Erik B. Sudderth and
William T. Freeman

Signal and Image Processing with Belief Propagation

Many practical signal processing applications involve large, complex collections of hidden variables and uncertain parameters. For example, modern communication systems typically couple sophisticated error correcting codes with schemes for adaptive channel equalization. Additionally, many computer vision algorithms use prior knowledge about the statistics of typical surfaces to infer the three-dimensional (3-D) shape of a scene from ambiguous, local image measurements. Probabilistic graphical models provide a powerful, general framework for designing systems like these. In this approach, graphs are used to decompose joint distributions into a set of local constraints and dependencies. Such modular structure provides an intuitive language for expressing domain-specific knowledge and facilitates the transfer of modeling advances to new applications. Once a problem has been formulated using a graphical model, a wide range of efficient algorithms for statistical learning and inference can then be directly applied.

In this column, we review a particularly effective inference algorithm known as belief propagation (BP). After describing its message-passing structure, we demonstrate the interplay of statistical modeling and inference in two challenging applications: denoising discrete signals transmitted over noisy channels, and dense 3-D reconstruction from stereo images.

GRAPHICAL MODELS AND FACTOR GRAPHS

Several different formalisms have been proposed that use graphs to represent probability distributions [1], [2]. For example, *directed* graphical models, or Bayesian networks, are widely used in artificial intelligence to deduce causal, generative processes. Special cases of interest in control and signal processing include *hidden Markov models* (HMMs) and continuous *state space* models. Alternatively, *undirected* graphical models, or *Markov random fields* (MRFs), provide popular models for the symmetric dependencies arising with spatial or image data.

In what follows, we focus on models defined by *factor graphs* [3], [4], which are often used in communications and information theory. These bipartite graphs have two sets of nodes or vertices $\mathcal{V}$ and $\mathcal{F}$. Each variable node $s \in \mathcal{V}$ is associated with a random variable x_s, which for now we assume takes values in some finite, discrete set $\mathcal{X}_s$. These hidden variables could represent signals, parameters, or other processes that influence the modeled system, but are not directly observed. Each factor node $f \in \mathcal{F}$ is then uniquely indexed by a subset $f \subset \mathcal{V}$ of the hidden variables that directly interact. In particular, the joint distribution of $x \triangleq \{x_s \mid s \in \mathcal{V}\}$ is defined as a normalized product of nonnegative potentials or compatibility functions:

$$p(x) \propto \prod_{s \in \mathcal{V}} \psi_s(x_s) \prod_{f \in \mathcal{F}} \psi_f(x_f). \quad (1)$$

In this expression, $x_f \triangleq \{x_s \mid s \in f\}$ are the hidden variables associated with factor node f, while the proportionality symbol represents division by a normalization constant chosen so that $\sum_x p(x) = 1$. We graphically illustrate this factorization by drawing edges linking each variable node to all dependent factors, as in the example graphs of Figure 1. In cases where factor nodes couple pairs of variable nodes $s, t \in \mathcal{V}$, we denote the corresponding compatibility function by $\psi_{st}(x_s, x_t)$.

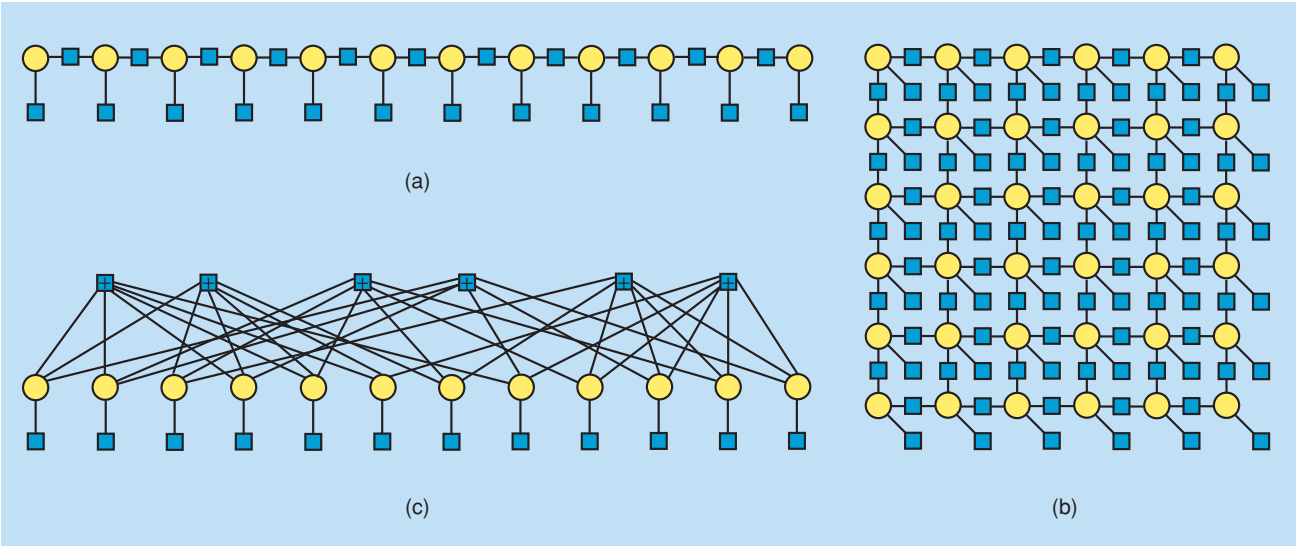
In many cases, factor graphs represent the hidden variables' posterior distribution $p(x \mid y)$ given observed data y. For notational simplicity, however, we assume that such observations have been implicitly encoded by appropriately redefining the local compatibilities $\psi_s(x_s)$.

MODELING TEMPORAL AND SPATIAL DEPENDENCIES

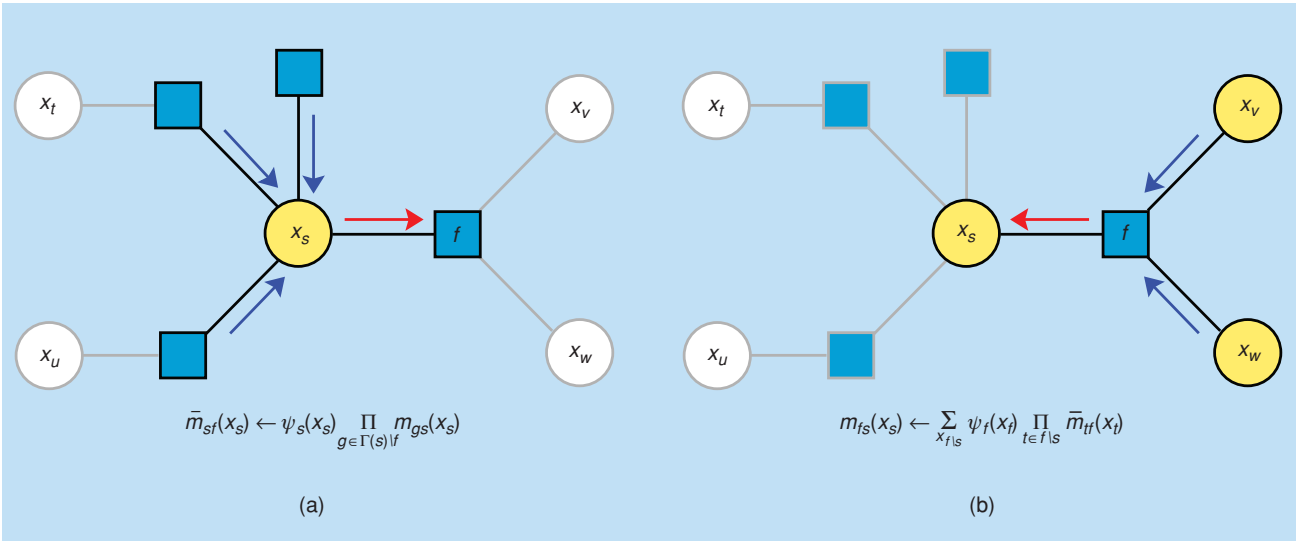
Many signal processing applications involve estimation of a temporal or spatial stochastic process. As illustrated in Figure 1(a) and (b), graphical models for these problems typically associate a hidden variable with each discrete sample of the underlying signal. The simplest associated factor graphs then combine two types of compatibility functions. Functions describing local observations, as represented by degree one factor nodes, can model potentially nonlinear or inhomogeneous measurements. Neighboring samples are then connected by "pairwise" compatibility functions encoding prior knowledge about the underlying process. In an oceanographic remote sensing application, for example, x_s could represent sea surface temperature at a particular site s, while Gaussian pairwise potentials $\psi_{st}(x_s, x_t) \propto \exp\{-(x_s - x_t)^2/2\sigma^2\}$ are matched to the expected spatial scale of temperature fluctuations. Observation compatibilities $\psi_s(x_s)$ might then encode knowledge about the accuracy of satellite-born instruments, and samples corrupted by inclement weather. Our later results explore in greater detail an application of a related Markov random field to stereo reconstruction.

Digital Object Identifier 10.1109/MSP.2007.914235

1053-5888/08/$25.00©2008IEEE



[FIG1] Factor graph representations of three probabilistic models. Circular nodes are random variables, which interact via square factor nodes (shaded). (a) An HMM. (b) A grid-structured MRF. (c) An LDPC code.



[FIG2] Computation of new belief propagation messages (red arrows) given incoming messages from neighboring nodes (blue arrows). (a) Message $\bar{m}_{sf}(x_s)$ is sent from variable node s to factor node f. (b) Message $m_{fs}(x_s)$ is sent from factor node f to variable node s.

DESIGNING LOW-DENSITY PARITY CHECK CODES

Graphical models also provide a unifying framework for the design and analysis of error correcting codes. In most graphical code representations, each variable node is associated with a bit $x_s \in \{0, 1\}$ in the transmitted code word, and observation compatibilities are determined by the noisy communication channel. For example, to model a binary symmetric channel with error probability ϵ, we set $\psi_s(x_s) = 1 - \epsilon$ when x_s equals the corresponding received bit, and $\psi_s(x_s) = \epsilon$ otherwise. Each factor node then constrains several bits via a *parity check*:

$$\psi_f(x_s, x_t, \ldots, x_v) = \begin{cases} 1 & x_s \oplus x_t \oplus \cdots \oplus x_v = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

The $\oplus$ operator represents modulo-2 addition, so that all valid code words have an even number of "active" bits connected to each parity check node.

In contrast with most signal processing applications, where graph structure is determined by physical and statistical relationships, graphical codes are engineered to have good error correcting properties. Figure 1(c) illustrates a regular low density parity check (LDPC) code [5], in which each parity check or variable node has a small, fixed number of neighbors. The neighborhood structure is chosen via a pseudorandom permutation, analogously to the random code ensembles used in many information theoretic proofs. In addition to having desirable asymptotic properties, the sparse graphical structure of LDPC codes makes them particularly well suited to

approximate inference algorithms like belief propagation.

BELIEF PROPAGATION

The *belief propagation* algorithm propagates information throughout a graphical model via a series of messages sent between neighboring nodes [2], [6]. Let $\Gamma(s)$ denote the set of factor nodes f that are directly connected to, or neighbor, variable node s. As summarized in Figure. 2, factor graph implementations of BP alternate between the computation of messages $\bar{m}_{sf}(x_s)$ from variables to factors, and messages $m_{fs}(x_s)$ from factors to variables.

Variable nodes s fuse information via the product of local compatibilities $\psi_s(x_s)$ with incoming messages $m_{gs}(x_s)$ from neighboring factor nodes. Each message is a $|\mathcal{X}_s|$-dimensional vector, whose nonnegative entries encode likelihoods of x_s given information from different parts of the graph. To avoid double-counting, the equation used to update the message $\bar{m}_{sf}(x_s)$ sent to factor node f excludes the incoming message $m_{fs}(x_s)$ from that same factor.

To compute an outgoing message $m_{fs}(x_s)$ from factor node f, we first rescale that factor's compatibility function $\psi_f(x_f)$ by the incoming messages $\bar{m}_{tf}(x_t)$ from neighbors t other than s ($t \in f \setminus s$). We then marginalize or sum this function over all possible neighboring states and thus produce a message function that depends only on $x_s \in \mathcal{X}_s$. For pairwise factors, this message update operation is equivalent to a matrix-vector product. More generally, message updates are generalized convolution operators that compress knowledge about x_f, preserving only those summary statistics that are informative about x_s. Due to the form of the message update equations in Figure 2, BP is also known as the *sum-product* algorithm [4].

As messages are iteratively updated, the BP algorithm propagates information throughout the graphical model. At any iteration, an estimate of the posterior marginal distribution of x_s, given information from all compatibilities, can be computed by multiplying together all of the incoming messages from neighboring factor nodes:

$$\hat{p}_s(x_s) \propto \psi_s(x_s) \prod_{f \in \Gamma(s)} m_{fs}(x_s). \qquad (3)$$

As before, the normalization constant is chosen so that $\sum_{x_s} \hat{p}_s(x_s) = 1$. The following sections review statistical uses of marginal estimates and survey cases in which the so-called belief estimates of (3) well approximate the true marginals. Note that these beliefs are invariant to rescalings $m_{fs}(x_s) \leftarrow \alpha_{fs} m_{fs}(x_s)$ of the messages by constants $\alpha_{fs} > 0$. For numerical stability, most implementations normalize each message so that its entries sum to one.

MAPPING APPLICATIONS TO INFERENCE PROBLEMS

One widely used, decision theoretic approach to statistical inference begins by defining the *cost* $C(x, \hat{x})$ of selecting $\hat{x}$ when the true, hidden state is x. Given observed data y, a decision $\hat{x}$ is then made by minimizing the expected cost $\mathbb{E}_{x|y}[C(x, \hat{x})\,|y]$. When this cost decomposes as $C(x, \hat{x}) = \sum_s C_s(x_s, \hat{x}_s)$, the posterior marginals $p(x_s \mid y)$ provide sufficient statistics for this decision. For example, if $C(x, \hat{x}) = ||x - \hat{x}||^2$, the decision minimizing the mean squared error is $\hat{x}_s = \mathbb{E}_{x_s|y}[x_s \mid y]$. Alternatively, setting $\hat{x}_s$ to the mode of $p(x_s \mid y)$ minimizes the number of misclassified variables (in coding, the bit error rate). For the many applications in which decomposable costs are appropriate, the marginal estimates computed by the sum-product form of BP can thus be extremely effective.

In some applications, one would instead like to minimize the probability that *any* part of the global state estimate $\hat{x}$ is incorrect (in coding, the word error rate). In these situations, a closely related max-product form of BP is more appropriate [1], [4].

DYNAMIC PROGRAMMING IN TREES

BP was first derived as an exact inference algorithm for tree-structured graphical models, in which no sequence of edges forms a cycle or loop. To illustrate this interpretation, consider the graphical model in Figure 2. The marginal distribution of x_s can be computed as follows:

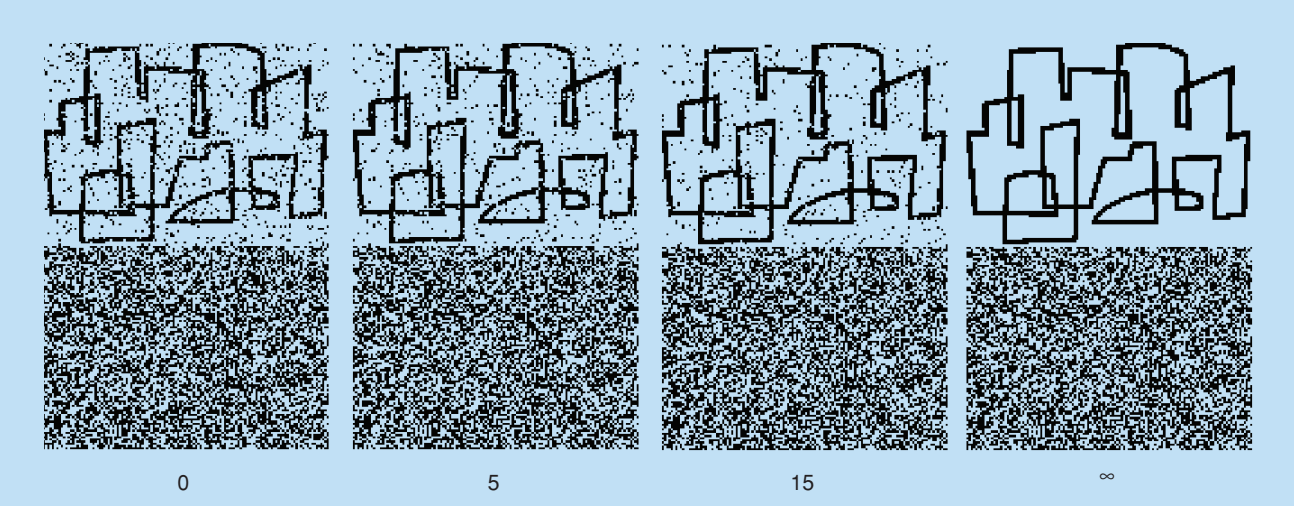
$$\begin{aligned} p(x_s) &\propto \sum_{x_t, x_u, x_v, x_w} \psi_s(x_s) \\ &\quad \times \psi_{st}(x_s, x_t)\, \psi_{su}(x_s, x_u) \\ &\quad \times \psi_{svw}(x_s, x_v, x_w) \qquad (4) \\ &\propto \psi_s(x_s) \cdot \left[\sum_{x_t} \psi_{st}(x_s, x_t) \right] \\ &\quad \cdot \left[\sum_{x_u} \psi_{su}(x_s, x_u) \right] \\ &\quad \cdot \left[\sum_{x_v, x_w} \psi_{svw}(x_s, x_v, x_w) \right]. \qquad (5) \end{aligned}$$

Here, we have used the distributive law to change the order in which the summation is carried out [4]. Note that the three summations in (5) are precisely equal to the messages computed by the BP algorithm in this graph. Using an induction argument, this approach can be generalized to show that BP computes exact marginals in any tree-structured model [6].

For trees, BP is essentially a distributed variant of dynamic programming. Its recursive, local decomposition of summations can provide dramatic computational savings. Consider, for example, a tree in which N variables taking one of K states are connected by pairwise compatibility functions. If ordered appropriately, each message must only be computed once, and exact marginals can be determined in $\mathcal{O}(NK^2)$ operations. In contrast, brute force enumeration of all joint states (as in (4)) scales as $\mathcal{O}(K^N)$.

LOOPY BELIEF PROPAGATION

The BP message update equations summarized in Figure 2 only involve passing messages between neighboring nodes. Computationally, it is thus straightforward to apply these same local message-updates in graphs with cycles. In most such models, however, this *loopy BP* algorithm will *not* compute exact marginal distributions. Algebraically, it fails because the presence of cycles eliminates the factorization underlying the dynamic programming derivation in (4) and (5). Probabilistically, problems arise from the different Markov properties, or conditional independencies, underlying graphs with cycles. In particular, the belief



[FIG3] Belief propagation decoding of a (3,6) regular LDPC code, given a binary input message (top half) corrupted by noise with error probability $\epsilon = 0.08$ (as set up in [5]). We show the received codeword including parity bits (zero iterations, left), the bits that maximize the posterior marginals after five and 15 BP iterations, and the final, correctly decoded signal after convergence to a loopy BP fixed point (the Stata center, home of the MIT Computer Science and Artificial Intelligence Laboratory).

update of (3) implicitly assumes that incoming messages are statistically independent. For graphs with cycles, however, loopy message passing leads to feedback and unmodeled message dependencies.

Despite these potential shortcomings, in practice loopy BP is often remarkably effective. It is probably best known as the decoding algorithm underlying LDPC and turbo codes [5], but has also been used with success in such areas as image processing and computer vision [3], [7], [8]. Intuitively, loopy BP seems to be most accurate when local regions of the graph have tree-like properties, so that messages are approximately independent. Indeed, recent theoretical results confirm that loopy BP has desirable properties when cycles are long or compatibilities are sufficiently weak. However, it can also be effective in graphs with short cycles, like the grid-structured MRFs arising in the dense stereo reconstruction application discussed later.

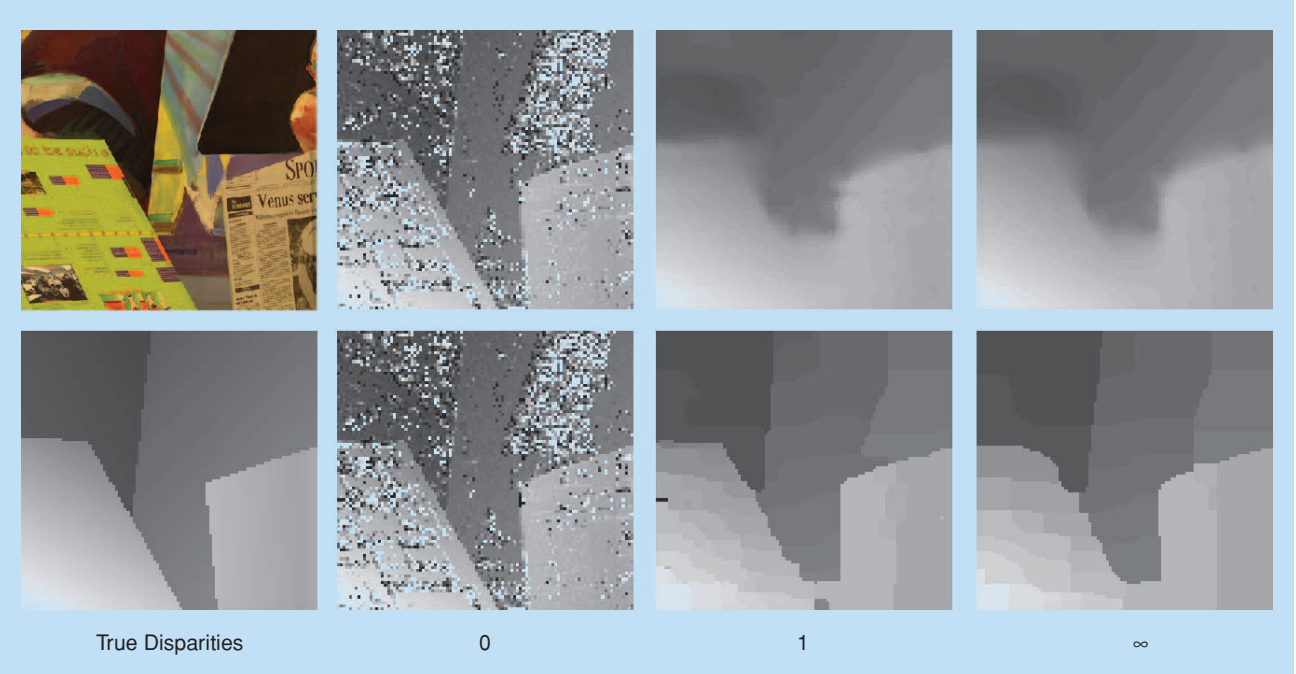
Motivated to explain the empirical success of loopy BP, the authors of [2] showed that when BP converges, the resulting marginal probabilities are stationary points of a quantity from statistical physics known as the Bethe approximation to the free energy. The true free energy is an objective function whose global minimizers are exact marginal distributions, and in tree-structured graphs is equivalent to the Bethe free energy. This work revealed connections between BP and variational methods for approximate inference, which also minimize approximate free energies. Further research has established local consistency properties of BP as well as bounds on the accuracy of its marginal estimates [1].

The preceding results assume that the BP message-passing updates have converged to some fixed point. BP may fail to converge in graphs with cycles, and different message update rules (which share the same fixed points) have been proposed to allow faster or more robust convergence. One modification employs damped message updates, so that the transmitted message is ρ times the message computed as in Figure 2 plus $(1 - \rho)$ times the message from the preceding iteration, for some $0 < \rho < 1$. This can lead to convergence in graphs with cycles for which BP's message-passing dynamics would otherwise be unstable or oscillate. Another approach updates messages sequentially along trees embedded within the graph [1], rather than in parallel over all nodes, allowing local observations to propagate more quickly.

For graphical models with long cycles or relatively weak dependencies, the basic BP message updates outlined in Figure 2 typically perform quite well. For applications with more complex statistical structure, there is a rich literature exploring alternative families of variational methods [1], [3], including a generalized BP algorithm based on higher-order extensions of the Bethe free energy [2]. Typically, however, such improvements in inference accuracy come at the cost of increased computation and implementation complexity.

APPLICATION: ERROR CORRECTING CODES

To demonstrate the practical behavior of loopy BP, we begin by decoding a simple binary message transmitted over a binary symmetric channel. As outlined previously, we construct an LDPC code by sparsely connecting parity checks to message bits in a random pattern. We use a so-called (3,6) regular LDPC code, in which each variable node is connected to three parity checks, and each factor node to six message bits as in Figure 1(c). For this rate one-half code, half of the message bits may be set arbitrarily to encode information, while the others are determined by the parity checks and thus create structured redundancy. Figure 3 shows an example systematic message encoding [5], in which the first half of the transmitted bits are the uncoded message of



[FIG4] Belief propagation stereo reconstruction for the "Venus" image from the Middlebury stereo database [9] . *Left:* Reference image (top) and ground truth disparities (bottom), where darker intensities correspond to more distant scene points. *Right:* BP disparity estimates based solely on local evidence (zero iterations), after one iteration, and at convergence. We compare a pairwise MRF with Gaussian potentials (top) to one with robust, truncated potentials (bottom).

interest. To aid visualization, we choose this message to be a simple binary image, but the LDPC code does *not* model or exploit this spatial structure.

To decode a noise-corrupted message via loopy BP, we pass messages in a parallel fashion. Each iteration begins by sending new messages from each variable node to all neighboring factor nodes and then updates the outgoing messages from each parity check. As illustrated in Figure 3, this allows information from unviolated parity checks to propagate throughout the message, gradually correcting noisy bits. These BP message updates continue until the thresholded marginal probabilities define a valid codeword or until some maximum number of iterations is reached. In practice, BP decoding typically fails due to oscillations in the message update dynamics, rather than convergence to an incorrect code word [5].

The outstanding empirical performance of message-passing decoders has inspired extensive research on alternatives to regular LDPC codes [5]. One widely used trick removes all cycles of length four from the factor graph, so that no two parity checks share a common pair of message bits. This reduces over-counting that can otherwise hamper loopy BP. In applications where transmitting million-bit blocks is feasible, extremely effective codes can be found by designing random *ensembles* of sparse parity check matrices whose typical members are good. For more moderate block lengths, graphical code design is a topic of ongoing research.

APPLICATION: DENSE STEREO RECONSTRUCTION

To further illustrate the practical behavior of loopy BP, we now consider an application in spatial signal processing. Many low-level computer vision tasks have a common form: gather local evidence, then propagate that evidence across space [8]. Belief propagation provides machinery to solve such spatial inference problems in factor graphs. Binocular stereo reconstruction, in which images captured at two offset locations are used to estimate scene depths, is typical of this type of inference problem. Textured regions often provide strong local evidence of the amount of depth-induced parallax between a pair of images, while untextured regions do not. For these areas, depths must be estimated using information from other image regions.

Camera calibration information is used in an initial stereo preprocessing step, called image rectification [9], in which the two images are prewarped and placed in row-by-row correspondence. After rectification, each row of the left camera image corresponds to the same row of the right image, with some unknown offset between matching pixels in the two images. Under the assumption of perspective projection, with cameras offset from one another by $\triangle$ and having focal length f, the scene depth z_s at pixel s is related to image displacement d_s as follows:

$$z_s = \frac{f\triangle}{d_s - \triangle}. \qquad (6)$$

We thus focus on inferring the image-based displacement, or disparity, between corresponding pixels, and later translate these into the desired scene depths.

MODELING DISPARITIES WITH PAIRWISE MRFs

Sophisticated graphical models relating image intensities to disparities can take many quantities into account [10], including occlusion boundaries and non-Lambertian reflectances. A simpler model, presented here, captures only the most basic features. Under the assumption of diffuse reflectance, 3-D scene locations produce the same intensities in different cameras. Let d_s denote a candidate disparity at location (i_s, j_s) in the left, or *reference*, image. In the simplest case, we use a Gaussian noise process of variance σ^2 to account for any deviations of the right camera intensity, $I_R(i_s + d_s, j_s)$, at the position of true disparity $i_s + d_s$, from the intensity at the corresponding position in the left camera, $I_L(i_s, j_s)$. We then define the likelihood of disparity d_s at image site s via the following observation potential:

$$\psi_s(d_s) \propto \exp\left\{ -\frac{1}{2\sigma^2} \times \left(I_L(i_s, j_s) - I_R(i_s + d_s, j_s)\right)^2 \right\}. \quad (7)$$

Note that even though the form of this potential is Gaussian, its dependence on image intensities typically makes it a multimodal, highly non-Gaussian function of the hidden disparity d_s. To improve the robustness of this matching process, our experiments extend the basic likelihood of (7) to aggregate pixel distances over a 3×3 window around each candidate disparity.

For realistic scenes, the local evidence for a match can be weak, particularly in textureless regions. We thus need to make assumptions about probable surface shapes to recover good depth estimates. Since many objects have smooth surfaces, it is reasonable to assume that the differences between disparities at neighboring positions are Gaussian distributed with mean zero. However, this fails to account for the very large disparity differences observed at object contours. For this reason, researchers often place *truncated* Gaussian compatibility functions on the depth differences between neighboring pixels:

$$\psi_{st}(d_s, d_t) \propto \exp \times \left\{ -\frac{1}{2\gamma^2} \min\left((d_s - d_t)^2, \delta_0^2\right) \right\}. \quad (8)$$

Here, γ encodes the expected smoothness of objects, while the threshold δ_0 sets the maximum degree to which large disparities are penalized. A similar threshold can be added to the observation potential of (7), to approximately account for specularities and occlusions.

In Figure 4, we apply the BP algorithm to two pairwise MRFs with $K = 50$ candidate disparities, estimating depths in a standard test scene [9]. We update messages by alternatively passing them along rows of the nearest–neighbor grid, and then along columns. When using truncated Gaussian potentials, BP very quickly converges to accurate disparity estimates. Indeed, several of the top performing stereo algorithms are based upon belief propagation [9], [10]. In contrast, applying the same message updates to a model with nontruncated potentials leads to significant blurring of the boundaries between this scene's planar regions.

LEARNING EFFECTIVE SIGNAL MODELS

The stereo reconstruction example demonstrates the importance of coupling BP with an accurate model for the underlying signal. While manual tuning of potential functions and parameters is sometimes feasible, complex applications demand more sophisticated, data-driven learning methods [8]. In graphical models, generalizations of the *expectation maximization* (EM) algorithm are often effective, in which a variational method like loopy BP is used to approximately infer summary statistics of hidden variables [1], [3].

EXTENSION: BELIEF PROPAGATION WITH CONTINUOUS VARIABLES

In many signal and image processing applications, hidden variables naturally take values in continuous spaces. In principle, loopy BP is easily extendable to such graphical models: one simply replaces the summations of Figure 2 by integrals. However, the resulting message functions typically lack tractable closed forms. A notable exception occurs when all variables are jointly Gaussian, so that messages can be parameterized by their mean and covariance. Loopy BP then becomes a generalized form of the Kalman filter [4].

For low-dimensional continuous variables, like the disparities needed for stereo reconstruction, a fixed discretization is often reasonable. For higher-dimensional spaces, care must be taken to keep BP message updates computationally feasible. When variables are coupled by regular, symmetric potentials, fast convolution methods like the fast Fourier transform can be very effective [7]. Alternatively, strong local evidence cues can be used to fix a data-driven lineup of candidate states for each variable [8]. More generally, Monte Carlo methods inspired by particle filters can dynamically adapt computational resources as message-passing proceeds. Variants of this nonparametric belief propagation (NBP) [11] approach have been used for visual recognition and tracking, sensor network localization, and equalization of communication channels [12].

AUTHORS

Erik B. Sudderth (sudderth@eecs.berkeley.edu) is a postdoctoral scholar in the Department of Electrical Engineering and Computer Science at the University of California, Berkeley. His research interests include graphical models, nonparametric Bayesian methods, and applications of machine learning in signal processing and visual scene analysis.

William T. Freeman (billf@mit.edu) is a professor of electrical engineering and computer science at the Massachusetts Institute of Technology. His research interests include Bayesian models of visual perception, and machine learning applied to computer vision and computer graphics. He enjoys flying cameras in kites.

(continued on page 141)

each rate-distortion point represents a separate nonscalable bit stream. The diagram additionally shows a rate-distortion curve representing the simulcast of the single-layer H.264/AVC bit stream with the fidelity of the SVC base layer and another single-layer H.264/AVC bit stream with the fidelity specified in the diagram.

In Figure 2(b), SVC spatial scalability is compared to H.264/AVC single-layer coding and simulcast of two spatial resolutions with H.264/AVC. Two spatial layers with a dyadic relation are provided. This means that the horizontal and vertical resolutions of the enhancement layer (4CIF) double the horizontal and vertical resolutions of the base layer (CIF). For each point on the 4CIF curve, the spatial scalable bit stream comprises the corresponding point on the CIF curve with about one-half to one-third of the overall bit rate.

The comparison shows that SVC can provide a suitable degree of scalability at the cost of approximately 10% bit rate increase compared to the bit rate of single-layer H.264/AVC coding. This bit rate increase usually depends on the degree of scalability, the bit rate range, and the spatial resolution of the included representations. The comparison also shows that SVC is clearly superior to simulcasting single-layer streams for different spatial resolutions or bit rates.

COMPLEXITY

Due to the substantially higher complexity of the underlying H.264/AVC standard relative to prior video coding standards, the computational resources necessary for decoding an SVC bit stream are usually higher than those for scalable profiles of older standards. However, due to the single-loop decoding feature of SVC, the required decoding complexity overhead for spatial and especially fidelity scalable coding relative to the underlying single-layer coding standard has been substantially reduced. For simple SVC configurations like fidelity scalability or dyadic spatial scalability, SVC decoders require 10–50% more computational resources than single-layer H.264/AVC decoders for the same target resolution and bit rate. The decoder complexity depends on the number of layers that are employed for interlayer prediction, the resolution ratios between the layers, and the bit rate. The minimization of the encoder complexity overhead for scalable coding has become an active research area in the video coding community.

FURTHER TECHNICAL DEVELOPMENTS

An extension of SVC with the goal of providing additional functionalities and improved coding efficiency is being investigated in the JVT. The current working items are bit-depth scalability, chroma format scalability, and fine-granular fidelity scalability with an improved drift control for low-delay coding.

RESOURCES

The main resources for SVC are the standard text itself, the standardized conformance bit streams, and the reference software. The standard is published by both ITU-T and ISO/IEC as so-called "twin texts" (i.e. they are technically aligned but published independently of each other). Other resources are listed in the "SVC Resources" sidebar.

PRODUCTS

The final draft of the SVC amendment was finalized in July 2007. Since SVC has not been released yet, products are in an early stage of deployment. The first product announcements and SVC implementations are listed in the "SVC Hardware and Software Products" sidebar.

AUTHORS

Heiko Schwarz (heiko.schwarz@hhi.fhg.de) is with the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, Germany. He is a coeditor of the SVC amendment for the H.264/AVC standard.

Mathias Wien (wien@ient.rwth-aachen.de) is with RWTH Aachen University, Germany. He is a coeditor of the SVC amendment for the H.264/AVC standard. SP

dsp APPLICATIONS continued from page 120

REFERENCES

[1] M.J. Wainwright and M.I. Jordan, "Graphical models, exponential families, and variational inference," Depart. Statistics, UC Berkeley, Tech. Rep. 649, Sept. 2003.

[2] J.S. Yedidia, W.T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," in *Exploring Artificial Intelligence in the New Millennium,* G. Lakemeyer and B. Nebel, Eds. San Mateo, CA: Morgan Kaufmann, 2002.

[3] B.J. Frey and N. Jojic, "A comparison of algorithms for inference and learning in probabilistic graphical models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 27, no. 9, pp. 1392–1416, Sept. 2005.

[4] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor graphs and the sum–product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[5] D.J.C. MacKay, *Information Theory, Inference & Learning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[6] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kaufman, 1988.

[7] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient belief propagation for early vision," *Int. J. Comput. Vis.*, vol. 70, no. 1, pp. 41–54, 2006.

[8] 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.

[9] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, no. 1, pp. 7–42, 2002.

[10] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 7, pp. 787–800, July 2003.

[11] E.B. Sudderth, A.T. Ihler, W.T. Freeman, and A.S. Willsky, "Nonparametric belief propagation," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2003, vol. 1, pp. 605–612.

[12] J. Dauwels and H.-A. Loeliger, "Phase estimation by message passing," in *Proc. Int. Conf. Communications*, 2004, pp. 523–527. SP