# Correspondence

## Adaptive Transforms for Image Coding Using Spatially Varying Wavelet Packets

Kannan Ramchandran, Zixiang Xiong,
Kohtaro Asai, and Martin Vetterli

*Abstract*—**We introduce a novel, adaptive image representation using spatially varying wavelet packets (WP's). Our adaptive representation uses the fast *double-tree* algorithm introduced in [1] to optimize an operational rate-distortion (R-D) cost function, as is appropriate for the lossy image compression framework. This involves jointly determining which filter bank tree (WP frequency decomposition) to use, and when to change the filter bank tree (spatial segmentation). For optimality, the spatial and frequency segmentations must be done jointly, not sequentially. Due to computational complexity constraints, we consider quadtree spatial segmentations and binary WP frequency decompositions (corresponding to two-channel filter banks) for application to image coding. We present results verifying the usefulness and versatility of this adaptive representation for image coding using both a first-order entropy rate-measure-based coder as well as a powerful space-frequency quantization-based (SFQ-based) wavelet coder introduced in [11].**

### I. Introduction

Linear transforms for signal decorrelation and energy compaction have received considerable attention, especially with applications to image coding. As is well known, the *signal-dependent* Karhunen–Loéve transform (KLT) provides an optimal basis for block decorrelation but is computationally overwhelming, as it has no specific structure that can be exploited. The computational advantage of the signal-independent discrete cosine transform (DCT) basis, and its good approximation to the KLT for the important class of highly correlated Markov-1 sources [2], have led to its popularity in image coding standards. These transforms, however, have a block constraint and, thus, fail to exploit interblock correlations. One solution is to use a set of overlapped bases (overlapped blocks), such as the lapped orthogonal transform (LOT) [3]. Recent studies on filter banks and wavelets have developed another flexible construction of an orthonormal basis, which is free from the block constraint [4], [5]. In this context, subband decompositions offer nonblock-constrained linear expansion bases using computationally efficient multirate filter bank structures.

The wavelet transform [4], [5] is a popular transform due to its ability to offer useful time-frequency localizations[1] [increasing time resolution with higher frequencies, and increasing frequency resolution with lower frequencies—see Fig. 1(a)]. However, if the time-frequency characteristics of a given signal do not match the time-frequency localizations offered by the wavelet, we have a mismatch, which results in an inefficient decomposition. Arbitrary subband decomposition trees, introduced in [6] as wavelet packets (WP's)

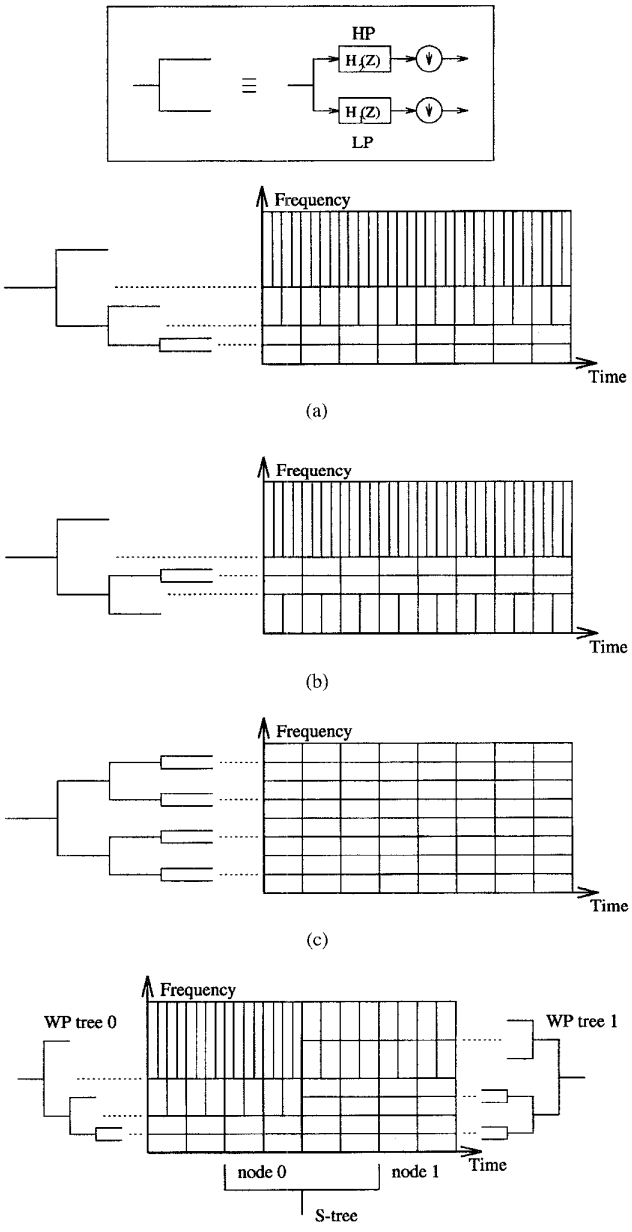



Fig. 1. Typical members of WP family expressed in tree forms and in the tiling of the time-frequency plane. (a) Wavelet decomposition. (b) Example of arbitrary WP decomposition. (c) STFT-like decomposition. (d) Example of joint time-frequency segmentation using the double-tree algorithm.

Manuscript received December 16, 1994; revised September 20, 1995. Portions of this paper were presented at the 1993 Proceedings of CISS, The John Hopkins University, March 1993. This work was supported by the National Science Foundation under Grant NSF 92-122 (RIA-94). The associate editor coordinating the review of this correspondence and approving it for publication was Dr. A. R. Reibman.

K. Ramchandran and Z. Xiong are with the University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA.

K. Asai is with Mitsubishi Electric Corporation, Japan.

M. Vetterli is with the University of California, Berkeley, CA 94704 USA.

Publisher Item Identifier S 1057-7149(96)04538-1.

[1] We will use "time-frequency" and "space-frequency" synonymously where there is no chance of confusion, though the former refers to temporally described one-dimensional (1-D) signals like speech, while the latter to spatially described two-dimensional (2-D) signals like images.

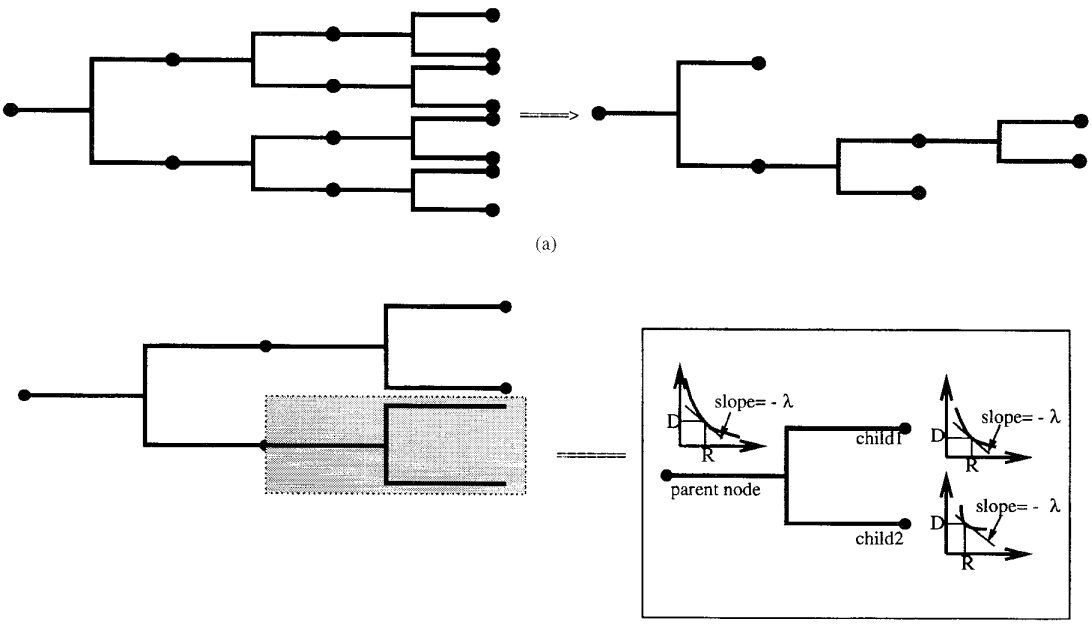



Fig. 2. Single-tree algorithm finds the best tree-structured WP basis for a given signal. (a) The algorithm starts from the full STFT-like tree, and prunes back from the leaf nodes to the root node until the best pruned subtree is obtained. (b) At each node, the split-merge decision is made according to the criterion: Prune, if $J(\text{parent node}) \leq [J(\text{child1}) + J(\text{child2})]$.

[see Fig. 1(b)], conceptually represent an elegant generalization of wavelets, including both the full-band short-time Fourier transform-like (STFT-like) tree [see Fig. 1(c)] as well as the wavelet tree in their rich library of transforms. WP trees have several attractive features. They are very efficiently implementable using filter bank structures. They further offer a very large library of orthogonal linear expansions (for example, a depth-5 two-dimensional (2-D) WP decomposition has a library of $5.60 \times 10^{78}$ bases!), while, more importantly, lending themselves to efficient fast algorithms to search this rich library for the basis that performs best with respect to an arbitrary input signal. A fast algorithm based on a somewhat *ad hoc* entropy criterion was formulated in [6]. A recently introduced fast algorithm [7] called the *single-tree algorithm* addresses this best WP basis selection in a lossy compression framework (of minimizing coding distortion for a target bit rate or vice versa). The optimized WP is thus *a signal-dependent* basis like KLT, while being nonblock-oriented and computationally attractive due to efficient subband filter bank structures.

Although WP's offer considerably more flexibility than the wavelet transform for dealing with classes of signals with diverse or unknown time-frequency characteristics, they still represent "static" time-frequency decompositions or nonadaptive frequency-tree splittings. A number of important signal classes (for applications like speech, images, and video signals) that typically exhibit time-varying characteristics are handled more efficiently if the frequency decompositions are less rigid. For image representations, adaptivity can be obtained, for example, by performing a spatial segmentation and adapting the WP frequency decomposition to each spatial segment. This leads to *spatially adaptive WP's*, i.e., WP decompositions that adapt spatially in order to best match the image's locally varying space-frequency characteristics.

In this work, extending our work of [8], we address the question of spatially varying WP's for image coding applications. We will restrict ourselves to tree structures in this search for spatial segmentation (using quadtree structures) and for frequency decomposition (using separable WP trees that are binary in each dimension), which lend themselves to computationally efficient implementation algorithms. Although we will consider separable two-channel filter banks in this paper, the algorithm can be easily extended to arbitrary $M$-channel banks as well. We show how a fast *double-tree* algorithm (introduced in [1]) can be used to do this joint segmentation [see Fig. 1(d)]. Experimental results showing the good performance of the double tree algorithm on 1-D signals like speech and synthetic signals were presented in [1]. Here, we present results of the application of the double-tree algorithm to image coding. Our experimental results include coders that use a first-order entropy-based rate measurement (approximated well by arithmetic coding, for example), as well as a more powerful space-frequency quantization-based (SFQ-based) "zerotree" coder [11], whose performance is one of the best in the published image coding literature.

It is important to point out that we are interested in an image- and rate-adaptive representation, with the spatially adaptive double-tree representation introduced here being a rich superset of the "static" single-tree WP representation of [6] and [7]. The double tree therefore reserves the right to degenerate to a single tree representation, if that should be the optimal choice for a given image. Indeed, it can be used as a tool to measure the "nonstationarity" in space-frequency characterization of a test image, and has applications beyond coding, which is, however, the thrust of this paper. We will see from our experiments that the best double-tree decomposition of some test images degenerates to a single tree, while for other images this is not true. Furthermore, even for the same image, the space-frequency tiling is rate sensitive, as is desirable for a compression application. The diversity of this tool is, therefore, that it includes both the wavelet and all the single tree representations as special cases, and performs no worse than them, at the price of increased complexity. In practice, it offers improved performance for a variety of tested images at a variety of coding bit rates of interest.

The paper is organized as follows. Section II summarizes the single-tree WP algorithm of [7] and addresses the double-tree algorithm for finding the joint time and frequency decomposition.

TABLE I
COMPARISON OF CODING RESULTS FOR DIFFERENT TREE ALGORITHMS (WAVELET TREE, SINGLE TREE, AND DOUBLE TREE) AT VARIOUS BIT RATES FOR THE STANDARD 512 × 512 LENA, BARBARA, AND HOUSE IMAGES, WHEN A SINGLE UNIFORM QUANTIZATION STEP SIZE IS USED FOR ALL HIGHPASS BANDS. (FIRST-ORDER ENTROPY IS USED AS RATE MEASURE)

| | | Lena | Barbara | House | | Lena | Barbara | House | | Lena | Barbara | House |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) |
| WT | 1.0 | 39.24 | 34.35 | 36.11 | 0.5 | 36.16 | 29.67 | 31.83 | 0.25 | 33.04 | 26.02 | 28.88 |
| ST | 1.0 | 39.32 | 36.28 | 36.73 | 0.5 | 36.26 | 31.67 | 32.11 | 0.25 | 33.24 | 28.06 | 28.97 |
| DT | 1.0 | 39.36 | 36.55 | 37.84 | 0.5 | 36.29 | 31.73 | 32.49 | 0.25 | 33.40 | 28.16 | 28.97 |

TABLE II
COMPARISON OF CODING RESULTS FOR DIFFERENT TREE ALGORITHMS (WAVELET-TREE, SINGLE-TREE, AND DOUBLE-TREE) AT VARIOUS BIT RATES FOR THE STANDARD 512 × 512 LENA, BARBARA, AND HOUSE IMAGES BY USING BIORTHOGONAL WAVELETS AND SPACE-FREQUENCY QUANTIZATION (ALSO INCLUDED AS A REFERENCE THE PSNR NUMBERS OF SHAPIRO'S CODER IN [10] FOR BARBARA AND LENA)

| | | Lena | Barbara | House | | Lena | Barbara | House | | Lena | Barbara | House |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) | Rate (b/p) | PSNR (dB) | PSNR (dB) | PSNR (dB) |
| EZW[10] | 1.0 | 39.55 | 35.14 | | 0.5 | 36.28 | 30.53 | | 0.25 | 33.17 | 26.77 | |
| WT | 1.0 | 40.51 | 36.94 | 37.61 | 0.5 | 37.31 | 31.16 | 32.80 | 0.25 | 34.24 | 27.12 | 29.70 |
| ST | 1.0 | 40.55 | 37.13 | 37.83 | 0.5 | 37.40 | 31.53 | 32.88 | 0.25 | 34.27 | 27.32 | 29.70 |
| DT | 1.0 | 40.55 | 37.58 | 38.01 | 0.5 | 37.40 | 32.13 | 33.00 | 0.25 | 34.27 | 27.85 | 29.70 |

Space-frequency quantization is briefly summarized in Section III. Section IV presents image coding applications using the double-tree algorithm, addressing practical design considerations in coupling SFQ with the double-tree image representation. Coding results with novel space-frequency tilings produced by the double-tree algorithm are also shown.

## II. FAST WP-BASED TREE ALGORITHMS

### A. Single-Tree Algorithm

We provide a brief description of the single-tree algorithm, referring the reader to [7] for details. The aim of the algorithm is to search for the best WP tree-structured decomposition for the whole *unsegmented* signal from a library of bases. Toward that end, we need two things: a cost function for basis comparison and a fast algorithm to do the search. For the first, we use the Lagrangian cost function $J = D + \lambda R$; the Lagrangian multiplier $\lambda$ here is an equalizing factor that balances rate and distortion, as is desired for image coding applications.

We now explain the fast single-tree algorithm for the case of 1-D signals as shown in Fig. 2. We first grow the the full subband or STFT-like tree [see Fig. 2(a)] to some fixed depth for the whole signal, and populate each tree node with the Lagrangian cost (for a fixed $\lambda$) of encoding each associated subband (or the original signal for the root node). Note that multiple quantization modes can be incorporated in the cost generation process, and the rate can be either estimated from the first-order entropy of the quantization indexes or obtained from the output bit rate of a real coder.

We then make a pruning decision at each parent node based on the comparison of the cost of the parent and the summation of the costs of its children. If the parent cost is smaller, we prune its children; otherwise, we keep the children [see Fig. 2(b)]. Note that the above tree-pruning process starts recursively from the leaf nodes, and proceeds toward the root node. In the end, a pruned subtree is obtained for a fixed $\lambda$.

Finally, the best $\lambda$ is searched to meet a given target bit rate through an iterative bisectional method, in which the searching interval of $\lambda$ is successively shrunk at each iteration until convergence.

The single-tree algorithm can be summarized as follows.

- Grow a full STFT-like tree to some fixed depth.
- For a fixed $\lambda$, populate each node of the full tree with the best Lagrangian cost $D + \lambda R$.
- Prune the full tree recursively, starting from the leaf nodes.
- Iterate over $\lambda$ bisectionally to meet the target bit rate.

The name for this algorithm is derived from the fact that a single (frequency) tree is optimally pruned. For a signal of size $N$, the computational complexity of this algorithm is $O(N \log N)$.

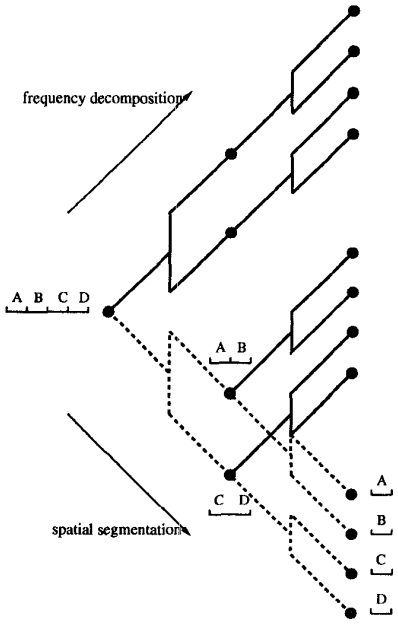



Fig. 3. Full double tree of depth-2 for a 1-D signal. Dotted lines represent spatial tree, where as solid lines represent frequency tree. To find the best time segmentation and the best WP tree for each segment jointly, we first run the single-tree algorithm for each possible dyadic time segment (the whole signal, its halfs, quarters, and so on) to search for the best WP tree structure, and record the Lagrangian cost associated with each WP tree. We then write the above-collected optimal Lagrangian costs in a binary tree, and run the single-tree algorithm one more time to find the best dyadic time-segmentation tree.

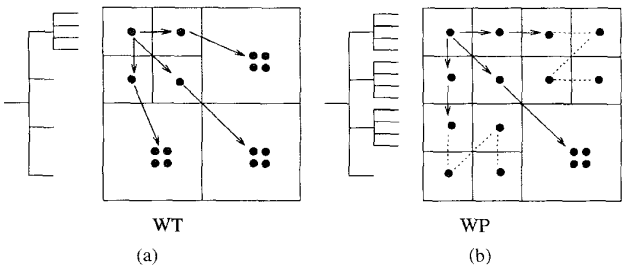



Fig. 4. Examples of spatial coefficient tree for different WP decompositions. (a) Wavelet case. (b) Arbitrary WP case. Arrows identify the parent-children dependencies.

It is easy to extend the single tree from one-dimensional (1-D) signals to 2-D images. It can be shown that the number of 2-D bases $S(d)$ searched by a single tree of depth-$d$ is given the recursion $S(d) = [S(d-1)]^4 + 1$, with $S(1) = 2$. For example, a depth-5 2-D WP decomposition has a library of $5.60 \times 10^{78}$ bases! We implement the single-tree algorithm for 2-D images using the self-referential quadtree data structures, and send the best pruned subtree as side information to assist decoding.

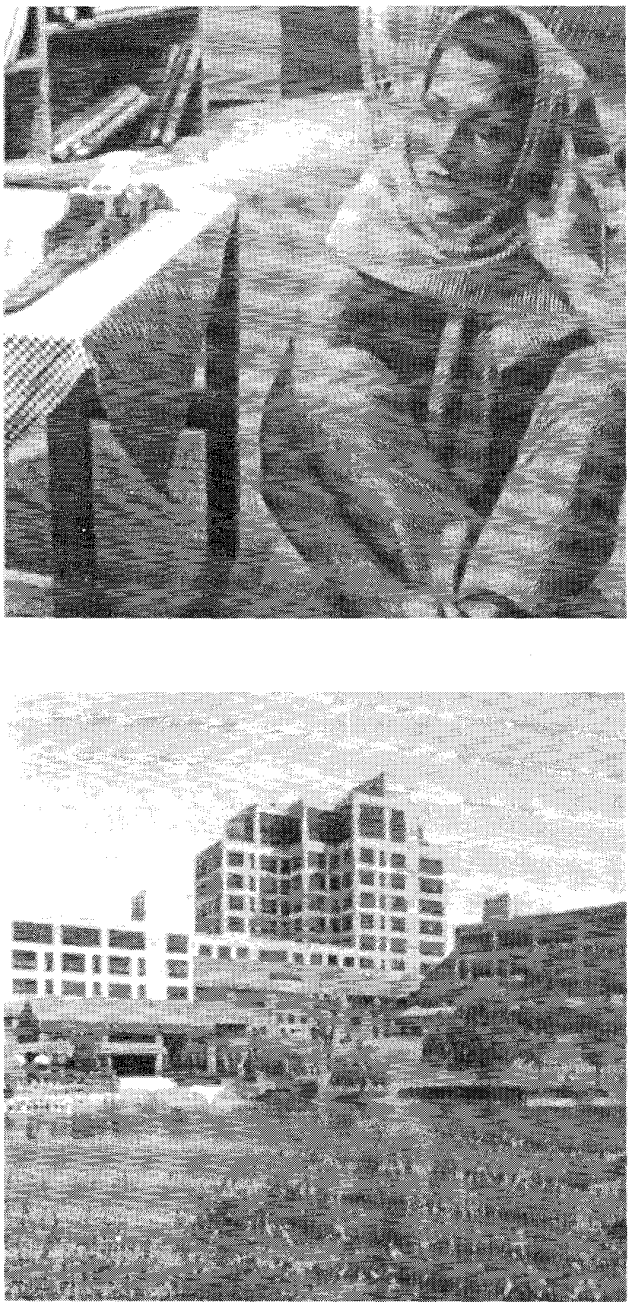
### B. Double-Tree Algorithm

The single tree effectively finds the best stationary frequency decomposition (from the WP tiling library) for the *unsegmented* signal taken as a whole. This is obviously limiting when the signal exhibits changing time-frequency characteristics. The double-tree algorithm [1] attempts to correct this shortcoming by addressing the joint search for (binary) time segmentations of the signal, along with the best WP frequency decompositions for each segment. The double tree thus represents a hierarchical extension of the single tree to accommodate binary time splits, so as to better address time-varying signal characteristics using time-varying filter banks.

The basic idea of the algorithm is simple, and it is based on the single-tree algorithm described earlier. We explain the double-tree structure through a 1-D example in Fig. 3, while the extension of the double-tree structure from 1-D signals to 2-D images is trivial. Suppose the signal consists of four quarters labeled as A, B, C, and D, respectively, in Fig. 3. We can grow a single tree on the whole signal (ABCD), or we can first segment the signal into two halves (AB and CD), and then grow individual single trees for each half, or we can further segment the halves into quarters (A, B, C, and D) before growing single trees for the quarters, and so on. In the end, we have a redundant double-tree structure for representing the original signal, allowing both tree-structured time segmentations and frequency decompositions.

To find the best time-segmentation and the best WP tree for each segment jointly, we first run the single-tree algorithm for each possible dyadic time segment (the whole signal, its halfs, quarters, and so on) to search for the best WP tree structure, and record the Lagrangian cost associated with each WP tree. We then write the above-collected optimal Lagrangian costs in a binary tree, and run the single-tree algorithm one more time to find the best dyadic time-segmentation tree.

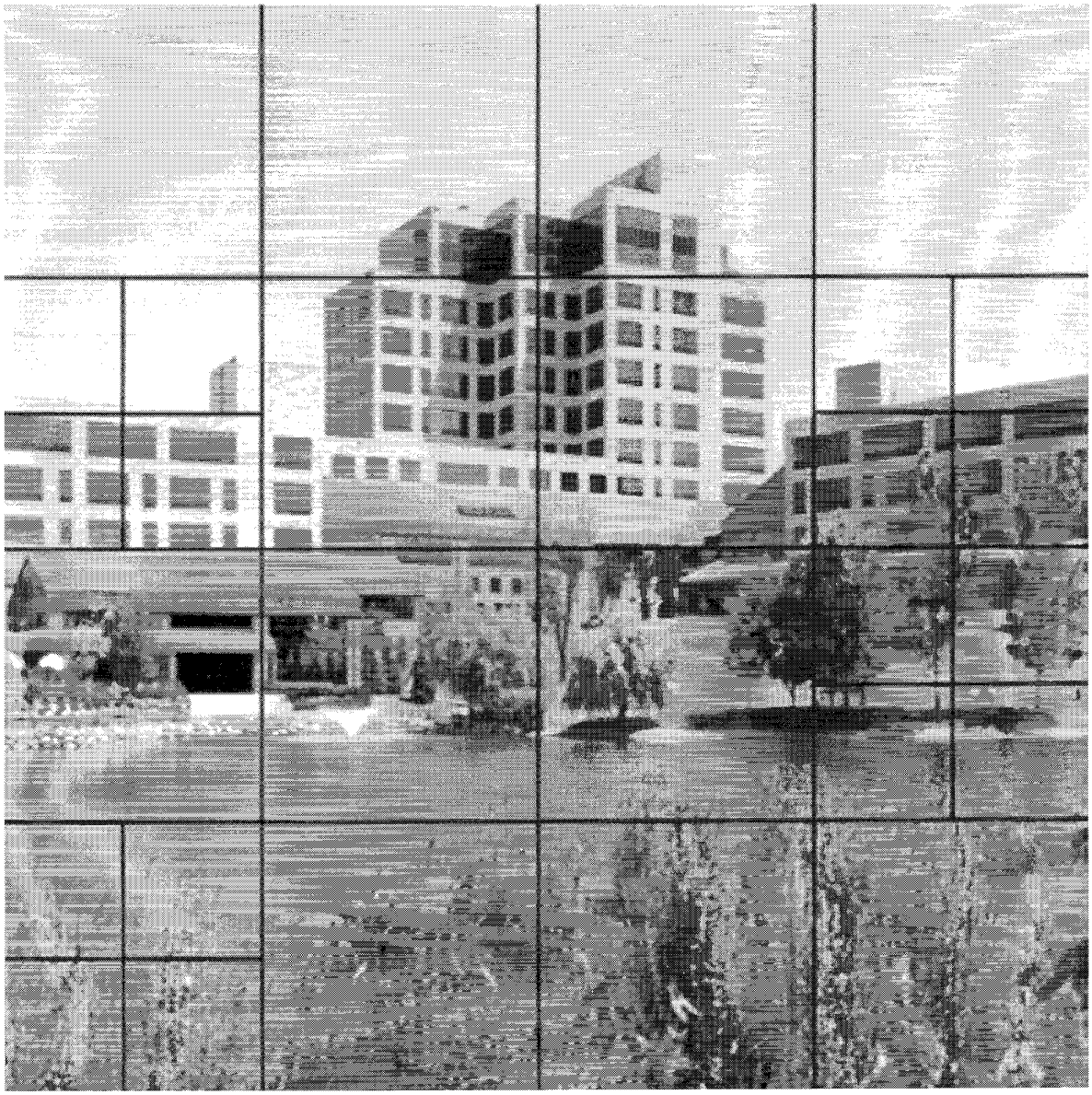
The name here is derived from the two kinds of trees that are pruned, frequency trees (corresponding to the solid-line trees of Fig. 3) associated with each dyadic segment of the original signal, and temporal tree (corresponding to the dashed-line tree of Fig. 3) associated with the time segmentations of the signal. The computational complexity of the double-tree algorithm can be shown to be of $O(N(\log N)^2)$ for a size $N$ signal.



(a)

Fig. 5. Coding results for the standard 512 × 512 Barbara and House images. (a) Original Barbara and House images. (Fig. 5 continued on next page.)

It can be shown that the number of 2-D bases $D(d)$ searched by a double tree of depth-$d$ is given by the recursion $D(d) = [D(d-1)]^4 + S(d) - S(d-1) + 1$, with $D(1) = 2$, where $S(d)$ is the number of bases searched by a single tree of depth-$d$. For example, a depth-5 2-D double-tree decomposition has a library of $6.5 \times 10^{96}$ bases. It is also obvious from Fig. 3 that the double tree is a generalization of the single tree (our experiment shows that the best double-tree decomposition of the Lena image indeed degenerates to



(b)

Fig. 5. (*Continued.*) (b) Double-tree segmentation and tiling for House at 1.0 b/pixel: Dark lines represent spatial segments; light lines the frequency boundaries of the WP tree. Note that the upper left corners are the lowpass bands in each spatial segment. The maximum double-tree depth is 5, and 341 b are sent as side information to convey the tiling information. (Fig. 5(b) continued on next page).

a single tree, highlighting that the space-frequency representation of the Lena image is stationary and is well captured by just a WP basis).

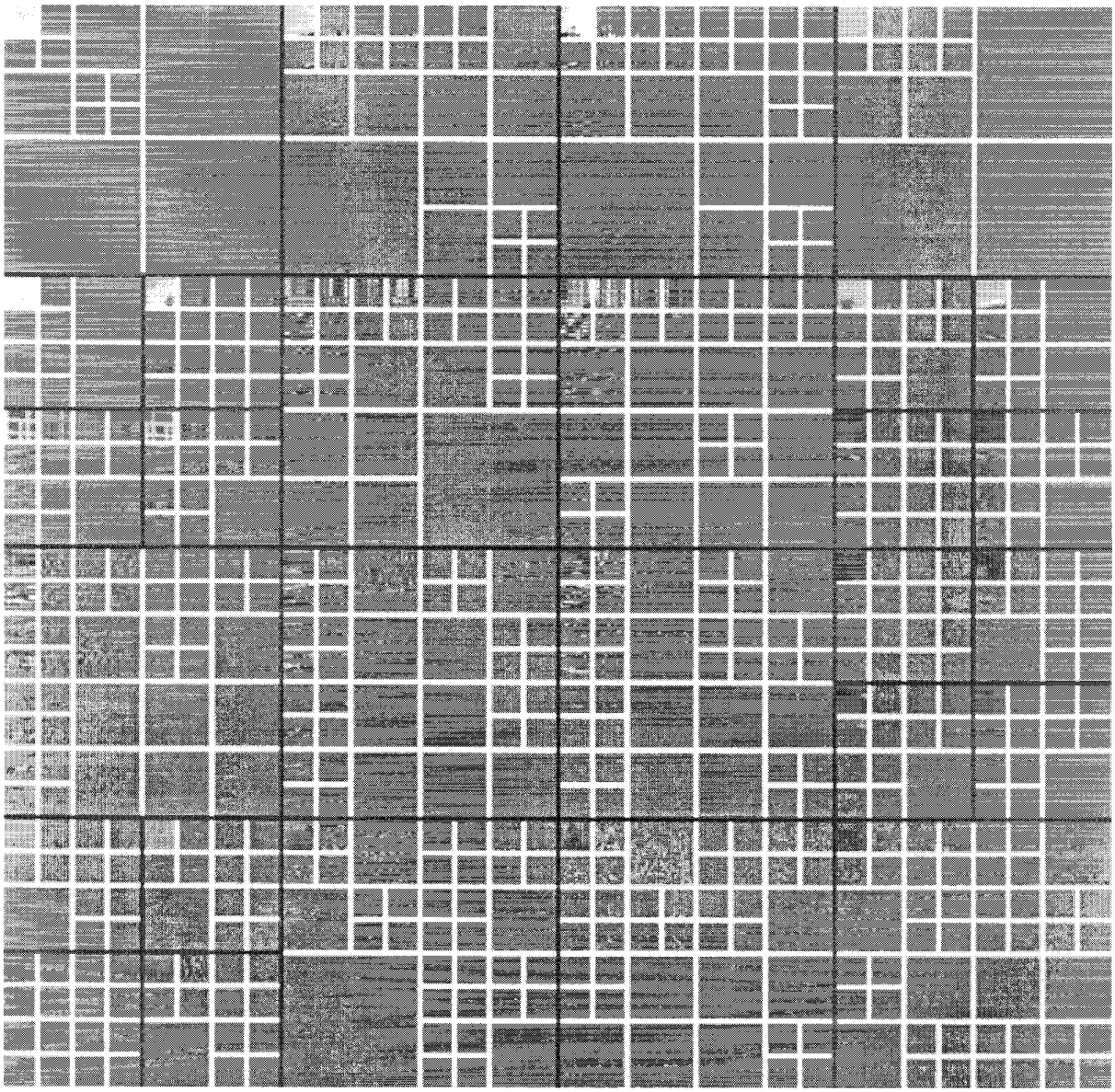
We now address the issue of side information for informing the decoder about the winning basis for the 2-D image case: for a double tree of maximum depth-$d$, we need $\sum_{k=1}^{k=d-1} 4^k$ b, with each bit specifying the spatial/frequency split decision at each node of the best double tree. In our experiments of Section III, where we have a depth-5 decomposition, for 512 $\times$ 512 images, this amounts to a total of 341 b, or 0.0013 b/pixel.

## III. Experiments

We have implemented the double-tree algorithm and applied it to the compression of images. The development of the algorithm assumed orthogonal sets of filters to ensure that quantization error (MSE) is transparent to the transform. In practice, many good linear phase biorthogonal filter sets, which are more desirable from a perceptual standpoint, are very close to orthogonal, so that the error introduced in the coefficient domain is very close to the final reconstruction error. We carried out extensive experiments, using both orthogonal and linear phase biorthogonal filters. For the results presented here, we use the 7–9 tap Daubechies biorthogonal filter set with symmetric extensions over the boundaries [9].

### A. Double-Tree Coding Using First-Order Entropy

We compare results using the same quantization strategies for all the expansions: the wavelet tree, the single tree, and the double tree. We use MSE and first-order entropy for distortion and rate, respectively, and use distortion plus $\lambda$ rate as our cost. The first-



(b)

Fig. 5. (*Continued.*) (b) (Fig. 5 continued on next page.)

order entropy rate measure can be achieved to a good approximation by using, for example, arithmetic coding of the quantization indexes. Table I gives the results when a single uniform scalar quantizer (picked optimally from a finite set of admissible quantizer choices) was used for all bands, with the picked quantization step size obviously depending on the target bit rate. As can be seen, the evolution in performance from the wavelet tree to the single tree to the double tree is strictly nondecreasing, as expected, with the relative gains being image- and rate-dependent. For the House image at 1 b/pixel, for example, the double tree achieves a gain of 1.7 dB and 1.1 dB in peak signal-to-noise ratio (PSNR) over the wavelet tree and the best single tree, respectively.

### *B. Double-Tree Coding Using Space-Frequency Quantization*

In addition to the scalar quantization and first-order entropy-based coder, we tested the double-tree algorithm on a "state-of-the-art" coder that uses a more powerful quantization strategy, and has coding performance ranking among the best in the literature. We provide a brief summary of its operation here, while referring the reader to [11] and [12] for details.

*Brief Summary of SFQ-Based Coding:* The state of the art in wavelet-based image coding has evolved from Shapiro's embedded zerotree wavelet coder [10], which applies the concept of zerotree quantization to exploit the space-frequency characterization of the wavelet transform. A powerful wavelet-based image coder employing space-frequency quantization (SFQ) was introduced in [11] with performance ranking among the best in the published literature. The SFQ coder exploits the space-frequency characterization of the wavelet representation. Like Shapiro's coder [10], it is built on the zerotree spatial data structure, i.e., it treats a wavelet image decomposition as a tree-structured representation, where a spatial coefficient tree is defined as the set of coefficients from all subbands

that correspond to the same spatial region of the image [see Fig. 4(a)]. Unlike Shapiro's coder, however, zerotree quantization in the SFQ coder is optimized in a rate-distortion sense. Two types of quantizers are applied to quantize the wavelet coefficients in SFQ: zerotree quantization of spatial coefficient trees and scalar quantization of frequency bands. Zerotree quantization results in a spatial subset of coefficients being "thrown out," or pruned, while scalar quantization addresses how to quantize the coefficients that survive the zerotree quantization operation. These two modes are coupled in the SFQ coder, which optimizes the tradeoff, in a rate-distortion framework, between the (tree-structured) subset of coefficients to throw away and the fidelity with which to represent the surviving coefficients. The optimally quantized wavelet coefficients are entropy-coded using adaptive arithmetic coding. See [11] for details.

While the SFQ coder of [11] was originally designed for a wavelet transform, it has been extended to an arbitrary WP transform in [12]. The basic idea is again to jointly match the spatial and frequency quantization modes of the coder to the space-frequency representation of the (arbitrary) WP transform. This is achieved by generalizing the definition of the spatial coefficient tree from WP to WP transforms, as shown in Fig. 4. This leads to considerable improvement in coding performance for the class of images whose space-frequency energy distribution is mismatched to the space-frequency tiling of the wavelet transform.

*SFQ-Based Coder for the Double-Tree Algorithm:* Since the double-tree algorithm produces a WP representation over separate spatial segments, the coder of [12] can be applied individually to each of the spatial segments found from the double-tree algorithm. This results in spatially adaptive space-frequency tilings that are matched to the image being coded. An ideal way to build a combined double tree and SFQ coder is to have a joint transform and quantizer design, where the SFQ quantizer is optimized for each possible candidate double-tree transformation. However, given the large number of double-tree transformations in the library ($6.5 \times 10^{96}$ for a depth-5 2-D decomposition), such an approach is not feasible. For this reason, we decouple the transform and quantizer design in our real coder and optimize double-tree transform and SFQ sequentially. We first search the library of possible transformations using the double-tree algorithm, using MSE and first-order entropy in the computation of costs. After the best basis has been found, we apply the SFQ scheme on that winning basis. If there is spatial segmentation in the best double-tree structure, SFQ coders are designed individually for each image segment, with the bit rate of each segment determined by the double-tree algorithm. Coding results for the standard 512 × 512 Lena, Barbara, and House images are tabulated in Table II. Note that the coding bit rates listed in Table II reflect the actual output bitstream of the arithmetic coder, which is part of the SFQ codec [11] (used as a "generic" high-performance wavelet codec to test our algorithm). The original Barbara and House images are shown in Fig. 5(a), the double-tree segmentation and tiling for House at 1 b/pixel in Fig. 5(b), and decoded Barbara and House images at 1 b/pixel in Fig. 5(c).

From the coding results in Table II, we see that the combined double-tree and SFQ coder gives better results than the single-tree-based SFQ for the Barbara image (0.4 to 0.6 dB at comparable bit rates). For the House image, the real coding gain of using the double-tree decomposition is about 0.2 dB over the single tree at moderate bit rates; at 0.25 b/pixel both the best double-tree and single-tree splits become a wavelet tree. For the Lena image, the wavelet basis turns out to be near optimal, and the best double-tree split degenerates to a single tree. In another words, there is no spatial segmentation in the best double-tree split of the Lena image. An explanation for the relatively smaller improvement from the single tree to the double tree

(c)

Fig. 5. (*Continued.*) (c) Decoded Barbara and House images, bit rate =1.0 b/pixel. PSNR = 37.58 dB for Barbara; PSNR = 38.01 dB for House.

when using the SFQ coder versus using the simpler entropy-based coder of Section III-A is that some of the spatial adaptivity of the double-tree algorithm is already being exploited by the sophisticated spatial quantization mode of the SFQ coder (note that the SFQ coder achieves higher PSNR's at comparable bit rates than does the entropy-based coder).

## IV. Discussion and Conclusion

A new adaptive image representation framework using spatially varying WP's is introduced in this paper via a double-tree structure

for image coding. This algorithm jointly searches for the best spatial segmentation and the best frequency decomposition to use for each segment. The main advantage of these adaptive representations is their versatility: They can adapt to a wide variety of image classes having varying space-frequency characteristics by searching efficiently through a very large library of tree-structured bases. Numerically, on a SPARC 5, calculating a 4-level wavelet transform of a $512 \times 512$ image took 1.08 s, while calculating the best single-tree basis (from among $4.9 \times 10^{19}$ bases) took 5.65 s, and calculating the best double-tree basis (from among $5.6 \times 10^{78}$ bases) took 21.18 s.

## REFERENCES


[1] C. Herley, J. Kovacevic, K. Ramchandran, and M. Vetterli, "Tilings of time-frequency plane: Construction of arbitrary orthogonal bases and fast tiling algorithms," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3341–3360, Dec. 1993.

[2] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[3] H. S. Malvar, *Signal Processing With Lapped Transforms*. Norwood, MA: Artech, 1992.

[4] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Commun. Pure Appl. Math.*, vol. XLI, pp. 909–996, 1988.

[5] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet decomposition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 674–693, 1989.

[6] R. Coifman and V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inform. Theory*, vol. 38, pp. 713–718, Mar. 1992.

[7] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Processing*, vol. 2, no. 2, pp. 160–176, Apr. 1993.

[8] K. Asai, K. Ramchandran, and M. Vetterli, "Image representation using time-varying wavelet packets, spatial segmentation and quantization," in *Proc. CISS*. Baltimore, MD: Johns Hopkins Univ., Mar. 1993.

[9] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing*, vol. 1, no. 2, pp. 205–221, Apr. 1992.

[10] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3445–3463, Dec. 1993.

[11] Z. Xiong, K. Ramchandran, and M. Orchard, "Joint optimization of scalar and tree-structured quantization of wavelet image decomposition," in *Proc. Asilomar Conf.*, Pacific Grove, CA, Nov. 1993, vol. 2, pp. 891–895.

[12] Z. Xiong, K. Ramchandran, M. T. Orchard, and K. Asai, "Wavelet packets-based image coding using joint space-frequency quantization," in *Proc. ICIP'94*, Austin, TX, Nov. 1994, vol. III, pp. 324–328.


# Cache Write Generate for Parallel Image Processing on Shared Memory Architectures

Craig M. Wittenbrink, Arun K. Somani, and Chung-Ho Chen

***Abstract*—We investigate *cache write generate*, our cache mode invention. We demonstrate that for parallel image processing applications, the new mode improves main memory bandwidth, CPU efficiency, cache hits, and cache latency. We use register level simulations validated by the UW-Proteus system. Many memory, cache, and processor configurations are evaluated.**

## I. INTRODUCTION

Cache memories play an important role in achieving higher performance in modern uni- and multiprocessors. When a high percentage of reads and writes are made to the cache, the effective bandwidth of the memory is that of the cache. Many prior studies have focused on read caching [7]. Here, we focus on write caching. Write buffers [7], write allocate [4], and write through [1], [5] do not address the removal of unnecessary traffic. To prevent unnecessary reads, many systems provide software control of cache write updating [6]. Word validate has been used by [1], [4], and [5], and write allocate has been used by [4].

C. M. Wittenbrink, in [9], investigated the effect of directly updating the line when it was known in advance that the line is to be written by using trace analysis. In this paper, we further investigate the cache write technique *cache write generate*. Cache write generate directly updates the cache on write misses, without reading from memory. We show that for a class of applications, the overall performance improvement is significant. We performed the analysis using hardware description language (HDL) simulations and performance measurements of each cache write technique.

Cache write generate (CWG) is defined as cache write validation on a write miss. The cache line is updated with the write and the cache line tag is modified to the address of the write. Writes that benefit from CWG are computed or initialized by the processor. Examples include dynamically allocated memories, stack segments, static memory segments, and temporary buffers. In image processing and vision applications [3], [8], these memory areas are easy to identify through explicit declaration or by the compiler. CWG is done only on memory areas denoted as generate, and a cache line in a generate memory area may lose its CWG ability to insure memory consistency. We have developed several schemes to provide self consistency, but do not discuss them due to space constraints. See our paper [10] for details.

## II. SIMULATION MODELS AND HARDWARE SYSTEM

### A. Cache Modes, Sizes, and Memory Timings

We compare the relative efficiency of the cache write generate policy to existing write caching controls, using single and multipro-

Manuscript received February 18, 1995; revised September 27, 1995. This work was supported by the Navy Coastal Systems Center. The associate editor coordinating the review of this correspondence and approving it for publication was Prof. A. M. Tekalp.

The authors are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA.

C. M. Wittenbrink is currently at the Baskin Center for Computer Engineering, University of California, Santa Cruz, CA 95064 USA (e-mail: craig@cse.ucsc.edu).

Publisher Item Identifier S 1057-7149(96)03172-7.