Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 51(65), Fascicola 2, 2006

Contributions in Recursive Filtering for B-spline Interpolation in Signal Processing

Liliana Stoica[1]

Abstract –The problem of interpolation a set of data is an old one, but the demanding of flexibility and high speed in operating on-line and in real time processing need to find new methods and improve the old ones [1]. The main properties of B-spline functions offer the possibility to implement algorithms of interpolation in a faster and optimal manner. A function can be represented by B-spline functions with a set of coefficients. For interpolative signal reconstruction it is necessary to calculate those coefficients. In this paper, for cubic spline interpolation it is analyzed a known algorithm and some of his deficiencies. Also there are relieved some possibilities for developing new algorithms that could eliminate those problems. It is presented another way to determine the initial coefficients by using the polynomial representation on short intervals of the spline function and his derivatives. Based on this results are made several observations for further use in improving the algorithm.

Keywords: interpolation, B-spline functions

I. INTRODUCTION

In many applications are given some samples of a signal and it is necessary to estimate the values between them. The interpolation problem it is an actual matter although great mathematicians with hundred years ago approached it. This interpolation process supposes to find a function that can approximate the given data. Usually, the signals in digital signal processing are represented by equally spaced samples. For interpolation can be used the spline functions with uniform knots and unit spacing. Polynomial spline functions have been used in many applications because of their main properties: continuous piecewise polynomial of degree n with derivatives up to order n-1. Another advantage is that a spline function can be obtained as a linear combination of shifted B-spline functions and a set of coefficients. The B-splines are maybe the simplest functions and the most used of them are cubic splines. The section II presents some properties of the B-spline functions and the problem of spline interpolation.

Having a set of data, to determine the interpolation function it is necessary to calculate the spline coefficients. In their work Michael Unser and his team give an algorithm to find those coefficients, algorithm that need some boundary conditions for calculating initial coefficients [3], [4], [5]. In section III is presented their algorithm that use simple digital filter techniques.

By section IV are analyzed some results obtained for known signals. Samples from sine and cosine functions, from a straight line are used to calculate the B-spline coefficients and then the interpolated values for a specific interpolation factor.

In section V it is presented a new approach for calculating initial B-spline coefficients. The solution does not need to perform an extension on Z for the input signal function. There are presented some results in comparison with the other algorithm.

II. B-SPLINE FUNCTIONS

The spline functions are piecewise polynomials of degree n with continuity of the spline and its derivatives up to order (n-1) at the knots [2]. In this work are used only functions with uniform knots and unit spacing.

A spline function $f^{n}(x)$ is uniquely characterized by the B-spline coefficients c(k) , where:

$$f^{n}(x) = \sum_{k \in Z} c(k)\beta^{n}(x-k) \qquad (1)$$

$\beta^{n}(x)$ is the B-spline function of degree n constructed from the (n+1) -fold convolution of a rectangular pulse β^{0} :

$$\beta^{0}(x) = \begin{cases} 1, & -1/2 < x < 1/2 \\ 1/2, & |x| = 1/2 \\ 0, & otherwise \end{cases} \qquad (2)$$

For n = 3 we have the cubic B-spline function which is often used for performing high-quality interpolation [3]:

[1] "Politehnica" University of Timisoara, Faculty of Electronics and Telecomunication, Bd. V. Pârvan No. 2, 300223 Timişoara, e-mail: liliana.stoica@etc.upt.ro

$$\beta^3(x) = \begin{cases} 2/3 - |x|^2 + |x|^3/2, & 0 \le |x| < 1 \\ (2-|x|)^3/6, & 1 \le |x| < 2 \\ 0, & 2 \le |x| \end{cases} \quad (3)$$

The B-spline interpolation problem traditionally it is resolved by using matrices and standard numerical techniques. The algorithms are long and are necessary many operations to determine the solutions. A faster way to resolve the problem is to use simpler digital filter technique. Unser use the discrete B-spline functions and define the indirect and direct B-spline transforms [3], [4]. The B-spline coefficients can be obtained by linear filtering. Having a set of data the spline coefficients are calculated such that the function goes through the data points exactly. The interpolated values of the signal are obtained also by digital filtering.

III. SPLINE INTERPOLATION ALGORITHM

To perform an interpolation process for a set of N samples it is necessary to find the interpolation function in (1).The input data are lettering by {*s*(0), *s*(1), *s*(2),…, *s*(N-1)}. For that we have to calculate the B-spline coefficients. A solution proposed by M. Unser [3], [4] is to apply for the input signal a digital filter, called direct B-spline filter:

$$(b_1^3)^{-1}(k) \quad \leftrightarrow \quad [B_1^3(z)]^{-1} = \frac{6}{z+4+z^{-1}} \quad (4)$$

b_1^3(k) is the discrete B-spline function of degree n=3 (cubic). It is said that the coefficients can be calculated by "direct B-spline transform".
This filter is implemented by 2 filters: first a causal filter and the second anti-causal. A recursive algorithm it is given to calculate the B-spline coefficients:

$$c^+(k) = s(k) + z_1\, c^+(k-1), \quad k = 1, \dots, N-1 \quad (5)$$

$$c^-(k) = z_1\, (c^-(k+1) - c^+(k)), \quad k = N-2, \dots, 0 \quad (6)$$

where: $z_1 = -2+\sqrt{3}$ and c(k) = 6c^-(k).
For that it is necessary to establish some initial conditions. To recover exactly the initial samples by convolving c(k) with b_1^3(k) are used mirror-symmetric boundary conditions: s(k) = s(l) for (k+l)mod (2N-2) = 0. The resulting signal is periodic with period 2N-2. For the first recursion we have the next initialization:

$$c^+(0) = \sum_{k=0}^{+\infty} s(k) z_1^k \quad (7)$$

Practically is not efficient to calculate this. For the periodic signal resulted by mirroring the input data the relation became:

$$c^+(0) = \frac{1}{1-z_1^{2N-2}} \sum_{k=0}^{2N-3} s(k) z_1^k \quad (8)$$

The author propose for using in practice the formula (9), with k_0 >log ε/log | z_1|, where ε is the desired level of precision [3]. In the literature it is not pointed out the exact signification and how ε have an effect on the calculating the coefficients process. By taking different values for ε it was observed that only a few coefficients are affected at the beginning and the end of the data string. For a given signal the intermediate values are the same in cases of different values for ε.

$$c^+(0) = \sum_{k=0}^{k_0} s(k) z_1^k \quad (9)$$

For the second recursion it is used:

$$c^-(N-1) = \frac{z_1}{(z_1^2-1)}(c^+(N-1) + z_1 c^+(N-2)) \quad (10)$$

Having the B-spline coefficients, the signal interpolation by an integral factor m it is completed by $f^n(x/m)$, denoted by $f_m^n(x)$. From (1) we obtain:

$$f_m^n(x) = \sum_{k\in Z} c(k) b_m^n(x-km) \quad (11)$$

This operation is called "indirect B-spline transform" and it is implemented by digital filtering [5], [6].

IV. THE ANALYZIS OF THE UNSER'S ALGORITHM

The presented algorithm was implemented and there were calculated the B-spline coefficients and the recovered signal in some particular cases. There are used more arrays of initial data with N samples of sine or cosine functions.
The samples s(k) are defined by s(k)=sin(2πk/M) or s(k)=cos(2πk/M) with k=0, …, N-1, were M=12, 50, 100 or 120. The N samples represent 1, 3 or 5 periods of the functions for different sampling frequencies. It was analyzed also the case of a straight line. The value for k_0 in (9) was selected to k_0=7. From the results there were done several observations.
The B-spline coefficients follow the signal variation and are close to the samples values. In figure 1 are presented the differences between the coefficients and the input data values for s(k)=sin(2πk/120) with N=361.
In all studied cases the values of the B-spline coefficients are almost equal with the samples values. In the case of cardinal spline functions the coefficients are exactly the input data. But those functions are no longer compactly supported. There is lost the advantage of calculating any value of the interpolation polynomial using maximum (n+1) B-spline functions

(for the sampling points are necessary only n functions).

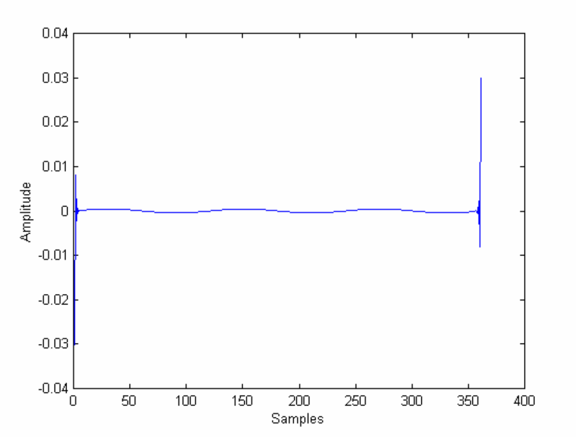

Fig.1Diferences between B-spline coefficients and input data

For sine and cosine functions have been calculated the B-spline coefficients for different sampling frequencies. There were taken one period of the signal and then three periods. In each case were obtained different values for the coefficients. Generally $c(\mathrm{k})$ depends on the signal sampling frequency (sampling unit). In Table 1 are compared the coefficients $c(\mathrm{k})$ for different sampling frequencies in the case of sine function: $\sin(2\pi k/M)$. The values are obtained on the same points α of the input function characteristic.

frequency the differences are increasing (10^{-2} for cosine and 10^{-1} for sine functions when M=12).

In the case of the odd functions by performing the mirror extension [5] for the input signal, the functions derivatives are no longer continues.

The coefficients obtained for one or more periods are not the same. For the second period, the coefficients are different in comparison with the first period, but are almost equals with the ones in the next periods. In Table 1 and Table 2 are observed the differences between $c(0)$ from the first period and his correspondents in second and third periods. Those last ones are almost equal.

For the intermediary periods the values are the same and some differences are present at the beginning of the first period and at the end of the last. The firsts and lasts few coefficients have values close to the others, but not equal.

Between firsts and lasts coefficients appear a sort of symmetry. It can be said that we have a “side effect”. This side effect can be due to the initial conditions.

Those coefficients introduce some errors at the beginning and the end of the sequence in the signal reconstruction.

In the case of the sine function we observe that the difference between $c(0)$ and $c(M)$ is greater than in the case of cosine function. For the odd functions the mirror-symmetric conditions introduce bigger errors.

Table 1. Coefficients for $s(\mathrm{k})=\sin(2\pi k/M)$, $k_0=7$

α	M	First period		Second period		Third period	
		k	$c(\mathrm{k})$	k	$c(\mathrm{k})$	k	$c(\mathrm{k})$
0	12	0	-0.3021395140	(M+0)	-0.0000000413	(2M+0)	0.0000000413
	120	0	-0.0302443872	(M+0)	0	(2M+0)	0
π/6	12	1	0.6043309293	(M+1)	0.5233729016	(2M+1)	0.5233727360
	120	10	0.5002284575	(M+10)	0.5002285152	(2M+10)	0.5002285152
π/3	12	2	0.8848157966	(M+2)	0.9065084347	(2M+2)	0.9065090142
	120	20	0.8664212038	(M+20)	0.8664212038	(2M+20)	0.8664212038

Table 2 present the same coefficients, but for the cosine function: $\cos(2\pi k/M)$. The values depend not only of the shape of the function, but also of the sampling frequency.

Studying the case of sine and cosine functions it is clearly that for great sampling frequencies the values of $c(\mathrm{k})$ and $s(\mathrm{k})$ are very close (differences of maximum 10^{-4} order for M=120). At smaller sampling

For the signal reconstruction and interpolation, in all cases studied, were observed those side effects. At the beginning and the end of the data sequence the errors are higher then the others.

Considering the input signal $s(\mathrm{k})=\sin(2\pi k/120)$ we calculated the B-spline coefficients and completed the interpolation by factor 2. For the first and last few points the errors are $e=10^{-3}$ and decrease fast to $e=10^{-8}$

Table 2. Coefficients for $s(\mathrm{k})=\cos(2\pi k/M)$, $k_0=7$

α	M	First period		Second period		Third period	
		k	$c(\mathrm{k})$	k	$c(\mathrm{k})$	k	$c(\mathrm{k})$
0	12	0	1.0467677172	(M+0)	1.0467457811	(2M+0)	1.0467457811
	120	0	1.0004237020	(M+0)	1.0004570305	(2M+0)	1.0004570305
π/6	12	1	0.9065025600	(M+1)	0.9065084377	(2M+1)	0.9065084377
	120	10	0.8664212037	(M+10)	0.8664212038	(2M+10)	0.8664212038
π/3	12	2	0.5233744655	(M+2)	0.5233728905	(2M+2)	0.5233728905
	120	20	0.5002285152	(M+20)	0.5002285152	(2M+20)	0.5002285152

in the interpolated points. The algorithm has excellent properties of convergence. At the input data points the values obtained are exactly excepting the 2 extremely points. For calculating every interpolated value are necessary the coefficient for the current point $c(\mathrm{k})$ and some anterior and posterior coefficients. For the first and last 2 points of the string some of those coefficients are not known (example $c(-1)$) and considered zero. This is one of the causes for the side errors that appear in every cases when perform interpolation.

For $s(\mathrm{k}) = \cos(2\pi \mathrm{k}/120)$ the results are comparative. It was done the interpolation by 2 and the errors at the beginning and the end of the results array are smaller, starting from 10^{-6} and decay faster then for the above function. Similar results have been obtained in the cases of the studied signals with other sampling unit (M=50, 100). The errors for the B-spline coefficients have a real significance in the interpolation process.

For initializing $c^{+}(0)$ are used a finite number k_0 of samples. Extending the signal by mirroring [3],[5] it was calculated $c^{+}(0)$ in the relation (8). The results for coefficients, signal reconstruction and interpolation by 2 are better than in the case of using (9) for initialization, with $\mathrm{k}_0=7$. For the cosine signal $\cos(2\pi \mathrm{k}/120)$ the interpolation errors at the beginning and the end of the string are much smaller then for $\mathrm{k}_0=7$. Compared with the same case of initialization for $\sin(2\pi \mathrm{k}/120)$ the results are better too. But the errors decay slower compared with the cosine function case. It can be said that the mirror extension present disadvantages for the odd functions.

The coefficients for the extended signals are similar with the values obtained in the second and third period of the signals for $\mathrm{k}_0=7$. It can be said that the values for $c(\mathrm{k})$ corresponding to the second period in Table 1 and Table 2 (where $\mathrm{k}_0=7$) in fact represents the coefficients for $\mathrm{k}_0= \infty$. If we can take the coefficients values from the intermediary period then the initialization is not necessary to be very precise. But in this case are necessary an increased amount of numerical operations.

If the input samples correspond to a straight line $s(\mathrm{k})= 1$, N=50, the B-spline coefficients $c(\mathrm{k})$ are calculated and presented in Table 3. At the beginning and at the end the coefficients have values different from 1. But they converge fast to 1. We perform the interpolation of the signal by the integral factor 2 and obtain the values $y(\mathrm{k}/2)$ presented in the third column of Table 3.

Excepting the first and last values, in data points the signal is exactly recovered. Between those points, the interpolated values have errors for some positions. There is shown again that the algorithm has a good convergence due to polynomial spline properties. But for some oscillations in coefficients sequence are obtained oscillations in the interpolated signal. This could be an inconvenient if the input signal has a small number of samples.

For the errors that appear in the process of determination the B-spline coefficients a possible cause is that on principle the input signal is considered of infinite duration (extended to $\mathrm{k}\in Z$) [3], [4]. In practice, for the recursive algorithm, the initialization is done in k = 0 and is limited to a k_0. If it is done a signal extension on both sides and performed the interpolation, in the center area the results are very good. In this way the middle area is the interval were the input samples are defined and the side errors present no more importance.

Table 3. Input signal $s(\mathrm{k})= 1$, $\mathrm{k}_0=7$

k	c(k)	y(k/2)
0	0.999637023	0.8333107558
		0.9791538800
1	1.0000097259	1
		1.0000026699
2	0.9999973939	1
		0.9999992845
3	1.0000006982	1
		1.0000001916
4	0.9999998128	1
…	…	…
30	1	1
		1
31	1	1
…	…	…
49	1	1
		0.9791666666
50	1	0.8333333333

In the case of the straight line we took for input much more samples. For calculating $c^{+}(0)$ it is used the relation (7):

$$c^{+}(0) = \sum_{k=0}^{+\infty} s(k) z_1^k = \sum_{k=0}^{+\infty} z_1^k = \frac{1}{1-z_1} \tag{12}$$

All the values for B-spline coefficients are equal to 1. The samples obtained for the interpolated signal by the integral factor 2 are presented in Table 4, along with the coefficients.

Table 4. Input signal $s(\mathrm{k})= 1$, $\mathrm{k}_0=\infty$

k	c(k)	y(k/2)
0	1	0.8333333333
		0.9791666666
1	1	1
		1
2	1	1
		1
3	1	1
		1
4	1	1
…	…	…
30	1	1
		1
31	1	1
…	…	…
49	1	1
		0.9791666666
50	1	0.8333333333

We see that still exist a side effect due to the 2 coefficients that are not known at the beginning and the end of the data string.

In conclusion, to reduce the side effect it is necessary to perform a signal extension on both sides, but it is essentially to establish the right way to do it. The presented examples show that the extension of a signal by using its mirror image it is not an optimal solution in all the cases. The extension can be performed for the coefficients string too. We searched another approach for calculate the B-spline coefficients.

V. NEW METHOD FOR CALCULATING THE INITIAL B-SPLINE COEFFICIENTS

It was searched a way to initialize the spline coefficients without performing the signal extension on Z. For the new coefficients it is used the notation: $c_n(k)$. For the given set of data $\{s(0), s(1), s(2),\ldots, s(N-1)\}$ we consider $f(x)$ the interpolation function. We seek for that one to be a cubic spline function. An important property of those functions is that they are piecewise polynomial functions.

From the convolution of the coefficients with the cubic B-spline function in formula (1) it can be write the next relation:

$$6f(k) = 4\, c_n(k) + c_n(k-1) + c_n(k+1) \tag{13}$$

This is happening in the function knots. Also in the sample points, the relations between the function derivatives and the B-spline coefficients are:

$$f'(k) = 0\, c_n(k) - \tfrac{1}{2} c_n(k-1) + \tfrac{1}{2} c_n(k+1) \tag{14}$$
$$f''(k) = -2\, c_n(k) + c_n(k-1) + c_n(k+1) \tag{15}$$

Those relations help us to evaluate the properties and the values for the B-spline coefficients. For k=2 it can be deduced:

$$c_n(2) = f(2) - f''(2)/6 \tag{16}$$
$$c_n(0) = c_n(2) - 2f'(1) \tag{17}$$
$$c_n(1) = \frac{6f(1) - c_n(0) - c_n(2)}{4} \tag{18}$$

The problem is how to calculate $f'(1)$ and $f''(2)$ from the known samples. The interpolation function is a B-spline (piecewise polynomial), so we can approximate $f(k)$ by a polynomial function on short intervals. With this polynomial and his derivatives we calculate the values for the first 3 coefficients.

Consider f a polynomial function of 4-th order (pass trough 5 points):

$$f(x) = a + b\,x + d\,x^2 + e\,x^3 + g\,x^4 \tag{19}$$

The function and the function derivatives of order 1 and 2 have been evaluated on the interval [0;4] and are obtained the next relations:

$$f'(1) = \frac{-3f(0) - 10f(1) + 18f(2) + f(4)}{12} \tag{20}$$

$$f''(2) = \frac{-(f(0) + f(4)) + 16(f(1) + f(3)) - 30f(2)}{12} \tag{21}$$

The main condition is that the interpolation function to pass through the input samples: $f(k)=s(k)$ for k=0,1…, N-1. By using (20) and (21) in relations (16), (17) and (18) are calculated the initial coefficients $c_n(0)$, $c_n(1)$ and $c_n(2)$ without performing any signal extension.

Those 3 coefficients have been calculated for the previous studied signals. Some results are presented in Table 5 for sine functions and Table 6 for cosine functions. The coefficients calculated with the new relations (in the last column) are compared with the corresponding coefficients in Unser's algorithm, for $k_0=7$ and $k_0=\infty$.

The coefficients for $s(k)=\sin(2\pi k/M)$ are presented in Table 5 for two situations: M=12 and M=120. As it can be seen the new values are much closer to the ideal values that the ones for $k_0=7$. The differences between $c_n(k)$ and $c(k)$ for $k_0=\infty$ are in order of 10^{-3} for M=12. For a higher sampling frequency the differences decrease to 10^{-7} (M-120).

Table 5. Coefficients for $s(k)=\sin(2\pi k/M)$

M	k	$c(k)$		$c_n(k)$
12	0	$k_0=7$	-0.3021395140	-0.0035163260
		$k_0=\infty$	-0.0000000413	
	1	$k_0=7$	0.6043309293	0.5244880516
		$k_0=\infty$	0.5233729016	
	2	$k_0=7$	0.8848157966	0.9055641193
		$k_0=\infty$	0.9065084347	
120	0	$k_0=7$	-0.0302443872	-0.0000000544
		$k_0=\infty$	0	
	1	$k_0=7$	0.0604638345	0.0523598917
		$k_0=\infty$	0.0523598753	
	2	$k_0=7$	0.1024047866	0.1045762250
		$k_0=\infty$	0.1045762359	

In case of $s(k)=\cos(2\pi k/M)$ the values are presented in Table 6. The results are similar: differences of order 10^{-7} for M=120.

Table 6. Coefficients for $s(k)=\cos(2\pi k/M)$

M	k	$c(k)$		$c_n(k)$
12	0	$k_0=7$	1.0467677172	1.0495366943
		$k_0=\infty$	1.0467457811	
	1	$k_0=7$	0.9065025600	0.9059470100
		$k_0=\infty$	0.9065084377	
	2	$k_0=7$	0.5233744655	0.5228276880
		$k_0=\infty$	0.5233728905	
120	0	$k_0=7$	1.0004237020	1.0004569306
		$k_0=\infty$	1.0004570305	
	1	$k_0=7$	0.9990948692	0.9990859898
		$k_0=\infty$	0.9990859389	
	2	$k_0=7$	0.9949740293	0.9949763183
		$k_0=\infty$	0.9949764222	

Having calculated the initial B-spline coefficients the next step is to establish the algorithm for calculating the others. It has to perform the interpolation with those new coefficients and compare the results with the existing ones.

VI. CONCLUSIONS

In the studied articles this cubic spline interpolation was used for image processing [3], [5], [6]. All the results referred to techniques used in this area. We took the algorithm and applied it for some usually digital signals. The observations and conclusions regarding the coefficients were used for finding an improved method to perform cubic spline interpolation. The B-spline coefficients depend of the sampling frequency, of the input samples values, are close to those and follow the signal variation. The algorithm has excellent properties of convergence due to the spline function nature. It presents some side errors that have a great importance in the interpolation process. Those errors are due to the finite length of the input signal and to the extension by mirroring for some functions. It was searched a way to eliminate the oscillations in interpolated signal by reducing the oscillations in coefficients series.

By the process presented in section V is given an alternative to calculate the initial terms with minimum errors. There were elaborated and tested a few algorithms for eliminating ones of the deficiencies in the presented one and reducing the interpolation errors. The B-spline coefficients are calculated in a simple manner and the side effect can be negligible in the interpolated signal. Those algorithms must be finished and then published in to a further work.

ACKNOWLEDGEMENTS

The author would like to address special thanks to Professor Eugen Pop for his patience, guidance and support.

REFERENCES

[1] T. Blu, P. Thevenaz, M. Unser, "Linear Interpolation Revitalised", *IEEE Transactions on Image Processing*, Vol. 13, No. 5, pp.710-719, May 2004

[2] Gh. Micula, *Functii Spline si aplicatii*, Editura Tehnica Publishing House, Bucuresti, 1978

[3] M. Unser, "Splines: A Perfect Fit for Signal and Image Processing", *IEEE Signal Processing Magazine*, Vol. 16, No. 6, pp. 22-38, Nov. 1999.

[4] M. Unser, A. Aldroubi, M. Eden, "B-Spline Signal Processing: Part I - Theory", *IEEE Transactions on Signal Processing*, Vol. 41, No. 2, pp. 821-833, Feb. 1993.

[5] M. Unser, A. Aldroubi, M. Eden, "B-Spline Signal Processing: Part II – Efficient Design and Applications", *IEEE Transactions on Signal Processing*, Vol. 41, No. 2, pp. 834-848, Feb. 1993.

[6] B. Vrcelj, P.P. Vaidyanathan, "Efficient Implementation of All-Digital Interpolation", *IEEE Transactions on Image Processing*, Vol. 10, No. 11, pp. 1639-1646, Nov. 2001