COMPACT IMAGE REPRESENTATION USING WAVELET LIFTING ALONG ARBITRARY TREES

Godwin Shen and Antonio Ortega

Signal and Image Processing Institute
University of Southern California
godwinsh@usc.edu, ortega@sipi.usc.edu

ABSTRACT

In this work, we present a method for achieving a compact image representation by applying wavelet lifting along an arbitrary combination of trees. This is done by extending a lifting transform along a single tree to a series of lifting transforms along a number of different trees, each of which exploits directionality differently. The trees are constructed in ways that force tree edges to follow the geometric flows inherent in images. In particular, we propose a method for constructing trees that trace out geometric flows using edges in the image. Our preliminary results demonstrate promising performance in terms of PSNR as a function of the percentage of retained coefficients, motivating its potential for image coding.

Index Terms— Wavelet transforms, Image representations, Image processing, Image analysis, Trees (graphs)

1. INTRODUCTION

Wavelet transforms are widely used for image processing. Simple *separable* extensions of orthonormal or bi-orthogonal 1D wavelet transforms exhibit good energy compaction properties; for smooth images the majority of the signal energy in the transform domain is concentrated in the lowest frequency bands. This leads to image coding techniques that can represent transform domain information efficiently at low bit rates. The separable nature of these standard wavelet-based methods limits their efficiency in representing directional information inherent in image data. Thus, transforms that capture directional information have the potential to lead to more efficient representations for image processing and coding. Numerous techniques have been proposed to capture directional information. Some of these (e.g., contourlets [1]) require some degree of oversampling, while others (e.g., bandelets [2] and directionlets [3]) are critically sampled. In this paper we focus on a novel critically sampled technique that can be viewed as an extension of [2, 3].

Bandelets [2] directly exploit directional information by partitioning the original image into blocks that contain nearly homogeneous geometric flows (i.e., pixel intensity tends to be aligned in a direction in a given block). Standard separable wavelet bases are then warped along these geometric flows and these warped bases are used to represent the images. This allows the transform to exploit directionality in a localized fasion. However, this approach relies on a block-based segmentation and so may not effectively capture geometric flows along longer and more complex contours. The choice of transforms is also limited to warped versions of separable 1D transforms. Note also that in order to fully specify the transform for encoding, it is necessary to transmit the choice of orientation in each block to the decoder. As an alternative to bandelets, Velisavljevic, et al, proposed *directionlets* [3], where anisotropic wavelet transforms are computed along discrete approximations of lines using integer lattices and are capable of exploiting directionality along lines in a given block of an image. Unlike bandelets, the filtering along a line is not strictly 1D and neighboring pixels to those directly on the line are used to compute the transform. As for bandelets, some overhead is required to indicate which directions are selected on a block by block basis. Thus both methods employ a block based partition and within each block the the geometric flow is parameterized using curves (bandelets) or line slopes (directionlets).

To motivate our approach we first observe that both bandelets and directionlets (and indeed conventional separable transforms) are designed by selecting *i) a set of directions or "paths" for traversing the pixels in the image and ii) a transform to be performed along these directions*. In both cases a single direction (line or curve) is chosen to traverse pixels within each block, so that we can see the transform as operating on a series of parallel paths through the pixels. This can be inefficient because a single direction may not represent well the geometric flow in a block. Moreover, even if a single direction provides a good representation, there exists some redundancy between those parallel paths that can only be exploited via a separable transform (as done in [2]).

The key novelty in our work is to show that it is possible to define transforms that operate on *any arbitrary trees*. Following the analogy, a tree defines paths to traverse the pixels in an image. The corresponding transform is invertible (and critically sampled if all pixels in the image are traversed.) This type of transform can enable directional representations, for example if trees are chosen to follow the geometric flow of an image. Our starting point is the lifting transform along trees we recently proposed in [4], an extension to 2D of earlier work in [5], in the context of data transport and compression in sensor networks. In this paper we extend the work in [4] by i) applying the tree-based lifting transform to image data, ii) introducing an approach to achieve a separable transform by using different trees to successively traverse the image, and iii) demonstrating that efficient representations can be obtained when choosing trees designed using image edge information. As in [2],[3] the proposed transform requires some overhead, namely, a description of the tree used to traverse image pixels. Overhead can be kept low in coding applications by, for example, defining tree growing techniques that can be run at encoder and decoder starting from some common information (e.g., a simplified edge map). Moreover, these methods are best suited to smooth images with small amounts of texture. In highly textured regions, the geometric flows in *bandelets* are hard to parameterize and the lines in *directionlets* will not exploit directionality effectively. Filtering along oriented trees has similar drawbacks in textured regions. Thus, we only consider applications to smooth images.

2. TREE BASED LIFTING TRANSFORMS

The idea of performing lifting transforms on arbitrary trees [4] originated in the context of sensor networks, where nodes are deployed in a highly irregular fashion and links / routes between nodes are ill-determined. Such a technique provides a natural means for compressing sensor data in a distributed manner as data is routed towards a given sink. A short summary of the work presented in [4] now follows, presented in the context of an image transform (assuming gray-scale images).

Consider an $M \times N$ image with MN pixels and assume a tree T has been selected that allows us to traverse the pixels in the image from leaves to the tree root. Let $x_{m,n}$ denote the intensity of pixel (m,n) in T, and suppose we index the root node of T by $(MN+1, MN+1)$. Let $\mathcal{C}_{m,n}$ and $\rho_{m,n}$ denote the set of children and the parent of pixel (m,n) in T, respectively. Finally, let depth(m,n) be the depth of pixel (m,n) in T, with depth$(MN+1, MN+1) = 0$. A lifting transform [6] can then be developed along T by specifying how to split pixels into even and odd sets and how to compute the predict and update filter coefficients at each level of decomposition.

For a 1-level transform, pixels are split into prediction and update sets according to their depth with respect to the root of the tree (the sink) which has depth zero, i.e., pixels of odd (even) depth are assigned as odd (even) pixels in the transform. A series of splitting trees can then be developed at each level of decomposition, where the tree at level j is constructed using the even pixels of the tree at the previous level. Thus, T_j will consist of pixels of even depth in T_{j-1} with a link between two pixels in T_j only if they are 2-hops apart in T_{j-1}. We would then split each T_j according to the parity of the depths. Using this construction, we denote the children and parent pixels of $(m,n) \in T_j$ as $\mathcal{C}_{m,n,j}$ and $\rho_{m,n,j}$, respectively.

Since the pixels are organized along a given tree, the lifting filter for a pixel could also be adapted to the structure of its neighbors along the tree as discussed in [4]. Ergo, the lifting filter for a pixel could be adapted to the relative distances between its neighbors in the tree, or on the number of paths merging into it, etc. For simplicity in experimentation, we employ the lifting filters proposed in our previous work. Given the sets of prediction and update pixels at each level of decomposition $j \in \{1, 2, \ldots, L\}$, denoted $\mathcal{P}_j$ and $\mathcal{U}_j$, and prediction and update operators given by $\mathbf{p}_{m,n,j}$ and $\mathbf{u}_{k,l,j}$ at nodes $(m,n) \in \mathcal{P}_j$ and $(k,l) \in \mathcal{U}_j$, respectively, and $\mathcal{N}_{m,n,j} = \mathcal{C}_{m,n,j} \cup \rho_{m,n,j}$, we compute the transform as follows:

For every $(k,l) \in \mathcal{P}_j$:

$$d_{k,l,j} = s_{k,l,j-1} + \sum_{(i_t,j_t)\in\mathcal{N}_{k,l,j}} \mathbf{p}_{k,l,j}(i_t,j_t)\, s_{i_t,j_t,j-1} \qquad (1)$$

and given every $d_{k,l,j}$, for each $(m,n) \in \mathcal{U}_j$ we have:

$$s_{m,n,j} = s_{m,n,j-1} + \sum_{(i_t,j_t)\in\mathcal{N}_{m,n,j}} \mathbf{u}_{m,n,j}(i_t,j_t)\, d_{i_t,j_t,j}. \qquad (2)$$

3. GENERALIZED TREE BASED TRANSFORMS

The tree based transform of [4] provides a general wavelet transform along a single tree. However, this is essentially a 1D transform in that it is run along 1D paths in the tree, and data from multiple 1D paths is only aggregated at the merge points in the tree. Thus, the transform proposed in [4] is well suited for sensor network data gathering but is not effective for image processing and representation, e.g., it may not exploit correlation across adjacent paths. Another drawback of this approach is that it will lead to long 1D filter kernels at higher levels of decomposition. This means smooth (update) coefficients may not be highly correlated along a splitting tree T_j, for large j, because the distance along the 1D path between successive coefficients can be large. This reduces the efficiency with which our transform can represent the image. In this paper, instead of using multiple levels of decomposition along a *single* tree, we propose a new approach to increase the efficiency of our representation, where the transform is computed over *multiple trees, with different orientations*, with a different tree being used at each level of decomposition.

We now outline a general framework for computing general 2D transforms along multiple trees in a separable manner, with the main goal of exploiting directionality in images. Suppose we are given an arbitrary method for constructing trees that follow the geometric flow in an image given a prespecified root node (or more generally, set of root nodes). For a one level decomposition (using a tree in one direction followed by 2 trees in another), we would first apply 1 level of decomposition along one tree T_1 oriented in one direction, then split the set of coefficients in T_1 into even (low pass) and odd (high pass) subsets (according to their depths in T_1) and then run a one level transform along a second tree $T_{1,l}$ and third tree $T_{1,h}$ (both oriented in totally different directions) over the even and odd subsets respectively. A simple example of such trees is shown in Figure 1, where links in the tree (denoted by black lines between pixels) are not allowed to cross edges in the images (denoted by blue asterisks). In this example, T_1 is a minimum spanning tree (MST) [7] of all of the pixels with the sink set as the center pixel in the left-most column. In addition, $T_{1,l}$ is an MST of all the even pixels in T_1 and $T_{1,h}$ is an MST of all the odd pixels in T_1, each with the sink set as the center pixel in the top row. While our examples are limited to MSTs, any type of spanning tree could be used in our method (i.e. shortest path trees, Steiner trees, etc. [7]). We could even construct spanning trees that trace out the directionality in the image by building links between pixels that follow the flows in an image, where flows could be defined arbitrarily (e.g. as in bandelets [2]). This would allow us to exploit directionality in a very general way.

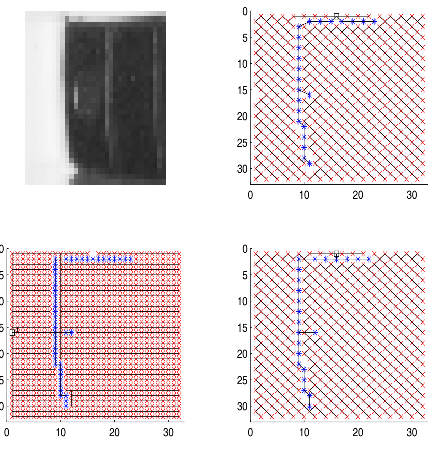


Fig. 1. Example of three trees used for 2-levels of decomposition. Red x's are non-edge pixels, black squares denote sinks, and blue asterisks are edges in image. Top left shows image block, bottom left T_1, top right $T_{1,l}$, bottom right $T_{1,h}$.

Clearly, both $T_{1,l}$ and $T_{1,h}$ are oriented vertically and so exploit different directional information than T_1 which is oriented horizontally. Note that each tree has links that follow (but do not cross over) edges and there are also links across paths (which we call merge points), making each transform inherently non-separable. Thus, our transform can exploit directionality in an image in arbitrary directions and in a non-separable fashion.

Applying this idea repeatedly at j-levels produces a subband decomposition similar to the decomposition of standard separable transforms as shown in Figure 2. Clearly, $E_{j,l}$ and $O_{j,l}$ represent the sets of even and odd nodes respectively taken from $T_{j,l}$, similarly for $E_{j,h}$ and $O_{j,h}$ with respect to $T_{j,h}$. Note that T_j is constructed using the even pixels of $T_{j-1,l}$, i.e., $E_{j-1,l}$. The Split(T_j) block splits nodes into E_j and O_j along T_j according to the method in Section 2, similarly for other split blocks. The P(T_j) and U(T_j) blocks compute predicts and updates along T_j according to the methods in Section 2, similarly for other predict and update blocks.

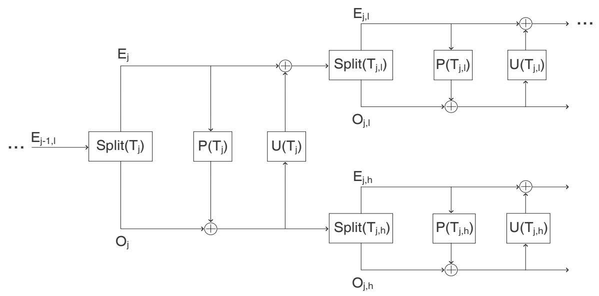


Fig. 2. Lifting structure at j-levels of decomposition.

Our proposed method can be viewed as a generalization of existing methods. As an example, consider the standard, separable extension of 1D transforms. This can be thought of as a special case of our transform, since filtering first along rows is equivalent to generating a tree with links along rows only (with appropriate subsampling and filtering) and filtering along columns second is equivalent to generating a tree with links along columns only (also with appropriate subsampling and filters). At higher levels of decomposition, similar trees are developed along the rows and columns of the corresponding subsampled images. Bandelets could also be developed in this context. In each block a tree would be generated with links that follow the geometric flow to perform the first stage of filtering. The second (still separable) stage of filtering would then be performed by constructing a tree with links running perpendicular to the geometric flow. In both cases, filtering is only done along 1D paths in a separable manner, thereby mimicking both standard techniques and bandelets. In Directionlets, directional information is captured along lines by applying integer lattices along these lines in a non-separable fashion. In our context, the combination of these lattices can be thought of as a single tree with the transform performed along this tree.

One major difference between what we propose here and conventional separable transforms is that pixel data is not always transformed along horizontal or vertical paths. Instead, the links of the tree can be forced to follow geometric flows throughout the image. The construction of transform trees can also be arbitrary, allowing great flexibility in exploiting directionality. The chosen trees may also have multiple merge points (e.g. pixels with multiple children as shown in Figure 1) that allow the transform to directly exploit correlation across adjacent paths in a non-separable fashion, a feature not found in standard methods nor in bandelets. Furthermore, directionlets only locally exploit directional information along lines on a block by block basis, and so may not represent data along arbitrarily complex contours as effectively as our proposed method.

4. EXPERIMENTAL RESULTS

Since image edges can be thought of as boundaries between relatively smooth regions, an edge map (loosely) defines the geometric flows in an image. If no transforms are performed across edges, this should contribute to reductions in the relative amounts of high pass energy in the transform. Thus, as a first step to identifying good transform trees, it is reasonable to compute an edge map and to then develop trees that trace out edges without crossing them. Such trees can be constructed in a unique manner given the edge map, so that in a coding application only the edge map needs to be sent to the decoder.

For the purposes of experimentation, first we generate an edge map of the image using standard edge detection methods (Robert's method is used for the following experiments.) This edge map is used to construct trees that do not have links that cross over edges in the image. This also tends to force pixels around edges to flow through the tree along the flow of the image edges.

In this case, we separate pixels into edge and non-edge pixels. Then for each tree we construct, we impose the following link constraints: (1) edge pixels will only be linked to other edge pixels and (2) non-edge pixels will only be linked to other non-edge pixels. Thus, any path in a given tree will contain only edge pixels or non-edge pixels but not both. We consider a four subband decomposition. Then, under the given link constraints stated above, we construct three MSTs T_1, $T_{1,l}$, and $T_{1,h}$ using the same method described in the example of Section 3. A one level transform is then performed along each tree. This produces four subbands given by the sets of LL pixels $E_{1,l}$, LH pixels $O_{1,l}$, HL pixels $E_{1,h}$, and HH pixels $O_{1,h}$. We applied this preliminary method to two 256×256 test images shown in Figure 3.

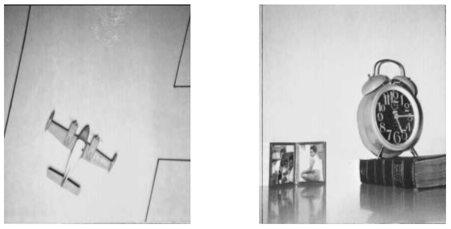
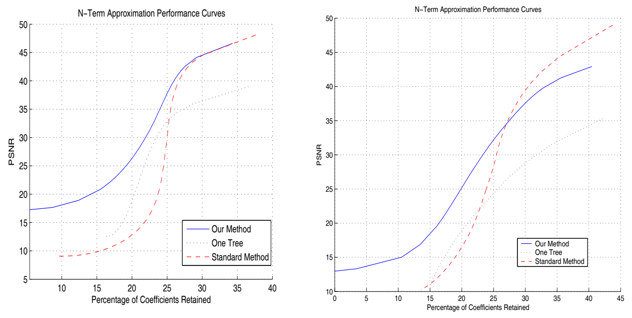


Fig. 3. Original images.

We evaluate this approach using the N term non-linear approximation problem (outlined in [3]) where we keep the N largest coefficients and set the rest to zero. As discussed in [3], this is a good preliminary indicator of coding performance. We compute the PSNR of each image as a function of the percentage of retained coefficients and compare a four subband decomposition using our proposed transform against a four subband decomposition for a standard separable 2D transform using the Haar basis and a four level decomposition using a single MST rooted at the center pixel of the leftmost column. The performance curves are shown in Figures 4(a) and 4(b). Our proposed method is clearly better than the standard method using a Haar basis. However, the standard method outperforms our method when more pixels are retained for the clock image. One possible reason is that our current filter design leads to a bi-orthogonal decomposition resulting in predict filters having more energy than the update filters. We are currently studying techniques to normalize the various filters (note that the filters depend on the structure of the tree). Secondly, thicker image edges could be two or three pixels tall and wide, so our transform may not perform well near thick edges. Nonetheless, our transform still does better for smaller percentages. This is not the case in the airplane image since edges are not as thick nor as detailed as in the clock image. Figures 5 and 6 show the reconstruction from our method (left) and from a Haar basis (right) obtained by retaining 25% of the coefficients. Even though the PSNR values are quite close for the clock, the reconstruction using our method is much sharper around the edges and contours. In particular, the numbers on the clock, the frame, and images inside the frame are quite blocky for the standard method, whereas they are much sharper for our transform. The airplane image has a PSNR gain of about 8 dB for our method over the standard method, and the reconstruction clearly reflects this improvement. These preliminary results are meant to demonstrate the potential merits of the proposed representation, but further work is needed to develop image coding applications. In particular there will be a trade-off between how well a tree captures directionality and the number bits needed to represent it. In terms of bits per pixel (bpp), the JBIG algorithm [8] is capable of compressing the edge map for the airplane image to about 0.0117 bpp and about 0.0118 bpp for the clock image. Since our proposed trees are constructed from the edge maps alone, the required overhead is very low.



(a) Airplane (b) Clock

Fig. 4. PSNR versus percentage of retained coefficients.

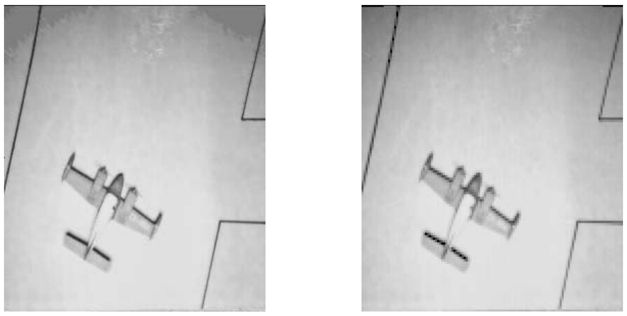


Fig. 5. Reconstruction with 25% for our method (left) and standard (right).

These improvements are the result of performing multiple levels of decomposition along multiple trees, where each tree exploits the inherent directionality in each image differently. Moreover, the transform is not performed across edges as in the standard methods, eliminating the inefficiency created by the large high frequency coefficients around the edges found there. Some cross path correlation is also exploited around pixels where paths in each tree merge, allowing us to achieve more de-correlation across pixels than standard methods.

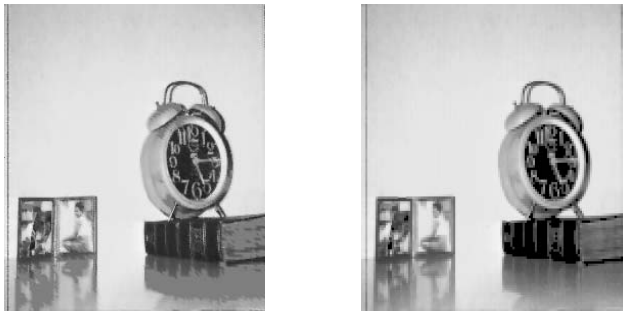


Fig. 6. Reconstruction with 25% for our method (left) and standard (right).

5. CONCLUSIONS AND FUTURE WORK

We have developed a general 2D transform based on wavelet lifting along arbitrary trees. This method can be viewed as a generalization of existing methods. The arbitrary nature of the trees allows us to exploit directionality and local homogeneity in a variety of ways. Since multiple trees can be applied to different stages of the transform, our method can repeatedly exploit directionality and homogeneity across all levels of decomposition. The results using N-term non-linear approximations are promising. We are currently investigating the application of this method to image coding.

6. REFERENCES

[1] M. Do and M. Vetterli, "The contourlet transform: An efficient directional multiresolution image representation," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2091– 2106, December 2005.

[2] E. Le Pennec and S. Mallat, "Sparse geometric image representations with bandelets," *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 423– 438, April 2005.

[3] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P.L. Dragotti, "Directionlets: Anisotropic multidirectional representation with separable filtering," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1916– 1933, July 2006.

[4] G. Shen and A. Ortega, "Optimized distributed 2D transforms for irregularly sampled sensor network grids using wavelet lifting," in *Proceedings of the 2008 International Conference on Acoustics, Speech and Signal Processing - ICASSP08*, Las Vegas, Nevada, USA, April 2008.

[5] A. Ciancio and A. Ortega, "A distributed wavelet compression algorithm for wireless sensor networks using lifting," in *Proceedings of the 2004 International Conference on Acoustics, Speech and Signal Processing - ICASSP04*, Montreal, Canada, May 2004.

[6] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," Technical report 1995:6, Industrial Mathematics Initiative, Department of Mathematics, University of South Carolina, (ftp://ftp.math.sc.edu/pub/imi_95/imi95_6.ps), 1995.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, 2nd edition, 2001.

[8] "JBIG, Progressive Bi-level Image Compression," ISO/IEC ITU Recommendation T.82, 1993.