International Journal of Computer Vision 40(1), 25–47, 2000

Learning Low-Level Vision

WILLIAM T. FREEMAN
Mitsubishi Electric Research Labs., 201 Broadway, Cambridge, MA 02139
Freeman@merl.com

EGON C. PASZTOR
MIT Media Laboratory, E15-385, 20 Ames St., Cambridge, MA, 02139
egon@media.mit.edu

OWEN T. CARMICHAEL
209 Smith Hall, Carnegie-Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213
otc@andrew.cmu.edu

Abstract. We describe a learning-based method for low-level vision problems—estimating scenes from images. We generate a synthetic world of scenes and their corresponding rendered images, modeling their relationships with a Markov network. Bayesian belief propagation allows us to efficiently find a local maximum of the posterior probability for the scene, given an image. We call this approach VISTA—Vision by Image/Scene TrAining.

We apply VISTA to the "super-resolution" problem (estimating high frequency details from a low-resolution image), showing good results. To illustrate the potential breadth of the technique, we also apply it in two other problem domains, both simplified. We learn to distinguish shading from reflectance variations in a single image under particular lighting conditions. For the motion estimation problem in a "blobs world", we show figure/ground discrimination, solution of the aperture problem, and filling-in arising from application of the same probabilistic machinery.

Keywords: vision and learning, belief propagation, low-level vision, super-resolution, shading and reflectance, motion estimation

1. Introduction

We seek machinery for learning low-level vision problems, such as motion analysis, inferring shape and reflectance from a photograph, or extrapolating image detail. For these problems, given *image* data, we want to estimate an underlying *scene* (Fig. 1). The scene quantities to be estimated might be projected object velocities, surface shapes and reflectance patterns, or missing high frequency details. These estimates are important for various tasks in image analysis, database search, and robotics.

Low-level vision problems are typically underconstrained, so Bayesian (Berger, 1985; Knill and Richards, 1996; Szeliski, 1989) and regularization techniques (Poggio et al., 1985) are fundamental. There has been much work and progress (for example, Knill and Richards, 1996; Landy and Movshon, 1991; Horn, 1986), but difficulties remain in working with complex, real images. Typically, prior probabilities or constraints are hypothesized, rather than learned.

A recent research theme has been to study the statistics of natural images. Researchers have related those statistics to properties of the human visual system (Olshausen and Field, 1996; Bell and Sejnowski, 1997; Simoncelli, 1997), or have used statistical characterizations of images to analyse and synthesize realistic textures (Heeger and Bergen, 1995; DeBonet and Viola, 1998; Zhu and Mumford, 1997; Simoncelli, 1997). These methods may help us understand the early stages

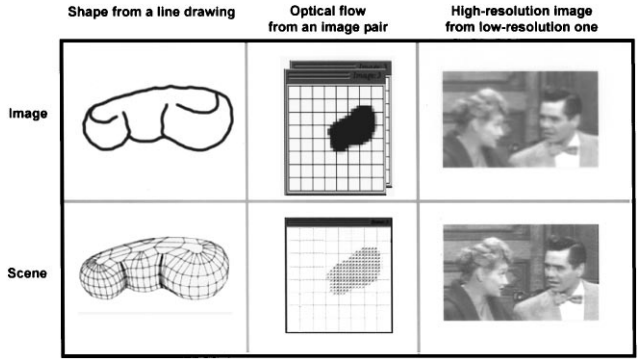


Figure 1. Example low-level vision problems. For given "image" information, we want to estimate an underlying "scene" that created it (idealized scene estimates shown).

of representation and processing, but unfortunately, they don't address how a visual system might *interpret* images, i.e., estimate the underlying scene.

We want to combine the two research themes of scene estimation and statistical learning. We study the statistical properties of a synthetically generated world of images labelled with their underlying scenes, to learn how to infer scenes from images. Our prior probabilities and rendering models can then be rich ones, learned from the training data.

Several researchers have applied related learning approaches to low-level vision problems, but restricted themselves to linear models (Kersten et al., 1987; Hurlbert and Poggio, 1988), too weak for many applications. Our approach is similar in spirit to relaxation labelling (Rosenfeld et al., 1976; Kittler and Illingworth, 1985), but our Bayesian propagation algorithm is more efficient and we use training data to derive propagation parameters.

We interpret images by modeling the relationship between local regions of images and scenes, and between neighboring local scene regions. The former allows initial scene estimates; the later allows the estimates to propagate. We train from image/scene pairs and apply the Bayesian machinery of graphical models (Pearl, 1988; Binford et al., 1988; Jordan, 1998). We were influenced by the work of Weiss (Weiss, 1997), who pointed out the speed advantage of Bayesian methods over conventional relaxation methods for propagating local measurement information. For a related approach, but with heuristically derived propagation rules, see Saund (1999).

We call our approach VISTA, Vision by Image/Scene TrAining. It is a general machinery that may apply to various vision problems. We illustrate it for estimating missing image details, disambiguating shading from reflectance effects, and estimating motion.

2. Markov Network

For given image data, y, we seek to estimate the underlying scene, x (we omit the vector symbols for notational simplicity). We first calculate the posterior probability, $P(x \mid y) = cP(x, y)$ (the normalization, $c = \frac{1}{P(y)}$, is a constant over x). Under two common loss functions (Berger, 1985), the best scene estimate, $\hat{x}$, is the mean (minimum mean squared error, MMSE) or the mode (maximum a posteriori, MAP) of the posterior probability.

In general, $\hat{x}$ can be difficult to compute without approximations (Knill and Richards, 1996). We make the Markov assumption: we divide both the image and scene into patches, and assign one node of a Markov network (Geman and Geman, 1984; Pearl, 1988; Jordan, 1998) to each patch. We draw the network as nodes connected by lines, which indicate statistical dependencies. Given the variables at intervening nodes, two nodes of a Markov network are statistically independent. We connect each scene patch both to its corresponding image patch and to its spatial neighbors, Fig. 2. For some problems where long-range interactions are important, we add layers of image and scene patches at other spatial scales, connecting scene patches to image patches at the same scale, and to scene patches at neighboring scales and positions. (Unlike Luettgen et al. (1994), this is not a tree because of the connections between spatial neighbors).

The Markov network topology of Fig. 2 implies that knowing the scene at position j: (1) provides all the information about the rendered image there, because x_j has the only link to y_j, and (2) gives information about nearby scenes values, by the links from x_j to nearby scene neighbors. We will call problems with these properties low-level vision problems.

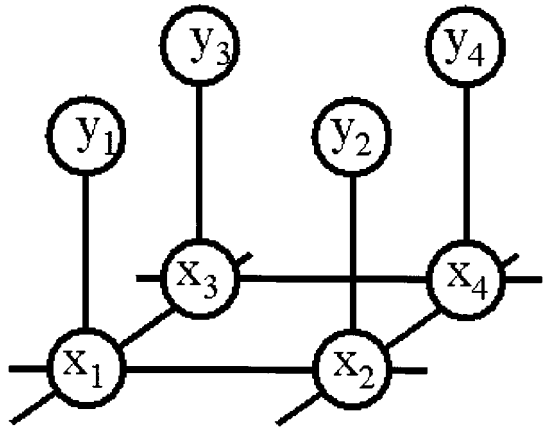


Figure 2. Markov network for vision problems. Each node in the network describes a local patch of image or scene. Observations, y, have underlying scene explanations, x. Lines in the graph indicate statistical dependencies between nodes.

Solving a Markov network involves a *learning* phase, where the parameters of the network connections are learned from training data, and an *inference* phase, when the scene corresponding to particular image data is estimated.

For a Markov random field, the joint probability over the scenes x and images y can be written (Besag 1974; Geman and Geman, 1984; Geiger and Girosi, 1991):

$$P(x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N) = \prod_{(i,j)} \Psi(x_i, x_j) \prod_k \Phi(x_k, y_k), \quad (1)$$

where we have introduced pairwise compatibility functions, Ψ and Φ, which are learned from the training data. (i, j) indicates neighboring nodes i, j and N is the number of image and scene nodes.

We can write the MAP and MMSE estimates for $\hat{x}_j$ by marginalizing (MMSE) or taking the maximum (MAP) over the other variables in the posterior probability. For discrete variables, the marginalization involves summations over the discrete values of the scene variables at each node, indicated by the summations below:

$$\hat{x}_{j\text{MMSE}} = \sum_{x_j} x_j \sum_{\text{all } x_i, i \neq j} \times P(x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N) \quad (2)$$

$$\hat{x}_{j\text{MAP}} = \arg\max_{x_j} \max_{[\text{all } x_i,\, i \neq j]} \times P(x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N). \quad (3)$$

For networks larger than toy examples, Eqs. (2) and (3) are infeasible to evaluate directly because of the high dimensionality of the scene variables over which $P(x_1, x_2, \ldots, x_N, y_1, y_2, \ldots, y_N)$ must be summed or maximized. When the networks form chains or trees, however, we can evaluate the equations.

2.1. *Inference in Networks Without Loops*

For networks without loops, the Markov assumption leads to simple "message-passing" rules for computing the MAP and MMSE estimates during inference (Pearl, 1988; Weiss, 1998; Jordan, 1998). The factorized structure of Eq. (1) allows the marginalization and maximization operators of Eqs. (2) and (3) to pass through Ψ and Φ factors with unrelated arguments. For example, for the network in Fig. 3, substituting Eq. (1) for $P(x, y)$ into Eq. (3) for $\hat{x}_{j\text{MAP}}$ at node 1 gives

$$\hat{x}_{1\text{MAP}} = \arg\max_{x1} \max_{x2} \max_{x3} P(x_1, x_2, x_3, y_1, y_2, y_3) \quad (4)$$

$$= \arg\max_{x1} \max_{x2} \max_{x3} \Phi(x_1, y_1)\Phi(x_2, y_2)\Phi(x_3, y_3) \Psi(x_1, x_2)\Psi(x_2, x_3) \quad (5)$$

$$= \arg\max_{x1} \Phi(x_1, y_1) \max_{x2} \Psi(x_1, x_2)\Phi(x_2, y_2) \max_{x3} \Psi(x_2, x_3)\Phi(x_3, y_3). \quad (6)$$

Each line of Eq. (6) is a local computation involving only one node and its neighbors. The analogous expressions for $x_{2\text{MAP}}$ and $x_{3\text{MAP}}$ also use local calculations. Passing local "messages" between neighbors, as described below, gives an efficient way to compute the MAP estimates.

Assuming a network without loops, Eqs. (3) and (2) can be computed by iterating the following steps (Pearl, 1988; Weiss, 1998; Jordan, 1998). The MAP

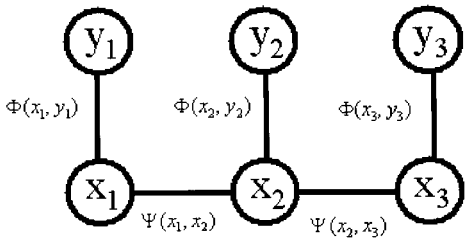


Figure 3. Example Markov network without any loops, used for belief propagation example described in text. The compatibility functions Φ and Ψ are defined below.

estimate at node j is

$$\hat{x}_{j\mathrm{MAP}} = \arg\max_{x_j} \Phi(x_j, y_j) \prod_k M_j^k, \qquad (7)$$

where k runs over all scene node neighbors of node j, and M_j^k is the message from node k to node j. We calculate M_j^k from:

$$M_j^k = \max_{[x_k]} \Psi(x_j, x_k)\Phi(x_k, y_k) \prod_{l \neq j} \tilde{M}_k^l, \qquad (8)$$

where $\tilde{M}_k^l$ is M_k^l from the previous iteration. The initial $\tilde{M}_j^k$'s are set to column vectors of 1's, of the dimensionality of the variable x_j.

To illustrate how these equations are used, we show how Eq. (7) reduces to Eq. (6) for the example of Fig. 3. First, a note about the compatibility matrices, Ψ and Φ. For a given observed image-patch, y_k, the image-scene compatibility function, $\Phi(x_k, y_k)$, is a column vector, indexed by the different possible states of x_k, the scene at node k. The scene-scene compatibility function, $\Psi(x_i, x_j)$, will be a matrix with the different possible states of x_i and x_j, the scenes at nodes i and j, indexing the rows and columns. Because the initial messages are 1's, at the first iteration, all the messages in the network are:

$$M_1^2 = \max_{x_2} \Psi(x_1, x_2)\Phi(x_2, y_2) \qquad (9)$$

$$M_2^3 = \max_{x_3} \Psi(x_2, x_3)\Phi(x_3, y_3) \qquad (10)$$

$$M_2^1 = \max_{x_1} \Psi(x_2, x_1)\Phi(x_1, y_1) \qquad (11)$$

$$M_3^2 = \max_{x_2} \Psi(x_3, x_2)\Phi(x_2, y_2). \qquad (12)$$

The second iteration uses the messages above as the $\tilde{M}$ variables in Eq. (8):

$$M_1^2 = \max_{x_2} \Psi(x_1, x_2)\Phi(x_2, y_2)\tilde{M}_2^3 \qquad (13)$$

$$M_2^3 = \max_{x_3} \Psi(x_2, x_3)\Phi(x_3, y_3) \qquad (14)$$

$$M_3^2 = \max_{x_2} \Psi(x_3, x_2)\Phi(x_2, y_2)\tilde{M}_2^1 \qquad (15)$$

$$M_2^1 = \max_{x_1} \Psi(x_2, x_1)\Phi(x_1, y_1). \qquad (16)$$

Substituting M_2^3 of Eq. (10) for $\tilde{M}_2^3$ in Eq. (13) gives

$$M_1^2 = \max_{x_2} \Psi(x_1, x_2)\Phi(x_2, y_2) \times \max_{x_3} \Psi(x_2, x_3)\Phi(x_3, y_3). \qquad (17)$$

For this example, the messages don't change in subsequent iterations. We substitute the final messages into Eq. (7) to compute the MAP estimates, for example,

$$\hat{x}_{1\mathrm{MAP}} = \arg\max_{x_1} \Phi(x_1, y_1) M_1^2. \qquad (18)$$

Substituting Eq. (17), the converged message value for M_1^2, in Eq. (18) above gives precisely Eq. (6) for $x_{1\mathrm{MAP}}$. The exact MAP estimates for x_2 and x_3 are found analogously.

It can be shown (Pearl, 1988; Weiss, 1988; Jordan, 1998) that after at most one global iteration of Eq. (8) for each node in the network, Eq. (7) gives the desired optimal estimate, $\hat{x}_{j_{\mathrm{MAP}}}$, at each node j.

The MMSE estimate, Eq. (3), has analogous formulae, with the $\max_{x_k}$ of Eq. (8) replaced by $\sum_{x_k}$, and $\arg\max_{x_j}$ of Eq. (7) replaced by $\sum_{x_j} x_j$. For Markov networks without loops, these propagation rules are equivalent to standard Bayesian inference methods, such as the Kalman filter and the forward-backward algorithm for Hidden Markov Models (Pearl, 1988; Luettgen et al., 1994; Weiss, 1997; Smyth et al., 1997; Frey, 1998; Jordan, 1998).

A second factorization of the joint probability can also be used instead of Eq. (1), although it is only valid for chains or trees, while Eq. (1) is valid for general Markov networks. This is a the chain rule factorization of the joint probability, similar to Pearl (1988). For Fig. 3, using the Markov properties, we can write

$$\begin{aligned} &P(x_1, y_1, x_2, y_2, x_3, y_3) \\ &\quad = P(x_1)P(y_1 \mid x_1)P(x_2 \mid x_1) \\ &\qquad \times P(y_2 \mid x_2)P(x_3 \mid x_2)P(y_3 \mid x_3). \end{aligned} \qquad (19)$$

Following the same reasoning as in Eqs. (4)–(6), this factorization leads to the following MAP update and estimation rules:

$$M_j^k = \max_{x_k} P(x_k \mid x_j)P(y_k \mid x_k) \prod_{l \neq j} \tilde{M}_k^l, \qquad (20)$$

$$x_{j\mathrm{MAP}} = \arg\max_{x_j} P(x_j)P(y_j \mid x_j) \prod_k M_j^k. \qquad (21)$$

where k runs over all scene node neighbors of node j. While the expression for the joint probability does

not generalize to a network with loops, we nonetheless found good results for some problems using these update rules (for Section 5 and much of Section 3).

2.2. Networks with Loops

For a network with loops, Eqs. (2) and (3) do not factor into local calculations as in Eq. (6). Finding exact MAP or MMSE values for a Markov network with loops can be computationally prohibitive. Researchers have proposed a variety of approximations (Geman and Geman, 1984; Geiger and Girosi, 1991; Jordan, 1998). Strong empirical results in "Turbo codes" (Kschischang and Frey, 1998; McEliece et al., 1998), layered image analysis (Frey, 2000) and recent theoretical work (Weiss, 1998; Weiss and Freeman, 1999; Yedidia et al., 2000) provide support for a very simple approximation: applying the propagation rules of Eqs. (8) and (7) *even in the network with loops*. Table 1 summarizes results from Weiss and Freeman (1999): (1) for Gaussian processes, the MMSE propagation scheme can converge only to the true posterior means. (2) Even for non-Gaussian processes, if the MAP propagation scheme converges, it finds at least a local maximum of the true posterior probability. Furthermore, this condition of local optimality for the converged solution of the MAP algorithm is a strong one. For every subset of nodes of the network which form a tree, if the remaining network nodes are constrained to their converged values, the values of the sub-tree's nodes found by the MAP algorithm are the *global* maximum over that tree's nodes (Weiss and Freeman, 2000). Yedidia et al. (2000) show that the MMSE belief propagation equations are equivalent to the stationarity conditions for the Bethe approximation to the "free energy" of the network. These experimental and theoretical results motivate applying the belief propagation rules of Eqs. (8) and (7) even in a Markov network with loops. (There is not the corresponding theoretical justification for applying Eqs. (20) and (21) in a network with loops; we rely on experiment).

Table 1. Summary of results from Weiss and Freeman (1999) regarding belief propagation results after convergence.

Belief propagation algorithm	Network topology	
	No loops	Arbitrary topology
MMSE rules	MMSE, correct posterior marginal probs.	For Gaussians, correct means, wrong covs.
MAP rules	MAP estimate	Local max. of posterior, even for non-Gaussians.

2.3. Representation

We need to chose a representation for the image and scene variables. The images and scenes are arrays of vector valued pixels, indicating, for example, color image intensities or surface height and reflectance information. We divide these into patches. For both compression and generalization, we use principle components analysis (PCA) to find a set of lower dimensional basis functions for the patches of image and scene pixels. We measure distances in this representation using a Euclidean norm, unless otherwise stated.

We also need to pick a form for the compatibility functions $\Phi(x_j, y_j)$ and $\Psi(x_j, x_k)$ in Eqs. (7) and (8), as well as the messages, M_j^k. One could represent those functions as Gaussian mixtures (Freeman and Pasztor, 1999) over the joint spaces $x_j \times y_j$ and $x_j \times x_k$; however multiplications of the Gaussian mixtures is cumbersome, requiring repeated pruning to restore the product Gaussian mixtures to a manageable number of Gaussians.

We prefer a discrete representation. The most straight-forward approach would be to evenly sample all possible states of each image and scene variable at each patch. Unfortunately, for reasonably sized patches, the scene and image variables need to be of a high enough dimensionality that an evenly-spaced discrete sampling of the entire high dimensional space is not feasible.

To address that, we evaluate $\Phi(x_j, y_j)$ and $\Psi(x_j, x_k)$ only at a restricted set of discrete points, a subset of our training set. (For other sample-based representations see Isard and Blake (1996), DeBonet and Viola (1998)). Our final MAP (or MMSE) estimates will be maxima over (or weights on) a subset of training samples. In all our examples, we used the MAP estimate. The estimated scene at each patch was always be some example from the training set.

At each node we collect a set of 10 or 20 "scene candidates" from the training data which have image data closely matching the observation, or local evidence, at that node. (We think of these as a "line-up of suspects", as in a police line-up.) We will evaluate probabilities only at those scene values. This simplification focuses the computational effort on only those scenes

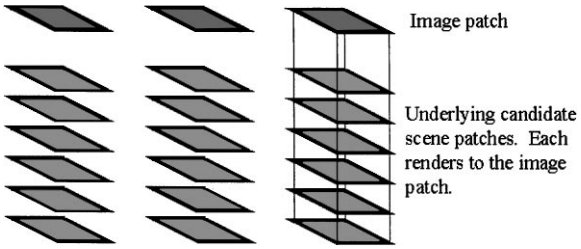


Figure 4. Showing the problem to be solved by Bayesian belief propagation. We break the observed image data into patches (top row). For each image patch, we gather a collection of candidate scene patches from the training database. Each scene can explain the observed image patch, some better than others. Neighboring image patches have their own sets of scene candidates (in each column). We must find at each location the scene candidate which both explains the local image data well, and is compatible with the best scene candidates at the neighboring locations. Bayesian belief propagation gives an approximate solution to this problem.

which render to the observed image data. The propagation algorithms, Eqs. (7) and (8) or Eqs. (21) and (20), become matrix operations involving relatively small vectors and matrices. Figure 4 shows symbolically the image data and scene candidates.

2.4. Learning the Compatibility Functions

We want to learn from our training data the compatibility functions relating neighboring nodes of the Markov network. We have explored two different approaches which give comparable results for our problems.

The first method uses the message-passing rules of Eqs. (21) and (20), based on the joint probability factorization which is not valid for a network with loops. So in using these update rules, we are effectively ignoring the presence of loops in both learning and inference. From the training data, we fit mixtures of Gaussians to the joint probabilities $P(y_j, x_j)$ and $P(x_k, x_j)$, for neighboring nodes j and k. We evaluate $P(x_k^l \mid x_j^m) = \frac{P(x_k^l, x_j^m)}{P(x_j^m)}$ at each of the scene candidates x_k^l (indexed by l) at node k and at each candidates x_j^m (indexed by m) at node j, giving a matrix of rows indexed by l and columns indexed by m. For a given image observation y_k at patch k, $P(y_k \mid x_k^l)$ becomes a column vector indexed by each scene candidate, l. We used these quantites in Eqs. (20) and (21) for the results shown in Sections 3 and 5, except for Figs. 14–16.

More properly, rather then using the conditional probabilities of Eqs. (21) and (20), Iterative Proportional Fitting (e.g., Smyth et al., 1997) should be used to iteratively modify the compatibility functions of Eq. (1) and Eqs. (7) and (8) until the empirically measured marginal statistics agree with those predicted by the model, Eq. (1). However, for the problems presented here, we found good results using the method described above.

The second method we used relied on the proper probability factorization for networks with loops, Eq. (1), but used a simple way to find the compatibility functions. We spaced the scene patches so that they *overlap* and used the scene patches themselves to estimate the compatibilities $\Psi(x_j, x_k)$ between neighbors. Let k and j be two neighboring scene patches. Let d_{jk}^l be a vector of the pixels of the lth possible candidate for scene patch x_k which lie in the overlap region with patch j. Likewise, let d_{kj}^m be the values of the pixels (in correspondence with those of d_{jk}^l) of the mth candidate for patch x_j which overlap patch k see Fig. 5. We say that scene candidates x_k^l (candidate l at node k) and x_j^m are compatible with each other if the pixels in their regions of overlap agree. We assume that the image and scene training samples differ from the "ideal" training samples by Gaussian noise of covariance σ_i and σ_s, respectively. Those covariance values are parameters of the algorithm. We then define the compatibility matrix between scene nodes k and j as

$$\Psi\left(x_k^l, x_j^m\right) = \exp^{-|d_{jk}^l - d_{kj}^m|^2/2\sigma_s^2} \tag{22}$$

The rows and columns of the compatibility matrix $\Psi(x_k^l, x_j^m)$ are indexed by l and m, the scene candidates at each node, at nodes j and k.

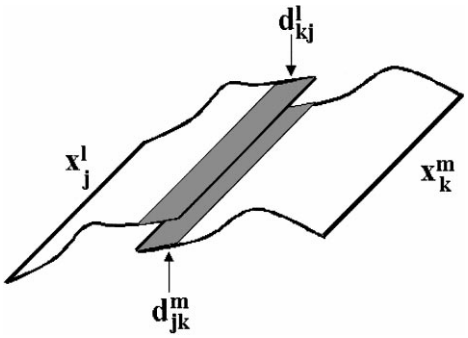


Figure 5. The compatibility between candidate scene explanations at neighboring nodes is determined by their values in their region of overlap. Let d_{kj}^l be the pixels of the lth scene candidate of patch j in the overlap region between patches j and k, and let d_{jk}^m be the (corresponding) pixels of the mth scene candidate belonging to patch k, next to patch j. Then the elements of the compatibility matrix between scene nodes j and k, $\Phi(x_j^l, x_k^m)$ (a matrix indexed by l and m), are Gaussians in $|d_{kj}^l - d_{kj}^m|$.

Note, this form for the compatibility matrix between scene nodes is not a constraint on the spatial smoothness of the scene patches; those can be as rough as the PCA representation of each patch can describe. It is a "uniqueness" constraint, requiring that the pixels in the region of overlap between patches have only one value.

We say that a scene candidate x_k^l is compatible with an observed image patch y_o if the image patch, y_k^l, associated with the scene candidate x_k^l in the training database matches y_o. It won't exactly match, so again we assume "noisy" training data and define the compatibility

$$\Phi\left(x_k^l, y_k\right) = \exp^{-|y_k^l - y_o|^2/2\sigma_i^2}. \tag{23}$$

We set σ_i to allow roughly 10 samples at each node to be within two standard deviations of the observed image patches, and set σ_s to allow roughly 5 or 10 matrix transitions to be appreciably different than zero. This sample-based method was used for the results of Section 4, and for Figs. 14–16.

It could be the case that two particular scene patches would never be next to each other, even though their pixel values agreed perfectly in their region of common support. The Gaussian mixture method would assign a low compatibility to those two scene patches abutting, while the sample-based method would assign them a high compatibility. However, the sample-based method is easier to work with and assumes the proper form for the posterior probability of a Markov network, Eq. (1).

Once we have specified the representation and the compatibility functions, we are ready to apply VISTA to vision problems.

3. Super-Resolution

For the super-resolution problem, the input *image* is a low-resolution image. The *scene* to be estimated is the high resolution version of the same image. (Note this is different than another problem sometimes called super-resolution, that of estimating a single high resolution image from multiple low-resolution ones). A good solution to the super-resolution problem would allow pixel-based images to be handled in an almost resolution-independent manner. Applications could include enlargment of digital or film photographs, upconversion of video from NTSC format to HDTV, or image compression.

At first, the task may seem impossible—the high resolution data is missing. However, we can visually identify edges in the low-resolution image that we know should remain sharp at the next resolution level. Furthermore, the successes of recent texture synthesis methods (Heeger and Bergen, 1995; DeBonet and Viola, 1998; Zhu and Mumford, 1997; Simoncelli, 1997), gives us hope to handle textured areas well, too.

Others (Schultz and Stevenson, 1994) have used a Bayesian method for super-resolution, hypothesizing the prior probability. In contrast, the VISTA approach learns the relationship between sharp and blurred images from training examples, and achieves better results. Among non-Bayesian methods for super-resolution, the fractal image representation used in compression (Polvere, 1998) (Fig. 13(c)) allows zooming, although its image generation model will not hold for all images.[1] Selecting the nearest neighbor from training data (Pentland and Horowitz, 1993) (Fig. 9(a)) ignores important spatial consistency constraints.

We apply VISTA to this problem as follows. By blurring and downsampling sharp images, we construct a training set of sharp and blurred image pairs. We linearly interpolate each blurred image back up to the original sampling resolution, to form an input *image*. The *scene* to be estimated is the high frequency detail removed by that process from the original sharp image, Fig. 7(a) and (b).

We employ two pre-processing steps in order to increase the efficiency of the training set. Each step exploits an assumption about the nature of images. First, we assume that images are Markov over scale (Luettgen et al., 1994) in a *bandpass* image representation, such as a Laplacian pyramid image decomposition (Burt and Adelson, 1983). Let H be the high-frequency pixel values, and M be the values of the next-highest spatial frequency band, which we will call the mid-frequency band, and L be the pixel values of all lower spatial frequencies in the image. We assume

Figure 6. Example images from a training set of 80 images from two Corel database categories: African grazing animals, and urban skylines. Sharp and blurred versions of these images were the training set for the test image of Figs. 9 and 10.

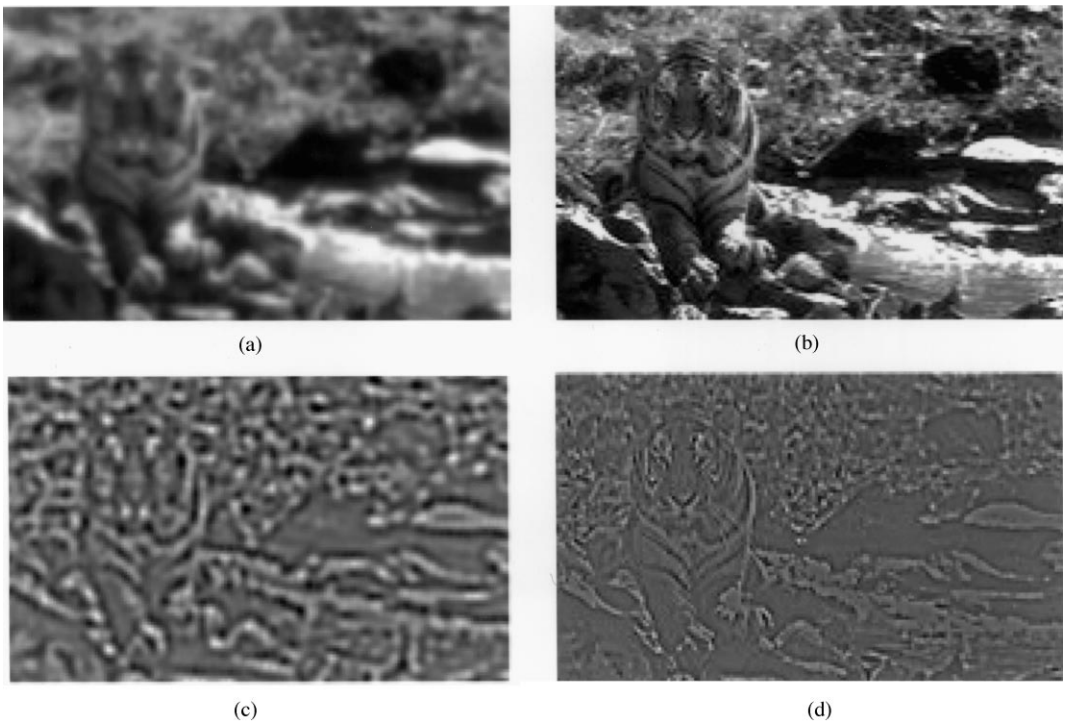


Figure 7. We want to estimate (b) from (a). The original image, (b) is blurred, subsampled, then interpolated back up to the original sampling rate to form (a). All images shown are at 170 × 102 resolution. The missing high frequency detail, (b) minus (a), is the "scene" to be estimated, (d) (this is the first level of a Laplacian pyramid (Burt and Adelson, 1983)). Two image processing steps are taken for efficiency: the low frequencies of (a) are removed to form the input bandpassed "image". We contrast normalize the image and scene by the local contrast of the input bandpassed image, yielding (c) and (d).

that highest resolution frequency band is conditionally independent of the lower frequency bands, given the second highest resolution frequency band:

$$P(H \mid M, L) = P(H \mid M). \tag{24}$$

Based on this assumption, to predict the highest frequency band, we will only examine the mid-frequency band, M, not all lower frequency bands of the image. This greatly reduces the variability we have to store in our training data, collapsing the training data for all possible low-frequency values into one value, dependent only on the mid-band image.

Second, we assume that the statistical relationships between image bands are independent of image contrast, apart from a multiplicative scaling. By taking the absolute value of the mid-frequency band, and blurring it, we form a "local contrast" image, which we use to normalize both the mid- and high-frequency bands. We make the training set from the contrast normalized mid- and high-frequency bands, shown in Fig. 7(c) and (d). This saves having to replicate the training set over all possible values of image contrast, and is a very simplified model of the contrast normalization which may take place in the mammalian visual system (Carandini and Heeger, 1994). We undo this normalization after estimating the scene. The functional forms of the filters used and the contrast normalization are given in the Appendix.

We break the image and scene into local patches. The choice of patch size is a compromise between two extremes. If the image patch size is too small, then each local image patch would give very little information for estimating the underlying scene variable. The Markov network model of patches only connected to their nearest neighbors would break down. However, the training database would be easy to store. On the other hand, a large patch size would disambiguate the underlying scene variables, but it would be prohibitive to learn the relationship between local image and scene patches. That storage requirement grows exponentially with the dimensionality of the image and scene patches. As a compromise, we seek an image and scene patch size which is big enough to give some useful information

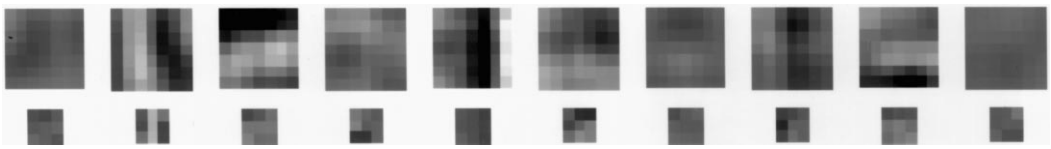


Figure 8. Some training data samples for super-resolution problem. The large squares are the *image* data (mid-frequency data). The small squares below them are the corresponding *scene* data (high-frequency data).

about the underlying scene, yet is small enough to allow learning the relationship between image and scene. We then rely on belief propagation to propagate local evidence across space.

We first describe our results using the gaussian mixtures method, employing Eqs. (20) and (21). We used 7×7 and 3×3 pixel patches, Fig. 8, from the training images and scenes, respectively. These were center-aligned, so that the image patch centered at pixels (i, j) covered all pixels $(i \pm 3, j \pm 3)$ and the corresponding scene patch covered all pixels $(i \pm 1, j \pm 1)$. Applying Principal Components Analysis (PCA) (Bishop, 1995) to the training set, we summarized each 3-color patch of image or scene by a 9-d vector. From 40,000 image/scene pair samples, we fit 15 cluster Gaussian mixtures to the observed joint probabilities $P(x_k, x_j)$ of neighboring scene patches k, j, assuming spatial translation invariance. One Gaussian mixture described the joint statistics of horizontal neighbors, and one described the statistics of vertical neighbors. We also fit Gaussian mixtures to the prior probability of a scene patch, $P(x_j)$, and the joint probability of image-scene pairs, $P(y_k, x_k)$, again assuming spatial translation invariance.

Given a new image, not in the training set, from which to infer the high frequency scene, we found the 10 training samples closest to the image data at each node (patch). The 10 corresponding scenes are the candidates for that node.

From the fit densities, we could evaluate the conditional probabilities used in the message update equation, Eq. (20): $P(x_k \mid x_j)$ and $P(y_k \mid x_k)$. We evaluated these conditional probabilities at the 10 candidate scene points at each node and at all possible combination of scene candidates (10×10) between neighboring nodes. For storage efficiency, we pruned frequently occurring image/scene pairs from the training set, based on a squared error similarity criterion. We propagated the probabilities by Eq. (20), and read-out the maximum probability solution by Eq. (21). We found experimentally that the reconstructed image retained more visually pleasing high frequency structure when we used a "maximum likelihood" readout of the estimated scene from Eq. (21), setting the prior probability term $P(x_j)$ to one.

To process Fig. 10(a), we used a training set of 80 images from two Corel database categories: African grazing animals, and urban skylines (Fig. 6). For reference, Fig. 9(a) shows the nearest neighbor solution, at each node using the scene corresponding to the closest image sample in the training set. Many different scene patches can explain each image patch, and the nearest neighbor solution is very choppy. Figures 9(b), (c) and (d) show the first 3 iterations of MAP belief propagation. The spatial consistency imposed by the belief propagation finds plausible and consistent high frequencies for the tiger image from the candidate scenes.

Figure 10 shows the result of applying this super-resolution method recursively to zoom two octaves. The algorithm keeps edges sharp and invents plausible textures. Standard cubic spline interpolation, blurrier, is shown for comparison.

Figure 11 explores the algorithm behavior under different training sets. Each training set corresponds to a different set of prior assumptions about typical images. Figure 11(a) is the actual high resolution image (192×232). (b) is the 48×58 resolution input image. (c) is the result of cubic spline interpolation to 192×232 resolution. The edges are blurred. (d) is an example image of a training set composed entirely of random noise images. (g) is the result of using that training set with the Markov network super-resolution algorithm. The algorithm successfully learns that the high resolution images relate to lower resolution ones by adding random noise. Edges are not maintained as sharp because the training set has no sharp edges in it. (e) is a sample from a training set composed of vertically oriented, multi-colored rectangles. Again, the super-resolution algorithm correctly models the structure of the visual world it was trained on, and the high-resolution image (h) shows vertically oriented rectangles everywhere. (f) is an example image from a training set of generic images, none of any teapots. Figure 12(b) shows other examples from that training set. The extrapolated

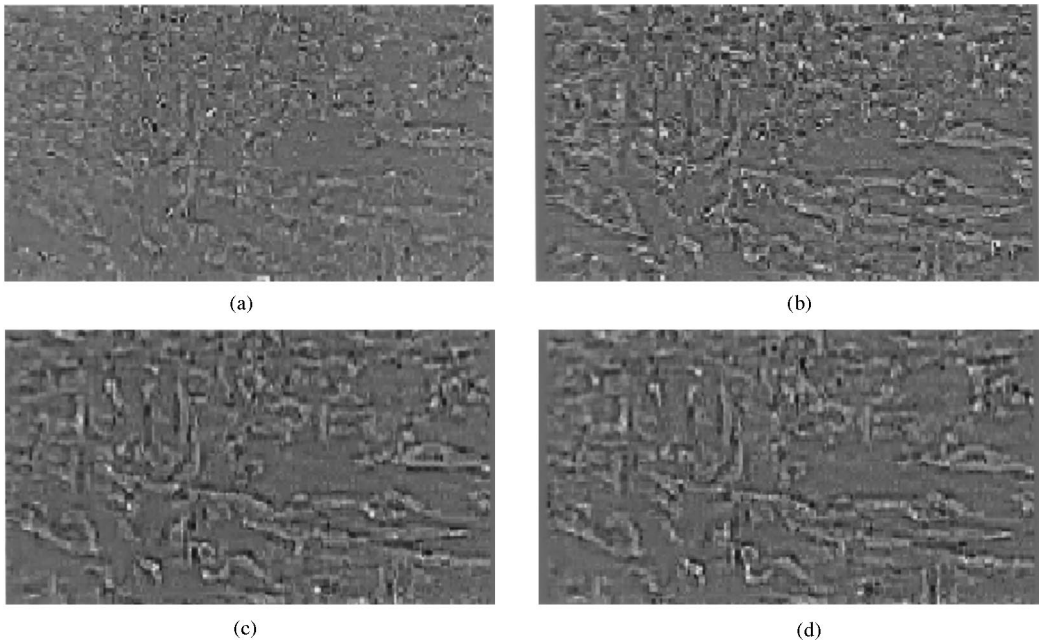


Figure 9. (a) Nearest neighbor solution. The choppiness indicates that many feasible high resolution scenes correspond to a given low resolution image patch. (b), (c), (d): iterations 0, 1, and 3 of Bayesian belief propagation. The initial guess is not the same as the nearest neighbor solution because of mixture model fitting to $P(y \mid x)$. Underlying the most probable guess shown are 9 other scene candidates at each node. 3 iterations of Bayesian belief propagation yields a probable guess for the high resolution scene, consistent with the observed low resolution data, and spatially consistent across scene nodes.

image, (i), maintains sharp edges and makes plausible guesses in the texture regions. The estimated images properly reflect the structure of the training worlds for noise, rectangles, and generic images.

Figure 13 depicts in close-up the interpolation for image (a) using two other training sets, shown in Fig. 12. Figure 13(d) was recursively zoomed up two octaves using the Markov network super-resolution algorithm with an ideal training set of images taken at the same place and same time (but not of the same subject). Figure 13(e) used a generic training set of images. Both estimates look more similar to the true high resolution result (f) than either cubic spline interpolation (b) or zooming by a fractal image compression algorithm (c). Edges are again kept sharp, while plausible texture is synthesized in the hair.

We also applied the method of Eqs. (8) and (7) to the super-resolution problem. This patch-overlap method to find the compatibility functions between nodes was faster to process, and typically gave fewer artifacts. Figures 14–16 were made using this sample-based method. Scene patches were 3×3 pixels, with a 1 pixel overlap between patches. This results in each scene pixel being described by two different scene patches. To output the final image, we averaged the scene results from each pixel where it was described by more than one patch. This method gives results with a silghtly different visual character than the Gaussian mixture method. It has fewer artifacts at edges (note the girl's nose), but is also smoother in regions of image texture.

As Figure 11 shows, the training set influences the super-resolution output. On the assumption that the image is similar to itself over different spatial scales, it is reasonable to try using the image itself, at a lower-resolution, as the training set for zooming up to a higher resolution. Figure 15 shows that that training set gives reasonable results for our common test image. We built a training set from all 90 degree rotations and transpositions of the image from which the 70×70 test image was cropped (top). After zooming up to 280×280 resolution by the patch-overlap version of the Markov network super-resolution algorithm, the results are comparable with the super-resolution results from other training sets.

Figure 16 shows a patch of texture, zoomed up two and four octaves up to 400% and 1600% magnification. (We used the patch overlap method to compute the compatibilities for belief propagation by Eqs. (8)

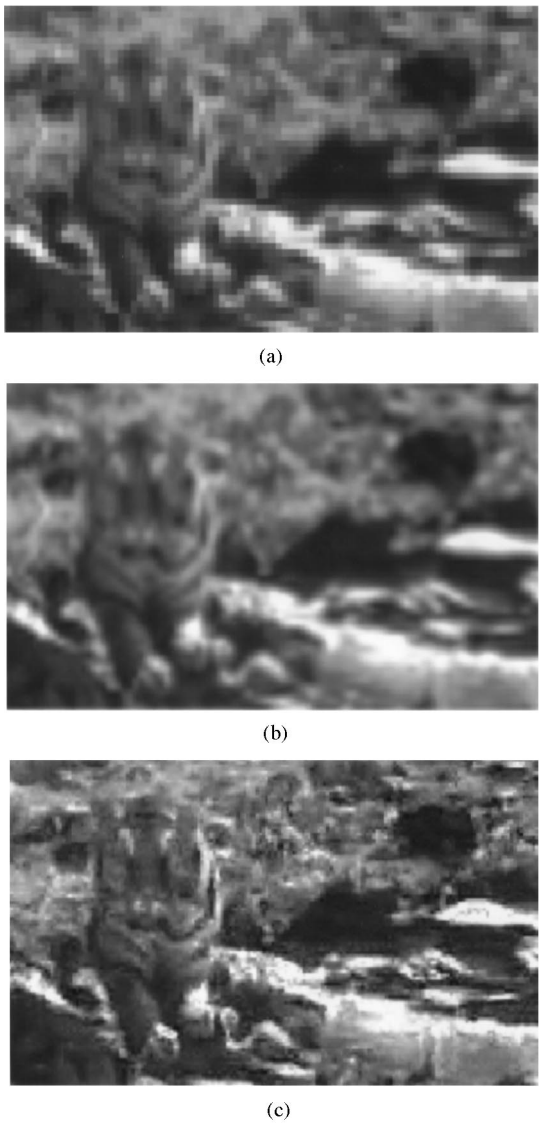


Figure 10. (a) 85 × 51 resolution input. (b) cubic spline interpolation in Adobe Photoshop to 340 × 204. (c) belief propagation in Markov network zoom to 340 × 204, recursively zooming up by one octave twice.

and (7). For comparison, zooming by pixel replication and cubic spline interpolation are shown as well. The algorithm "makes-up" detail which, while almost certainly not correct, is plausible and visually pleasing.

As emphasized by other authors (e.g., Field, 1994), the visual world has much more structure than would images of random collections of pixel values. The results of this section show that we can exploit this structure to estimate missing resolution detail.

4. Shading and Reflectance Estimation

We turn to a second low-level vision application, that of estimating shading and reflectance properties from a single image. Figure 17, left, illustrates the problem, with an image pair due to Adelson (1995). The top image looks like a raised bump, with the intensity variations due to shading effects. The bottom image looks like two crescents drawn on a flat piece of paper, with the intensity variations due to surface reflectance changes. Yet each image has nearly exactly the same intensities everywhere; one is a sheared version of the other. Clearly a local look-up of scene structure from image intensities will not allow us to distinguish the causes of the crescent image or the bump image. Furthermore, while people report consistent interpretations for the crescent and bump images (data from Freeman and Viola, 1998), each image has multiple feasible scene explanations, shown in the middle and right of Fig. 17. The shape explanation for the crescents image requires non-generic alignment of the assumed lighting direction (from the left) with the inferred shape (Freeman, 1994).

While disambiguating shading from reflectance is fundamental to interpreting images by computer, it has received relatively little research attention. Shape-from-shading algorithms typically assume constant or known surface albedo markings (Horn and Brooks, 1989). Sinha and Adelson (1993) have addressed this problem, but in a blocks world with presegmented junctions and regions. Generalization to the world of real images has proved difficult. A Bayesian approach using pixel-based image representations was taken by Freeman and Viola (1998), who derived the likelihood of reflectance from the prior probability penalty required of a shape interpretation of the image. Here we take a more general approach, explicitly solving for the reflectance and shape combination that best explains the image data, using the VISTA approach.

We focus on a simplified version of the problem: we assume just one light direction, and one fixed reflectance function (Lambertian). Generalizing to other light directions involves taking a new training set over a sampling of different light directions. This simplified setting retains the fundamental ambiguity we focus on: how can we distinguish shading from paint?

We apply to this problem domain the same procedure we used for super-resolution. We first generate a training set of image and scene patches. Here the scene consists of two pixel arrays, one describing the

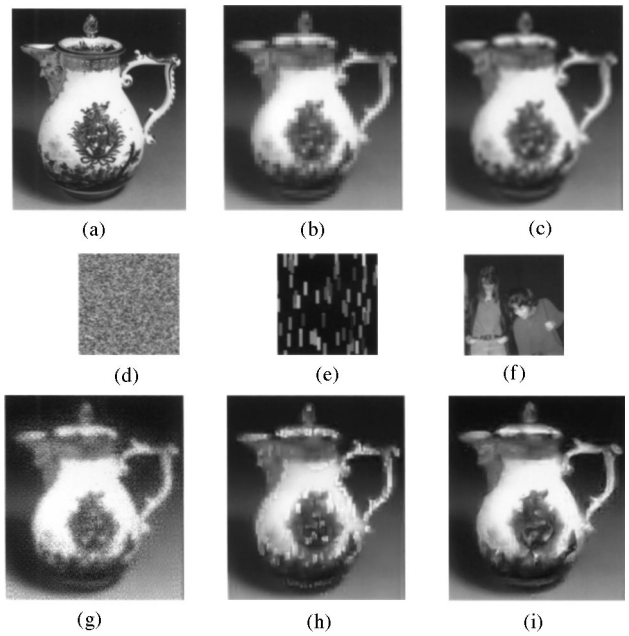


Figure 11. Effect of different training sets on super-resolution outputs. (a), at 192 × 232 resolution, was blurred, and subsampled by 4 in each dimension to yield the low-resolution input, (b), at 48 × 58 resolution. Cubic spline interpolation to full resolution in Adobe Photoshop loses the sharp edges, (c). We recursively zoomed (b) up two factors of two using the Markov network trained on 10 images from 3 different "worlds": (d) random noise, (e) colored rectangles, and (f) a generic collection of photographs. The estimated high resolution images, (g), (h), and (i), respectively, reflect the statistics of each training world.

Figure 12. Sample images from the 10 images in each of the (a) "picnic" and (b) "generic" training sets. Sharp and blurred versions of these images were used to create the training data for Fig. 13(d) and (e). The generic training set was also used for Figs. 14 and 16.

reflectance function and one describing the shape by a range map (where pixel intensities indicate distance from the camera).

Our training set consisted of computer-generated examples of images such as those in Fig. 18. Randomly placed and oriented ellipses were used as either reflectance images on a flat range map, or as range images with a flat reflectance map. At a global scale, which is shape and which is reflectance is perceptually obvious from looking at the rendered images. At a local scale,

Figure 13. (a) Low-resolution input image. (b) Cubic spline 400% zoom in Adobe Photoshop. (c) Zooming luminance by public domain fractal image compression routine (Polvere, 1998), set for maximum image fidelity (chrominance components were zoomed by cubic spline, to avoid color artifacts). Both (c) and (d) are blurry, or have serious artifacts. (d) Markov network reconstruction using a training set of 10 images taken at the same picnic, none of this person. This is the best possible fair training set for this image. (e) Markov network reconstrution using a training set of *generic* photographs, none at this picnic or of this person, and fewer than 50% of people. The two Markov network results show good synthesis of hair and eye details, with few artifacts, but (d) looks slightly better (see brow furrow). Edges and textures seem sharp and plausible. (f) is the true full-resolution image.

however, the images are ambiguous; Fig. 20 shows different scene explanations for a given patch of image data. Both shading and paint scene explanations render to similar image data. We generated 40 such images and their underlying scene explanations at 256×256 spatial resolution.

Next, given a training image, we broke it into patches, Fig. 19. Because long range interactions are important for this problem, we used a multi-scale approach, taking patches at two different spatial scales, of size 8×8 and 16×16 pixels. The image patches were sampled with a spatial offset of 7 and 14 pixels, respectively, ensuring consistent alignment of patches across scale, and a spatial overlap of patches, used in computing the compatibility functions for belief propagation with Eqs. (8) and (7). As in the other problems, each image patch in the Markov network connects to a node describing the underlying scene variables. For this multi-scale model, each scene node connects to its neighbors in both space and in scale.

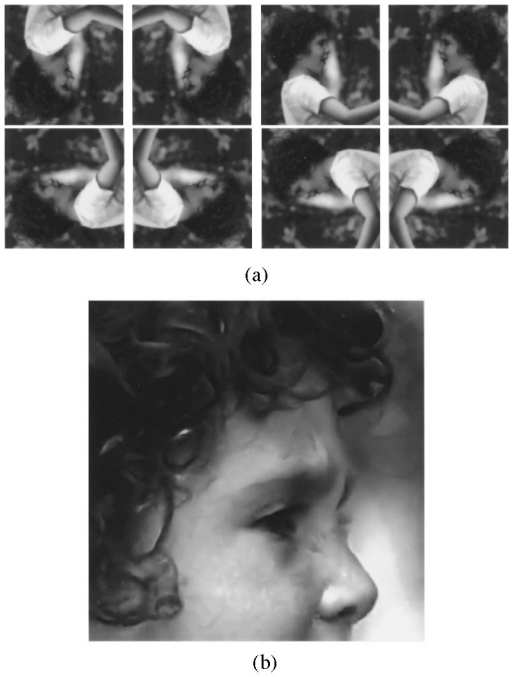
4.1. Selecting Scene Candidates

For each image patch, we must select a set of candidate scene interpretations from the training data. For this problem, we found that the selection of candidates required special care to ensure obtaining a sufficiently diverse set of candidates. The difficulty in selecting candidates is to avoid selecting too many similar ones. We want fidelity to the observed image patch, yet at the same time diversity among the scene explanations. A collection of scene candidates is most useful if at least one of them is within ϵ distance of the correct answer.

We seek to maximize the probability, $\hat{P}$, that at least one candidate x_i^j (the jth scene candidate at node i) in

(a) (b)

Figure 14. Super-resolution results using the patch-overlap method to find the scene patch compatibilities. 280 × 280 super-resolution result, starting from the 70 × 70 sized image of Fig. 13(a). Image was made using the generic training set (with 99,275 image/scene pair samples), and the overlapped patches method of determining the scene-scene compatibility functions. (a) After no iterations of belief propagation. Note the roughness from incompatible neighboring scene candidates. (b) After 10 iterations of belief propagation (although results appeared to converge after 3 or 4 iterations). Texture rendition is slightly worse than results of Gaussian mixture method, Fig. 13, although there appear to be fewer artifacts. The true high resolution scene is given in Fig. 13(f).



(a) (b)

Figure 15. Using a lower-resolution version of the image itself as a training set. As Fig. 11 shows, super-resolution results depend on the training set. It is reasonable to try using the image itself at low resolution to generate examples of high resolution detail. (a) We used images of all 90 degree rotations and transpositions of the uncropped version of Fig. 13(a), resulting in a training set of 72,200 image/scene pairs. Starting from Fig. 13(a), we used VISTA to zoom up two octaves, giving (b), which compares will with Markov network zooms using other training sets, and with the true high resolution image, Fig. 13(f). We used the patch overlap method to compute the compatibilities for belief propagation by Eqs. (8) and (7).

the collection S is within a threshold distance, ϵ of the true scene value, $\hat{x}_i$, given the local observation, y_i, at the ith patch:

$$\hat{P}(S) = \max_{x_i^j \in S} P\left(\left|\hat{x}_i - x_i^j\right| < \epsilon \mid y_i\right). \tag{25}$$

We use a greedy approach to select the set of candidates, S. Assume we have already selected some set of candidates, S_0, and we want to decide which new candidate to add to our selected set to maximize $\hat{P}$. There may be a very probable candidate close to one already in our set. Choosing that candidate would add little to $\hat{P}(S)$, because its region of the scene parameter space within distance ϵ would be already accounted for by the nearby, previously selected candidate.

For a given selection of scene candidates, S_0, the utility of an additional candidate x_i^j is

$$U\left(x_i^j\right) = \int_{|x' - x_i^j| < \epsilon} P(x' \mid y_i)\delta(S_0, x')\, dx', \tag{26}$$

where

$$\delta(S_0, x') = \begin{cases} 1 & \text{if } \left|x' - \bar{x}\right| > \epsilon, \forall \bar{x} \in S_0 \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

Comensurate with our rough initial estimates of the probability that each scene is the correct one, we use a simple approximate criterion to select the best scene candidate to add to S_0. Before any belief propagation, our only estimate of $P(x_i^j \mid y_i)$ is the compatibility function, $c\Phi(x_i^j, y_i)$ (c is a normalization constant). We divide our estimated probability of each scene patch, $c\Phi(x_i^j, y_i)$, by the number of selected scene patches within a distance ϵ of this candidate x_i^j. Thus, we approximate Eq. (26) by

$$\int_{|x' - x_i^j| < \epsilon} P(x' \mid y_i)\delta(S_0, x')\, dx' \approx \frac{c\Phi\left(x_i^j, y_i\right)}{N\left(x_i^j, S_0\right)}, \tag{28}$$

where $N(x_i^*, S_0)$ is the number of scene candidates $\bar{x}$ in S_0 such that $|\bar{x} - x_i^j| < \epsilon$. Then the best scene candidate to add to the set S_0 is

$$x_i^* = \max_j \frac{\Phi\left(x_i^j, y_i\right)}{N\left(x_i^j, S_0\right)}, \tag{29}$$

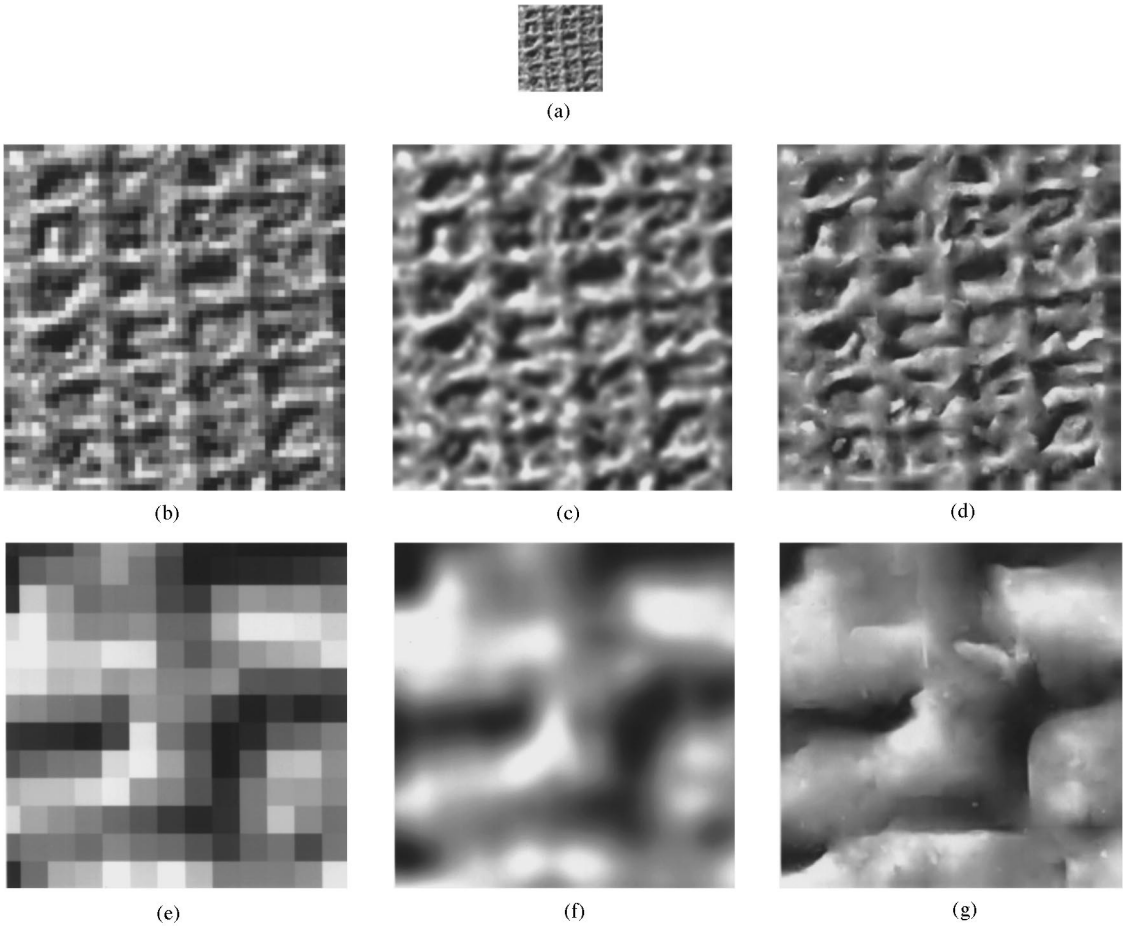


Figure 16. Repeated zooms of a 50 × 50 pixel resolution texture image (a), in 3 different ways. (b) 400% zoom and (e) 1600% zooms, by pixel replication. (c) and (f) by cubic spline interpolation in Adobe Photoshop. (d) and (g) by the VISTA markov network belief propagation approach, using the "generic" training set depicted in Fig. 12 and the patch-overlap method of computing the compatibility matrices between nodes. The high resolution details added by the algorithm in (d) and (g), while almost certainly not veridical, are visually plausible.

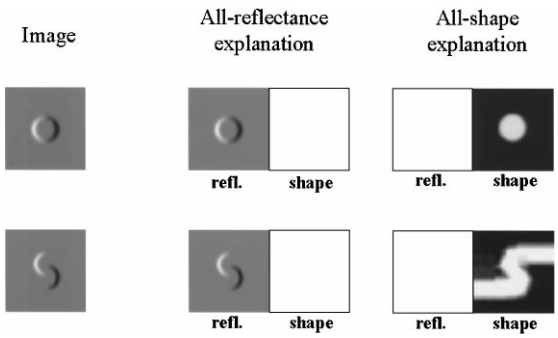


Figure 17. The problem of distinguishing shading from paint. The two images at the left (from Adelson, 1995) are very similar, yet give very different perceptual interpretations. Adding to the difficulty of the problem, each image can, in principle, have multiple different feasible interpretations, shown in the middle and right.

This procedure produces a diverse set of scene patches which are all reasonably good explanations of the observed image patch. Figure 20(a) shows a set of scene candidates selected only based on the distance of their rendered images from the observed image patch. Note there are many similar scene patches. Figure 20(b) shows the set selected using the selection criterion described above. This collection includes a more diverse set of scene explanations, yet each still describes the input image relatively well.

4.2. Compatibility Functions

For this problem, we used the patch overlap method to compute the compatibility functions, Ψ and Φ. In

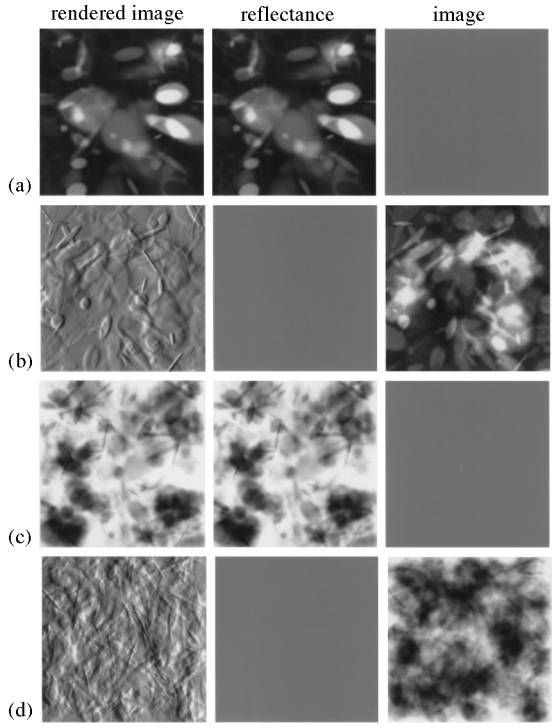


Figure 18. Examples from training set for shading and reflectance disambiguation. Ellipsoids of random orientation, size, and amplitude were added to bitmapped images. These bitmaps were treated either as reflectance images (a and c) or as range maps (b and d), and were used to generate a total of 40 rendered images, combined with the shape and reflectance explanations which generated them.

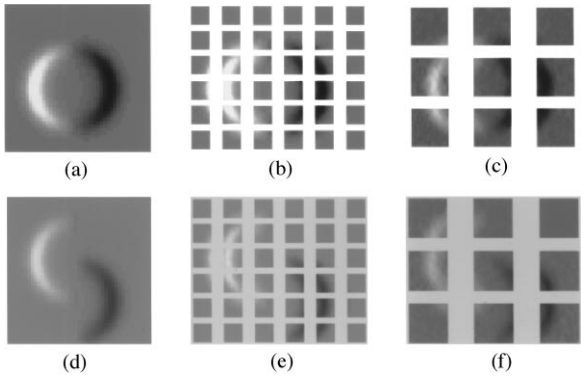


Figure 19. The input images, (a) and (d), are broken into patches at two different spatial scales, (b) and (e), and (c) and (f). In the Markov network, each image patch is connected with a node describing the underlying scene variables. Scene nodes connect to their neighbors in both space and in scale.

computing the distance between the pixels of two scenes, we scaled the reflectance distances by 0.5 relative to the shape differences, in order to put them on a comensurate scale relative to their amplitude ranges. To obtain robustness to outliers, we used an L1-norm (instead of the L2-norm) for distance measurements for both images and scenes. Figure 21 shows a typical compatibility matrix.

To compute the compatibilities between neighboring patches at different scales, we first interpolated the lower-resolution patch by a factor of 2 in each dimension so that it had the same sampling rate as the high resolution patch. Letting d^l_{jk} be the pixels of the lth candidate in the high resolution patch k, and d^m_{kj} be the pixels of the mth candidate in the interpolated low-resolution patch j, we take as the compatibility,

$$\Psi\left(x^l_k, x^m_j\right) = \exp^{-|d^l_{jk} - d^m_{kj}|^2/2\sigma_s^2}, \tag{30}$$

where we scale σ_s to give the same per pixel variance as for the compatibility function between patches at the same scale. The compatibility function $\Psi(x^l_k, x^m_j)$ is different between each pair of nodes k and j, and is indexed by the scene candidate indices at each node, l and m.

A reflectance explanation is feasible for any image, yet we want to allow for a shape explanation, when appropriate. So we add a prior penalty term to $\Phi(x_k, y_k)$, penalizing (in the log domain) by the L1-norm distance of the reflectance from a flat reflectance image. This discourages every image from being explained as reflectance variations on a flat surface.

Having defined the training set (Fig. 18), the network nodes and connections (Fig. 19), the scene candidates (Fig. 20) and the compatibility functions (Fig. 21), we can use the Markov network to infer the underlying scene for a new input image. Figures 22 and 23 show Bayesian belief propagation iterations for the bump and crescent test images of Fig. 17. As expected, the Markov network gives similar initial interpretations for the two images, since on a local scale they are spatially offset versions of each other nearly everywhere. The belief propagation arrives at different most-probable scenes, converging to a shape explanation for the bump image, and converging to a reflectance explanation for the crescent image, finding no good shape to explain it.

In our training samples, there were few non-generic samples, by which we mean ones with significant shape structure made invisible by coincidental alignment with the assumed light direction. (There are a few, however, note Fig. 23). Were such samples more prevalant, as they can be in a much larger training set, we would want to add a term penalizing those non-generic interpretations, as described in Freeman (1994), in order to penalize shape interpretations such as Fig. 17, bottom right.

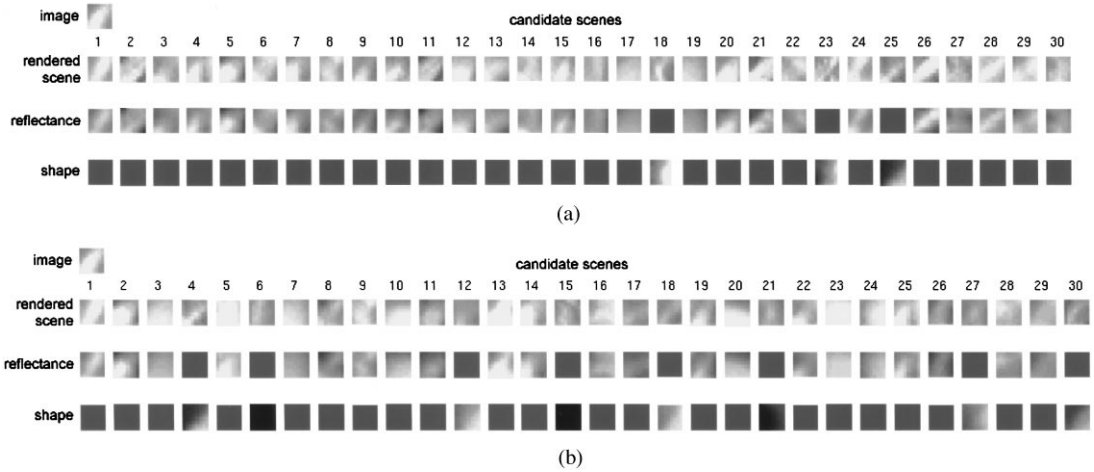


Figure 20. Selection of candidate scenes, without (a) and with (b) the diversity criterion described in the text. A diverse set of candidate explanations leads to better image interpretations.

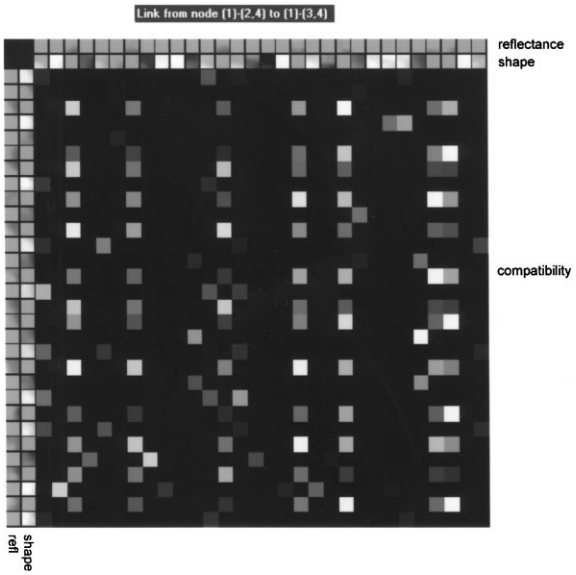


Figure 21. Compatibility function between two nodes (node [3, 4], layer 1 to node [2, 4], layer 1). The reflectance and shape scene candidates at node [3, 4], shown next to the rows, identify each row. The scene candidates for node [2, 4] identify each column. The compatbility matrix value is depicted by the brightness of each square at each row and column intersection.

Figure 24 shows the VISTA approach applied to several other images. The left images of b, d, f, and h show that our training data doesn't fit these images very well. However, the reconstructed scenes are of the appropriate type (reflectance or shading) in each case.

5. Motion Estimation

Finally, we apply VISTA to the problem of motion estimation. The *scene* data to be estimated are the projected velocities of moving objects. The *image* data are two successive image frames. Because we felt long-range interactions were important, we built Gaussian pyramids (e.g., Jahne, 1991) of both image and scene data, connecting patches to nearest neighbors in both scale and position.

Luettgen et al. (1994) applied a related message-passing scheme in a multi-resolution quad-tree network to estimate motion, using Gaussian probabilities. While the network did not contain loops, the authors observed artifacts along quad-tree boundaries, which were artificial statistical boundaries of the model.

For the motion estimation problem, to accurately match the two frames of input images at a patch, the training data needs to contain essentially all possible local image patches cross all possible image motions, which can be a prohibitively large set. In other work (Freeman, et al., 2000), we have applied the belief propagation method to estimate the motion of real images, but used a brightness constancy assumption to generate candidate scene interpretations for each image patch. Here, we enumerate all possible observed input images, but we restrict ourselves to a synthetic world of moving constant intensity blobs, of random intensities and shapes, in order to use the same learning machinery for this problem as we did for the previous two.

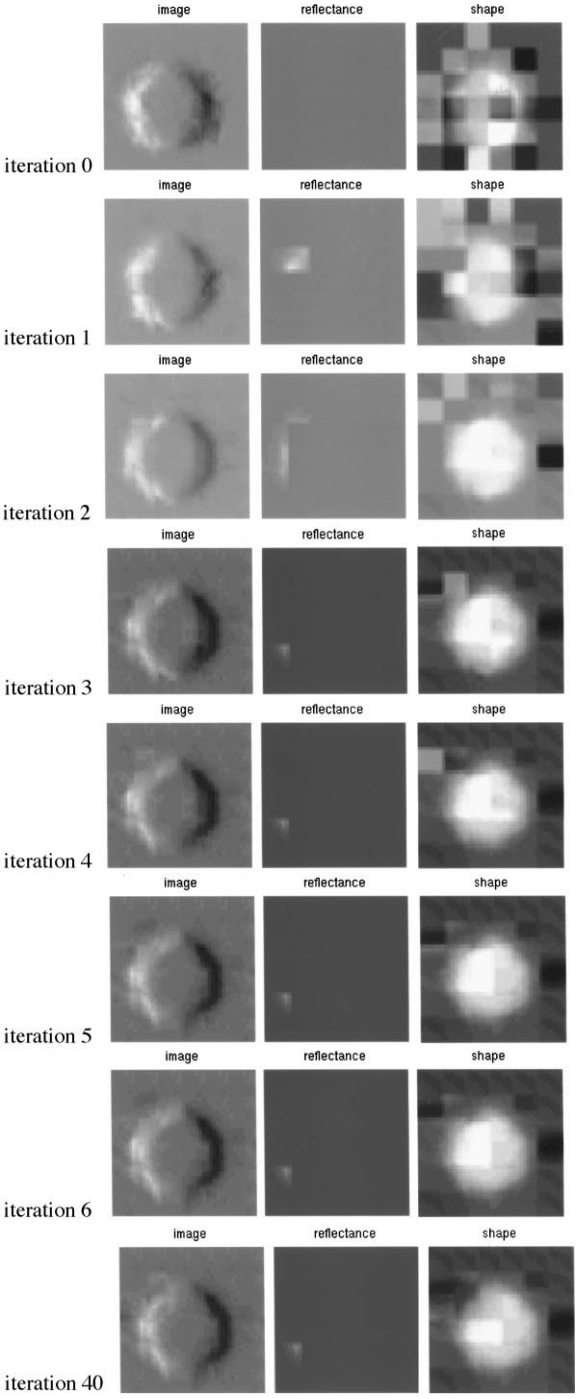


Figure 22. Iterations of the belief propagation for shading/reflectance determination for bump image. The left-most column shows the image rendered from each of the selected candidate scenes. Since each scene candidate was selected to explain the observed image, the left column stays nearly constant over the different choices for scene explanations. After 5 or 6 iterations, the scene estimate makes only small changes (compare with iteration 40).

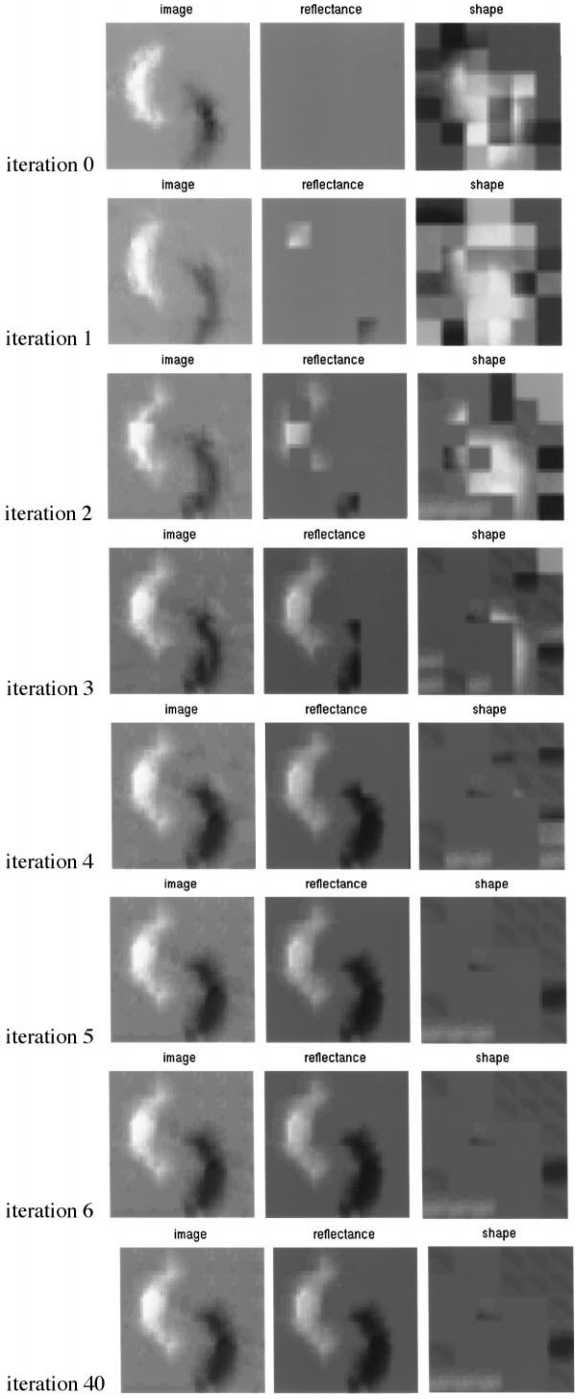


Figure 23. Initial iterations and final solution for crescent problem. The reconstructed shape has a few samples with non-generic shapes relative to the assumed lighting direction, yielding shape structures invisible to the rendered image. The initial scene guess, based on local information alone, is similar to that for the bump image of Fig. 22, but after several iterations of belief propagation, the reflectance explanation becomes more probable.

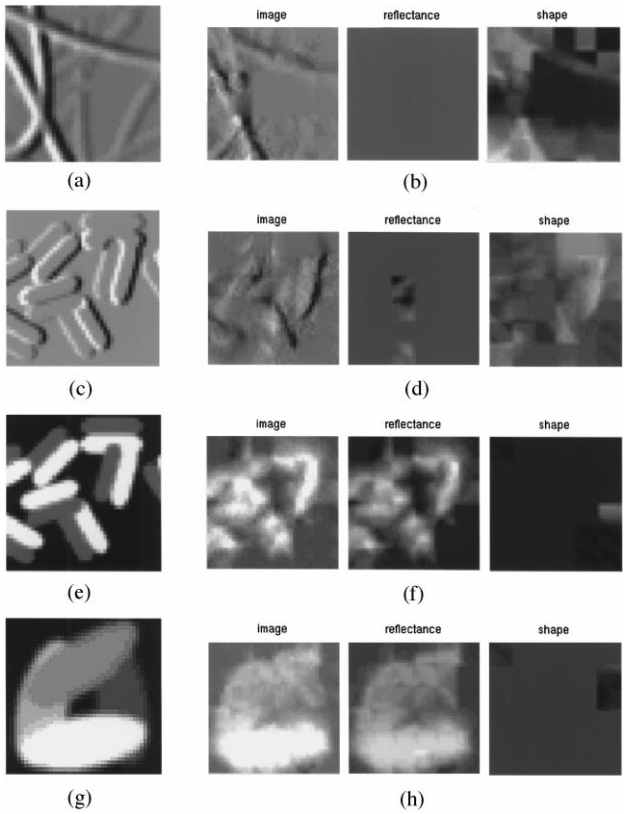


Figure 24. Several images from a database (Freeman and Viola, 1998) of images labelled by naive observers as being caused by shading effects (a, c) or reflectance effects (e, g). The algorithm interpretation agrees with the appearance, and labelling by the subjects. The rendered images of the scene interpretations are not especially faithful to the input images, showing that the training data, depicted in Fig. 18, is not a good match for these images. However, in each case, the scene interpretation is generally correct.

We wrote a tree-structured vector quantizer, to code 4 by 4 pixel by 2 frame blocks of image data for each pyramid level into one of 300 codes for each level, and likewise for scene patches.

During training, we presented approximately 200,000 examples of irregularly shaped moving blobs of a contrast with the background randomized to one of 4 values. For this vector quantized representation, we used co-occurance histograms to measure the joint probabilities of neighboring scene vectors and of image/scene pairs. From those joint probabilities, we calculated the conditional probabilities used in the message passing and belief update rules of Eqs. (21) and (20), see Freeman and Pasztor (1999).

Figure 27 shows six iterations of the inference algorithm as it converges to a good estimate for the underlying scene velocities. For this problem with this training data, the machinery leads to figure/ground segmentation, aperture problem constraint propagation, and filling-in (see caption). The resulting inferred velocities are correct within the accuracy of the vector quantized representation.

6. Discussion

A limitation of the VISTA approach is that one must find a set of candidate scene patches for any given input image patch. In the implementations of this paper (Freeman, et al., 2000), we relied on a training set which enumerated a coarse sampling of all possible input patch values.

We illustrated two approaches that allow this enumeration to be successful. One is to allow only a restricted class of input images. The moving blobs were such a restricted visual world. The shading/reflectance images were also restricted, in that they did not include occluding edges and other features. The second

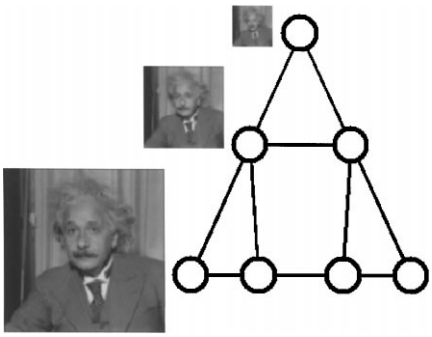


Figure 25. Schematic illustration of multi-scale representation used for motion analysis problem. The image data is presented to the Markov network at multiple resolutions. Each scene node (representing a patch of velocity data at some resolution and position) connects with its neighbors both in space and across scale. Each scene node also has connections (not shown) with image data at the corresponding position and scale.

approach is to pre-process the input images to remove extraneous complexity. This was the approach we used for the super-resolution problem. The image patches were both band-pass filtered and contrast normalized, which allowed adequate fitting of the natural images with reasonably sized training sets.

7. Summary

We described an approach we call VISTA—Vision by Image/Scene TrAining. One specifies prior probabilities on scenes by generating typical examples, creating a synthetic world of scenes and rendered images. We break the images and scenes into a Markov network, and learn the parameters of the network from the training data. To find the best scene explanation given new

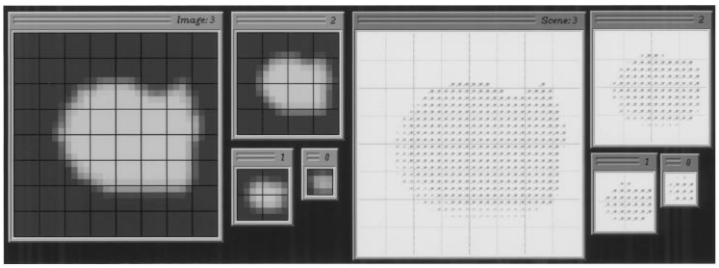


Figure 26. Motion estimation problem. First of two frames of image data (in gaussian pyramid), and corresponding frames of velocity data. The left side shows just one of two image frames. The right side shows (red, darker) motion vectors from the second time frame obscuring (blue, lighter) motion vectors from the first. The scene representation contains both frames. Each large grid square is one node of the Markov network.

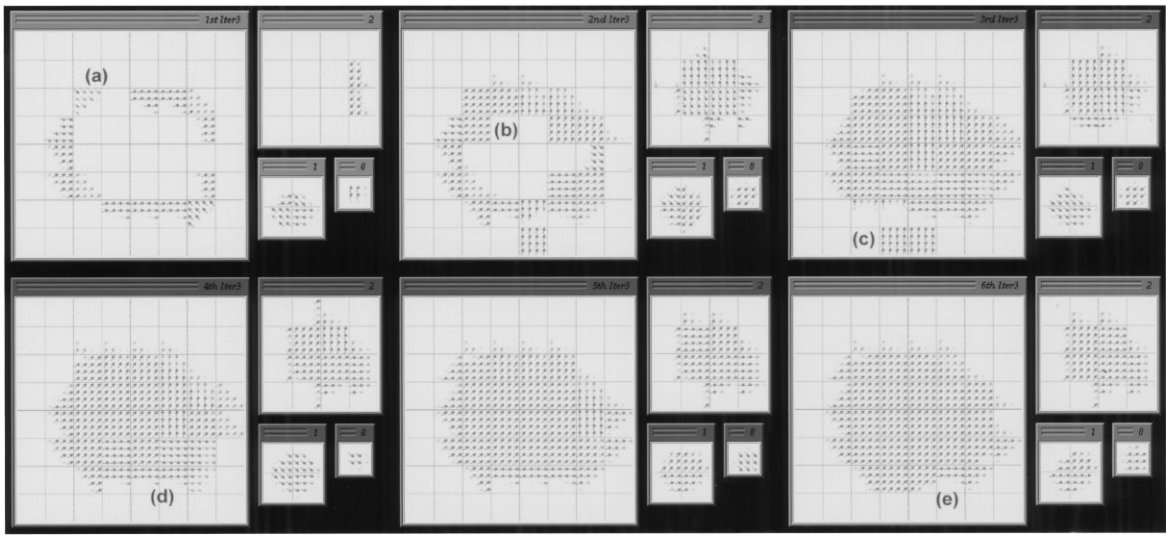


Figure 27. The most probable scene code for Fig. 26(b) at first 6 iterations of Bayesian belief propagation. (a) Note initial motion estimates occur only at edges. Due to the "aperture problem", initial estimates do not agree. (b) Filling-in of motion estimate occurs. Cues for figure/ground determination may include edge curvature, and information from lower resolution levels. Both are included implicitly in the learned probabilities. (c) Figure/ground still undetermined in this region of low edge curvature. (d) Velocities have filled-in, but do not yet all agree. (e) Velocities have filled-in, and agree with each other and with the correct velocity direction, shown in Fig. 26.

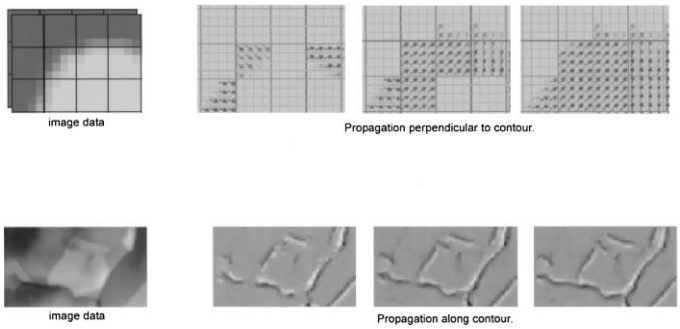


Figure 28. Comparison of algorithm behaviors across problems. For the motion estimation problem, the belief propagation algorithm properly learns to "fill-in" motion information, propagating perpendicularly to the image contour. For the super-resolution problem (example image from Freeman and Pasztor, 1999) propagation can occur along the direction of an image contour, as a hypothesized image contour is extended along its direction (see horizontal line at left, extending to the right). This different behavior occurs using the same probabilistic machinery for the two different problems.

image data, we apply belief propagation in the Markov network even though it has loops, an approach supported by experimental and theoretical studies.

We used very similar machinery for the three problems we discussed. The training data for each particular vision problem yielded different algorithm behavior. Figure 28 shows a comparison of the information propagation between motion estimation and super-resolution. For the motion problem, filling-in propagated interpretations perpendicularly to image contours; for the super-resolution problem, the information propagated along the center contour of the image shown. In each case, the propagation was appropriate to the problem at hand.

The intuitions of this paper—propagate local estimates to find a best, global solution—have a long tradition in computational vision and have been implemented in many ways (Barrow and Tenenbaum, 1981; Rosenfeld et al., 1976; Horn, 1986; Poggio et al., 1985). The power of the VISTA approach lies in the large training database, allowing rich prior probabilities, the selection of scene candidates, which focuses the computation on scenes that render to the image, and the Bayesian belief propagation, which allows efficient inference.

Applied to super-resolution, VISTA gives results that we believe are the state of the art. Applied to shape-from-shading the algorithm shows an ability to distinguish shading from paint for some simple images. Applied to motion estimation, the same method resolves the aperture problem and appropriately fills-in motion over a figure. The technique shows the benefits of applying machine learning methods and large databases to problems of visual interpretation.

Appendix A: Filters Used for Super-Resolution

A.1. Pre-Filter Before Subsampling, to Create Training Data

```
0.25 0.5 0.25
```

applied separably in each dimension.

A.2. Contrast Normalization

Below are the values of the upper-left 7×7 quadrant of the 15×15 filter used in contrast normalization. The square of the mid-band image is blurred by this low-pass filter. After taking the square root, a small constant, 0.01, is added to avoid division by zero later. During contrast normalization, the mid- and high-frequency bands are divided by this blurred energy image.

```
     0  0.0000  0.0004  0.0012  0.0024  0.0031  0.0032
0.0000  0.0004  0.0015  0.0036  0.0057  0.0068  0.0071
0.0004  0.0015  0.0037  0.0065  0.0086  0.0095  0.0097
0.0012  0.0036  0.0065  0.0088  0.0099  0.0103  0.0103
0.0024  0.0057  0.0086  0.0099  0.0103  0.0103  0.0103
0.0031  0.0068  0.0095  0.0103  0.0103  0.0103  0.0103
0.0032  0.0071  0.0097  0.0103  0.0103  0.0103  0.0103
```

A.3. Mid-Band Filter

We do all the processing at the sampling rate of the high resolution band pixels to be estimated. We first double the pixel resolution in each dimension, bilinearly interpolating between samples. This is effectively a low-pass filter.

Then we remove the low-frequencies from the interpolated image, taking advantage of the assumption of Eq. (24), that the lowest image frequencies do not help predict the highest image frequencies, given the mid-band frequencies.

This low-pass filter, L, is applied in the frequency domain. It is rotationally symmetric, with a value in radial spatial frequency, r

$$L(r) = \frac{1 - \exp(-r^2/0.02)}{1 + \exp(-(r - 0.25)/0.075)}, \qquad (31)$$

where r ranges from 0 to $\frac{\pi}{2}$ at the largest distance from the origin in the baseband.

Acknowledgments

We thank E. Adelson, J. Haddon, M. Jordan, K. Murphy, J. Tenenbaum, P. Viola, Y. Weiss, and J. Yedidia for helpful discussions. Two anonymous reviewers also gave comments which improved the paper.

Note

1. However, a nice Photoshop plug-in which uses an undisclosed technique for super-resolution, perhaps fractal-based, is available from http://www.altamira-group.com/html/buyit/order.html.

References

Adelson, E.H. 1995. Personal communication.

Barrow, H.G. and Tenenbaum, J.M. 1981. Computational vision. *Proc. IEEE*, 69(5):572–595.

Bell, A.J. and Sejnowski, T.J. 1997. The independent components of natural scenes are edge filters. *Vision Research*, 37(23):3327–3338.

Berger, J.O. 1985. *Statistical Decision Theory and Bayesian Analysis*. Springer: Berlin.

Besag, J. 1974. Spatial interaction and the statistical analysis of lattice systems (with discussion). *J. Royal Statist. Soc. B*, 36:192–326.

Binford, T., Levitt, T. and Mann, W. 1988. Bayesian inference in model-based machine vision. In *Uncertainty in Artificial Intelligence*, J.F. Lemmer and L.M. Kanal (Eds.), Morgan Kaufmann: Los Alos, CA.

Bishop, C.M. 1995. *Neural Networks for Pattern Recognition*. Oxford.

Burt, P.J. and Adelson, E.H. 1983. The Laplacian pyramid as a compact image code. *IEEE Trans. Comm.*, 31(4):532–540.

Carandini, M. and Heeger, D.J. 1994, Summation and division by neurons in primate visual cortex. *Science*, 264:1333–1336.

DeBonet, J.S. and Viola, P. 1998. Texture recognition using a non-parametric multi-scale statistical model. In *Proc. IEEE Computer Science Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA.

Field, D.J. 1994. What is the goal of sensory coding. *Neural Computation*, 6:559–601.

Freeman, W.T. 1994. The generic viewpoint assumption in a framework for visual perception. *Nature*, 368(6471):542–545.

Freeman, W.T., Haddon, J.A., and Pasztor, E.C. 2001. Learning motion analysis. In *Statistical Theories of the Brain*, R. Rao, B. Olshausen, and M. Lewicki (Eds.), MIT Press, Cambridge, MA. See also http://www.merl.com/reports/TR2000-32.

Freeman, W.T. and Pasztor, E.C. 1999. Learning to estimate scenes from images. In *Adv. Neural Information Processing Systems*, Kearns, M.S., Solla, S.A., and Cohn, D.A. (Eds.). Vol. 11, Cambridge, MA. See also http://www.merl.com/reports/TR99-05/.

Freeman, W.T. and Viola, P.A. 1998. Bayesian model of surface perception. *Adv. in Neural Information Processing Systems*, Vol. 10.

Frey, B.J. 1998. *Graphical Models for Machine Learning and Digital Communication*. MIT Press: Cambridge, MA.

Frey, B.J. 2000. Filling in scenes by propagating probabilities through layers and into appearance models. In *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, Hilton Head Island, S.C.

Geiger, D. and Girosi, F. 1991. Parallel and deterministic algorithms from MRF's: Surface reconstruction. *IEEE Pattern Analysis and Machine Intelligence*, 13(5):401–412.

Geman, S. and Geman, D. 1984. Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Pattern Analysis and Machine Intelligence*, 6:721–741.

Heeger, D.J. and Bergen, J.R. 1995. Pyramid-based texture analysis/synthesis. In *ACM SIGGRAPH*. In *Computer Graphics Proceedings*, Annual Conference Series, pp. 229–236.

Horn, B.K.P. 1986. *Robot Vision*. MIT Press: Cambridge, MA.

Horn, B.K.P. and Brooks, M.J. (Eds.). 1989. *Shape From Shading*. The MIT Press: Cambridge, MA.

Hurlbert, A.C. and Poggio, T.A. 1988. Synthesizing a color algorithm from examples. *Science*, 239:482–485.

Isard, M. and Blake, A. 1996. Contour tracking by stochastic propagation of conditional density. In *Proc. European Conf. on Computer Vision*, pp. 343–356.

Jahne, B. 1991. *Digital Image Processing*. Springer-Verlag: Berlin.

Jordan, M.I. (Ed.). 1998. *Learning in Graphical Models*. MIT Press: Cambridge, MA.

Jordan, M.I., Kearns, M.J., and Solla, S.A. (Eds.), MIT Press, Cambridge, MA. See also http://www.merl.com/reports/TR98-05.

Kersten, D., O'Toole, A.J., Sereno, M.E., Knill, D.C., and Anderson, J.A. 1987. Associative learning of scene parameters from images. *Applied Optics*, 26(23):4999–5006.

Kittler, J. and Illingworth, J. 1985. Relaxation labelling algorithms—a review. *Image and Vision Computing*, 3(11):206–216.

Knill, D. and Richards, W. (Eds.). 1996. *Perception as Bayesian Inference*. Cambridge Univ. Press: Cambridge, London.

Kschischang, F.R. and Frey, B.J. 1998. Iterative decoding of compound codes by probability propagation in graphical models. *IEEE Journal on Selected Areas in Communication*, 16(2):219–230.

Landy, M.S. and Movshon, J.A. (Eds.). 1991. *Computational Models of Visual Processing*. MIT Press: Cambridge, MA.

Luettgen, M.R., Karl, W.C., and Willsky, A.S. 1994. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Trans. Image Processing*, 3(1):41–64.

McEliece, R., MacKay, D., and Cheng, J. 1998. Turbo decoding as as an instance of pearl's 'Belief Propagation' algorithm. *IEEE J.*

on Sel. Areas in Comm., 16(2):140–152.

Olshausen, B.A. and Field, D.J. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann: Los Altos, CA.

Pentland, A. and Horowitz, B. 1993. A practical approach to fractal-based image compression. In *Digital Images and Human Vision*. A.B. Watson (Ed.). MIT Press: Cambridge, MA.

Poggio, T., Torre, V., and Koch, C. 1985. Computational vision and regularization theory. *Nature*, 317(26):314–139.

Polvere, M. 1998. Mars v. 1.0, A quadtree based fractal image coder/decoder. http://inls.ucsd.edu/y/Fractals/.

Rosenfeld, A., Hummel, R.A., and Zucker, S.W. 1976. Scene labeling by relaxation operations. *IEEE Trans. Systems, Man, Cybern*, 6(6):420–433.

Saund, E. 1999. Perceptual organization of occluding contours generated by opaque surfaces. In *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*., Ft. Collins, CO.

Schultz, R.R. and Stevenson, R.L. 1994. A Bayesian approach to image expansion for improved definition. *IEEE Trans. Image Processing*, 3(3):233–242.

Simoncelli, E.P. 1997. Statistical models for images: Compression, restoration and synthesis. In *31st Asilomar Conf. on Sig., Sys. and Computers*, Pacific Grove, CA.

Sinha, P. and Adelson, E.H. 1993. Recovering reflectance and illumination in a world of painted polyhedra. In *Proc. 4th Intl. Conf. Comp. Vis*., pp. 156–163.

Smyth, P., Heckerman, D., and Jordan, M.I. 1997. Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 9(2):227–270.

Szeliski, R. 1989. *Bayesian Modeling of Uncertainty in Low-level Vision*. Kluwer Academic Publishers: Boston.

Weiss, Y. 1997. Interpreting images by propagating Bayesian beliefs. *Adv. in Neural Information Processing Systems*, Vol. 9. pp. 908–915.

Weiss, Y. 1998. Belief propagation and revision in networks with loops. Technical Report 1616, AI Lab Memo, MIT, Cambridge, MA 02139.

Weiss, Y. and Freeman, W.T. 1999. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. Technical Report UCB.CSD-99-1046, Berkeley Computer Science Dept. www.cs.berkeley.edu/~yweiss/gaussTR.ps.gz.

Weiss, Y. and Freeman, W.T. 2001. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. Info. Theory*. Special issue on codes on Graphs and Iterative Algorithms. See also: http://www.merl.com/reports/TR99-39.

Yedidia, J.S., Freeman, W.T., and Weiss, Y. 2000. Generalized belief propagation. Technical Report 2000–26, MERL, Mitsubishi Electric Research Labs., www.merl.com.

Zhu, S.C. and Mumford, D. 1997. Prior Learning and Gibbs Reaction-Diffusion. *IEEE Pattern Analysis and Machine Intelligence*, 19(11).