19th IMEKO TC 4 Symposium and 17th IWADC Workshop
Advances in Instrumentation and Sensors Interoperability
July 18-19, 2013, Barcelona, Spain

Operators to calculate the derivative of digital signals

Lluís Ferrer-Arnau[1], Juan Mon-Gonzalez[1], Vicenç Parisi-Baradad[2]
[1] Departament d'Enginyeria Electrònica, Universitat Politècnica de Catalunya (UPC)
08034 Barcelona, Spain. luis.jorge.ferrer@upc.edu
[2] SARTI Research Group. Electronics Dept. Universitat Politècnica de Catalunya (UPC).
Rambla Exposició 24, 08800, Vilanova i la Geltrú. Barcelona. Spain.+(34) 938 967 200.
www.cdsarti.org , vicenc.parisi@upc.edu

Abstract- This paper presents an FIR operator with a very low computational cost for calculating the first derivative of discrete band-limited signals. The basic idea is to use cubic spline polynomials for interpolating among the samples and estimate the derivative of the discrete signal as a derivative of the polynomial. This method also allows estimating the derivative at the points between samples. The main contribution of this work is to develop a highly efficient FIR operator with only 9 coefficients for implementing the derivative operator algorithm. The resulting filter can be easily implemented in a microcontroller or an FPGA since only 5 additions and 3 fixed-point multiplications are necessary. If the frequency range of the input signal is between 0 and 0.2 times the sampling frequency (taking 5 samples per cycle), the maximum error in calculating the derivative does not exceed 1.03 %.

Keywords**:** *First derivative operator, cubic B-splines, FIR filters, low computational cost, detecting singular points of a discrete signal, DA conversion.*

I. Introducción

It is first worth noting that in this study we used the concept of digital frequency as a quotient of the sampled signal frequency over the sampling frequency. Its value varies between 0 and 1, 0.5 being the Nyquist frequency value. For example, if we sample a continuous sinusoidal signal of 100 Hz at a sampling frequency of 500 Hz, the frequency of the sampled signal will be 0.2. Another concept is an FIR filter, which is a finite impulse response filter.

Calculating the derivatives of discrete signals is an operation performed in many fields, since it allows detecting as much the singular points of a signal as the maximum variation points, as well as the points of local maxima or minima. Some examples are: in the field of image processing, it is used to detect edges; in measuring the quality of electrical signals, it can be used for transient detection; for control systems, it is used in implementing PID digital.

The frequency response of an ideal derivative operator would be *D(w) = I·w*, a linear response of the magnitude with the frequency, and a constant phase shift of 90°.

One of the easiest ways of implementing the derivative in the discrete domain is the central difference equation (1), where *T* is the sampled period.

$$d(n) = \frac{1}{2\cdot T}(x(n+1) - x(n-1)) \tag{1}$$

The derivative of the current point (*n*) is approximated from the average differential between the future value (*n*+1) minus the past value (*n*-1). A delay is needed from one sample. This formula is the result of averaging with a low pass filter the current derivative minus the past derivative (2), and it prevents the high frequency noise from affecting the calculation of the derivative, which is very sensitive to noise.

$$d(n) = \frac{1}{2\cdot T}\left[(x(n) - x(n-1)) + (x(n+1) - x(n))\right] = \frac{1}{2\cdot T}(x(n+1) - x(n-1)) \tag{2}$$

There are other methods for approximating differentiators based on FIR filter design methods, [1-3]. For example the one based on Fourier's work, which is used for finding the coefficients of the frequency response. Another method is the one proposed by T. W. Parks and J. H. McClellan [4, 5], which is based on minimizing the maximum error. The drawback of these methods is that they cannot estimate the value of the derivative at points located between samples.

In this paper we propose interpolating the discrete signal with polynomial cubic splines and taking the derivative from the polynomial itself at every point. Using cubic splines provides continuity up to the second derivative of the junction points (knots) between the polynomials. As such, the system could be used even for estimating the second derivative of the signal. This method is not new, having already been proposed by Unser in [6, 7]; but it is based on applying an anti-causal IIR filter, which involves having to work with finite signals before being able to estimate the derivative. The contribution of this work is based on the design of a single operator that is efficient both in computational cost and the hardware required for performing this operation. Calculating the derivative is achieved by applying an FIR filter with only 9 coefficients, which means it can be applied in real time as it has a constant delay of only four samples.

The most efficient way to implement the derivative operator in a microcontroller or FPGA has been studied. In the end, implementation in fixed-point and 16-bit words was achieved, needing a total of only 5 additions and 3 multiplications. It is important to note that if the digital frequency of the sampled signal is in the range of 0 to 0.2 (5 samples per period), the maximum error in the derivative calculation is less than 1%. At frequencies greater than 0.3, the filter ceases behaving like a derivative operator and behaves like a low pass filter. This can be considered an advantage since the derivative is very sensitive to high frequency noise.

This article is organized according to the following sections: First, in section II, we make a brief introduction to splines, B-splines and the system of equations which implies calculating the coefficients of the polynomials. Section III provides a more detailed look at the central difference operator for calculating the derivative, equations (1) and (2), a study of the accuracy of their calculations, and the frequency range of the input signal which allows an acceptable error. Section IV presents the new proposed method for calculating the derivative at the points where the samples are located, providing a study of both the absolute and the relative errors committed in the valid frequency range. Section V proposes a method for finding the derivative at points located between samples. Section VI is an application study of the algorithm's practical application in a microcontroller or FPGA, and it compares the results of the calculated derivatives of the operator in section III with those of the operator proposed in section IV. Finally, the conclusions of this work are presented.

II. Introduction to splines

The splines are curves or functions defined piecewise by polynomials of different degrees. The splines of order 1 are straight lines that connect the different samples. The junction points of the polynomial are called knots and at this point the splines have the characteristic of having continuity in the curve as well as in derivatives up to order less than the spline. For example with cubic splines the continuity in the knots of polynomials is insured up to the second derivative.

One of the first book or reference work on the spline is [8]. At first it was applied over the field of graphic design basically define continuous curves from several points, interpolating or approximating, without needing that these points have to be evenly spaced.

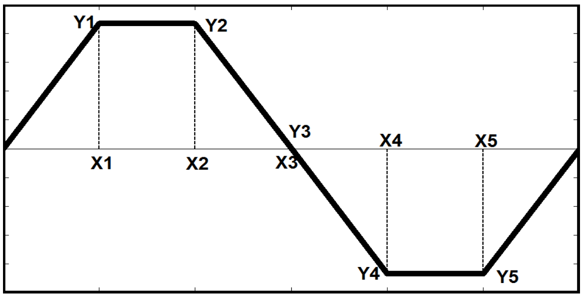


Figure 1. Example of interpolation with linear splines.

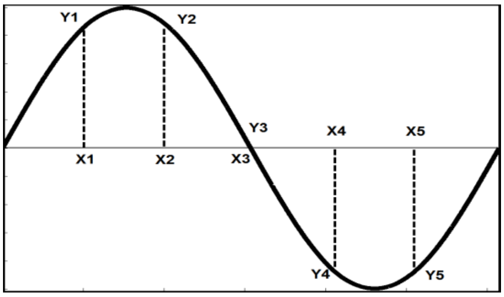


Figure 2. Example of interpolation with cubic splines.

It was in early 1990's Michael Unser professor and director of research group "Biomedical Imaging group" of the Federal Polytechnic School of Lausanne who developed much of the mathematical theory to apply splines in signal processing [6, 7].

$$P_n(x) = Y_n + b_n \cdot (x - x_n) + c_n \cdot (x - x_n)^2 + d_n \cdot (x - x_n)^3 \qquad (3)$$

$$P_n(x_n + 1) = Y_n + b_n + c_n + d_n = P_{n+1}(x_{n+1}) = Y_{n+1} \qquad (4)$$

$$b_n + 2 \cdot c_n + 3 \cdot d_n = b_{n+1} \qquad (5)$$

$$2 \cdot c_n + 6 \cdot d_n = 2 \cdot c_{n+1} \qquad (6)$$

You could say that the basic function of polynomials splines or B-spline functions is a conversion of digital signal to an analog signal, especially in order to define mathematical functions that in the discrete world are not well defined, as derivatives of different order, integration, differential geometry, etc. In [8] C. de Boor explains how to find the coefficients of polynomials splines which is considering the system of equations associated with these polynomials, (3), (4), (5) and (6).

We propose that the polynomial coefficients can be calculated using the discrete convolution between the samples and coefficients that can be represented perfectly by an anti-causal FIR filter [9, 10].

III Central difference operator

This operator described by equations (1) and (2) estimates the derivative as the future value minus the signal's past value, and as such it will always imply a sample delay in its calculation. It is one of the easiest methods for estimating the derivative of a discrete signal, and also has a lower computational cost. It requires only one subtraction and the multiplication of the sampling frequency.

$$D(f) = I \cdot \sin(2 \cdot Pi \cdot f) \tag{7}$$

In Figure 3 we can see the magnitude of the frequency response, equation (7). Since its magnitude has only an imaginary part, there is no phase error. In Figure 4 the relative error as a percentage is observed. Notice that for a digital frequency below 0.04, the error is less than 1%; and for frequencies below 0.1, the error is less than 6%. From frequencies of 0.25 to 0.5, the error increases further because it behaves like a low pass filter rather than a derivative operator.

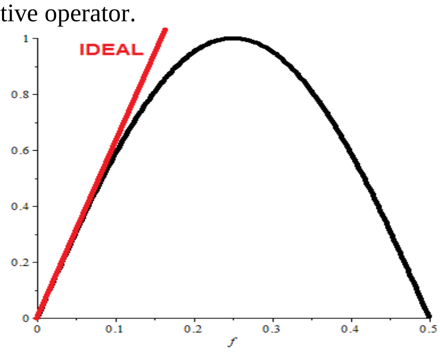


Figure 3. Magnitude of the frequency response of Central difference operator, and ideal response.

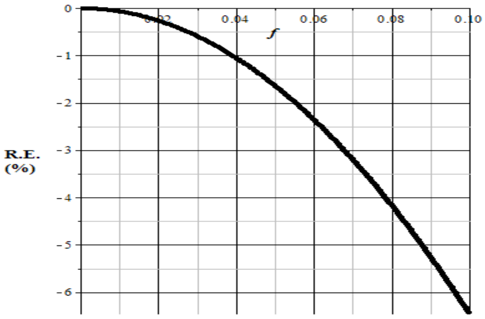


Figure 4. Relative error as a percentage between ideal response and Central difference operator.

IV Derivative operator based on cubic splines

If the signal is interpolated with cubic splines, the derivative corresponds to coefficient "b" of the polynomial in equation (3). In the works of [9] and [10], a FIR filter is developed for calculating these coefficients. Here the filter is optimized to find the most efficient means for implementing it. It was studied, using various numbers of coefficients and various levels of precision in decimal numbers, until the best empirical means was obtained for minimizing the relative error. In the end, the filter defined by equation (8) was obtained. With only 13 coefficients, the maximum relative error is less than 0.8% in the digital frequency range of an input signal between 0 and 0.2 (5 samples per period).

$$\begin{aligned} D(z) = &-0.001 \cdot (z^6 - z^{-6}) + 0.004 \cdot (z^5 - z^{-5}) - 0.015 \cdot (z^4 - z^{-4}) + 0.057 \cdot (z^3 - z^{-3}) \\ &- 0.217 \cdot (z^2 - z^{-2}) + 0.811 \cdot (z - z^{-1}) + 0 \end{aligned} \tag{8}$$

On the other hand, when the filter was tested with only 9 coefficients equation (9), it was found that the maximum relative error did not exceed 1.03 %. This has the advantage of having a lower computational cost and shorter delay of only 4 samples; whereas the filter with 13 coefficients has a delay of 6. This delay means that, in each moment, the derivative of the current input sample is not calculated, but that which corresponds to 4 or 6 samples before the signal. This time delay depends on the sampling frequency. In the same way as the operator in the previous section, if the frequency is higher, the operator acts as a low pass filter. Notice that the central coefficient of equations (8) and (9) is zero.

$$D(z) = \frac{1}{128}\left[-1 \cdot (z^4 - z^{-4}) + 6 \cdot (z^3 - z^{-3}) - 27 \cdot (z^2 - z^{-2}) + 104 \cdot (z - z^{-1})\right] \tag{9}$$

In equation (10) can see the discrete-time Fourier transform of the derivative operator based on just 9 coefficients. Note that, as a good derivative operator, it has only an imaginary part and that the real part is zero. This implies that the output signal, without considering a delay of 4 samples, will always be 90 degrees out of phase in respect to the input signal.

$$D(f) = I \cdot \frac{1}{64} \cdot \left[104 \cdot \sin(2 \cdot \mathrm{Pi} \cdot \mathrm{f}) - 27 \cdot \sin(4 \cdot \mathrm{Pi} \cdot \mathrm{f}) + 6 \cdot \sin(6 \cdot \mathrm{Pi} \cdot \mathrm{f}) - \sin(8 \cdot \mathrm{Pi} \cdot \mathrm{f})\right] \tag{10}$$

Figure 5 shows the magnitude of the frequency response for a derivative operator of 9 coefficients and the line that corresponds to the ideal response.

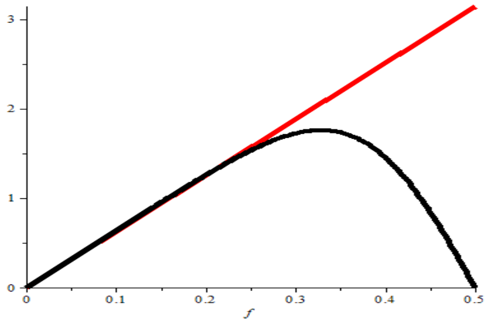


Figure 5. Magnitude of the frequency response of derivative operator based on cubic splines with 9 coefficients.

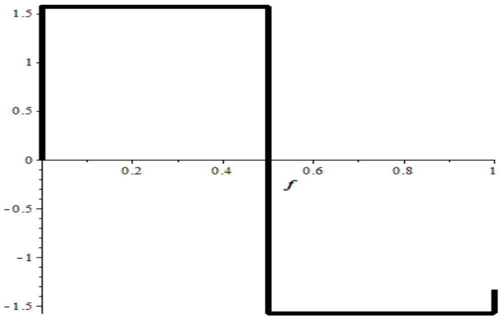


Figure 6. Phase, in radians, of the frequency response of derivative operator based on cubic splines with 9 coefficients, equation (9).

As a percentage in calculating the derivative using either 9 or 13 coefficients, the difference in relative error is small, as can be seen by comparing the graphs in Figures 7 and 8.

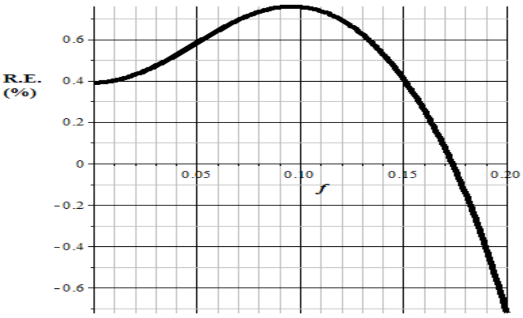


Figure 7. Relative error as a percentage between ideal response and cubic spline derivative operator with 13 coefficients.

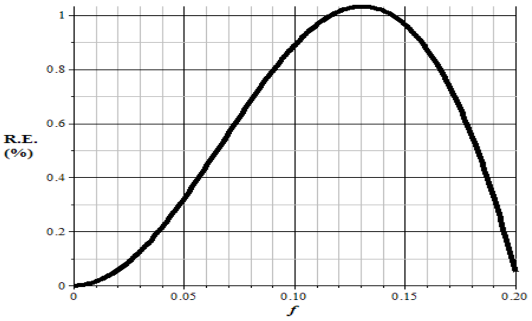


Figure 8. Relative error as a percentage between ideal response and cubic spline derivative operator with 9 coefficients.

V Calculate the derivative at the points between samples

There are 2 ways to calculate the derivative at points between samples. The first is to interpolate the signal to achieve the desired resolution and then apply the operator derived from Section IV. The second method is to calculate coefficients "c" and "d" of the spline polynomial in equation (3) and apply the derivative directly from this equation. This second method is currently being studied and is expected to yield results in the near future.

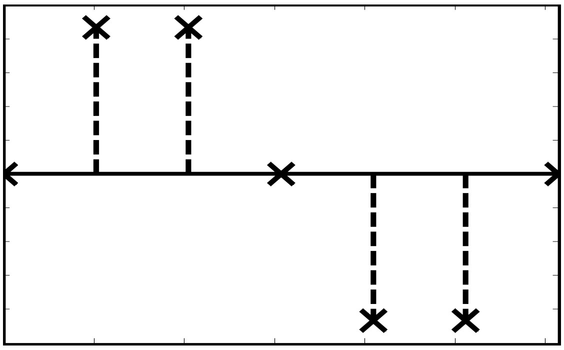


Figure 9. Example of discrete signal obtained by sampling a continuous sine wave.

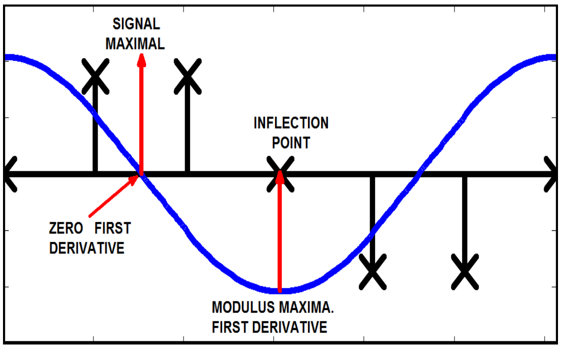


Figure 10. Continuous first derivative of the discrete signal of Figure 9.

For interpolating, it is recommended to use the interpolation filter, equation (11) described in [10], which needs only 5 multiplications and 10 additions per interpolated sample. The interpolation process can be performed iteratively as many times as needed. Each interpolation involves increasing by 2 the total number of samples and the sampling frequency.

Figure 9 shows a sine wave sampled with only 6 samples per period. In Figure 10 is shown the first continuous derivative of the signal of the Figure 9, and its singulars points, obtained by interpolating the discrete signal with cubic splines, and applying the derivative operator of equation (9).

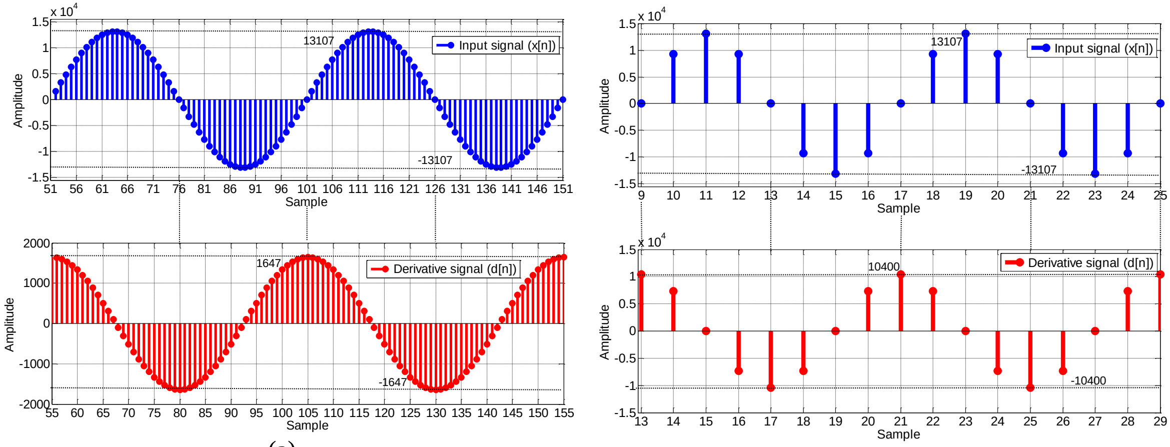
$$I_P(z) = 1 + 0.6\cdot\left(z + z^{-1}\right) - 0.127\cdot\left(z^3 + z^{-3}\right) + 0.034\cdot\left(z^5 + z^{-5}\right) - 0.009\cdot\left(z^7 + z^{-7}\right) + 0.002\cdot\left(z^9 + z^{-9}\right) \qquad (11)$$

VI Comparative between Cubic Splines algorithm and Central-Difference algorithm

The proposed algorithm has been tested on a set of sine wave signals with different frequencies and the results have been compared with the traditional Central-Difference algorithm equation (1). Special attention has been paid to the influence of the digital frequency in the maximum error of the derivative calculated.

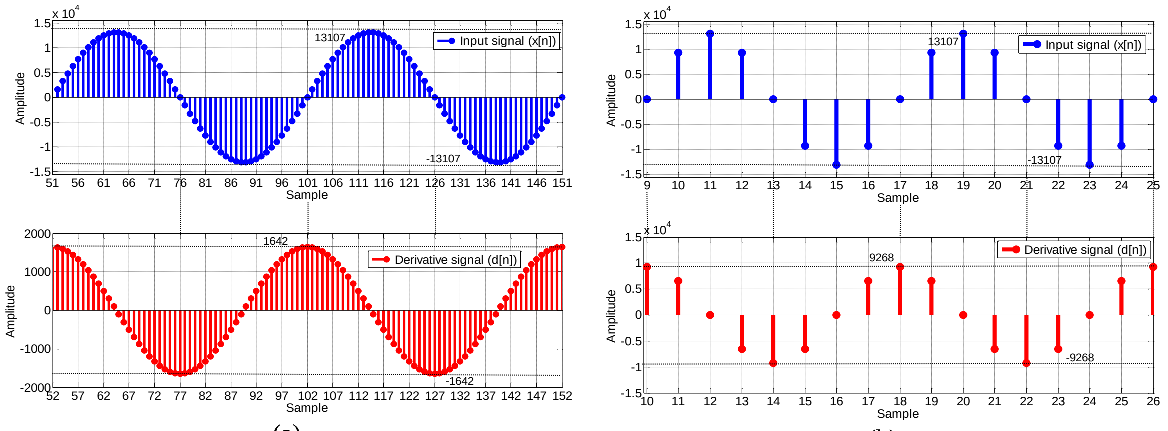
The relative error obtained in fixed-point arithmetic with 16 bit word for both algorithms is shown in Table 1. Notice that the relative error in the Central-Difference algorithm increases with the digital frequency. However, the maxim relative error with the Cubic Splines algorithm is about 1 % for digital frequencies less than 0.2 that is expected according to the Figure 8.

The derivative calculated of sine wave input signal with frequency of 160 Hz and 1 kHz with Cubic Splines algorithm in fixed-point arithmetic with 16 bit word is plotted in Figure 11. It is important to point out than the output signal, *d[n]*, is delayed 4 samples respect the input signal, *x[n]*. In order to compute the magnitude of the calculated derivative, the amplitude of *d[n]* has been multiplied by f_{sample}.



(a) (b)

Figure 11. Cubic Splines algorithm: (a) f_{sample}=8 kHz, f=160 Hz, (b) f_{sample}=8 kHz, f=1 kHz.



(a) (b)

Figure 12. Central-Difference algorithm: (a) f_{sample}=8 kHz, f=160 Hz, (b) f_{sample}=8 kHz, f=1 kHz.

Figure 12 shows the derivative calculated with Central-Difference algorithm. In this case the output delayed is 1 sample.

The magnitude of the derivative of the input signal of Figures 11 and 12 can be calculated according to (12), where *A* and *B* are the amplitude of *x[n]* and *d[n]* respectively. In Figure 11a *A* is equal to 13107 and *B* is 1647.

$$\frac{\partial x(t)}{\partial t}=\frac{\partial(A\cdot\sin(2\cdot\pi\cdot f\cdot t))}{\partial t}=A\cdot 2\cdot\pi\cdot f\cdot\sin(2\cdot\pi\cdot f\cdot t+\frac{\pi}{2})=f_{sample}\cdot d(t)=B\cdot f_{sample}\cdot\sin(2\cdot\pi\cdot f\cdot t+\frac{\pi}{2}) \qquad (12)$$

Table 1. Evaluation of Relative Error

		Cubic Splines Algorithm	**Central-Difference algorithm**
Input Frequency	**Digital Frequency**	**Fixed-Point (16 bit)**	**Fixed-Point (16 bit)**
160 Hz	0.02	0.00 %	-0.31 %
400 Hz	0.05	0.32 %	-1.64 %
800 Hz	0.1	0.89 %	-6.45 %
1000 Hz	0.125	1.03 %	-9.97 %
1600 Hz	0.2	0.04 %	-24.32 %

VII. Conclusions

This paper presents an efficient means for estimating the first derivative of discrete signals based on interpolating the signal with cubic splines. The derivative is taken from the polynomial spline as the derivative of the discrete signal. Although this method has been previously proposed, this paper provides an efficient way to apply it without using anti-causal IIR filters.

The relative error has been studied as a percentage and it has been shown that the error is less than 0.8 % when using an FIR filter with only 13 coefficients; and it is less than 1.03 % when using an operator with only 9 coefficients. The latter requires only 5 multiplications and 3 additions in fixed-point arithmetic for its implementation.

A study has also been made of the error committed in estimating the derivative with the central difference operator. In order to obtain an error of less than 1% with this operator, the digital frequency signal has to be less than 0.04 (25 samples per period). The error is much larger than that of the derivative operator proposed in this paper, but it should be noted that there is an advantage of low computational cost from having only one subtraction and one multilpication.

Currently, we continue to study various operators for estimating the derivative of discrete signals and the optimal means for implementing them.

References

[1] Krishna, B.T.; Rao, S.S., "On design and applications of digital differentiators," Advanced Computing (ICoAC), 2012 Fourth International Conference on , pp.1,7. 13-15 Dec. 2012.

[2] L. R. Rabiner and K. Steiglitz, "The design of wide-band recursive and nonrecursive digital differentiators," IEEE Trans. Audio Electroacoust, vol. AU-18, no. 2, pp. 204–209, Jun. 1970.

[3] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing.Englewood Cliffs, NJ: Prentice-Hall, 1989.

[4] Parks, T.; McClellan, J., "Chebyshev Approximation for Nonrecursive Digital Filters with Linear Phase," Circuit Theory, IEEE Transactions on , vol.19, no.2, pp.189,194, Mar 1972.

[5] T. W. Parks and C. S. Burrus. Digital Filter Design . John Wiley and Sons, 1987.

[6] M. Unser, A. Aldroubi and M. Eden, "B-spline signal processing: Part I-theory," IEEE Trans. SignalProcessing, vol. 41, no. 2, pp. 821-833, 1993.

[7] M. Unser, A. Aldroubi and M. Eden, "B-spline signal processing: Part II-efficient design and applications," IEEE Trans. Signal Processing, vol. 41, no. 2, pp. 834-848, 1993.

[8] C. de Boor, A practical guide to splines. New York: Springer-Verlag, 1978.

[9] Ferrer-Arnau, Ll. [et al.]. Efficient representation of contours using splines implemented with FIR filters. A: 7th International Conference on Next Generation Web Services Practices. "7th IEEE International Conference on Next Generation Web Services Practices: proceedings". pp. 125-128, 2011.

[10] Ll. Ferrer-Arnau, R. Reig-Bolaño, P. Marti-Puig, A. Manjabacas, V. Parisi-Baradad, "Efficient cubic spline interpolation implemented with FIR filters", International Journal of Computer Information Systems and Industrial Management Applications. ISSN 2150-7988, vol. 5, pp. 098-105, 2012.

[11] Ferrer-Arnau, Ll.; Parisi, V. Efficient implementation with FIR filters of operators based on B-splines to represent and classify signals of one and two dimensions. "Proceedings of the 2012 Barcelona Forum on Ph.D. Research in Communication and Information Technologies". pp. 21-22, Barcelona: 2012.