Understanding Belief Propagation and its Applications

Dan Yuan
dyuan@soe.ucsc.edu

Department of Computer Engineering
University of California, Santa Cruz

Abstract

"Inference" problem arise in computer vision, AI, statistical physics and coding theory. The rationale behind the belief propagation is an efficient way to solve inference problems by propagating local messages around neighborhoods [5]. Although researchers proved that the belief propagation (BP) converges to a unique fixed point (fixed probabilistic belief) on singly connected graphs [1], they also have shown that the BP is able to produce great results on graphs with loops, empirically [2]. However, a theoretical understanding and proof of this performance has not been achieved. Since images can be easily represented as the loopy graphs, where graph nodes are associated with pixels or image patches, many image processing researchers have experienced with BP, with promising results. In this report, we will focus on the principle of BP, and its applications in computer vision.

1. Introduction

Problems involving probabilistic belief propagation arise in a wide variety of applications including error correcting codes [10], speech recognition and medical diagnosis. The basic problem is perhaps best illustrated by a simple example. Consider Fig 1. The task is to infer the states of three hidden nodes (A, B, C) using observed the states of three evidence nodes (E_A, E_B, E_C). We assume a probability distribution on the variables formulated as:

$$P(A,B,C \mid E_A, E_B, E_C) = \frac{1}{Z} e^{-J(A,B,C,E_A,E_B,E_C)} \qquad (1)$$

where Z is a normalizing function and :

$$J = J_1(A,B) + J_2(B,C) + J_3(A,E_A) + J_4(B,E_B) + J_5(C,E_C) \qquad (2)$$

Intuitively, there is a cost named compatibility on each link between two neighboring nodes. We assume only the pair-wise compatibility between two nodes. This is more restrictive than the general MRF formulations. The total cost is decomposed into five additive costs. Alternatively, the probability function P can be thought of as factoring into five multiplicative potential functions Ψs :

$$P = \Psi(A,B)\Psi(B,C)\Psi(A,E_A)\Psi(B,E_B)\Psi(C,E_c) \qquad (3)$$

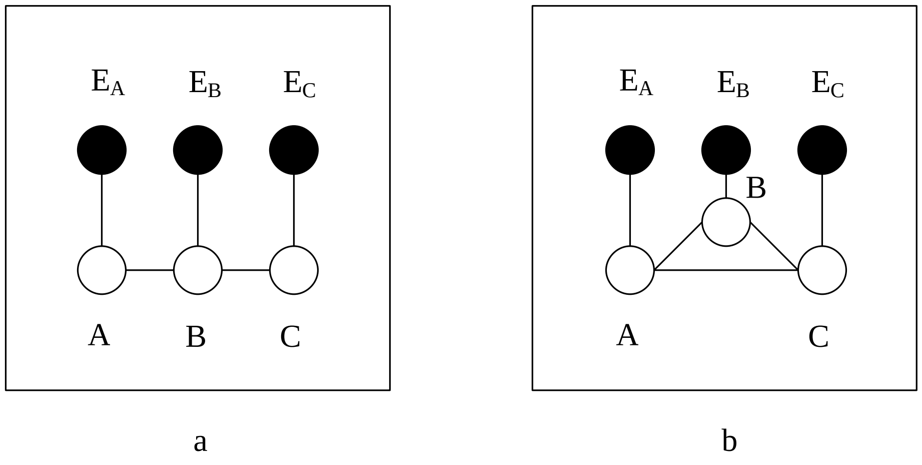

Figure 1: a. An example of the type of problems typically solved using belief propagation. Observed nodes are denoted by filled circles. A link between any two nodes implies a probabilistic compatibility constraint. b. A simple network with a loop.

The task of "inferring the states of the hidden nodes is to calculate the probability associated to each state of that hidden node given all the evidence e.g. $P(A = a \mid E)$. This task can be done by exhaustive enumeration. Apparently, the most likely sequence can be found by trying all possibilities and the belief function can be calculated by marginalization:

$$P(A = a \mid E) = \sum_{b,c} P(A = a, B = b, C = c \mid E) \qquad (4)$$

A naïve exhaustive enumeration is, of course, usually intractable as the number of the hidden nodes increases.

J. Pearl [1] described local message propagating schemes for inferring the states of the hidden nodes in these types of networks. The algorithm consists of simple local updates that can be executed and are guaranteed to converge to the correct probabilities. The term ***belief update*** is used to describe the scheme for computing the function of probabilistic belief. His algorithm is equivalent to schemes proposed independently in a wide range of fields including information theory, signal processing and optimal estimation.

It was proved that Pearl's algorithm would work for the singly connected graphs. In many applications especially image processing applications, however, the networks are loopy. Consider for example, the network shown in Fig 1b, where we would have a compatibility link between the first and the last hidden nodes.

$$P = \Psi(A,B)\Psi(B,C)\Psi(C,A)\Psi(A,E_A)\Psi(B,E_B)\Psi(C,E_c) \qquad (5)$$

Pearl showed that his algorithm is not guaranteed to work for such a network. Despite this fact, several groups [2][3] have recently reported excellent experimental results in inference in original networks with loops by using Pearl's algorithm.

2. The Max-Product Algorithm in Pair-wise Markov Random Field

Pearl's algorithm was described for directed graphs, but in this report we focus on undirected graphs. Every directed graphical model can be transformed into an undirected graphical model before doing inference. An undirected graphical model (or a Markov Random Field) is a graph in which the nodes represent variables and arcs represents compatibility relations between them (ref Fig1.)

The advantage of preprocessing the graph into one with pair-wise cliques is that the description and the analysis of belief propagation become simpler. In every iteration, each node sends a message to each of its neighbors and receives a message from each neighbor. Let x_i and x_j be two neighboring nodes in the graph. We denote by $m_{ij}(x_j)$ the message that node x_i sends to node x_j, by $m_{ii}(x_i)$ the message that y_i sends to x_i, and by $b_i(x_i)$ the belief at node x_i. Consider the general loopy graph in Fig 2.

The max-product update [9] rules are:

$$m_{ij}(x_j) \leftarrow a \max_{x_i} \Psi_{ij}(x_i, x_j) m_{ii}(x_i) \prod_{x_k \in N(x_i) \backslash x_j} m_{ki}(x_i) \qquad (6)$$

$$b_i(x_i) \leftarrow a m_{ii}(x_i) \prod_{x_k \in N(x_i)} m_{ki}(x_i) \qquad (7)$$

where a denotes a normalizing constant and $N(x_i) \backslash x_j$ means all nodes neighboring x_i, except x_j.

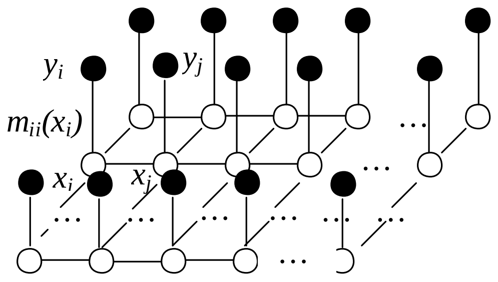

Figure 2: A very loopy graph with hidden nodes and observed nodes (black)

The procedure is initialized with all message vectors set to constant functions. Observed nodes do not receive messages and they always transmit the same vector—if y_i is observed to have value

y^* then $m_{ii}(x_i) = \Psi_{ii}(x_i, y^*)$. The normalization of m_{ij} in equation 6 is not necessary—whether or not the message are normalized, the belief b_i will be identical. However, normalizing the messages avoids numerical underflow and adds to the stability of the algorithm. We assume throughout this report that all nodes simultaneously update their messages.

For singly connected graphs it is easy to show that:

- The algorithm converges to a unique fixed belief regardless of initial conditions in a finite number of iterations.
- At convergence, the belief for any value x_i of a node i is the maximum of the posterior, conditioned on that node having the value x_i: $b_i(x_i) = a \max_x P(x \mid y, x_i)$.
- Define the max-product assignment, x^* by $x_i^* = \arg\max_{x_i} b_i(x_i)$ (assuming a unique maximizing value exists). Then x^* is the MAP assignment.

The max product assignment assumes there is only a unique maximizing x_i exists for all $b_i(x_i)$. Ties can arise when the MAP assignment is not unique, e.g. when there are two assignments that have identical posterior and both maximize the posterior.

3. Generalizing Belief Propagation (GBP)

In ordinary BP, all messages are always from a single node to another node, it is natural to expect that messages from groups of nodes to other groups of nodes could be more informative, and thus lead to better inference. That is the basic intuitive idea behind generalized belief propagation (GBP). The mathematical justification of GBP algorithms is that, if we define messages and message-update rules appropriately, we can show that fixed points of a GBP algorithm are equivalent to the stationary points of a corresponding Kikuchi approximation to the free energy. In this report, we are not going to give the procedure to construct a GBP algorithm in the general case, or the proof of the equivalence to a Kikuchi approximation. Interested readers can refer to [4] for the extended reading. Also, for details of constructing a GBP algorithm by the canonical methods, readers can refer to [5].

4. BP and Junction Tree Algorithm

A standard algorithm for exact inference in graphical model is the Junction Tree Algorithm [12], which will be briefly illustrated in Fig 3. The graph is first triangulated from the original loopy graph (Fig 3a). Fig 3b shows a triangulated version which presents a sufficient condition for a graph to have a junction tree. We then collect all the maximal cliques of the junction tree, and group them together in a tree (Fig 3c). The potentials on the spanning trees are defined so that the trees set the equivalent MRF to the original problem. Once we have the equivalent junction tree, we can run BP on it and obtain marginals of the cliques. Thereby, we can further calculate the marginal of individual node in each clique. The entire process above is so called Hugin Algorithm for undirected graphs. If we have a directed graph, we can add a moralization step before the triangulation.

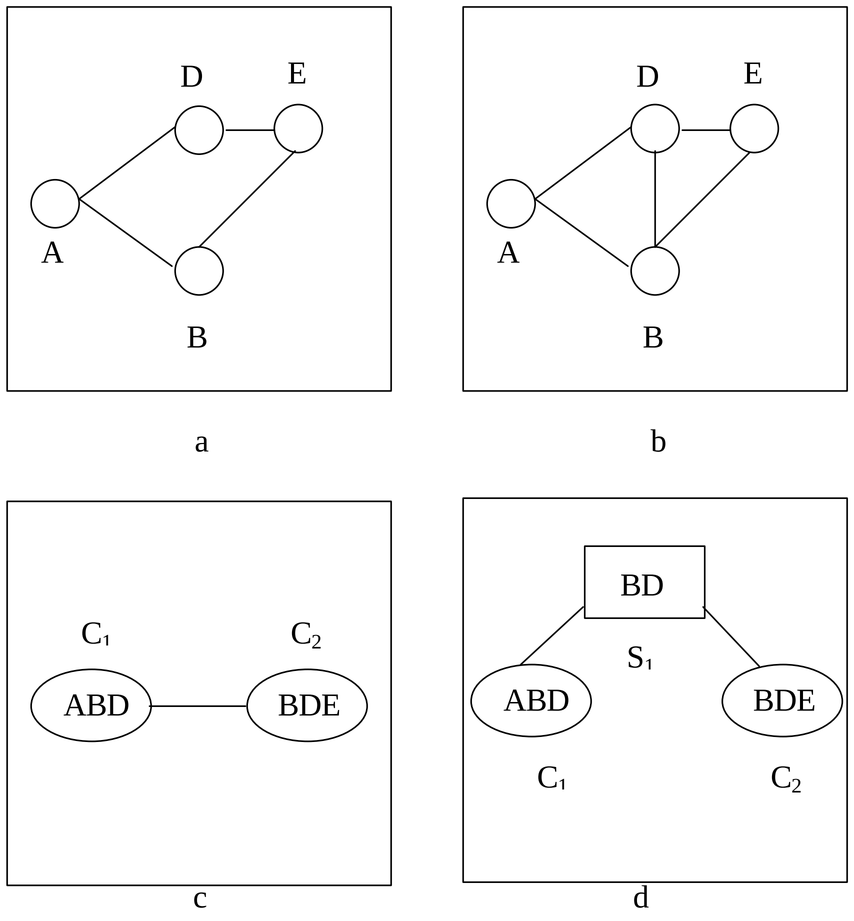

Figure 3: a. an undirected graph with a loop. b: the same graph after triangulation: a chord added. c: the junction tree is a spanning tree

There is another algorithm named Shafer-Shenoy algorithm. In this algorithm, messages between cliques are presented by a function of the state, which is equivalent to running BP on a tree in which we have an extra node between clique nodes (Fig 3.d).

For a graph with small number of loops, the transformation from the original graph to a junction tree could be performed so that we could obtain an exact inference by running BP. However, if there is a very loopy graph such as a grid, we would not decompose the very loopy graph to a more complicated junction tree. Instead, we would rather run the BP on the original loopy graph by ignoring the loops. The tradeoff is that we obtain an approximated inference.

5. Applications of Belief Propagation in Computer Vision

The belief propagation is powerful machinery for learning low-level vision problems, such as motion analysis, unwrapping phase images [3], stereo matching [8], inferring shape and reflectance from photograph [7], or extrapolating image detail [6]. For these problems, given the image data, we want to estimate an underlying scene. 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.

In this report, we will focus on two of these applications which are the noise removal and image segmentation enhancement. The graph models for both of these tasks are typically very loopy ones (ref. 2) because these models are based on images. Each hidden node denotes a pixel or a patch of the image, and each observed node connected to the hidden node has the observed image values (pixels or patches). Although there is no theoretical understanding why the BP sometimes also works well on loopy graphs, the experiments in this project show really promising results.

5.1 Noise Removal using BP

There are a few kinds of noise including salt and pepper noise, white noise. Filters are widely used for removing noise from images. However, a specific filter only can deal with a certain kind of noise. Fig 4 shows two images contaminated with salt and pepper noise and white Gaussian noise.

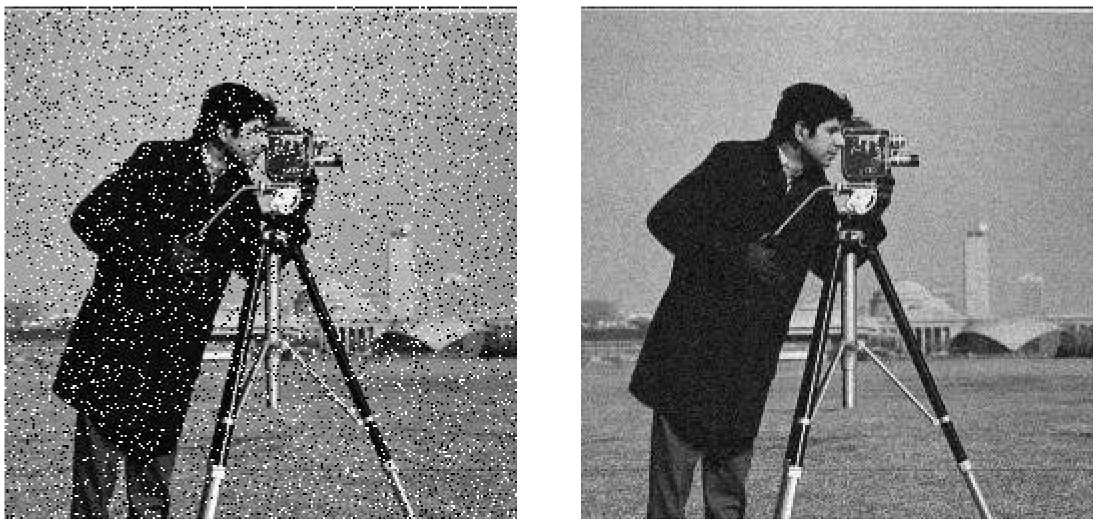

Figure 4: Left. Camera Man with salt and pepper noise; Right. Camera Man with white Gaussian noise

Whenever noise emerges on an image, the piecewise continuous image surface of the underlying scene is not satisfied anymore. Therefore, we could choose the MRF prior [11] for continuous surface as the initialization of the potential functions in BP. In specifying prior cliques potentials for continuous surfaces, only pair-wise clique potential are normally used for piece-wise continuous surfaces. They can be defined by

$$\Psi(x_i, x_j) = g(x_i - x_j) \qquad (8)$$

where $g(x_i - x_j)$ is a function encouraging the smoothness caused by the difference of neighboring pixels. For the purpose of restoration, the function g is generally even, i.e.

$$g(h) = g(-h) \qquad (9)$$

and non-increasing, i.e.

$$g'(h) \leq 0 \qquad (10)$$

We could formulate g using the exponential form (Gibbs distribution)

$$g(h) = v_2 \exp(-|h| / T) \qquad (11)$$

where v_2 is a positive normalizing constant and T is a positive constant called *temperature*.

The intuition of using Eq (11) to model the potential function is the following:

Whenever the difference between the neighboring pixels $|x_i - x_j|$ is small, the potential function would be large to make at both pixels high enough beliefs (the *a posterior* of the original observed state) that the local image patch is smooth enough, and therefore their states keep invariant. Meanwhile, small potential function means the difference between neighboring pixels is large, which disobeys the image smoothness constraint. As a result, we suspect that one of the neighboring pixels is contaminated by noise, whose underlying state is most likely the same as its neighbor. The propagation of message between such pixels gradually "kicks out" the noise.

Certainly, we could also use the following Gaussian distribution (A special case of Gibbs distribution family) for modeling potential function:

$$g(h) = v_2 \exp(-h^2 / 2s^2)$$

where we could let $T = 2s^2$.

The program takes a noisy image as input, and produces a cleaner version. Each node stores information about its neighbors, such as node index, messages to be passed and potentials. Fig 5 shows the results after the BP processing to the two images in Fig 4.

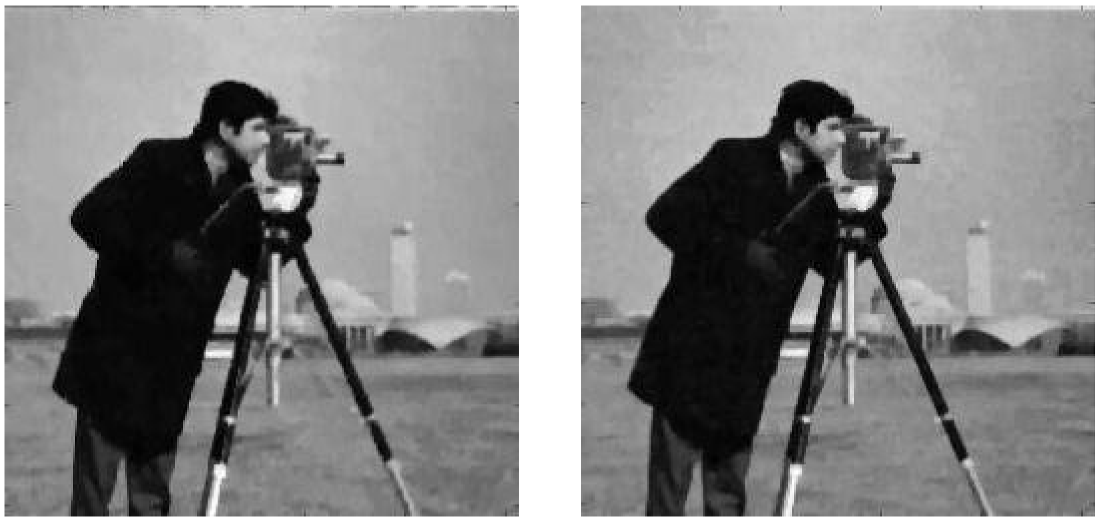

Figure 5: Left. Restored image which had salt and pepper noise; Right: Restored image which had white gaussian noise

Both of the restored images are a little blurry because we encouraged smoothness constraint. Therefore, high frequency details are removed together with noise, which is the drawback of the BP method.

5.2 Image Segmentation Enhancement

Image segmentation is a widely used processing and pre-processing in computer vision tasks. The segmentation result influence the following processing so much that a highly efficient and accurate segmentation algorithm is expected. Unfortunately, there is not an algorithm which is able to segment images perfectly. There are a lot of problems existing, such as over-segmentation, under-segmentation. Usually, merging and splitting algorithms still should be applied for the segmentation enhancement. Fig 6 shows a segmented picture which is over-segmented. The BP algorithm will be used as a merging algorithm in this project.

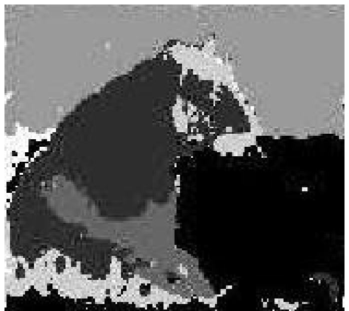

Figure 6: An over-segmented image some patches of which should be merged together.

Instead of using piecewise continuous MRF prior as the model, I use piecewise constant prior because for this problem our input is just an over-segmented image, which has only several labels to indicate different segments. The goal of merging small segments and getting rid of mistakenly segmented patches would be achieved by using BP. The piecewise constant potential function can be defined by

$$\Psi(x_i, x_j) = g(x_i - x_j) \qquad (12)$$

where $g(x_i - x_j)$ is a function only allowing constant labels between neighboring pixels. A delta function is an ideal candidate for defining the function $g(x_i - x_j)$.

$$g(h) = v_2 \exp(-{}^{(1-d(h))}\!/_{T}) \quad (13)$$

where v_2 a positive normalizing constant and T is a positive constant called temperature.

The intuition is very similar to the Eq (11) except that we are only concerned with whether the neighboring pixel is identical, because we want to group pixels as much as possible.

The program takes the segmented image as the input, and runs enough iterations for the propagation of belief. However, we cannot determine when the algorithm will converge. The only guess is that a sufficiently large number of iterations guarantees that the result is quite close to the fixed converging point. Fig 7 shows an enhanced segmentation image with a small number of iterations.

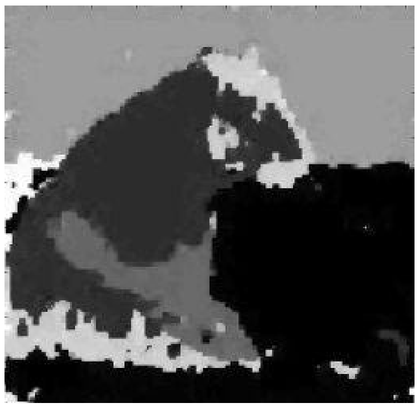

Figure 7: An Enhanced Segmentation Image with a small number of iterations

By comparing Fig 6 and Fig 7, we can see that there is a little improvement for the merging of segments. However, if we run the BP for a larger number of iterations, and also use intermediate results as the new inputs, we could improve the result better. Fig 8 shows a result by taking a large number of iterations.

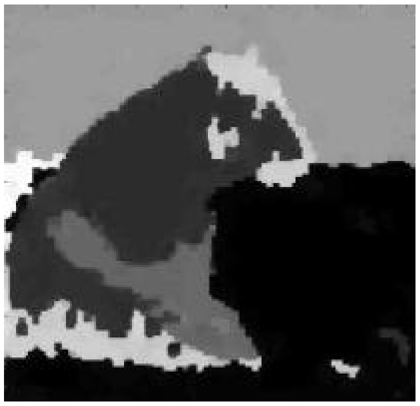

Figure 8: An Enhanced Segmentation Image with a large number of iterations

6. Conclusion

The belief propagation as an inference tool is quite powerful to solve lots of problems in different areas. We in this project focus on the application in computer vision area. The de-noising of image and image segmentation are two of the computer vision tasks which could be performed by the belief propagation. However, the theory of BP is much more than what we have mentioned in this report. The relationship between BP and Kikuchi energy and Bethe energy is one of them. The application using these related theories is much wider. We will study more about these in the future.

Reference

[1] J. Pearl *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[2] Kevin P. Murphy, Yair Weiss, Michael I. Jordan *Loopy Belief Propagation for Approximate Inference: An Empirical Study* In Proc. Uncertainty in AI, 1999.

[3] B. J. Frey, R. Koetter and N. Petrovic. *Very loopy belief propagation for unwrapping phase images*. Neural Information Processing Systems Conference, 2001.

[4] Jonathan S. Yedidia, William T. Freeman, Yair Weiss *Bethe free energy, Kikuchi approximations, and belief propagation algorithm* Available online http://www.merl.com/papers/TR2001-16/

[5] Jonathan S. Yedidia , William T. Freeman, Yair Weiss *Understanding belief propagation and its generalizations* MERL TR-2001-22, January 2002.

[6] William T. Freeman, Egon C. Pasztor, Owen T. Carmichael *Learning Low-Level Vision* MERL TR-2000-05a, July 2000

[7] Yair Weiss *Deriving intrinsic images from image sequences* In Proceedings of International Conference on Computer Vision, 2001

[8] Jian Sun, Heung-Yeung Shum, Nanning Zheng *Stereo Matching Using Belief propagation* In Proceedings of European Conference on Computer Vision, 2002.

[9] Yair Weiss, Willian T. Freeman *On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs* IEEE Transactions on Information Theory 47:2 pages 723-735. 2001.

[10] S.Benedetto, G. Montorsi, D.Divsalar, and F.Pollara. *Soft-out decoding algorithms in iterative decoding of turbo codes.* Technical Report 42-142, JPL TDA, 1996.

[11] Stan Z. Li *Markov Random Field Modeling in Image Analysis.* Springer 2001

[12] Cowell, R. (1998). *Introduction to inference for Bayesian Networks.* In M. Jordan (Ed.), Learning in Graphical Models. MIT Press.

[13] Shafer G.R. and Shenoy P.P. (1990). *Probability Propagation*. Annals of Mathematics and Artificial Intelligence 2:327-352.