論文

曲線上の最短点検出を利用した自由形状変形法

福山大学工学部　正会員　西田友是
〃　松田亮治
〃　高栄一寿

〈あらまし〉　CADシステムにおいてインタラクティブな曲線の修正変形などの処理は重要である．また，曲線の変形のみでなく，その曲線の変形を利用した形状処理も有用である．本稿では，スクリーン上において指定した任意の点と曲線との最短点を高速検出する方法を提案し，この検出法を応用した自由形状変形法を議論する．直線と頂点との距離は簡単に算出できるが，曲線の場合一般に解析的に解くのは困難である．n次曲線の場合，(2n−1)次式を解く必要があり，安定にかつ高速処理するには問題がある．本稿では曲線としてBezier曲線を用い，Bezier Clipping法を用いて，1次式の反復計算のみの効率のよい高速アルゴリズムを提案する．この検出法は，曲線上の任意の点のパラメータ値の算出(いわゆる逆問題)，複数曲線の中からの曲線選択，曲線のインタラクティブな変形などの広範囲に応用できる．本論文では，特に有用な利用法として，2次元図形の自由形状変形，モーフィングへの適用法を提案する．また，メタボールによって表現された人体モデルのインタラクティブな形状変形の例により，提案法の有効性を示す．

〈**Summary**〉　In geometric designs (or CAD systems), the interactive process of curves is important. This paper proposes a new calculation algorithm of the closest point on a curve to a point on a screen and its application method to deformation/morphing of images. Even though it is easy to evaluate the distance to a line segment, the distance to a curved segment is in general difficult. We have to solve degree (2n−1) of the polynomial to find the closest point on a degree n curve, so we have problems of the computational cost and robustness. We propose an effective algorithm for calculating the closest point on a Bezier curve by using the *Bezier Clipping Method* which is an iteration method using only linear equations.

The method proposed here can be applied to wide variety of cases such as point inversion for curves (finding parameter values at specified points on curves), selecting the closest curve within multiple curve, and the interactive deforming of curves. As for the applications, this paper also proposes an interactive deformation of curves/images and image morphing, and demonstrates the usefulness of the proposed method by using various examples such as a deformed human body modeled by metaballs.

1. まえがき

CADにおいてインタラクティブな曲線の修正変形などの処理は重要である．本稿では，スクリーン上において指定した任意の点と曲線との最短点を検出する方法(projection for curves)，およびそれを利用した自由形状変形法を提案する．図形の変形は，CADを始めエンターテインメント分野と広範囲に利用されている．2次元画像の変形の場合，画像上に制御用プリミティブとして線分やメッシュを重ね，これらを移動させることによって実現される．本稿では，制御用プリミティブとして曲線を使用する方法を提案するものである．

任意の点Qと曲線との最短点を検出する方法としては，曲線をいくつかに分割する方法(各サンプリング点

"Free Form Deformation by Detecting the Closest Point on a Curve" by Tomoyuki NISHITA (Member), Ryouji MATSUDA, and Kazuhisa TAKAE (Fukuyama University).

での距離を計算し，その中から最短となる点を抽出)，および数値解析法[1](曲線上のある点Pでの接線とベクトルQPとの直交条件から算出)がある．これらの方法は，計算時間を要すので，効率的な方法を提案する．なお，本稿では代表的なパラメトリック曲線であるBezier曲線を対象に考える．提案する検出法は次の特徴をもつ．

1)　Bezier Clipping法により，1次式でかつ少ない反復計算で解を求めることができる．したがって，リアルタイムの処理が可能でインタラクティブシステムに適用できる．

2)　距離を指定し，その範囲内での曲線上の最近点を効率よく算出できる．

3)　形状変形など，適用範囲が広い．

提案する最近点の検出法の応用としては，次のものが挙げられる．

1)　曲線上の任意の点のパラメータ値の算出(いわゆる逆問題：point inversion for curves)

2)　スクリーン上の複数曲線の中からの曲線選択

3)　曲線のインタラクティブな変形

4)　自由形状変形FFDやモーフィング(あるいはwarping)

5)　曲線に沿って移動する物体の衝突(干渉)問題

6)　点列データの近似曲線を得る際の距離誤差の評価

7)　オフセットカーブ(曲線からの距離が一定の曲線)の算出

以上のように多方面に応用できるパワフルなツールである．本論文では，1)から4)について議論する．

本論文では，まず基本的な考え方について述べ，最近点の検出方法，その応用例として，自由形状変形やモーフィングについて議論し，最後に人体モデルへの適用例を挙げて，提案法の有効性を示す．

2.　曲線上の最近点の検出法

2.1　従来法

任意の点と曲線との最短点を検出する一方法として，曲線をいくつかに分割し，各サンプリング点での距離を計算し，その中から最短となる点を抽出する方法がある．この方法では，パラメータ空間での精度が10^{-n}の場合(例えばn=3)，10^n個ものサンプリング点において，座標と距離を算出しなければならない(図1(a)参照)．

一方，数値解析法としては，パラメータuで表現される曲線C(u)と任意の点Qを考えると，曲線上のある点Pでの接線C'(u)とベクトルQPが直交するとき，距離が極小値をとることを利用する．すなわち，次式が成立する(図1(b)参照)．

$$C'(u)\cdot(C(u)-Q)=0. \qquad (1)$$

なお，上式は$(x(u)-x_q)\frac{dx(u)}{du}+(y(u)-y_q)\frac{dy(u)}{du}=0$に相当する．n次の曲線の場合，(2n−1)次式を解く必要がある．一般にはニュートン法で解かれる．しかし，ニュートン法の場合初期値が必要であり，また複数解が生じるので(極小値がいくつか存在)，どの解が最小値を与えるかの吟味が必要である．

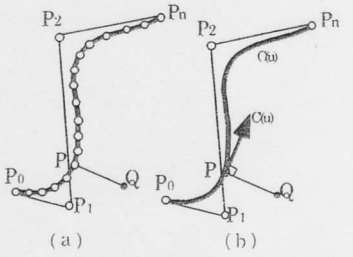


図1　最短点の検出法

Fig. 1　Finding the closest point on a curve

以上の点により，高速処理には従来法は十分とはいえない．

2.2　提案法

本論文では，式(1)を効率的に解く方法を提案する．ほとんどのパラメトリック曲線はBezier曲線に変換できるので，ここではn次Bezier曲線を対象とする．式(1)の左辺を関数gとし，次式を定義する．

$$g(u)=C'(u)\cdot(C(u)-Q) \qquad (2)$$

ここで，

$$C'(u)=\sum_{i=0}^{n-1}(P_{i+1}-P_i)B_i^{n-1}(u),$$

$$C(u)-Q=\sum_{i=0}^{n}(P_i-Q)B_i^n(u) \qquad (3)$$

また，$B_i^n(u)=\binom{n}{i}u^i(1-u)^{n-i}$はBernstein多項式である．

最近点を算出するには，g(u)=0を解く必要がある．3次曲線の場合，5次式を解くことになり，高速に安定して解を得るには問題である．高次の曲線の場合，この問題はより重要となる．提案法では，著者の一人が既に開発しているBezier Clipping法を用いることにより，1次式の反復計算のみで効率よくかつ安定に解を算出できる．このBezier Clipping法は，曲面のレイトレーシング[2]のために開発されたもので，Bezier曲線の凸包の性質を利用する方法である．

提案法は，解が存在する可能性のない区間を幾何学的性質を利用して抽出し，それらの区間の曲線を切り捨てることによって解の存在区間を収束させる方法で，わず

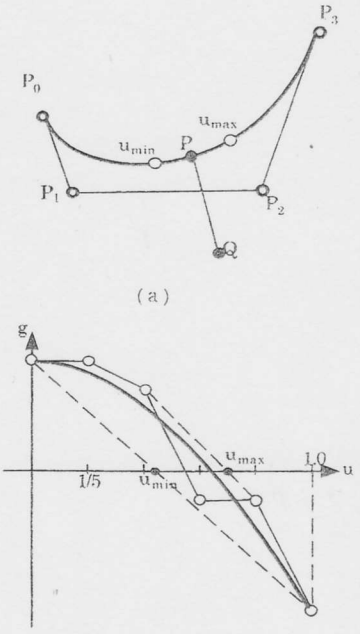


図 2　g(u)=0 を満たす区間の抽出(3 次曲線の場合)
Fig. 2　Extracting the interval satisfying g(u)=0 (in the case of cubic Bezier curve)

か数回(3 から 8)程度の反復で必要な精度の解が得られる．したがって，前述のサンプリング法に比較し 2 桁程度も計算量が少なくてよい方法である．

提案法は，式(2)の関数 g に対して単に Bezier Clipping 法を適用するのではなく，次節で述べるように，解の存在区間を算出するのには関数 g を用いるが，分割するのは曲線 C である．すなわち，解の存在範囲を g からまず抽出し，その範囲(区間)で曲線 C を分割し，次に C から関数 g(Bezier 関数で表現)に変換する．この処理を反復することにより解に収束する．

2.3　提案手法の処理手順

制御点 $P_i(x_i, y_i)(i=0, \cdots, n)$ で構成される n 次 Bezier 曲線 C(u) と任意の点 $Q(x_q, y_q)$ を考える．また，指定した距離 R_{max} 内での曲線上の最近点 P を考える．なお，曲線 C を微分したものはホドグラフと呼ばれるが，この曲線を C' とする．図 2(a)に，3 次曲線の例と任意の点 Q を示し，図 2(b)には関数 g(u) およびその制御点の凸包を示す．以下，図 3 を参照してアルゴリズムを説明する．

1)　点 Q が原点となるよう曲線 C を平行移動し(式 3 参照)，2 端点 P_0, P_n と Q との距離 d_0, d_n を算出する．これらと R_{max} との最小値を d_{min} とする．$d_{min}=\min(d_0, d_n, R_{max})$

2)　半径 d_{min} の円を囲むバンド(P_0P_n に垂直)により

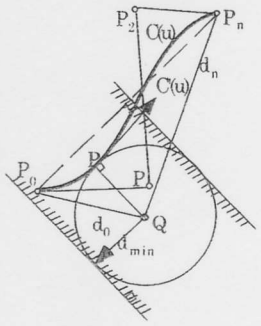


図 3　幅 d_{min} のバンドを用いた Bezier 曲線のクリップ
Fig. 3　Clipping the Bezier curve by using the band with width d_{min}

曲線 C をクリップする(図 3 参照)．

3)　曲線 C の制御点から C' を求め(式 3 参照)，C と C' から関数 g(式 2 参照)を求める．

4)　もし，g のすべての制御点が正またはすべてが負(すなわち異符号を保有しない)なら，この区間に解はないから，次の区間へ処理を進め，3)へ戻る．

5)　g(u)=0 を満たす区間を抽出(図 2(b)参照：範囲 $[u_{min}, u_{max}]$ を算出)する．

6)　もし，区間幅 $[u_{max}-u_{min}]$ があまり減少しないなら，曲線 C を 2 分割し，その一区間を処理中の区間とし，3)へ

7)　もし，$u_{max}-u_{min}>\varepsilon$(許容誤差)なら，この範囲以外をクリップし，3)へ(図 2(a)において区間 $[u_{min}, u_{max}]$ のみの曲線が抽出され，より直線に近くなる．)

8)　$u=(u_{min}+u_{max})/2$ を極小値とみなし距離 d を計算し，d_{min} と比較し小さい方を新たな d_{min} とする(その点の u, x, y も記憶)．他の区間があれば 3)へ戻る．

前述のように，上のアルゴリズムにおいて，関数 g(u) を直接解くのではなく，この g から解の有無，および解の範囲を抽出する．範囲が求まると，曲線 C を分割し関数 g に変換する．ここで，曲線の分割は de Castejou の方法を用いる．この分割は，曲線の次数の 2 乗に比例するから，関数 g より低次の曲線 C を分割する方法が有利である．

このアルゴリズムでは，反復計算が進むにしたがって，曲線が直線に近づくので，収束は加速度的に速くなる．ステップ 1)において 2 端点 P_0, P_n との距離を利用したのは，これらの点を必ず曲線が通過するから，点 Q との最短距離はこれらの距離より必ず短い性質があるためである．少しでも短い曲線を初期値として用いる方が収束が早いためこの前処理が有効である．ステップ 2)において，本来最短点は半径 R_{max} の円の内部のみを対象とすればいい．すなわち円の内部の曲線のみ必要であ

るが，これを簡易に算出するためこの円を包含するバンドで処理する。ここで，曲線をなるべく短く切り取るためには，P_0P_n に垂直なバンドが効率的である．ステップ6)において，関数gが複数解を持つ可能性がある場合(この場合区間幅が減少しない)，曲線を2分割してそれらの解を算出する．ステップ8)においては，gの複数解のうち，最近点を与える点を抽出するために距離を比較している．

3. インタラクティブな曲線の編集

一般に，曲線の変形(編集)は制御点を移動させる方法によって行われている．しかし，制御点を移動させる場合，実際の曲線がどの程度移動するか(または形状が変化するか)は予測が困難である．そこで，曲線上の指定した一点を直接移動する方法を考える．すなわち，スクリーン上においてマウスを用いてインタラクティブに曲線を変形する方法を提案する．曲線付近の点Qをクリックすると曲線上の最近点Pのパラメータuが判明する．この点Pをマウスの移動ベクトルにしたがって移動(マウスで指定した点Qのドラッグによる移動に同期してPが移動)，すなわち，変形後もマウス位置(すなわちQ)とPとの距離が一定である変形を考える．ここで，Qが曲線上にあるなら変形後もマウス位置が曲線を通過する．変形の方法としては，すべての制御点を移動させる，あるいはいくつかの選ばれた制御点を移動させる方法がある．また，有理曲線の場合は，重み係数を変化させることも考えられる．ここでは，点Pに最も近い制御点のみを移動する方式を採用する．

uの値からパラメータ空間で最も近い制御点を選択し，その制御点 P_k の移動ベクトル ΔP_k は，点Pでのマウスの移動ベクトル ΔP から次式で逆算できる(図4参照)．

$$\Delta P_k = \Delta P / B_k^n(u) \qquad (4)$$

なお，制御点 P_k のパラメータ値 u_k は k/n に相当する．例えば，3次曲線で，指定した点(最近点)でのuがu＝0.25の場合，P_1(u＝0.33に相当)が選択(uが近いので)

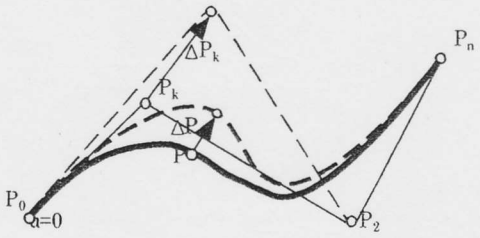


図4　曲線の変形

Fig. 4　Shape modification of a curve

され移動させる．以上の方法により，ユーザが望む位置付近を曲線が通過するように変形できる．また，指定した点を必ず通過するような曲線の変形も容易である．これは，指定した点Pの最近点を検出し，それらの距離差 ΔP から式(4)を利用して制御点 P_k を移動させることで実現できる．ここで，複数の指定した点を通過する曲線を厳密に得るには連立方程式を解く必要があるが，指定した1点を簡易に移動修正するには有効な方法である．以上のように，提案法はデザイナが感性に基づいて曲線を操作するには，直感的で非常に有効な方法である．

実際に，曲線の編集に適用する際には，複数の曲線を扱うようである．この場合，指定した点に最も近い曲線を選択し，かつそれを変形する処理が必要となる．

以下に複数の曲線のうちから最近の曲線を選択する方法について説明する．単に，各々の曲線に対する最短距離を算出し，その中から最小距離となる曲線を選ぶという方法ではなく，次のように最適化する．すなわち，最初の曲線との最短距離を算出したら，それを R_{max}(2.3節参照)として次の曲線に利用することによって，次の曲線の探索空間を狭くすることができる．これは提案法での，指定した距離内にある最近点を検出する機能を利用したものである．

4. Bezier曲線を用いた図形の変形

一般に図形を変形するには，その図形に制御用プリミティブである線分，メッシュあるいは包含箱を重ね，それを変形することによって目的の図形を変形する方法が用いられている．ここでは，制御用プリミティブとして曲線を利用するものである．変形する特徴的な部分に制御用プリミティブを配置し，それらを動かすことから，この方法はfeature based deformationといえる．一般に変形する対象の境界は曲線が多い．したがって，曲線を利用する方法は理想的な方法といえる．曲線と任意の点との関係は，2.3で述べたBezier曲線と任意の点との距離(および最短点を与えるパラメータ値)によって定義することができる．

本章では，1つの図形の変形(warping)，および2つの図形間のモーフィングについて以下に議論する．

4.1 1つの図形の変形

図形の変形法としては，特徴点(線分で指定)を利用する方法[3),4)]，メッシュを変形する方法(mesh warping[5)~8)])，FFD法[9)]がある．ここでは，制御用プリミティブとして，Bezier曲線を利用したfield warpingを考える．

Beierら[3)]は複数の線分を使用した．すなわち，直線

と任意の点との距離 v とその点の投影点でのパラメータ u(0<u<1.)を利用した．これを曲線に発展させて考える．もし，Bezier 曲線の次数が 1 の場合は線分と一致するから，提案法は Beier の方法の拡張アルゴリズムといえる．提案法においても，ある点 P と曲線 C の最短距離 v と投影点のパラメータ u を用いるが，これらの 2 つのパラメータは前節の方法によって簡単に算出できる．

線分を利用する方法に比べ，曲線を使用する方法の利点は次のようである．

1) 少ない曲線数で変形できる．

2) スムーズな変形ができる．線分を利用する方法の場合，いくつもの線分を曲線に沿って並べ，かつこれらの移動が必要である．

3) 曲線上の 1 点のみを移動するのみでも，複雑な変形ができる．

また，メッシュを利用して変形(mesh warping[5])する方法と比較すると，曲線を利用する変形法は次の特徴がある．

1) 図形の境界を変形させる場合，メッシュ法では，いくつもの格子点を境界に沿うように移動させる必要がある．しかし，曲線を利用する場合，境界に沿う変形は曲線上の点を数回移動することで実現できる．

2) メッシュを利用する方法(FFD 法も)は，制御点の移動と図形の移動距離が一致していないが，提案法では曲線上の点の移動量と変化量は同じである．したがって，ユーザフレンドリなインタラクティブシステムが実現できる．

複数の曲線を用いるが，まず，ある曲線 C_k についてのパラメータを説明する．任意の点 P の曲線 C_k に対するパラメータを (u_k, v_k) とすると，これらのパラメータは次式によって定義できる(図 5 参照)．

$$P=C_k(u_k)+v_kN(u_k) \qquad (5)$$

ここで，N は曲線上の点 $C_k(u_k)$ での単位法線ベクトルで，v_k は点 P と曲線との最短距離である．

曲線の変形後の点の新座標の算出は次の方法で行う．曲線 C_k が変形して $C^d{}_k$ になると，同じパラメータの点に移動するから，新しい点 P($P_k{}^d$ と表記)は次式によって求まる．

$$P_k{}^d=C^d{}_k(u_k)+v_kN^d(u_k) \qquad (6)$$

曲線が m 個の場合，曲線 C_k(k=1, …, m)が変形された後の座標 P^d は，各曲線から得られた式(6)の座標の重み平均として算出する．すなわち，次式となる(図 5(b)参照)．

$$P^d=\sum_{k=0}^{m} w_kP_k{}^d \qquad (7)$$

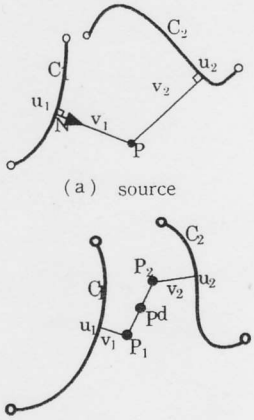


図 5 ソースおよびディスティネーション画像の対応メタサークルの抽出

Fig. 5 Finding correspond metacircles on the source and destination images

ここで，w_k は曲線 C_k に対する重み係数(v の関数)であり，点が曲線上にあるとき最も大きな値となる(文献 3)と同じ定義なので計算式は省略)．

変形しようとする図形としては，点列で表現される図形，多角形の集合，後述のメタサークルがあるが，いずれも頂点座標を上記の方法で移動すれば変形が実現できる．

4.2 メタサークルで表現される 2 つの図形間のモーフィング

2 次元カラー画像のモーフィングに関しては，制御プリミティブ(直線か曲線か)は異なるが[3] の方法と同様な方法を用いることができるのでここでは省略し，2 値画像に適した方法を議論する．特にインターネットなどによる画像の伝送を考慮し，少ない情報量の図形表現法を考える．

2 値画像を少ない情報量で表現する方法としては次の方法がある．Ranjan ら[10] は円の集合で図形を表現し，それを変形することによって 2 つの図形間の変形を行った．彼らは，図形を円の集合で表現することによって，データ量を減少させた．また，2 つの図形のマッチする円の組合せを求め，それらの円の変形軌跡を補間によって求めた．この方法では，対応する円を検出できるが，この処理はかなり複雑である．すなわち，ドローネ三角形を利用した輪郭内の円の算出，円の中心座標の並びからのスケルトン算出，隣接する円の接続関係の算出など複雑な過程が必要である．また，この方法による図形の境界線(輪郭)は円孤の集合であるので C^0 接続であり，スムーズではない．Chung ら[11] も，同様に画像を定義するため，円を使った．提案法では，このスームスネス

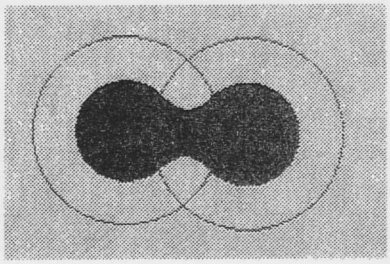
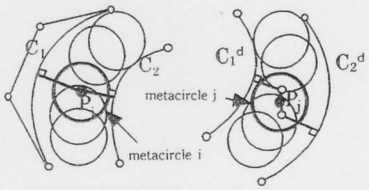


図6　メタサークルの融合
Fig. 6　Fusion of metacircles



(a) source　　(b) destination

図7　ソースおよびディスティネーション画像のメタサークルの対応関係
Fig. 7　Matching of metacircles on source and destination images

の問題を克服するため，円同士が融合するメタサークル(メタボールの2次元版)で図形の形状を表現する(図6参照)．したがって，滑らかな境界曲線を表現することができる．

ここでは，少数のBezier曲線を使った会話的な方法を提案する．この曲線は，図形のスケルトンとして入力される．前述のように，Ranjanの方法ではマッチする円の探索が複雑である．しかし，提案法では，2つの図形の特徴的な部分にBezier曲線を配置し，その曲線からの相対位置(距離)パラメータ(u, v)が一致するメタサークル(あるいは近いもの)を簡単に検出できる方法である．

1)　メタサークルによる図形の表現

メタサークルとはメタボールの2次元版である．すなわち，メタボールと同じように，円の中心に濃度を与え，それにより生じる濃度分布の等濃度曲線(等値曲線)で図形を定義する(図6参照)．メタサークルを定義するには，各円の中心位置，中心濃度，および色を指定する．任意の点における濃度は，円の中心点からの距離によって定義される．点(x, y)における濃度は次式で定義される．

$$f_i(x, y)=f(r/R_i) \qquad (8)$$

ここで，R_iはメタサークルiの有効半径，rは円の中心$P_i(x_i, y_i)$から点(x, y)までの距離，fは濃度分布関数でWiville[12),13)]の6次式を使用した．なお，rが0の点では$f(r)=1$であり，$r>R_i$では$f(r)=0$である．

n個のメタサークルがある場合，図形の境界線は次式を満たす曲線(等値曲線)で表わされる．

$$f(x, y)=\sum_{i=0}^{n} q_i f_i(x, y)=T \qquad (9)$$

ここで，Tは閾値で一般に0.5が用いられる．q_iは中心での濃度値である(負の値も許される)．また，1つのメタサークルしかない場合は，半径$R_i/2$の円が境界線となる．

簡単な形状のものについては，メタサークルは人手で入力する．また，複雑なものは，輪郭線上のサンプリング点(折れ線近似のため)を入力し，これをもとにmedial axises[14)]を使用して図形領域内の円を算出する．さらに，各サンプリング点での濃度を計算し，その誤差(Tとfとの差)を計測し，誤差が最小になるようにメタサークルの位置，半径，濃度を変化させて，最適なパラメータ(x_i, y_i, R_i, q_i)を算出する．この最適化の計算には，最急降下法を利用した．なお，この方法は村木[15)]の方法の2次元版ともいえるが，村木の方法では一つのメタボールの分裂を繰り返すことによって，サンプリング点を通過するボールを算出するので，多数のボールを得るまで処理時間を要す．それに対し，提案法は初期値としていくつかのメタサークルを利用するので，高速に最適値に収束することができる．

2つの図形，すなわち，ソースとディスティネーション画像に対して，独立に図形領域内を塗りつぶすためのメタサークルを求めておき，次の方法でメタサークル同士の対応関係を求め，各メタサークルの位置，半径，密度を補間して中間画像を生成する．

2)　メタサークルのマッチング

ソースとディスティネーションの画像上の対応する位置にBezier曲線をおく(図7は，ソースとディスティネーションのメタサークルを挟むように曲線を配置した例である)．例えば，動物の場合なら，背骨から首へのスケルトンや足など3，4本でいい．なお，ソースとディスティネーションでのメタサークルの数は一致する必要はなく，次の手順でマッチングを行う．これらの曲線を$C_k(k=1, \cdots, m)$とし，ボールjの中心座標を$P_j(x_j, y_j)$とする．

以下にソースとディスティネーション画像を表現するためメタサークルの対応関係の算出アルゴリズムを述べる．

1)　ソース画像のメタサークルiの中心を曲線kに対する曲線座標系(u^k_j, v^k_j)で表わし，式(6)からディスティネーション画像での位置$P_j(u^d, v^d)$を算出し，各々

のメタサークルをソース画像にマッピングする．

2) ディスティネーション画像において P_j に最も近い座標のメタサークルを割り当てる．

3) ディスティネーションにおいて，割り当てられていないメタサークルがあれば，1)と同様に，ディスティネーション画像で算出した座標から，ソース画像での位置を算出し，2)と同様にマッチするメタサークルを探索する．

なお，1つのメタサークルが複数のメタサークルに対応することも許すものとし，指定した距離内に対応するメタサークルがない場合には，ダミーとして半径0のサークルを発生する．したがって，中間の画像においては，ソースとディスティネーション画像でのメタサークルの数の最大値よりメタサークル数は多くなる．

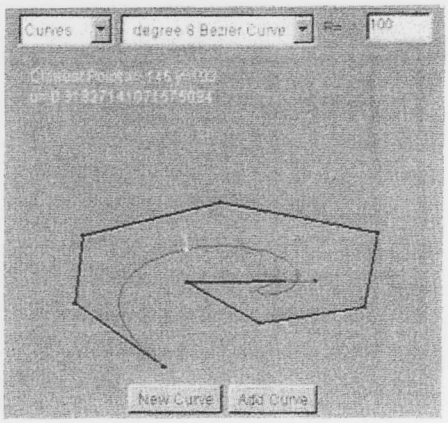
3) 中間図形の補間

対応するサークルが求まっているので，これらのサークルについて，中心位置，半径，および濃度を時間の関数として線形補間する．なお，中心位置の補間に関しては，次の方法も選択できる．メタサークルを直接補間するだけでは不適当な場合がある．この場合，まず，対応する曲線(スケルトン曲線)の制御点どうしを線形補間(または指定した軌跡で移動)し，その後にそれらの曲線で決まる位置にメタサークルを配置する．もし必要なら，制御点の補間軌跡を指定することもできるようにした．

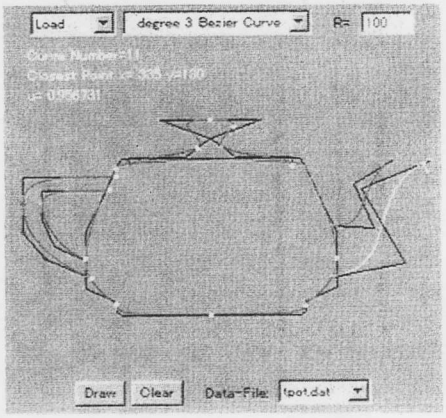
4.3 メタボールで表現される形状の変形

測定したデータに基づき，メタボールにより3次元物体を構成する方法を，筆者らは既に提案しているが[16]，この方法は精度が十分ではなかった．そこで，この方法を改良した方法を用いた．3次元物体をいくつかの断面によって構成し，各断面においてラフにメタボールを配置し，断面の輪郭上のサンプリング点において，誤差が最小になるようにメタボールの位置を最適化する．この最適化には，4.2の1)と同様に最急降下法を用いた．この方法によって，例えば人体モデルの場合，測定データと数ミリ単位の誤差でメタボールによる近似が実現できた．

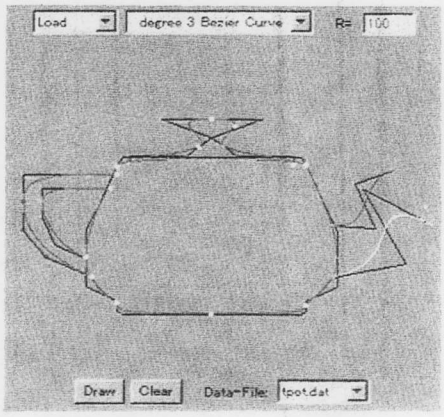
上記方法において，サンプリング点の変形移動に提案法を適用する．すなわち，サンプリング点に沿って曲線を配置し(形状の断面境界に沿って曲線を配置することと等しい)，この曲線をインタラクティブに変形することによって，サンプリング点もスムーズに移動される．このサンプリング点を元にメタボールの位置，半径を最適化することによって，メタボールで構成された3次元形状の自由形状変形が実現できる．



(a) degree 8 planer Bezier curve



(b) teapot composed of multiple curves



(c) teapot composed of multiple curves

図 8 曲線の最近点の検出および変形 Java Applet

Fig. 8 Java Applet for detection of the closest point on curves and deformation of the curves

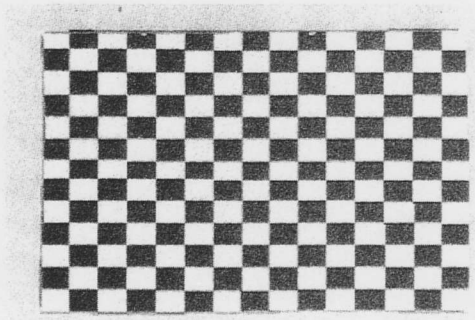
5. 適用例

提案した最近点検出手法を，3次から9次までのBezier曲線について実験したが，ほとんどの場合，4回程度の分割またはクリップ(2回の分割よりなる)で解が算出できた．なお，プログラムはインタラクティブ性の優れたJava言語を用いて開発した．サンプリング法との比較を行った結果，パラメータ空間で10^{-3}の精度の計算の場合，提案法は平均で100倍高速であった(Javaプログラムを C言語版に変換して測定)．

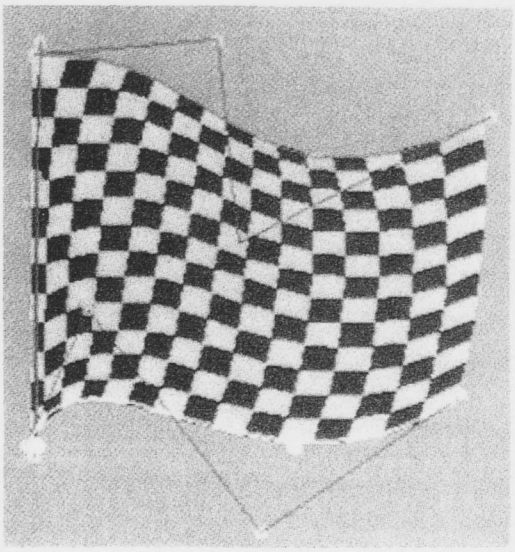
図8に最近点の検出のJava Appletを示す．(a)は高次の曲線の例として，8次Bezier曲線上の最近点とその距離を検出した例を示したものである．図(b)は複数の3次Bezier曲線を用いてティーポットの輪郭を表現したもので，指定した点に最も近い曲線を抽出し(図中白色で表現された曲線)，それを修正している場面である(図(c)が修正後)．

図9はワーピングの例である．図(a)，(b)はチェックの柄の旗(多角形の集合で表現)を変形する例である．この場合，初期画像では，長方形の旗の上，下，左側に3つの曲線を配置(図(a)参照)し，それらを変形することで生成した結果の画像である．上，下の曲線のみをクリックし，曲線を移動することで，ワーピングさせている．左の曲線は変形させないため，旗の左境界は直線のまま保持している．また，上下曲線に近い点は曲線にスムーズに沿って移動していることが分かる．なお，これはJava Appletでリアルタイムで動作可能であり，わずか数回のマウス操作(上下曲線付近を2個所のみドラッグ)で実現できた．

この方法では，クリックした点と同期して曲線が移動

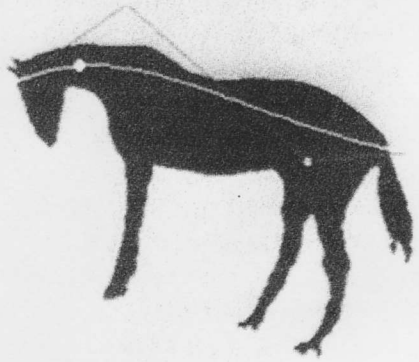


(a)



(b)

図 9　ワーピングの例
Fig. 9　Example of warping



(a)

(b)

図 10　ワーピングの例
Fig. 10　Examples of warping

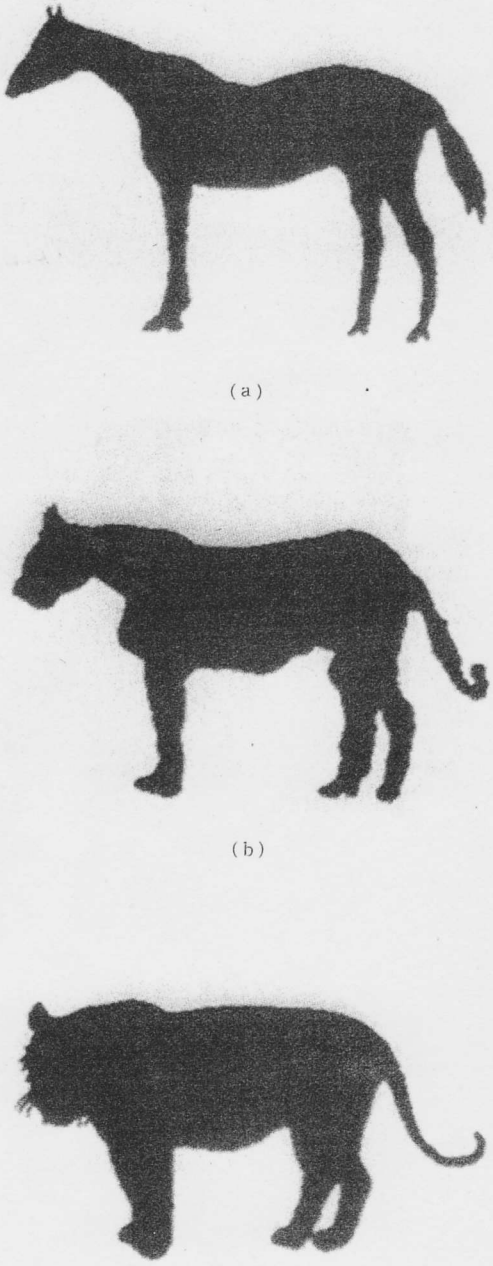


図 11　馬から虎へのモーフィングの例
Fig. 11　An example of morphing from a horse to a tiger

する．したがって，デザイナーが動かしたい点を望む量移動することができる．また，移動したくない部分にも曲線を置き，その曲線を移動させない限り変化しない．このように変形したい部分と変形したくない部分をコントロールできる．

表 1　メタサークルの数
Table 1　The numbers of metacircles for Fig. 10

Object	number of metacircles
horse	301
tiger	178

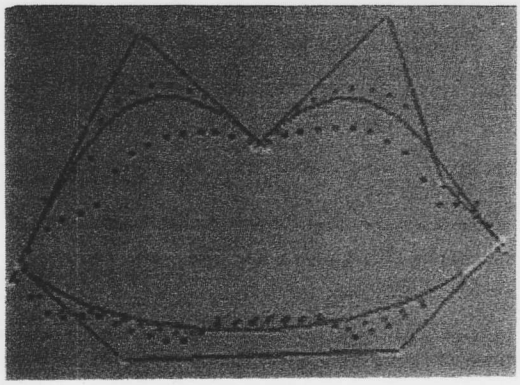
図 10 はメタサークルで表現された馬を変形した例である(変形前の形状は図 11(a)参照)．図(b)では，スケルトンとして使用したわずか 1 本の曲線を変形するのみで，首や背中の部分の変形が実現された．図(c)は，メタサークルの配置を示しており，かつ頭の部分が上方へ変形された例である．この場合，前後の足の部分にも曲線が置かれ，これを固定することによって頭部のみの変形が実現された．

図 11 はモーフィングの例として，馬から虎への変形を表示している．図(a)はソース画像(馬)で，(c)は最終画像(虎)であり，また，図(b)はこれらの画像間の変形途中の画像である．**表 1** に，この例に対するメタサークルの数を示す．

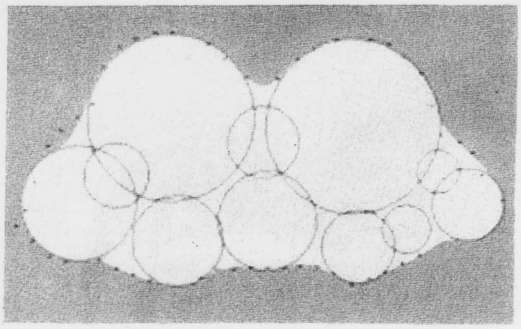
図 12 はメタボールで表現された人体を局所的に変形(胸部の変形)した例である．このモデルは人体のいくつかの水平断面を元にメタボールを配置して作成されたものである．各断面において，輪郭を構成するサンプリング点を利用してメタボールを配置したものである(サンプリング点を等濃度面が通過するようにメタボール位置，半径および濃度を最適化)．これらのサンプリング点は，提案法によってインタラクティブに変形できる．移動されたサンプリング点を輪郭とするようにメタボールを最適化することによって，人体の局部変形が実現できる．この最適化計算には前述のように最急降下法を用いた．図(a)は，断面の境界線上に沿って 3 本の Bezier 曲線を重ねている．さらに，曲線を変形することによって，サンプリング点は曲線に沿ってスムーズに移動する．図(b)において，移動されたサンプリング点に適合するようにメタボールが最適化されたものである．図(c)はオリジナルの人体であり，(d)は変形後の人体である(胸部が膨らんでいる)．

6. むすび

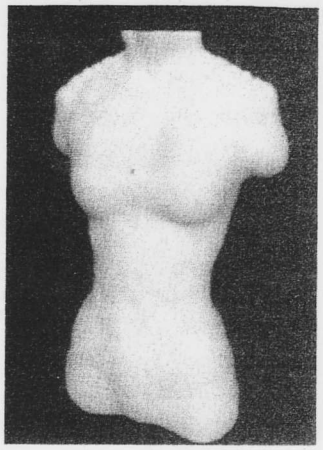
この論文では，Bezier 曲線と任意の点との最近点を検出する方法を提案し，その応用例として，インタラクティブな曲線の変形法および図形の変形(warping/morphing)について提案した．提案した最短点の検出法は次の特徴をもつ．1) Bezier Clipping 法により，1 次式の少ない反復計算で解を求めることができる．リアルタイムの処理が可能でインタラクティブシステムに適用で



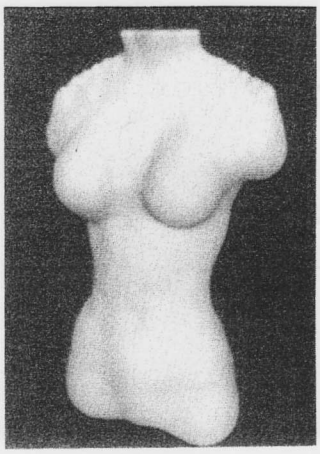
(a)　deformation of boundary



(b)　optimization of metaballs



(c)　original



(d)　deformed

図 12　メタボールで表現された人体の局所変形

Fig. 12　A human body modeled by metaballs (some of sample points on the cross sections of metaball surface are deformed by the proposed method)

きる．2) 指定距離内での最近点の検出が効率よく算出できる．

また，図形の変形法については，次の特徴がある．

1)　一つの図形の変形(warping)は少数の曲線で実現できる．

2)　滑らかな変形(ひずみが少ない)を実現することができる．

3)　曲線上の一点を移動することによって，複雑な変形を実行することができる．

4)　変形量を直接指定できるので，インタラクティブシステムに，とても効果的である．

さらに，2値画像をメタサークルで表現する方法，およびメタボールで表現された形状を変形する方法を議論した．これらの方法は，近年注目されているインターネットを利用したデータの伝送や変形に有用である．

(1998年3月20日受付)

参考文献

1) L. Piegle, W. Tiller: "The NURBS BOOK", pp. 294-234 (1995).
2) T. Nishita, T. Sederberg, M. Kakimoto: "Ray Tracing Trimmed Rational Surface Patches", Computer Graphics, Vol. 24, No. 4, pp. 337-345 (1990-8).
3) T. Beier, S. Neely: "Feature-Based Image Metamorphosis", Computer Graphics, 26, 2, pp. 35-42 (1992).
4) Y. Sato I. Sato, K. Ikeuchi, "3D Shape and Reflectance Morphing", Proc. of International Conference on Shape Modeling and Applications '97, pp. 234-242 (1997).
5) G. Wolberg: "Digital Image Warping", IEEE Computer Society Press (1990).
6) T. Nishita, K. Fujii, E. Nakamae: "Metamorphosis using Bezier Clipping", Pacific Graphics '93, pp. 162-173 (1993).
7) T. Nishita, S. Takita, E. Nakamae: "A Display Algorithm of Brush Strokes using Bezier Functions", CG International

'93, pp. 244-257 (1993-6).
8) T. Nishita, E. Nakamae: "A Method for Displaying Metaballs by using Bezier Clipping", Computer Graphics Forum, Vol. 13, No. 3, pp. 271-280 (1994-9).
9) T. Sederberg, S. Parray: "Free Form Deformation of Solid Geometric Models", Computer Graphics, Vol. 20, No. 4, pp. 151 160.
10) V. Ranjian, A. Fournier: "Matching and Interpolation of Shapes using Unions of Circles", EUROGRAPHICS '96, Vol. 15, No. 3, pp. 129 142 (1996)
11) J. Chung, N. Ohnishi: "Chain of Circles for Matcing and Recognition of Planer Shapes", IDY97-52, pp. 23-30 (1997).
12) B. Wyvill, C. McPheeters, G. Wyvill: "Data structure for soft objects ", The Visual Computer, 2, pp. 227-234 (1986)
13) B. Wyvill, C. McPheeters, G. Wyvill: "Animating soft objects", The Visual Computer, 2, pp. 235 242 (1986)
14) B. Bittar, N. Tsingos, M. Gascuel: "Automatic Reconstraction of Unstructured 3D Data: Combining a Medial Axix Implicit Surfaces", EUROGRAPHICS95, Vol. 14, No. 3, pp. 458 468 (1995).
15) S. Muraki: "Volumetric Shape Description of Range Data using Blobby Model", Proc. of SIGGRAPH91, Vol. 25, No. 4, pp. 227 235 (July 1991).
16) 松田, 西田: 「メタボールとボクセルデータの統合化造形システムの開発」, 画像電子学会誌, Vol. 26, No. 4, pp. 314 324.

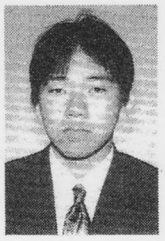
西 田 友 是 (正会員)

昭46, 広島大・工・電気工卒. 昭48, 同大学院修士課程修了. 同年, マツダ入社. 昭54, 福山大学工学部電子電気工学科講師. 昭59, 同大学助教授. 平2, 同大学教授. コンピュータグラフィックスに関して, 3次元物体のリアルな表現法, 照明シミュレーション, 景観予測などの研究に従事. 昭62, 情報処理学会研究賞受賞. 工学博士. 情報処理学会, 電気学会, 電子情報通信学会, ACMの各会員.

松 田 亮 治

昭55, 福井大・工・繊維工卒. 昭57, 同大学院修士課程修了. 同年, 広島県立福山繊維工業試験場(現広島県立東部工業技術センター)に入所. コンピュータグラフィックスを用いた衣服の着装シミュレーションや着心地などの研究開発に従事. 現在, 福山大学大学院工学研究科電子情報専攻博士課程在籍中. 情報処理学会, 繊維学会, 日本繊維機会学会の各会員.

高 栄 一 寿



平9, 福山大学・工・電子・電気工卒. 同年, 同大学大学院修士課程入学. コンピュータグラフィックスに関して, ヴォリュームビジュアライゼーションおよびCADシステムなどで使用できる曲線処理のアルゴリズムの研究に従事. 情報処理学会の学生会員.

た自由形
始めエン
いる. 2
ティブと
ることに
ィブとし

法として
リング点