Wavelet Bayesian Network Image Denoising

Jinn Ho and Wen-Liang Hwang, *Senior Member, IEEE*

***Abstract*— From the perspective of the Bayesian approach, the denoising problem is essentially a prior probability modeling and estimation task. In this paper, we propose an approach that exploits a hidden Bayesian network, constructed from wavelet coefficients, to model the prior probability of the original image. Then, we use the belief propagation (BP) algorithm, which estimates a coefficient based on all the coefficients of an image, as the maximum-a-posterior (MAP) estimator to derive the denoised wavelet coefficients. We show that if the network is a spanning tree, the standard BP algorithm can perform MAP estimation efficiently. Our experiment results demonstrate that, in terms of the peak-signal-to-noise-ratio and perceptual quality, the proposed approach outperforms state-of-the-art algorithms on several images, particularly in the textured regions, with various amounts of white Gaussian noise.**

***Index Terms*— Bayesian network, image denoising, wavelet transform.**

I. Introduction

COMPLEX phenomena usually involve a large number of hidden variables and data sources. Graphical models provide a unifying framework for modeling the probability distributions of such phenomena by decomposing joint probability distributions into a set of local constraints and dependencies [1]. After formulating a problem as a graphical model, a wide range of statistical learning and inference algorithms can be applied directly to derive a solution. Bayesian networks are probably the most popular type of (directed) graphical model. In this paper, our objective is to construct a Bayesian network from a single image for denoising purposes. To do this, we need to overcome two difficulties: 1) constructing a Bayesian network is computationally inefficient and 2) the data over-fitting problem, which exaggerates minor fluctuations in the input data.

The construction of a Bayesian network involves prior knowledge of the probability relationships between the variables of interest. Learning approaches are widely used to construct Bayesian networks that best represent the joint probabilities of training data [2]–[5]. In practice, an optimization process based on a heuristic search technique is used to find the best structure over the space of all possible networks. However, the approach is computationally intractable because it must explore several combinations of dependent variables to derive an optimal Bayesian network. The difficulty is resolved in this paper by representing the data in wavelet domains and restricting the space of possible networks by using certain techniques, such as the "maximal weighted spanning tree" (MWST). Three wavelet properties - sparsity, clustering, and persistence - can be exploited to reduce the computational complexity of learning a Bayesian network. First, the wavelet transform of a natural image tends to be sparse with large coefficients at the edges. The sparsity reduces the number of variables required to construct a graph. Second, the adjacent wavelet coefficients tend to have similar values as a cluster. Third, wavelet coefficients at the same location and orientation tend to be positively correlated in adjacent scales.

The over-fitting problem occurs because the underlying network is too complex; for example, there may be too many parameters for the number of observations [6]. In the effort of using hidden Markov tree (HMT) model [7]–[9] to capture the joint statistics of wavelet coefficeints, the marginal probability of each wavelet coefficient is modeled as a mixed density function with a hidden state variable; for example, an M-state Gaussian mixture model for a wavelet coefficient consists of M states and M Gaussian conditional probability density functions (pdf), one for each state. To generate wavelet coefficients, the HMT first draws a state value s based on the pdf p, and then draws an observation according to the conditional probability f of the state. The pdf of the wavelet coefficient is given by $\sum_{s=1}^{M} p(s) f(.|s)$, where the conditional pdf $f(.|s)$ is written as a parametric formula that depends on certain parameters. Because several parameters are used to estimate a wavelet coefficient, the HMT approach can only be used to model the marginal probability when the number of training images is large; then the underlying parameters can be estimated accurately. If there is only one image, the over-fitting problem under the HMT approach would be severe. Thus, instead of associating each wavelet coefficient with a random variable, we split all the wavelet coefficients into equal-size blocks and assume that the blocks are independently sampled from a matrix of random variables (called a wavelet patch). The approach allows us to estimate non-parametric statistics, which do not require a pdf assumption, from the samples in the wavelet patch.

The image denoising problem is particularly serious in modern image capturing devices because the increase in the sensor's density per unit area of a chip reduces the signal-to-noise (SNR) and increases the capturing device's sensitivity to noise [10]. The state-of-the-art denoising algorithms are based on the non-local means approach [11]–[13], which exploits the self-similarity and redundancy in an image. The most representative approach is the block matching and 3-D filtering

Manuscript received October 5, 2011; revised July 6, 2012; accepted August 26, 2012. Date of publication September 21, 2012; date of current version February 4, 2013. This work was supported in part by the National Science Council, under Grant NSC-100-2221-E-001-017. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Hitoshi Kiya.

J. Ho is with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan (e-mail: hjinn@iis.sinica.edu.tw).

W.-L. Hwang is with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan, and also with the Department of Information Management, Kainan University, Luchu 33857, Taiwan (e-mail: whwang@iis.sinica.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2012.2220150

(BM3D) algorithm [13]. It combines similar 2-D patches that can be overlapped to form a 3-D group, and then uses collaborative 3-D filtering techniques to perform non-local filtering. The filtered blocks are returned to their original positions, and the final estimate of a pixel is computed as the weighted average of the estimates of the pixel in several different blocks. This simple approach is very efficient and it yields better results than regular non-local means approaches. In [14], a denoising Wiener filter, motivated by the statistical analysis of the performance bounds of patch-based methods, is proposed. The filter's parameters are estimated from geometrically as well as photometrically similar patches. Recent developments in sparse representation have been used together with the non-local means approach for noise reduction purposes [15], [16]. The sparse model approach assumes that image patches can be represented sparsely by an over-complete redundant dictionary, which can be learned from a family of training data. In [17], Milanfar demonstrates that the non-local means approach as well as recent multi-dimensional filtering approaches (e.g., bilateral filtering and kernel regression) are operations that are adaptive to the local structure of an image. He also presents a general framework for understanding the basic principles behind the approaches.

The Bayesian approach is also widely used to resolve the image denoising problem. The Bayesian formula indicates that the denoising problem is essentially a prior probability modeling and estimation task. If $y = x + n$, where n is white Gaussian noise with known variance, then the Bayesian formula is $P(x|y) = P(y|x)P(x)$, where $P(y|x) = P_n(y-x)$ is the noise probability. The maximum-a-posterior (MAP) solution of $P(x|y)$ is determined by the priori probability $P(x)$. The structure of image priors is usually modeled by Markov Random Fields (MRFs), where the probability of a pixel depends solely on the joint probability of the pixels in its neighborhood. According to the Hammersley-Clifford theorem, the probability distribution of an MRF is the Gibbs distribution whose energy function is the sum of the potential functions defined on the cliques (i.e., maximal complete subgraphs) in image neighborhoods.

Many wavelet-based denoising algorithms integrate the wavelet properties in MRFs to retain the structure of a denoised image. The main differences between the algorithms are the methods used for neighborhood selection, the modeling of the original image over the neighborhood, and the techniques employed to derive solutions. In [18], the estimated wavelet coefficient at index l (position, scale), $\hat{w}_l$, is obtained by

$$\hat{w}_l = w_l \; P(x_l|M) \approx w_l \; P(M|x_l) \; P(x_l),$$

where w_l is the observed wavelet coefficient, x_l is the hidden label, $P(x_l|M)$ is the probability of the label l as an edge, and M is the measurement derived by inter-scale Lipschitz exponent estimation [19], [20]. The prior probability $P(x_l)$ at the scale of l is modeled by a 2×2 MRF, where the potential functions between two neighboring wavelet coefficients on hand-chosen cliques are defined as $x_i x_j$, with each variable taking the value 1 or -1. The approach is extended in [21] by incorporating robust inter-scale estimators M and $P(x_l|M)$ and a generalized anisotropic MRF prior $P(x_l)$ on each scale.

Simoncelli examined the empirical statistical properties of images in adjacent scales and presented an inter-scale probability model for the wavelet coefficients in two adjacent scales [22]. The joint statistical model assumes that the density of an estimated wavelet coefficient is conditionally Gaussian, where the variance is a linear combination of the squared coefficients in a local neighborhood. The neighborhood consists of coefficients at other orientations and adjacent scales, as well as adjacent spatial locations. For example, Simoncelli's model uses a neighborhood comprised of the 12 nearest spatial neighbors in the same subband, the 5 nearest coefficients in subbands at other orientations on the same scale, the 9 nearest coefficients in the adjacent subband of the coarser scale, and some coefficients in other subbands.

The BLS-GSM algorithm [23] models the distribution of a vector of wavelet coefficients in a 3×3 region, together with the coefficient at the center location and the same orientation at the next coarser scale, as a Gaussian Scale Mixture (GSM). Then, the Bayesian least square method is used to estimate the wavelet coefficient at the center of the neighborhood system. In [24], a mixture of Gaussian Scale Mixtures (MGSM) is proposed to make the GSM model adaptive to the image content. Its denoising performance is almost as good as that of the BM3D algorithm. In [25], a dimension reduction technique is applied to the MGSM to reduce the computational cost and avoid the curse of dimensionality problem in learning the high number of free parameters of the model.

The MRF is modeled on an undirected graph. In this paper, we propose an approach that uses a hidden directed graph to model the prior probability of an image. Specifically, the graph is a Bayesian network with a multi-layer network structure constructed from the wavelet coefficients of an image. Our approach has two advantages over existing approaches. First, the MAP solution can be derived by using the standard BP algorithm [26], [27]. BP inference passes messages forward from coefficients at coarser scales to finer scales and backward from finer scales to coarser scales. Forward message passing tends to smooth fluctuations in fine scales, while backward message passing tends to retain the fine details of an image. Thus, our method is more capable of retaining the fine structure of an image than existing approaches. The second advantage is that the hidden structure is derived by a data-adaptive process. In addition, the prior probabilities over inter-scale edges and intra-scale edges are modeled in a similar way to those in [22] and [28] respectively. We also analyze the complexity of estimating the solution of a Bayesian network and show that BP inference can derive the MAP solution efficiently provided that the Bayesian network is a spanning tree. To evaluate our approach, we compare its performance with that of other approaches, including BM3D, and demonstrate that it yields a better peak-signal-to-noise ratio (PSNR) as well as better perceptual quality on the textured areas of an image.

The remainder of this paper is organized as follows. In Section II, we explain the rationale behind the proposed data-adaptive graph modeling of an image. We also analyze the

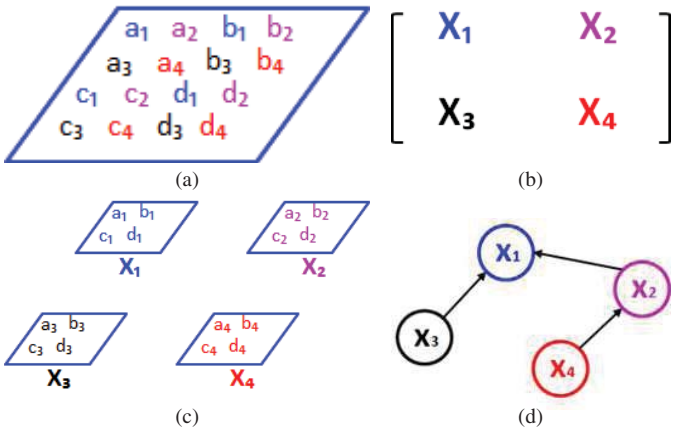


Fig. 1. Constructing a hidden directed graph from a wavelet subband. (a) Subband of 4 × 4 coefficients. (b) Wavelet patch is a matrix of 2 × 2 random variables, X_1, X_2, X_3, and X_4. (c) Subband is divided into four rectangles, each containing 2 × 2 coefficients. The coefficients at the same location in each rectangle are assigned to the node at the same location in the wavelet patch, and the coefficients of the same color are grouped as the observations of a node. (d) Directed edges are constructed to form a spanning tree.

computational cost of using BP inference to estimate the MAP solution for the proposed approach. In Section III, we present the method used to construct and model wavelet Bayesian networks; and in Section IV, we describe how the networks are used for denoising. In Section V, we discuss the proposed denoising algorithm and compare its performance with that of other approaches. Section VI contains some concluding remarks.

II. Data-Adaptive Hidden Network Approach

In this section, we describe the proposed framework for constructing a data-adaptive graph structure and formulating the prior probability of the original image in the wavelet domain. We also explain how the graph structure is used with the BP algorithm to derive the MAP solution.

We assume that the wavelet coefficients of the original image represent a realization of a hidden graph. To construct the graph, we first create a matrix of random variables (a wavelet patch) for each subband (indexed by orientation and scale), and assign the subband coefficients as the observed data of the random variables. Then, a subgraph (network) is constructed from the wavelet patch as follows. First, we associate each random variable with a node; hence, the coefficients assigned to the random variable can be regarded as node's observed data. Second, the arcs (directed edges) in the subgraph are derived according to a data dependence measurement between the observed data in any pair of nodes. In this way, we can construct a one-layer network from the wavelet subband and associate the subband's coefficients with the nodes in the network. The procedure is shown in Fig. 1. Finally, the one-layer networks of adjacent subbands can be linked by inter-scale arcs, from coarser scales to finer scales, to form a multi-layer network structure, as shown in Fig. 2(a)–(c).

Two problems may arise during the above construction procedure: 1) the coefficient and wavelet patch association problem, which involves associating subband coefficients with a wavelet patch; and 2) the graph selection problem, i.e.,

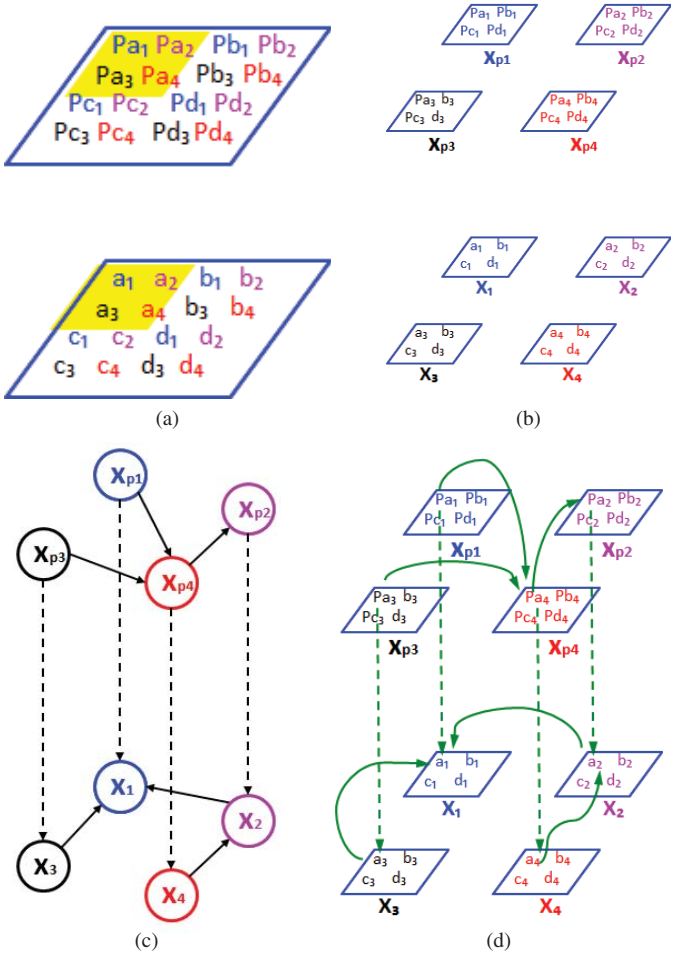


Fig. 2. Constructing a multilayer hidden network. (a) Two subbands, with the coarser subband on top. (b) Procedure creates two wavelet patches, each of which is associated with a subgraph. Subband coefficients are assigned to the nodes, as specified in Fig. 1(c). (c) Nodes in the two-layer network are linked by intra-scale (solid) arcs and inter-scale (dashed) arcs. (d) To derive the prior probability, the hidden coefficients are organized into groups a, b, c, and d. In group a, the coefficients are $\{Pa_i, a_i | i = 1, 2, 3, 4\}$ and they are highlighted in the local blocks of the subbands in (a). Coefficients in the other groups can be derived in a similar manner.

determining the type of graph to construct. To solve the first problem, we propose the following heuristic procedure. Assume that the wavelet patch is a matrix of $m \times m$ random variables. Let the size of a subband be $N \times N$ and let m divide N. We partition the subband into $(\frac{N}{m})^2$ rectangular blocks, each of which contains $m \times m$ coefficients. Then, the coefficient at location (i, j) in each block is assigned as a realization of the random variable at location (i, j) in the wavelet patch. Thus, each random variable has $(\frac{N}{m})^2$ sampled observations.

For the second problem, we analyze the computational cost of a graph structure for which the MAP solution can be derived efficiently by the standard BP algorithm. The standard implementation of the message passing algorithm in BP on $m \times m$ cliques runs in $\mathcal{O}(N^2 k^{m \times m} T)$, where N^2 is the number of coefficients in a subband, k is the number of labels for each coefficient, and T is the number of iterations. Basically, computing each message takes $\mathcal{O}(k^{m \times m})$ time and there are $\mathcal{O}(N^2)$ messages per iteration. The computational

cost of the conventional BP algorithm can be reduced if the algorithmic techniques in [29]–[31] are used for Bayesian inference. If parallelism is also exploited as described in [32], BP inference can achieve a near linear parallel scaling performance. We use a graph $G(V, E)$ to represent the (one-layer) subgraph structure, where V is the node set and E is the arc set. If G is a loopy Bayesian network, the BP algorithm sometimes yields surprisingly good approximate results; however, sometimes, it fails to produce any results, even after a large number of iterations. BP can derive an exact MAP solution in two iterations if G is a directed-acyclic-graph (DAG), but the inference cost depends on the structure of the DAG. A DAG that incurs a high inference cost can be constructed as follows: let the nodes in G be indexed from 1 to m^2. For each node j, we link an arc from any node of a smaller index to it. Thus, the message passing at node j involves k^{j-1} operations, where k is the label of a wavelet coefficient. The total cost of the operation for all nodes is

$$\sum_{j=1}^{m\times m-1} k^{j-1} = \frac{k^{m\times m-1}-1}{k-1}. \tag{1}$$

For typical values of k and m, which are 512 and 4 respectively, BP inference of the DAG is too high to be of practical use.

If we assume that G is a spanning tree structure, we can show that the average computation time for BP inference is $\mathcal{O}(m^2k)$. Let us assume that any two nodes have an equal chance of being linked by an arc in a spanning tree. As the tree contains m^2-1 edges, each node has 1 in-degree at most; and, in total, all the nodes have m^2-1 out-degrees. Because all the nodes are treated equally, the average out-degree of each node in a spanning tree is $\frac{m^2-1}{m^2} \leq 1$. BP inference in a spanning tree is comprised of two phases: forward message passing and backward message passing, which use the in-degree arcs and out-degree arcs respectively to pass messages to a node. Because the average in-degree and out-degree arc of a node is at most 1, the average number of messaging passing operations on a node is $\mathcal{O}(k)$. There are m^2 nodes, so the average number of message passing operations is $\mathcal{O}(m^2k)$ for BP inference in spanning trees. The analysis shows that the cost of BP inference on a spanning tree can be significantly less than that in a general DAG. However, the power of a spanning tree to express the structure of a wavelet patch is limited. For a subgraph of $m \times m$ nodes, there are $m^2 \times m^2$ pairwise relations, but a spanning tree only uses m^2-1 of them. Thus, a spanning tree cannot fully capture a complicated structure that contains more than m^2-1 highly correlated pairwise relations and a smooth region in which all the nodes are highly correlated to one another.

The joint probability of a spanning tree $G = (V, E)$ can be formulated by the dependency structure in G as follows. Let $f(v_i|u_i)$ be the probability function associated with arc $u_i \rightarrow v_i$, where $u_i, v_j \in V$, and let u be the root of the tree with probability $f(u)$. Then, the joint probability of G is

$$\mathbf{f}(G) = \prod_{u_i \rightarrow v_i \in E} f(v_i|u_i) f(u). \tag{2}$$

Note that the intra-scale clustering property of wavelet coefficients indicates that the neighboring coefficients in a subband are positively correlated. Thus, the pairwise joint probability can be modeled as a measurement of $|u_i - v_i|$, resembling the marginal statistics of the gradient of neighboring nodes with values u_i and v_i.

We can stack the subgraphs of two adjacent scales to form a two-layer graph. Let $G_c = (V_c, E_c)$ be the graph corresponding to a subband at the coarser scale of G. Based on the inter-scale persistence property, inter-scale arcs can be constructed between V_c and V, denoted as A, by combining the parent node and child node (nodes at the same location) in G_c and G respectively. Let $p(v_k)$ in V_c be the parent node of v_k in V; and let u_c and u be the root of the tree in G_c and G respectively. Because of the node dependency in the graphs, the joint probability of the resulting directed graph can be formulated as follows:

$$\mathbf{f}(G_c, G) = \Big[f_c(u_c) \prod_{u_i \rightarrow u_j \in E_c} f_c(u_j \mid u_i)\Big] \times \Big[f(u|p(u)) \prod_{v_k \rightarrow v_l \in E} f(v_l \mid v_k, p(v_l))\Big] \tag{3}$$

where $p(u) \rightarrow u \in A$, and $p(v_l) \rightarrow v_l \in A$. Equation (3) gives the joint probability of all the nodes in the graph shown in Fig. 2(c). The prior probability of the hidden coefficients in the graph is derived as follows. We take one coefficient from each node to form a data group; for example, Fig. 2(d) has four data groups. The joint probability of each data group can be written in the form of Equation (3) by replacing the node variables with the hidden coefficients. Finally, it is assumed that the joint probability of all the data groups is the product of that of each data group.

After constructing the hidden network, we create a layer of observation nodes for each noisy wavelet subband. Next, we assign the noisy coefficients to the observation nodes in the same way as the ideal subband coefficients are assigned to the hidden nodes, as shown in Fig. 3(a) and (b). Then, arcs are created to link the observation nodes to the corresponding hidden nodes, as shown in Fig. 3(c). BP inference can now be applied to derive the MAP solution for the denoising problem. Recall that each node in a wavelet patch has $(\frac{N}{m})^2$ realizations. If we take one wavelet coefficient from each node and use the BP algorithm to estimate the solution, the denoising process makes $(\frac{N}{m})^2$ BP inferences. Note that, as shown in Fig. 3(d), all the wavelet coefficients located in the same rectangular blocks in various subbands are estimated simultaneously by one BP inference.

We calculate the computational cost of image denoising under the proposed model as follows. Because each node has one in-degree and one out-degree inter-scale edge at most, the number of operations that BP inference performs for the multi-layer graph is J times the number performed for one wavelet patch. For a spanning tree, each inference involves an average of $\mathcal{O}(m^2k)$ operations and there are $(\frac{N}{m})^2$ BP; thus, the proposed algorithm performs an average of $\mathcal{O}(N^2k)$ operations for one wavelet patch. As a result, the average BP inference cost of the J layers network is $\mathcal{O}(JN^2k)$.

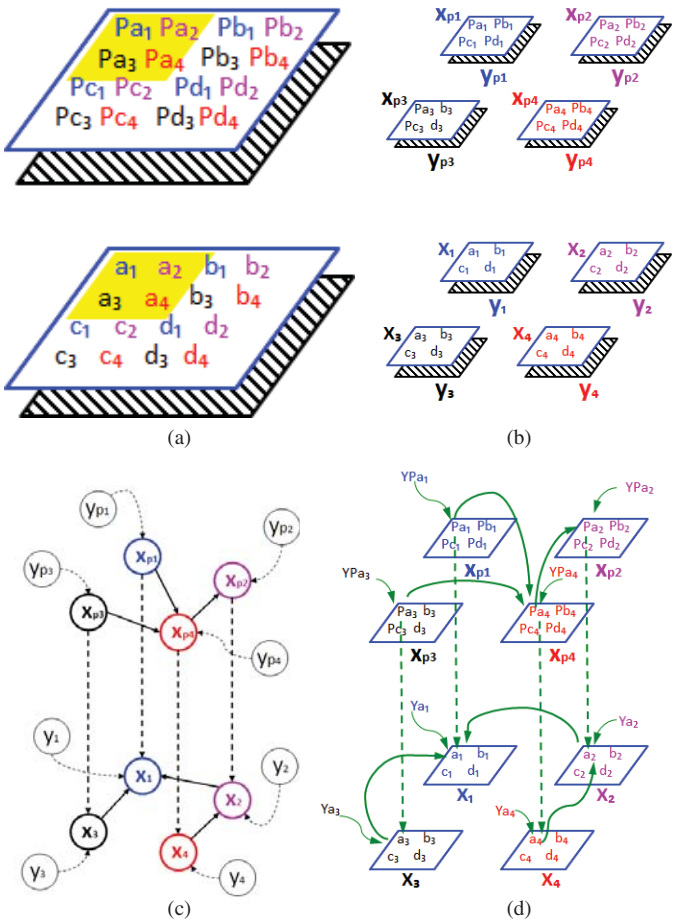


Fig. 3. (a) Top layer and the bottom (dashed) layer of each group are composed of hidden subband coefficients and coefficients observed at the corresponding subband, respectively. (b) Coefficients in observed (dashed) layers are organized in the same way as the coefficients in the hidden layers. (c) Observation nodes $\{y_{pi}, y_i\}$ are created and linked to hidden nodes $\{x_{pi}, x_i\}$. (d) For denoising proposes, data in the network is organized into four groups and BP inference is applied to each group. For example, $\{yp_{ai}, pa_i, ya_i, a_i | i = 1, 2, 3, 4\}$ is a group. The other three data groups can be derived in a similar manner.

We have explained the basic approach used to construct a spanning tree from a wavelet patch and stack the spanning trees to form a multi-layer network. In addition, we have shown that if the subgraph structure is a spanning tree, the exact MAP solution can be derived efficiently by BP inference. Before describing the construction in detail, we remark that the wavelet coefficients at the same location and scale, but in different orientations, are correlated. However, they are not necessarily positively correlated, so their potential functions are not suitable for modeling as a function of their absolute difference. Therefore, we construct a wavelet Bayesian network for the wavelet subbands in each orientation.

III. Constructing Wavelet Bayesian Networks

A Bayesian network, denoted as $B = (V, E, P)$, comprises a set of random variables and their conditional dependencies represented by a directed acyclic graph in which the nodes represent the elements in V. Each edge element in E takes the form of a directed arc $x \rightarrow y$, where x and $y \in V$. The likelihood $p(y \mid x) \in P$ of an edge $x \rightarrow y \in E$ is the conditional probability of observing y given that x exists.

We call the Bayesian networks constructed in wavelet domains wavelet Bayesian networks (WBNs). Our primary objective is to construct a WBN from the undecimated discrete wavelet transform (DWT) of a single image. Initially, wavelet decomposition of an image F yields three images of wavelet coefficients with horizontal, vertical, and diagonal orientations respectively, and one approximate image of F. Then, at the next scale, the approximated image is further decomposed to obtain three images of the wavelet coefficients and one coarser approximate image of F. Let $\mathcal{W}_j^h F(u, v)$, $\mathcal{W}_j^v F(u, v)$, and $\mathcal{W}_j^d F(u, v)$ denote, respectively, the horizontal, vertical, and diagonal images of the wavelet coefficients at scale 2^j; and let $A_j F(u, v)$ represent the approximated image at the same scale. If the undecimated DWT is decomposed J times, we will have wavelet coefficients $\mathcal{W}_j^h F$, $\mathcal{W}_j^v F$, and $\mathcal{W}_j^d F$ with $j = 1, \ldots, J$ and the coarsest approximate image $\mathcal{A}_J F$.

To construct a WBN, we first group subbands with the same orientation together to obtain a horizontal-group, a vertical-group, and a diagonal-group of wavelet coefficients. Then, we construct a Bayesian network B for each group. Let $B^h = (V^h, E^h, P^h)$, $B^v = (V^v, E^v, P^v)$, and $B^h = (V^d, E^d, P^d)$ denote the Bayesian networks constructed from the horizontal-group, vertical-group, and diagonal-group of wavelet coefficients respectively. The WBN $B = (V, E, P)$ is derived from B^h, B^v, and B^d by

$$V = V^h \cup V^v \cup V^d \tag{4}$$

$$E = E^h \cup E^v \cup E^d \text{ and} \tag{5}$$

$$P = P^h \cup P^v \cup P^d. \tag{6}$$

Next, we explain how to construct the Bayesian network $B^u(V^u, E^u, P^u)$ that corresponds to the u-orientation, where $u \in \{h, v, d\}$.

A. Vertex Set V^u

Let the size of the input image F be $N \times N$. If J wavelet decompositions are applied to F, there will be J subbands of size $N \times N$ in each orientation. Let a wavelet patch (matrix of random variable) be of size $m \times m$. For each subband, a graph of m^2 variable nodes are formed. We then associate each random variable in the wavelet patch to a variable node in the graph. Without loss of generality, we assume that m divides N. Each subband can then be partitioned into $(\frac{N}{m})^2$ blocks, each of size $m \times m$. Then, $(\frac{N}{m})^2$ wavelet coefficients sampled from the subband are assigned to each variable node. Let $x_j^h(i, k)$, with $j = 1, \ldots, J$ and $i, k = 0, \ldots, m-1$, denote the (i, k) variable node in the j-th subband. In our construction, the $(\frac{N}{m})^2$ wavelet coefficients assigned to node $x_j^u(i, k)$ are sampled from $\mathcal{W}_j^u F(i + mp, k + mq)$ with $p, q = 0, \ldots, \frac{N}{m} - 1$. If we represent each node as a vertex in the Bayesian network B^u, then the vertex set of B^u will be

$$V^u = \{x_j^u(i, k) | i, k = 0, \cdots, m-1; \quad j = 1, \cdots, J\} \tag{7}$$

and the $(\frac{N}{m})^2$ wavelet coefficients can be regarded as sampled from some (unknown) distribution of a random variable. Figs. 1(a), (b), and (c) show the procedure used to construct the vertex set for a subband of 4×4 coefficients.

B. Edge Set E^u

The arcs (directed edges) in B^u can be divided into two disjoint sets, E_o^u and E_i^u, where E_o^u comprises the (inter-scale) edges incident to vertices at different scales, and E_i^u comprises the (intra-scale) edges incident to vertices at the same scale. The persistence property of the wavelet transforms indicates that large/small values of wavelet coefficients tend to occur at the same spatial locations in subbands at adjacent scales. The property can be used to construct arcs in E_o^u by linking a vertex at the coarser scale $j+1$ to the vertex of the same index at the finer scale j; that is,

$$E_o^u = \{x_{j+1}^u(i,k) \rightarrow x_j^u(i,k) |\ i,\ k = 0, \ldots, m-1 \text{ and } j = 1, \cdots, J-1\}. \tag{8}$$

The edges in E_i^u represent the connections between vertices at the same scale and orientation. Constructing the edges corresponds to deriving the Bayesian network on the nodes $x_j^u(i,k)$ that best represent the joint probability of the nodes at the same scale j and orientation u. However, as discussed in Section II, BP inference is computationally intractable if the Bayesian network is a general graph. Thus, we limit the solution space to spanning trees so that we can derive an efficient solution by using the *maximal weighted spanning tree* (MWST) algorithm [33]–[36]. A maximal weighted spanning tree is a spanning tree whose weight is greater than or equal to the total weight of every other spanning tree. The optimum weighted spanning tree can be derived by minimizing the relative entropy (Kullback-Leibler distance) $D(p||q)$ between the probability functions p and q.

In the following, we show how the spanning tree that minimizes $D(p||q)$ is equivalent to the tree that maximizes the weighted summation of conditional mutual information on all the edges of the tree. Let $\mathbf{x}$ be the vector of variables $x_1, \ldots, x_n$; let $p(i)$ and $b(i)$ denote the indices of the parent nodes and the sibling nodes of x_i respectively; and let q be the induced probability of the spanning tree. Then, we have

$$q(\mathbf{x}) = \prod_{j=1}^{n} p(x_j \mid x_{p(j)}, x_{b(j)}). \tag{9}$$

Note that nodes at the coarsest scale do not have parent nodes. To find the optimal spanning tree, we minimize the relative entropy between $q(\mathbf{x})$ and the joint probability $p(\mathbf{x})$ as follows:

$$\begin{aligned} D(p||q) &= \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} \\ &= \sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x}) - \sum_{\mathbf{x}} p(\mathbf{x}) \log q(\mathbf{x}). \end{aligned} \tag{10}$$

Since minimizing $D(p||q)$ is equivalent to maximizing $\sum_{\mathbf{x}} p(\mathbf{x}) \log q(\mathbf{x})$, we can derive the following:

$$\begin{aligned} &\sum_{\mathbf{x}} p(\mathbf{x}) \log q(\mathbf{x}) \\ &= \sum_{\mathbf{x}} p(\mathbf{x}) \log \prod_j p(x_j \mid x_{p(j)}, x_{b(j)}) \qquad (11) \\ &= \sum_{\mathbf{x}} \sum_{j, b(j)=\emptyset} p(\mathbf{x}) \log \frac{p(x_j, x_{p(j)})}{p(x_{p(j)})} \\ &\quad + \sum_{\mathbf{x}} \sum_{j, b(j)\neq\emptyset} p(\mathbf{x}) \log \frac{p(x_j, x_{p(j)}, x_{b(j)})}{p(x_{p(j)}, x_{b(j)})} \\ &= \sum_{\mathbf{x}, j} p(\mathbf{x}) \log \frac{p(x_j, x_{p(j)})}{p(x_{p(j)})} \\ &\quad + \sum_{\mathbf{x}, j, jb\neq\emptyset} p(\mathbf{x}) \left[\log \frac{p(x_j, x_{p(j)}, x_{b(j)})}{p(x_{p(j)}, x_{b(j)})} - \log \frac{p(x_j, x_{p(j)})}{p(x_{p(j)})} \right]. \end{aligned}$$

The second term in the last line of the above equation can be re-written as follows:

$$\begin{aligned} &\sum_{\mathbf{x}} \sum_{j, b(j)\neq\emptyset} p(\mathbf{x}) \log \frac{p(x_j, x_{p(j)}, x_{b(j)}) \cdot p(x_{p(j)})}{p(x_{p(j)}, x_{b(j)}) \cdot p(x_j, x_{p(j)})} \\ &\quad = \sum_{\mathbf{x}} \sum_{j, b(j)\neq\emptyset} p(\mathbf{x}) \log \frac{p(x_j, x_{b(j)} \mid x_{p(j)})}{p(x_{b(j)} \mid x_{p(j)}) \cdot p(x_j \mid x_{p(j)})} \\ &\quad = \sum_{j, b(j)\neq\emptyset} \left[\sum_{x_j, x_{b(j)}, x_{p(j)}} p(x_j, x_{b(j)}, x_{p(j)}) \right. \\ &\qquad \left. \log \frac{p(x_j, x_{b(j)} \mid x_{p(j)})}{p(x_{b(j)} \mid x_{jp}) \cdot p(x_j \mid x_{p(j)})} \right] \\ &\quad = \sum_{j, b(j)\neq\emptyset} \sum_{x_{p(j)}} p(x_{p(j)}) \sum_{x_j, x_{b(j)}} p(x_j, x_{b(j)} \mid x_{p(j)}) \\ &\qquad \log \frac{p(x_j, x_{b(j)} \mid x_{p(j)})}{p(x_{b(j)} \mid x_{p(j)}) \cdot p(x_j \mid x_{p(j)})} \\ &\quad = \sum_{j, b(j)\neq\emptyset} \sum_{x_{p(j)}} p(x_{p(j)}) \cdot I(x_j, x_{b(j)} \mid x_{p(j)}). \end{aligned} \tag{12}$$

From the results of Equations (11) and (12), we conclude that minimizing $D(p||q)$ over q is equivalent to maximizing the weighted summation of conditional mutual information $\sum_{j, b(j)\neq\emptyset} \sum_{p(j)} p(x_{p(j)}) \cdot I(x_j, x_{b(j)} \mid x_{p(j)})$, where the weight of the arc $x_i \rightarrow x_j$ is defined as

$$p(x_{p(i)}) I(x_i, x_j | x_{p(i)}). \tag{13}$$

To find a tree with the maximum total arc weight of m^2 nodes, we use Kruskal's algorithm [41]. The steps are as follows.

The complexity of the algorithm is $\mathcal{O}(m^2 \times m^2 \log_2(m^2 \times m^2))$. To calculate the mutual information $I(x_i, x_j)$ of variables x_i and x_j, we need to estimate the probability distribution functions of $p(x_i, x_j)$, $p(x_i)$, and $p_j(x_j)$. Recall that each variable comprises $(\frac{N}{m})^2$ sampled wavelet coefficients. Let $\tilde{n}(x_i = u, x_j = v)$ denote the number of samples such that x_i and x_j assume the values of u and v respectively. The sample joint frequencies $f(x_i = u, x_j = v)$ are computed as follows:

$$f(x_i = u, x_j = v) = \frac{\tilde{n}(x_i = u, x_j = v)}{\sum_a \sum_b \tilde{n}(x_i = a, x_j = b)}; \tag{14}$$

Algorithm 1 Kruskal's Algorithm

1) Calculate the $m^2 \times m^2$ weights of the arcs between any two nodes. Note that the weight of arc $x \to y$ may be different from that of arc $y \to x$.
2) Sort the weights of the $m^2 \times m^2$ arcs and compile a list in non-increasing order so that the weight w_i is not less than the weight w_j with $i < j$.
3) Assume that the initial spanning tree is empty and add the arc of weight w_1 to the empty tree.
4) Do until the tree is a spanning tree:
   a) Let w_i be the next weight on the list.
   b) Check if adding the arc of weight w_i to the tree creates a cycle.
5) End

and the sample probabilities are calculated as follows:

$$f(x_i = u) = \sum_b f(x_i = u, x_j = b)$$
$$f(x_j = v) = \sum_a f(x_i = a, x_j = v). \tag{15}$$

It is well-known that the joint frequencies $f(x_i = u, x_j = v)$, $f(x_i = u)$, and $f(x_j = v)$ are maximum-likelihood estimators for the probabilities $p(x_i = u, x_j = v)$, $p(x_i = u)$, and $p(x_j = v)$ respectively. Therefore, the sample mutual information can be computed as

$$\hat{I}(x_i, x_j) = \sum_{u,v} f(x_i = u, x_j = v) \log \frac{f(x_i = u, x_j = v)}{f(x_i = u) f(x_j = v)}. \tag{16}$$

Then, we use $\hat{I}(x_i, x_j)$ instead of $I(x_i, x_j)$ when calculating the weights in the above algorithm. For each scale $j \in \{1, \ldots, J\}$, we execute Kruskal's algorithm to obtain a spanning tree. The edge set E_i^u contains all the edges in all the spanning trees.

Figure 1(d) shows an example of a network with intra-scale edges derived by the MWST algorithm from the four nodes in Fig. 1(c); and Fig. 2(c) shows an example of a multilayer network where the inter-scale and intra-scale edges are constructed from the nodes in Fig. 2(b). The WBN B in Fig. 4 is comprised of three oriented Bayesian networks, B^h, B^v, and B^d.

C. Probability Model P^u

There are two types of arcs in a Bayesian network B^u: 1) the inter-scale parent-child arc, which connects a node with its coarser-scale parent; and 2) the intra-scale sibling arc, which connects two nodes of the same scale. To obtain the probability inference, we need to model the probability function on each arc.

Simoncelli [22] exploited the persistence property of wavelet transforms and proposed a joint statistical model of a "child" coefficient conditioned on the coarse-scale "parent" coefficients at the same spatial locations in all orientations.

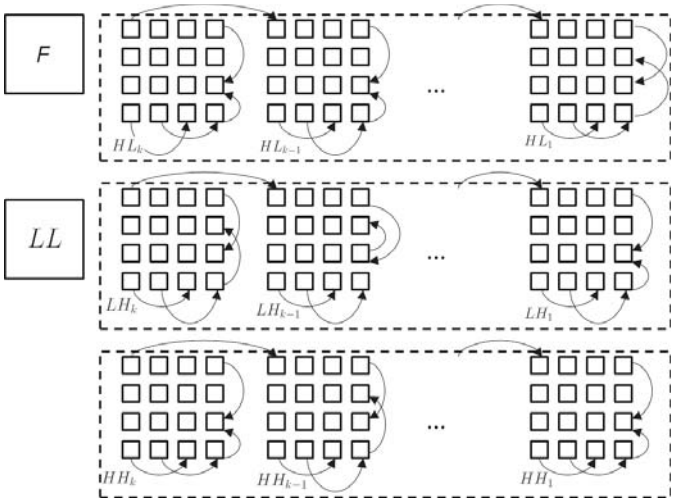


Fig. 4. WBN B has three components, B^h, B^v, and B^d, derived from the horizontal, vertical, and diagonal orientation wavelet coefficients, respectively, as shown from top to bottom. F and LL are, respectively, the input and the coarsest approximation of F. k is the scale index.

Let $\{x_{p_k}\}$ comprise the parent coefficients of the child coefficient x. Then, the probability function of x conditioned on $\{x_{p_k}\}$ is defined as the following Markov model:

$$p(x|\{x_{p_k}\}) = \mathcal{N}(0; \sum_k 2w_k x_{p_k}^2 + \alpha^2) \tag{17}$$

which is a conditional Gaussian density function whose variance is a linear function of the squared coefficients in $\{x_{p_k}\}$ with weights $\{w_k\}$, and α is the bias of the variance. We simplify Simoncelli's joint probability model to represent the pdf of a parent-child arc in E_o^u. Specifically, we remove the dependency of the coarse-scale parents' coefficients in all orientations except that of the child coefficient and set the bias of variance α to zero. Thus, there is only one parent coefficient x_p for x. Then, the joint probability of the parent-child arc in E_o^u is modeled as

$$f_o^u(x|x_p) = \mathcal{N}(0; 2wx_p^2) \tag{18}$$

where w is a chosen parameter.

Other researchers have observed that large/small wavelet coefficients of the same sign tend to occur near each other in a subband. The observations are summarized as the clustering property of the wavelet transform. The property means that, in a subband, there is a strong positive correlation between wavelet coefficients that are in close proximity and dependent on each other. The ROF approach [28] exploits the clustering property between neighboring pixels by using TV norms, which are essentially L_1 norms of the derivative of the values of two neighboring coefficients, to estimate the discontinuities in an image. Equivalently, the joint probability function between two coefficients in close proximity is modeled as the Laplacian distribution of the absolute of the difference of two wavelet coefficients [42]. In the construction of intra-scale edges, Kruskal's algorithm selects the arc $x \to z \in E_i^u$, where the mutual information between x and z is high; that is, x and z are highly correlated. Thus, we utilize a similar concept to

model the probability of z conditioned on x as the following Laplacian distribution function:

$$f_i^u(z|x) \propto \lambda \exp(-\lambda|x-z|) \tag{19}$$

where λ is the scale parameter of the Laplacian distribution, and $\lambda|x-z|$ is the potential function of the local 1×1 clique.

IV. Wavelet Bayesian Networks for Denoising

In this section, we consider using the wavelet Bayesian network to model the prior probability of the original image for the image denoising problem, which involves removing white and homogenous Gaussian noise with zero mean and known variance from an image.

To infer the probability for denoising, we associate each variable node x in Bayesian network B with an observation node y and create the arc $y \to x$. The probability function of x conditioned on the observed value of y is modeled as

$$f_n(x|y) \propto \frac{1}{\sigma_n} \exp(-\frac{(x-y)^2}{2\sigma_n^2\rho}) \tag{20}$$

where σ_n^2 is the variance of the zero mean Gaussian white noise and ρ depends on the scale and wavelets.

Recall that the variable nodes in the Bayesian network B are represented by $\{x_j^u(i,k)|u \in \{h,v,d\};\ j=1,\ldots,J;\ i,k=0,\ldots,m-1\}$ (Equation (7)). Let $y_j^u(i,k)$ denote the observation node corresponding to $x_j^u(i,k)$; and let Y, E_n, and P_n denote the collections of $\{y_j^u(i,k)\}$, the arcs $\{y_j^u(i,k) \to x_j^u(i,k)\}$, and the probability functions $\{f_n(x_j^u(i,k)|y_j^u(i,k))\}$ respectively. The WBN B_n for image denoising is constructed and represented as $B_n = (V \cup Y, E \cup E_n, P \cup P_n)$.

Let a noisy image $Z = F + N$, where F is the original image and N is zero-mean white Gaussian noise. As shown by the simple example in Fig. 3(d), each wavelet coefficient of Z is assigned to one observation node in B_n. That is, the coefficient $\mathcal{W}_j^u Z(i+mp, k+mq)$ is assigned to observation node $y_j^u(i,k)$, where $p,q = 0,\cdots\frac{N}{m}-1$; thus, each observation node has $(\frac{N}{m})^2$ observation values and realizations. If we take one wavelet coefficient from each observation node, we can obtain $(\frac{N}{m})^2$ realizations of B_n, denoted as (p,q) with $p,q=0,\cdots\frac{N}{m}-1$. Note that, similar to the example shown in Fig. 3(d), the wavelet coefficients assigned to the (p,q)-th realization are taken locally from the (i,k)-blocks of all subbands at scale 2^j and orientation u.

We use the message passing algorithm to obtain the estimated wavelet coefficients of each realization. First, we convert WBN B_n to a factor graph F_n, and then use the max-product algorithm to derive the estimated wavelet coefficients. The conversion of B_n to F_n and the max-product message passing schemes are standard techniques. For completeness, we provide them in Appendix A. The last step of the max product algorithm calculates the marginal probability of each variable node V in F_n. Let $\mathcal{N}(x)$ represent the neighboring factor nodes of variable node x in F_n. In addition, let x_p and x_c denote, respectively, the parent variable node and child variable node of x in B_n; and let $\{x_j\}$ denote the sibling variable nodes of x in B_n. The value of $\hat{x}$ can be estimated based on whether x has a child node.

Case 1: x has a child node x_c.

$$\begin{aligned}\hat{x} &= \arg\max_x \prod_{D\in\mathcal{N}(x)} \mu_{D\to x}(x)\\ &= \arg\max_x \left(\frac{1}{x}\exp\big[-J_c(x)\big]\right)\end{aligned} \tag{21}$$

where

$$J_c(x) = \left[\frac{(x-y)^2}{2\sigma_y^2} + \frac{x^2}{2\omega {x_p}^2} + \frac{{x_c}^2}{2\omega x^2} + \lambda\sum_j |x-x_j| + \Psi\right]. \tag{22}$$

In Equation (22), $\Psi = \Psi(y, x_p, x_c, \{x_j\})$ is independent of x, and σ_y^2 is the variance of the wavelet coefficients associated with observation node y. The variance σ_y^2 can be written as $\sigma_n^2\rho$, where ρ depends on the scale and the wavelets. In Appendix B, we show that $\rho = 1$ if the wavelets are orthogonal. Let

$$K(x) = \frac{1}{x}\exp\big[-J_c(x)\big]; \tag{23}$$

then

$$\frac{d}{dx}\log K(x) = \frac{d}{dx}\big[\log\frac{1}{x} - J_c(x)\big] = -\frac{1}{x} - J_c'(x). \tag{24}$$

According to Equation (22)

$$J_c'(x) = \frac{x-y}{\sigma_y^2} + \frac{x}{\omega {x_p}^2} - \frac{{x_c}^2}{\omega x^3} + \lambda\sum_j \text{sign}(x-x_j) \tag{25}$$

where $\text{sign}(x)$ returns the sign of x if $x \neq 0$; otherwise, it returns 0. Hence

$$\begin{aligned}\frac{1}{x} + J_c'(x) = \frac{x-y}{\sigma_y^2} + \frac{1}{\omega x}\big[\frac{x^2}{{x_p}^2} - \frac{{x_c}^2}{x^2} + \omega\big]\\ +\lambda\sum_j \text{sign}(x-x_j).\end{aligned} \tag{26}$$

We use the following algorithm to estimate the root $\hat{x}$ of $K'(x) = 0$ iteratively. Substituting x_{t-1} (the estimated value after the $(t-1)$-th iteration) into $\frac{x^2}{{x_p}^2} - \frac{{x_c}^2}{x^2} + \omega$ and $\lambda\sum_j \text{sign}(x-x_j)$ in Equation (26), we can re-write Equation (26) as a quadratic equation of x with two roots at

$$\begin{aligned}&\left[y - \lambda\sigma_y^2\sum_j \text{sign}(x_{t-1}-x_j)\right]\\ &\quad\pm sqrt\Bigg(\Big[y - \lambda\sigma_y^2\sum_j \text{sign}(x_{t-1}-x_j)\Big]^2\\ &\quad - \frac{4\sigma_y^2}{\omega}\left[\frac{{x_{t-1}}^2}{{x_p}^2} - \frac{{x_c}^2}{{x_{t-1}}^2} + \omega\right]\Bigg).\end{aligned} \tag{27}$$

The root that is closer to x_{t-1} is chosen as x_t.

Case 2: x does not have a child node (x is a node at the finest wavelet scale). We can set $x_c = 0$ in Equation (22) and obtain

$$J(x) = \frac{(x-y)^2}{2\sigma_y^2} + \frac{x^2}{2\omega {x_p}^2} + \lambda\sum_j |x-x_j|; \tag{28}$$

and then

$$J'(x) = \frac{(x-y)}{\sigma_y^2} + \frac{x}{\omega {x_p}^2} + \lambda\sum_j sign(x-x_j). \tag{29}$$

Algorithm 2 WBN Denoising Algorithm

1) Wavelet representation: calculate the undecimated DWT of an $N \times N$ noisy image Z to obtain the horizontal, vertical, and diagonal subbands $\{\mathcal{W}_j^u Z | j = 1, \ldots, J\}$, $\{\mathcal{W}_j^v Z | j = 1, \ldots, J\}$, and $\{\mathcal{W}_j^d Z | j = 1, \ldots, J\}$, respectively, as well as the coarsest approximate image $\mathcal{A}_J Z$.
2) Create a vertex set (V^u): let the parameter m divide N. For each subband $\mathcal{W}_j^u \hat{Z}$, create m^2 variable nodes $\{x_j^u(i,k) |\ i,k = 0, \ldots m-1\}$; then assign $(\frac{N}{m})^2$ wavelet coefficients $\{\mathcal{W}_j^u \hat{Z}(i+mp, k+mq) | p,q = 0, \ldots, \frac{N}{m}-1\}$ as samples for variable node $x_j^u(i,k)$.
3) Create sibling edges (E_i^u) for nodes at the same scale and orientation: derive the empirical probability of each variable from the frequency counts of $(\frac{N}{m})^2$ wavelet coefficients assigned to the node; then, use Kruskal's algorithm to derive the maximal weighted spanning tree from the m^2 nodes in each subband.
4) Create coarse-to-fine inter-scale edges (E_o^u) at each orientation: $x_{j+1}^u(i,k) \rightarrow x_j^u(i,k)$ for $j = 1, \ldots, J-1$. The probability functions P_o^u and P_i^u associated with edges in E_o^u and E_i^u are modeled according to (18) and (19), respectively. Let $P^u = P_o^u \cup P_i^u$.
5) Create WBN B: let $B^u(V^u, E^u = E_o^u \cup E^u, P^u = P_o^u \cup P_i^u)$ be the WBN of orientation u. Then, WBN is defined as $B(V = V^h \cup V^v \cup V^d, E = E^h \cup E^v \cup E^d, P = P^h \cup P^v \cup P^d)$.
6) Create observation nodes (Y) and arcs (E_n): create observation node $y_j^u(i,k) \in Y$ for variable node $x_j^u(i,k)$ and an arc $y_j^u(i,k) \rightarrow x_j^u(i,k) \in E_n$.
7) Derive the conditional probability of noisy observation (P_n): associate the probability functions defined in (20) with the arcs linking the observation nodes to the variable nodes. Let P_n denote the collection of probability functions.
8) Create WBN $B_n(V \cup Y, E \cup E_n, P \cup P_n)$ and associate the $(\frac{N}{m})^2$ wavelet coefficients $\{\mathcal{W}_j^u Z(i+mp, k+mq) | p,q = 0, \ldots \frac{N}{m}-1\}$ with observation node $y_j^u(i,k)$. Then, convert WBN B_n to a factor graph F_n, as described in Appendix A.
9) Estimate the wavelet coefficients in each realization (p,q) of B_n, as described in (27) and (30).
10) Reconstruct the denoised image $\hat{F}$ from the wavelet coefficients obtained in Step (9) and the coarsest approximating image $\mathcal{A}_j Z$.

The root of $J'(x) = 0$ can be derived by the following fixed point algorithm. Let x_t be the estimated value after the t-th iteration. Then, x_t can be derived from x_{t-1} as follows:

$$x_t = \left(1 + \frac{\sigma_y^2}{\omega {x_p}^2}\right)^{-1} \left(y - \sigma_y^2 \lambda \sum_j sign(x_{t-1} - x_j)\right). \quad (30)$$

The iterative estimation stops when the difference between the values x_{t-1} and x_t is smaller than some given threshold.

V. Denoising Algorithm and Experimental Results

In this section, we present our denoising algorithm, explain its implementation, and compare its performance with that of other methods. The proposed algorithm is summarized in Algorithm 1. The steps are as follows: Step (1) calculates the undecimated DWT of the input image; Steps (2) to (5) construct the WBN B; and Steps (6) to (8) create the WBN B_n used for denoising purposes; In Step (9), the wavelet coefficients are estimated from B_n by applying the max-product algorithm to the factor graph F_n for each realization of B_n. We use CDF 9/7 filters to process the undecimated DWT. Because CDF 9/7 filters are close to orthogonal wavelet filters, the noise variance of subbands at all scales can be set at σ_n^2, which is the image noise variance (see Appendix B). The variance σ_n^2 is used in the Wiener filtering in Step (2) as well as in deriving the MAP estimation of the wavelet coefficients in Step (9). The frequency count in Step (4) indicates the number of wavelet coefficients in a quantization bin. The size of a subband's quantization bin is set at $\frac{1}{4}$ of the standard deviation, measured from the wavelet coefficients in the subband.

We conduct experiments on two sets of images. Set I contains nine gray scale images (size 512×512 or 256×256) downloaded from the USC-SIPI image database [43]; and Set II (shown in Fig. 6) contains nine gray scale textures, some of which are from the Brodatz texture set. The parameter settings of the WBN denoising algorithm evaluated in the experiments are: $J = 4$ (the number of wavelet decompositions), $\omega = 0.64$ (Equation (18)), $\lambda = 0.45$ (Equation (19)), and the parameter $m = 4$ in Step (3). Each subband represents a 512×512 image and contains 4×4 nodes. The WBN B has $16 \times 4 \times 3$ variable nodes because there are four subbands in each of the three orientations. Hence, WBN B_n has $2 \times 16 \times 4 \times 3$ nodes, half of which are variable nodes and the rest are observation nodes. For each observation node in B_n, we assign 128×128 wavelet coefficients. Note that this number is large enough to derive the empirical probability in Step (4) of the WBN algorithm. There are also 128×128 realizations of B_n.

Next, we present the experimental results derived by our algorithm and compare its performance with that of two state-of-the-art algorithms: the BM3D algorithm [13] and the BLS-GSM algorithm [23]. The source codes for the BLS-GSM and BM3D methods are available from the websites of the respective authors. Our algorithm is executed on an Intel Core2Quad Q9300 CPU, with Windows XP and Matlab2007a. Tables I and II list the averages of five denoised PSNR results of the three compared methods for the images in Sets I and II respectively; white noise was added with $\sigma_n^2 = 10, 15, \ldots, 35$. In the tables, we divide the images into two groups and compare the average PSNR gain of our method over the other methods on the images in each group. Table III summarizes the results in Tables I and II. Note that the Figureprint and Baboon images are regarded as texture images because they are dominated by texture information. Interestingly, the PSNR gain of our method over BM3D is 0.1 dB for texture images (Group 2), but only 0.02 dB for the other images (Group 1). Moreover, from Table III, we observe that the gain of BM3D

TABLE I
AVERAGE PSNR RESULTS FOR IMAGE SET I

Image	Noise Level	PSNR		
		BLS−GSM	*BM3D*	*Our*
Einstein 512 × 512	$\sigma_n = 10$	34.1406	34.4392	**34.4837**
	$\sigma_n = 15$	32.6818	33.0331	**33.0874**
	$\sigma_n = 20$	31.7412	32.1694	**32.2287**
	$\sigma_n = 25$	31.0201	31.4186	**31.4682**
	$\sigma_n = 30$	30.4306	30.8709	**30.9175**
	$\sigma_n = 35$	29.9372	30.3777	**30.4116**
Boat 512 × 512	$\sigma_n = 10$	33.5080	33.8883	**33.9127**
	$\sigma_n = 15$	31.6452	32.1067	**32.1375**
	$\sigma_n = 20$	30.3194	30.8554	**30.8872**
	$\sigma_n = 25$	29.2846	29.8356	**29.8612**
	$\sigma_n = 30$	28.4395	29.0954	**29.1247**
	$\sigma_n = 35$	27.7301	28.2992	**28.3286**
Barbara 512 × 512	$\sigma_n = 10$	33.1518	34.9567	**34.9654**
	$\sigma_n = 15$	30.7724	33.0666	**33.0735**
	$\sigma_n = 20$	29.0984	31.7376	**31.7546**
	$\sigma_n = 25$	27.8214	30.7176	**30.7354**
	$\sigma_n = 30$	26.7998	29.7049	**29.7116**
	$\sigma_n = 35$	25.9775	**28.8879**	28.6753
Lena 512 × 512	$\sigma_n = 10$	35.9226	36.6367	**36.6375**
	$\sigma_n = 15$	34.1105	34.8782	**34.8886**
	$\sigma_n = 20$	32.8047	33.0567	**33.1246**
	$\sigma_n = 25$	31.7891	**32.5501**	32.4874
	$\sigma_n = 30$	30.9633	**31.6531**	31.5637
	$\sigma_n = 35$	30.2780	**31.0301**	30.9353
Cameraman 256 × 256	$\sigma_n = 10$	33.3739	34.1355	**34.1624**
	$\sigma_n = 15$	31.0334	31.8449	**31.8702**
	$\sigma_n = 20$	29.5762	30.3797	**30.4054**
	$\sigma_n = 25$	28.5242	29.4118	**29.4837**
	$\sigma_n = 30$	27.6872	28.5516	**28.6476**
	$\sigma_n = 35$	26.9689	27.8758	**27.9645**
House 256 × 256	$\sigma_n = 10$	35.3223	36.6638	**36.7135**
	$\sigma_n = 15$	33.7257	34.9028	**34.9476**
	$\sigma_n = 20$	32.5237	33.7349	**33.7782**
	$\sigma_n = 25$	31.5753	32.9084	**32.9406**
	$\sigma_n = 30$	30.7650	32.1240	**32.1574**
	$\sigma_n = 35$	30.0579	31.5103	**31.5385**
Pepper 256 × 256	$\sigma_n = 10$	34.0542	34.6889	**34.7184**
	$\sigma_n = 15$	32.0136	32.7290	**32.7602**
	$\sigma_n = 20$	30.5676	31.2671	**31.2957**
	$\sigma_n = 25$	29.4553	30.2223	**30.2607**
	$\sigma_n = 30$	28.5640	29.3090	**29.3276**
	$\sigma_n = 35$	27.8161	28.5795	**28.6014**
Fingerprint 512 × 512	$\sigma_n = 10$	32.1994	32.4628	**32.7284**
	$\sigma_n = 15$	29.9283	30.2822	**30.5975**
	$\sigma_n = 20$	28.3391	28.8085	**29.0732**
	$\sigma_n = 25$	27.0946	27.7112	**28.0364**
	$\sigma_n = 30$	26.0682	26.8245	**27.1262**
	$\sigma_n = 35$	25.1981	26.0995	**26.4075**
Baboon 512 × 512	$\sigma_n = 10$	30.3735	30.5798	**30.6623**
	$\sigma_n = 15$	27.8549	28.1390	**28.2505**
	$\sigma_n = 20$	26.1750	26.5307	**26.6275**
	$\sigma_n = 25$	24.9483	25.3495	**25.4374**
	$\sigma_n = 30$	24.0063	24.4407	**24.5238**
	$\sigma_n = 35$	23.2643	23.6554	**23.7522**

TABLE II
AVERAGE PSNR RESULTS FOR IMAGE SET II

Image	Noise Level	PSNR		
		BLS−GSM	*BM3D*	*Our*
F0 512 × 512	$\sigma_n = 10$	31.6031	32.52	**32.6212**
	$\sigma_n = 15$	29.6083	30.3213	**30.4143**
	$\sigma_n = 20$	28.2910	28.8595	**28.9314**
	$\sigma_n = 25$	27.2514	27.7279	**27.8087**
	$\sigma_n = 30$	26.3668	26.7751	**26.8644**
	$\sigma_n = 35$	25.6029	25.9243	**26.0136**
F7 512 × 512	$\sigma_n = 10$	29.9444	29.8987	**30.1253**
	$\sigma_n = 15$	27.2804	27.3043	**27.3765**
	$\sigma_n = 20$	25.5299	25.6216	**25.6782**
	$\sigma_n = 25$	24.2851	24.3522	**24.4186**
	$\sigma_n = 30$	23.3451	23.401	**23.4862**
	$\sigma_n = 35$	22.602	22.5878	**22.6574**
p3 512 × 512	$\sigma_n = 10$	31.1789	31.3155	**31.3385**
	$\sigma_n = 15$	28.8139	29.0087	**29.0274**
	$\sigma_n = 20$	27.2571	27.4678	**27.4936**
	$\sigma_n = 25$	26.1114	26.3278	**26.3414**
	$\sigma_n = 30$	25.2116	25.4362	**25.4535**
	$\sigma_n = 35$	24.4756	24.7034	**24.7153**
F3 512 × 512	$\sigma_n = 10$	29.2042	29.3575	**29.5361**
	$\sigma_n = 15$	26.3055	26.3832	**26.5738**
	$\sigma_n = 20$	24.3449	24.354	**24.5363**
	$\sigma_n = 25$	22.8854	22.8154	**23.0236**
	$\sigma_n = 30$	21.7442	21.6195	**21.8514**
	$\sigma_n = 35$	20.8213	20.6102	**20.9144**
w2 512 × 512	$\sigma_n = 10$	29.0994	29.4042	**29.496**
	$\sigma_n = 15$	26.499	26.798	**26.8816**
	$\sigma_n = 20$	24.8784	25.1843	**25.1547**
	$\sigma_n = 25$	23.7188	24.0512	**24.1376**
	$\sigma_n = 30$	22.8012	23.1637	**23.2473**
	$\sigma_n = 35$	22.0279	22.3965	**22.5062**
w6 512 × 512	$\sigma_n = 10$	29.6007	30.0207	**30.1433**
	$\sigma_n = 15$	27.1382	27.5495	**27.6887**
	$\sigma_n = 20$	25.5041	25.964	**26.1186**
	$\sigma_n = 25$	24.2672	24.8184	**24.9782**
	$\sigma_n = 30$	23.2643	23.9149	**24.1045**
	$\sigma_n = 35$	22.4208	23.1486	**23.3228**
F6 512 × 512	$\sigma_n = 10$	28.5821	28.5956	**28.7025**
	$\sigma_n = 15$	25.4728	25.5213	**25.6175**
	$\sigma_n = 20$	23.3581	23.4603	**23.563**
	$\sigma_n = 25$	21.7955	21.9042	**22.0154**
	$\sigma_n = 30$	20.6055	20.6784	**20.8706**
	$\sigma_n = 35$	19.6868	19.6259	**19.8154**
F8 512 × 512	$\sigma_n = 10$	29.0382	28.9781	**29.1012**
	$\sigma_n = 15$	26.1317	26.1037	**26.2387**
	$\sigma_n = 20$	24.2006	24.2121	**24.3382**
	$\sigma_n = 25$	22.7888	22.8208	**22.9647**
	$\sigma_n = 30$	21.7053	21.734	**21.883**
	$\sigma_n = 35$	20.8487	20.8181	**21.0225**
S2 512 × 512	$\sigma_n = 10$	29.3274	29.3933	**29.4072**
	$\sigma_n = 15$	26.5777	26.5264	**26.5942**
	$\sigma_n = 20$	24.895	24.6771	**24.9125**
	$\sigma_n = 25$	23.8043	23.4976	**23.8315**
	$\sigma_n = 30$	23.059	22.7346	**23.0627**
	$\sigma_n = 35$	22.523	22.2268	**22.5434**

over BLS-GSM is smaller on texture images (0.06 dB) than on the other images (1.0 dB). Texture images contain rich across-scales information. Since both BLS-GSM and our method exploit inter-scale coefficients for data estimation, they are more adaptive to texture images than non-texture images.

In Figures 5, 6, and 7, we perceptually compare some images that were denoised by the three methods. The perceptual quality of the images denoised by BM3D and our method is better than that of the BLS-GSM images. The highlighted regions in the figures compare certain details of the denoised images derived by all three methods with the corresponding

TABLE III
SUMMARY OF OUR PSNR GAIN

Image Group	Averge PSNR Gain	
	versus BLS-GSM	versus BM3D
Group 1 (First seven images in Table I)	1.0239	0.0207
Group 2 (Fingerprint + Baboon + Images in Table II)	0.2922	0.1268
All Images in Tables I and II	0.9403	0.0594

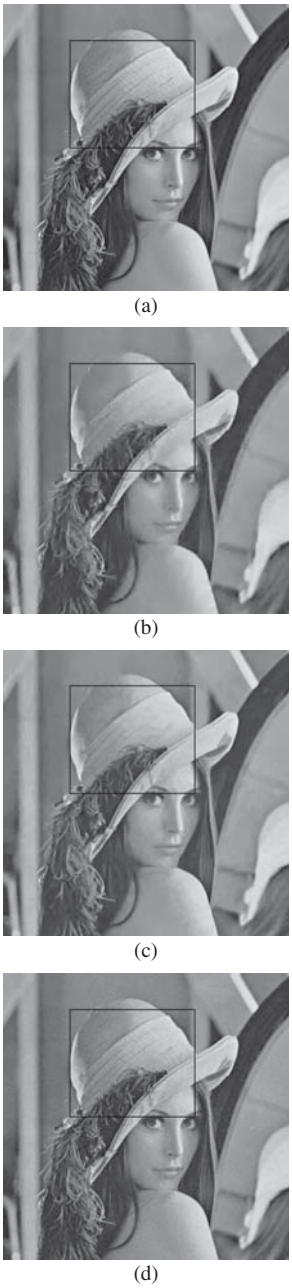


Fig. 5. Comparison of the denoised images derived by BLS-GSM, BM3D, and our algorithm. The noise standard deviation is $\sigma_n = 25$. (a) Original *Lena* image. (b) Denoised result of the BLS-GSM algorithm. (c) Denoised result of the BM3D algorithm. (d) Denoised result of our algorithm.

regions in the original images. We observe that BM3D tends to over-smooth textured regions, whereas the proposed method preserves more image details, as shown in the highlighted

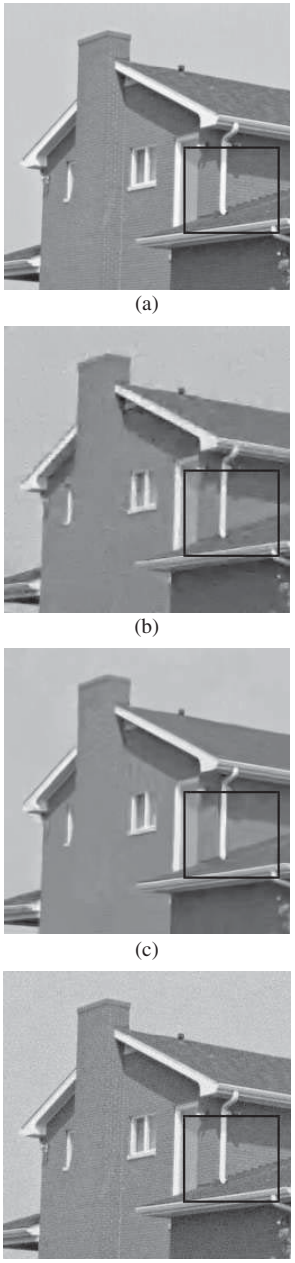
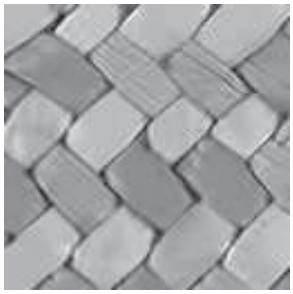


Fig. 6. Comparison of the denoised images derived by BLS-GSM, BM3D, and our algorithm. The noise standard deviation is $\sigma_n = 25$. (a) Original *House* image. (b) Denoised result of the BLS-GSM algorithm. (c) Denoised result of the BM3D algorithm. (d) Denoised result of our algorithm.

regions. Note that our method recovers the brick textures of the *House* image quite well, but the images recovered by BM3D and BLS-GSM are over-smooth. However, in some smooth regions, such as the sky regions of the *House* image, where the pixels are highly correlated with each other, our method's spanning tree structure does not impose sufficient constraints on the pair of pixels; thus, the visual quality of our method is more noisy than that of BM3D and BLS-GSM. Comparison of the denoised results of the *F0* image shows that a lot of details are removed by the BM3D method, but they are retained by our method, which carries information from the coarse scale to the fine scale and vice versa during the data estimation process.

(a)

(b)



(c)

(d)

Fig. 7. Comparison of the denoised images derived by BLS-GSM, BM3D, and the proposed algorithm. The standard deviation of the noise is $\sigma_n = 25$. (a) Top-left-corner of the original *F0* image. (b) Denoised image derived by the BLS-GSM algorithm. (c) Denoised image derived by the BM3D algorithm. (d) Denoised image derived by our algorithm.

The computational cost of our method is comprised of the cost of constructing a WBN and that of BP inference on the WBN. Let k be the number of labels for a wavelet coefficient; let an image be of size $N \times N$; and let J be the number of scales. The computational cost of constructing a WBN is $\mathcal{O}(3J(m^4 \log_2 m^4 + km^4 + N^2))$, where $m^4 \log_2 m^4$ is the complexity of Kruskal's algorithm for deriving an MWST on a wavelet patch, km^4 is the complexity of computing the conditional mutual information, and N^2 is the complexity of deriving the histogram. As mentioned in Section II, the average cost of the BP inference on a WBN is $\mathcal{O}(3JN^2k)$. The running times for our current unoptimized Matlab implementation, with $m = 4$, $J = 4$, and $k = 300$, on a Windows XP PC with Intel Core2Quad Q9300 CPU, is approximately 2500 seconds for 512×512 images.

VI. Conclusion

The Bayesian formula indicates that the denoising problem is essentially a prior probability modeling and estimation task. In this paper, we present a constructive data-adaptive procedure that derives a hidden graph structure from the wavelet coefficients. The graph is then used to model the prior probability of the original image for denoising purposes. Moreover, we show that if the network is a spanning tree, the standard BP algorithm can estimate MAP efficiently. We compare our denoised results with those derived by other approaches, including BM3D, and demonstrate that our method yields a better PSNR and better perceptual quality on the textured areas of an image. Extending our method to content sensitive wavelet patches is an issue that merits future study. We will also investigate ways to speed up our algorithm's execution time.

Appendix A
Conversion of a WBN to a Factor Graph and the Use of the Max-Product Algorithm

Bayesian networks and factor graphs are closely connected because one representation can always be converted into the equivalent form of the other representation [37]–[40].

Next, we describe the conversion of a network B_n to a factor graph F_n and explain the max-product message passing algorithm. Let $B_n = (V \cup Y, E \cup E_n, P \cup P_n)$ be a WBN constructed according to the description in Sections III and IV. In addition, let $x \to z$ be an arc in $E \cup E_n$ such that x is the source node and z is the sink node of the arc. We associate each variable node $z \in V$ with a factor node D_z. The vertices of the factor graph F_n consist of variable nodes $V \cup Y$ and factor nodes $D = \{D_z | z \in V\}$. The (undirected) edges in the factor graph F_n are constructed from B_n as follows. For each source node u of $u \to x$, there is an edge (u, D_x) between the variable node u and the factor node D_x; and for each sink node v of $x \to v$, there is an edge (x, D_v) between variable node x and factor node D_v. There are three types of arcs in B_n: 1) arcs from observation nodes to hidden variable nodes; 2) parent-child (inter-scale) arcs; and 3) (intra-scale) sibling arcs. In the following, we explain how to convert each type of arc in B_n to a factor graph F_n.

- Type 1: $y \to x$, where y is an observation node and x is a variable node in B_n. The arc is converted to two (undirected) edges (y, D_x) and (x, D_x).
- Type 2: $x_p \to x$, where x_p is a parent variable node at a coarser scale and x is the child variable node at a finer scale. The parent-child arc is converted to two edges (x_p, D_x) and (x, D_x).
- Type 3: sibling arc $x_b \to x$, where x_b is the variable node at the same scale and orientation as the variable node x. The sibling arc is converted to two edges

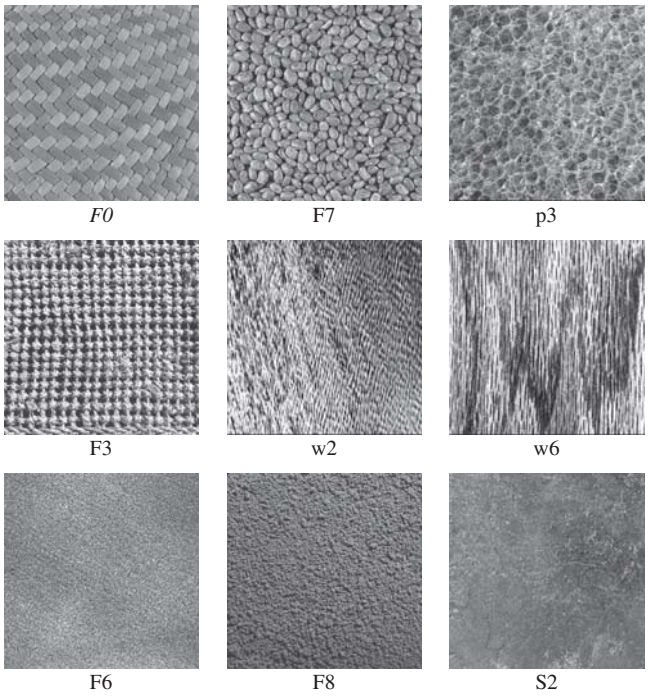


Fig. 8. Nine textures from image set II.

(x_b, D_x) and (x, D_x). Figs. 8(a) and 8(b) illustrate the conversion of a small Bayesian network B_n to an equivalent factor graph F_n.

The probability inference of a factor graph is derived through forward and backward message passing. To facilitate message passing, the factor graph F_n is arranged as a spanning tree with an arbitrarily chosen variable node in V as its root. As shown in Fig. 8(c), a tree is obtained by choosing x_1 as the root vertex for the factor graph in Fig. 8(b). The variable nodes and factor nodes in F_n are organized as alternating levels of the tree as follows: level 0 is the root of the tree; the nodes at level 1 are factor nodes, and the nodes at level 2 are variable nodes and so on. Note that all the variable nodes in Y are at the leaf level of the tree. Forward message passing begins with all nodes at the leaf level; then, messages are forwarded level by level to the root. Each vertex stores received messages for later use. Then, backward message passing commences at the root level and messages progress level by level to the leaf level. Thus, each edge in F_n passes messages in both directions [26].

APPENDIX B
NOISE VARIANCE IN EACH SUBBAND

The coefficients obtained by applying the j-th wavelet's decomposition to an $m \times m$ image $\tilde{N}$ of Gaussian white noise of variance σ_n^2 can be written as

$$\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \tilde{N} \begin{bmatrix} H_{0,j}^T & H_{1,j}^T, \end{bmatrix} \tag{31}$$

where $H_{0,j}$ and $H_{1,j}$ are, respectively, the scaling filter and wavelet filter at scale 2^j. If undecimated wavelet decomposition is applied, the dimensions of matrices $H_{0,j}$ and $H_{1,j}$ will be the same as that of image $\tilde{N}$.

Applying the vec operation to Equation (31), we obtain

$$vec\left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \tilde{N} \begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix}\right) = \left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \otimes \begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix}\right) vec(\tilde{N}) = \tilde{A}\, vec(\tilde{N}). \tag{32}$$

Because $E\{vec(\tilde{N})vec(\tilde{N})^T\} = \sigma_n^2 I$, where I is the identity matrix, we have

$$\begin{aligned} tr(E\{\tilde{A}\tilde{A}^T\}) &= \sigma_n^2 \cdot tr\left(\left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \otimes \begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix}\right) \left(\begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix} \otimes \begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix}\right)\right) \\ &= \sigma_n^2 \cdot tr\left(\left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix}\right) \otimes \left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix}\right)\right) \\ &= \sigma_n^2 \cdot tr\left(\begin{bmatrix} H_{0,j} \\ H_{1,j} \end{bmatrix} \begin{bmatrix} H_{0,j}^T & H_{1,j}^T \end{bmatrix}\right)^2 \\ &= \sigma_n^2 \cdot (tr(H_{0,j}H_{0,j}^T) + tr(H_{1,j}H_{1,j}^T))^2. \end{aligned} \tag{33}$$

The noise variances in subbands LL, LH, HL, and HH are

$$\sigma_n^2 \frac{(tr(H_{0,j}H_{0,j}^T))^2}{m^2} = \sigma_n^2 \rho_{j,LL}, \tag{34}$$

$$\begin{aligned} \sigma_n^2 \frac{(tr(H_{0,j}H_{0,j}^T)tr(H_{1,j}H_{1,j}^T))}{m^2} &= \sigma_n^2 \rho_{j,HL} \\ &= \sigma_n^2 \rho_{j,LH} \end{aligned} \tag{35}$$

$$\sigma_n^2 \frac{(tr(H_{1,j}H_{1,j}^T))^2}{m^2} = \sigma_n^2 \rho_{j,HH}. \tag{36}$$

respectively. If orthogonal wavelets are used, then the value of ρ is equal to 1 for all subbands.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their insightful comments, which have helped to significantly improve the quality of this paper.

REFERENCES

[1] M. J. Jordan, *Learning in Graphical Models*. Cambridge, MA: MIT Press, 1998.
[2] G. F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," *Mach. Learn.*, vol. 9, no. 4, pp. 309–347, Oct. 1992.
[3] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowlege and statistical data," *Mach. Learn.*, vol. 20, no. 3, pp. 197–243, 1995.
[4] D. Heckerman, "A tutorial on learning Bayesian networks," Microsoft Research, Mountain View, CA, Tech. Rep. MSR-TR-95-06, 1995.
[5] D. M. Chickering, D. Heckerman, and C. Meek, "A Bayesian approach to learning Bayesian networks with local structure," in *Proc. 13th Conf. Uncertainty Artif. Intell.*, 1997, pp. 80–89.
[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001.
[7] M. S. Crouse, R. D. Nowak, and R. G. Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models," *IEEE Trans. Signal Process.*, vol. 46, no. 4, pp. 886–902, Apr. 1998.
[8] G. Chang, B. Yu, and M. Baraniuk, "Spatially adaptive wavelet thresholding with context modeling for image denoising," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1522–1531, Sep. 2000.

[9] J. K. Romberg, H. Choi, and R. G. Baraniuk, "Bayesian tree-structured image modeling using wavelet-domain hidden Markov models," *IEEE Trans. Image Process.*, vol. 10, no. 7, pp. 1056–1068, Jul. 2001.
[10] P. Chatterjee and P. Milanfar, "Is denoising dead?" *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 895–911, Apr. 2010.
[11] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Model. Simul.*, vol. 4, no. 2, pp. 490–530, 2005.
[12] C. Kervrann and J. Boulanger, "Optimal spatial adaptation for patch-based image denoising," *IEEE Trans. Image Process.*, vol. 15, no. 10, pp. 2866–2878, Oct. 2006.
[13] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.
[14] P. Chatterjee and P. Milanfar, "Patch-based near-optimal image denoising," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1635–1649, Apr. 2011.
[15] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
[16] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Non-local sparse models for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis.*, Kyoto, Japan, 2009, pp. 2272–2279.
[17] P. Milanfar, "A tour of modern image filtering: New Insights and methods, both practical and theoretical," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 106–128, Jan. 2013.
[18] M. Malfait and D. Roose, "Wavelet-based image denoising using a Markov random field a priori model," *IEEE Trans. Image Process.*, vol. 6, no. 4, pp. 549–565, Apr. 1997.
[19] S. Mallat and W. L. Hwang, "Singularity detection and processing with wavelets," *IEEE Trans. Inf. Process.*, vol. 38, no. 2, pp. 617–643, Mar. 1992.
[20] C. L. Tu, W. L. Hwang, and J. Ho, "Analysis of singularities from modulus maxima of complex wavelets," *IEEE Trans. Inf. Process.*, vol. 51, no. 3, pp. 1049–1062, Mar. 2005.
[21] A. Pizurica, W. Philips, I. Lemahieu, and M. Acheroy, "A joint inter- and intrascale statistical model for Bayesian wavelet based image denoising," *IEEE Trans. Image Process.*, vol. 11, no. 5, pp. 545–557, May 2002.
[22] E. P. Simoncelli, "Bayesian denoising of visual images in the wavelet domain," in *Bayesian Inference in Wavelet Based Models* (Lecture Notes in Statistics), vol. 141. New York: Springer-Verlag, 1999, ch. 18, pp. 291–308.
[23] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using Gaussian scale mixtures in the wavelet domain," *IEEE Trans. Image Process.*, vol. 12, no. 11, pp. 1338–1351, Nov. 2003.
[24] J. A. Guerrero-Colon, "Bayesian methods for the restoration of digital camera images in overcomplete pyramids," Ph.D. dissertation, Escuela Tecnica Superior de Ingenierias Informatica, Univ. Granada, Granada, Spain, 2008.
[25] B. Goossens, A. Pizurica, and W. Philips, "Image denoising using mixtures of projected Gaussian scale mixtures," *IEEE Trans. Image Process.*, vol. 18, no. 8, pp. 1689–1702, Aug. 2009.
[26] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*. Cambridge, MA: MIT Press, 1998.
[27] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," in *Exploring Artificial Intelligence in the New Millennium*. San Francisco, CA: Morgan Kaufmann, Jan. 2003, ch. 8, pp. 239–236.
[28] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, nos. 1–4, pp. 259–268, 1992.
[29] P. Felzenszwalb and D. Huttenlocher, "Efficient belief propagation for early vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 1. Mar. 2004, pp. 261–268.
[30] J. Yedidia, W. T. Freeman, and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2282–2312, Jul. 2005.
[31] X. Lan, S. Roth, D. Huttenlocher, and M. Black, "Efficient belief propagation with learned higher-order Markov random fields," in *Proc. Eur. Conf. Comput. Vis.*, vol. 2. 2006, pp. 269–282.
[32] J. Gonzalez, Y. Low, and C. Guestrin, "Parallel splash belief propagation," *J. Mach. Learn. Res.*, vol. 1, pp. 1–48, Oct. 2009.
[33] C. K. Chow and C. N. Lui, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Inf. Theory*, vol. 14, no. 3, pp. 462–467, May 1968.
[34] D. Geiger, "An entropy-based learning algorithm of Bayesian conditional trees," in *Proc. 8th Conf. Annu. Conf. Uncertainty Artif. Intell.*, 1992, pp. 92–97.
[35] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, pp. 131–163, Jan. 1997.
[36] D. Grossman and P. Domingos, "Learning Bayesian network classifiers by maximizing conditional likelihood," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, pp. 1–8.
[37] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 325–343, Mar. 2000.
[38] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*. San Mateo, CA: Margan Kaufmann, 1988.
[39] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.
[40] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *Int. J. Comput. Vis.*, vol. 70, no. 1, pp. 41–54, Oct. 2006.
[41] J. B. Kruskal, "On the shortest spanning subtrees of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, 1956.
[42] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, "Learning low-level vision," *Int. J. Comput. Vis.*, vol. 40, no. 1, pp. 25–47, 2000.
[43] *The USC-SIPI Image Database*. (1977) [Online]. Available: http://sipi.usc.edu/database/index.html

Jinn Ho received the M.S. degree in mathematics from National Taiwan University, Taipei, Taiwan, in 1997.

He is currently a Research Assistant with the Institute of Information Science, Academia Sinica, Taipei, Taiwan. His current research interests include PDE methods for image processing, numerical methods, and statistical analysis for gene expression.

Wen-Liang Hwang (SM'05) received the B.S. degree in nuclear engineering from National Tsing Hua University, Hsinchu, Taiwan, the M.S. degree in electrical engineering from the Polytechnic Institute of New York, New York, and the Ph.D. degree in computer science from New York University, New York, in 1993.

He was a Post-Doctoral Researcher with the Department of Mathematics, University of California, Irvine, in 1994. In 1995, he was a member of the Institute of Information Science, Academia Sinica, Taipei, Taiwan, where he is currently a Research Fellow. He has co-authored *Practical Time-Frequency Analysis* (San Diego, CA: Academic Press, 1998). His current research interests include wavelet analysis, mathematical modeling, and signal, image, and video processing.

Dr. Hwang was a recipient of the Academia Sinica Research Award for Junior Researchers in 2001. He is an Associate Editor of the *Journal of Wavelet Theory and Applications* and the *International Journal of Wavelets, Multiresolution and Information Processing*.