International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
www.ijset.com, Volume No.1, Issue No.4, pg : 96-101 01 Oct. 2012

A Comparison of Hardware Implementations of Biorthogonal 9/7 2D-DWT: Lifting Structure versus Flipping Structure

J.Jeevitha, T.Mary Neebha, G.Vijila

Department of Electronics & Communication Engineering, Karunya University, Coimbatore
Email: jeevithajoseph@karunya.edu

***Abstract*—In this paper, we compare filter banks for calculating the DWT—the lifting method. We look at the traditional lifting structure and a recently proposed “flipping” structure for implementing lifting. Both filter bank structures are implemented on an Altera field-programmable gate array. Flipping structure can provide a variety of hardware implementations to improve and possibly minimize the critical path as well as the memory requirement of the lifting based discrete wavelet transform by flipping conventional lifting structures. The quantization of the coefficients plays an important role in the performance of all structures, affecting both image compression quality and hardware metrics. We design several quantization methods and compare the best design for both approaches. The results show that for the same image compression performance, the flipping structure gives the smallest and fastest, low-power hardware.**

Index Terms—Discrete wavelet transform (DWT), field-programmable gate array (FPGA), flipping, lifting, quantization.

I.INTRODUCTION AND BACKGROUND

Since the discrete wavelet transform (DWT) was deduced by Mallat [1], many researches on wavelet-based signal analysis and compression have derived fruitful results due to the well time-frequency decomposition. The DWT has been adopted as the transform coder in emerging image coding standards, such as JPEG2000 still image coding and MPEG-4 still texture coding. However, more arithmetic operations may be required for the DWT than the discrete cosine transform (DCT) because of the filter computation. Contrary to the block-based DCT, the DWT is basically frame-based. The huge amount of the memory size and access bandwidth becomes a bottleneck of the implementation for two-dimensional (2-D) DWT [2]. For one-dimensional (1-D) DWT, several convolution-based architectures have been proposed [3]–[6] because the DWT computation is intrinsically the filter convolution. After the appearance of the lifting scheme [7] and a factorization method of lifting steps [8], the lifting scheme has been widely used to reduce the computation of DWT and the control complexity of boundary extension. Some lifting based architectures have been proposed [9]–[11], which are all based on the direct implementation of the factorized lifting scheme. On the other hand, several architectures have been proposed to implement 2-D DWT, including the direct method, semi-systolic routing, systolic routing, systolic-parallel, parallel-parallel, single input multiple data (SIMD), and one-level 2-D architectures [4]–[6]. According to the evaluation in [2], RAM-based 2-D DWT architectures are preferred due to their feasibility and regularity. Moreover, the line-based method has been proposed to shrink the internal memory requirement from a frame size to a few line buffers with proper memory management. Using the lifting scheme to construct VLSI architectures for DWT could outperform using convolution in many aspects, but the critical path of lifting-based architectures is potentially longer than that of convolution-based ones. Although pipelining can reduce the critical path, it will prolong the latency and require more registers for the 1-D architecture such that larger memory size is required for the 2-D line-based architecture. For all implementations quantized coefficients that give the same image compression performance are used. Hardware properties of the implementations are then compared.

II. DWT ARCHITECTURE

There have been many VLSI architectures proposed for hardware implementation of DWT. For 1-D DWT, the architectures are mainly convolution-based and lifting-based. The direct and line-based architectures are the most feasible implementations for 2-D DWT. Hardware performance metrics include size or cost, throughput, and the energy required to process an image. Before implementation, hardware cost is estimated in terms of T, the total number of nonzero terms used when writing all the lifting coefficients in canonical-signed-digit (CSD) format [7]. After implementation, the hardware cost is measured directly in terms of the number of logic elements used. Compression performance is evaluated by comparing the compressed and original images using PSNR.

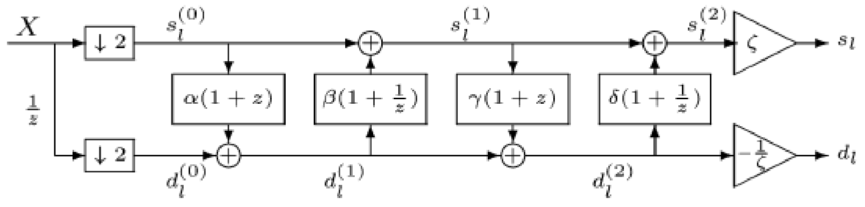

Fig. 1. Lifting structure of the biorthogonal 9/7 DWT (analysis).

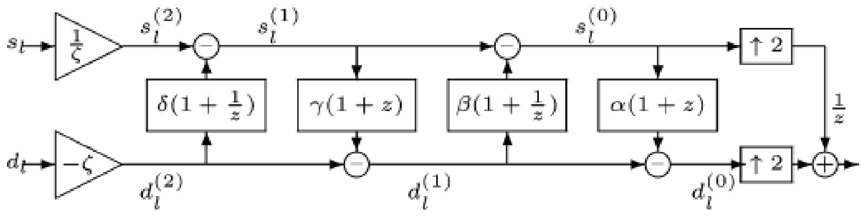

Fig. 2. Lifting structure of the biorthogonal 9/7 DWT (synthesis).

A. Lifting Based Structure

According to [2] any DWT filter bank of perfect reconstruction can be decomposed into finite sequence of lifting steps. The decomposition corresponds to a factorization for the polyphase matrix of the target wavelet filter into a sequence of altering upper and lower triangular matrices and a constant diagonal matrix which can be expressed as follows:

$h(z) = h_e(z^2) + z^{-1} h_o(z^2)$

$g(z) = g_e(z^2) + z^{-1} g_o(z^2)$

$$P(z) = \begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix}$$

$$P(Z) = \begin{bmatrix} 1 & a(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ b(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & c(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ d(1+z) & 1 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

The lifting approach replaces the 9-tap and 7-tap filters with a cascade of 2-tap symmetric filters. Lifting is a poly-phase structure; one stage of analysis is depicted in Fig. 1 and one stage of synthesis is shown in Fig. 2. The unquantized lifting coefficients α, β, γ, δ and ζ, obtained by factoring the 9/7 poly-phase matrix [3], are irrational. Hence, they need to be approximated for implementation in fixed-point hardware. A set of rational lifting coefficients was obtained by moving two of the four analysis LPF zeros away from z=-1.

In terms of image compression performance, the rational coefficients generate peak signal-to-noise ratio (PSNR) values nearly identical to the irrational coefficient PSNR values (in infinite precision) [5]. However, the rational coefficients are more readily implemented in hardware; all except γ, ζ and 1/ζ have finite binary representations and can be represented exactly.

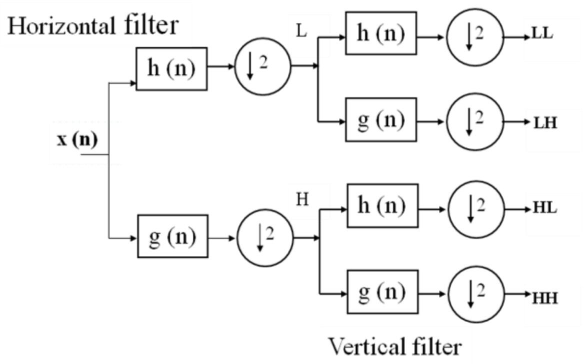

Fig :3. Implementation Flow of 2-D DWT

B. Flipping Based Structure

The “flipping” structure proposed in [6], is an alternative structure for implementing the lifting approach. This structure has the advantage of reduced critical path compared to the traditional lifting structure. Additionally, when the rational lifting coefficients are used, the stage which employs can be flipped so that its filtering can be achieved exactly without need for approximation. Figs. 3 and 4 show the analysis and synthesis stages for the “flipping” implementation.

A fast hardware implementation of the lifting structure using rational coefficients requires that α, ζ, and 1/ζ be quantized (i.e., represented in fixed-point); similarly for the convolution structure, the coefficients in h_n and f_n must be quantized. The flipped structure with rational coefficients requires quantization of ζ only.

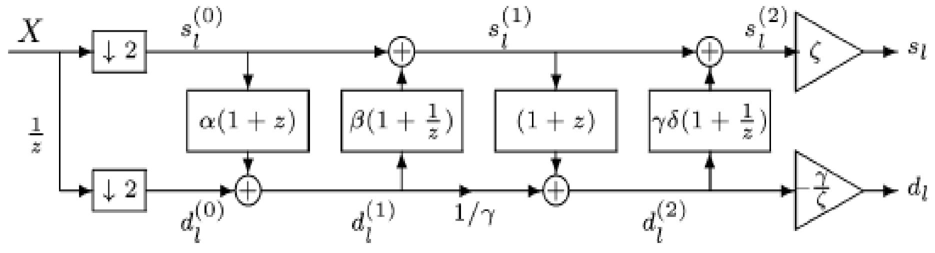

Fig. 4. Flipping structure for the biorthogonal 9/7 (analysis).

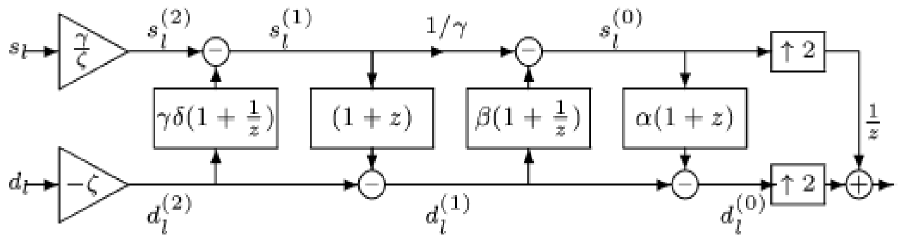

Fig. 5. Flipping structure for the biorthogonal 9/7 (synthesis).

III. QUANTIZATION METHODS

A. Quantization Objectives

The goal of a filter bank is PR; the synthesis section should exactly invert the analysis section so that the reconstructed image will equal the original image. This is possible when aliasing and distortion are avoided. Aliasing is routinely avoided by deriving the high pass filters (HPFs) from the LPFs). PR then reduces to satisfying the no-distortion condition. The infinite precision 9/7 filters meet the no-distortion condition; however, filters implemented in fixed-point hardware do not. The aim of coefficient quantization is to minimize hardware requirements while maintaining image compression performance.

Coefficient quantization affects image compression performance in a number of ways. It changes the magnitude response of the filters in the filter bank. It has been shown that for compression performance, the magnitude response of the quantized analysis filters must closely approximate the magnitude response of the unquantized 9/7 filters. In addition, quantization perturbs the location of the zeros of the filters. F(z) has four zeros at that are critical to image compression performance. Perturbation of these zeros causes dc leakage through the analysis HPF G(z), and degrades the subjective quality of the compressed and reconstructed images by introducing the checker-boarding artifact [2].

Fast filter banks are multiplierless implementations where multiplication is performed by shifting and adding. In a multiplierless implementation, the number of nonzero CSD digits T can be used to control the trade-off between hardware cost and image compression performance. T corresponds roughly to hardware cost; it represents the number of terms that must be added to perform the filter computation. In general, the higher the T, the closer the quantized PSNR values are to the unquantized PSNR values; conversely, smaller T implies less hardware but worse PSNR. Quantization can be posed as a problem of allocating a fixed number T of CSD terms to the filter coefficients or lifting coefficients.

For the convolution and the lifting approaches, implementations that use the “best” quantized coefficient set are compared. The “best” set of quantized coefficients is the set that requires the smallest T. Here reasonable” is defined by the absence of checker boarding in the compressed image and PSNR degradation (from the unquantized case) of no more than 0.1 dB at compression ratios ranging from 1:1 to 100:1.

B. Lifting Implementation

The lifting structure is inherently orthogonal: when synthesis immediately follows analysis (i.e., no compression), analysis is exactly inverted regardless of the quantized values of the lifting coefficients. PR is lost after coefficient quantization. Fixed-point approximations of the rational lifting coefficients require approximations for γ, ζ, $1/\zeta$; α, β and δ can be represented exactly with a few CSD terms.

In LSGC, the gain factors ζ and $1/\zeta$ include $\sqrt{2}$ and $1/\sqrt{2}$; we avoid implementing these factors in hardware by lumping them together and bit-shifting the DWT coefficients after one level of row and column filtering. γ and ζ are quantized to γ' and ζ' by allocating the remaining Ts to each coefficient. We use gain compensation to modify from its original value of 1.25 for better reconstruction of an image’s dc component;

Forward Lifting transform	
Splitting	$S\,l(0) = x2l$ $d\,l(0) = x2l+1$
Predict 1 Update 1	$d\,l(1) = d\,l(0) + \alpha\,[S\,l(0) + S\,l{+}1(0)]$ $S\,l(1) = S\,l(0) + \beta\,[d\,l(1) + d\,l{-}1\,(1)]$
Predict 2 Update 2	$d\,l(2) = d\,l(1) + \gamma\,[S\,l(1) + S\,l{+}1(1)]$ $S\,l(1) = S\,l(0) + \delta\,[d\,l(2) + d\,l{-}1\,(2)]$
Normalization	$S\,l(3) = \zeta\,S\,l(2)$ $d\,l(3) = d\,l(2) / \zeta$

Fig:6 Forward Lifting transform

Lifting Inverse transform	
Undo Normalization	$S\,l(2) = S\,l(3) /\zeta$ $d\,l(2) = d\,l(3)\ \zeta$
Undo Predict 1 Undo Update 1	$S\,l(1) = S\,l(2) - \delta\,[d\,l(2) + d\,l{-}1\,(2)]$ $d\,l(1) = d\,l(2) - \gamma\,[S\,l(1) + S\,l{+}1(1)]$
Undo Predict 2 Undo Update 2	$S\,l(0)= d\,l(1) - \beta\,[d\,l(1) + d\,l{-}1\,(1)]$ $d\,l(0) = d\,l(1) - \alpha\,[S\,l(1) + S\,l{+}1(1)]$
Merge	$X2l = S\,l(0)$ $x2l{+}1 = d\,l(0)$

Fig: 7. Inverse Lifting transform

Architecture	Critical path delay	Multi pliers	Adders	Registe rs
Convolution based (Adder tree +Symmetric property)	$T_m + 4\,T_a$	9	14	7
Lifting Scheme	$4T_m + 8T_a$	4	8	4
Lifting Scheme (with pipelining-4 stages)	$T_m + 2T_a$	4	8	10

Fig: 8. Comparison – (9, 7) Filter

C. Flipping Implementation

The filtering stage employing γ in the traditional lifting implementation can be implemented exactly if this stage is "flipped" as described in [11].

Forward flipping transform	
Splitting	$S\,l(0) = x2l$ $d\,l(0) = x2l+1$
Predict 1 Update 1	$d\,l(1) = d\,l(0) + \alpha\,[S\,l(0) + S\,l{+}1(0)]$ $S\,l(1) = S\,l(0) + \beta\,[d\,l(1) + d\,l{-}1\,(1)]$
Predict 2 Update 2	$d\,l(2) = (1/\gamma)\,d\,l(1) + [S\,l(1) + S\,l{+}1(1)]$ $S\,l(1) = S\,l(0) + \gamma\,\delta\,[d\,l(2) + d\,l{-}1\,(2)]$
Normalization	$S\,l(3) = \zeta\,S\,l(2)$ $d\,l(3) = \gamma\,d\,l(2) / \zeta$

Fig: 9.Forward flipping transform

Inverse Flipping transform	
Undo Normalization	$S\,l(2) = \gamma\,S\,l(3) /\zeta$ $d\,l(2) = d\,l(3)\ \zeta$
Undo Predict 1 Undo Update 1	$S\,l(1) = S\,l(2) - \gamma\,\delta\,[d\,l(2) + d\,l{-}1\,(2)]$ $d\,l(1) = d\,l(2) - [S\,l(1) + S\,l{+}1(1)]$
Undo Predict 2 Undo Update 2	$S\,l(0)= (1/\gamma)S\,l(1) - \beta\,[d\,l(1) + d\,l{-}1\,(1)]$ $d\,l(0) = d\,l(1) - \alpha\,[S\,l(1) + S\,l{+}1(1)]$
Merge	$X2l = S\,l(0)$ $x2l{+}1 = d\,l(0)$

Fig: 10.Inverse Flipping transform

Once flipped, the coefficients 1/γ(1.25) and γδ (0.375) have to be implemented in hardware; both these coefficients have finite binary representations and require no approximation. Thus flipping the third stage in the traditional lifting structure enables all the lift and update steps to be implemented exactly when rational lifting coefficients are used. The gain stages for the low pass and high pass branches can then be quantized using MUA, LSCG [9].

IV. RESULTS

The traditional lifting and the flipping designs were used to compute the one-level, non-expansive, symmetric extension DWT of three eight-bit grayscale images from the ITU standard digitized image set [10]. PR (1:1) and three compression ratios (8:1, 32:1 and 100:1) are examined.

Digital hardware for one analysis level of each structure has been implemented. A multiplier less architecture is used; computations are organized in a fully pipelined tree of carry save adders with a ripple carry adder at the base.

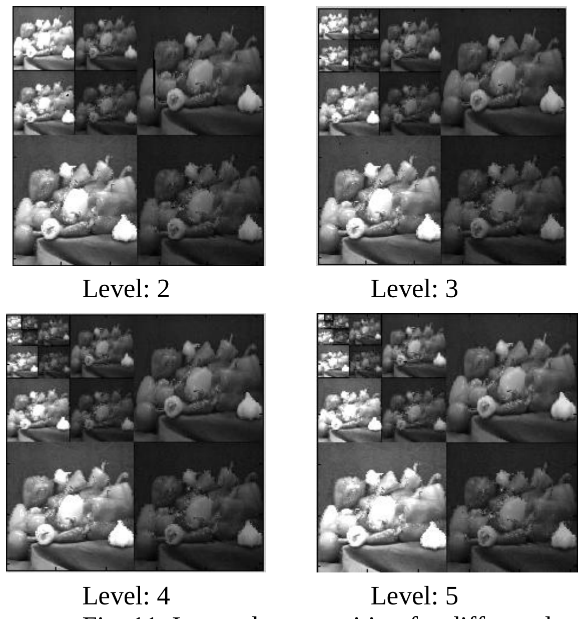

Fig: 11. Image decomposition for different levels

The maximum possible operating frequency is obtained via timing analysis and indicates system throughput. An estimate of dynamic energy required by the analysis side to perform the computations of a one-level, two-dimensional DWT of a 512x512 block of random pixels is shown. The actual hardware cost depends on a number of factors such as bit widths of the filter coefficients and the architecture of the filter bank.

Architecture	Multiplier	Adder	Critical path
Lifting + no pipe	4	8	$4T_m + 8T_a$
Lifting + 4 stages	4	8	$T_m + 2T_a$
Flipping + no pipe	4	8	$T_m + 5T_a$
Flipping + 3 stages	4	8	$T_m + T_a$

Fig: 8. Comparison of Several Architectures – (9, 7) Filter

The lifting structure employs four filters that are in cascade. The bit widths of internal signals grow with each level of filtering and so the adders used to implement multipliers in the subsequent filters are wider and slower. Thus the lifting structure is bigger and slower than the convolution structure. However, the flipping structure employs coefficients with very narrow bit widths. As a result, the bit width of internal signals in the flipping structure grows slowly compared to the traditional lifting structure. Thus, narrower adders can be used; this explains the significantly smaller number of logic elements required for the flipping structure. Due to its narrow bit widths, the flipping structure results in the highest throughput and lowest energy consumption compared to the traditional lifting implementations.

Images	Lifting			Flipping		
	Computation time (sec)	Entropy	PSNR	Computation time (sec)	Entropy	PSNR
Cameraman	2.5475	7.0097	31.9864	2.3434	7.0097	53.6748
Football	2.1885	6.7129	30.8654	1.8593	6.7129	54.8776
Peppers	2.1256	6.9908	32.8744	1.9380	6.9908	52.7944
Tape	2.4000	6.7421	29.9887	2.0000	6.7421	49.2673
Pears	2.1720	7.1463	33.9887	1.9531	7.1463	50.8765

Fig: 13. Lifting Vs Flipping for images of size 256x256

Image	Lifting			Flipping		
	Computation time (sec)	Entropy	PSNR	Computation time (sec)	Entropy	PSNR
Cameraman	12.9875	7.0097	29.234	12.3634	7.0097	36.9844
Football	13.5885	6.7129	27.455	13.2453	6.7129	37.9982
2Peppers	13.3456	6.9908	31.083	13.1257	6.9908	39.7562
Tape	13.7460	6.7421	36.975	12.8978	6.7421	40.0372
Pears	13.5817	7.1463	20.682	13.0531	7.1463	32.6882

Fig: 14.Lifting Vs Flipping for images of size 512x512

IV. CONCLUSION

In this paper, an efficient VLSI architecture, called flipping structure, is compared with traditional lifting structure. The problem of serious timing accumulation for the traditional lifting-based architectures is addressed by flipping some computing units such that the critical path can be greatly reduced. Thus, the flipping structure can minimize the size of the internal buffer for line-based DWT architectures under specified timing constraints. For the same image compression performance, the flipping structure based on rational coefficients resulted in the smallest, fastest hardware and the lowest energy consumption compared to the traditional lifting structure. In addition to reducing the critical path, it can also provide well-featured architectures.

V. REFERENCES

[1] Wei Zhang, Zhe Jiang, Zhiyu Gao, Yanyan Liu "An Efficient VLSI Architecture for Lifting-Based Discrete Wavelet Transform" IEEE Circuits and Systems Society, Volume: 59 , Issue: 3Page(s): 158 – 162,March 2012

[2] Jain, R., Panda, P.R."An Efficient Pipelined VLSI Architecture for Lifting-Based 2D-Discrete Wavelet Transform" Circuits and Systems, 2007, IEEE International Symposium, pp1377 – 1380, 27-30 May 2007

[3] ITU-T Recommandation T.800. JPEG2000 Image Coding System—Part I, ITU Std (2002, Jul.). [Online].Available: http://www.itu.int/ITU-T/

[4] G. Strang and T. Nguyen, Wavelets and Filter Banks, 1st ed. Wellesley, MA: Wellesley-Cambridge Press, 1996. Acoustics, Speech, Signal Processing, vol. 5, May 2004, pp. 193–196.

[5] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," J. Fourier Anal. Appl, vol. 4, no. 3, pp. 247–269, 1998.

[6] Chao-Tsung Huang, Po-Chih Tseng, Liang-Gee Chen "Flipping structure: an efficient VLSI architecture for lifting-based discrete wavelet transform" Volume: 1
Page(s): 383 - 388 vol.1, 2002.

[7] D. Tay, "A class of lifting based integer wavelet transform," in Proc. IEEE Int. Conf. Image Processing, vol. 1, 2001, pp. 602–605.

[8] Z. Guangjun, C. Lizhi, and C. Huowang, "A simple 9/7-tap wavelet filter based on lifting scheme," in Proc. IEEE Int. Conf. Image Processing, vol. 2, 2001, pp. 249–252.

[9] C. Huang, P. Tseng, and L. Chen, "Flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform," in Proc. IEEE Asia-Pacifc Conf. Circuits and Systems, vol. 1, 2002, pp. 383–388.

[10] K. A. Kotteri, A. E. Bell, and J. E. Carletta, "Design of multiplier less, high-performance, wavelet filter banks with image compression applications," IEEE Trans. Circuits Syst. I, Reg Papers, vol. 51, pp. 483–494, Mar. 2004.

[11] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "VLSI architecture for discrete wavelet transform based on B-spline factorization," in Proc. IEEE Workshop Signal Processing Systems, Aug. 2003, pp. 346–350.

[12] S. Barua, K. A. Kotteri, A. E. Bell, and J. E. Carletta, "Optimal quantized lifting coefficients for the 9/7wavelet,in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing, vol. 5, May 2004, pp. 193–196.

[13] ITU-T Recommendation T.24. Standardized Digitized Image Set, ITU Std (1998, Jun.). [Online]. Available: http://www.itu.int/ITU-T/

[14] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," IEEE Trans. Circuits Syst. Video Techno., vol. 6, pp. 243–250, Jun. 1996.