# Bayesian Belief Propagation for Image Understanding

**Yair Weiss**
Computer Science Division
UC Berkeley
Berkeley, CA 94705
Phone: 510-642-5029
yweiss@cs.berkeley.edu

## Abstract

A central theme of computational vision research has been the realization that reliable estimation of local scene properties requires propagating measurements across the image. Many authors have suggested solving vision problems using architectures of locally connected units updating their activity in parallel. Unfortunately, the convergence of traditional relaxation methods on such architectures has proven to be excruciatingly slow and in general they do not guarantee that the stable point will be a global minimum.

In this paper we show that a scheme in which *Bayesian Beliefs* about image properties are propagated between neighboring units may yield convergence times which are several orders of magnitude faster than traditional methods and avoids local minima. In particular, for some vision problems the scheme provably does not take "too many iterations" in the sense of Marr [11]: at every time step, the local estimates at a given location are optimal given the information which has already been propagated to that location. We illustrate the algorithm's performance on real images and synthetic images and compare it to several existing methods.[1]

[1]A preliminary version of this paper appeared in [23]

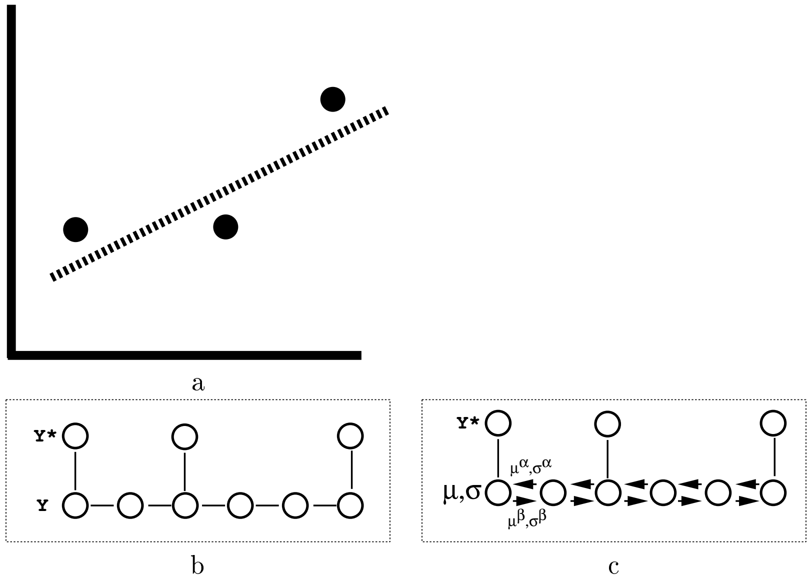



Figure 1: **a.** a prototypical ill-posed problem **b.** Traditional relaxation approach: dense array of units represent the value of the interpolated function. Units update their activity based on local information and the activity of neighboring units. **c.** The Bayesian Belief Propagation (BBP) approach. Units transmit probabilities and combine them according to probability calculus in two non-interacting streams.

Many problems in vision are analogous to the problem shown in figure a – the interpolation of a function from sparse local noisy data. Many solutions in vision are analogous to that shown in figure b — the function is represented by a set of discrete nodes each representing the value of the function at a given location. The interpolating function is found by minimizing a cost function consisting of two terms — a data cost and a coherence cost, e.g.:

$$J(Y) = \sum_k w_k (y_k - y_k^*)^2 + \lambda \sum_i (y_i - y_{i+1})^2 \qquad (1)$$

where $y_k$ is the value of the function at location $k$, $y_k^*$ is the observed data and we have defined $w_k = 0$ for grid points with no data, and $w_k = 1$ for points with data.

Since $J$ is quadratic, any local update in the direction of the gradient will converge to the optimal estimate. This yields updates of the sort:

$$y_k \leftarrow y_k + \eta_k \left(\lambda\left(\frac{y_{k-1} + y_{k+1}}{2} - y_k\right) + w_k (y_k^* - y_k)\right) \qquad (2)$$

Relaxation algorithms differ in their choice of $\eta$: $\eta = 1/(\lambda + w_k)$ corresponds to Gauss-Seidel relaxation and $\eta = 1.9/(\lambda + w_k)$ corresponds to successive over relaxation (SOR) [20].

There is something attractive about update rules of the type in equation 2 —- they consist of simple, local operations that can be carried out in parallel. This is particularly appealing from the point of view of modeling human perception — one can imagine similar operations being carried out by simple neuronal circuits.

Perhaps for this reason, relaxation type models have been proposed to model a wide variety of early vision inference tasks (e.g. [1, 8, 12, 22, 3]).

Unfortunately, the convergence of many relaxation algorithms is excruciatingly slow and this seriously detracts from their attractiveness as models of early human vision. Indeed, Marr (1982) argued against taking the Marr and Poggio (1976) stereo algorithm as a model of human stereo analysis because it required too many iterations. He defined "too many iterations" in the following sense. In any scheme where information is only propagated locally, there is an obvious limit on how fast the information can reach a given unit: i.e. after $n$ iterations each unit can only know about information within distance $n$. Thus there is a minimal number of iterations required for all data to reach all units. Marr distinguished between two types of iterations – those that are needed to allow the information to reach the units, versus those that are used to refine an estimate based on information that has already arrived. Marr stated that he does not believe the brain uses iterations of the second type — to further refine an estimate after all the information has arrived.

A second problem with local iterations of the form given in equation 2 occurs when either the data or the coherence costs in equation 1 are nonquadratic. Now $J$ may have multiple local minima and there is no guarantee that these local updates will converge to the global minimum.

Given these problems, is it time to give up on relaxation as a model of early vision? In this paper, we show that for certain vision problems, once can define local updates that are guaranteed to (1) use only as many iterations as are needed for the data to reach all nodes and (2) avoid local minima. This is done by viewing the cost functional in equation 1 as the log of a posterior probability [21] and then using the machinery of Bayesian Belief Propagation [13] to derive the local updates.

The organization of this paper is as follows. Section 1 illustrates the distinction between traditional relaxation and Bayesian belief propagation (BBP) on the simple example of 1D interpolation. Section 2 analyzes the convergence property of BBP and proves that it will never take "too many" iterations as long as the network is singly connected. Section 3 shows examples of using BBP on two vision problems for which the graph is singly connected — calculation of Hildreth optical flow and figure/ground discrimination based on convexity. Section 4 deals with multiply connected networks.

## 1 An illustrative example

To illustrate the difference between the two types of relaxation algorithms, we consider again the 1D interpolation problem.

To derive the BBP updates for this problem, note that minimizing $J(Y)$ in equation 1 is equivalent to maximizing the posterior probability of $Y$ given $Y^*$ assuming the following generative model:

$$y_{i+1} = y_i + \nu \tag{3}$$

$$y_i^* = w_i y_i + \eta \tag{4}$$

Where $\nu \sim N(0, \sigma_R)$, $\eta \sim N(0, \sigma_D)$. The ratio of $\sigma_D^2$ to $\sigma_R^2$ plays a role similar to that of $\lambda$ in the original cost functional.

The advantage of considering the cost functional as a posterior is that it enables us to use the methods of Hidden Markov Models, Bayesian Belief Nets and Optimal Estimation to derive local update rules (cf. [13, 14, 4]). Denote the posterior by $P_i(u) = P(Y_i = u|Y^*)$, the Markovian property allows us to factor $P_i(u)$ into three terms: one depending on the local data, another depending on data to the left of $i$ and a third depending on data to the right of $i$. Thus:

$$P_i(u) = c\alpha_i(u)L_i(u)\beta_i(u) \tag{5}$$

where $\alpha_i(u) = P(Y_i = u|Y^*_{1,i-1}), \beta_i(u) = P(Y_i = u|Y^*_{i+1,N}), L_i(u) = P(Y^*_i|Y_i = u)$ and $c$ denotes a normalizing constant that is independent of $u$.[2]. Now, denoting the conditional $C_i(u,v) = P(Y_i = u|Y_{i-1} = v)$, $\alpha_i(u)$ can be written in terms of $\alpha_{i-1}(v)$:

$$\alpha_i(u) = c \int_v \alpha_{i-1}(v)C_i(u,v)L_{i-1}(v)dv \tag{6}$$

where c denotes another normalizing constant. A symmetric equation can be written for $\beta_i(u)$.

This suggests a propagation scheme where units represent the probabilities given in the left hand side of equations 5–6 and updates are based on the right hand side, i.e. on the activities of neighboring units. Specifically, for a Gaussian generating process the probabilities can be represented by their mean and variance. Thus denote $P_i \sim N(\mu_i, \sigma_i)$, and similarly $\alpha_i \sim N(\mu^\alpha_i, \sigma^\alpha_i)$ and $\beta_i \sim N(\mu^\beta_i, \sigma^\beta_i)$. Performing the integration in 6 gives a Kalman-Filter like update for the parameters:

$$\mu_i \leftarrow \frac{\frac{w_i}{\sigma_D}Y^*_i + \frac{1}{\sigma^\alpha_i}\mu^\alpha_i + \frac{1}{\sigma^\beta_i}\mu^\beta_i}{\frac{w_i}{\sigma_D} + \frac{1}{\sigma^\alpha_i} + \frac{1}{\sigma^\beta_i}} \tag{7}$$

$$\sigma_i \leftarrow \frac{1}{\frac{w_i}{\sigma_D} + \frac{1}{\sigma^\alpha_i} + \frac{1}{\sigma^\beta_i}} \tag{8}$$

$$\mu^\alpha_i \leftarrow \frac{\frac{1}{\sigma^\alpha_{i-1}}\mu^\alpha_{i-1} + \frac{w_{i-1}}{\sigma_D}Y^*_{i-1}}{\frac{1}{\sigma^\alpha_{i-1}} + \frac{w_{i-1}}{\sigma_D}} \tag{9}$$

$$\sigma^\alpha_i \leftarrow \sigma_R + (\frac{1}{\sigma^\alpha_{i-1}} + \frac{w_{i-1}}{\sigma_D})^{-1} \tag{10}$$

(the update rules for the parameters of $\beta$ are analogous) The iterations are initialized with all messages set to "flat" distributions $\sigma^\alpha_i = \sigma^\beta_i = \infty$.

It is interesting to compare the BBP update rules to the relaxation update rules in equation 2. In both cases, messages are passed between neighboring nodes but there are two important differences:

- in BBP the messages are probability distributions rather than single state estimates.
- in BBP the messages are passed in two, non-interacting streams — a leftward and a rightward stream.

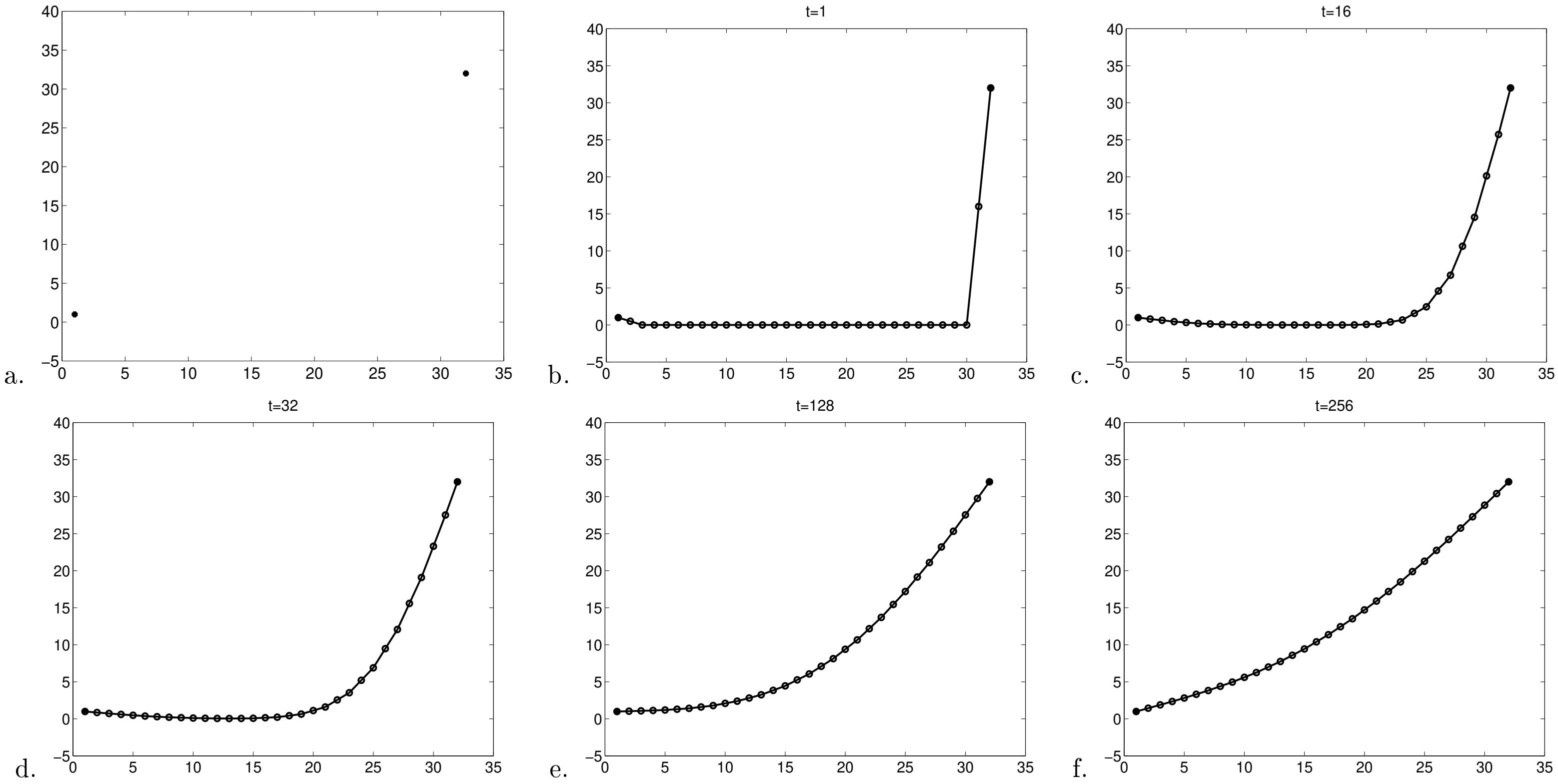



Figure 2: **a.** Sparse data from a straight line. We wanted to see how many iterations are needed for local relaxation to recover the straight line. **b-f** The estimate using Gauss-Seidel after $1, 16, 32, 128$ and $256$ iterations.

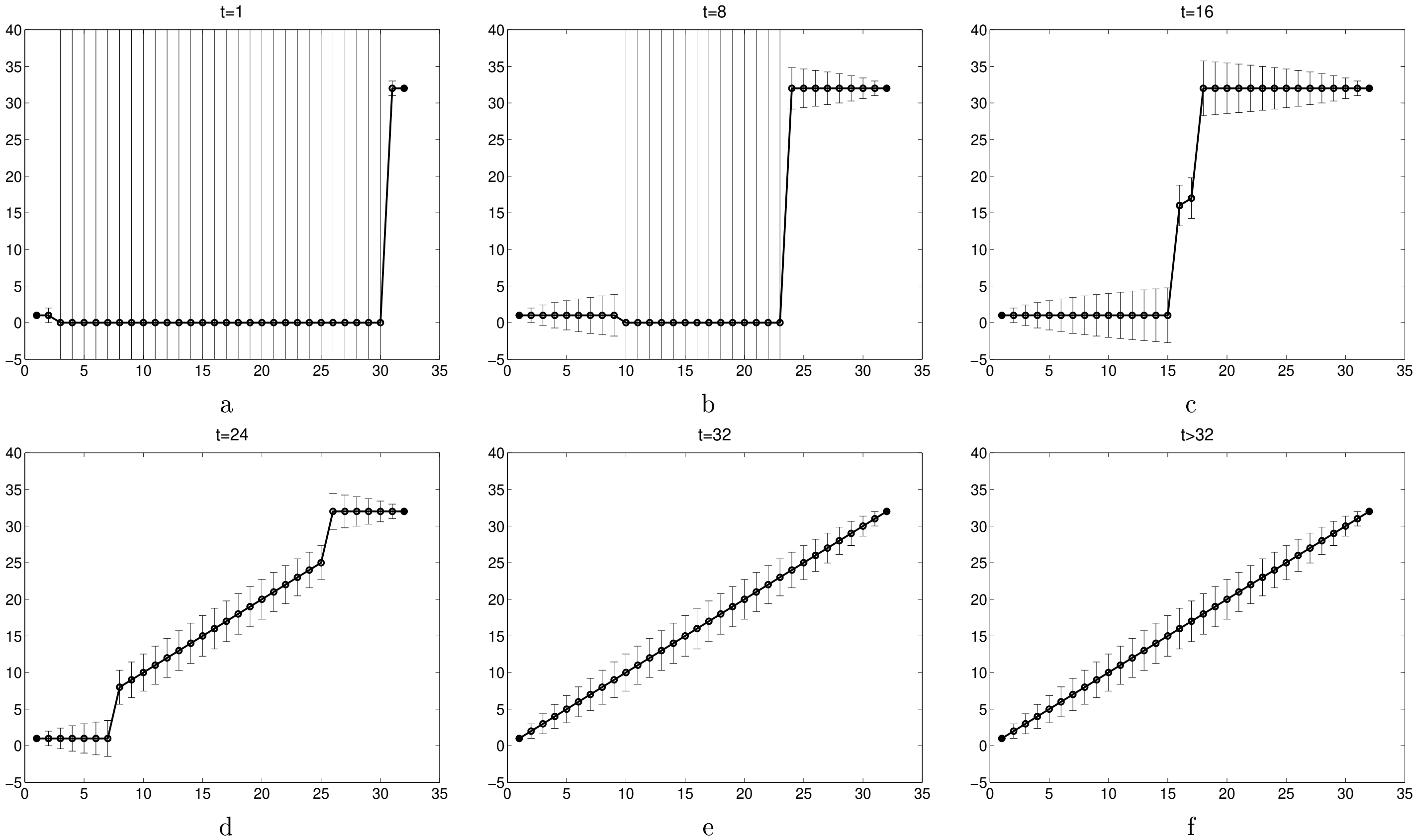



Figure 3: **a.-f** The estimates using BBP after $1, 8, 16, 24$ and $32$ iterations. The algorithm converges after 32 iterations – neither the means nor the standard deviations change.

Figure 2–3 shows the difference in performance between the two approaches as a function of iteration. An iteration refers to one update of all units. We used a synchronous update schedule for BBP and an asynchronous, random order, update schedule for Gauss-Seidel (the synchronous update schedule for Gauss-Seidel gives Jacoby iterations whose convergence is far slower than Gauss-Seidel). We used $\sigma_D = 0.001$,$\sigma_P^2 = 1$ and we used 32 nodes to represent the function.

The data is shown in figure 2a — the sampled data were simply two points from the line $y = x$: $(1,1), (32,32)$ and the optimal answer in this case can be computed by inverting a $32x32$ matrix and it gives the original $y = x$ straight line. How many iterations do the the two types of relaxation need to arrive at this answer?

For the Gauss-Seidel updates we need an initial condition and we arbitrarily initialized all $y_i = 0$. Figure 2b–d show the estimate as a function of iteration. After 32 iterations, the estimate is far from correct and shows the strong influence of the initial conditions. This illustrates the distinction that Marr made — after 32 iterations, sufficient time has gone by so that all information has reached all the units. Any additional iterations are used to refine estimates given information that has already arrived. Obviously, Gauss-Seidel relaxation suffers from the problems Marr raised — even after 256 iterations the estimate is not the correct, linear function.

The dynamics of BBP on this problem are quite different. Since the messages are probabilistic there is no influence of initial conditions — the messages are capable of representing the statement "I don't have any evidence to pass on". After 32 iterations, the estimates converge to the correct answer — no additional iterations are required to refine the estimate. Furthermore after any $t < 32$ iterations, all nodes have the optimal estimate given all the information that has already arrived. Consider for example, the case $t = 1$. Nodes in the center have infinite variance, since no information has arrived yet. The node immediately adjacent to the datapoints, however, estimate their height to be identical to that of the adjacent datapoint. Indeed it is easy to see from the cost function 1 that if the only information that has arrived is a single datapoint, then the optimal estimate is a horizontal line going through that datapoint. Consider, however, what happens at iteration $t = 16$. Now the two center nodes have received the information from both datapoints and immediately, their estimate is the correct one. Again, this is the optimal estimate given all the information that has arrived. In the next section we prove that BBP has this property.

## 2 Formulation and Convergence

Let us formalize the conversion of a cost function minimization to a BBP algorithm. Suppose we have a scene attribute $y$ that we want to estimate at all locations indexed by $i$ and we have some local evidence $y_i^*$ at some of the locations. We assume our cost function is of the form:

$$J(Y) = \sum_{i,j} J(y_i, y_j) + \sum_i J(y_i, y_i^*) \tag{11}$$

(where some of the terms may be zero)

[2]Note that in equation 3 the prior distribution $P(y_i = u)$ is independent of $u$ for all $i$

This corresponds to assuming a probability distribution in which:

$$P(Y, Y^*) = \frac{1}{Z} e^{-J(Y)} \tag{12}$$

The first stage is to construct an undirected graph $G$ corresponding to the probability $P$. All nodes $y_i$ are connected to $y_i^*$. If there exists a term $J(y_i, y_j)$ in the cost function then we add an edge connecting nodes $y_i$ and $y_j$. The undirected graph $G$ represents graphically the conditional independence assumptions in $P$ — if any two nodes are separated by a third node, then their values are conditionally independent given that node.

The graph $G$ also serves as the data structure for propagation. With each such edge we associate a transition matrix $M_{ij}$ defined by:

$$M_{ij}(u, v) = e^{-J(y_j = u, y_i = v)} \tag{13}$$

In BBP, each node $y_i$ sends a message to its neighbor $y_j$ which represents a probability distribution over the value of $y_j$. In addition, each node calculates a *belief* — a probability distribution over its own value given all the data. For discrete scene attributes, both the messages and the beliefs can be represented as vectors. Thus if the scene attribute can take on one of two possible values, then the messages will be vectors of length 2 whose elements sum to 1. The message that node $y_i$ sends to $y_j$ is updates as follows:

- Combine all messages coming into $y_i$ except for that coming from $y_j$ into a vector $\vec{v}$. The combination is done by multiplying all the message vectors element by element.
- Multiply $\vec{v}$ by the matrix $M_{ij}$ corresponding to the link from $y_i$ to $y_j$.
- Normalize the product $M_{ij}\vec{v}$ so it sums to 1. The normalized vector is sent to $y_j$.

The belief vector for a node $y_j$ is obtained by combining all incoming messages to $y_j$ (again by multiplying the message vectors element by element) and normalizing.

The procedure is initialized with all message vectors set to $(1, 1, \cdots 1)$. Evidence nodes $y_i^*$ do not receive messages and they always transmit the same vector — whose components are simply $e^{-J(y_i, y_i^*)}$. The normalization of the outgoing messages is not theoretically necessary — whether the message are normalized or not, the belief vector will be identical. Thus one could use an equivalent algorithm where the messages are not normalized. However, as Pearl has pointed out, normalizing the messages avoids numerical underflow and adds to the stability of the algorithm.

If the scene attributes $y_i$ are not discrete, then the messages and beliefs are continuous density functions and the matrix $M_{ij}$ is replaced by a linear integral operator in the usual fashion. In many cases the density functions may be represented in parametric form, e.g. $\mu, \sigma$ for a Gaussian. Note that if the graph $G$ is simply a chain (as in the example of the previous section) the procedure outlined in this section gives rise to two noninteracting streams of messages — the $\alpha$ and $\beta$ update rules given in the previous section.

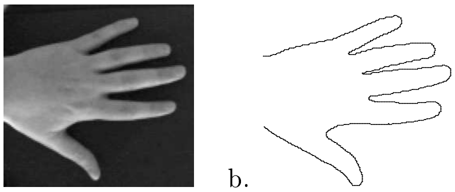


Figure 4: **a.** the first frame of a sequence. The hand is translated to the left. **b.** contour extracted using standard methods

Having defined the BBP procedure, we can state its convergence properties. The first claim is simply a restatement of the result in Pearl (88) regarding the correctness of belief propagation in singly connected Bayesian networks. The second claim is, we believe, novel and shows that BBP doesn't take "too many iterations" in the sense of Marr (82).

*Claim 1:* If the graph $G$ is singly connected, then BBP converges in $d$ iterations where $d$ is the maximal distance between any two nodes in the graph. At convergence the belief vector at a node $y_i$ is equal to the posterior probability of $y_i$ given all the measurements $Y^*$.

*Proof:* The proof is identical to that given in Pearl (88) for the directed graph case. The conditional independence assumptions embodied by the graph guarantee that the calculations are exact.

*Claim 2:* If the graph $G$ is singly connected, then after $t$ iterations the belief vector at a node $y_i$ is equal to the posterior probability of $y_i$ given all measurements within distance $t$ i.e. $y_j^*$ s.t. $d(y_i, y_j^*) \leq t$

*Proof:* We first construct a new graph $G'$ which contains only nodes whose distance from $y_i$ is less than $t$. From the previous claim, if we run BBP until convergence on $G'$ the belief at $y_i$ will be equal to the posterior of $y_i$ given all evidence in $G'$. Now, by inspecting the update rules it is easy to show that the incoming messages to $y_i$ in the full graph $G$ after $t$ iterations are identical to the incoming messages to $y_i$ in $G'$ at convergence.

*claim 3:* Let $\hat{y}_i$ be the value $u$ that maximizes the belief at node $y_i$. Then $\hat{y}_i$ is the optimal estimate of $y_i$ in the mean squared sense.

*Proof:* This follows directly from claim 2.

Note that $\hat{y}_i$ does not in general correspond to the MAP estimate of $\hat{y}_i$ (except for the important special case of quadratic costs). In what follows we will refer to the "BBP estimate" as $\hat{y}_i$. There exist also version of BBP that calculate the MAP estimate [13].

## 3 Results — singly connected graphs.

Figure 4a shows the first frame of a sequence in which the hand is translated to the left. Figure 4b shows the bounding contour of the hand extracted using standard techniques.

### 3.1 Motion propagation along contours

Local measurements along the contour are insufficient to determine the motion. Hildreth [5] suggested to overcome the local ambiguity by minimizing the following cost functional:

$$J(V) = \sum_k (dx_k^t v_k + dt_k)^2 + \lambda \sum_k \|v_{k+1} - v_k\|^2 \tag{14}$$

where $dx, dt$ denote the spatial and temporal image derivatives and $v_k$ denotes the velocity at point $k$ along the contour. This functional is analogous to the interpolation functional (eq. 1) and the derivation of the relaxation and BBP updates are also analogous.

The velocity is represented by a chain of units, each representing the local velocity at a point along the contour. Each velocity node is connected to a local evidence node, that observes the local spatiotemporal derivatives. Since the corresponding graph is a chain, there will be two, noninteracting streams of messages, each represented by a mean and a variance. Here the mean $\mu^\alpha$ is a 2 vector and the variance is a $2x2$ inverse covariance matrix $P^\alpha$ . Similarly the message sent from the local evidence nodes to the local velocity nodes is given by a mean $\mu_i^l$ (the normal velocity) and a local inverse covariance matrix $P_i^l$ [18].

The update rules for the messages are $2D$ analogs of the $1D$ case. For example the update for $\mu_i^\alpha$:

$$\mu_i^\alpha \leftarrow (P_{i-1}^\alpha + P_i^l)^\dagger (P_{i-1}^\alpha \mu_i^\alpha + P_i^l \mu_i^l) \tag{15}$$

where $P^\dagger$ denotes the pseudo-inverse of $P$. The advantage of working with inverse covariance matrices is that infinite uncertainty can be represented easily. Thus the messages are initialized with $P_i^\alpha, P_i^\beta$ set to the zero matrix.

Figure 5a shows the estimate of motion based solely on local information. The estimates are wrong due to the aperture problem. Figure 5b shows the performance of three propagation schemes: gradient descent, SOR and BBP. Gradient descent converges so slowly that the improvement in its estimate can not be discerned in the plot. SOR converges much faster than gradient descent but still has significant error after 500 iterations. BBP gets the correct estimate after 3 iterations ! (Here and in all subsequent plots an iteration refers to one parallel update of all units in the network). This is due to the fact that after 3 iterations, the estimate at location $k$ is the optimal one given data in the interval $[k-3, k+3]$. In this case, there is enough data in every such interval along the contour to correctly estimate the motion. Figure 5c shows the estimate produced by SOR after 500 iterations. Even with simple visual inspection it is evident that the estimate is quite wrong. Figure 5d shows the (correct) estimate produced by BBP after 3 iterations.

### 3.2 Direction of figure propagation

The extracted contour in figure 4 bounds a dark and a light region. *Direction of figure* (DOF) (e.g. [17]) refers to which of these two regions is figure and which is ground. A local cue for DOF is convexity - given three neighboring points along the contour we prefer the DOF that makes the angle defined by those points acute rather than obtuse. Figure 6a shows the results of using this local cue on the hand contour. The local cue is not sufficient.

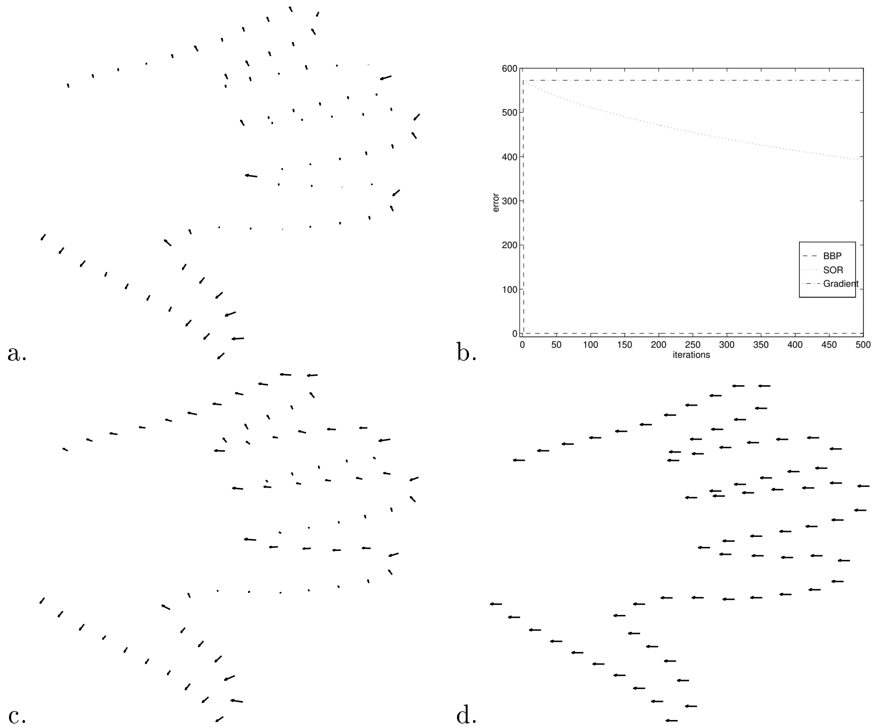



Figure 5: **a.** Local estimate of velocity along the contour. **b.** Performance of SOR, gradient descent and BBP as a function of time. BBP converges orders of magnitude faster than SOR. **c.** Motion estimate of SOR after 500 iterations. **d.** Motion estimate of BBP after 3 iterations.

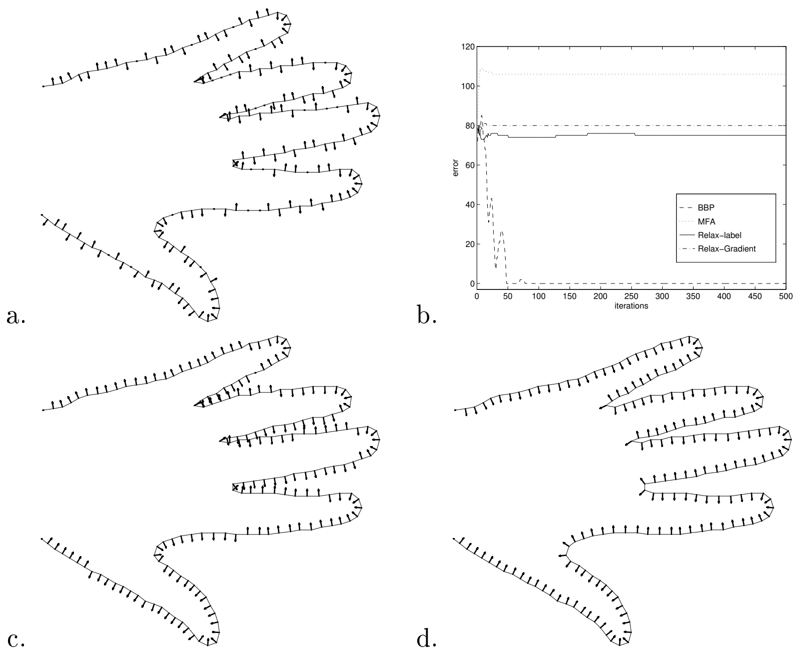



Figure 6: **a.** Local estimate of DOF along the contour. **b.** Performance of MFA,gradient descent, relaxation labeling and BBP as a function of time. BBP is the only method that converges to the global minimum. **c.** DOF estimate of MFA after convergence. **d.** DOF estimate of BBP after convergence.

We can overcome the local ambiguity by minimizing a cost functional that takes into account the DOF at neighboring points in addition to the local convexity. Denote by $L_k$ a binary vector of length 2 that represents the DOF at point $k$ along the contour. Thus $L_k = (1,0)^T$ or $L_k = (0,1)^T$. Define the cost:

$$J(L) = \sum_m \sum_k V_k(m) L_k(m) - \lambda \sum_m \sum_k L_k(m) L_{k+1}(m) \tag{16}$$

with $V_k(m)$ determined by the acuteness of the angle at location $k$.

We compared BBP to four "relaxation labeling" type relaxations. These are updates that estimate a *continuous* vector $\hat{L}_k$ and update the value of $\hat{L}_k$ at every iteration based on the values at the neighboring locations. For a simple chain graph:

$$\hat{L}_k \leftarrow f(V_k, \hat{L}_{k-1}, \hat{L}_k, \hat{L}_{k+1}) \tag{17}$$

As in the Gaussian relaxation case discussed earlier, different relaxation labeling algorithms differ in their choice of $f$. For continuous relaxation labeling, one replaces $L_k$ in equation 16 with $\hat{L}_k$ and chooses a change in $\hat{L}_k$ so the cost is guaranteed to be decreased. (see [9] for a review of relaxation labeling methods ). Alternatively, one can adopt a mean-field approach [3] and minimize:

$$J(L) = \sum_m \sum_k V_k(m) \hat{L}_k(m) - \lambda \sum_m \sum_k \hat{L}_k(m) \hat{L}_{k+1}(m) \tag{18}$$

$$+ \; T \sum_m \sum_k \log(\hat{L}_k(m)) \hat{L}_k(m) \tag{19}$$

where $T$ is a "temperature" parameter. Mean-Field annealing (MFA) refers to gradually reducing the temperature parameter during the iterations [15].

Figure 6b shows the performance of four propagation algorithms on this task: three traditional relaxation labeling algorithms (MFA, Rosenfeld et al, constrained gradient descent) and BBP. All three traditional algorithms converge to a local minimum, while the BBP converges to the global minimum. Figure 6c shows the local minimum reached by MFA and figure 6d shows the correct solution reached by the BBP algorithm. Recall (section 2) that BBP is guaranteed to converge to the correct posterior given all the data.

Figure 7 shows the output of the different algorithms on a figure from the cover of a United Airlines magazine. Here there is no correct answer but perceptually the DOF appears to flip somewhere so it always points to the interior of the hand. Indeed BBP converges to this "perceptual" solution, while none of the relaxation algorithms do. Again, this is a local minima problem — the solution obtained by the relaxation operators has higher cost in equation 16 than the one obtained by BBP.

## 4 Results — multiply connected graphs.

### 4.1 Multiply connected graphs with a single loop

Weiss [24] has recently analyzed BBP in networks with a single loop. He has shown conditions for which BBP is guaranteed to converge to the optimal estimate.

One can show using those results that using BBP for obtaining optical flow along *closed* contours will converge to the optimal answer. Figure 8 shows an example. Figure 8a shows the correct flow for a rotating square. Figure 8b shows the estimates of BBP and SOR as a function of iteration. Here BBP converges in 18 iterations to the correct answer, while SOR converges in roughly 200 iterations. The difference can again be simply attributed to the proof in section 2. For any point along the circumference of the square, the correct flow can be determined given the information from two corners straddling the point. In this case, we used 18 discrete nodes along each edge of the square and therefor BBP converges exactly in 18 iterations — no additional iterations are needed to refine the estimate. SOR, on the other hand, needs to continually refine the estimate.

We have also obtained similar results using BBP for DOF propagation in closed contours. Even though there is a loop, BBP converges to the correct answer while the relaxation operators often converge to local minima.

### 4.2 BBP in a grid

In many vision problems, the graph is a grid. How well does BBP work on these problems?

Figure 9 shows a synthetic example. The goal is to compute a dense figure ground labeling and there is local evidence at every pixel giving a local probability of being figure. White corresponds to high probability and black to low probability. The

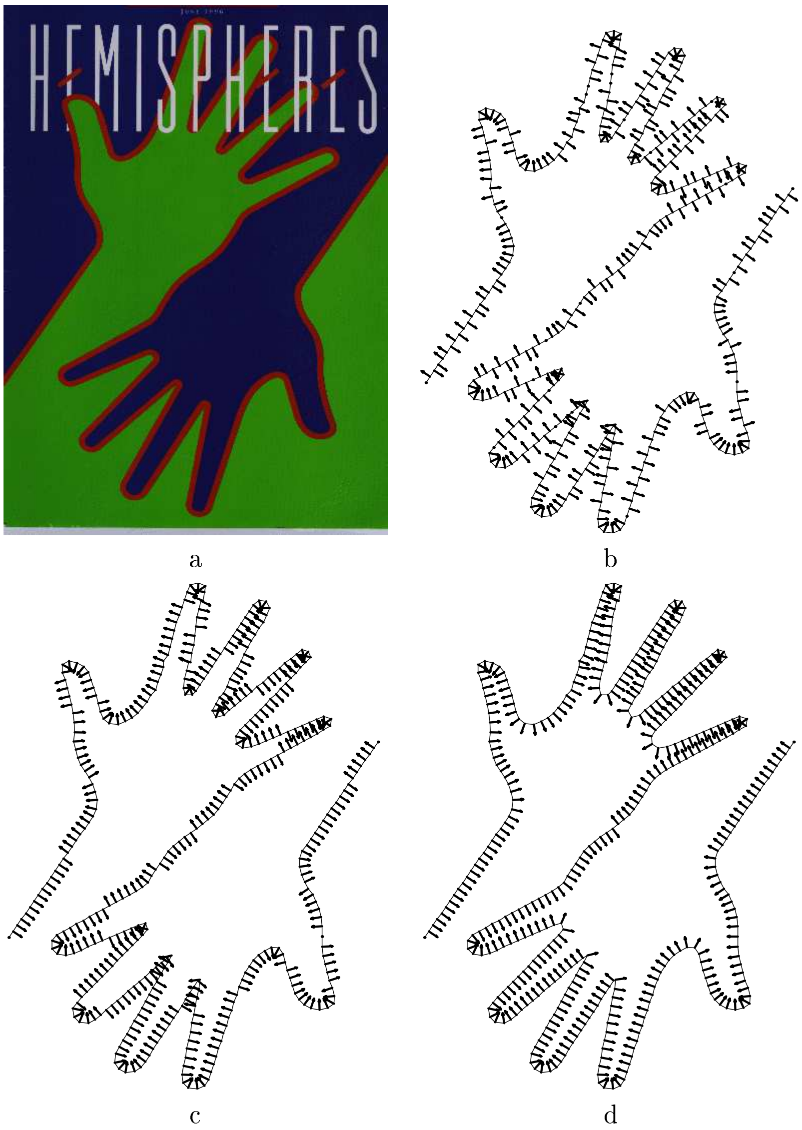



Figure 7: **a:** A figure in which the DOF appears to shift perceptually. **b:** The local DOF estimates. **c:** The output of a relaxation labeling algorithm. It converges to a local minimum. **d:** The output of BBP. Even though the algorithm knows nothing about hands, the DOF shifts so that the interiors of the hand are always figure.

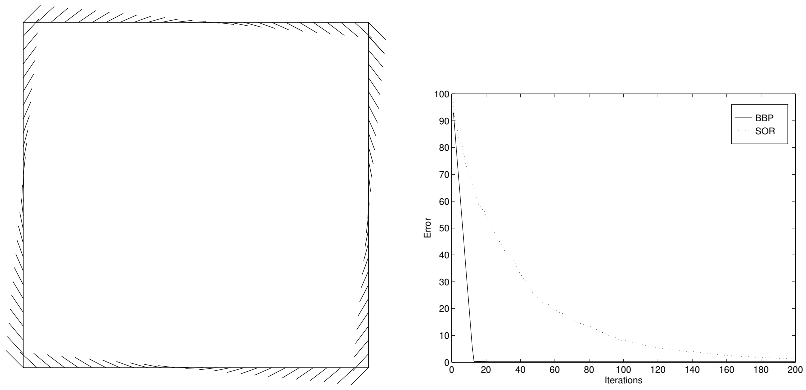



Figure 8: The success of BBP is not limited to singly connected networks. We show the result of running SOR and BBP for finding optical flow in a rotating square. Both algorithms are minimizing the Hildreth cost functional. Here there is a single loop but BBP still converges to the correct answer and does so orders of magnitude faster than SOR.

cost function is:

$$\begin{aligned} J(L) &= \sum_m \sum_{ij} V_{ij}(m) L_{ij}(m) \\ &- \lambda \sum_m \sum_{ij} (L_{ij}(m) L_{i+1,j}(m) + L_{ij}(m) L_{i-1,j}(m) + L_{ij}(m) L_{i,j+1}(m) + L_{ij}(m) L_{i,j-1}(m)) \end{aligned} \tag{20}$$

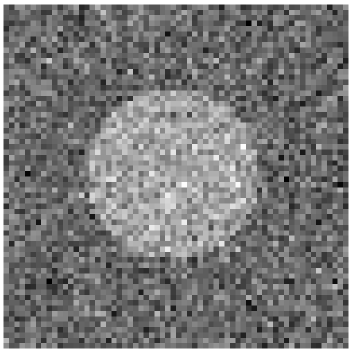
This is a direct 2D analog of equation 16 but the coherence is now over all neighbors of a node in a grid.

Figure 9a shows the synthetic-ally generated local probabilities. Perceptually, the figure appears to be a circle in the center of the image. Figure 9b shows the probability computed using BBP. We find that BBP does converge (in this case in 17 iterations). Although there is no longer the guarantee of optimality, it is interesting to see that BBP does converge to numbers that look reasonable. Figure 9c shows the output of MFA at convergence. The output is similar to that obtained by BBP and convergence times are of similar orders of magnitude (depending on the annealing schedules for MFA). One difference between the two outputs, is that in BBP the final outputs still has the probabilistic interpretation — note that pixels near the border of the circle are more gray than those in the center. At MFA (depending on the final temperature) this information may be lost. This is illustrated in figure 9d.

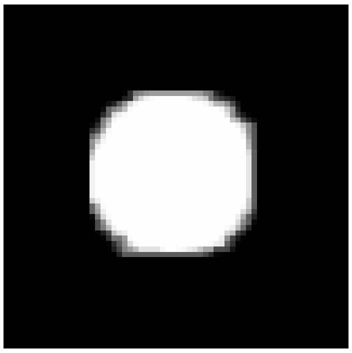
### 4.3 BBP in quadtree architectures

Willsky and his colleagues [10] have shown that MRFs can be approximated with hierarchical or multi-resolution models. We have been experimenting with BBP on quadtree architectures.

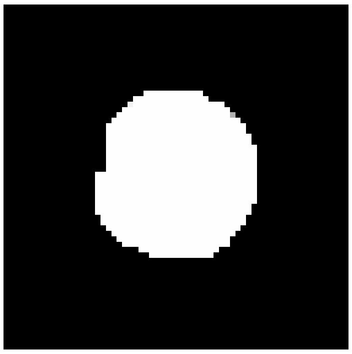
Figure 10 shows an example where we are interested in minimizing the Horn and Schunck [6] cost functional. This functional is the 2D analog of the Hildreth functional — coherence is measured at all four nearest neighbors on a grid. Following [10]



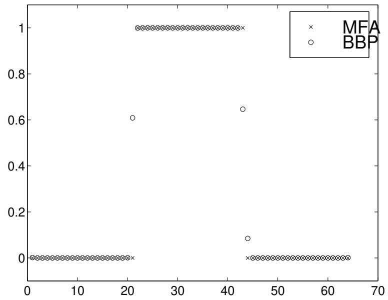
a



b



c




d

Figure 9: **a.** a synthetic 2D figure-ground problem. bright pixels represent high probability of being figure. **b.** The probability of being figure after BBP has converged. Despite the presence of multiple loops, BBP converges (in 17 iterations). **c.** The output at convergence of MFA on the same problem. **d.** Cross sections from the MFA and BBP outputs. Note that in MFA the probabilistic information has been lost.

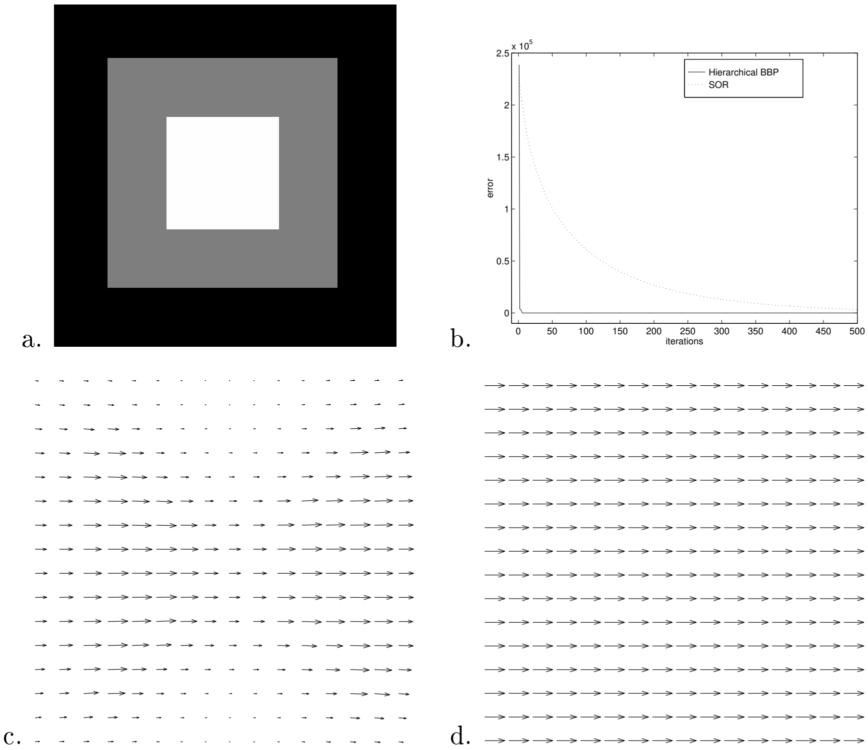



Figure 10: **a**. A single frame from a translating square sequence. **b.** The relative errors of BBP and SOR, where BBP operates in a quadtree.

we approximate the cost functional with a cost functional operating on a quadtree architecture. Since the quadtree is singly-connected BBP is guaranteed to converge. Figure 10b shows the relative error as a function of iterations. Figure 10c shows the output of SOR after 100 iterations. Again, we find that SOR takes "too many iterations" — the image is of size $64x64$ so 64 iterations are sufficient to propagate all information to all nodes. Nevertheless after 100 iterations, the flow is quite wrong. Figure 10d shows the output after 10 iterations using hierarchical BBP. Since enough information has been propagated, the estimates are correct and no additional iterations are required to refine the estimate.

Note that similar plots have appeared in [10] but there is an important difference. Here, we are interested in measuring performance as a number of iterations in a parallel architecture. Hence an iteration refers to a single update of all nodes. In [10] the quadtree algorithm was *noniterative* — the output was calculated in a single sweep up and down the tree.

The quadtree BBP retains some attractive properties of BBP along contours — at every iteration $t$ the estimate is the "optimal" one given all evidence within distance $t$ where distance is measured along the quadtree. However, the optimality is not in the sense of the original probability distribution but rather in terms of the quadtree approximation. In particular, this leads to rather noticeable artifacts along the quadtree boundaries [10]. Freeman and Pasztor [2] have recently experimented with combining quadtree propagation with grid propagation and have obtained encouraging results.

## 5 Discussion

The appeal of local, parallel computations as metaphors for early visual computation goes back to the Gestalt psychologists, and was a driving force behind the influential "intrinsic images" model [1]. In this paper we asked "does local propagation have to be so slow?". We showed that if nodes propagate probabilities and combine them in accordance with the probability calculus, local propagation does not have to be slow. In fact, for certain vision problems it can be shown that such a scheme will never take "too many iterations" in the sense of Marr (1982) — it will never require additional iterations to refine an estimate after all the information has arrived. Although this result is only true for problems in which the graph is singly connected, we showed results showing similar behavior may hold for a larger class of problems.

Since our interest was in models that may serve as metaphors of early visual calculations in the brain, we did not consider global, non-iterative solutions to similar problems. For example, quadratic cost functionals such as Hildreth's can be minimized in one pass by inverting a huge matrix. Similarly, the figure ground functional we discussed can be minimized using the principle of dynamic programming. Even the 2D figure ground functional can be minimized in polynomial time using max-flow algorithms [7]. These types of solutions were not the focus of this paper, although finding local, parallel versions of these algorithms may be an interesting topic to pursue.

While we would not naively attempt to translate the BBP update rules into neuronal circuitry, it is encouraging to note that the main building blocks of the BBP rules are (1) a distributed representation of probabilities (2) weighted linear summation of the probabilities and (3) normalization. Heeger and Simoncelli [19] have argued for these three building blocks in their feed-forward model for the transformation of data from area V1 to area MT. Thus the basic building blocks needed to perform fast, probabilistic propagation of early visual constraints may indeed exist in the brain.

## References


[1] H.G. Barrow and J.M. Tenenbaum. Recovering intrinsic scene characteristics from images. In A. Hanson and E. Riseman, editors, *Computer Vision Systems*. Academic Press, 1978.

[2] W.T. Freeman and E.C. Pasztor. Learning to estimate scenes from images. In M.S. Kearns, S.A. Solla, and D.A. Cohn, editors, *Adv. Neural Information Processing Systems 11*. MIT Press, 1999.

[3] D. Geiger and F. Girosi. Parallel and deterministic algorithms from MRFs: surface reconstruction. *IEEE Trans. PAMI*, 13(5):401–412, 1991.

[4] Arthur Gelb, editor. *Applied Optimal Estimation*. MIT Press, 1974.

[5] E. C. Hildreth. *The Measurement of Visual Motion*. MIT Press, 1983.

[6] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artif. Intell.*, 17(1–3):185–203, August 1981.

[7] H. Ishkikawa and D. Geiger. Segmentation by grouping junctions. In *Proc. CVPR*, 1998.

[8] D. Kersten. Transparency and the cooperative computation of scene attributes. In J. Landy and J. Movshon, editors, *Computational models of visual processing*. MIT Press, Cambridge, Massachusetts, 1991.

[9] S.Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer-Verlag, 1995.

[10] Mark R. Luettgen, W. Clem Karl, and Allan S. Willsky. Efficient multiscale regularization with application to the computation of optical flow. *IEEE Transactions on image processing*, 3(1):41–64, 1994.

[11] D. Marr. *Vision*. H. Freeman and Co., 1982.

[12] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Nature*, 194:283–287, 1976.

[13] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[14] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech recognition*. PTR Prentice Hall, 1993.

[15] A. Rangarajan and R. Chellapa. Markov random field models in image processing. In M. Arbib, editor, *Handbook of brain theory and neural networks*, pages 564–567. MIT Press, 1995.

[16] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6:420–433, 1976.

[17] P. Sajda and L. H. Finkel. Intermediate-level visual representations and the construction of surface perception. *Journal of Cognitive Neuroscience*, 1994.

[18] E.P. Simoncelli, E.H. Adelson, and D.J. Heeger. Probability distributions of optical flow. In *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, pages 310–315, 1991.

[19] E.P. Simoncelli and D.J. Heeger. A model of neuronal responses in visual area MT. *Vision Research*, 38(5):743–761, 1998.

[20] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge, 1986.

[21] R. Szeliski. *Bayesian modeling of uncertainty in low-level vision*. Kluwer, 1989.

[22] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Trans. PAMI*, 8:413–424, 1986.

[23] Y. Weiss. Interpreting images by propagating Bayesian beliefs. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, 1996.

[24] Y. Weiss. Belief propagation and revision in networks with loops. Technical Report 1616, MIT AI lab, 1997.