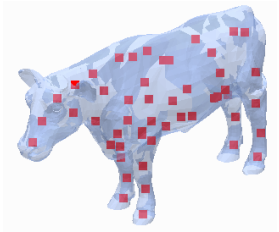

図 1 メッシュの内側の制御点
Fig. 1 control points inside mesh

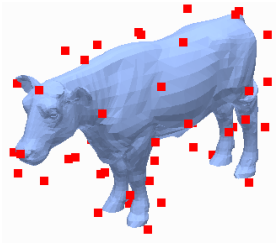

図 2 メッシュの外側の制御点
Fig. 2 control points outside mesh

る．また，$\boldsymbol{r}$ が隣接するメッシュの面集合を $F = \{F_1, F_2, \ldots\}$ とする．このとき，以下の 3 つの条件のうちどれか 1 つでもを満たすとき，$\boldsymbol{q}$ はメッシュの外側に位置すると判断し，削除する．

- $\boldsymbol{n} \cdot (\boldsymbol{q} - \boldsymbol{r})/|\boldsymbol{q} - \boldsymbol{r}| > 0$
- メッシュを囲む直方体の外側に $\boldsymbol{q}$ が位置する．
- ベクトル $\boldsymbol{q} - \boldsymbol{r}$ と F に含まれる面の面法線の外積が 0 より大きい．

次に，メッシュの内側に位置するボロノイ点を 8 分木を用いて再帰的に削除する．具体的には，各 8 分木のセルの内側で最も隣接する 2 つの頂点を求め，それらを削除し，削除された 2 つの頂点の中点に新たに点を配置する．これはセル内の頂点数がユーザーが指定した数になるまで行なわれる．制御点に割り当てられる球基底関数の半径は，各制御点とそれに最も隣接する制御点の距離を求め，その値に比例して（例えば 1.5 倍）定義する．

ステップ 5 – 8 では，メッシュの外側に位置する制御点（**図 2**）を生成している．メッシュ頂点をコピーした頂点集合を作成し，各頂点をその点法線ベクトルの比だけ移動させる（本稿の実験では 10%）．モデルの近くに制御点を配置することによって，ユーザーが急速に大規模局所変形を行なうことを容易にする．次に，点集合に含まれる点数がユーザーの指定した数になるまで制御点を再帰的に削除する．内側と外側の 2 つの制御点集合の和集合をとり，それを変形空間の制御点として定義する．そして，メッシュ頂点は制御点に関してパラメータ化される．

2.2.2 変形空間の不連続の回避

実際に変形を行なうとき，各メッシュ頂点は制御点の移動に応じて座標が移動する．しかし，どの制御点の影響領域にも含まれないメッシュ頂点が存在するとき，不連続な変形が生じてしまう．この問題を解決するために，マスター制御点と呼ばれる制御点を導入する．この制御点はすべてのメッシュ頂点を含むとても大きな影響領域を持つ．すべてのメッシュ頂点がマスター制御点に対してパラメータ化を行なわれることにより，不連続な変形を完全に防ぐことができる．

3. メッシュ形状のねじる変形

本節では，2 節で説明した既存手法に新たに加える，メッシュ形状の一部に対してねじる変形を行なうアルゴリズムについて述べる．ねじる変形を行なうアルゴリズムは，ユーザーが入力した回転軸と回転角を基に変形を行なう．アルゴリズムの大まかな流れを以下に示し，詳細について次の段落以降で説明する．

(1) ねじる変形を行なう場所の回転軸を決定する．
(2) 局所座標系を決定する．
(3) ねじる変形の影響領域を求める．
(4) 影響領域に含まれるメッシュ頂点の局所座標系に関するパラメータ化を行なう．
(5) 局所座標系において，回転角に基づいてメッシュ頂点を回転する．
(6) 局所座標系のパラメータを用いて世界座標系の座標を計算する．

3.1 ねじる変形を行なうアルゴリズム

ステップ 1 では，ねじる変形を行なうメッシュの一部に回転軸を決定する．回転軸はウィンドウ上からユーザーの入力によって与えられる．ただし，本研究ではメッシュの内側に回転軸が生成されることを前提としている．回転軸を表わすベクトルを $\boldsymbol{l}_a$，その両端点を p_s, p_t とする．

ステップ 2 では，回転軸を基に局所座標系を決定する．局所座標系の 3 つの軸を表わす直交基底ベクトル $\boldsymbol{A}_\alpha, \boldsymbol{A}_\beta, \boldsymbol{A}_\gamma$ のうち，$\boldsymbol{A}_\alpha$ は $\boldsymbol{l}_a$ を用いる．$\boldsymbol{A}_\beta$ は $\boldsymbol{A}_\alpha$ と任意のベクトル $\boldsymbol{t}$（例えば，$\boldsymbol{t} = (0.5, 0.5, 0.5)$）の外積 $\boldsymbol{A}_\alpha \times \boldsymbol{t}$ を計算して求められるベクトルである．同様に，$\boldsymbol{A}_\gamma$ は $\boldsymbol{A}_\alpha$ と $\boldsymbol{A}_\beta$ の外積 $\boldsymbol{A}_\alpha \times \boldsymbol{A}_\beta$ を計算して求めたベクトルとする．局所座標系の原点は，回転軸の中点とする．

スッテプ 3 では，ねじる変形の影響領域，すなわち変形するとき移動するメッシュ頂点を求める．回転軸上に複数の影響点と呼ばれる点を配置し，各影響点から一定距離 d 以内に位置するメッシュ頂点の点集合を求める．そして，それらの点集合の和集合をとり，それをねじる変形の影響領域に含まれるメッシュ頂点とする．回転軸の長さを d_l としたとき，影響点の個数 M_e は次の式で求める．

$$M_e = \left\lceil \frac{d_l}{d \times 0.8} \right\rceil - 1$$

また，回転軸上の影響点の位置は，p_s から $d_l/(M_e+1)$ の間隔で配置する．

ステップ 4 では，影響領域に含まれるメッシュ頂点の局所座標系に関するパラメータを求める．パラメータは式（1）を用いて計算する．ただし，$\boldsymbol{p}_j$ の座標として局所座標系の原点の座標を用いる．

スッテプ 5 では，ユーザーが入力した回転角に基づいて，メッシュ頂点を局所座標系において回転する．回転角を θ，頂点 v_i のパラメータを $\boldsymbol{Ut}_i = (\alpha t_i, \beta t_i, \gamma t_i)$ としたとき，頂点 v_i の回転角 θ_i は次式で計算する．

$$\theta_i = \theta \times (\alpha t_i + 0.5)$$

実際に回転した後の座標を計算するときは，同次座標系の回転行列を用いて計算する．

ステップ 6 では，ステップ 5 で求めた回転後の局所座標系の座標を用いて世界座標系の座標を計算する．世界座標は PB-FFD 法で求めるときと同様に，式（4）を用いて計算する．ただし，$M = 0, \widehat{\phi}_j(\boldsymbol{u}_{ij}) = 1$ と設定して計算を行なう．

4. 計算機実験

本節では，提案したアルゴリズムを実装し，計算機実験を行なった結果を述べる．本実験で入力として用いたメッシュは，cow2 モデルと square pillar モデルである．

2 つのモデル及び回転軸を**図 3** と**図 5** に示し，ねじる変形を行なった結果を**図 4** と**図 6** に示す．入力モデルに対して滑らかにねじる変形が行なえていることがわかる．しかし，**図 7** に見られるように，ユーザーにとってねじれて欲しい部分のメッシュ頂点（丸で囲んだ部分）が影響範囲に含まれていないといった問題が発生することがあった．この問題が発生する原因は，ねじる変形の影響領域の求め方にあると考えられる．したがって，影響領域の求める手法の改良を考える必要があるといえる．

5. 結　　論

本研究では，PB-FFD 法に基づくねじる変形アルゴリズムを提案した．また，計算機実験を行ない，滑らかなねじる変形結果を得ることができたことと，現状の問題点を確認することができた．

今後の課題としては，ねじる変形の影響領域の求め方の改良が考えられる．適切に影響領域を定めることができるようになれば，ユーザーの要求通りに変形を行なうことが可能になると考えられる．

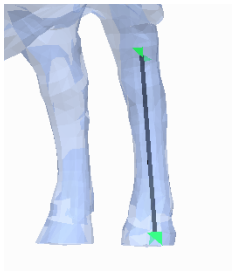

図 3 cow2 モデルに入力した回転軸
Fig. 3 input axis for cow2 model

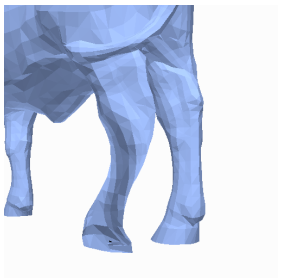

図 4 cow2 モデルを変形した結果
Fig. 4 result of deformed cow2 model

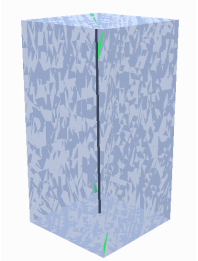

図 5 square pillar モデル
Fig. 5 square pillar model

図 6 square pillar モデルを変形した結果
Fig. 6 result of deformed square pillar model

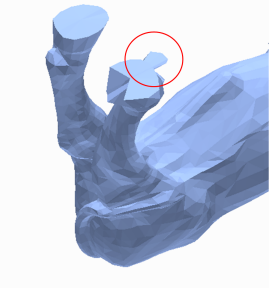

図 7 提案手法の問題点
Fig. 7 proposal method problem

参　考　文　献

1) McDonnell, K.T. and Qin, H.: PB-FFD: A Point-Based Technique for Free-Form Deformation, *journal of graphics, gpu, and game tools*, Vol. 12, No. 3, pp. 25–41 (2007).
2) Borrel, P. and Rappoport, A.: Simple constrained deformations for geometric modeling and interactive design, *ACM Transations on Graphics*, Vol.13, No.2, pp.137–155 (1994).
3) Moccozet, L. and Thalmann, N.M.: Dirichlet Free-Form Deformations and their Application to Hand Simulation, *Computer Animation*, pp.93–102 (1997).