Anisotropic 2-D Wavelet Packets and Rectangular Tiling: Theory and Algorithms

Dan Xu* and Minh N. Do*

*Department of Electrical and Computer Engineering and Beckman Institute
University of Illinois at Urbana-Champaign

Email: {danxu, minhdo}@uiuc.edu; Web: www.ifp.uiuc.edu/~danxu

ABSTRACT

We propose a new subspace decomposition scheme called *anisotropic wavelet packets* which broadens the existing definition of 2-D wavelet packets. By allowing arbitrary order of row and column decompositions, this scheme fully considers the adaptivity, which helps find the best bases to represent an image. We also show that the number of candidate tree structures in the anisotropic case is much larger than isotropic case. The *greedy algorithm* and *double-tree algorithm* are then presented and experimental results are shown.

Keywords: Anisotropic wavelet packets, rectangular tiling, best bases, dynamic programming.

1. INTRODUCTION

The ability to represent signals in a large family of bases, where a “best” one can be adaptively selected for a given signal, is a powerful concept. Wavelet packets¹ (WP) provide one of such scheme. Wavelet packets can be seen as an extension of the wavelet transform via general tree-structured filter banks.² In the one-dimensional wavelet transform, the building block two-channel filter bank is iteratively applied to the lowpass channels. The 1-D wavelet packet scheme extends this by allowing the two-channel filter bank to be applied to further decompose any channel. This results in a large number of bases corresponding to different binary decomposition trees.

This idea can be easily extended to higher dimensions using separable filter banks. We start with the building block of the 2-D wavelet transform that consists of a 1-D two-channel filter bank applied to each row, followed by a 1-D two-channel filter bank applied to each column* (see Figure 1). In the 2-D wavelet transform, this equivalent four-channel filter bank is iteratively applied to the lowpass channels, whereas 2-D wavelet packets allow it to be applied to any channel. The reason for the coupling of the row and column transform steps is so that the resulting basis functions have the same scale along each dimension.³ We thus refer to such commonly used wavelet packets as *isotropic* wavelet packets.

Recently, several studies in constructing sparse expansions for natural images with smooth contours^{4,5} have revealed that anisotropy is a desired feature for new image expansions. More precisely, to capture smooth contours in images, the expansion should contain basis functions with variety of shapes, in particular with different aspect ratios. This prompts us to consider relaxing the coupling of the row and column decomposition step in the isotropic wavelet packet. The result is a new family of bases where the row and column decomposition steps can be taken at arbitrary order at each node. We will refer to this family of bases as *anisotropic* wavelet packets, which obviously contain isotropic wavelet packets as a subset. It can be seen that each resulting anisotropic wavelet packet basis corresponds to a binary tree where each node is assigned to either a “row” or a “column” value.

In the frequency-domain, the isotropic wavelet packet scheme can be seen as a quad-tree partitioning of the frequency spectrum, where each further decomposition step on a subband would divide its spectrum into four square regions. Thus the isotropic wavelet packets have a dual family in the space-domain as quad-tree partitioning into square regions. For the anisotropic wavelet packet family, we see that its dual in the space-domain is a partitioning scheme into rectangular regions, where each of the regions can be divided further into two halves, along either row or column. We will refer to such a scheme as *rectangular tiling*.

*The order of row and column steps can be reversed, which would give the identical result

While anisotropic WP and rectangular tiling seem somewhat trivial extensions, to our best knowledge, it has not been considered before. A key question, which will be addressed in this paper, is whether these extensions provide significant gains without significantly increasing the computational complexity. After briefly reviewing the main result of isotropic WP in Section 2, we will formally introduce and study the properties of the anisotropic wavelet packets in Section 3. A main result is the calculation of the number of anisotropic WP bases, which is significantly larger than the number of isotropic WP bases. In Section 4, we consider a restricted class of anisotropic wavelet packets and rectangular tiling where all decomposition in one dimension are taken before another dimension. Such a restriction might be appropriated in certain applications. Experiments with real images in Section 5 show the potential gains of anisotropic WPs in providing better bases than isotropic WPs.

2. BACKGROUND

2.1. Two-Dimensional Separable Filter Banks

Separable filter banks are the building blocks of the 2-D wavelet transform. Figure 1 shows a typical 2-D separable filter banks. We denote V_j^2 as the j th level 2-D approximation space, V_j^r and V_j^c as the j th level 1-D row and

Figure 1: 2-D separable filter banks. The H_0 and H_1 variables represent low-pass and high-pass digital filters, respectively. The filtering is divided into two steps, first along the rows and then along the columns.

column approximation subspaces, respectively. Based on the assumption

$$V_j^2 = V_j^r \otimes V_j^c \quad (1)$$

the orthonormal bases can also be separated as

$$\phi_{j,(n_1,n_2)}^2(x_1, x_2) = \phi_{j,n_1}(x_1)\phi_{j,n_2}(x_2) = \frac{1}{2^j} \phi\left(\frac{x_1 - 2^j n_1}{2^j}, \frac{x_2 - 2^j n_2}{2^j}\right) \quad (2)$$

Here we denote $\phi_{j,(.,.)}^2(\cdot, \cdot)$ as the j th level 2-D scaling function, $\phi_{j,.}$ as the j th level 1-D scaling function, x_1 and x_2 as variables in the spatial domain for row and column subspaces respectively, and n_1 and n_2 as shifts for the row and column subspaces respectively. Another result of this assumption is the decomposition of 2-D approximation space of up to $J - 1$ levels

$$\begin{aligned} V_{j-1}^2 &= V_{j-1}^r \otimes V_{j-1}^c = (V_j^r \oplus W_j^r) \otimes (V_j^c \oplus W_j^c) \\ &= \underbrace{(V_j^r \otimes V_j^c)}_{V_j^2} \oplus \underbrace{(W_j^r \otimes V_j^c)}_{W_j^{2,(1)}} \oplus \underbrace{(V_j^r \otimes W_j^c)}_{W_j^{2,(2)}} \oplus \underbrace{(W_j^r \otimes W_j^c)}_{W_j^{2,(3)}} \end{aligned} \quad (3)$$

Here we denote W_j^r and W_j^c as the j th level 1-D row and column detail subspaces, respectively, $W_j^{2,(1)}$, $W_j^{2,(2)}$ and $W_j^{2,(3)}$ as the j th level 2-D detail subspaces of horizontal, vertical and diagonal directions, respectively. Equation 3 gives us the expression of orthonormal bases for V_{j-1}^2 , which are

$$\phi_{j,n_1}(x_1)\phi_{j,n_2}(x_2), \phi_{j,n_1}(x_1)\psi_{j,n_2}(x_2), \psi_{j,n_1}(x_1)\phi_{j,n_2}(x_2), \psi_{j,n_1}(x_1)\psi_{j,n_2}(x_2) \quad (4)$$

2.2. Two-Dimensional Isotropic Wavelet Packets

The notion of wavelet packets gives freedom to the choice of decomposition from both approximation and detail subbands, instead of allowing further decomposition only from the approximation subband. We will refer to these wavelet packets as *isotropic wavelet packets* to distinguish them from another class that we will describe in a later section. The basic idea is the space decomposition of $W_j^{p,q}$, the j th level detail subspace, with indices p and q , $0 \leq p \leq 2^i$, $0 \leq q \leq 2^j$. That is,

$$W_j^{p,q} = W_{j+1}^{2p,2q} \oplus W_{j+1}^{2p+1,2q} \oplus W_{j+1}^{2p,2q+1} \oplus W_{j+1}^{2p+1,2q+1} \quad (5)$$

Isotropic wavelet packet uses quadtree as its tree structure. A typical tree structure for 2-D wavelet packet is shown in Figure 2.

Figure 2: Space decomposition in isotropic wavelet packet tree. After one decomposition, the space $W_j^{p,q}$ generates four subspaces, in which level j becomes $j+1$, and indexes $\{p, q\}$ become $\{2p, 2q\}$, $\{2p+1, 2q\}$, $\{2p, 2q+1\}$, $\{2p+1, 2q+1\}$.

The number of different wavelet packet bases of V_j^2 is equal to the number of admissible quadrees. An admissible quadtree is a quadtree with all its nodes having 0 or four children. We denote the number of bases in a 2-D isotropic WP tree of up to level J . Then it is easy to see that $Q_0 = 1$ and $Q_J = Q_{J-1}^4 + 1$. Based on this recursive formula, the following proposition provides an estimate of Q_J .

PROPOSITION 1 (³ PROP. 8.5). *The number Q_J of wavelet packet bases in a 2-D isotropic wavelet packet tree of up to level J satisfies*

$$2^{4^{J-1}} \leq Q_J \leq 2^{\frac{49}{48}} 4^{J-1} \quad (6)$$

The algorithm for finding the best basis of 2-D isotropic wavelet packets is based on the dynamic programming.^{6,7} In a 2-D isotropic wavelet packet tree, each node has an orthonormal basis $\mathcal{B}_j^{p,q}$ of $W_j^{p,q}$. We define the cost of f with respect to basis $\mathcal{B} = \{g_m\}_{0 \leq m < M}$ as the partial sum

$$C(f, \mathcal{B}) = \sum_{m=0}^{M-1} \Phi\left(\frac{|\langle f, g_m \rangle|^2}{\|f\|^2}\right) \quad (7)$$

Here $\Phi(\cdot)$ is any concave function. We call the cost the *additive cost* if for any orthonormal bases $\mathcal{B}^0$ and $\mathcal{B}^1$

$$C(f, \mathcal{B}^0 \cup \mathcal{B}^1) = C(f, \mathcal{B}^0) + C(f, \mathcal{B}^1) \quad (8)$$

PROPOSITION 2 (³ PROP. 9.10). *Suppose that C is an additive cost function. If*

$$C(f, \mathcal{B}_j^{p,q}) < C(f, \mathcal{O}_{j+1}^{2p,2q}) + C(f, \mathcal{O}_{j+1}^{2p+1,2q}) + C(f, \mathcal{O}_{j+1}^{2p,2q+1}) + C(f, \mathcal{O}_{j+1}^{2p+1,2q+1})$$

then

$$\mathcal{O}_j^{p,q} = \mathcal{B}_j^{p,q} \quad (9)$$

otherwise

$$\mathcal{O}_j^{p,q} = \mathcal{O}_{j+1}^{2p,2q} + \mathcal{O}_{j+1}^{2p+1,2q} + \mathcal{O}_{j+1}^{2p,2q+1} + \mathcal{O}_{j+1}^{2p+1,2q+1} \quad (10)$$

Here f is the input 2-D signal. $\mathcal{O}_j^{p,q}$ is the optimal basis of $W_j^{p,q}$.

We can see Proposition 2 guarantees a recursive dynamic programming algorithm. By iterating the above process from bottom of the tree to the top, the overall optimal basis will be obtained. The computational complexity of this algorithm is $O(J^2 \log_2 J)$ for the wavelet packet trees of up to level J .⁶

Although 2-D isotropic wavelet packets allow freedom in the selection of a subband for further decomposition, it is somewhat rigid in the 2-consecutive-step row-column decomposition. In a real image, the dominant directionality and texture for each subband drives us to think of a more flexible way of decomposition, which can take suitable transforms according to the actual feature of one given subband. We try to find a more effective way to represent image in wavelet domain. This usually requires large set of candidate decompositions and fast algorithms to find the best tree. The solution is the so called 2-D anisotropic wavelet packets, which take row-wise and column-wise decomposition separately and adaptively.

3. ANISOTROPIC WAVELET PACKETS AND RECTANGULAR TILING

The isotropic wavelet packet is a quadtree like structure which performs row and column decompositions consecutively in one step. Since row and column decompositions respectively express the vertical and horizontal discontinuity, this structure seems cumbersome when dealing with some images where a prevalent directionality exists in certain subbands. By enabling an arbitrary order of row and column decomposition, we propose the 2-D anisotropic wavelet packets here.

3.1. Anisotropic Wavelet Packet Decomposition

In anisotropic wavelet packets, each row decomposition is not necessarily followed by a column decomposition. Rather, for each step, either a row or a column decomposition will be chosen. In this way, it is possible to have the case of several consecutive column decompositions or any combination of row and column decompositions.

Denote the anisotropic wavelet packet spaces as $W_{i,j}^{p,q}$ (see Figure 3). Here i and j are row level and column level respectively. Variables p and q are row and column indexes respectively to specify a certain space. The anisotropic wavelet packet spaces are made up of row spaces W_i^p and column spaces W_j^q .

$$W_{i,j}^{p,q} = W_i^p \otimes W_j^q \quad (11)$$

Note that for an image with size $2^M \times 2^N$, i and j satisfy

$$\begin{aligned} 0 &\leq i \leq M \\ 0 &\leq j \leq N \end{aligned} \quad (12)$$

For a row decomposition, the anisotropic wavelet packet spaces satisfy

$$W_{i,j}^{p,q} = W_{i+1,j}^{2p,q} \oplus W_{i+1,j}^{2p+1,q} \quad (13)$$

For a column decomposition

$$W_{i,j}^{p,q} = W_{i,j+1}^{p,2q} \oplus W_{i,j+1}^{p,2q+1} \quad (14)$$

Two-dimensional row (column) decomposition divides only the row (column) space in the tensor product expression of 2-D space. And for several steps of space decomposition, we are able to have any combination of row and column decompositions. This actually increases the flexibility of space decomposition, compared to the 2-D isotropic wavelet packets, where a row decomposition is always followed by a column decomposition. Figure 4 shows 8 varieties of decompositions inside a square. Note that only type (1) and type (8) are possible in isotropic wavelet packets. We make a selection of row or column decomposition in the first step and then make another selection in the second step. Both row and column decomposition in the first step can lead to a 2-D isotropic wavelet packet decomposition in the second step. Therefore the isotropic wavelet packets are just special cases of anisotropic wavelet packets. Note that a two-step decomposition has more than three varieties which are listed in the figure.

Figure 3: Index for $W_{i,j}^{p,q}$ in the frequency domain. Both frequency dimensions are normalized to the range 0 to 1. The i and j variables represent row level and column level, respectively. Axis f_1 is divided uniformly into 2^i segments, with each segment 2^{-i} in width and f_2 into 2^j segments, with each segment 2^{-j} in width. The p and q variables are indices for row and column and are such that $0 \leq p \leq 2^i - 1$, $0 \leq q \leq 2^j - 1$. To find the position of $W_{i,j}^{p,q}$, first we divide f_1 and f_2 axes into 2^i and 2^j uniform segments. Then $W_{i,j}^{p,q}$ is the intersecting area of the p th segment of f_1 axis and the q th segment of f_2 axis, which is shown as the shaded area, standing for the tensor product of row and column spaces. For a further decomposition, a column decomposition of $W_{i,j}^{p,q}$ for example, level j becomes level $j+1$ and each segment is divided in half, such that the width of each segment of f_2 axis becomes $2^{-(j+1)}$. The resulting lower part of the shaded area is indexed by $2q$, and the upper part indexed by $2q+1$.

Each step of decomposition corresponds to a 2-D row(column) wavelet transform which takes 1-D discrete wavelet transform along rows(columns). Given an input matrix with size $M \times N$, the row transform yields two coefficient matrices with size $M \times N/2$ each, one for approximation subspace and one for detail subspace, while the column transform yields another two coefficient matrices with size $M/2 \times N$ each, also one for approximation subspace and one for detail subspace. The 2-D row(column) decomposition is the counterpart of 2-D row(column) wavelet transform in the space decomposition sense. Here we show the results for one-step row-wise and column-wise decomposition in Figure 5. Note that in the detail subband for both row and column transform, the coefficients are nonzero only on two parallel lines. In this way, the original image is well compressed.

The structure for anisotropic wavelet packets is a binary-tree like structure, instead of the quadtree structure in isotropic case. Each node is assigned a value from the value set $\{row, col, null\}$, standing for a row decomposition, a column decomposition or no decomposition at all in a certain tree branch. Note that the level of the tree cannot exceed $2L$ for an image with size $2^L \times 2^L$. (Moreover, the level of row or column decomposition cannot exceed L either.) Figure 6 is a typical tree structure for anisotropic wavelet packets.

3.2. Rectangular Tiling

The separate and arbitrary decomposition of subspaces in row and column is equivalent to a rectangular tiling in the frequency domain. Instead of dividing each block into four equally sized sub-blocks, rectangular tiling brings about freedom to both number and direction of sub-blocks obtained in each one-step decomposition. There are two advantages of this. First, in the sense of the selection set size, the possible ways of decomposition are increased largely. Thus more near-to-optimal decomposition can be obtained. Second, the adaptive selection of row or column decomposition comes more naturally, similar to the scheme human uses when looking at a picture.

Figure 4: Eight possible decompositions of a unit square on frequency domain for anisotropic WP. Left side represents the tree structure of the decomposition. Right side represents the corresponding tiling in frequency domain.

Figure 5: 2-D Row and column transform. Black area represents zero coefficients, while white represents non-zeros. (a)Original image. (b)Row transform. The left part is the approximation subband. The right part is the detail subband, in which only one line of coefficients are non-zeros (shown as the gray vertical line). (c)Column transform. The upper part is the approximation subband. The lower part is the detail subband, in which all the coefficients are zeros.

A comparison of decomposition structure between quadtree-based isotropic wavelet packets and rectangular tiling-based anisotropic case is shown in Figure 7.

3.3. Number of Bases

We will show the large number of possible anisotropic wavelet packet tree structures here. Note that the more candidate tree structures we have, the better bases can be chosen from the basis family.

PROPOSITION 3. *The number $A_{I,J}$ of wavelet packet bases in an anisotropic wavelet packet tree of row level I and column level J satisfies*

$$A_{I,J} = 1 + A_{I-1,J}^2 + A_{I,J-1}^2 - A_{I-1,J-1}^4 \quad (15)$$

for $I \geq 1$ and $J \geq 1$, and $A_{I,0} = 1 + A_{I-1,0}^2$, $A_{0,J} = 1 + A_{0,J-1}^2$ and $A_{0,0} = 1$.

Proof: We will prove this result by induction. Denote $A_{I,J}$ as the number of different anisotropic tree expansion with row level I and column level J . In the first level tree branch, that is the first decomposition a tree can possibly have. It can only be a row decomposition, or a column decomposition or no decomposition at all. If the first level is a row decomposition, the whole tree is made up of two subtrees that are set of trees of up to $I - 1$ row levels and J column levels. similarly, if the first level is a column decomposition, the whole tree is made up of two subtrees that are set of trees of up to I row levels and $J - 1$ column levels. In the third case, the tree has level 0 that is reduced to the root. Note that the first two cases have an overlap set. In frequency domain, this

Figure 6: Binary-tree like anisotropic wavelet decomposition

Figure 7: Comparison between different decomposition structures

set is all the tiling with an isotropic decomposition in the first level. The number of bases in this set is $A_{I-1,J-1}^4$. Therefore, we have the recursive formula

$$A_{I,J} = 1 + A_{I-1,J}^2 + A_{I,J-1}^2 - A_{I-1,J-1}^4 \quad (16)$$

Note that if I and/or J are 0, the corresponding row and/or column decomposition cannot be done. Therefore we have $A_{I,0} = 1 + A_{I-1,0}^2$, $A_{0,J} = 1 + A_{0,J-1}^2$ and $A_{0,0} = 1$. Because of symmetry between row and column decomposition, $A_{I,J}$ equals to $A_{J,I}$.

A comparison among $A_{J,J}$, Q_J (see Equation (6)), and S_J (see Equation (18)) can be found in Section 4.

3.4. Adaptive Basis Selection using Greedy Algorithm

Because of allowing arbitrary combination of row and column decompositions, the best basis selection algorithm in isotropic WP does not work here in anisotropic WP. We make the adaptive basis selection by greedy algorithm, which is a simple but very fast algorithm to search for the suboptimal tree structure. It starts from the original image and ends at some local optimal step. Although this ignores a possible large set of candidates, the result shows that it works very well with respect to both compression rate and computational cost.

We first start from the root node, which is the original image. Then we compare the cost for no decomposition C_{null} , row decomposition C_{row} , and column decomposition C_{col} to determine the lowest cost C .

$$\begin{aligned} C_{null} &= C(f, \mathcal{B}_{i,j}^{p,q}) \\ C_{row} &= C(f, \mathcal{B}_{i+1,j}^{2p,q}) + C(f, \mathcal{B}_{i+1,j}^{2p+1,q}) + \Delta C_{tree,row} \\ C_{col} &= C(f, \mathcal{B}_{i,j+1}^{p,2q}) + C(f, \mathcal{B}_{i,j+1}^{p,2q+1}) + \Delta C_{tree,col} \\ C &= \min\{C_{null}, C_{row}, C_{col}\} \end{aligned} \quad (17)$$

In equation 17, $C(f, \mathcal{B})$ is the cost of f with respect to basis family $\mathcal{B}$ and is required to be additive. Refer to equation 7 and 8 for more about this. The notations $\Delta C_{tree,row}$ and $\Delta C_{tree,col}$ are for the additional cost of row

and column decompositions, respectively. In determining C we decide whether decomposition is needed and in which direction to decompose. If further decomposition is needed, add the property “row” or “col” to the node. Otherwise, set the property as “null”. Repeat the above procedure for each node from top nodes to leaf nodes, until no more nodes can be decomposed.

The computational complexity varies for different images. In the worse case, the complexity will be $O(J^3)$ for a level of J . But normally, the computational cost is much less than this bound and thus usually yields a fast convergence of this algorithm.

For the optimal basis selection in anisotropic WP, we can do a recursive search from top to the bottom of the tree.⁸

4. DOUBLE TREE ALGORITHM

In this section, we consider some special cases, e.g. cases where all column transforms are taken after all row transforms, or all row transforms after all column transforms. This subset is denoted as $\mathcal{D}$. We use the double tree algorithm⁹ here, which is first used in time-frequency optimality problem. Note that the cost function here, the entropy, is additive, and we want to achieve simultaneous optimality in both row and column. If we consider the subset $\mathcal{D}$, we can get a suboptimal solution to our problem.

Given the maximum level J , we first build up a complete binary tree with only row decompositions. For each node in this all-row tree, we use the best tree search algorithm in isotropic case to find the best all-column tree associated with that node (Figure 8). Here the “best” means the minimum cost of a signal represented by

Figure 8: Double tree algorithm. Dotted lines are decompositions along row dimension. Solid lines are decompositions along column dimension.

a family of bases, with respect to a certain cost function, which can be an entropy function, for example. We record the least cost in column dimension for each node in the all-row tree. Then in the all-row tree, we find the best tree according to the least cost in column, using the best tree search algorithm similar to the one in isotropic case. Specifically, from bottom to top of the all-row tree, we access each node and do a pruning if necessary. The criterion for the pruning is based upon whether the cost of the parental node is more than the sum of costs of its child nodes. In the all-row tree, the costs are the least costs calculated in the all-column tree.

Note that the size of subset $\mathcal{D}$ is still much larger than the isotropic wavelet packets. We have the following proposition.

PROPOSITION 4. *The number S_J of elements in $\mathcal{D}$ for double trees with up to level J satisfies*

$$S_J = 2 \sum_{k=0}^{B_J-1} B_J^{M_J(k)} - B_J^2 \quad (18)$$

where B_J is the number of wavelet packet bases in a 1-D isotropic wavelet packet tree of up to level J and satisfies $B_J = B_{J-1}^2 + 1, B_0 = 1$ (see Mallat's book, Proposition 8.1). Here $M_J(k)$ is the number of leaf nodes for the k th

tree structure pattern in the set of all-row (or all-column) trees of up to J levels, and satisfies

$$\begin{cases} M_J(0) = 1 \\ M_J(k) = M_{J-1}(m) + M_{J-1}(n) \\ m = \lfloor \frac{k-1}{B_{J-1}} \rfloor \\ n = (k-1) \bmod B_{J-1}, \text{ for } k = 1, 2, \dots, B_J - 1. \end{cases} \quad (19)$$

where $\lfloor x \rfloor$ is the nearest integer less than or equal to x . $a \bmod b$ is the modulus by dividing a into b .

Proof: The subset $\mathcal{D}$ is a union of $\mathcal{D}_1$, the set of all-row-first decomposition and $\mathcal{D}_2$, the set of all-column-first decomposition. These two sets overlap only when all the row and column tiling separators in the transform domain hit two sides of the boundary, as an example shown in Figure 9 (c). For set $\mathcal{D}$, the tiling structure can only be three types which are shown in Figure 9. Type (a) is in set $\mathcal{D}_1$ and type (b) is in set $\mathcal{D}_2$. For (c), both $\mathcal{D}_1$ and $\mathcal{D}_2$ have this structure. This overlap has a total of B_J^2 structures.

Figure 9: Three types of tiling. (a) All the vertical separators hit both sides of the boundary (b) All the horizontal separators hit both sides of the boundary (c) All the horizontal and vertical separators hit both sides of the boundary

Because of symmetry, the number of elements in $\mathcal{D}_1$ equals to that of $\mathcal{D}_2$. Denote this number as T_J . Then

$$S_J = 2T_J - B_J^2 \quad (20)$$

Now we calculate T_J . For any tree in $\mathcal{D}_1$, we first choose an arbitrary all-row tree of up to depth J . For each leaf node in this all-row tree, we can further add an all-column tree rooted from the leaf node of the all-row tree (see Figure 10). Each of these all-column tree has B_J number of structures. Note that for each tree structure we get here, there is a unique tiling structure corresponding to it. Denote the number of leaf nodes for the k th pattern in an all-row tree of up to L levels as $M_J(k)$, $k = 0, 1, 2, \dots, B_J - 1$.

$$T_J = \sum_{k=0}^{B_J-1} B_J^{M_J(k)} \quad (21)$$

For $M_J(k)$, $k = 0$ corresponds to the pure root case and thus $M_J(0) = 1$. For $k \geq 1$, the leaf nodes are made up of leaf nodes from left subtree and right subtree (see Figure 10). Both of them are of up to $(J-1)$ levels. Therefore we have

$$M_J(k) = M_{J-1}(m) + M_{J-1}(n) \quad (22)$$

for $m, n = 0, 1, 2, \dots, B_{J-1} - 1$. We can put an index to k like this

$$k = mB_{J-1} + n + 1 \quad (23)$$

or

$$\begin{cases} m = \lfloor \frac{k-1}{B_{J-1}} \rfloor \\ n = (k-1) \bmod B_{J-1} \end{cases} \quad (24)$$

□

Now we make a comparison among Q_J , S_J and $A_{J,J}$ in Table 1. We can see that both $A_{J,J}$ and S_J are much greater than Q_J . This will establish a large set of candidates to achieve better decomposition of a 2-D signal. The number S_J is smaller than $A_{J,J}$, which is compatible with the definition of double tree.

Figure 10: An example of double tree. A shaded circle stands for a leaf node with an all-column tree rooted from it.

J	Q_J	S_J	$A_{J,J}$
1	2	8	8
2	17	1785	6857
3	8.35×10^4	4.86×10^{11}	2.32×10^{15}
4	4.87×10^{19}	3.94×10^{45}	2.89×10^{61}
5	5.61×10^{78}	2.88×10^{181}	7.00×10^{245}

Table 1: Comparison among the number of bases from isotropic WP (Q_J), double tree (S_J) and anisotropic WP ($A_{J,J}$) for up to decomposition level J .

5. EXPERIMENTAL RESULTS

We have implemented the greedy and double tree algorithms. The results are compared among best tree search algorithm for 2-D wavelets, 2-D isotropic wavelet packets, greedy algorithm and double tree algorithm.

Figure 11 is the comparison of overall entropy. We can see from this chart that both greedy and double tree algorithm achieve lower overall entropy than the existing algorithm. Therefore possibly higher compression rate can be obtained by using these algorithms.

For an 8×8 artificial image which is shown in Figure 12, the decomposition results are illustrated by greedy tree, double tree and isotropic decomposition. Both double-tree and greedy algorithms get the same result where three non-zero coefficients remain in coefficient matrix, while in the isotropic case, the number increases to eight. Figure 12 also shows different tiling structures of the best isotropic wavelet packet tree, greedy tree and double tree. If we denote the set of isotropic wavelet packets as $\mathcal{I}$, the set of anisotropic ones as $\mathcal{A}$, we have $\mathcal{I} \in \mathcal{A}$. Moreover, from the results we can see, the optimal structure is usually in the set $\mathcal{S} = \mathcal{A} - \mathcal{I}$. This proves that we should consider the anisotropic case to find the optimal decomposition structure.

We also show the results for a real image here. Figure 13 shows the best tiling structure for isotropic wavelet packet tree, greedy tree and double tree.

6. CONCLUSION

By introducing the arbitrary order of row and column decomposition, the anisotropic wavelet packets create a much larger set of wavelet packet bases and thus better decomposition structure can be obtained.

Experiments show that the two suboptimal algorithms, greedy and double tree algorithms outperform the existing isotropic wavelet packets with respect to over entropy, which can possibly bring about better compression. The greedy algorithm converges very fast and will be of much practical use. The double tree algorithm also runs at an acceptable time cost.

Figure 11: Comparison of overall entropy performance of 4 different wavelet decomposition approaches

Figure 12: Coefficients in transform domain for an 8×8 artificial image. Dark grey stands for coefficient 0. Black lines are tiling separators which show the tiling structure and also reflect the corresponding tree structure. (a)Original image (8×8 , entropy=3.1781) (b)Isotropic case (8 non-zero coefficients, entropy=1.5301) (c)Greedy tree (3 non-zero coefficients, entropy=0.7215) (d)Double tree (3 non-zero coefficients, entropy=0.7215)

REFERENCES

1. R. R. Coifman, Y. Meyer, and M. V. Wickerhauser, "Wavelet analysis and signal processing," in *Wavelets and Their Applications*, M. B. Ruskai, G. Beylkin, R. R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, eds., pp. 153–178, Jones and Bartlett, Boston, 1992.
2. M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice-Hall, 1995.
3. S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 2nd ed., 1999.
4. E. J. Candès and D. L. Donoho, "Curvelets – a suprisingly effective nonadaptive representation for objects with edges," in *Curve and Surface Fitting*, A. Cohen, C. Rabut, and L. L. Schumaker, eds., Vanderbilt University Press, (Saint-Malo), 1999.
5. M. N. Do and M. Vetterli, "Contourlets," in *Beyond Wavelets*, G. V. Welland, ed., Academic Press, New York, 2003. to appear, <http://www.ifp.uiuc.edu/~minhdo/publications>.
6. R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," **38**, pp. 713–718, March 1992.
7. K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," **2**, pp. 160–175, April 1993.
8. I. Pollak, M. N. Do, and C. A. Bouman, "Optimal tilings and best basis search in large dictionaries," in *Proc. of Thirty-Seventh Annual Asilomar Conference on Signals, Systems, and Computers*, (Pacific Grove, CA), 2003. to appear.

Figure 13: Best tiling in transform domain for Lena image. (a)Original image. Entropy=12.2468 (b)Isotropic WP tree. Entropy=6.9185 (c)Greedy tree. Entropy=6.9166 (d)Double tree. Entropy=6.9166

9. C. Herley, J. Kovačević, K. Ramchandran, and M. Vetterli, “Tilings of the time-frequency plane:construction of arbitrary orthogonal bases and fast tiling algorithms,” *IEEE Trans. Signal Proc.* **41**, pp. 3341–3359, Dec. 1993.