Merging B-Sspline Curves or Surfaces Using Matrix Representation

Sz. Béla M. Szilvási-Nagy
Budapest University of Technology and Economics

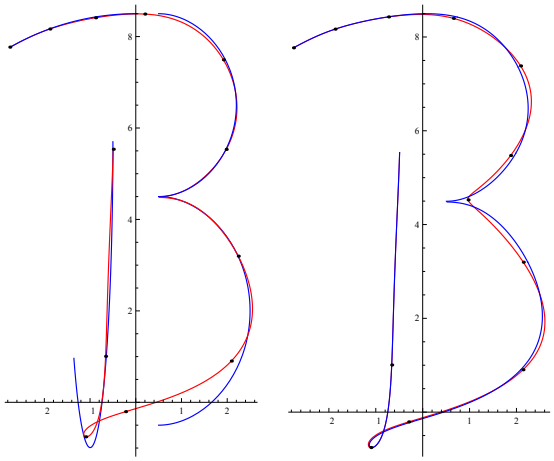


Figure 1. Merging of disjoint and C^0-continuous B-spline curves.

Abstract

This paper presents a simple and general technique to transform non-uniform B-Splines to the Bézier or to power-basis form. This conversion is very practical in representing spline curves or surfaces by explicit parametric vector functions in matrix form. We apply the proposed technique in a merging method of B-spline curve segments. The method merges B-spline curves iteratively with each other, and it is also extended to stitch B-spline surface patches even if they are overlapping each other or have possible gaps between them.

1. Introduction

In the 1970s—when the use of B-spline representation generally spread in curve and surface design—several algorithms were developed to derive the power basis or the Bézier representation of B-splines. First Cohen and Riesenfeld [1982] derived the matrix form of uniform B-splines. Then Choi et al. [1990] described the general matrix

 ISSN 2331-7418

representation of NURBS curves and surfaces of arbitrary degree. Later Grabowski et al. [1992] and Liu et al. [2002] presented analogous approaches. Recently Romani and Sabin [2004] proposed an algorithm for direct matrix transformation between uniform B-spline and Bézier representation. Later Casciola and Romani [2004] generalized the computations for non-uniform B-splines. The reason for the increased interest is that using the matrix representation of B-spline curves or surfaces provides fast and efficient computations in various B-spline algorithms.

This paper presents an alternative method to convert non-uniform B-splines of arbitrary degree to matrix representation form. Thanks to a convenient reformulation of the de Boor-Cox recursion, the conversion of the B-spline functions can be computed very simply and generally for arbitrary knot sequences.

Merging separately created B-Spline curves or surface patches is frequently used in geometric modeling. It is an important procedure if we would like to combine two or more given geometric models. Merging corresponding curve segments or surface patches into one reduces the size of the geometric data. Therefore, several different algorithms were developed to merge B-spline curves or surfaces ([Hu et al. 2002], [Tai et al. 2003], [Chen and Wang 2010], [Pungotra et al. 2010]). The algorithm developed by Hu et al. extends B-spline curves based on an unclamping method. Tai et al. present an approximate merging method of two adjacent B-spline curves applying perturbation on the control points of the curves. Chen et al. describe also an approximating merging method based on the minimization of the approximation error in the L_2-norm. Pugora et al. present a merging technique of B-spline surfaces. A stitching method was shown in Szilvasi et al. [2013] using interpolation and fairing for specific surface patches.

Our paper presents a merging procedure of B-spline curves that is formulated as a linear optimization problem. The matrix representation provides the description of the curves by explicit vector functions. This merging method is different from the previously published ones, since the objective function of the optimization process is a composed distance function including continuous L^2-norms of the functions and, also, pointwise derivatives. The matrix representation provides the efficiency and stability of the integral computation while minimizing the objective function. In our algorithm, the B-spline curve segments are joined one by one to the prior resulting curve, where the number of unknown control points is six in each step. In this way, the number of linear equations to be solved is independent of the number of curves to be merged. Moreover, in this iterative procedure the shape of the final merged curve can be adjusted piecewise by setting appropriate weights in the error function.

The paper is organized as follows. In the second section, a reformulation technique of the de Boor-Cox formula is presented to generate the matrix representation of the non-uniform spline basis. In the third section, a curve-merging algorithm is described for fourth-degree B-spline curves using the matrix representation form. We

also give a symbolic solution for merging uniform B-spline curves (see Appendix B). Our algorithm merges the input B-spline curves iteratively with each other. These curves may join in their end points, overlap or have gaps between them. Merging of B-spline surface patches are also shown using the developed curve-approximation method for their isoparametric curves. This technique can be applied to find the B-spline representation of the processed surface in milling, where the processed patches computed along the tool path [Szilvasi-Nagy et al. 2013] can be merged as shown in Example 5.

2. Matrix Representation of the Basis Functions

In order to represent the basis functions of non-uniform B-splines in matrix form, first we describe a reformulation technique for the de Boor-Cox recursion. From this recursion formula, we can generate the representation matrix of the basis functions in the non-normalized Bernstein basis. Then, we can apply a simple transformation from the non-normalized Bernstein basis either to the polynomial space spanned by the power basis or to the (normalized) Bernstein basis. Thus, with the algebraic reformulation of the B-spline recursion, we gain the conversion matrices of the B-spline functions to the Bernstein basis and also to the power basis.

2.1. Reformulation of the de Boor-Cox Algorithm

We present here a simple reformulation of the de Boor-Cox recursion in order to represent each segment of the basis functions over the unit interval. Basis functions of order k over the knot vector $\{t_1, \dots t_n\}$ are defined by the de Boor-Cox formula as

$$\begin{aligned} N_i^1(t) &= \begin{cases} 1, & t \in [t_i, t_{i+1}) \\ 0, & \text{otherwise,} \end{cases} \\ N_i^k(t) &= \alpha(t_i, t_{i+k-1}; t)\, N_i^{k-1}(t) + \alpha(t_{i+k}, t_{i+1}; t)\, N_{i+1}^{k-1}(t), \end{aligned}$$

where the function α is defined as

$$\alpha(A, B; t) = \frac{t - A}{B - A} \tag{1}$$

for arbitrary parameters A, B, where $A \neq B$, and for all $t \in \mathbb{R}$. If we would like to maintain the non-negativity of B-splines, here we have to restrict ourselves to $t \in [A, B]$ (or $t \in [B, A]$ when $A > B$). This function has several properties, which are

very advantageous in B-spline computations. The main properties are

$$\begin{aligned}
\alpha(A,B;A) &= 0, && (2)\\
\alpha(A,B;B) &= 1, && (3)\\
1-\alpha(A,B;t) &= \alpha(B,A;t), && (4)\\
\frac{1}{\alpha(A,B;t)} &= \alpha(A,t;B), \qquad \text{if } t \neq A && (5)\\
\alpha(A,B;t) &= \alpha(A,C;t)\cdot\alpha(A,B;C). && (6)
\end{aligned}$$

These properties can be easily derived from the definition of the function α (see Equation (1)). Using the properties (Equations (4) and (6)), a further identity can be derived:

$$\alpha(A,B;t) = \alpha(C,D;t)\alpha(A,B;D) + \alpha(D,C;t)\alpha(A,B;C) \tag{7}$$

for any $C \neq D$, real numbers. To keep the positivity of the basis functions these values are chosen later as $A \leq C < D \leq B$, if $A < B$, then $[C,D] \subseteq [A,B]$.

According to Equation (7), we can generate the pieces of the basis functions restricted to one knot interval $[t_j, t_{j+1})$ by rewriting the recursion as

$$\begin{aligned}
N_i^1(t) &= \begin{cases} 1, & t \in [t_i, t_{i+1}) \\ 0, & \text{otherwise,} \end{cases}\\
N_i^k(t) &= \alpha(t_j,t_{j+1};t)\left[\alpha(t_i,t_{i+k-1};t_{j+1})\,N_i^{k-1}(t)\right.\\
&\quad \left.+\,\alpha(t_{i+k},t_{i+1};t_{j+1})\,N_{i+1}^{k-1}(t)\right]\\
&\quad +\alpha(t_{j+1},t_j;t)\left[\alpha(t_i,t_{i+k-1};t_j)\,N_i^{k-1}(t)\right.\\
&\quad \left.+\,\alpha(t_{i+k},t_{i+1};t_j)\,N_{i+1}^{k-1}(t)\right],
\end{aligned} \tag{8}$$

where $t \in [t_j, t_{j+1})$. According to this form, we transform all segments of the basis functions from the knot span $[t_j, t_{j+1})$ and represent them over the unit interval as follows (see Figure 2):

$$\begin{aligned}
N_i^1(t(u)) &= \begin{cases} 1, & i = j \\ 0, & \text{otherwise,} \end{cases}\\
N_i^k(t(u)) &= u\left[\alpha(t_i,t_{i+k-1};t_{j+1})\,N_i^{k-1}(t(u))\right.\\
&\quad \left.+\,\alpha(t_{i+k},t_{i+1};t_{j+1})\,N_{i+1}^{k-1}(t(u))\right]\\
&\quad +(1-u)\left[\alpha(t_i,t_{i+k-1};t_j)\,N_i^{k-1}(t(u))\right.\\
&\quad \left.+\,\alpha(t_{i+k},t_{i+1};t_j)\,N_{i+1}^{k-1}(t(u))\right],
\end{aligned} \tag{9}$$

where $u \in [0,1)$, $t \in [t_j, t_{j+1})$ and $u(t) = \alpha(t_j, t_{j+1}, t)$.

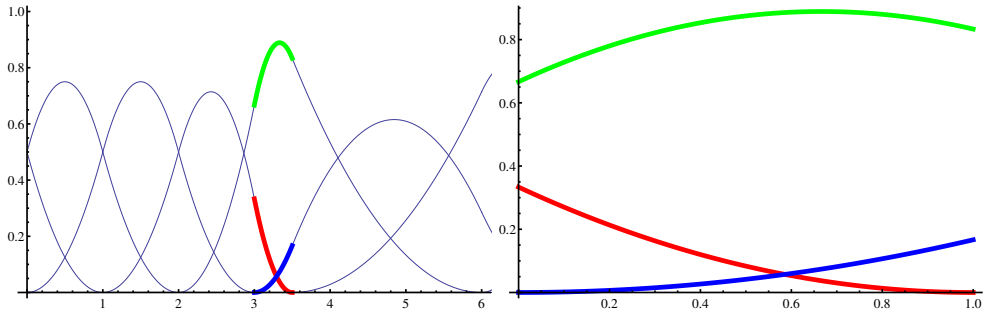


Figure 2. Quadratic basis functions are defined on the knot vector $\{-2,-1,0,1,2,3,3.5,6,7,8,9\}$ over the knot span [0,6] (left) and transformed to the unit interval from the knot span [3,3.5] using the rewritten recursion formula (9) (right).

2.2. Coefficient Matrices

In this section we generate the matrix representation of the basis functions of order k defined on an arbitrary knot vector $\{t_1, \dots t_n\}$. The basis functions are piecewise polynomials on each knot span $[t_j, t_{j+1})$. Over the knot spans, where $j = k, k+1, \dots, n-k$, the basis functions have k different, non-zero polynomial segments. These segments can be represented by a matrix equation in the non-normalized Bernstein basis $\{u^{k-1}, u^{k-2}(1-u), \dots, (1-u)^{k-1}\}$ over the unit interval:

$$\begin{pmatrix} N_1^k(t(u)) \\ N_2^k(t(u)) \\ \vdots \\ N_{n-k}^k(t(u)) \end{pmatrix} = \mathbf{C}^k \cdot \begin{pmatrix} u^{k-1} \\ u^{k-2}(1-u) \\ \vdots \\ (1-u)^{k-1} \end{pmatrix}, \qquad \begin{array}{l} t \in [t_j, t_{j+1}), \\ \\ u \in [0,1), \end{array} \tag{10}$$

where $\mathbf{C}^k \in \mathbb{R}^{n-k \times k}$. For each $k = 2, 3, \dots$, this matrix contains several rows, where all elements are zeros; those are the coefficients of the basis functions that are zero over the knot span $[t_j, t_{j+1})$. The non-zero rows for each j (from row $j+1-k$ to row j), where $j \geq k$, give the coefficients of the basis functions with the support containing the interval $[t_j, t_{j+1})$.

Algorithm 1 shows how to generate the coefficient matrix $\mathbf{C}^k$ of the equation system (10). The procedure is based on the reformulated de Boor-Cox recursion, Equation (9). It collects the coefficients of the functions $u^{k-i}(1-u)^{i-1}$ for all segments $i = 1, \dots k$ of the basis functions $N_p^k(u)$, $p = 1, \dots, n-k$ to the matrix $\mathbf{C}^k$.

If we represent the segments of the basis functions from a given knot span $[t_j, t_{j+1})$ in the matrix equation form (10), then it is easy to transform this representation to a matrix representation in the power basis, $\{u^{k-1}, u^{k-2}, \dots, u, 1\}$ and also in the Bernstein basis:

$$\left\{ \binom{k-1}{0} u^{k-1}, \binom{k-1}{1} u^{k-2}(1-u), \dots, \binom{k-1}{k-1} (1-u)^{k-1} \right\}.$$

Algorithm 1 `BSCoeffMatrix` $(k, \mathbf{v} = \{t_1, \ldots, t_n\}, j)$.

$s := 1$
$\alpha(A, B; t) := \dfrac{t - A}{B - A} \qquad A \neq B.$
$\mathbf{C}^1[p, 1] \Leftarrow \begin{cases} 1, & p = j \\ 0, & \text{otherwise} \end{cases}$ {initialization}
for $s = 2$ to k **do**
  $\widehat{\mathbf{C}}[p, i] \Leftarrow \begin{cases} \mathbf{C}^{s-1}[p, i], & 1 \leq p \leq n - k;\ 1 \leq i < s \\ 0, & \text{otherwise} \end{cases}$
  for $i = 1$ to s **do**
    for $p = 1$ to $n - k$ **do**
      $\mathbf{C}^s[p, i] := \alpha(t_i, t_{i+s-1}; t_{j+1}) \cdot \widehat{\mathbf{C}}[p, i]$
      $+\alpha(t_{i+s}, t_{i+1}; t_{j+1}) \cdot \widehat{\mathbf{C}}[p + 1, i]$
      $+\alpha(t_i, t_{i+s-1}; t_j) \cdot \widehat{\mathbf{C}}[p, i - 1]$
      $+\alpha(t_{i+s}, t_{i+1}; t_j) \cdot \widehat{\mathbf{C}}[p + 1, i - 1]$
      {see Eq. (9)}
    end for
  end for
end for
return $\mathbf{C}^k$

In order to find the transformation matrix of the basis functions to the power basis, it is sufficient to find the transformation matrix $\mathbf{P}^k$ from the non-normalized Bernstein basis to the power basis for polynomials of degree $k - 1$. Then, together with the transformation matrix $\mathbf{C}^k$ computed by Algorithm 1, the transformation matrix can be derived as the multiplication of the two matrices:

$$\begin{pmatrix} N_1^k(t(u)) \\ N_2^k(t(u)) \\ \vdots \\ N_{n-k}^k(t(u)) \end{pmatrix} = \mathbf{C}^k \cdot \mathbf{P}^k \cdot \begin{pmatrix} u^{k-1} \\ u^{k-2} \\ \vdots \\ 1 \end{pmatrix}, \qquad \begin{matrix} t \in [t_j, t_{j+1}), \\ \\ u \in [0, 1). \end{matrix} \tag{11}$$

The $\mathbf{P}^k$ matrix is a lower-triangular matrix with the elements

$$\mathbf{P}^k[i, l] = \begin{cases} (-1)^{i-l+1} \cdot \binom{i-1}{l-1}, & l \leq i, \\ \\ 0, & \text{otherwise}, \end{cases}$$

where l and $i = 1, \ldots, k$. This matrix can be easily derived from the following equa-

tion according to the binomial-theorem:

$$u^{k-i}(1-u)^{i-1} = u^{k-i} \cdot \sum_{l=1}^{i} \binom{i-1}{l-1} (-u)^{i-l}.$$

The basis functions can be transformed to the Bernstein basis with the help of a diagonal matrix $\mathbf{B}^k$, which contains the normalization constants of the basis functions $u^{k-i}(1-u)^i$:

$$\mathbf{B}^k[l,i] = \begin{cases} \dfrac{1}{\binom{k-1}{i-1}}, & i = l, \\ \\ 0, & \text{otherwise}, \end{cases}$$

where l and $i = 1, \ldots, k$. Thus all segments of the basis functions from a given knot span $[t_j, t_{j+1})$ can be represented in the matrix equation form

$$\begin{pmatrix} N_1^k(t) \\ N_2^k(t) \\ \vdots \\ N_{n-k}^k(t) \end{pmatrix} = \mathbf{C}^k \cdot \mathbf{B}^k \cdot \begin{pmatrix} \binom{k-1}{0} u^{k-1} \\ \binom{k-1}{1} u^{k-2}(1-u) \\ \vdots \\ \binom{k-1}{k-1} (1-u)^{k-1} \end{pmatrix}, \tag{12}$$

where $t \in [t_j, t_{j+1})$ and $u \in [0, 1)$. The conversion matrix from the (normalized) Bernstein basis to the power basis can be also found; see, for example, [Farin 1990].

3. Application of the Matrix Form to Merge B-spline Curves and Surfaces

Here we present an iterative algorithm that merges adjacent, separate, or overlapping B-spline curve segments of degree four. The input segments are merged one by one to the output curve. The matrix representation form of the input curves allows us to compute the approximating output curve with continuous approximation condition along the corresponding curve segments and pointwise difference of the tangent vectors. These conditions are included into an objective function that is quadratic in the required control points. Therefore, the optimization leads to a system of linear equations whose solution is the unknown control points of the output curve. They can be expressed as the linear combinations of the control points of the input curves. Moreover, the matrix form provides that the geometric properties of two corresponding curves can be described and compared without reparametrization, since all arcs of the B-spline curves are parametrized over the unit interval.

First we present how to merge B-spline curves with non-uniform parametrization. In Section 3.2 we show the merging process in the uniform case. Then, in Section 3.3,

we show examples and analyze the structure of the objective function and the effect of different knot vectors.

The algorithm can be generalized for higher-degree curves, and it is possible to extend this curve-merging algorithm to stitch surface patches together by merging the corresponding isoparametric curves.

3.1. The Iterative Merging Process in the Non-uniform Case

In the non-uniform case, the shape of the basis functions depend on the knot values, and they have to be determined over each knot interval individually for each B-spline curve.

The vector equation of one segment of a fourth-degree B-spline curve is

$$\mathbf{r}(t) = \sum_{i=0}^{4} \mathbf{b}_i N_i(t), \quad t \in [t_4, t_5], \tag{13}$$

where $\mathbf{b}_i \in \mathbb{R}^3$, $i = 0 \dots 4$ are the control points of the curve and $N_i(t)$, $i = 0 \dots 4$ are the basis functions over the periodic knot vector $t_0 < t_1 < \cdots < t_9$.

A composite B-spline curve consisting of s segments over the parameter interval $[t_4, t_{4+s}]$ is determined by $s + 4$ control points, $\mathbf{b}_0$, $\mathbf{b}_1, \dots, \mathbf{b}_{s+3}$, and by the basis functions $N_i(t)$, $i = 0, \dots, s+3$ over the periodic knot vector $t_0 < t_1 < \cdots < t_{s+8}$. The jth curve segment over the parameter interval $[t_{j+3}, t_{j+4}]$ is represented by the vector function

$$\mathbf{r}_j(t) = \sum_{k=0}^{4} \mathbf{b}_{j+k-1} N_{j+k-1}(t), \quad j = 1, \dots, s. \tag{14}$$

Our merging strategy is to join the curve segments defined separately one by one.

(1) Specifying the input data.

We suppose that the first curve $\mathbf{r}_1(t)$ has $s \geq 4$ segments $\mathbf{r}_{1,1}(t), \dots, \mathbf{r}_{1,s}(t)$ with the control points $\mathbf{p}_{1,0}, \dots, \mathbf{p}_{1,n-1}$, where $n = s + 4$. For the case $s < 4$, we refer to Appendix C. A stitching method was also shown for this case in [Szilvási-Nagy and Béla 2013]. The next curve segment represented by $\mathbf{r}_2(t)$ with the control points $\mathbf{p}_{2,0}, \dots, \mathbf{p}_{2,4}$ will be joined to $\mathbf{r}_1(t)$ (see Figure 3).

A simple algorithm for generating the B-spline representation of one curve segment is shown in Appendix A.

We reparametrize each segment of $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$ over the unit interval and use the matrix representation of the B-spline functions expressed by Equation (11). This simplifies the computations, so $\mathbf{r}_{1,i}(u)$ and $\mathbf{r}_2(u)$ are the input forms of the given curves, where $0 \leq u \leq 1$ and $i = 1, \dots, s$.

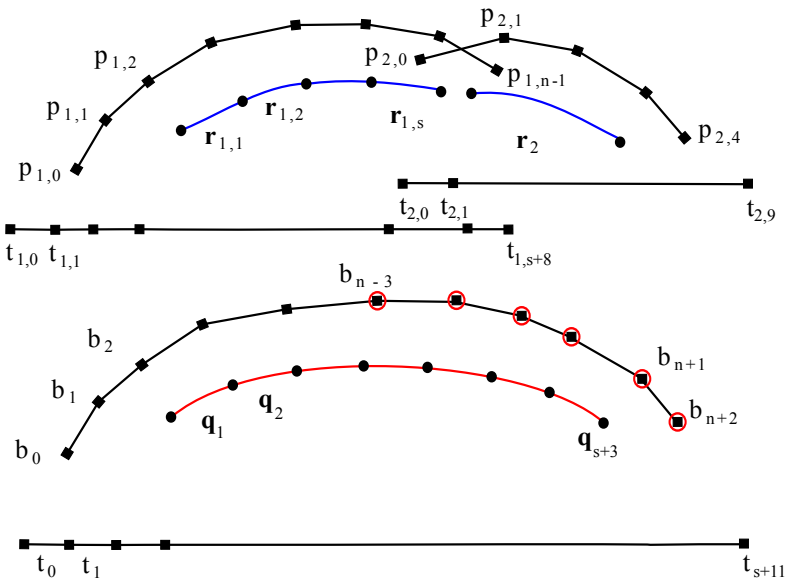


Figure 3. Input curves $\mathbf{r}_{1,i}$ $(i = 1, \ldots, s)$, $\mathbf{r}_2$ and the required approximating curve $\mathbf{q}_j$ $(j = 1, \ldots, s + 3)$ with control points.

(2) Dividing $\mathbf{r}_2$ into three parts.

In order to have enough control points of the new merged curve sufficiently approximating the input curves, the curve segment $\mathbf{r}_2(u)$ will be divided into three parts, for example, by parameter transformations $\mathbf{r}_{2,1}(u) = \mathbf{r}_2(u/3)$, $\mathbf{r}_{2,2}(u) = \mathbf{r}_2(1/3 + u/3)$, $\mathbf{r}_{2,3}(u) = \mathbf{r}_2(2/3 + u/3)$, $u \in [0, 1]$.

(3) Generating the B-spline basis functions for the vector function representing the required output curve.

First we set the parameter intervals (the knot vector) of the required approximating curve $\mathbf{q}(t)$. It will have $s+3$ segments and is defined with the periodic knot vector $t_0 < t_1 < \cdots < t_{s+11}$. The knot intervals are computed proportional to chord lengths or arc lengths of the curve segments $\mathbf{r}_{1,i}(u)$, $i = 1, \ldots, s$ and $\mathbf{r}_{2,j}(u)$, $j = 1, \ldots, 3$ composing the required $\mathbf{q}(t)$. Then, the basis functions are computed by the formula in Equation (11).

(4) Setting the control points of the required curve.

The control points of the new curve are $\mathbf{b}_i$, $i = 0, \ldots, n + 2$, from which $\mathbf{b}_i = \mathbf{p}_{1,i}$, $i = 0, 1, \ldots, n - 4$, and the last six will be computed from the approximation condition.

(5) Computing the vector functions of the new curve segments.

The vector functions of the new curve segments $\mathbf{q}_i(u)$, $i = 1, \ldots, s+3$ are computed by the the corresponding coefficient matrices (11) over the parameter interval $0 \leq u \leq 1$. The last six control points are unknown.

(6) Setting the objective function and computing the unknown control points.

The approximation condition is to minimize the difference between the input and the merged curves expressed by the target function

$$\begin{aligned} T(\mathbf{b}_{n-3}, \mathbf{b}_{n-2}, \ldots, \mathbf{b}_{n+2}) = & \sum_{i=s-2}^{s} \int_0^1 |\mathbf{r}_{1,i}(u) - \mathbf{q}_i(u)|^2 du \\ & + \sum_{i=1}^{3} \int_0^1 |\mathbf{r}_{2,i}(u) - \mathbf{q}_{s+i}(u)|^2 du \\ + a \cdot \left(\sum_{i=s-2}^{s} |\dot{\mathbf{r}}_{1,i}(0) - \dot{\mathbf{q}}_i(0)|^2 + \sum_{i=1}^{3} |\dot{\mathbf{r}}_{2,i}(0) - \dot{\mathbf{q}}_{s+i}(0)|^2 \right), & \end{aligned} \tag{15}$$

where $\dot{\mathbf{r}}_{k,i}(u) = d\mathbf{r}_{k,i}(u)/du$ denotes the derivative of the function $\mathbf{r}_{k,i}(u)$, $k = 1, 2$. The squared distance between the new and the input curves in the first two terms are computed by numerical integration. Experiments have shown that the squared distances between the first derivatives in the third term have to be included into the target function, where the factor $a = 0.3$ led to the best result in many cases. The target function is a quadratic polynomial of the unknown six control points. The integration can be carried out directly on the functions with their explicit matrix representation. The minimization leads to a system of six linear equations. The solution together with the unchanged control points determines the control polygon of the merged curve with the vertices $\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{n+2}$.

The steps (1)–(6) have to be repeated with each new curve segment that is to be merged to the last resulting curve. At each merging, the number of the curve segments and the control points grows, $s = s + 3$ and $n = n + 3$.

3.2. Symbolic Solution in the Uniform Case

In the case of uniform knot vectors, all the basis vectors have constant coefficients in the matrix $\mathbf{M}$. The basis functions parametrized over the unit interval are

$$(N_0(t)\, N_1(t)\, N_2(t)\, N_3(t)\, N_4(t)) = (t^4\ t^3\ t^2\ t\ 1) \cdot \mathbf{M}, \tag{16}$$

$$\text{where} \quad 0 \le t \le 1 \quad \text{and} \quad \mathbf{M} = \frac{1}{24}\begin{pmatrix} 1 & -4 & 6 & -4 & 1 \\ -4 & 12 & -12 & 4 & 0 \\ 6 & -6 & -6 & 6 & 0 \\ -4 & -12 & 12 & 4 & 0 \\ 1 & 11 & 11 & 1 & 0 \end{pmatrix}.$$

In this case the computation of the merged new curve is shorter, though the algorithm is the same as in the non-uniform case.

(1) Specifying the input data.

The vector functions of the segments of the input curves are computed by the coefficient matrix $\mathbf{M}$ in Equation (16).

(2) Setting the control points of the required approximating curve.

From the $n+3$ control points of the required curve, the first $n-4$ are taken from those of the input curve $\mathbf{r}_1(t)$, i.e., $\mathbf{b}_i = \mathbf{p}_{1,i}$, $i = 0, 1, \ldots, n-4$. Then the remaining six points can be precomputed symbolically. Therefore, minimization of the target function will not be carried out in each merging process. The minimization of the target function, Equation (15), leads to a system of six linear equations. This can be solved symbolically for the six unknown control points. These new control points are expressed by linear combinations of the control points of the input curves, $\mathbf{p}_{1,n-6}, \mathbf{p}_{1,n-5}, \ldots, \mathbf{p}_{1,n-1}$ and $\mathbf{p}_{2,0}, \ldots, \mathbf{p}_{2,4}$ (see Appendix B), because each control point has the effect on five curve segments. This symbolic solution has the advantage that the pre-computed control points can be applied in each merging step in such a way that the control points of the actual input curves are substituted into the formula.

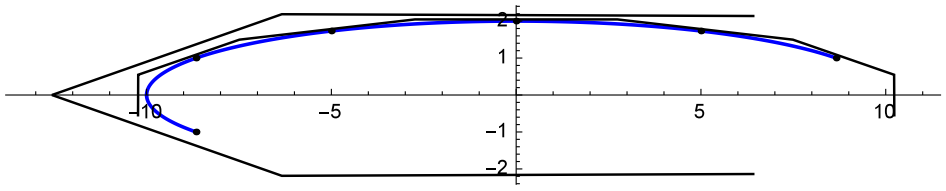
The examples show that the symbolic solution in the uniform case resulted in a satisfying curve, if the difference in the curvatures of the input curves is not "very large."

3.3. Examples

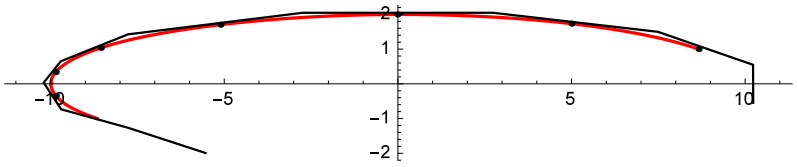
In Example 1 (see Figure 4), six B-spline curves are merged into one B-spline curve that approximates an ellipse. The ellipse has the equation

$$\mathbf{r}(t) = \mathbf{i}\, 10\cos(t) + \mathbf{j}\, 2\sin(t).$$

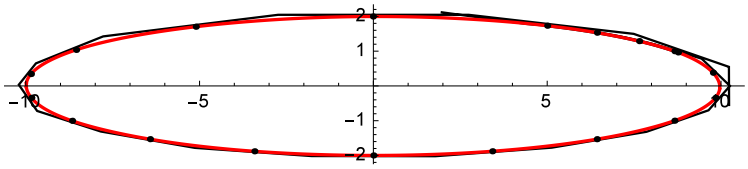
The first arc consists of four segments in $t \in [\frac{\pi}{6}, \frac{5\pi}{6}]$, arcs 2–5 are defined over the parameter intervals of the length $\frac{\pi}{3}$, the last arc defined for $t \in [\frac{\pi}{6}, \frac{2\pi}{6}]$ repeats the first segment of the first arc. In Figure 4 (a), the first and second arcs are shown with their control polygons. In Figure 4 (b), the merged curve consisting of $4+3$ segments



(a) Two input B-spline curves approximating the arcs of an ellipse with their control polygons. The first curve consists of four segments, the second one has one segment generated by the longer control polygon.



(b) Merged curve of the B-spline curves shown in (a) with the new control polygon.



(c) B-spline curve approximating an ellipse generated by merging four additional curve segments to the curve shown in (b). Each new input segment is approximated with three B-spline segments in the resulting curve.

Figure 4. Example 1.

is shown with the computed control polygon. Five more curve segments have been merged to the first four curve segments, each divided into three parts (black points in Figure 4 (c)). At the end of the process, the B-spline curve has 19 segments (see Figure 4 (c)).

The relative errors with respect to the arc lengths of the merging is measured by the integral of the squared distances between the new and the input curves divided by the arc lengths, computed for each pair of corresponding curve segments. In the columns of Table 1, the relative errors in the last three merging steps are shown for the segments influenced by the six control points computed. The error is maximal at the joining points (e.g., 0.0044 in the last step of Example 1), and it is zero, where the computed new control points have no effect on the curve segment. This example shows that the error is typically larger where the average curvature of the adjacent input curves are significantly different. This is the case in steps 2 and 5, where one segment of the ellipse with high curvature and a flat elliptic arc have been merged.

Seg. No.	Error of Step (3)	Error of Step (4)	Error of Step (5)
1–7	0	0	0
8	$5.1 \cdot 10^{-10}$	0	0
9	$1.3 \cdot 10^{-7}$	0	0
10	$1.7 \cdot 10^{-7}$	0	0
11	$7.1 \cdot 10^{-8}$	$1.1 \cdot 10^{-10}$	0
12	$1.8 \cdot 10^{-8}$	$4.3 \cdot 10^{-8}$	0
13	$4.8 \cdot 10^{-9}$	$7.4 \cdot 10^{-8}$	0
14	-	$2.2 \cdot 10^{-8}$	$3.9 \cdot 10^{-6}$
15	-	$5.0 \cdot 10^{-9}$	$2.7 \cdot 10^{-3}$
16	-	$7.2 \cdot 10^{-10}$	$4.4 \cdot 10^{-3}$
17	-	-	$1.9 \cdot 10^{-3}$
18	-	-	10^{-4}
19	-	-	$2 \cdot 10^{-5}$

Table 1. Relative errors in merging of one curve segment in the last three steps (Example 1 with uniform knot vector)

The arc length parametrization resulted in a solution which approximates the input curve segments in the last merging step with the maximal error of $5.9 \cdot 10^{-4}$ (instead of $4.4 \cdot 10^{-3}$). The relative errors are shown in Table 2 for each curve segment computed in the last (5th) merging step.

Seg. No.	Error (uniform)	Error (chord length)	Error (arc length)
14	$3.9 \cdot 10^{-6}$	$2 \cdot 10^{-5}$	$1.9 \cdot 10^{-6}$
15	$2.7 \cdot 10^{-3}$	$5.4 \cdot 10^{-3}$	$4.4 \cdot 10^{-4}$
16	$4.4 \cdot 10^{-3}$	$1.2 \cdot 10^{-2}$	$5.9 \cdot 10^{-4}$
17	$1.9 \cdot 10^{-3}$	$2.4 \cdot 10^{-4}$	$2.1 \cdot 10^{-4}$
18	10^{-4}	10^{-4}	$8.6 \cdot 10^{-5}$
19	$2 \cdot 10^{-5}$	$4.2 \cdot 10^{-5}$	$2.6 \cdot 10^{-5}$

Table 2. Relative errors in merging of the last curve segment (Example 1 with non-uniform knot vector)

In Example 2 (see Figure 5), a wavy curve is generated from approximately circular and elliptical arcs. Two merging steps have been computed in arc-length parametrization. In the left part of Figure 5, the differences in the curvatures are smaller; in the right part of the figure they are larger. The error of the merging is hardly visible in the first case. The maximal relative errors with respect to the arc length are $1.6 \cdot 10^{-5}$ and $1.5 \cdot 10^{-4}$, respectively.

The parts of the wavy curve on the right side of Figure 5 have been merged also with uniform parametrization using the symbolic solution. The maximal relative error

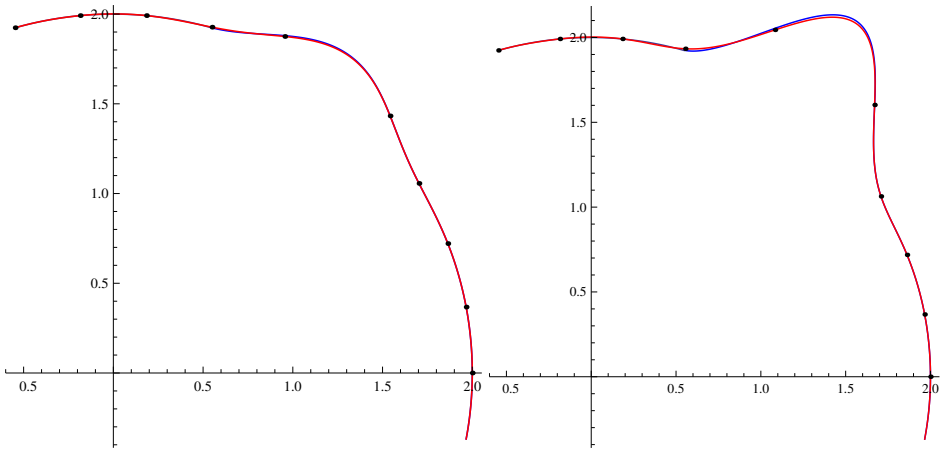


Figure 5. B-spline curves merged from three input curves. The first input curve consists of four segments, both the second and third have one segment each, which are divided into three parts during the merging process (Example 2). In the figure on the left, the second input curve segment has smaller curvature than the one in the figure on the right.

was 0.0025. This resulting curve is not shown, because the difference from the first solution is not visible in the figure of this size.

Since a larger error occurs when the difference in the average curvatures of the adjacent curves are larger, we tried to apply also a curvature-adjusted parametrization of Patterson [1989]. Unfortunately, the numerical error of the computation with the rational formulas (instead of polynomials) led to larger merging errors than in the arc-length parametrization. So we are not able to conclude, whether the result could be better theoretically or practically.

Remark. The computations with three different knot vectors (uniform, chord length, and arc length) show that the error is maximal at the joining point, and the arc-length parametrization gives the best result. Calculating with uniform knots has the advantage that the precomputed formula of the symbolic solution gives the new control points directly in each merging step.

Remark. During the iterative merging method, the number of unknowns is constant in each iteration step. Moreover, in each step, the approximation error occurs only along the last three segments of the output curve of the former iteration step and along the merged input segment (gives three new segments to the output curve). Thus, the error does not accumulate with the growing number of merged curve segments.

3.4. A Smoothing Term in the Objective Function

The objective function can be extended by further terms expressed with the second derivatives. Such a function is typically used for smoothing the resulting curve by

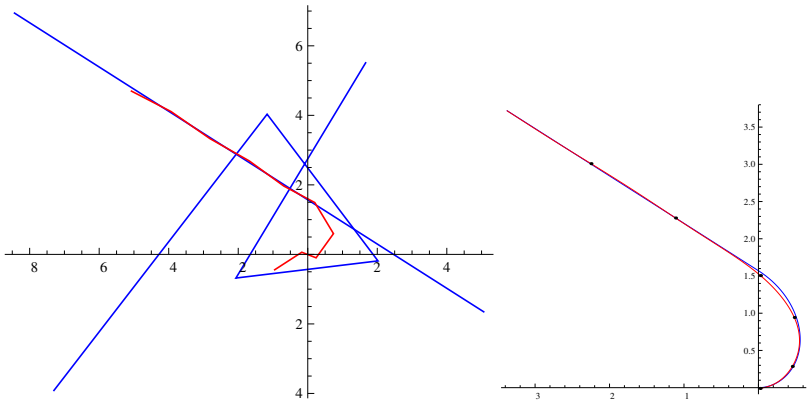


Figure 6. On the left, the control polygons of the input curves are shown in blue, and that of the merged B-spline curve in red. On the right, the corresponding curves are shown (Example 3). The resulting control polygon is not straight for the last straight-line segment.

reducing the variation of curvature. Consequently, the approximation error grows for wavy curves. If one of the curves to be merged is a straight-line segment, then the objective function will be completed by a smoothing term computed by the second derivative of the resulting curve, that is,

$$\sum_i \int_0^1 |\ddot{\mathbf{q}}_i(u)|^2 du,$$

where i is the index of segments that approximates the straight line. This smoothing function is also quadratic in the unknown six control points. Therefore, a system of six linear vector equations has to be solved in this case too.

In Example 3 (see Figure 6), a curve of spiral form and a straight-line segment are given. The merging with the help of the objective function in Equation (15) using arc-length parametrization led to the solution in Figure 6. Computing with the smoothing term, the resulting curve fit the straight line segment more exactly (Figure 7). The maximal error is reduced from 10^{-3} to 10^{-12}.

Another case, where the smoothing term provides a better solution is, when the given curves do not join exactly, but there are gaps between them. Since there is nothing to approximate in a gap, the resulting curve of the merging process should be as smooth as possible.

In Example 4 (see Figure 8), the parts of the letter B are given, such that a gap is between them. The solutions are smooth B-spline curves joining the two given curves on the upper- and the lower-part separately. Then these two curves are merged into a single B-spline curve.

Remark. Our merging algorithm works well also with overlapping input curves. An example is shown in Figure 9.

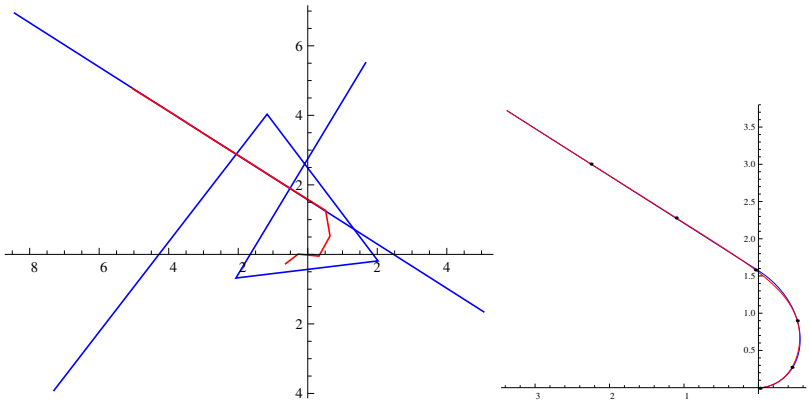


Figure 7. The same inputs as in Figure 6, the result has been computed with smoothing term in the objective function (Example 3). The second part of the resulting control polygon is straight.

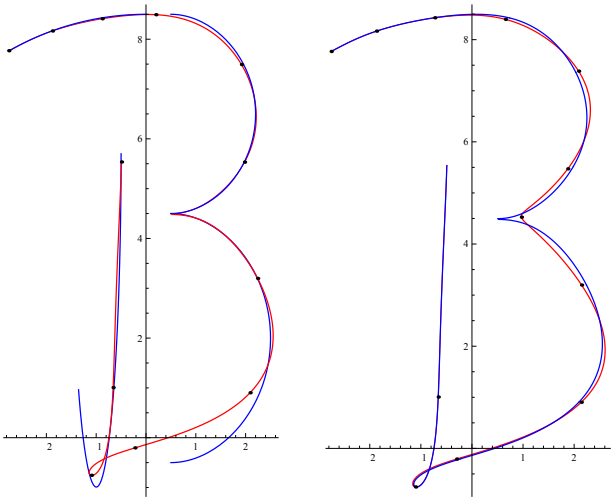


Figure 8. On the left two curves are merged in the upper and lower parts; on the right the result is a single B-spline curve (Example 4).

Remark. In the objective function, constant weights can be added to the terms in order to influence some segments of the resulting curve. No general conclusions can be observed from these experiments.

3.5. Merging B-spline Surface Patches

In applications, surface patches may be generated separately; for example, in milling, the processed surface is represented by a set of patches generated by the milling tool along the tool path. If these patches are represented by B-spline vector functions, then the B-spline representation of the processed surface can be computed row by row along each tool path.

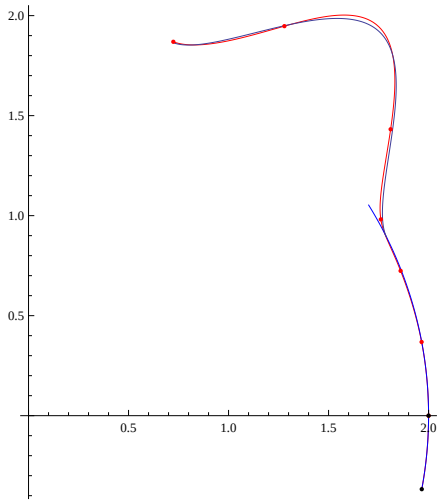


Figure 9. Merging overlapping curve segments.

The vector equation of a B-spline surface of degree 4 determined by the control points $\mathbf{b}_{ij},\ i = 0,\ldots,n,\ j = 0,\ldots,m$ and by the periodic knot vectors $u_0 < u_1 < \ldots, < u_{n+4},\ v_0 < v_1 < \cdots < v_{m+4}$ is given for each patch by

$$\mathbf{r}_{ij}(u,v) = \begin{pmatrix} u^4 & u^3 & u^2 & u & 1 \end{pmatrix} \cdot \mathbf{N}_i \cdot \mathbf{B} \cdot \mathbf{M}_j^{\mathrm{T}} \cdot \begin{pmatrix} v^4 \\ v^3 \\ v^2 \\ v \\ 1 \end{pmatrix},$$

$$u \in \Delta u_i,\ v \in \Delta v_j, \quad i = 1,\ldots,n-3,\ j = 1,\ldots m-3,$$

where $\mathbf{N}_i$ and $\mathbf{M}_j$ are the coefficient matrices of the basis functions over the u-knots and v-knots, respectively, and T in the exponent denotes the transpose of the matrix. The matrix $\mathbf{B}$ consists of the 5×5 control points of the surface patch $\mathbf{r}_{ij}(u,v)$.

The merging process is carried out on the control nets, row by row, using the method shown for curves, first in the u-parameter direction for each fixed v-parameter interval. The result of this merging process is a surface stripe of one segment width in the v-direction. These stripes are then merged in the v-parameter direction, computing with the columns of the control nets. The result is independent of the choice in which parameter direction we start the merging process. In Figure 10, a helical surface is approximated by 6×2 input patches, and these patches are merged into a B-spline surface. The resulting surface is shown with the boundary curves of the input patches. The magnitudes of the relative errors in this merging are the same as in the curve merging. In Figure 11, the input patches are given with gaps between them, the resulting B-spline surface fills the gaps, and it is C^3-continuous everywhere. A synthetic example for merging overlapping surface patches like in milling is shown in

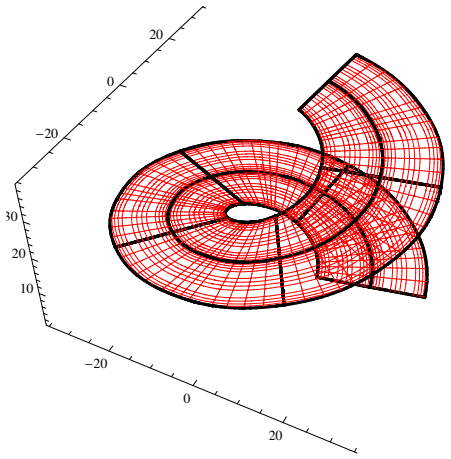


Figure 10. Patches approximating a helical surface are shown with their boundary curves merged into a B-spline surface.

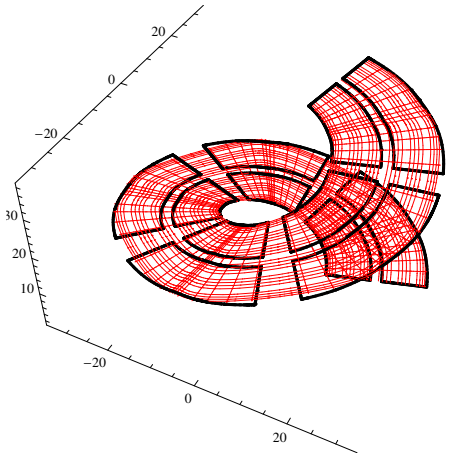


Figure 11. Gaps are between the given patches, the resulting B-spline surface approximates the input patches.

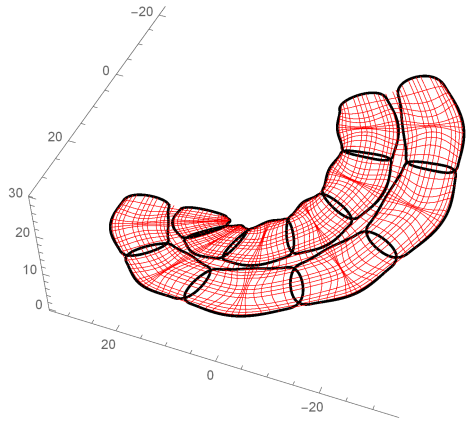


Figure 12. B-spline surface approximates overlapping input patches.

Figure 12. Further dynamic figures of the surface merging can be found at `http://www.math.bme.hu/~kongeo/JCGT-paper/surface-merging.html`.

4. Conclusions

We have presented a new, elegant reformulation of the de Boor-Cox recursion and applied it to compute the matrix representation of B-splines over arbitrary knot vectors. Using the matrix form of the basis functions, we give an approximation method to merge B-spline curves and surfaces of degree four. By using periodic knot vectors, we avoid the investigations of the continuity of the derivatives at the joining points, which usually leads to inconvenient computations with additional conditions. In our merging method, the resulting fourth-degree B-spline curves are C^3 continuous, assuming that all the knot values have multiplicity 1. Our iterative merging leads to a system of six linear vector equations in each merging step while minimizing an objective function (which is, in fact, a composed distance function) with six unknown control points independently of the number of curve segments. This computation does not require non-linear optimization, and it is less time-consuming and more stable than the computation of all the control points in one step with a large number of input curve segments shown in [Chen and Wang 2010]. Moreover, the matrix representation provides us fast and efficient computations during the merging procedure. It allows us to include the continuous distance of the functions and the pointwise derivatives in the objective function. We have developed a symbolic merging solution with uniform knot vectors for the case when the arc lengths and curvatures of the input curve segments do not vary very much. We have investigated also the effect of the structure of the objective function by including first and second derivatives of the vector functions describing the curves. Our experience probably helps to choose appropriate knot vectors and objective functions in the solution of similar problems. The computations and the figures (except Figure 3) were made by the symbolic algebraic program package, Mathematica.

Acknowledgements

The second author is supported by a joint project between the TU Berlin and the BUTE.

A. Algorithm for the Symbolic Computation of the B-spline Representation of One Curve Segment

For the generation of one B-spline segment of degree four over the parameter interval $[0, 1]$, we take three interpolation points, $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_m$ and two tangent vectors, $\mathbf{t}_0$, and $\mathbf{t}_1$ of the given curve. We then compute the control points by solving the system of five linear equations

$$\mathbf{r}(0) = \mathbf{p}_0,\ \mathbf{r}(1) = \mathbf{p}_1,\ \mathbf{r}(0.5) = \mathbf{p}_m,\ \left.\frac{d\mathbf{r}}{du}\right|_{u=0} = \mathbf{t}_0,\ \left.\frac{d\mathbf{r}}{du}\right|_{u=1} = \mathbf{t}_1.$$

In the uniform case, the solution, i.e., the five control points, is expressed symbolically by the linear combination of the input data. The relative error of this interpolation for a circular arc with $\text{radius} = 1$ and $\text{central angle} = \frac{\pi}{3}$, measured by the integral $\int_0^1 |\mathbf{r}(t) - \mathbf{c}(t)|^2 \, dt$ is $\approx 10^{-16}$, where $\mathbf{r}(t)$ represents the computed B-spline curve and $\mathbf{c}(t)$ represents the circular arc. This excellent result was the reason we started to work with fourth-degree B-splines.

The resulting control points are

$$\begin{aligned}
\mathbf{b}_0 &= 0. - 30.1667\mathbf{p}_1 - 46.1667\mathbf{p}_0 + 77.3333\mathbf{p_m} + 6.33333\mathbf{t}_1 - 16.3333\mathbf{t}_0 \\
\mathbf{b}_1 &= 7.83333\mathbf{p}_1 + 11.8333\mathbf{p}_0 - 18.6667\mathbf{p_m} - 1.66667\mathbf{t}_1 + 2.66667\mathbf{t}_0, \\
\mathbf{b}_2 &= -6.16667\mathbf{p}_1 - 6.16667\mathbf{p}_0 + 13.3333\mathbf{p_m} + 1.33333\mathbf{t}_1 - 1.33333\mathbf{t}_0, \\
\mathbf{b}_3 &= 11.8333\mathbf{p}_1 + 7.83333\mathbf{p}_0 - 18.6667\mathbf{p_m} - 2.66667\mathbf{t}_1 + 1.66667\mathbf{t}_0, \\
\mathbf{b}_4 &= -46.1667\mathbf{p}_1 - 30.1667\mathbf{p}_0 + 77.3333\mathbf{p_m} + 16.3333\mathbf{t}_1 - 6.33333\mathbf{t}_0
\end{aligned}$$

B. Merging of two B-spline Curves Computed Symbolically

The first curve consists of four segments with the control points $\mathbf{p}_{1,i}$ $(i = 0, \ldots, 7)$, the second curve is one B-spline segment with the control points $\mathbf{p}_{2,j}$, $(j = 0 \ldots, 4)$ (see Section 3.2, Example 2 and Figure 13). The new control points of the merged curve are $\mathbf{b}_k$ $(k = 5, \ldots, 10)$. These are computed by minimizing the target function in Equation (15).

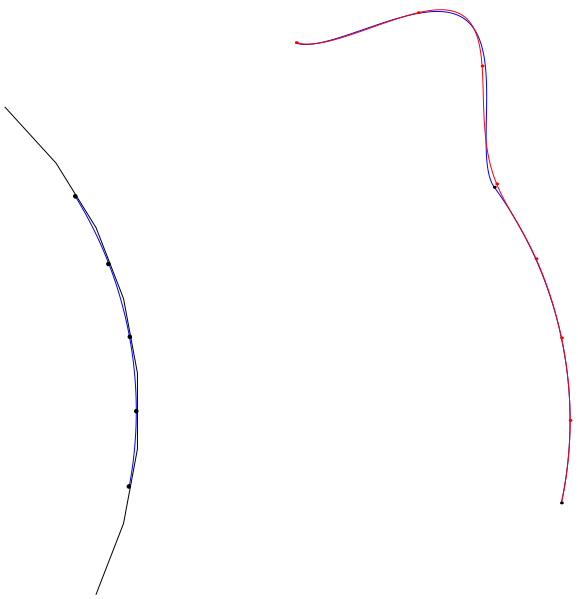


Figure 13. Left: the first input curve consists of four segments; Right: one segment merged to the first curve (it has been divided into three parts during the computation).

The computed new control points are:

$$\begin{aligned}
\mathbf{b}_5 &= 0.0012\mathbf{p}_{1,4} + 1.0831\mathbf{p}_{1,5} + 0.2684\mathbf{p}_{1,6} + 0.0053\mathbf{p}_{1,7} \\
&\quad -0.0087\mathbf{p}_{2,0} - 0.1473\mathbf{p}_{2,1} - 0.1834\mathbf{p}_{2,2} - 0.0186\mathbf{p}_{2,3} \\
\mathbf{b}_6 &= -0.0093\mathbf{p}_{1,4} - 0.4007\mathbf{p}_{1,5} - 0.0289\mathbf{p}_{1,6} + 0.0012\mathbf{p}_{1,7} \\
&\quad +0.0392\mathbf{p}_{2,0} + 0.6093\mathbf{p}_{2,1} + 0.721\mathbf{p}_{2,2} + 0.0682\mathbf{p}_{2,3} \\
\mathbf{b}_7 &= 0.006\mathbf{p}_{1,4} + 0.206\mathbf{p}_{1,5} - 0.01\mathbf{p}_{1,6} - 0.0017\mathbf{p}_{1,7} \\
&\quad -0.0137\mathbf{p}_{2,0} + 0.0708\mathbf{p}_{2,1} + 0.5585\mathbf{p}_{2,2} + 0.1835\mathbf{p}_{2,3} \\
\mathbf{b}_8 &= -0.006\mathbf{p}_{1,4} - 0.1831\mathbf{p}_{1,5} + 0.0222\mathbf{p}_{1,6} + 0.0021\mathbf{p}_{1,7} \\
&\quad +0.0121\mathbf{p}_{2,0} + 0.1586\mathbf{p}_{2,1} + 0.6108\mathbf{p}_{2,2} + 0.3709\mathbf{p}_{2,3} + 0.0123\mathbf{p}_{2,4} \\
\mathbf{b}_9 &= 0.01081\mathbf{p}_{1,4} + 0.3173\mathbf{p}_{1,5} - 0.04845\mathbf{p}_{1,6} - 0.0041\mathbf{p}_{1,7} \\
&\quad -0.02099\mathbf{p}_{2,0} - 0.152\mathbf{p}_{2,1} + 0.2803\mathbf{p}_{2,2} + 0.5554\mathbf{p}_{2,3} + 0.0617\mathbf{p}_{2,4} \\
\mathbf{b}_{10} &= -0.048\mathbf{p}_{1,4} - 1.3884\mathbf{p}_{1,5} + 0.2267\mathbf{p}_{1,6} + 0.0184\mathbf{p}_{1,7} \\
&\quad +0.0918\mathbf{p}_{2,0} + 0.7142\mathbf{p}_{2,1} + 0.5708\mathbf{p}_{2,2} + 0.6294\mathbf{p}_{2,3} + 0.1852\mathbf{p}_{2,4}
\end{aligned}$$

C. Symbolic Merging of Two B-spline Segments into One B-spline Curve Consisting of Six Segments

The input data are two B-spline segments of fourth degree represented by the vector functions $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$, $t \in [0,1]$ with the control points $\mathbf{p_j}$ and $\mathbf{q_j}$ $(j = 0,\ldots,4)$ respectively. Each of the segments will be divided into three segments by parameter transformation so that all are defined over the parameter interval $[0,1]$. The new curve approximating the merged given curves consists of six segments. It is determined by a periodic uniform knot vector and the control points $\mathbf{b_j}$ $(j = 0,\ldots 9)$. These control points are computed by minimizing a target function, which is quadratic in these unknown points. The target function contains the squared differences of the six segments of the new curve and the corresponding input segments of $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$, respectively and the squared differences of the corresponding first derivatives at the endpoints of these curve segments with a factor 0.3. The minimization of the target function leads to a system of 10 linear equations, the solution of which are the required control points of the new curve.

$$\begin{aligned}
\mathbf{b_0} &= 0.185185\mathbf{p_0} + 0.673269\mathbf{p_1} + 1.0643\mathbf{p_2} + 1.20012\mathbf{p_3} + 0.137186\mathbf{p_4} \\
&\quad -0.043639\mathbf{q_0} - 0.879116\mathbf{q_1} - 1.20012\mathbf{q_2} - 0.137186\mathbf{q_3}, \\
\mathbf{b_1} &= 0.0617284\mathbf{p_0} + 0.545444\mathbf{p_1} + 0.167238\mathbf{p_2} - 0.264593\mathbf{p_3} - 0.0316061\mathbf{p_4} \\
&\quad +0.0101115\mathbf{q_0} + 0.203132\mathbf{q_1} + 0.276939\mathbf{q_2} + 0.0316061\mathbf{q_3}, \\
\mathbf{b_2} &= 0.0123457\mathbf{p_0} + 0.376159\mathbf{p_1} + 0.6707\mathbf{p_2} + 0.217973\mathbf{p_3} + 0.0177297\mathbf{p_4} \\
&\quad -0.00578833\mathbf{q_0} - 0.115145\mathbf{q_1} - 0.156245\mathbf{q_2} - 0.0177297\mathbf{q_3}, \\
\mathbf{b_3} &= 0.179171\mathbf{p_1} + 0.513481\mathbf{p_2} + 0.029867\mathbf{p_3} - 0.0173048\mathbf{p_4} \\
&\quad +0.00601424\mathbf{q_0} + 0.116148\mathbf{q_1} + 0.155318\mathbf{q_2} + 0.0173048\mathbf{q_3},
\end{aligned}$$

$$
\begin{aligned}
\mathbf{b_4} &= 0.0730448\mathbf{p_1} + 0.76253\mathbf{p_2} + 0.639503\mathbf{p_3} + 0.0412879\mathbf{p_4} \\
&\quad -0.0113164\mathbf{q_0} - 0.206975\mathbf{q_1} - 0.269133\mathbf{q_2} - 0.0289422\mathbf{q_3}, \\
\mathbf{b_5} &= -0.0210533\mathbf{p_1} - 0.169111\mathbf{p_2} - 0.0913328\mathbf{p_3} + 0.000987459\mathbf{p_4} \\
&\quad +0.033399\mathbf{q_0} + 0.539482\mathbf{q_1} + 0.646888\mathbf{q_2} + 0.0607409\mathbf{q_3}, \\
\mathbf{b_6} &= 0.010092\mathbf{p_1} + 0.0728503\mathbf{p_2} + 0.0279194\mathbf{p_3} - 0.00274794\mathbf{p_4} \\
&\quad -0.010092\mathbf{q_0} + 0.112335\mathbf{q_1} + 0.60171\mathbf{q_2} + 0.187933\mathbf{q_3}, \\
\mathbf{b_7} &= -0.00794608\mathbf{p_1} - 0.0522839\mathbf{p_2} - 0.0123355\mathbf{p_3} + 0.00353283\mathbf{p_4} \\
&\quad +0.00794608\mathbf{q_0} + 0.114012\mathbf{q_1} + 0.567891\mathbf{q_2} + 0.366838\mathbf{q_3} + 0.0123457\mathbf{q_4}, \\
\mathbf{b_8} &= 0.0124406\mathbf{p_1} + 0.0781984\mathbf{p_2} + 0.0123489\mathbf{p_3} - 0.00652041\mathbf{p_4} \\
&\quad -0.0124406\mathbf{q_0} - 0.0658527\mathbf{q_1} + 0.358021\mathbf{q_2} + 0.562076\mathbf{q_3} + 0.0617284\mathbf{q_4}, \\
\mathbf{b_9} &= -0.0535164\mathbf{p_1} - 0.329979\mathbf{p_2} - 0.040927\mathbf{p_3} + 0.0297809\mathbf{p_4} \\
&\quad +0.0535164\mathbf{q_0} + 0.329979\mathbf{q_1} + 0.226112\mathbf{q_2} + 0.599849\mathbf{q_3} + 0.185185\mathbf{q_4}
\end{aligned}
$$

Index of Supplemental Materials

The computation and generation of segment merging and surface merging examples were made using Mathematica. The segment merging program code in .cdf format can be downloaded from: http://www.math.bme.hu/~kongeo/JCGT-paper/segment-merging.cdf. The file can be opened by the Wolfram CDF Player (freely available at: http://www.wolfram.com/cdf-player). The input and output surfaces of the surface merging example are visualized on dynamic figures. They show the B-spline patches to be merged and the resulting surfaces. These figures are exported in .cdf file format and embedded in the page http://www.math.bme.hu/~kongeo/JCGT-paper/surface-merging.html. The dynamic figures can be directly seen on the web page with the plug-in of the CDF Player (http://www.wolfram.com/cdf-player) or they can be downloaded together in one .cdf file http://www.math.bme.hu/~kongeo/JCGT-paper/surface-merging.cdf.

References

CASCIOLA, G., AND ROMANI, L. 2004. A generalized conversion matrix between non-uniform B-spline and Bézier Representations with applications in CAGD. In *Multivariate Approximation: Theory and Applications*, 24-29 aprile 2003, Cancun, Mexico. URL: http://amsacta.unibo.it/853/1/preprint.pdf. 56

CHEN, J., AND WANG, G.-J. 2010. Approximate merging of B-spline curves and surfaces. *Applied Mathematics-A Journal of Chinese Universities 25*, 4, 429–436. URL: http://dx.doi.org/10.1007/s11766-010-2169-1. 56, 73

CHOI, B. K., YOO, W. S., AND LEE, C. S. 1990. Matrix representation for NURBS curves and surfaces. *Computer-Aided Design 22*, 235–240. URL: http://www.sciencedirect.com/science/article/pii/001044859090052E. 55

COHEN, E., AND RIESENFELD, R. F. 1982. General matrix representations for Bezier and B-spline curves. *Computers in Industry 3*, 9–15. URL: http://www.sciencedirect.com/science/article/pii/0166361582900276. 55

FARIN, G. 1990. *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*. Academic Press, Cambridge, MA. 61

GRABOWSKI, H., AND LI, X. 1992. Coefficient formula and matrix of nonuniform B-spline functions. *Computer-Aided Design*. 56

HU, S.-M., TAI, C.-L., AND ZHANG, S.-H. 2002. An extension algorithm for B-splines by curve unclamping. *Computer-Aided Design 34*, 415–419. URL: http://www.sciencedirect.com/science/article/pii/S0010448501001087. 56

LIU, L., AND WANG, G. 2002. Explicit matrix representation for NURBS curves and surfaces. *Computer Aided Geometric Design 19*, 409–419. URL: http://www.sciencedirect.com/science/article/pii/S0167839602001243. 56

PATTERSON, R. R., AND BAJAJ, C. 1989. Curvature adjusted parametrization of curves. Tech. Rep. 89–907, Purdue e-pubs. URL: http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1772&context=cstech. 68

PUNGOTRA, H., KNOPF, G. K., AND CANAS, R. 2010. Merging multiple B-spline surface patches in a virtual reality environment. *Computer-Aided Design 42*, 847–859. URL: http://www.sciencedirect.com/science/article/pii/S0010448510000965. 56

ROMANI, L., AND SABIN, M. A. 2004. The conversion matrix between uniform B-spline and Bézier representation. *Computer Aided Geometric Design 21*, 549–560. URL: http://www.sciencedirect.com/science/article/pii/S0167839604000597n. 56

SZILVÁSI-NAGY, M., AND BÉLA, S. 2013. Stitching B-spline curves symbolically. *KoG 17*, 3–8. URL: http://master.grad.hr/hdgg/kog_stranica/kog17/1szilvasy-KoG17.pdf. 56, 62

SZILVASI-NAGY, M., MATYASI, G., AND BELA, S. 2013. Geometric simulation of locally optimal tool paths in three-axis milling. *Journal for Geometry and Graphics 17*, 223–235. URL: http://www.heldermann.de/JGG/JGG17/JGG172/jgg17019.htm. 57

TAI, C.-L., HU, S.-M., HUANG, Q.-X., AND QI-XING. 2003. Approximate merging of B-spline curves via knot adjustment and constrained optimization. *Computer-Aided Design 35*, 893–899. URL: http://www.sciencedirect.com/science/article/pii/S0010448502001768". 56

Author Contact Information

Sz. Béla
Dept. of Geometry,
Budapest University of Technology
and Economics
Műegyetem rkp. 3-6.
Budapest, H-1111 Hungary
belus@math.bme.hu

M. Szilvási-Nagy
Dept. of Geometry,
Budapest University of Technology
and Economics
Műegyetem rkp. 3-6.
Budapest, H-1111 Hungary
szilvasi@math.bme.hu

Sz. Béla and M. Szilvási-Nagy, Merging B-Spline Curves or Surfaces Using Matrix Representation, *Journal of Computer Graphics Techniques (JCGT)*, vol. 6, no. 3, 1–24, 2017
http://jcgt.org/published/0006/03/01/

Received:	2015-10-08		
Recommended:	2016-08-26	Corresponding Editor:	Eric Enderton
Published:	2017-08-15	Editor-in-Chief:	Marc Olano