# 直感的な曲線操作を実現するベクタ変換の検討

河村 圭 石井 大祐 渡辺 裕
Kei Kawamura Daisuke Ishii Hiroshi Watanabe

## 1. まえがき

ベクタグラフィクスの再生環境が普及しつつあり，コンテンツ需要が増加している．そのため，ラスタ表現から自動的にベクタ表現を取得するベクタ変換技術が重要となっている．また，得られたベクタ表現を再利用や微修正できるように，編集性（曲線操作のしやすさ）が求められる．

ベクタ表現（ベジエ曲線）は「通過点」と「制御点」から構成されており，曲線の頂点付近に通過点が存在していると，曲線を操作しやすいという特徴がある．これまで，少ない符号量と高い再現性を実現する変換手法は提案されているが，ツール利用者による曲線操作は十分に考慮されていなかった．

そこで，我々は変換性能を維持したまま，容易な曲線操作が可能なベクタ表現を生成する手法を提案している．本稿では特に，得られたベクタ表現の曲線操作について，曲率と通過点の位置関係に着目した定量的な評価手法を提案する．

## 2. ベクタ変換手法とベジエ曲線

ベクタ変換とは，2 値のビットマップ（ラスタ表現）からベクタ表現に変換することである．本稿では，与えられたディジタル曲線からベジエ曲線を得ることをベクタ変換と定義する．

Medioni らは，B スプライン曲線の当てはめと，そこで得られる曲率を元に角（Corner）を検出する手法を提案している [1]．これは曲率の計算に必用なディジタル曲線の微分を，B スプライン関数を用いて実現していることになる．

この手法は曲率のみを基準にベクタ変換を実現しているため，通過点の個数が増大してしまう．また，パラメータの設定が困難なために変換精度も低いという問題がある．

Selinger らは，多角形近似と頂点の曲線近似によるベクタ変換を提案している [2]．線分とパスの距離が 1/2 画素以下という制約条件の下で，線分の本数と誤差が最小となる多角形で近似を行う．通過点を多角形の辺の中点に，制御点を辺上に置いて頂点を滑らかなベジエ曲線で置換する．最後に，多角形近似の際に分割されてしまった長い曲線を連結する．

この手法は，高い変換性能を有しているが，曲率の小さな位置に通過点を置くため，曲線操作が困難になるという問題がある．

*A Study on Vectorization for Realizing Intuitive Curve Manipulation

†早稲田大学大学院 国際情報通信研究科，
Graduate School of Global Information and Telecommunication Studies, Waseda University.

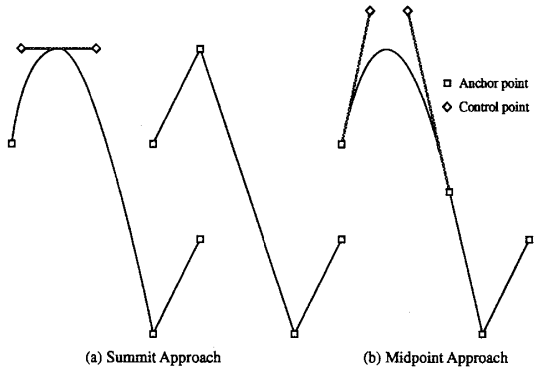


図 1 An example of two approaches. (a) Anchor point is laid on the summit. (b) Anchor point is laid on the middle of summits.

ここで，ベジエ曲線の描画方針には大きく分けて 2 種類ある．図 1 に示すように，後述する曲線の頂点に付近に通過点を置く手法と，頂点同士の中程におく通過点を置く手法である．Medioni らの手法は前者であり，Selinger らの手法は後者である．前者の場合，曲線の膨らみの変更やカーブとコーナーの相互変換が容易である．

## 3. 提案する操作性の評価手法

### 3.1 ディジタル曲線の曲率

曲線の局所的な曲がり具合を定量的に表現する手法として曲率がある．まず，ある曲線の微少区間を円で近似したとき，その円の半径を曲率半径と定義する．また，曲率半径の逆数を曲率とする．なお直線の曲率は 0 となる．さらに，右回りと左回りで正負が逆転する．

曲線が媒介変数表現で与えられるとき，曲率 $\kappa$ は

$$\kappa = \frac{x'y'' - x''y'}{(x'^2 + y'^2)^{\frac{3}{2}}} \tag{1}$$

となる．ここで，曲線が離散値として与えられるとき（ディジタル曲線），微分の計算方法に自由度がある．

一方，曲線の構造を表す手法として空間尺度 $\sigma$ を導入した空間尺度フィルタリングがある．尺度の異なるさまざまなガウス関数を畳み込んだ曲線に対して，尺度に対する変曲点の位置変化を追跡する手法である．この考え方を基にすると，曲線の微分計算についても空間尺度を導入することで，さまざまな空間尺度による曲率が算出できる．また，微分の計算方法を定量的に扱うことが出来る．

以上の考察から，ディジタル曲線の微分を

$$x'[n] = G'[\sigma, n] * x[n], \quad (2)$$
$$y'[n] = G'[\sigma, n] * y[n], \quad (3)$$
$$x''[n] = G'[\sigma, n] * x'[n], \quad (4)$$
$$y''[n] = G'[\sigma, n] * x'[n] \quad (5)$$

と定義する．ここで，$G$ はガウス関数，$\sigma$ は分散，$*$ は畳込み演算子である．また，ディジタル曲線の媒介変数表現として

$$|x[n+1] - x[n]| \in \{0, 1\}, \quad (6)$$
$$|y[n+1] - y[n]| \in \{0, 1\} \quad (7)$$

を仮定している．

### 3.2 提案するベクタ変換の概要と評価手法

ベジエ曲線の編集には，曲線の膨らみを変更する場合と，カーブとコーナーを相互変換する場合が想定される．いずれの場合も，曲線の頂点付近に通過点が存在していると，編集しやすいことが知られており，直感的な曲線操作を実現できる．

ここで，自由曲線のうち凸曲線区間に対して，凸の頂点が曲線の頂点であるといえる．しかし，凸曲線区間の取り方には自由度があり，さらに空間尺度によっても凸曲線区間が異なるため，曲線頂点は一意に決定できない．そこで，空間尺度を固定したときに曲率が極値（極大，または極小）になる点を曲線の頂点であると定義する．ただし，曲率が極値になっていても絶対値が小さいときは，曲線が穏やかに曲がっていることを表している．この場合，空間尺度が少し大きくなるだけで頂点とは定義されないと考えられる．そこで，曲率がしきい値 $\mathbf{th}$ より大きい場合のみ，「頂点」であるとする．

我々は Selinger らの手法を改善し，以下のように頂点ではなく辺を曲線近似するベクトル変換を提案する．通過点を頂点の中線上に，制御点をベジエ曲線が辺に接するように置いて滑らかな曲線で置換する．その結果，通過点が曲線の頂点付近に置かれることが多くなる．これは多角形の頂点は曲率が大きいという仮定に基づいている．

なお，通過点は曲線の頂点にあることが望ましいが，頂点以外の位置にあっても編集性には影響を与えない．実際，変換精度を維持するために，頂点以外にも通過点が存在している．従って，評価手法としては曲線の頂点付近に通過点が存在している確率を採用する．

## 4. 実験と考察

約 260 個の頂点を有する 2 値のイラストを入力としてベクタ変換を適用し，曲率と通過点の位置関係を調査した．なお，ベクタ変換の従来手法として Selinger らの手法を用いる．パラメータ $\sigma$ は，1 未満にすると微分が成立せず，大きすぎると大局的な頂点しか抽出できない．本稿では予備実験により $\sigma = 1$，$\mathbf{th} = \mathbf{2}/\pi$ を採用する．

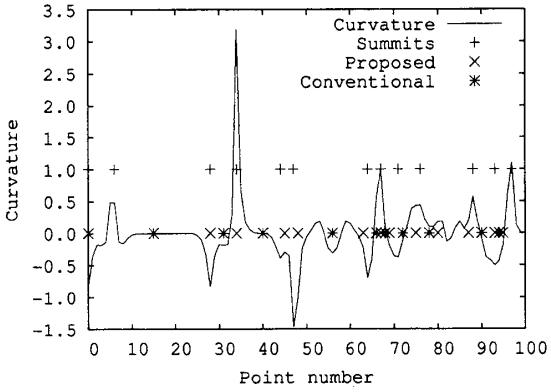


図 2 Point and Curvature

表 1 Summits Ratio

| Proposed Method | Conventional Method |
|---|---|
| 70.4% | 32.3% |

図 2 にディジタル曲線を構成する点の番号と，その点の曲率（Curvature），提案した曲線の頂点（Summits），提案手法による通過点（Proposed），従来手法による通過点（Conventional）を示す．この図より，提案手法は多くの場合曲線の頂点に近い位置に通過点が存在していることが確認できる．一方，従来手法ではほとんどの通過点は曲線の頂点の間に存在していることも確認できる．

また，画像全体について，曲線の頂点付近に通過点が存在する頂点の割合を表 1 に示す．従来手法でも 30%程度は曲線の頂点と通過点が一致している．これは，頂点がカーブではなくコーナーと判定されたためであると考えられる．提案手法は従来手法に比べて割合が向上していることから，直感的な編集が可能な部分が大幅に増えているといえる．

## 5. まとめ

本稿では，ベクタ変換において直感的な曲線操作を実現するために，曲線の頂点付近に通過点をおく手法について述べた．さらに，曲率と通過点の位置に着目して，定量的な評価手法を提案した．提案手法は空間尺度の概念を導入し，定量的にディジタル曲線の曲率計算を実現するとともに，曲線の頂点として曲率の極値を用いることを提案した．実験により，定量的に曲線の頂点に高い確率で通過点が置かれることを確認した．

## 参考文献

[1] G. Medioni, *et al.*, "Corner detection and curve representation using cubic B-splines," IEEE Int. Conf. on Robotics and Automation 1986, Volume 3, pp. 764–769, Apr. 1986.

[2] Peter Selinger, "Potrace: a polygon-based tracing algorithm," `http://potrace.sourceforge.net/potrace.pdf`, 2003.