Bull. Austral. Math. Soc. 65D07, 41A15
Vol. 38 (1988) [307–313]

CONTROL POINT INSERTION FOR B-SPLINE CURVES

Heinz H. Gonska and Andreas Röth

Inserting new knots into B-spline curves is a well-known technique in CAGD to gain extra flexibility for design purposes. However, from a user's point of view, the insertion of knots is somewhat unsatisfactory since the newly generated control points sometimes show up in unexpected locations. The aim of this note is to show that these problems can be circumvented by inserting the control vertices directly, thus also providing a more natural user interface.

1. Basics on B-spline Curves

In recent years, various algorithms have been designed to insert new knots into the knot vector upon which a (closed or open) B-spline curve of order k is based. As examples we mention the work of Cohen, Lyche, and Riesenfeld (the so-called Oslo algorithm, see [**5**, **7**]), Böhm's algorithm [**1**], and a paper by Böhm and Prautzsch [**2**].

For completeness, we mention the following basics. For the open case (and we will be dealing exclusively with this one in the present paper), one starts off with a *knot vector* $\mathbf{t} = (t_i)_{i=1}^{n+k}$, $n,\ k \in \mathbb{N}$. This is an ordered sequence of real numbers satisfying the additional condition

$$t_i < t_{i+k} \text{ for all } 1 \leqslant i,\ i+k \leqslant n+k.$$

Based upon the knot vector $\mathbf{t}$, a sequence of normalised B-splines $(B_{i,k,\mathbf{t}})$, $1 \leqslant i \leqslant n$, of order k corresponding to the knot vector $\mathbf{t}$ can be constructed along the lines given in de Boor's book [**3**]. The functions $B_{i,k,\mathbf{t}}$ exhibit some extraordinary properties which are of particular interest for CAGD purposes. If one makes the additional assumptions that

$$t_1 = \ldots = t_k \text{ and } t_{n+1} = \ldots = t_{n+k},$$

then the parametric curve

$$f : [t_k,\ t_{n+1}] \to \mathbb{R}^2$$

given by the equation

$$f(x) = \sum_{i=1}^{n} \mathbf{P}_i \cdot B_{i,k,\mathbf{t}}(x), \qquad x \in [t_k,\ t_{n+1}],$$

Received 18 December 1987

is the vector-valued version of the variation-diminishing spline operator which Schoenberg devised in 1965 as a generalisation of the famous Bernstein operators (see for example Marsden [8] and the references cited there). The curve f yields a graph in 2-space which depends upon the *control points* $\mathbf{P}_i \in \mathbb{R}^2$, $1 \leqslant i \leqslant n$. They in turn define a *control polygon* which we denote by $\mathbf{P} = \mathbf{P}_1\mathbf{P}_2 \dots \mathbf{P}_n$.

The curve described by f has some remarkable features as far as Computer Aided Design and Approximation Theory are concerned.

1. If all points $\mathbf{P}_i$, $1 \leqslant i \leqslant n$, lie on a straight line, then, for $x \in [t_k, t_{n+1}]$, $f(x)$ will also lie on that same straight line.
2. The points $\mathbf{P}_1$ and $\mathbf{P}_n$ are the endpoints of the curve.
3. The graph of $f(x)$, $x \in [t_k, t_{n+1}]$, lies in the convex hull of the points $\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_n$.
4. The curve $f(x)$ is variation-diminishing in the sense that any straight line in 2-space cuts the curve f no more often than it cuts the corresponding polygon $\mathbf{P}$.

A proof of properties 1, 2 and 3 is quite straightforward. An elegant proof of property 4 is obtained by combining results in papers of de Boor and DeVore [4] and of Goldman [6]. It is of particular interest that de Boor and DeVore demonstrated the theoretical relevance of Böhm's knot insertion algorithm [1] in quite an impressive manner. See [10] for a similar approach.

Such *knot insertion algorithms* are procedures doing the following job:

1. They refine the knot vector $\mathbf{t} = (t_i)_{i=1}^{n+k}$ to a knot vector $\hat{\mathbf{t}} = (\hat{t}_i)_{i=1}^{l+k}$ such that
   1.1. $\mathbf{t}$ is contained in $\hat{\mathbf{t}}$,
   1.2. all new knots are in the open interval (t_k, t_{n+1}),
   1.3. $\hat{t}_i < \hat{t}_{i+k}$ for $1 \leqslant i \leqslant l$.
2. They find the new B-splines $B_{i,k,\hat{\mathbf{t}}}$ corresponding to the new knot sequence $\hat{\mathbf{t}}$ and represent the original curve $f(x)$, $\hat{t}_1 = \dots = \hat{t}_k \leqslant x \leqslant \hat{t}_{l+1} = \dots = \hat{t}_{l+k}$, in the form

$$f(x) = \sum_{j=1}^{l} \mathbf{D}_j \cdot B_{j,k,\hat{\mathbf{t}}}(x),$$

   that is in terms of the 'new' B-splines corresponding to the 'new' knot sequence $\hat{\mathbf{t}}$ without changing the shape of the curve.

The result of such a procedure is that additional control vertices are generated which, together with some old ones to be retained, and others to be discarded, still generate the same curve as originally given, but with extra flexibility gained through the use of more control points.

While, from a theoretical point of view, the principle of knot insertion is very interesting, from the viewpoint of a practical designer this is definitely not the best way to generate extra control handles. Using knot insertion, the newly generated points $\mathbf{D}_j$ sometimes show up in quite unexpected locations. It is thus the aim of the present note to emphasise that it is also possible to insert control vertices directly (subject to some minor side conditions, if necessary or desired) rather than knots, and have the system automatically determine suitable knot locations (including multiplicities, if so desired). It is shown by an example at the end of this paper that the approach described here provides a much more user-friendly interface than the knot insertion approach.

2. Control Point Insertion

The control point insertion algorithm described below is based on Böhm's well-known algorithm for inserting new knots in B-spline curves.

Let $\mathbf{t} = (t_i)_{i=1}^{n+k}$ be the given vector of knots, and $\mathbf{P} = \mathbf{P}_1\mathbf{P}_2\ldots\mathbf{P}_n$ the control polygon. Given a new knot τ such that $t_{\nu-1} < \tau \leqslant t_\nu$, the new sequence $\hat{\mathbf{t}}$ is defined by

$$\hat{t}_i = \begin{cases} t_i,\ 1 \leqslant i \leqslant \nu - 1, \\ \tau,\ i = \nu, \\ t_{i-1},\ \nu + 1 \leqslant i \leqslant n + k + 1. \end{cases}$$

The new control points $\mathbf{D}_j$, $1 \leqslant j \leqslant n+1$, are given by

$$\mathbf{D}_j = \alpha_j \cdot \mathbf{P}_j + (1 - \alpha_j) \cdot \mathbf{P}_{j-1},$$

where

$$\alpha_j = \begin{cases} 1,\ j \leqslant \nu - k, \\ \dfrac{\hat{t}_\nu - \hat{t}_j}{\hat{t}_{j+k} - \hat{t}_j} \doteq \dfrac{\hat{t}_\nu - t_j}{t_{j+k-1} - t_j},\ \nu - k + 1 \leqslant j \leqslant \nu - 1, \\ 0,\ \nu \leqslant j. \end{cases}$$

Hence $\mathbf{D}_j$ subdivides the segment $\mathbf{P}_{j-1}\mathbf{P}_j$ at a ratio of $\alpha_j/(1-\alpha_j)$.

In other words, inserting a new knot τ makes $k-2$ old control vertices $\mathbf{P}_j$ vanish and yields $k-1$ new ones. Each new point $\mathbf{D}_j$ is a convex combination of $\mathbf{P}_j$ and $\mathbf{P}_{j-1}$, hence lies on the line segment joining these two points. This means that new control points can only be placed on the polygon $\mathbf{P}$.

Let us assume that we have picked a new control point $\widetilde{\mathbf{P}}$ on the polygon $\mathbf{P}$. Then our algorithm *first searches* $\mu \in \{2, \ldots, n\}$ such that $\widetilde{\mathbf{P}}$ lies on the segment joining $\mathbf{P}_\mu$ and $\mathbf{P}_{\mu-1}$, and $\alpha \in [0, 1]$ so that

$$\widetilde{\mathbf{P}} = \alpha \cdot \mathbf{P}_\mu + (1 - \alpha) \cdot \mathbf{P}_{\mu-1}.$$

Let us first consider the case where $\mathbf{P}_{\mu-1} \neq \widetilde{\mathbf{P}} \neq \mathbf{P}_\mu$, hence $\alpha \in (0,1)$. If we want $\widetilde{\mathbf{P}}$ to show up as a new control vertex after using Böhm's algorithm, we have to determine τ in the equation

$$\alpha = \alpha_\mu = \frac{\tau - t_\mu}{t_{\mu+k-1} - t_\mu}.$$

Solving for τ gives $\tau = \alpha(t_{\mu+k-1} - t_\mu) + t_\mu$. From $0 < \alpha < 1$ and $t_\mu < t_{\mu+k-1}$, it is clear that

$$\tau \in (t_\mu, t_{\mu+k-1}). \tag{2.1}$$

Next, we find ν so that

$$t_{\nu-1} < \tau \leqslant t_\nu,$$

and also the corresponding $\mathbf{D}_j$'s. In view of (2.1), we have

$$\mu < \nu \leqslant \mu + k - 1, \text{ or } \nu - k + 1 \leqslant \mu \leqslant \nu - 1.$$

Thus the new set of control vertices $\mathbf{D}_j$ is given by

$$\begin{aligned}
\mathbf{D}_1 &= \mathbf{P}_1, \\
&\dots \\
\mathbf{D}_{\nu-k} &= \mathbf{P}_{\nu-k}, \\
\mathbf{D}_{\nu-k+1} &= \alpha_{\nu-k+1} \cdot \mathbf{P}_{\nu-k+1} + (1 - \alpha_{\nu-k+1}) \cdot \mathbf{P}_{\nu-k}, \\
&\dots \\
\mathbf{D}_{\nu-1} &= \alpha_{\nu-1} \cdot \mathbf{P}_{\nu-1} + (1 - \alpha_{\nu-1}) \cdot \mathbf{P}_{\nu-2}, \\
\mathbf{D}_\nu &= \mathbf{P}_{\nu-1}, \\
&\dots \\
\mathbf{D}_{n+1} &= \mathbf{P}_n,
\end{aligned}$$

and in particular

$$\mathbf{D}_\mu = \alpha_\mu \cdot \mathbf{P}_\mu + (1 - \alpha_\mu) \cdot \mathbf{P}_{\mu-1}.$$

Hence $\mathbf{D}_\mu = \widetilde{\mathbf{P}}$, and the $\widetilde{\mathbf{P}}$ picked on the original polygon is a member of the new set of control vertices.

Example 2.1. Let $n = 7$, $k = 4$ and $t = (0,0,0,0,1,2,3,4,4,4,4)$. Then the control polygon is of the form $\mathbf{P} = \mathbf{P}_1 \dots \mathbf{P}_7$. Pick a new control point

$$\widetilde{\mathbf{P}} = \frac{1}{2} \cdot \mathbf{P}_3 + \frac{1}{2} \cdot \mathbf{P}_2.$$

The newly generated knot τ is then given by

$$\tau = (t_6 - t_3) \cdot \frac{1}{2} + t_3 = 1,$$

so that the new knot sequence is $\hat{\mathbf{t}} = (0,0,0,0,1,1,2,3,4,4,4,4)$. Hence the knot τ generated by control point insertion may coincide with one in the original knot sequence $\mathbf{t}$.

The possible coincidence observed in the last example may be an undesirable feature from the user's point of view since the degree of differentiability of the splines $B_{2,4,\hat{\mathbf{t}}}, \ldots, B_{5,4,\hat{\mathbf{t}}}$ is now only equal to one at $\tau = 1$ (in the definition of all splines mentioned $\tau = 1 = \hat{t}_5 = \hat{t}_6$ is used as a double knot). For the user this phenomenon is hardly predictable: he picks $\widetilde{\mathbf{P}}$, and α and τ are generated by the algorithm. However, during a typical interactive session, $\widetilde{\mathbf{P}}$ will be picked using a lightpen or a mouse or some other interactive media. Since this procedure is only accurate to a certain degree, it will be hardly noticed by the user (if at all) if we slightly perturb a critical value of α in the case that

$$\tau = t_j \text{ for some } j \in \{\mu + 1, \ldots, \mu - k + 2\}.$$

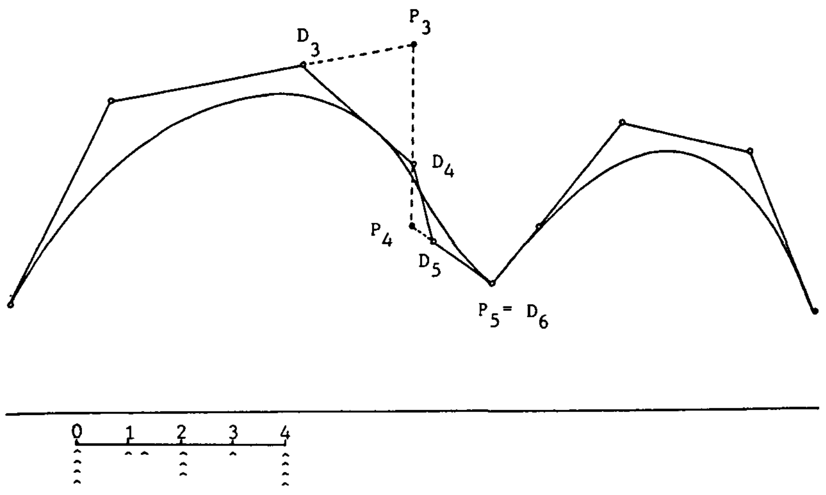
Although this triggers a minor shift of $\widetilde{\mathbf{P}}$, we did not observe any disadvantages.

Another question at hand is if control point insertion can be used to systematically generate multiple knots. This is indeed the case: If we allow $\widetilde{\mathbf{P}}$ to coincide with P_i, $i \in \{2, \ldots, n-k+1\}$, then any of the knots $t_{k+1}, \ldots, t_n$ can be given a multiplicity greater then one. We note here that the knots $t_1 = \ldots = t_k$ and $t_{n+1} = \ldots = t_{n+k}$ already have maximal multiplicity and thus should not be inserted again. The control point insertion algorithm can be easily modified to handle this case as well.

The example below is taken from a demonstration session using a prototype implementation of the above algorithm (Apple Macintosh/MacPascal).

Example 2.2. The original curve $(k = 4)$ was based on the knots $(0,0,0,0,1,2,2,2,3,4,4,4,4)$; note the non-differentiability of the curve at the point $\mathbf{P}_5$. Part of the control polygon of the original curve is indicated by the dotted line.

$\mathbf{D}_3$ was *inserted* as the new control point. Our system generated the additional knot $\tau = 1.3$, as well the new control points $\mathbf{D}_4$ and $\mathbf{D}_5$, all of them corresponding to the new knot sequence $(0,0,0,0,1,1.3,2,2,2,3,4,4,4,4)$. The points $\mathbf{P}_3$ and $\mathbf{P}_4$ (as well as the dotted portions in the above picture) constitute part of the original control polygon and would not be visible during further design of the curve. The same result would have been obtained by first "guessing" 1.3 as a new knot, and then inserting it using Böhm's algorithm.



References

[1] W. Böhm, 'Inserting new knots into B-spline curves', *Comput. Aided Design* **12** (1980), 199–201.

[2] W. Böhm and H. Prautzsch, 'The insertion algorithm', *Comput. Aided Design* **17** (1985), 58–59.

[3] C. de Boor, *A Practical Guide to Splines* (Springer, New York, Heidelberg, Berlin, 1978).

[4] C. de Boor and R. DeVore, 'A geometric proof of total positivity for spline interpolation', *Math. Comp.* **45** (1985), 497–504.

[5] E. Cohen, T. Lyche and R. Riesenfeld, 'Discrete B-splines and subdivision techniques in Computer-Aided Geometric Design and Computer Graphics', *Comput. Graphics Image Proc.* **14** (1980), 87–111.

[6] R.N. Goldman, 'Pólya's urn model and Computer-Aided Geometric Design', *SIAM J. Algebraic Discrete Methods* **6** (1985), 1–28.

[7] T. Lyche and K. Moerken, 'Making the Oslo algorithm more efficient', *SIAM J. Numer. Anal.* **23** (1986), 663–675.

[8] M.J. Marsden, 'On uniform spline approximation', *J. Approx. Theory* **6** (1972), 249–253.

[9] A. Röth, *Knoten- und Kontrollpunkteinsetzalgorithmen für B-Spline-Kurven im CAGD* (Diplomarbeit, Universität Duisburg, 1987).

[10] Jia-ye Wang, Guo-zhao Wang and Qun-sheng Peng, 'The geometric properties and order of approximation of parametric B-splines', (Chinese), *Chinese Ann. Math. Ser A* **5** (1984), 625–632.

Professor H.H. Gonska
Dept. of Mathematics and Computer Science
Drexel University
Philadelphia, PA 19104
United States of America
and
Fachbereich Mathematik
Universität Duisburg
D-4100 Duisburg 1
West Germany

Dipl.-Math. Andreas Röth
Fachbereich Mathematik
Universität Duisburg
D-4100 Duisburg 1
West Germany