TWO-LEVEL INTERPOLATION VLSI ARCHITECTURE FOR IMAGE RESAMPLING USING CUBIC SPLINES

Lírida ALVES DE BARROS and Nicolas DEMASSIEUX

Departement Electronique
ENST/Télécom-Paris
46, Rue Barrault, 75634 - Paris Cedex 13, France

RESUME

Cet article présente une architecture implémentant le ré-échantillonnage spatial avec des splines cubiques et basée sur une approche d'interpolation à deux niveaux. Des propriétés de l'algorithme sont prises en compte pour optimiser le compromis performance-coût. Il s'agit d'une architecture pouvant travailler en temps réel pour des fréquences de standards de télévision tels que le CCIR 601.

ABSTRACT

This work presents an architecture which implements spatial resampling with cubic splines interpolation. A two-level interpolation approach is used and properties of the algorithm are exploited to improve the performance-cost tradeoff of the hardware. The architecture can work in real-time for television standards frequencies such as CCIR 601.

1. INTRODUCTION

Resampling an image is equivalent to changing it from one sampling coordinate system to another. Examples of applications are image resizing and television standard conversion [1-2]. This transformation implies in the evaluation of the signal values at absents instants (or positions).

Resampling can be conceptually divided in two processes : interpolation of the discrete signal and sampling of the interpolated signal. Interpolation consists of fitting a continuous function to the discrete points in the digital signal and corresponds to low pass filtering. This interpolated signal may be sampled at any point. Sampling the interpolated signal is equivalent to interpolating the signal with a sampled interpolation function such as nearest neighbor, linear interpolation and cubic splines. The signal interpolation and sampling are usually combined and the signal is interpolated only at the points of the new sampling coordinate system.

In this work, we propose a high performance architecture which implements high quality spatial resampling with cubic splines interpolation.

This paper is organized as follows. In Section II, we describe the cubic spline algorithm for the interpolation. Sections III and IV respectively present the resampler architecture and the two-level interpolation approach proposed. Conclusions are drawn in Section V.

2. CUBIC SPLINES FILTERING

Let (x_k,y_k), $1 \le k \le p$ be p points of a 2D space. The cubic splines consist of $p-1$ cubic polynomials which join these points thus defining a continuously derivable function (Figure 1).

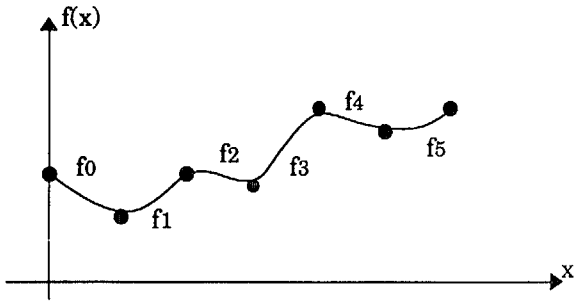

Figure 1 : Example of a spline consisting of 6 piecewise cubic polynomials.

With some constraints, this function may be used for interpolation [3]. Cubic splines are interpolation functions which have better performance than the nearest neighbor or linear interpolation algorithms, but they are computationally more expensive because they extend over four points and their coefficients are the response of a third order polynomial such as (Figure 2) :

$$h(x) = 1.5x^3-2.5x^2+1 \qquad \text{for } |x| \in [0,1]$$

$$h(x) = -0.5x^3-2.5x^2-4x+2, \qquad \text{for } |x| \in [1,2]$$

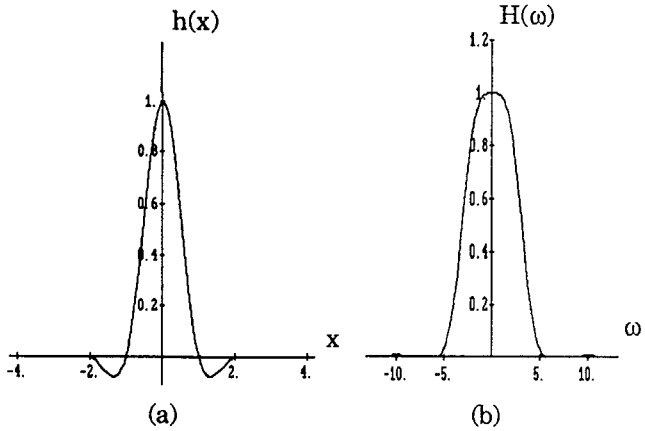

Figure 2 : (a) space and (b) frequency domain responses of a cubic spline function.

Cubic splines filters present a flat pass zone response and a good performance in the stop zone. In the next section, we discuss trade-offs for an efficient hardware implementation.

3. RESAMPLING ARCHITECTURE

Let L/M be the ratio of the sampling frequencies for resampling, where L and M are integers.

Figure 3 shows a zoom example. The image is oversampled by a factor L before a downsampling by a factor M. Oversampling of input image is carried out by inserting zero samples. We can see that the function needs to be sampled at L points minimum and that $4L$ coefficients are necessary (Figure 4).

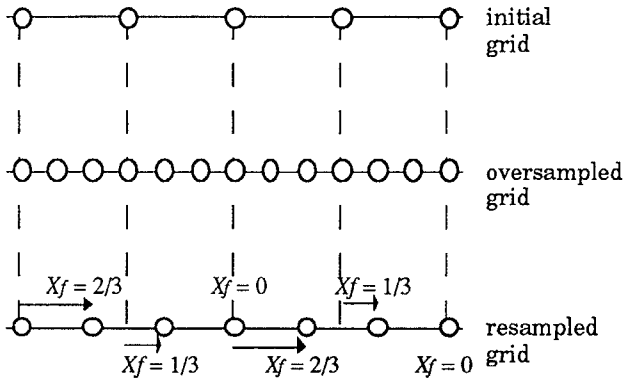

Figure 3 : Spatial resampling for L=3, M=2.

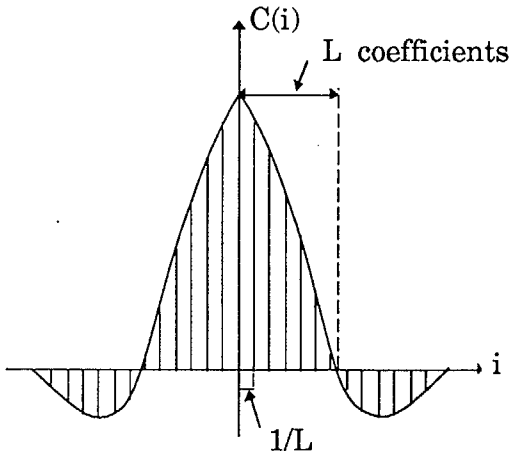

Figure 4 : Filter oversampled by L.

Filtering performed using $4L$ coefficients is not optimal because only four coefficients will be

multiplied by no zero samples. We use a more interesting hardware implementation based on a storage of coefficients (Figure 5).

The choice of the coefficients is based on the fractional distance (X_f, Y_f) between the pixel in the output sampling grid and the pixels in the input sampling grid. For a given L, there are L different values of X_f denoted X_{fi}, $i=1, \ldots, L$.

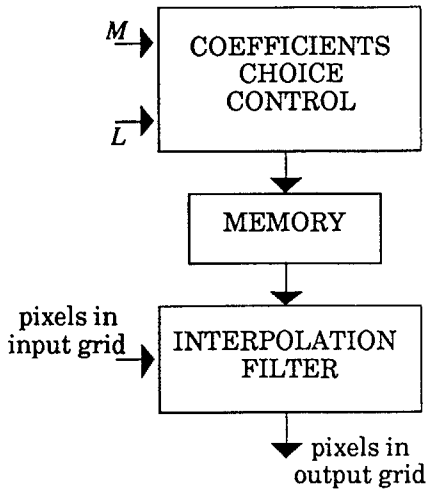

Figure 5 : Block diagram of the resampler.

The L values of Xf can be precomputed and stored in a memory. For 1D processing, the evaluation of the L values X_{if} may be performed as a function of the previous value X_{fi-1} iteratively as following :

$$X_{fi} = \frac{(LX_{fi-1} + M) mod_L}{L}$$

To make the resampler implementation independent of the value of L, we propose to store the interpolation cubic spline function oversampled by a factor V, where V is an integer. Because the cubic spline function is symmetric, permutation may be realized and only half of the total number of coefficients need to be stored in memory.

4. TWO-LEVEL INTERPOLATION APPROACH

If the filter is oversampled by V ($V>L$), the system is able to perform resampling by several L/M ratios. If L is an exact divisor of V (within the limits of the precision provided by the number of bits), coefficients are presents in the memory and can directly be exploited. Otherwise, a first level interpolation is performed for computing missing coefficients (Figure 6). We can use a neighbor or a linear interpolation algorithm.

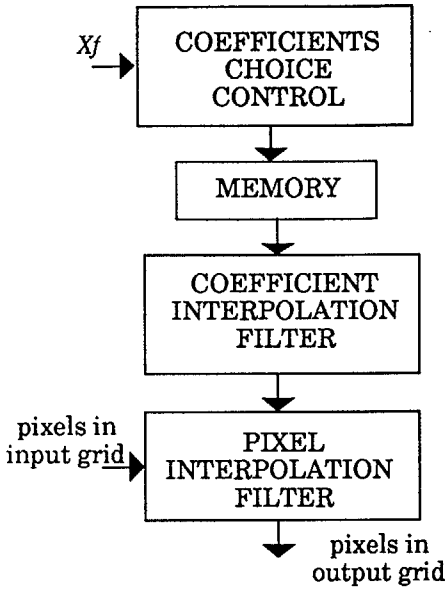

Figure 6 : Block diagram of the two-level resampler.

If a linear interpolation is used for the coefficients, the operation to be performed is :

$$C_{Xf} = C_i(1-X_{ff}V)+C_{i+1}X_{ff}V \quad \text{(I)}$$

or alternatively,

$$C_{Xf} = C_iV(C_{i+1} - C_i)X_{ff} \quad \text{(II)}$$

where X_{ff} is the fractional distance in respect to the nearest neighbor coefficient (Figure 7).

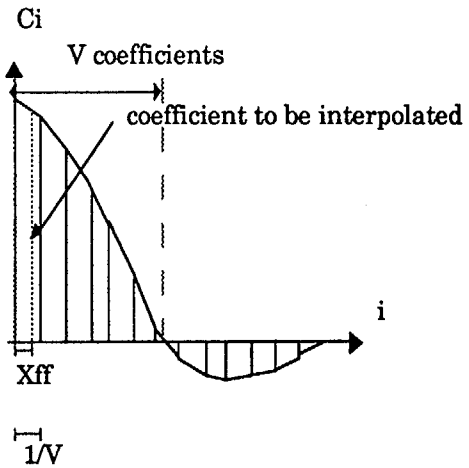

Figure 7 : First level interpolation.

For this first level of interpolation, if *(I)* is used, three multiplications per sample are necessary and only coefficients are stored. Using *(II)*, coefficients and products $V(C_{i+1}-C_i)$ are stored and only one multiplication per sample is necessary. This is a typical case of computation/memory trade-off depending on the relative cost of memory and arithmetic operations.

A comparison in silicon area for the two implementations shows that *(II)* has better performances. In fact, the gain in area grows with V (Figure 8) [4].

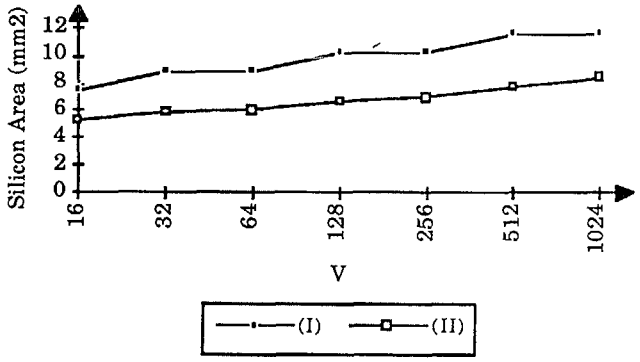

Figure 8: Comparison between interpolation using equations (I) and (II), 0.8 µm CMOS standard cells : N=4, Silicon area for architectures (I) and (II).

Memory is organized to allow simultaneous access to necessary data coefficients for the first level interpolation. For rectangular input and output 2D sampling grids, resampling may be implemented by cascading two 1D structures (Figure 9).

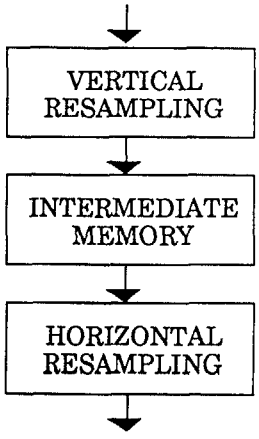

Figure 9 : 2D resampling by cascading two 1D structures.

If the image is horizontally scanned, vertical resampling before the horizontal one allows a reduction in storage area since the stored data have 8 bits (pixels) compared to the 16 bits data otherwise (intermediate results).

5.CONCLUSIONS

This work has proposed an implementation of an image resampler using cubic splines. A two-level interpolation approach is used and properties of the algorithm are exploited to improve the performance-cost tradeoff of the hardware. The storage of coefficients and the two-level interpolation improve the resampler performance and increases the number of possible values for the L/M resampling ratios. An implementation using a 0,8 µm CMOS technology and $V=16$ of a 2D separable filtering and equation *(II)* for the coefficient interpolation implies a 25 mm^2 silicon area for the heart of the resampler circuit. The circuit can work as a real-time image resampler with high image quality for television standards frequencies such as CCIR 601.

6. REFERENCES

[1] R. Paquin and E. Dubois, "A spatio-temporel gradient method for estimating the displacement field in time-varying imagery", Computer Vision, Graphics, and Image Processing, Vol. 21, pp. 205-221, 1983.

[2] G. Marcone and S. Miceli, "Two-dimensional interpolation for standard conversion between HDTV and common TV", Signal Processing of HDTV, L. Chiariglione (ed.), North-Holland, pp. 471-485, 1988.

[3] J.A. Parker, R.V. Kenyon and D.E. Troxel, "Comparison of interpolating methods for image resampling", IEEE Trans. Medical Imaging, pp. 31-39, (mar. 1983).

[4] L.A. Barros, "Architecture intégrée pour le ré-échantillonnage d'images animées", Internal Report ENST Paris, (jun. 1993).