LEARNING A WAVELET TREE FOR MULTICHANNEL IMAGE DENOISING

Zhen James Xiang, Zhuo Zhang, Pingmei Xu, Peter J. Ramadge

Dept. of Electrical Engineering, Princeton University, Princeton NJ

ABSTRACT

We propose a new multichannel image denoising algorithm. To exploit important inter-channel dependencies, we first use dynamic programming to learn an explicit dyadic tree representation of the common structure of the channels. Based on this dyadic tree, optimal Haar wavelet thresholding is then applied to denoise the image. In addition to the original channels, the algorithm can employ multiple derived channels to improve tree learning. Experimental results confirm that the approach improves multichannel image denoising performance both in PSNR and in edge preservation.

***Index Terms*—** Wavelet transforms, image enhancement, signal denoising, dynamic programming.

1. INTRODUCTION

Improving the performance of multichannel (color, multispectral) image denoising beyond the baseline of separate channel denoising requires exploiting inter-channel signal dependency. Hence a fundamental problem in multichannel denoising is learning signal structure from noisy multichannel data. Intuitively, although the image channels have different pixel values, a *"common structure"* is inherited from the imaged scene. This is exhibited in common contours, edges, textures, etc., in the channels. Existing multichannel denoising algorithms use this intuition implicitly, e.g. by enforcing inter-channel dependencies on wavelet shrinkage thresholds [1, 2] or aggregating color channels into a "luminance" channel via color-space transformation (e.g. RGB to YUV) or more refined adaptive transformations [3, 4].

The key contribution of this paper is a framework and algorithm to learn joint signal structure from multichannel data as a foundation for subsequent image denoising (Fig. 1). Specifically, we learn a single, regularized dyadic tree that jointly represents the structure of the image channels. This tree is then used as a common decomposition structure for subsequent wavelet denoising of each channel. Hence our approach explicitly models the aforementioned *"common structure"* using an adaptive dyadic tree.

To define dyadic trees we start with some basic notation. A single channel image is a function $f\colon [0,1]^d \to \mathbb{R}$ ($d = 2, 3$). We assume f has spatial resolution 2^m, i.e., pixels/voxels are subcubes of $[0,1]^d$ of side-length 2^{-m}.

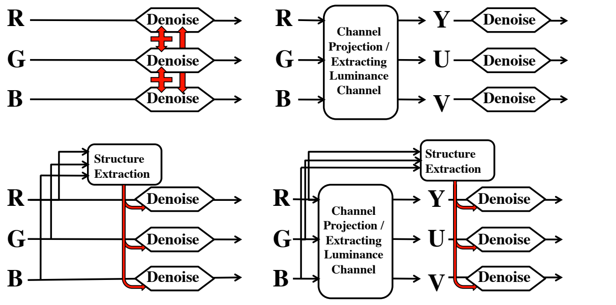

Fig. 1. Schematic of common multichannel denoising (top) and our approach (bottom). R,G,B, etc. are examples of image channels.

A dyadic tree $\mathfrak{T}$ is a binary tree satisfying: **(c1)** the root node is $J_{0,0} = [0,1]^d$; **(c2)** non-leaf node $J_{k,l}$ has color $c \in \{1, \ldots, d\}$. Its children, $J_{k+1,2l}$, $J_{k+1,2l+1}$ are congruent hyperrectangles obtained by cutting $J_{k,l}$ with the $d-1$ dimensional hyperplane through the center of $J_{k,l}$ perpendicular to axis-c; and **(c3)** leaf nodes are pixels/voxels.

The Haar transform and the two morphological Haar wavelet transforms [5] of f are computed on such a dyadic tree $\mathfrak{T}$ as follows. Associate leaf nodes $J_{dm,l}$ with f's values on these pixels/voxels $\Psi_{dm}(l) = f(x)|_{x \in J_{dm,l}}$ and non-leaf node $J_{k,l}$ with value $\Psi_k(l)$ calculated from the values of its children, $\Psi_{k+1}(2l)$ and $\Psi_{k+1}(2l+1)$, using:

$$\text{(Haar) } \Psi_k(l) = \left(\Psi_{k+1}(2l) + \Psi_{k+1}(2l+1)\right)/\sqrt{2}, \quad (1)$$
$$W_{k,l}(f) = \left(\Psi_{k+1}(2l) - \Psi_{k+1}(2l+1)\right)/\sqrt{2}. \quad (2)$$

$$\text{(Morphological) } \Psi_k(l) = \Psi_{k+1}(2l) \vee \Psi_{k+1}(2l+1), \quad (3)$$
$$W^{\vee}_{k,l}(f) = \Psi_{k+1}(2l) - \Psi_{k+1}(2l+1). \quad (4)$$

$$\text{(Morphological) } \Psi_k(l) = \Psi_{k+1}(2l) \wedge \Psi_{k+1}(2l+1), \quad (5)$$
$$W^{\wedge}_{k,l}(f) = \Psi_{k+1}(2l) - \Psi_{k+1}(2l+1). \quad (6)$$

where $\vee = \max$ and $\wedge = \min$.

Dyadic trees have proved useful in machine learning [6] and image processing [7]. Willett and Nowak [7] use dyadic trees to model sublevel sets (i.e. image contours) and [8] connects dyadic trees with wavelet representations of real-valued functions. Moreover, in [9], the complexity of f under an *adaptive* dyadic tree $\mathfrak{T}$, measured by

$$E_{\mathfrak{T}}(f) = \sum_{k=0}^{dm-1} \sum_{l=0}^{2^k-1} \alpha_{k+1} \left(|W_{k,l}^{\vee}(f)| + |W_{k,l}^{\wedge}(f)| \right), \quad (7)$$

regularizes the single-channel image denoising problem:

$$\min_{(f,\mathfrak{T})} : \quad \|f - \tilde{f}\|_2^2 + \lambda E_{\mathfrak{T}}(f), \quad (8)$$

where $\tilde{f}$ is the noisy observation. This is solved by iterated optimizations over $\mathfrak{T}$ and f. Because $\max$, $\min$ operators appear in (7), second order cone programming (SOCP) is required to solve (8) for f and this limits the method's speed.

2. PROBLEM FORMULATION

We base the denoising component of our method on the Haar wavelet transform (1-2). This is an orthogonal transform, so minimizing the MSEs in each band will ensure the global optimization, and we can denoise by hard/soft thresholding the wavelet coefficients. This avoids the need for SOCP.

Paralleling (7), bring in a complexity measure of f based on the 0- or 1-norm of its Haar wavelet coefficients on tree $\mathfrak{T}$:

$$H_{\mathfrak{T}}(f) = \sum_{k=0}^{dm-1} \sum_{l=0}^{2^k-1} 2^{-(1-s)k} |W_{k,l}(f)|^p, \quad p \in \{0, 1\} \quad (9)$$

where s controls wavelet coefficient weighting across levels. We then formulate the multichannel denoising problem for an n-channel image $f = (f_1, f_2, \ldots, f_n)$ as

$$\min_{(f_1,\ldots,f_n,\mathfrak{T})} : \textstyle\sum_{j=1}^{n} (\mu_j \|f_j - \tilde{f}_j\|_2^2 + \lambda_j H_{\mathfrak{T}}(f_j)), \quad (10)$$

where $\tilde{f}_j$ is the noisy observation of the j-th channel. Note (10) is a joint optimization. One solution approach is to first learn a tree that represents the multichannel image structure:

$$\mathfrak{T}_{opt} \in \arg\min_{\mathfrak{T}} \{\textstyle\sum_{j=1}^{n} \lambda_j H_{\mathfrak{T}}(\tilde{f}_j)\}. \quad (11)$$

Then, with a fixed $\mathfrak{T}_{opt}$, denoise each channel by solving:

$$\hat{f}_j \in \arg\min_{f_j} \{\mu_j \|f_j - \tilde{f}_j\|_2^2 + \lambda_j H_{\mathfrak{T}_{opt}}(f_j)\}. \quad (12)$$

These steps can then be iterated, although in Section 4 one iteration proves adequate. Separating tree learning (11) from (10) as above, may be suboptimal but brings other advantages: (a) Faster denoising because (11) and (12) can be efficiently solved; (b) We can use existing wavelet denoising algorithms that can adaptively determine thresholds by noise level estimation (no need to set μ_j manually) and for $p = 0, 1$ (12) has a close form solution via wavelet hard/soft thresholding; (c) Tree learning can be augmented to explicitly include derived channels such as Y in the YUV representation, brightness in the CB color model and value in the HSV color model [3]. We can even include random linear combinations of the f_j channels. In what follows let $g_1, g_2, \ldots, g_{n'}$ denote the channels to be used for learning. Using the g_j and (9), equation (11) becomes:

$$\begin{aligned} \mathfrak{T}_{opt} &\in \arg\min_{\mathfrak{T}} \{\textstyle\sum_{j=1}^{n'} \lambda_j H_{\mathfrak{T}}(g_j)\} \quad (13) \\ &= \arg\min_{\mathfrak{T}} \{\sum_{j=1}^{n'} \lambda_j \sum_{k=0}^{dm-1} \sum_{l=0}^{2^k-1} 2^{-(1-s)k} |W_{k,l}(g_j)|^p\} \end{aligned}$$

3. ALGORITHMS

We solve (13) by dynamic programing. Let $\vec{g}$ denote the vector function $(g_1, g_2, \ldots, g_{n'})$ and for a hyperrectangle B, let $\mathrm{avg}_B\, \vec{g}$ be the average value of $\vec{g}$ on B. Each leaf node B of a dyadic tree $\mathfrak{T}$ is a hyperrectangle at level $k = dm$ (a pixel/voxel) with $\mathrm{avg}_B\, \vec{g} = \vec{g}(x)|_{x \in B}$. Since B cannot be split, it is the optimal tree rooted at B and has zero cost. We store this information in dictionary $\mathcal{D}_{dm}$ as entry $a(B) = (\mathrm{avg}_B\, \vec{g}, B, 0)$. We now recursively construct similar dictionaries for tree levels $k = dm - 1, \ldots, 0$. Assume we have $\mathcal{D}_{k+1}$. A hyperrectangle B with color c is valid at level k if there exist hyperrectangles B_U^c and B_V^c at level $k + 1$ with

Algorithm 1 Dynamic Program to Solve (13)

```
1:  For all $2^{dm}$ leafs $B$ store entry:
        $a(B) = (\vec{g}(x)|_{x \in B},\ \{B\},\ 0)$ in dictionary $\mathcal{D}_{dm}$.
2:  for $k = dm-1, \ldots, 1, 0$ do
3:      Initialize $\mathcal{D}_k = \phi$.
4:      for all $a(U) \in \mathcal{D}_{k+1}$ do
5:          for $c = 1, 2, \ldots, d$ do
6:              if the $c$-th dimensional side-length of $U$ is 1 then
7:                  continue
8:              end if
9:              $V \leftarrow$ The unique hyperrectangle that could be the
                sibling of $U$ under a parent node $B$ of color $c$.
10:             $B \leftarrow U \cup V$.
11:             Retrieve entries $a(U)$ and $a(V)$ from $\mathcal{D}_{k+1}$:
                    $a(U) = (\mathrm{avg}_U(\vec{g}),\ \mathfrak{T}_{U,opt},\ L(\mathfrak{T}_{U,opt}))$,
                    $a(V) = (\mathrm{avg}_V(\vec{g}),\ \mathfrak{T}_{V,opt},\ L(\mathfrak{T}_{V,opt}))$.
12:             $\mathrm{avg}_B\, \vec{g} \leftarrow \{\mathrm{avg}_U(\vec{g}) + \mathrm{avg}_V(\vec{g})\}/2$,
                $\mathfrak{T}_{B,opt} \leftarrow \{\mathfrak{T}_{U,opt} \leftarrowtail B \rightarrowtail \mathfrak{T}_{V,opt}\}$,
                $L(\mathfrak{T}_{B,opt}) \leftarrow \mathcal{G}(a(U), a(V))$
13:             Retrieve the existing entry for $B$ from $\mathcal{D}_k$:
                    $a'(B) = (\mathrm{avg}'_B\, \vec{g},\ \mathfrak{T}'_{B,opt},\ L(\mathfrak{T}'_{B,opt}))$.
14:             if $a'(B) = \phi$ or $L(\mathfrak{T}_{B,opt}) < L(\mathfrak{T}'_{B,opt})$ then
15:                 Replace $a'(B)$ with the entry:
                        $a(B) = (\mathrm{avg}_B\, \vec{g},\ \mathfrak{T}_{B,opt},\ L(\mathfrak{T}_{B,opt}))$.
16:             end if
17:         end for
18:     end for
19: end for
20: Retrieve $a([0,1]^d)$ from $\mathcal{D}_0$ and output $\mathfrak{T}_{[0,1]^d,opt}$.
```

entries $a(B_U^c)$ and $a(B_V^c)$ in $\mathcal{D}_{k+1}$ such that splitting B using dimension c yields B_U^c and B_V^c. Connecting the optimal trees $\mathfrak{T}_{B_U^c,opt}$ and $\mathfrak{T}_{B_V^c,opt}$ rooted at B_U^c and B_V^c, respectively, to root B yields tree $\mathfrak{T}_B = \{\mathfrak{T}_{B_U^c} \leftarrowtail B \rightarrowtail \mathfrak{T}_{B_V^c}\}$. The cost of this tree, $L(\mathfrak{T}_B)$, is the loss (13) calculated within hyperrectangle B by replacing $\sum_{k=0}^{dm-1}$ in (13) with $\sum_{k'=0}^{dm-k-1}$. This cost is minimized by optimizing over the cut direction c. Using (13), this can be done recursively as follows:

$$\min_c \{ L(\mathfrak{T}_{B_U^c,opt}) + L(\mathfrak{T}_{B_V^c,opt}) +$$
$$\sum_{j=1}^{n'} \lambda_j 2^{-(1-s)k} \, |(\frac{1}{\sqrt{2}})^{dm-k} 2^{dm-k-1} (\operatorname*{avg}_{B_U^c} g_j - \operatorname*{avg}_{B_V^c} g_j)|^p \, \}.$$

The expression inside the curly brackets can be computed from $a(B_U^c)$ and $a(B_V^c)$ and is denoted as $\mathcal{G}(a(B_U^c), a(B_V^c))$. This yields the optimal tree $\mathfrak{T}_{B,opt}$ rooted at B and this is stored in $\mathcal{D}_k$ as entry $a(B) = (\text{avg}_B\, \vec{g}, \mathfrak{T}_{B,opt}, L(\mathfrak{T}_{B,opt}))$. In this way, we compute $\mathcal{D}_k$ from $\mathcal{D}_{k+1}$. $\mathcal{D}_0$ has only one entry and gives $\mathfrak{T}_{opt}$. This bottom up dynamic programming algorithm shown in **Algorithm 1**.

With $\mathfrak{T}_{opt}$ determined, (12) can be optimally solved by standard wavelet denoising algorithms. To do so, transform f_j to a 1-D function $\{f_j(x) | x \in J_{dm,l}\}_{l=0}^{2^{dm}-1}$ and apply the wavelet denoising algorithm. In (13), λ_j controls the relative weight of the j-th channel. If prior information is at hand, assign larger λ_j to channels with better structural information. Otherwise, the noise level of g_j determines λ_j, with λ_j smaller for noisier channels. We set $\lambda_j = 1/\hat{n}_j$ where $\hat{n_j} = \text{median}\{|x_i|\}/0.6745$ is Donoho's noise estimator with $\{x_i\}$ the wavelet coefficients of g_j in the HH band [10].

4. EXPERIMENTS

We first tested performance of our algorithm on single channel (grayscale) images. This benchmarks the algorithm's baseline performance. We compared Haar wavelet threshold denoising using: (a) a fixed $\mathfrak{T}$ (column-wise(c), row-wise(r) or column-row-mixed (m) trees and report the best), (b) conventional 2D separable wavelets, (c) our tree $\mathfrak{T}_{opt}$ learnt from the signal. We also compare with (d) the morphological denoising algorithm of [9] and (e) the total variation method. For wavelet methods, we divided each image into 128×128 blocks, applied the denoising to each block, and incorporated partial cycle spinning by averaging the results of 64 shifted, processed, and inverse shifts of each block (shifts: $(i,j), 0 \leq i,j \leq 7$ pixels). In our method, we set $(p,s) = (1, 1.5)$ for soft thresholding and $(0.001, 1)$ for hard thresholding and did one iteration: find $\mathfrak{T}_{opt}$ and denoise. For all methods, we grid-searched for the best threshold or regularization coefficient.

Performance (Fig. 2) improves as we move from (a) fixed $\mathfrak{T}$, to (b) 2D separable filter, to (c) adapted $\mathfrak{T}_{opt}$. On a single channel our algorithm is competitive with total variation denoising but not quite as good as morphological denoising with adaptive trees [9]. This penalty reflects the suboptimality of our scheme. However, our algorithm is much faster since SOCP is replaced with wavelet shrinkage.

To test the multichannel denoising performance we used the color images in Fig. 4. We first used the "kitchen" image corrupted by white Gaussian noises of input PSNR 20dB (independent across channels). We tested performance using various $\{g_j\}$ in (13) to learn the tree: (a) each separate color channel or all color channels; (b) luminance channels: Y in YUV, Brightness (Br) in CB, Value (Val) in HSV [3], and all of these channels. (p,s) are set to $(0.001, 1)$ and λ_j are set based on Donoho's noise estimator. Denoising was accomplished using the Rice Wavelet Toolbox. For each trial, we divided the image into 128×128 blocks and applied partial cycle spinning of 16 shifts ($(i,j), 0 \leq i,j \leq 3$ pixels). We report denoising performance (PSNR - averaged over 10 random realizations) on each channel and the whole image (Fig. 3). Trees learnt from luminance channels performed better than those learnt from color channels and each color channel had better denoising performance using trees learnt from other color channels. We posit that this is due to overfitting: trees learnt from a color channel also fit the channel

	Soft Thresholding			Hard Thresholding				
ID	fix($\mathfrak{T}$)	2D	$\mathfrak{T}_{opt}$	fix($\mathfrak{T}$)	2D	$\mathfrak{T}_{opt}$	[9]	TV
1	16.1(m)	16.7	17.6	18.4(m)	19.4	20.5	20.9	18.7
2	17.9(m)	18.9	18.9	19.0(m)	19.9	20.1	21.0	20.2
3	21.4(m)	21.8	22.6	22.2(m)	22.4	23.0	23.9	23.2
4	19.5(m)	19.9	20.1	20.2(m)	20.6	20.8	21.7	21.3

Fig. 2. Single channel denoising. Test images (indexed as 1 to 4 from left to right). Output PSNR (dB) is shown with best fixed tree $\mathfrak{T}$ for input PSNR=10dB (i.i.d. Gaussian noise).

	Color channels used to learn the tree			
Channel	R	G	B	R,G,B
R	**27.3**(.07)	**28.2**(.06)	**28.0**(.06)	**28.4**(.05)
G	**28.2**(.03)	**27.3**(.06)	**28.1**(.03)	**28.4**(.05)
B	**28.1**(.05)	**28.2**(.04)	**27.3**(.06)	**28.4**(.04)
Image	**27.8**(.02)	**27.8**(.03)	**27.8**(.03)	**28.4**(.01)
	Luminance channels used to learn the tree			
Channel	Y	Br [3]	Val [3]	Y,Br,Val
R	**28.6**(.05)	**28.6**(.05)	**28.4**(.04)	**28.7**(.05)
G	**28.3**(.05)	**28.7**(.02)	**28.4**(.03)	**28.6**(.04)
B	**28.5**(.06)	**28.6**(.05)	**28.4**(.03)	**28.7**(.04)
Image	**28.5**(.02)	**28.6**(.02)	**28.4**(.01)	**28.7**(.01)

Fig. 3. Multichannel denoising (Haar wavelet, hard thresholding using tree learnt from indicated channels). Mean PSNR (dB) and standard deviation over 10 noise realizations reported.

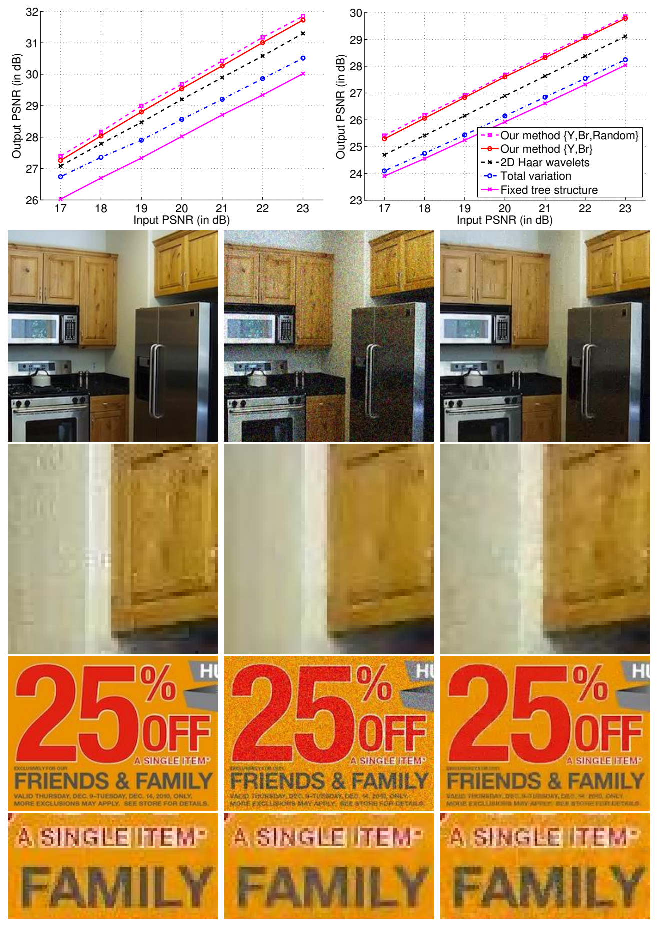

Fig. 4. Top: denoising performance vs input PSNR for image "kitchen" (left) and "coupon" (right). 2nd & 4th row: original, noisy (PSNR=20dB), our result {Y,Br,Random}. 3rd & 5th row show denoised patches: ours {Y,Br,Random}, 2D wavelet, total variation.

noise, which then can't be removed in the subsequent denoising using the same tree. Most importantly, using the {Y, Br, Val} combination for tree learning gave better performance than using any single luminance channel. So by incorporating multiple (derived) channels for tree learning our algorithm can boost denoising performance.

Finally, we compared performance for different input noise levels. For tree learning, we used the {Y, Br} channels ($n' = 2$) and {Y, Br, and 4 random mixtures} ($n' = 6$). We used cycle spinning with 64 shifts ($(i, j), 0 \leq i, j \leq 7$ pixels). Other aspects remained the same as above. We compared our algorithm with: 2D Haar wavelet hard thresholding, Haar wavelet thresholding with a fixed $\mathfrak{T}$, and total variation denoising. The image was transformed into YUV coordinates for denoising. The results (Fig. 4) indicate that our algorithm had the best performance and that adding the 4 random mixture channels gave an small extra PSNR gain. The zoomed patches in Fig. 4 indicate some artifacts of our method but also suggest that it has advantages in edge preservation.

5. CONCLUSION

We have modeled the common structure among image channels as a shared dyadic tree and used this tree for wavelet denoising. The experimental results show that for the same Haar wavelet denoising algorithm, using different dyadic trees for the wavelet decomposition yields very different performance. Our algorithm efficiently learns the optimal dyadic tree from combinations of the channels, including the original color channels or derived luminance channels, and outperforms conventional wavelet or total variation denoising methods. Best performance is achieved when using multiple mixed channels to jointly estimate the dyadic tree.

6. REFERENCES

[1] P. Scheunders, "Wavelet thresholding of multivalued images," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 475–483, 2004.

[2] F. Luisier and T. Blu, "SURE-LET multichannel image denoising: interscale orthonormal wavelet thresholding," *IEEE Transactions on Image Processing*, vol. 17, no. 4, pp. 482–492, 2008.

[3] T.F. Chan, S.H. Kang, and J. Shen, "Total variation denoising and enhancement of color images based on the CB and HSV color models," *Journal of Visual Communication and Image Representation*, vol. 12, no. 4, pp. 422–435, 2001.

[4] N.X. Lian, V. Zagorodnov, and Y.P. Tan, "Edge-preserving image denoising via optimal color space projection," *IEEE Transactions on Image Processing*, vol. 15, no. 9, pp. 2575–2587, 2006.

[5] H. Heijmans and J. Goutsias, "Nonlinear multiresolution signal decomposition schemes-Part II: Morphological wavelets," *IEEE Transactions on Image Processing*, vol. 9, no. 11, pp. 1897–1913, 2000.

[6] C. Scott and R.D. Nowak, "Minimax-optimal classification with dyadic decision trees," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1335–1353, 2006.

[7] R.M. Willett and R.D. Nowak, "Minimax optimal level-set estimation," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2965–2979, 2007.

[8] Z.J. Xiang and P.J. Ramadge, "Morphological wavelets and the complexity of dyadic trees," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010.

[9] Z.J. Xiang and P.J. Ramadge, "Morphological wavelet transform with adaptive dyadic structures," in *IEEE International Conference on Image Processing*, 2010.

[10] D.L. Donoho and I.M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage.," *Journal of the American Statistical Association*, vol. 90, no. 432, 1995.