Efficient Image Resizing Using Finite Differences

Arrate Muñoz Thierry Blu Michael Unser

Biomedical Imaging Group, DMT/IOA
Swiss Federal Institute of Technology, Lausanne
CH-1015 Lausanne EPFL, SWITZERLAND
{Arrate.Munoz,Thierry.Blu,Michael.Unser}@epfl.ch
http://bigwww.epfl.ch

Abstract— **We present an optimal spline-based algorithm for the enlargement or reduction of digital images with arbitrary scaling factors [1].This projection-based approach is realizable thanks to a new finite difference method that allows the computation of inner products with analysis functions that are B-splines of any degree n. For a given choice of basis functions, the results of our method are consistently better that those of the standard interpolation procedure; the present scheme achieves a reduction of artifacts such as aliasing and blocking and a significant improvement of the signal-to-noise ratio.**

I. INTRODUCTION

Image resizing (magnification or reduction) is a typical operation in image processing [6]. Among its domains of application, we can highlight biomedical imaging, digital publishing and multimedia. The standard method for image size reduction consists in fitting the data with a continuous model and resampling the function at the desired rate [4]. A potential problem of the approach is the creation of aliasing and blocking artifacts.

In [9], it was demonstrated that it is possible to reduce these undesired effects by approximating the continuous model by its projection onto a given space prior to resampling. The approach may be viewed as a generalized version of anti-aliasing filtering where the prefilter is matched to the approximation space. This kind of formulation also yields higher quality results for image magnification with non-integer factors. Initially, the method was restricted to orthogonal projection and its weakness was the difficulty to perform an exact numerical implementation of the optimal prefilter for high order splines ($n \geq 2$). Lee *et al.* extended the approach using oblique projections in spline spaces to increase the computational speed [5]. The analysis function they used was a box (the B-spline of degree 0). Here, we present a futher extension that allows us to compute both oblique and orthogonal projections (least-squares approximations) for splines of any degree n.

What makes the approach feasible in this more general setting is the new finite difference method presented in Section 3; it allows an *exact* computation of the required inner products for analysis functions that are B-splines of any degree n. The method works for both reduction and magnification of images with an arbitrary scaling factor and any translation value. When the scale parameter is a power of two, it is equivalent to a wavelet processing because splines satisfy a two-scale difference equation [11].

II. PHILOSOPHY OF THE APPROACH

The image resizing problem can be solved using separable basis functions. Consequently, the complexity is reduced from 2-D to 1-D.

In order to simplify the description of our algorithm, it is advantageous to use a continuous signal processing representation of operators defined in the continuous domain. For that reason, we introduce *the digital-to-analog operator* $s(n) \mapsto s_\delta(x) = \sum_n s(n)\delta(x-n)$, where δ is Dirac's mass distribution.

The schematic continuous-time domain representation of the whole algorithm is given in Fig. 1.

All boxes denote convolutions; h, q and g are digital filters while the others are continuously-defined convolutions.

The affine transformation $s\left(\frac{x}{a}+b\right)$ is represented via a combination of shift (convolution with $\delta(x+b)$), and resizing $s(x) \to s\left(\frac{x}{a}\right)$ represented as $-\!\textcircled{\%a}\!-$.

We now describe the four main steps of the method.

A. Interpolation

The first step of the method is to take the discrete input data $s(k)$ and construct a continuous interpolating model $s(x) = \sum_k c(k)\varphi(x-k)$ where the $\varphi(x-k)$'s are some specified basis functions.For this purpose, we take the samples $s(k)$ and convolve them with an appropriate prefilter g to get the coefficients: $c(k) = (g * s)(k)$.

---

[1] A demonstration is available on the web at http://bigwww.epfl.ch/demo/resize

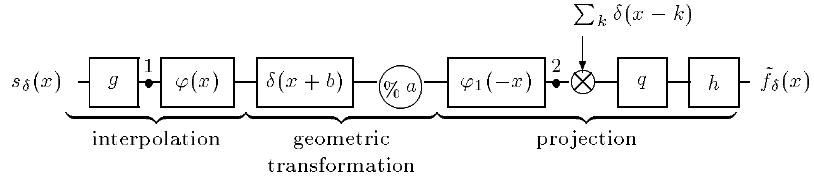

Fig. 1. General scheme for the projection based resizing method

The continuous-time function $s(x)$ is obtained by convolution with $\varphi(x)$.

The prefilter $G(z) = \frac{1}{\sum_k \varphi_k z^{-k}}$ is chosen such that the interpolation requirement is satisfied $s(x)\mid_{x=k} = \sum_l c(l)\varphi(k-l)$. In other words, $g(k)$ is the convolution inverse of the sequence $\varphi(k)$.

B. Affine transformation (conceptual step)

$$f(x) = s\left(\frac{x}{a} + b\right)$$

The function $s(x)$ is rescaled and shifted. If $a < 1$, it corresponds to image reduction; otherwise, the image is enlarged.

C. Projection based signal approximation

If we apply the standard method, we get the resized version just by resampling $f(x)$ at the integers.

Instead, we will think in terms of approximation theory. We would like to find the best approximation of $f(x) \in V_{\varphi_2}$ where $V_{\varphi_2} = \mathrm{span}_k\{\varphi_2(x-k)\}$.

We know that the least squares solution to this problem is the orthogonal projection of $f(x)$ onto V_{φ_2}. It is also possible, although suboptimal, to choose an oblique projection.

To calculate the projection, we use the analysis and synthesis function $\varphi_1(x)$ and $\varphi_2(x)$, which are not necessarily biorthonormal. Their cross-correlation sequence is given by

$$a_{12}(k) = \langle \varphi_1(x), \varphi_2(x-k)\rangle$$

First, we calculate the inner-products $c_1(k) = \langle f(x), \varphi_1(x-k)\rangle$ which is equivalent to prefiltering $f(x)$ with $\varphi_1(-x)$ and sampling thereafter.

If $a_{12}(k) \neq \delta_k$, the projection of $f(x)$ onto $V(\varphi_2)$ perpendicular to $V(\varphi_1)$ requires an additional digital filtering correction. Thus,

$$\begin{aligned} \tilde f(x) &= P_{2\perp 1} f(x) \qquad (1)\\ &= \sum_k (c_1 * q)(k)\varphi_2(x-k) \\ &= \sum_k c_2(k)\varphi_2(x-k) \end{aligned}$$

To satisfy the biorthogonality condition, the appropriate correction filter q is the convolution inverse of a_{12}: $q = a_{12}^{-1} \leftrightarrow \frac{1}{\sum_k a_{12}(k) z^{-k}}$.

This is equivalent to use the synthesis function $\phi_2 = q_\delta * \varphi_2(x-k)$, where $\varphi_1(x)$ and $\phi_2(x)$ are biorthogonal.

If $\varphi_1(x) \in V_{\phi_2}$ we have an orthogonal projection, otherwise we have an oblique projection.

Note that, in general, no set of coefficients $c_2(k)$ can be found expressing $f(x)$ as an exact linear combination of shifted synthesis functions. The function $\tilde f(x)$ is thus only an approximation of $f(x)$.

In the case of the orthogonal projection, we obtain a resized image with minimum loss of information (the approximation is optimal). In the case of an oblique projection, the approximation is usually only slightly suboptimal depending on the angle between $V_{\varphi_1} = \mathrm{span}_k\{\varphi_1(x-k)\}$ and $V_{\varphi_2} = \mathrm{span}_k\{\varphi_2(x-k)\}$ [8]. Moreover, the rate of convergence depends on the approximation order properties of the synthesis function alone; the analysis function has essentially no influence on the asymptotic approximation error [7].

D. Resampling the projection at the integers

Finally, we have to resample the projection at the integers ($\tilde f(l) = \tilde f(x)\mid_{x=l}$). The output is $\tilde f_\delta(x) = \sum_l \tilde f(l)\delta(x-l)$.

III. IMPLEMENTATION USING UNIFORM SPLINES

We now describe how this method can be implemented exactly using B-splines as basis functions.

A. Definitions

B-splines are basis functions for the polynomial spline functions. The explicit time-domain formula for the B-spline of degree-n is:

$$\beta^n(x) = \Delta^{n+1} * \frac{x_+^n}{n!} * \delta\left(x + \frac{n+1}{2}\right) \qquad (2)$$

with

$$x_+^n = \begin{cases} x^n & \text{if } x \geq 0 \\ 0 & \text{else.} \end{cases}$$

Δ is the finite difference operator: $\Delta = \delta(x) - \delta(x-1)$.

B-splines of different degrees are related by the differentiation formula

$$D^{n_1+1}\beta^{n+n_1+1}(x) = \Delta^{n_1+1}\beta^n\left(x - \frac{n_1+1}{2}\right) \qquad (3)$$

where D is the usual differentiation operator.

B. Why use splines?

We chose to work with splines because they have excellent approximation properties and belong to the class of functions that have minimum support for a given order n: This means that the computational complexity is minimized [3]. In addition, we have an *exact* formula for the computation of the inner products; this is typically not possible for other wavelet-like basis functions.

C. Projection method using splines

We now particularize the method to work with splines. Our algorithm has the following parameters: $\varphi(x) = \beta^n(x)$, $\varphi_1(x) = \beta^{n_1}(x)$, and $\phi_2 = a_{12}^{-1} * \beta^{n_2}$. This implies that $a_{12}(k) = \beta^{n_1+n_2+1}(x)\mid_{x=k}$.

D. Computing inner products

All the steps in the method are simple and can be performed using digital filtering, except for the computation of the scalar product for an arbitrary resizing factor a (not assumed to be a power of two as in the case of wavelets). In this section, we introduce an efficient method to compute inner products with B-splines of any degree n. The key formula is derived from (3):

$$\langle f(x), \beta^{n_1}(x-y)\rangle = \Delta^{n_1+1} D^{-(n_1+1)} f\left(y + \frac{n_1+1}{2}\right)$$

where $D^{-(n_1+1)}$ is the (n_1+1)-order integration defined by

$$D^{-(n_1+1)} f(x) = \frac{x_+^{n_1}}{n_1!} * f(x) \qquad (4)$$

In effect, we can compute the inner product rather simply by applying finite differences to the (n_1+1)-fold integral of f. What make an exact computation possible and tractable analytically is the fact that the (n_1+1)-fold integral of a spline is a spline with a corresponding increase of the degree. Specifically,

$$D^{-(n_1+1)} s(x) = \sum_k d(k) \beta^{n+n_1+1}\left(x - k - \tfrac{n_1+1}{2}\right)$$

with $d(k) = (\Delta^{-(n_1+1)} * c)(k)$, $s(x) = \sum_k c(k)\beta^n(x-k)$ and the inverse finite difference operator $\Delta^{-(n_1+1)} \leftrightarrow (1-z)^{-(n_1+1)} = \left(\sum_{k\geq 0} z^{-k}\right)^{n_1+1}$.

E. Derivation of the algorithm

We will give a graphical derivation of our algorithm by successive modification of the block diagram in Fig. 1.

We will use the exchange rules given in Figs. 2, 3 and 4.

Figs. 5, 6 and 7 give the manipulations that are neccesary to get the final implementation of the algorithm shown in Figure 8. In the diagram, we have extracted the operators between the marks 1 and 2 in Fig. 1 for simplicity.

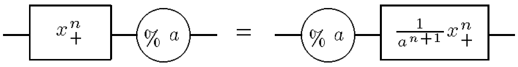

Fig. 2. Exchange rule for x_+^n

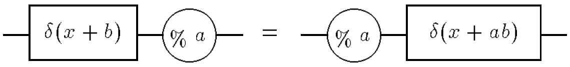

Fig. 3. Exchange rule for a shift by b

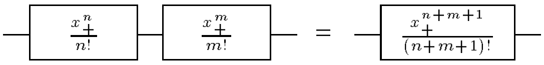

Fig. 4. Convolution of two x_+^n.

F. Practical implementation

The practical implementation of the method is based on the equivalent block diagram in Fig. 8. We now briefly summarize the main implementations steps:

i Digital prefiltering with the (causal and anti-causal) exponential filter $g = (b^n)^{-1}$ implemented in a recursive fashion to get $c(k)$ (interpolation coefficients) from $s(k)$ (input samples) [10].

ii (n_1+1)-running sums corresponding to the operator $\Delta^{-(n_1+1)}$.

iii Geometric transformation and resampling using spline interpolation model of degree $(n+n_1+1)$ (basis function $\phi(x)$).

iv (n_1+1)-centered finite differences, corresponding to the filter with z-transform $\Delta^{n_1+1} \leftrightarrow (1-z)^{n_1+1}$.

v Digital postfiltering with the sampled synthesis function $q * h(k) = \phi_2(x)\mid_{x=k}$.

In our implementation, the input signal is extended using symmetric mirror boundary conditions. A difficulty in the algorithm consists in keeping these conditions through steps 2 to 5. A clear description of how this is achieved will be given in [1].

G. Results

The performance of the algorithm was evaluated and compared with the standard one that fits the image with a spline of the same degree and then resamples it at the required rate. The original image (Fig. 9.a) was first reduced and then magnified back to its original size. Table 1 and 2 provide the final SNR measures. We observe that, for low order splines, the orthogonal projection method performs much better than the standard one (by more than 5 dB for $a = \sqrt{\pi}$ and $n = 0$). For higher order splines, the results are still significantly better for the optimal case with low reduction factors, and much better for high reduction factors.

In Tables 3 and 4, we see that oblique projection only brings a slight degradation when com-

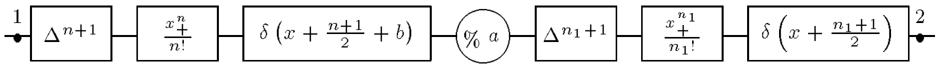

Fig. 5. Extract from figure 1. Substitution of $\beta^n(x)$ and $\beta^{n_1}(x)$ by their explicit expressions using equation (2).

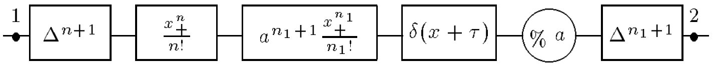

Fig. 6. Using the rule in Fig. 2, the boxes are reorganized in such a way that both the one-side power functions and the shifts are kept on the left side of the scale change a. In this way, the support of the spline kernel will not depend on it. In the graph, $\tau = \frac{n+1}{2} + \frac{n_1+1}{2a} + b$

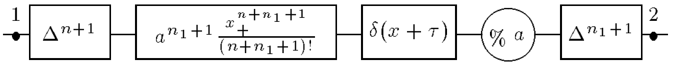

Fig. 7. Application of convolution rule in Fig. 4

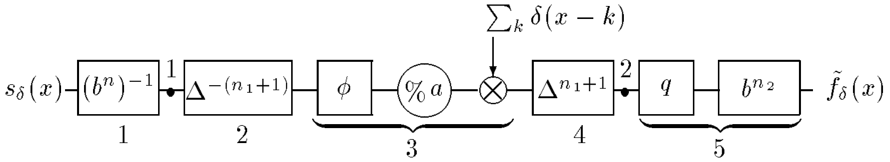

Fig. 8. Equivalent form of Fig. 1. Since $\Delta^{-(n+n_1+1)}\Delta^{n+n_1+1} = I$ we can re-write the algorithm in Fig. 7 using B-splines instead of polynomials. The spline kernel is $\phi(x) = a^{n_1+1}\beta^{n+n_1+1}(x+\tau)$. Since the filters in 4 and 5 are all digital, we are allowed to push the sampling step towards the resizing box.

n	optimal	standard
0	30.3524 dB	25.5223 dB
1	34.6657 dB	31.5789 dB
2	36.8907 dB	36.1911 dB
3	37.4675 dB	36.4505 dB

TABLE I

Least squares projection vs standard method. SNR results for $a = \sqrt{\pi}$

n	optimal	standard
0	24.7703 dB	22.8756 dB
1	27.0704 dB	25.7173 dB
2	27.8819 dB	26.6222 dB
3	28.1338 dB	26.5937 dB

TABLE II

Least squares projection vs standard method. SNR results for $a = \pi$

n_1	projection	
0	37.1769 dB	
1	37.3996 dB	
2	37.4663 dB	
3	37.4675 dB	← orthogonal projection

TABLE III

Oblique vs orthogonal projection. SNR results for $a = \sqrt{\pi}$ and $n = n_2 = 3$

n_1	projection	
0	27.7889 dB	
1	28.0458 dB	
2	28.1231 dB	
3	28.1338 dB	← orthogonal projection

TABLE IV

Oblique vs orthogonal projection. SNR results for $a = \pi$ and $n = n_2 = 3$

pared to the orthogonal scheme, while the computational complexity is proportional to with n_1. These results are consistent with the theory developed in [7] and [3]. An example of the type of improvement that can be obtained over standard interpolation is shown in Fig. 9; note that the signal model is the same in both cases: piecewise linear. We observe that by using projections and low order splines most of the high frequency information is kept in this case.

IV. Conclusions

In the paper, we have generalized Lee *et al.'s* method for signal resizing using both oblique and orthogonal projections. We have demonstrated that this method performs better that the standard interpolation and resampling approaches; its main advantage is aliasing supression.

A nice property of the present algorithm is that its complexity per output point does not depend on the scaling factor; this was not the case in an earlier version of the method [9].

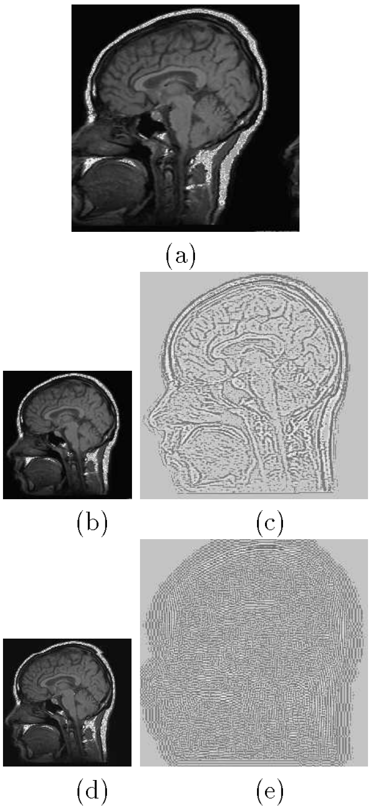

Fig. 9. Example of image reduction by a factor $a = \sqrt{\pi}$ using linear splines: (a) Original Magnetic Resonance (MR) image; (b) Reduced image using standard method; (c) Difference between the original signal and the enlarged version of the imagen (b); (d) Reduced image using orthogonal projection (e) Difference between the original and the enlarged version of the image (d).

References

[1] A. Muñoz, T. Blu and M. Unser, "Optimal resizing using projection method," *in preparation*, 1999.
[2] T. Blu and M. Unser, "Quantitative Fourier analysis of approximation techniques: Part I-Interpolators and projectors," *IEEE Trans. Signal Process.*, in press.
[3] T. Blu, P. Thévenaz and M. Unser, "Minimum support interpolators with optimum approximation properties," *IEEE Int. Conf. on Image Processing*, Chicago, IL, USA, October 4-7, 1998, vol. III, pp. 242-245.
[4] H. S. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-26, pp. 508-517, 1978.
[5] C. Lee, M. Eden and M. Unser "High-quality image resizing using oblique projection operators," in *IEEE Trans. Image Process.*, vol. 7, no. 5, pp. 679-692, May 1998.
[6] W. K. Pratt, Digital image processing, New York; Wiley, 1978.
[7] M. Unser, "Approximation power of biorthogonal wavelet expansions," *IEEE Trans. Signal Process.*, vol. 44, no. 3, pp. 519–527, March 1996.
[8] M. Unser and A. Aldroubi, "A general sampling theory for nonideal acquisition devices," *IEEE Trans. Signal Process.*, vol. 42, no. 11, pp. 2915-2925, November 1994.
[9] M. Unser, A. Aldroubi, and M. Eden "Enlargement or reduction of digital images with minimum loss of information," *IEEE Trans. Image Process.*, vol. 4, no. 3, pp. 247-258, March 1995.
[10] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part I- Theory and Part II- efficient design and applications," *IEEE Trans. Signal Process.*, vol. 41, no. 2, pp. 834–848, February 1993.
[11] M. Unser, A. Aldroubi, and M. Eden "The L_2 polynomial spline pyramid," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15, no. 4, pp. 364-379, April 1993.