On-Line Adaptation in Image Coding with a 2-D Tarp Filter

Patrice Simard, David Steinkraus, and Henrique Malvar
Microsoft Research
One Microsoft Way, Redmond, WA 98052
{patrice, v-davste, malvar}@microsoft.com

Abstract

On-line adaptation to nonstationary distributions is essential to good performance in image coding. Fixed-size contexts (with adaptive tables) are also widely used, in conjunction with arithmetic encoders, in state-of-the-art codecs. In contrast, we propose a simple two-dimensional filter that directly outputs the probability distribution function (PDF) estimate necessary to drive an adaptive arithmetic encoder. The filter is isotropic, in the sense that the impact of a previously encoded bit depends only on its value and distance to the bit to be coded. Surprisingly, this simple filter yields results comparable to or better than JPEG2000. It also brings an interesting distinction between on-line and off-line learning, and their relative importance in compression.

1. Introduction

Many wavelet signal coders [1] are based on the structure shown in Figure 1. The wavelet coefficients are quantized (divided by a quantization step Q and then rounded to nearest integers), and the resulting indices are encoded without loss by an entropy encoder. The entropy coder typically uses a probability distribution function (PDF) over the quantized values, and the number of bits it produces is close (typically within less than 1%) to the entropy of the PDF. The PDF is computed adaptively from previously encoded coefficients (kept in the "store" box), by counting frequencies for each context, for example.

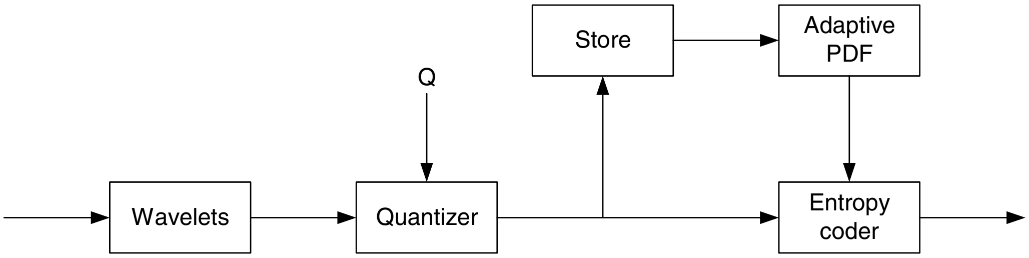

Figure 1. Simplified block diagram of a wavelet-based signal coder.

Higher compression is obtained by increasing Q, which decreases the entropy of the PDF because more coefficients quantize to zero.

The "Adaptive PDF" box typically consists of a table that associates a probability to the values of the pixel to be encoded, as a function of a context of neighboring pixels. Interestingly, it is widely believed that the context of neighboring pixels captures essential information, such as edges and patterns, and that this information is necessary to achieve high compression.

This paper challenges that view by showing that an isotropic filter can achieve similar compression rates for grayscale images, with little optimization. By isotropic we mean that the filter response has circular symmetry centered on the pixel to be predicted. In other words, the filter is the same regardless of the (causal) direction through which we look at past data. This isotropic nature of the filter prevents it from capturing information such as edge positions or other complex patterns.

In the next section we describe the filter and its implementation. In Section 3 we compare our performance to that of JPEG2000 on a popular grayscale image set. An extension to binary (fax) image coding is discussed in Section 4, and conclusions are presented in Section 5.

2. The Tarp Filter

Consider the task of processing a long series of binary symbols and estimating the probability that the next bit will be equal to one. A typical assumption is that such probability varies smoothly. Such an assumption is generally true for wavelet images, especially in places where there is little activity. Then the probability of the next symbol being one is small and highly correlated with how far away the other nonzero symbols are. We assume that the probability of the predicted symbol being one is proportional to an exponential decay of the distance of the current symbol to the other ones. That allows for probability estimation with simple recursive filters, as we discuss next.

2.1. A simple 1-D filter for density estimation

Let us start by considering a one-dimensional scenario. We can build an estimate of the probability that the symbol to be predicted is one by the simple first-order recursive filter:

$$p[t] = ap[t-1] + (1-a)v[t] \tag{1}$$

where $p[t]$ is the estimate of the probability of getting a one for position $t + 1$, $v[t]$ is the observed value (0 or 1) at position t, and a is a learning rate parameter between 0 and 1, which controls how quickly the probability estimate adapts to the data.

It is easy to show that p is the convolution of $v[t]$ with the filter impulse response $f[t] = a^t(1-a)$ for $t \geq 0$, and 0 otherwise. It is also easy to show that f is a probability density function (bounded by 0 and 1, with the integral summing to unity), and that for every one in the data stream $v[t]$, $p[t]$ is a sum of Parzen windows, shaped by f. From the properties of Parzen windows [2], we know that p/n (where n is the number of ones) is a legitimate density estimation function with the expected value of p being equal to the true probability of the source. The parameter a controls adaptation speed, i.e. the size of the equivalent Parzen window. It can also be viewed as a smoothing factor; the noisier the data, the higher we should set the value of a.

The main advantage of this algorithm is simplicity, since very few operations are involved. However, to achieve good performance we need to adjust the parameter a appropriately, and that depends on the probability distribution of v, which we usually do not know a priori. However, in many compression systems filters similar to f are used with good results, when v comes from quantized wavelet coefficients [1].

2.2. Combining three 1-D filters to create the Tarp 2-D filter

If we generalize the simple filter discussed before to 2-D, scanning order becomes an issue. One possibility is to use a 1-D filter, scanning the pixels in blocks, or to use a Peano scan [3] to capture some of the 2-D correlation. We chose instead the usual raster scanning for images, line-by-line, from left to right, trying to extract the maximum information from all previously seen pixels (i.e. located in a line above or located to the left of the current pixel). This is done by using three 1-D filtering steps. The first filter goes from left to right and is similar to the 1-D filter described above. The second filter goes from right to left, and is done after each full line has been processed. The resulting probabilities are kept in a buffer. The third filter goes from top to bottom for each column, using the probability computed in the previous line. The resulting 2-D filter impulse response is shown in Figure 2 and Table 1. Because of the appearance of that response when plotted in a 3-D graph, we call it a “Tarp” filter.

The computation of the Tarp filter is summarized by two sets of equations. The first set implements the top-to-bottom and left-to-right 1-D filter:

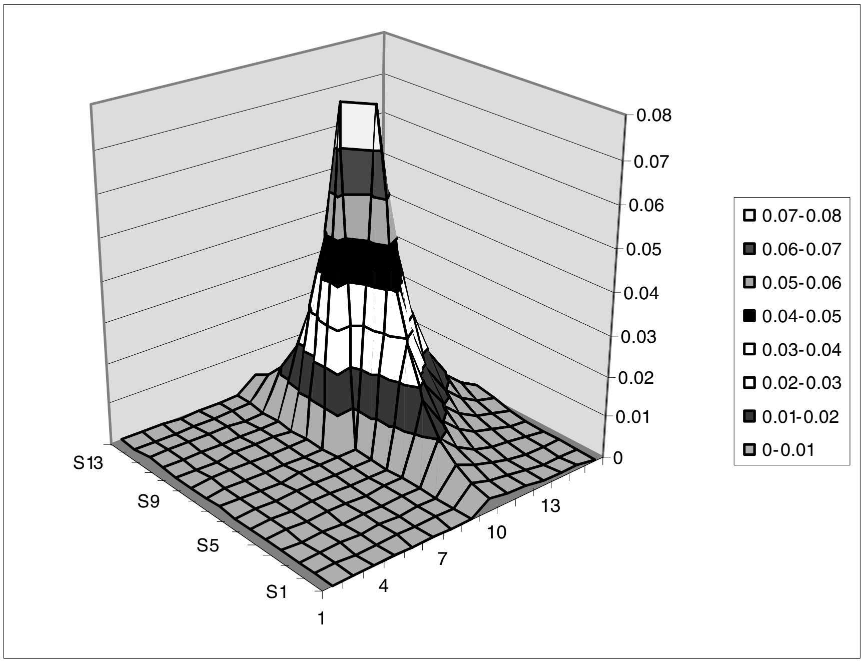

Figure 2. Typical 2-D response of a “Tarp” filter.

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	$\boxed{0}$	0.1250	0.0625	0.0313	0.0156	0.0078
0.0078	0.0156	0.0313	0.0625	0.1250	0.0625	0.0313	0.0156	0.0078	0.0039
0.0039	0.0078	0.0156	0.0313	0.0625	0.0313	0.0156	0.0078	0.0039	0.0020
0.0020	0.0039	0.0078	0.0156	0.0313	0.0156	0.0078	0.0039	0.0020	0.0010
0.0010	0.0020	0.0039	0.0078	0.0156	0.0078	0.0039	0.0020	0.0010	0.0005

Table 1. Typical impulse response of a 2-D Tarp filter, for a pixel centered at the rectangle (for a = 0.5).

$$\begin{aligned} p[i,j] &= a\left(p_1[i,j-1] + p_2[i-1,j]\right) \\ p_1[i,j] &= ap_1[i,j-1] + \frac{(1-a)^2}{2a}v[i,j] \\ p_2[i,j] &= p_1[i,j] + ap_2[i-1,j] \end{aligned} \tag{2}$$

where p is the true probability of v being one (it can't depend on $v[i,j]$, which has not yet been seen), and i and j are respectively the line and column indices. The probability p is computed from the left probability estimate $p_1[i,j-1]$ and the above probability estimate $p_2[i-1,j]$. Next, the left probability estimate is updated using the actual value $v[i,j]$. Note that the normalizing coefficient $(1-a)^2/2a$ ensures that the sum of all the probabilities is 1. Finally, the probability $p_2[i,j]$ is updated for use in the next row computation. Note that $p_2[i,j]$ implements a vertical decay of the probabilities for each column. It also requires that one row of estimates $p_2[i-1,:]$ be kept in memory.

Once a full row has been coded, the row probabilities p_2 are updated one more time by running a third filter p_3 backwards (from right to left) on the line that was just processed:

$$\begin{aligned} p_2[i,j] &= p_2[i,j] + ap_3[i,j+1] \\ p_3[i,j] &= ap_3[i,j+1] + \frac{(1-a)^2}{2a}v[i,j] \end{aligned} \tag{3}$$

This whole computation can be viewed as an effective way to implement a convolution with the function in Figure 2. Note that the support of this function is strictly causal (only depends on previously seen pixel). The convolution can be viewed as a sum of Parzen windows (with the shape depicted in Figure 2), and the result is an accurate density estimator for 1s of the binary stream $v[i,j]$. As with the 1-D filter previously described, the Tarp filter ensures that the expected value of the probability estimate p matches that of the signal, under stationary assumptions.

The initial conditions at the boundaries are given by:

$$\begin{aligned} p_2[-1,-1] &= \varepsilon \frac{1+a}{2a} \\ p_2[-1,j] &= \frac{1+a}{1+a^2} p_2[-1,j-1] \\ p_2[i,-1] &= \frac{1-a}{1+a} p_2[i-1,0] \end{aligned} \tag{4}$$

where ε=0.001 is an a priori estimate of the probability of $v[i,j]$ being equal to one.

2.3. Using the Tarp filter for coding/encoding

The Tarp filter can be used to encode bit planes of quantized wavelet coefficients, as in [4], for example. For a given bit plane, $p[i,j]$ is the probability that the symbol to be encoded, $v[i,j]$, is equal to one. On both the encoder and the decoder, an arithmetic encoder encodes (decodes) $v[i,j]$ using the probability $p[i,j]$ computed using (2). Once $v[i,j]$ has been processed, the next probability $p[i,j+1]$ is computed, and so on until the end of the line. At the end of the line, a backward pass is made over the whole line using (3). The next line is then processed.

An appropriate value for a must be determined empirically. For bit-plane encoding of quantized wavelets coefficients, we found that a = 0.6 is experimentally optimal (in the sense of producing the lowest bit rate at the output of the entropy encoder) for all planes and subbands. Each bit plane can be encoded independently, although some refinement strategy can also be used (if the higher bits are one, a probability of 0.5 might be a better estimate than $p[i,j]$ (although $p[i,j]$ would probably be very close to 0.5 already). The sign bit is encoded when needed using 0.5 as the probability, as usual (although for some images sign prediction could lead to improvement [5]).

It is also possible to use the Tarp filter on the magnitude of the quantized wavelet coefficients. The drawback of doing so is that encoding is no longer progressive. The two main advantages are speed, since only one Tarp filter needs to be updated, instead of one per bit plane, and slightly better compression. The Tarp filter can also be interpreted differently than as a probability estimator; the Tarp is essentially computing a 2-D running average of the magnitude of coefficients. For each magnitude, we need to associate a PDF over all the possible quantized values, resulting in a table indexed by magnitude and quantized value. The table can be adaptive or pre-stored. In that sense, the Tarp filter is similar to the transform coefficient energy estimator of the EQ coder [6]. Our preliminary results using a Tarp on the magnitude indicates an improvement of 2% to 3% over the results reported in Section 3. Such improvement comes with a significant increase in complexity, and requires adaptive tables and modeling the PDF with a smooth curve function of the magnitude (without this modeling, the table has too many entries to adapt quickly enough).

3. Experimental Results

We have adapted our bit-plane wavelet image coder [4] to use Tarp by replacing the context-adaptive Run-Length-Rice encoder by a Tarp filter followed by a nonadaptive arithmetic encoder [7].

Our first experiment was to encode the gray scale images from the same Kodak grayscale image set used in [4], and compare the results to those of JPEG, PWC [4] and JPEG2000. For all codecs, intermediate bitstream files were generated and decoded, so the compressed file size included bookkeeping and format overheads. For each image, the usual 7-9 biorthogonal wavelet transform is computed, with 5 subband levels. On each band (LH, HL, HH), the Tarp filter is run independently on each bit plane. The probability estimate generated by each bit-plane Tarp is then used by the arithmetic encoder to encode each bit-plane symbol. For each non-zero symbol, the sign is coded independently in raw mode, i.e. without any compression.

A typical set of results for the image set are show in Table 2, for a peak signal-to-noise ratio (PSNR) of 40.0 dB (which leads to high visual quality, with essentially imperceptible quantization artifacts for most images). Similar results are obtained at other PSNR

Image	**JPEG**	**PWC** [4]	**EZ-HLBT** [8]	**JPEG2000**	**Tarp**
1	2.64	2.52	2.43	2.45	2.40
2	1.27	1.04	1.08	1.06	0.98
3	0.75	0.62	0.60	0.57	0.56
4	1.30	1.05	1.04	1.06	0.98
5	2.50	2.33	2.32	2.24	2.19
6	1.92	1.74	1.68	1.69	1.62
7	0.94	0.78	0.81	0.72	0.72
8	2.80	2.58	2.55	2.45	2.45
9	0.90	0.75	0.73	0.68	0.68
10	1.06	0.87	0.86	0.81	0.80
11	1.75	1.55	1.54	1.50	1.45
12	1.03	0.85	0.83	0.79	0.78
13	3.32	3.15	3.10	3.17	3.01
14	2.26	2.01	2.00	2.00	1.89
15	1.10	0.93	0.94	0.89	0.85
16	1.35	1.14	1.10	1.11	1.06
17	1.25	1.02	1.03	0.99	0.95
18	2.26	2.02	1.98	2.02	1.91
19	1.64	1.43	1.38	1.39	1.33
20	0.93	0.78	0.85	0.72	0.72
21	1.63	1.44	1.44	1.42	1.35
22	1.69	1.49	1.45	1.50	1.40
23	0.56	0.38	0.40	0.37	0.35
24	2.02	1.87	1.87	1.80	1.76
Average	**1.62**	**1.43**	**1.42**	**1.39**	**1.34**

Table 2. Compression performance, in bits per pixel, of several grayscale image codecs, for a PSNR of 40 dB.

settings. For comparison with embedded zerotree-based coders, we have also included results for the EZ-HLBT (embedded zerotree with hierarchical lapped biorthogonal transform) codec [8].

We note that the Tarp-based codec had the best compression performance for that group of codecs, in the Kodak image set. That was indeed a surprising result, since we were not expecting to obtain better results than JPEG2000, for example, which uses more sophisticated context modelling. The improvement over JPEG2000 is only a bit above 3%, but that is quite surprising, in view of the simplicity of the Tarp approach. Note that the Tarp codec is not using contextual information other than the distance of non-zero bits with respect to the currently coded bit.

4. Tarp on Binary Images

In this experiment, we used the Tarp filter on binary (fax) images. This experiment shows the limitation of the Tarp filter. Binary images differ substantially from wavelet images in the sense that unlike wavelets coefficients, they are not residual images. The neighboring pixels are highly correlated. We used images f00_200 to f10_200 from the 200 dpi standard CCITT test images [9] as a test set. When encoded with Tarp ($a = 0.14$), the results are 66% worse than JBIG2 (with clustering turned off), and 49% worse than BLC [9]. In other words, the Tarp filter yields poor performance compared to context-based methods for images in which neighboring pixel values are highly correlated. It also suggests that the Tarp's "blindness" to direction may be more useful for predicting the variance rather than the actual value of a pixel.

4.1. A Hybrid Tarp and Context Predictor

Since the obvious limitation of Tarp is its inability to take advantage of contextual patterns, we attempted to remedy to this deficiency by building a hybrid predictor using both a Tarp filter predictor and a contextual predictor. The two competing predictors each output a PDF to be used by the arithmetic encoder. The first PDF is computed using the Tarp filter. Since this predictor adapts quickly but has little information about how the bits are arranged around the bit to be predicted, the second predictor is built to complement the first. It uses a large context to capture local patterns and is trained off-line (since on-line adaptation is already captured by the Tarp filter). The context has 18 bits and has the shape shown if Figure 3.

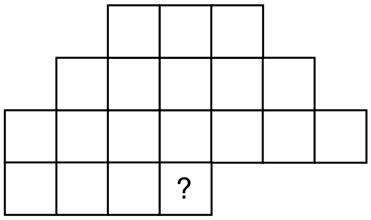

Figure 3. Context pattern used for the experiments in Section 4.

The context predictor is trained off-line on about 20 fax-like images (different from those in the test set) and the PDF of the next bit (whose location is marked by a question mark in Figure 3) is the frequency computed on the training set as a function of the context. The PDFs from the two predictors compete and the winner is determined by a gate function, as depicted in the Figure 4.

Several gating functions were tried, such as picking the predictor with minimum entropy, picking Tarp if the Tarp entropy was less than a threshold, etc. We have also tried to learn the gating function from the training set. Let P_Tarp and P_context be respectively the probability outputted by the Tarp predictor and the context predictor. We built a large bucketized table (P_Tarp by P_context) and found the optimal gating function on the training set for each combination (P_Tarp, P_context). We then use this gating function on the test set. In any case, the data needed for the decisions are obtained from previously sent pixels, so there is no need for the encoder to send the gating value. All our attempts yielded similar results. When predicting wavelet coefficients, using Tarp is always better than our hybrid attempts, while when predicting binary fax images, using the context predictor alone is always the better solution.

For predicting wavelet coefficients, we used 12 images from the Kodak set for training, and the remaining 12 for testing. Combining off-line training with on-line training is not a new concept, and has been used successfully in [10]. We ran experiments on the LH0 band. The Tarp alone was better than any of our other hybrid attempts. The only exception was when we used an oracle which always chose a posteriori the best predictor: the oracle was 38% better than Tarp alone. Context alone was 83% worse than Tarp alone.

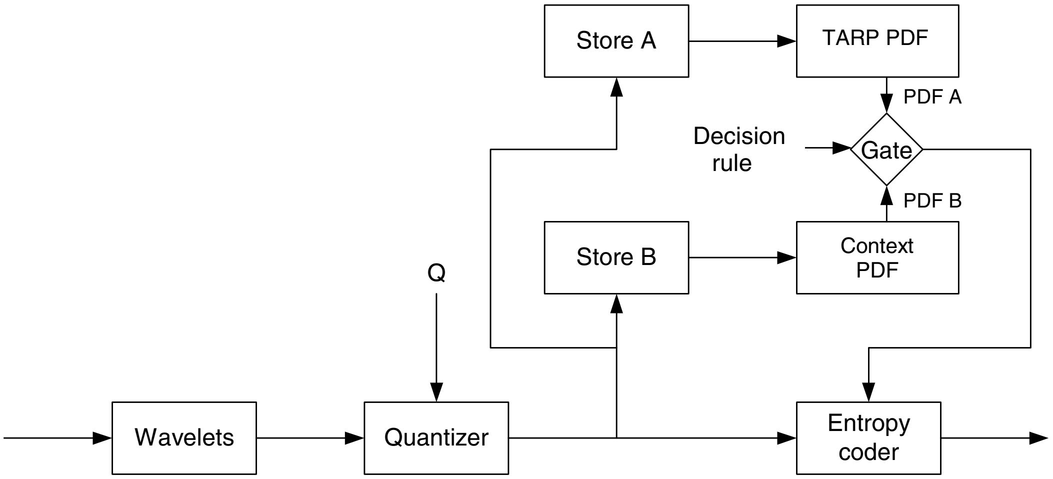

Figure 4. Basic system for encoding with two distinct PDF predictors. A gating function based on previously encoded data determines which PDF estimate to use for each coefficient.

For the binary case, we used binary images from our own scans for training. In this case, we found that always using the Context Predictor is the optimal gating strategy. It is about 11% worse than JBIG2 and 2% worse than BLC. With an oracle, the hybrid solution beats everything else by 43%.

Both hybrid results were surprising to us. We thought that surely contextual information would help the Tarp filter in predicting wavelet coefficients. The answer was negative, which indicates that with wavelet coefficients, accurate topological information does not provide useful information to the Tarp filter. We surmise that wavelet coefficients have little correlation to justify predicting coefficient values, but that the variance can be accurately estimated by an isotropic filter such as the Tarp. The second experiment with binary images is equally surprising, at the other extreme. On-line adaptation such as provided by the Tarp filter is of little benefit when compared to a pure context-method trained off-line. This suggests that with a larger training set and larger tables, we could probably match state-of-the-art performance with a pure off-line method. In that case, the table can be aggressively optimized (using decision trees, for instance), because most contexts in high-dimensional space are too rare to impact compression and can therefore be pruned out.

5. Conclusion

In this paper we introduce a very simple codec based on 2-D “Tarp” filtering. Because the filter is isotropic, it cannot incorporate accurate contextual information. Surprisingly, this simple filter yields state-of-the-art compression performance and even beats JPEG2000 on image compression. This shows that accurate contextual information is not necessary for good performance in image compression. This conclusion is reinforced by an experiment in which we tried to combine the Tarp filter with a predictor which uses extensive contextual information and is trained off-line. We could not improve on our results using the second predictor, which further suggests that context contains little information not already captured by the Tarp.

In another set of experiments, we use the Tarp filter to encode binary images. In this setting, the Tarp filter yielded poor performance, suggesting that for binary images, contextual information is essential for good performance. This conclusion was reinforced by trying to combine the Tarp filter with a predictor that uses extensive contextual information. In that case, the optimal combination was to completely ignore the Tarp filter and base every prediction using context, further suggesting that Tarp was making poor use of context information.

The Tarp filter seems to be better for predicting images where the coefficients have little correlation, such as wavelet coefficients. We surmise that the strength of the Tarp filter comes from its ability to smoothly track variance, as opposed to the ability to predict the coefficient values.

References

[1] P. Simard and H. S. Malvar, “A wavelet coder for masked images,” *Proc. IEEE Data Compression Conf.,* Snowbird, UT, pp.93–102, Mar. 2001.

[2] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* New York, NY: Wiley and Sons, 1973.

[3] R. J. Stevens, A. F. Lehar, and F. H. Preston, “Manipulation and presentation of multidimensional image data using the Peano scan,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 5, pp. 520–533, Sept. 1983.

[4] H. S. Malvar, “Fast progressive wavelet coding,” *Proc. IEEE Data Compression Conf.,* Snowbird, UT, pp. 336–343, Mar. 1999.

[5] A. Deever and S. S. Hemami, “What's your sign?: efficient sign coding for embedded wavelet image coding,” *Proc. IEEE Data Compression Conf.,* Snowbird, UT, pp. 273–282, Mar. 2000.

[6] S. LoPresto, K. Ramchandran, and M. T. Orchard, “Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework,” in *Proc. IEEE Data Compression Conf.,* Snowbird, UT, pp. 221–230, Mar. 1997.

[7] A. Mo at, R. M. Neal, and I. H. Witten, “Arithmetic coding revisited,” *ACM Trans. Information Systems,* vol. 16, pp. 256–294, 1998.

[8] H. S. Malvar, “Lapped biorthogonal transforms for transform coding with reduced blocking and ringing artifacts,” *Proc. Int. Conf. Acoustics, Speech, and Signal Processing,* Munich, Germany, pp. 2421–2424, April 1997.

[9] H. S. Malvar, “Fast adaptive encoder for bi-level images,” *Proc. IEEE Data Compression Conf.,* Snowbird, UT, Mar. 2001.

[10] C. J. C. Burges, P. Simard, and H. S. Malvar, “Improving wavelet image compression with neural networks,” *Microsoft Research Tech. Rep.* MSR-TR-2001-47, June, 2001. (Abstract in *Proc. IEEE Data Compression Conf.,* Snowbird, UT, pp. 486, Mar. 2001.).