5C-06

最近点検出による3次元ベジエ曲線・Bスプラインの編集

An Editing Method for 3D-Bezier Curves and B-Spline by using Detection of Closest Points

西田友是
Tomoyuki Nishita
プロメテックＣＧリサーチ/デジタルハリウッド大学
nishita@shudo-u.ac.jp

1. はじめに

コンピュータグラフックスや CAD では、種々の曲線・曲面の作成・編集が重要な要素の一つである。本稿では 2 次元あるいは 3 次元曲線をスクリーン上でポインティングデバイス（マウス）を利用して、インタラクティブに簡易に編集するシステムを提案する。代表的な曲線としてベジエ曲線、Ｂスプライン曲線，ＮＵＲＢＳが用いられる。こうした曲線は制御点（重みや節点も必要な曲線も）で定義される。したがって、制御点をマウスで移動する編集が一般的である。本稿では制御点に限らず曲線上の任意の点（マウスからの最短点）を掴んで編集する。そのため、曲線と点との距離をベジエ関数で表現し、効率的に距離計算を行う方法を提案する。この関数の制御点の凸包を利用し、Bezier Clipping 法[1]を適用して最短点を抽出できる方法である。

2. 従来法および基本的な方法

曲線上の任意の点を掴んで編集するが、曲線上の正確な点ではなく近傍点で処理する。そのため、本稿では点と曲線の（2 乗）距離関数を利用する。複数の曲線を扱う場合、まずスクリーン上のプローブ（マウスで指定）に近い曲線を選択し、プローブに最短の曲線上の点の座標およびパラメータ値を抽出する。次にマウスドラッグに応じ曲線を変形する。あるいは、指定点での曲線分割、制御点の追加、指定点での傾き編集もできるものとする。曲線としては主としてベジエ曲線、Ｂスプライン曲線（有理曲線も考慮）を扱うが、Ｂスプライン曲線はベジエ曲線に変換できるので、ベジエ曲線に対応できれば十分である。著者は20年前に有理ベジエ曲面に関してはレイトレーシング法で曲面との交点を求める方法を発表[1]（Bezier Clipping 法）しているが,本稿ではBezier Clipping 法を利用した最短点検出を用いる方法を論じる。また、筆者らは、点・曲線・曲面間の最短距離の計算法についても発表している[3, 4]. これらを応用した効率的な距離関数計算を提案する。指定点に近い曲線は指定点を中心にした円（3 次元なら球）に交差する曲線および区間を抽出できる点に特徴がある。本稿で採用する Bezier Clipping 法は、次のものに応用されている. 多項式の解、曲線と線分との交点、直線と曲面との交差、曲線と点との最短点の抽出、有理ベジエ曲面のレイトレーシング、曲線同士の交差、平面と曲面との距離[2, 3]、レイマーチング[5]。いずれも高次多項式であり、最短点を与えるパラメータの計算に尽きる. 提案法は基本的に曲線の再分割により、解に収束する方法であり、分割区間は線形計算のみで実現できる。

3. 提案法

著者は既に曲線の最短距離計算法を発表しているが[2-4]、そのまま適用するとコストがかかるので改良する。有理曲線でも適用できるが、簡単のため 2 次元で非有理曲線と点との距離についてまず説明する。まず, 制御点P_k (x_k, y_k)をもつ n 次ベジェ曲線（パラメータ t）を考える。

$$\mathrm{P(t)} = \sum_{k=0}^{n} P_k B_k^n(t) \qquad (1)$$

ここで、$B_k^n(t)$はバーシュタイン多項式で、曲線上の点$\mathrm{P}(t)$、曲線 P への投影関数q、および 2 点 PQ 間の距離の 2 乗距離関数Dを考える。基本的にはベクトルの内積の形式であるが、実際に適用する際はすべてベジエ関数に変換される。

$$D(t) = \big(\mathrm{P}(t) - Q\big) \cdot \big(\mathrm{P}(t) - Q\big) \qquad (2)$$

$$\mathrm{q(t)} = \mathrm{P'(t)} \cdot (\mathrm{P(t)} - \mathrm{Q}) \qquad (3)$$

曲線上の点$\mathrm{P(x(t), y(t))}$の$(\mathrm{x, y})$座標は式(1)のベジエ関数で定義されているものとする。点Pの$(\mathrm{x, y})$座標にベジエ曲線の式を代入すると、これら両式ともパラメータtに関するバーシュタイン多項式(有理の場合は分数形式)となる。また、式(3)は点Pの接線とベクトルPQの内積が0になる点である。

(1) 曲線との近傍距離

近傍円(または球)と曲線の交差判定は式(2),(3)を利用できる。円や球でも曲線への距離は、$2n$ 次のベジエ関数であり、この関数が解があるかで交差判定ができる。中心(x_q, y_q)および半径 R の円とn次ベジエ曲線(パラメータt)の場合、式(2)を修正した関数$e(t, R) = \mathrm{D}(t) - R^2$を利用する。

$$e(t, R) = \mathrm{D}(t) - R^2 = \Big(\sum_{k=0}^{n}(x_k - x_q)B_k^n(t)\Big)^2 + \Big(\sum_{k=0}^{n}(y_k - y_q)B_k^n(t)\Big)^2 - R^2 \qquad (4)$$

整理すると（ベジエ関数間の積の公式利用）

$$e(t, R) = \sum_{k=0}^{2n} d_k B_k^{2n}(t) \qquad (5)$$

上式の最小値が最短距離であるが、この高次式を精度よく解く必要があるのは距離が 0 に近い時のみである。制御点の凸包の性質から，（2n+1）個の制御点の最小値d_{min}で距離を代表できる。指定距離より小さい時は、精度よく計算するため、この関数を微分した関数 $e'(t) = \sum_{k=0}^{2n-1}(d_{k+1} - d_k)\, B_k^{2n-1}(t)$ （隣接した制御点の差分で構成）が 0 になる区間を求めこの区間で曲線をクリップし、距離の最小値を計算できる。この分割

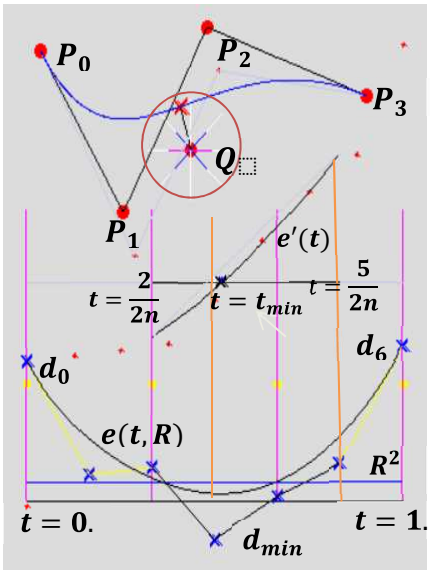

図 1 円と曲線の交差範囲及び最近点

された区間の関数は、区間が微小なら制御点と関数は誤差がなくなり、分割区間の制御点の最小値で十分である。図1に曲線からの距離関数$e(t,R)$および距離関数の最小値を求める関数$e'(t)$(区間はクリップされている)を示す。この距離関数の制御点の値も点Qと制御点P_iへのベクトル$(p_i = P_i - Q)$の内積の線形和で簡易に求まる。例えば次数n=3なら各制御点は下記のようである。

$$d_k=\{(p_0\cdot p_0),(p_0\cdot p_1),0.1(p_0\cdot p_2)+0.9(p_1\cdot p_1),0.4(p_0\cdot p_3)+0.6(p_1\cdot p_1),0.1(p_1\cdot p_3)+0.9(p_2\cdot p_2),(p_n\cdot p_{n-1}),(p_n\cdot p_n)\} \quad (6)$$

なお、図1は距離関数$e(t,R)$（半径Rの円からの距離）を示し、距離の最小値は関数の制御点d_kの最小値d_{min}（図中の×印）で近似できる、距離関数を微分した関数の制御点の凸包から、極値の存在区間を抽出し再帰計算で解に収束する、$d_{min} > R^2$ なら近傍でない曲線とみなせる。近傍の場合でも、第一段階では精密に区間抽出する必要ない。距離関数の制御点は2n＋1個、すなわちパラメータ区間は2n区間あるので、まずは解の無い区間がわかればいい。単に$(d_k > R^2, d_{k+1} > R^2)$である区間kは切り捨てることができる（図1では半分である区間[2/6,5/6]）。一般に距離関数は下に凸であるので、両端あるいは片端はクリップできる。クリップした区間をもとに、再帰的にクリッピングし、近傍円との交差の可能性、さらに最短距離を算出する。Bスプライン曲線(式7参照)については、ベジエ曲線に変換して、各ベジエ曲線に関して、近傍判定及び最短点検出を行えばいい。

$$\mathrm{P(t)} = \sum_{k=0}^{m} P_k N_k^n(t) \quad (7)$$

上式でN_k^nは次数のスプライン基底関数である。例えばm点の3次Bスプライン曲線の場合m－3個のベジエ曲線に分解できる。

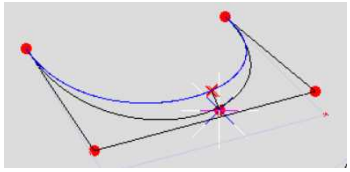

図2　ベジエ曲線上の最短点を利用した変形（星印がマウス）

（2）　曲線の編集

編集としては、①局所変形、②曲線の分割・切断、③傾き制御を考える。①；指定した点がマウスの移動に応じて変形させる。これは指定点に近い制御点(端点は除く)を動かす。ベジエ曲線、Bスプライン曲線それぞれ式（1）、（7）の点P_kの重み係数の大きい制御点を選択し、マウスの移動ベクトルに比例して移動させる（図2参照）。②；指定点のパラメータを用いて、ベジエ曲線の場合はドカステジョの方法で、Bスプライン曲線の場合De Boor-Coxの方法で曲線を分割する。これは、ノット挿入と等しい操作である。③；図3にベジエ曲線の編集を示す(曲線分割後、分割点Qの分割後の前後の制御点Q_LQ_Rを移動)。分割点は必ず曲線は通過する、分割した際の新たな制御点が指定点の前後に生じる。これらの3点を同一線にくるように制限して、制御点Q_Lを移動し傾きを制御する(分割後赤と黒の曲線に分割され、青曲線のように傾きが修正)。

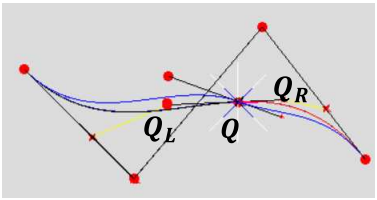

図3　指定点での分割・傾き編集

（3）　3次元曲線の処理

距離関数は2次元から3次元に容易に拡張できる。式(4)ではx、y成分で表現されているが、z成分を追加すればいい。また、式(6)での内積についても、z成分を付加すればいい。3次元表示の場合、制御点およびプローブは球で表現し、球とのレイトレーシングによりマウスで制御点を選択できるようにした(図5参照)。球のみでなくレイとの最短点も検出可能とした。またマウスの動きに応じスクリーン方向の移動、あるいはx、y、z成分を指定して移動させられる。

4. 計算例

基本的には携帯端末など種々のプラットホームで動作できるように、WebGl, JavaScriptで開発した。図4に2次元の3次Bスプライン曲線の場合の変形、分割の計算例を示す。図5に3次元のBスプライン曲線の場合の変形分割の計算例を示す(右の例は6点の3次Bスプライン曲線のハート形の回転体)。同図右の例のように、制御点・制御多角形が非表示の場合、特に最短点の移動は有効である。制御点の球の大きさで遠近感がわかり、かつプローブ球の底面メッシュへの最近点の表示で奥行きが把握できる。

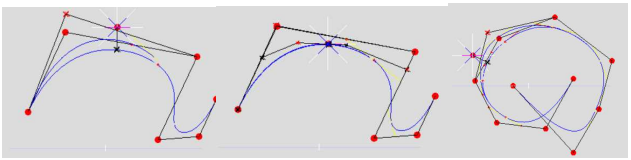

図4　2次元の3次Bスプライン曲線の変形分割（星状がマウス）

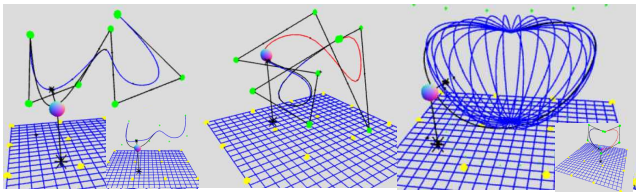

図5　3次元の3次Bスプライン曲線の変形・分割

5. おわりに

本稿では、2次元あるいは3次元の曲線（ベジエ、Bスプライン曲線）を、制御点に限らず曲線上をマウスで掴み、変形・分割・傾き修正を行う方法を提案した。曲線との近傍距離を計算する関数を提案し、この関数がベジエ関数であることを利用し、制御点およびBezier Clipping法を用い簡易に距離計算できる方法について提案した。

n次ベジェ曲線と距離判定関数では、次数がn次曲線なら、2n次のベジエ関数である。高次多項式を解くようであるが、幾何学的性質を用いた線形計算のみで実現できる特徴がある。なお、有理曲線の最短点検出法[4]は開発済なので、で今後NRUBSにも拡張したい。

参考文献

[1] T. Nishita, T. Sederberg, M. Kakimoto, "Ray Tracing Trimmed Rational Surface Patches," Computer Graphics, Vol.24, No.4, pp.337-345, 1990-8.
[2] T. Nishita and E. Nakamae, "A Shading Model for Atmosphere Scattering Considering Luminous Intensity Distribution of Light Sources," Computer Graphics, Vol.21, No.3,, pp.303-310. 1987-7
[3] 西田,出本,「曲面と多角形との最短距離検出法」Visual Computing / グラフィクスとCAD合同シンポジウム,11, 2017-6.
[4] 西田、「有理ベジエ曲面と球やメタボールの干渉計算」、芸術科学会論文誌 Vol.20, No.4, pp.204-209, 2021-11
[5] 西田、「レイマーチング法による円柱曲面・オフセット曲面の表示」, VC+VC202 2 , 19 (short) , 2022-10