対話的な点を用いた線画作成のための近傍点によりリパラメタライズ可能な張力のあるベジェ曲線*

○佐藤 信 † 三輪 譲二‡
岩手大学 † 岩手大学 ‡

概要

対話的な点入力により線画を作成するための，張力のあるベジェ曲線を提案する．その概形を点で指定することにより，曲線を作成可能であるので，曲線に関する数学的知識を必要とせずに，直感的な操作で線画を作成可能である．提案のアルゴリズムでは，曲線が通過するべき通過制御点，その近くを曲線が通過するべき近傍制御点そして張力により，曲線を作成する．この近傍制御点の個数は，曲線の自由度に制限されないので，曲線形状に合わせて自由に指定可能である．このことから，この曲線の表現形式は，初期形状を洗練化して，目的形状を作成する対話的な線画作成に適している．

1 はじめに

本稿では，対話的な点入力により直観的に曲線形状を操作して，線画を作成するための，ベジェ曲線の表現形式の変換を提案する．この曲線では，曲線が通過するべき通過制御点，その近くを曲線が通過するべき近傍制御点そして張力により曲線を表現する．

2 線画作成のためのベジェ曲線

2.1 関連研究との比較

ベジェ曲線の表現形式を目的に合わせて変換する研究としては，Rasala[2] がある．その手法では，曲線の自由度に対応した通過点のみを使用して，ベジェ曲線を表現している．本稿で提案する表現形式では，曲線の自由度により制限されずに，近傍制御点を使用可能である．多項式曲線で張力を表現する研究としては，Barsky[1] があり，1 階導関数と 2 階導関数で張力を表現している．本稿の提案では，1 階導関数のみで張力を表現し，張力が増加すると，曲線は近傍制御点の周囲を通過する形状から通過制御点を結ぶ直線形状に近づくようにしている．

2.2 ベジェ曲線の表現形式変換

ここで説明のために，n 次ベジェ曲線の第 i セグメントの k 階導関数 $\boldsymbol{Q}_i^{(k)}(u)(k \geq 0)$ を，以下に示す．

$$\sum_{j=0}^{n} \boldsymbol{V}_{i,j} B_{i,j}^{(k)}(u) = \boldsymbol{Q}_i^{(k)}(u) \tag{1}$$

ここで，$\boldsymbol{V}_{i,j}$ と $B_{i,j}^{(k)}(u)$ は，それぞれ制御点と基底関数である．(1) 式から，曲線通過点 $\boldsymbol{P}_i(u)$ と 1 階導関数 $\boldsymbol{D}_i^{(1)}(u)$ は，以下のとおりである．

$$\sum_{j=0}^{n} \boldsymbol{V}_{i,j} B_{i,j}^{(0)}(u) = \boldsymbol{P}_i(u) \tag{2}$$

$$\sum_{j=0}^{n} \boldsymbol{V}_{i,j} B_{i,j}^{(1)}(u) = \boldsymbol{D}_i^{(1)}(u) \tag{3}$$

曲線の表現で使用する $\boldsymbol{P}_i(u)$ と $\boldsymbol{D}_i^{(1)}(u)$ に関する (2) 式と (3) 式を，その自由度に合わせて，一意に形状を決定可能な個数を用意する．それを以下のとおりに表現する．

$$\boldsymbol{V} = M^{-1}\boldsymbol{Q} \tag{4}$$

ここで，$\boldsymbol{V}$ は $\boldsymbol{V}_{i,j}$ を要素とするベクトル，M は $B_{i,j}^{(k)}(u)$ を要素とする行列，そして $\boldsymbol{Q}$ は $\boldsymbol{P}_i(u)$ と $\boldsymbol{D}_i^{(1)}(u)$ を要素とするベクトルである．

2.3 近傍制御点による形状制約を用いた曲線

以下の手順に示すように，通過制御点と近傍制御点に関する制約条件を用いて曲線を作成する．なお，近傍制御点に関する制約方程式は以下のとおりである．これは，(2) 式に誤差項を追加したものである．

$$\sum_{j=0}^{n} \boldsymbol{V}_{i,j} B_{i,j}^{(0)}(u) + \alpha \boldsymbol{E}_l = \boldsymbol{N}_i(u) \tag{5}$$

*Bezier Curve with Tension Using Neighbour Point Reparametarization for Interactive Drawing
†Makoto Satoh, Iwate University
‡Jouji Miwa, Iwate University

ここで，$\boldsymbol{N}_i(u)$ は近傍制御点である．また，α は張力を調節するための係数であり，$\boldsymbol{E}_l$ は l 番目の近傍制御点と曲線上の対応点までの距離を調節するための変数である．

1. (2) 式と (4) 式を用いて通過制御点に関する制約条件を，そして，(5) 式と (4) 式を用いて近傍制御点に関する制約条件を，$\boldsymbol{P}_i(u)$,$\boldsymbol{D}_i^{(1)}(u)$ を独立変数として作成する．ここで，張力係数 α を決定する．
2. 制約条件の連立方程式を，独立変数 $\boldsymbol{P}_i(u)$,$\boldsymbol{D}_i^{(1)}(u)$ と E_l のノルムが最小値を持つように解く．
3. (4) 式に $\boldsymbol{P}_i(u)$ と $\boldsymbol{D}_i^{(1)}(u)$ を代入して，ベジェ曲線の制御点を求める．

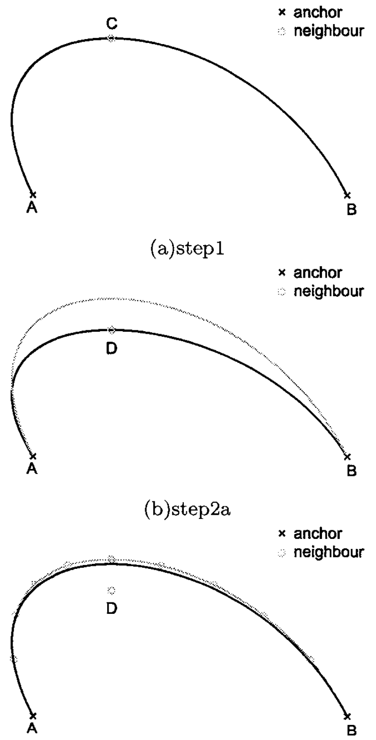
2.4 通過制御点と近傍制御点の例

1 セグメントのベジェ曲線により，曲線を作成する例を説明する．通過制御点を $\boldsymbol{P}_0(0)$ と $\boldsymbol{P}_0(1)$，近傍制御点を $\boldsymbol{N}_0(0.5)$，そして張力を調節するための 1 階導関数を $\boldsymbol{D}_0^{(1)}(0)$ と $\boldsymbol{D}_0^{(1)}(1)$ とする．

1. $\boldsymbol{P}_0(0)$ と $\boldsymbol{P}_0(1)$ について (2) 式を作成する．また，$\boldsymbol{D}_0^{(1)}(0)$ と $\boldsymbol{D}_0^{(1)}(1)$ について (3) 式を作成する．
2. (4) 式の逆行列を求める．
3. $\boldsymbol{P}_0(0)$ と $\boldsymbol{P}_0(1)$ について (2) 式と (4) 式を用いて，通過制御点に関する制約条件を作成する．また，$\boldsymbol{N}_0(0.5)$ について (5) 式と (4) 式を用いて，近傍制御点に関する制約条件を作成する．ここで，α を決定する．
4. 作成した制約条件の線形連立方程式を，独立変数 $\boldsymbol{P}_0(0)$,$\boldsymbol{P}_0(1)$, $\boldsymbol{D}_0^{(1)}(0)$,$\boldsymbol{D}_0^{(1)}(1)$ と $\boldsymbol{E}_l$ のノルムが最小値を持つように解く．
5. (4) 式に $\boldsymbol{P}_0(0)$,$\boldsymbol{P}_0(1)$, $\boldsymbol{D}_0^{(1)}(0)$,$\boldsymbol{D}_0^{(1)}(1)$ を代入して，ベジェ曲線の制御点を求める．

3 実装と結果の検討

曲線形状洗練化を Java 言語を使用して実装した例を，図 1 に示す．図 1(a) では，通過制御点 A と B，近傍制御点 C そして張力係数 $\alpha = -0.0001$ により曲線 AB を作成している．図 1(b) は，図 1(a) の近傍制御点 C を D に変更して曲線を洗練化している．図 1(c) は，図 1(a) の曲線上の点を自動的にサンプリングした近傍制御点と近傍制御点 D で曲線を洗練化している．洗練化前の曲線から等間隔に作成した近傍制御点を使用することにより，洗練化前の形状特徴を維持しながら近傍制御点 D に近づくように形状を洗練化している．



(a)step1

(b)step2a

(c)step2b

図 1: Examples of curve refinement.

これらの例から，曲線の通過点と近傍点を指定するという直感的な操作で，曲線形状の洗練化が可能であり，制御点を変更することにより初期形状から対話的に形状を洗練化することが可能であることが分かる．

4 おわりに

対話的な点入力により直観的に曲線形状を操作して，線画を作成するために，近傍制御点を用いたベジェ曲線の表現形式の変換を提案した．この曲線を使用した作画方式についての研究は，今後の課題である．

参考文献

[1] Barsky, B. A.: Exponential and Polynomial Methods for Applying Tension to an Interpolating Spline Curve, *Comput. Vision, Graphics and Image Process. (USA)*, Vol. 27, pp. 1–18 (1984).

[2] Rasala, R.: Explicit Cubic Spline Interpolation Formulas, *Graphics Gems*, pp. 579–584 (1990).