A Constant-Space Belief Propagation Algorithm for Stereo Matching*

Qingxiong Yang* Liang Wang† Narendra Ahuja*
*University of Illinois at Urbana Champaign †University of Kentucky
http://vision.ai.uiuc.edu/~qyang6/

Abstract

In this paper, we consider the problem of stereo matching using loopy belief propagation. Unlike previous methods which focus on the original spatial resolution, we hierarchically reduce the disparity search range. By fixing the number of disparity levels on the original resolution, our method solves the message updating problem in a time linear in the number of pixels contained in the image and requires only constant memory space. Specifically, for a 800×600 *image with* 300 *disparities, our message updating method is about* $30\times$ *faster (*1.5 *second) than standard method, and requires only about* 0.6% *memory (*9 *MB). Also, our algorithm lends itself to a parallel implementation. Our GPU implementation (NVIDIA Geforce 8800GTX) is about* $10\times$ *faster than our CPU implementation. Given the trend toward higher-resolution images, stereo matching using belief propagation with large number of disparity levels as efficient as the small ones makes our method future-proof. In addition to the computational and memory advantages, our method is straightforward to implement*[1].

1. Introduction

Stereo vision is becoming more and more mature especially because of publically available performance testing such as the Middlebury benchmark [7], which allows researchers to compare their algorithms against all the state-of-the-art algorithms. The available benchmark suggests that global energy optimization methods such as belief propagation (**BP**) [3, 8] is essential to the state-of-the-art algorithms [5, 14, 1].

The BP algorithm works by passing messages around the graph defined by the four-connected image grid. Until recently, BP was too computationally intensive for real time applications even for low resolution image and small number of disparity levels. Let N be the image size, $\mathcal{L}$ be the number of disparity levels, and T be the number of iterations, the computational complexity is originally $O(4TN\mathcal{L}^2) = O(TN\mathcal{L}^2)$ [8]. Felzenszwalb and Huttenlocher [2] show that the complexity of BP can be reduced to $O(TN\mathcal{L})$ using min convolution method. Besides, [2] hierarchically estimates the messages, and speeds up convergence of the original BP problem. Yang *et. al.* [15] also propose a fast-converging BP method with computational complexity sub-linear in T by updating only the messages of the non-converging pixels. Additionally, [15] shows that the GPU implementation of the Hierarchically BP (**HBP**) method [2] can achieve near video rate (16 FPS) with low resolution image (320×240) and small number of disparity levels (16).

Besides the slow speed, BP is also known to be memory intensive, which makes it difficult to match the data rates of the state-of-the-art cameras which can provide high-resolution stereo image pairs at video rate. For instance, the Bumblebee XB3 camera [4] can capture stereo pairs with resolution 1280×960 and more than 400 disparities at 16 FPS. A standard four-connected BP algorithm requires at least $4N\mathcal{L}$ variables to store all the messages and $N\mathcal{L}$ for the data cost. If stored using 16-bit integers (if hierarchical BP is used), a BP-based algorithm will required at least 4.6GB RAM for such a stereo pair, which makes it impractical to run on ordinary computers.

Several methods have been proposed recently to reduce the memory requirement of standard BP but mostly at the cost of increasing the running time of either message updating [18] or data cost computation [9, 6]. Additionally, most of them only reduce the memory for storing the messages [18, 9]. For standard BP, the memory used to store the data cost is about $1/4$ (or $1/3$ for HBP) of that used for messages, which is still too large for high-resolution images. One exception is the method presented in [6]. Instead of storing the data cost, the data cost is re-computed whenever it is needed.

Yu *et al.* [18] study the feasibility of applying compression techniques to the messages in the BP algorithm

*The support of HP Labs under an Innovation Research Award is gratefully acknowledged. This work was done when Liang Wang† was supported by funding from Dr. Ruigang Yang, and the authors would like to thank Dr. Tianli Yu for his help and discussion on implementing his algorithm [18].

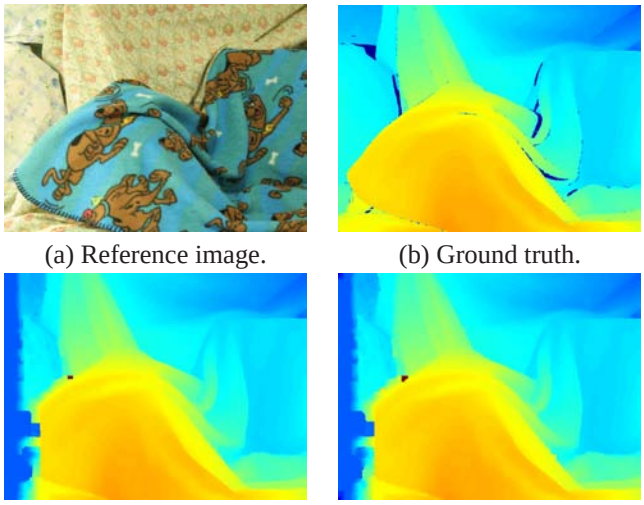
[1] A reference implementation is available at author's website.

978-1-4244-6983-3/10/$26.00 ©2010 IEEE

to improve the memory efficiency. This method can reduce the memory cost to 12.5% for floating point precision or $30 - 50\%$ for 8-bit integer precision. However, the sequential compression and decompression operations would slow the system. Yu *et al.* also present an envelope point transform (EPT) based method. An extra pass of standard BP is required as a pre-processing step in this technique, which makes it slower than standard BP. This EPT method is not suitable for parallel implementation because a sharing buffer is used to update the messages at every pixel. Instead of treating the messages as generic data, Wang *et al.* [9] preserve only a plausible subset of search space for stably matched pixels detected from the results of stereo estimates based on joint bilateral filtering [10]. The preprocessing step for locating the stably matched pixels and the use of joint bilateral filtering stereo matching method (slower than standard BP) slows the system. Additionally, this method is not suitable for parallel implementation either. Liang *et al.* [6] split the MRF into many tiles and perform BP within each one. Global optimality can be preserved by storing the outgoing boundary messages of a tile and use them when performing BP in the neighboring tiles. This method can reduce the memory used to store the data cost. However, the data cost has to be recomputed in each iteration and for each spatial resolution level (if hierarchical implementation is used). [18], [9] and [6] essentially sacrifice speed for memory efficiency. None of them can improve the computational complexity of standard BP. Besides, the memory complexity of these methods is still $O(\mathcal{L})$, although they do reduce the memory cost while maintaining comparable accuracy.

Here, we describe a constant space $O(1)$ BP (**CSBP**) method. The memory cost (including data cost) of our method is independent of the number of disparity levels $\mathcal{L}$. Besides, the runtime of our method is also independent of $\mathcal{L}$ if the computation of data cost is not taken into account. To our knowledge, the presented method is the most efficient BP yet developed. This means that it will perform increasingly better as the number of disparity levels $\mathcal{L}$ increases. Given the trend toward higher-resolution images, and consequently a larger number of disparity levels, stereo matching using BP for large $\mathcal{L}$ as efficient as the small ones makes the described algorithm future-proof.

Unlike previous memory reduction methods which focus on the original spatial resolution, we hierarchically reduce the disparity range to be searched. As concluded in [18], only a small number of disparity levels and the corresponding message values at each pixel (e.g., 2 for *Teddy* data set [7]) are needed to losslessly reconstruct the BP messages. However, there is no an efficient way to compute these disparity levels. Instead of correctly locating these disparity levels, we hierarchically reduce the disparity label set as the spatial resolution increases. This method is very efficient as the number of messages at a coarser level is much smaller compared to at the original level, thus the computation is very cheap even with large number of disparity levels. If we fix the number of disparity levels at the original spatial resolution to be a constant, and repeatedly double it as the spatial resolution decreases, the memory cost of the algorithm is $O(1)$, which is linear in the image resolution and independent of $\mathcal{L}$. Our experiments conducted using the Middlebury data sets [7] show that over 99% pixels with correct disparity values are preserved by our method on average when the number of disparity levels at the original spatial resolution is set to 2. Using this parameter setting, our memory requirement drops linearly from 3.8% of the standard HBP to 0.7% (including the storage of the data cost), as $\mathcal{L}$ increases from $50 - 300$. Unlike previous methods, ours is also computationally efficient. Compared with standard HBP, the speedup factor is $6 - 30$ or $4.2 - 13.4$ if the computation of the data cost is included. Also, our algorithm lends itself to a parallel implementation. Our GPU implementation (NVIDIA Geforce 8800GTX) is about $10\times$ faster than our CPU implementation. Fig. 1 visually compares our method with standard HBP, where (c) and (d) are disparity maps obtained using standard HBP and our method, respectively. Note that there are very few noticeable differences between (c) and (d).



(a) Reference image. (b) Ground truth.

(c) Standard HBP. (d) Ours.

Figure 1. Visual evaluation on *Cloth3* data set [7]. (a) is the reference image, (b) is the ground truth, (c) and (d) are the disparity maps produced using standard HBF and our method with $\mathcal{L} = 150$ disparities. The image resolution is 800×600. The runtime of standard HBP is **24.66** seconds or 25.72 seconds if the computation of the data cost is included, and the memory is 967 MB including the storage of the data cost. In contrast, the runtime of our method is **1.55** seconds or 3.55 seconds if the computation of the data cost is included, and the memory is 13 MB including the storage of the data cost. The speed is tested with a 2.5 GHz Intel Core 2 Duo processor on a DELL XPS laptop computer.

The limitation of our method is that it is generally less accurate than standard HBP around depth discontinuities. To solve this problem, we propose a joint bilateral filtering based post processing method. The computation complexity of this method is independent of $\mathcal{L}$. Besides, there is no additional memory requirement.

2. Hierarchical Belief Propagation

In this section, we briefly review the max-product BP algorithm [11] we have adopted. The max-product BP algorithm works by passing messages around the graph defined by the four-connected image grid. Each message is a vector of dimension given by the number of possible labels $\mathcal{L}$, and at each iteration, the new messages are computed as follows

$$M^t_{\mathbf{X},\mathbf{Y}}(d) = \underset{d_{\mathbf{X}}}{\operatorname{argmin}}(E_{D,\mathbf{X}}(d_{\mathbf{X}}) + \sum_{s\in N(\mathbf{X}),\mathbf{X}\neq\mathbf{Y}} M^{t-1}_{s,\mathbf{X}}(d_{\mathbf{X}}) + h(d_{\mathbf{X}},d)), \quad (1)$$

where $M^t_{\mathbf{X},\mathbf{Y}}$ is the message vector passed from pixel $\mathbf{X}$ to one of its neighbors $\mathbf{Y}$, $E_{D,\mathbf{X}}$ is the data term of pixel $\mathbf{X}$, and $h(d_{\mathbf{X}},d)$ is the jump cost. d is the label that minimizes the total energy for pixel $\mathbf{X}$, which contains the data term and the smoothness term

$$\begin{aligned} E_{\mathbf{X}}(d) &= E_{D,\mathbf{X}}(d) + E_{S,\mathbf{X}}(d) \\ &= E_{D,\mathbf{X}}(d) + \sum_{\mathbf{Y}\in N(\mathbf{X})} M_{\mathbf{Y},\mathbf{X}}(d). \end{aligned} \quad (2)$$

The common cost functions for the jump cost $h(d_{\mathbf{X}},d)$ are based on the degree of difference between labels. In order to allow for discontinuities, the truncated linear model is commonly adopted

$$h(d_{\mathbf{X}},d) = \rho\cdot\mathbf{min}(\mid \mathbf{d_X} - \mathbf{d} \mid, \eta), \quad (3)$$

where ρ is a scalar constant and η is a constant controlling when the cost stops increasing. Eqn.3 is defined under the assumption of piecewise-smooth surfaces. Let $\mathcal{L}$ be the number of disparity levels. In this paper, we set $\eta = \mathcal{L}/8$ and $\rho = 10$.

The global energy is observed empirically to converge after a certain number of iterations [2]. Finally, the label d that minimizes $E_{\mathbf{X}}(d)$ individually at each pixel is selected.

This standard BP algorithm is too slow to be practical. Felzenszwalb [2] proposed a hierarchical algorithm which runs much faster than the previous algorithms while maintaining comparable accuracy. The main difference between HBP and standard BP is that HBP works in a coarse-to-fine manner. The basic steps are: (a) initialize the messages at the coarsest level to all zeros, (b) apply BP at the coarsest level to iteratively refine the messages. (c) use refined messages from the coarser level to initialize the messages for the next level. Specifically, if $\mathbf{X}$ is a pixel at a coarser level, and its corresponding pixels at the finer level are $\mathbf{X}'_i$, $i \in [1,4]$ as shown in Figure 2, then

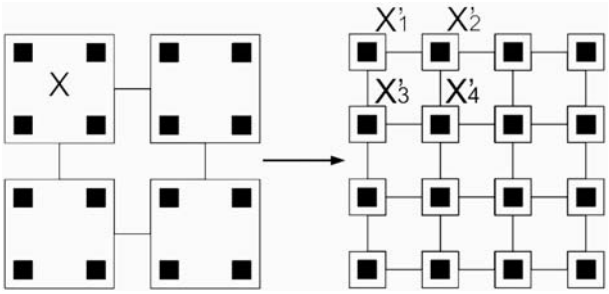


Figure 2. Illustration of two levels in the coarse-to-fine method. Each node $\mathbf{X}$ in left figure corresponds to a block of four nodes $\mathbf{X}'_i$ in the right figure.

$$E_{D,\mathbf{X}} = \sum_{i\in[1,4]} E_{D,\mathbf{X}'_i}, \quad (4)$$

$$M_{\mathbf{X}'_i,\mathbf{Y}'_{i,j}} = M_{\mathbf{X},\mathbf{Y}_j}, i,j \in [1,4], \quad (5)$$

where $Y'_{i,j}$ are the four neighbors of pixel $\mathbf{X}'_i$, and $\mathbf{Y}_j$ are the corresponding four neighbors of pixel $\mathbf{X}$. For instance, assume $Y'_{i,j}$ is the upper pixel of $\mathbf{X}'_i$, then $\mathbf{Y}_j$ is also the upper pixel of $\mathbf{X}$. Hence the number of the messages at the finer lever is four times larger than those at the coarser level. As a result, the messages will be hierarchically refined while the data term stays unchanged, since the data term for the coarser level is the sum of the corresponding four data terms of the finer level as in Eqn. 4. Finally, the refined messages and the data term are used to compute the total energy in Eqn.2.

Two main parameters $\mathcal{S}$ and T define the behavior of this HBP algorithm, $\mathcal{S}$ is the number of levels and T is the number of iterations at each level. In the paper, we experimentally choose $\mathcal{S} = 5$ and $T = 5$.

3. Constant-Space Belief Propagation (CSBP)

We first give a detailed description of our constant-space algorithm in Sec. 3.1, then analyze the complexity in Sec. 3.2 and conclude that the memory required by our algorithm is linear in the image resolution but independent of the maximum disparity $\mathcal{L}$. If the computation of the data term is excluded, the running time of the algorithm is also independent of $\mathcal{L}$. We next evaluate the accuracy of our method by comparing it with standard HBP in Sec. 3.3, and discuss the limitations of our method in Sec. 3.4. We finally present an efficient post-processing method to improve the reconstruction quality in Sec. 3.5.

[2] Loopy BP is not guaranteed to converge.

3.1. Detailed Algorithm

Our approach is inspired by the conclusion in [18] that on average, only a small number of disparity levels and the corresponding message values are needed at each pixel (e.g., 2 for the *Teddy* [7] data set) to losslessly reconstruct the BP messages. This suggests that the message updating step involves much redundant computation. It is computationally expensive to estimate the disparity levels as an extra pass of standard BP will be required. Additionally, the number of required disparity levels will be different from pixel to pixel. As a result, the method presented in [18] is slower than standard BP, and can only reduce the memory cost of BP to about 12.5%.

In our approach, the problem is solved in another manner. We perform BP in a coarse-to-fine manner similar to [2] as presented in Sec. 2. The messages are updated at each level, and then propagated to the finer level and refined there. However, from the conclusion in [18], we know that only a few disparity levels are required. As result, we not only apply the coarse-to-fine scheme to the spatial domain, but also to the depth domain. Specifically, we gradually reduce the number of disparity levels as the messages propagate from coarsest level to original level.

Our CSBP approach is summarized in Alg. 1, which is the same as standard HBP except for steps 1, 2, 8, 9 and 10. Steps 1 and 8 show that unlike standard HBP, where the data term is computed once and stored hierarchically, our method re-computes the data term at each level (but not for each iteration). The disadvantage of our data cost computation method is that there is redundant computation and the speed is a bit slower ($9/8$ as described in Sec. 3.2) than standard HBP. However, the advantage is that the required memory for storing the data term will not depend on the maximum disparity $\mathcal{L}$. This advantage is very important for high-resolution images. For a 1280×960 image pair with 400 disparities, standard HBP requires about 1.5GB RAM to store the data term, which is a big cost for ordinary computers and impractical for embedded systems. Step 9 in Alg. 1 is also different from standard HBP where the energy is computed only once at the fine level. Step 2 and 10 shows that unlike standard HBP where the number of disparity levels is the same for every spatial resolution, our method gradually reduces the number of disparity levels as the spatial resolution increases. Recall that BP works by looking for fixed points of the message update rule. Intuitively, the closer the messages are to the fixed points, the fewer the required number of disparity levels. On the extreme case, if the messages are the same as the fixed points, then only a single disparity level is needed for each pixel, and it is the estimated disparity value. Thus the intuition of reducing the search range hierarchically is based on the behavior of HBP, which updates messages hierarchically, and refines the messages to approach the fixed points.

Algorithm 1 Constant-Space Belief Propagation

1: Compute the data term E_D at the coarsest level (level $\mathcal{S}-1$): $O(N\mathcal{L})$.
2: Only $k_{\mathcal{S}-1}$ smallest data costs are selected and passed to step 5 at each pixel. The other data costs and the corresponding disparity levels are treated as outliers: $O(N\mathcal{L}/8)$.
3: Initialize messages to zero.
4: **for** $s = \mathcal{S}-1$ to 0 **do**
5: -Iteratively update the message vector at each pixel and for each disparity candidate according to Eqn. 1: $O(400N)$.
6: **if** $s > 0$ **then**
7: Initialize the messages for the next level using Eqn. 5 and the selected disparity levels.
8: Compute the data term for the next level using the selected disparity levels: $O(3.5N)$.
9: Compute the total energy at each pixel and for each disparity candidate according to Eqn. 2: $O(3.5N)$.
10: Select $k_{s-1} = k_s/2$ disparity levels and message values corresponding to the k_{s-1} smallest energy values and pass them to step 5 at each pixel. The other disparity levels and messages are treated as outliers and won't be taken into account. *Note that the size of the message vector and the data term at each pixel will be reduced by half at this step*: $O(32N)$.
11: **else**
12: Compute the total energy at each pixel and each disparity candidate according to Eqn. 2.
13: **end if**
14: **end for**
15: The disparity value that minimizes the total energy individually at each pixel is selected.

3.2. Speed and Memory Analysis

If we set the number of disparity levels at the fine level to be $k_0 = 2$ which is independent of the maximum disparity $\mathcal{L}$, then the computational complexity and memory requirement of Alg. 1 is independent of $\mathcal{L}$ except for steps 1 and 2. The required memory in step 1 and 2 is actually also independent of $\mathcal{L}$ since only $k_{\mathcal{S}-1} = k_0 2^{\mathcal{S}-1} = 2^{\mathcal{S}}$ variables are required to be stored at each pixel location. Hence, our method requires only constant memory, and if the computation of the data term is excluded, our method is also constant-time. Constant-space and constant-time means that the complexity of the algorithm remains same even if the disparity range $\mathcal{L}$ becomes very large. We next present the detailed analysis of the complexity of each step in Alg. 1. The computational complexities of each step are

shown in blue text at the end of that step in Alg. 1.

The main computation of Alg. 1 is the iterative message updating at step 5. However, because the selected disparity levels maybe different for neighboring pixels, the min convolution method presented in [2] cannot be directly used to update a message vector, thus the computational complexity of step 5 in Alg. 1 is $O(k_s^2)$ [8]. Assuming the number of spatial resolution levels $\mathcal{S} = 5$ and the number of iterations at each scale is $T = 5$, the computational complexity of updating the messages is $O(\sum_{s=\{0,\cdots,\mathcal{S}-1\}} T \cdot 4(N/4^s)k_s^2 = O(T \cdot 16\mathcal{S}N) = O(T \cdot 80N) = O(400N)$ according to [8]. In HBP, the variables used to store the messages can be repeatedly used at different levels. Our method inherits this property, and requires $4Nk_0 = 8N$ variables. Simultaneously, our method requires extra $Nk_0 = 2N$ variables to store the selected disparity levels. So our message updating method requires a total of $10N$ variables.

Comparing with standard HBP, our message updating method requires three extra steps: step 8, 9 and 10 in Alg. 1. The computational complexity of step 8 is the same as step 9, which is $O(\sum_{s=\{1,\cdots,\mathcal{S}-2\}}(N/4^s)k_{s+1}) = O(3.5N) = O(N)$. Step 8 requires at most $Nk_0 = 2N$ variables which can be borrowed from step 2 to store the data term, and step 9 requires at most $k_{\mathcal{S}-1} = 32$ variables to store the energy values at each pixel. These 32 variables can be repeatedly used at each pixel, thus the extra memory requirement is tiny compared to the other steps. Step 10 is mainly an operation of selecting the k_s smallest values from k_{s+1} values, and can be computed directly in $O(k_s k_{s+1})$ or $O(k_s \log(k_{s+1}))$ time using heap sort. For simplicity, we assume the complexity of this operation is $O(k_s k_{s+1})$. The computational complexity of step 10 is then $O(\sum_{s=\{1,\cdots,\mathcal{S}-2\}}((0.25)^s N)k_s k_{s+1}) = O(32N) = O(N)$. There is no extra memory requirement in step 10.

To sum up, if the number of iterations is $T = 5$, the computational complexity of message updating is $O(400N) + O(3.5N)+O(3.5N)+O(32N) = O(439N) = O(N)$, and the total number of variables required is $10N$. Hence, the computational and memory complexity of our message updating method is linear in the image resolution N. Although our method requires the data term to be re-computed at every level (step 8 in Alg. 1), its computational complexity is $O(3.5N)$, which is only a very small portion of the whole message updating process ($O(439N)$).

However, the computation of the data term at the coarsest level (step 1 and 2 in Alg. 1) is linear in $\mathcal{L}$ and may become the bottleneck of the algorithm. Specifically, the computational complexity of step 1 and 2 is $O(N\mathcal{L}) + O((0.25)^{\mathcal{S}-1}N \cdot k_{\mathcal{S}-1} \cdot \mathcal{L}) = O(N\mathcal{L}) + O(\frac{1}{8}N\mathcal{L}) = O(N\mathcal{L})$. Thus if $\mathcal{L}$ is large, the runtime of step 1 and 2 is comparable to the runtime of the messages updating steps (step $4 - 14$). However, the number of variables used to stored the data term is $k_{\mathcal{S}-1} \cdot (0.25)^{\mathcal{S}-1}N = N/8$. This is much smaller than $2N$, which is the largest number of variables used in step 8. Thus step 8 can share memory with steps 1 and 2 in Alg. 1, and that there is no additional memory requirement for steps 1 and 2.

3.3. Performance Evaluation

We now quantitatively evaluate the accuracy of our method using standard test suite [7]. We numerically compare the disparity maps obtained using standard HBP and our CSBP method, and the results are summarized in Table 1. As can be seen, the accuracy is well preserved in our method. We next evaluate the efficiency of our method with respect to the number of disparity levels $\mathcal{L}$. We use *Cloth3* data set [7] for this experiment. The reference image is presented in Fig. 1 (a). The resolution of the image is 800×600. The ground-truth disparity map is presented in Fig. 1 (b). (c) shows the disparity map obtained using standard HBP, and (d) is the disparity map obtained using our method. $\mathcal{L}$ is set to 150 to obtain (c) and (d). (c) and (d) are visually very similar, which proves the accuracy of our method. Quantitative evaluation of both methods using the ground-truth disparity map presented in Fig. 1 (b) in terms of $\mathcal{L}$ is presented in Table 2.

Algorithm	Data Set			
	Tsukuba	Venus	Teddy	Cones
Standard HBP	1.80%	1.22%	10.4%	5.61%
Ours	2.00%	1.48%	11.1%	5.98%

Table 1. Quantitative evaluation of the performance of the standard HBP and our method on the data sets in [7]. The numbers are the percentage of pixels with misestimated disparities for different subsets of the test data sets.

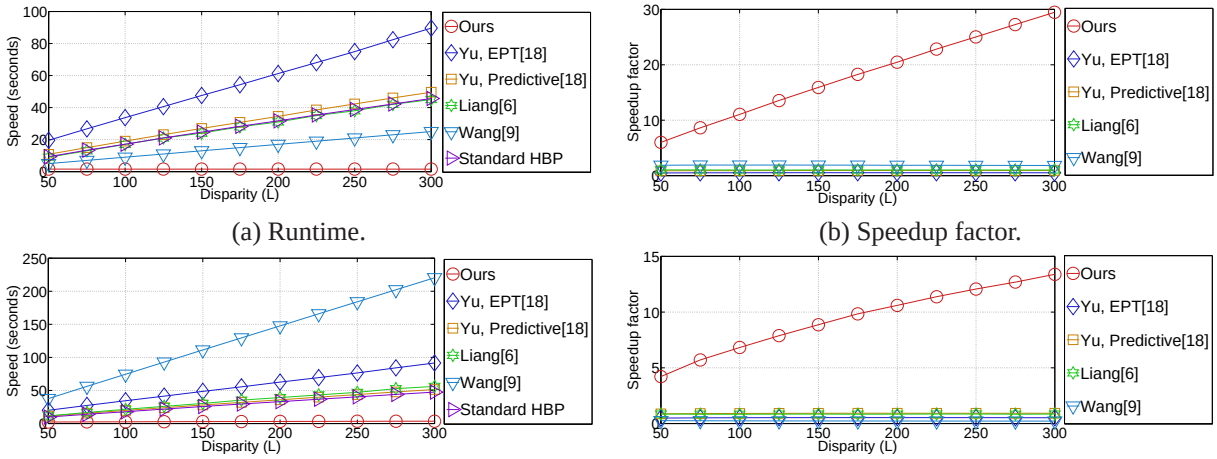
To demonstrate the efficiency of our method, we compare the runtime and memory requirement of different BP methods in Fig. 3 and 4. *Cloth3* data set [7] is used in this experiment. As shown in Fig. 3 (a), our method is independent of $\mathcal{L}$ if the data cost computation is excluded while the runtimes of other methods are linear in $\mathcal{L}$ and generally slower than standard HBP. Fig. 3 (b) shows that the speedup factor of our method is $6 - 30$ as $\mathcal{L}$ increases from $50 - 300$. Fig. 3 (c) compares the speed of the methods when the data cost computation is included. Note that the runtimes of Wang [9] and Liang's [6] methods obviously increase (by comparing to standard HBP), and are larger than standard HBP. Fig. 4 compares the memory requirements. Apparently, our method requires constant memory and the other methods are linear in $\mathcal{L}$.

3.4. Limitations

One of the limitations of our CSBP method is that the runtime for updating the messages (step 5 in Alg. 1) is the same for every spatial resolution ($s = 0, \cdots, \mathcal{S} - 1$). How-

$\mathcal{L}$	125	150	175	200	225	250	275	300
Standard HBP	12.3%	12.3%	12.4%	12.4%	14.1%	19.4%	39.2%	50.9%
Ours	12.7%	12.8%	12.8%	12.8%	14.4%	19.4%	41.4%	50.9%

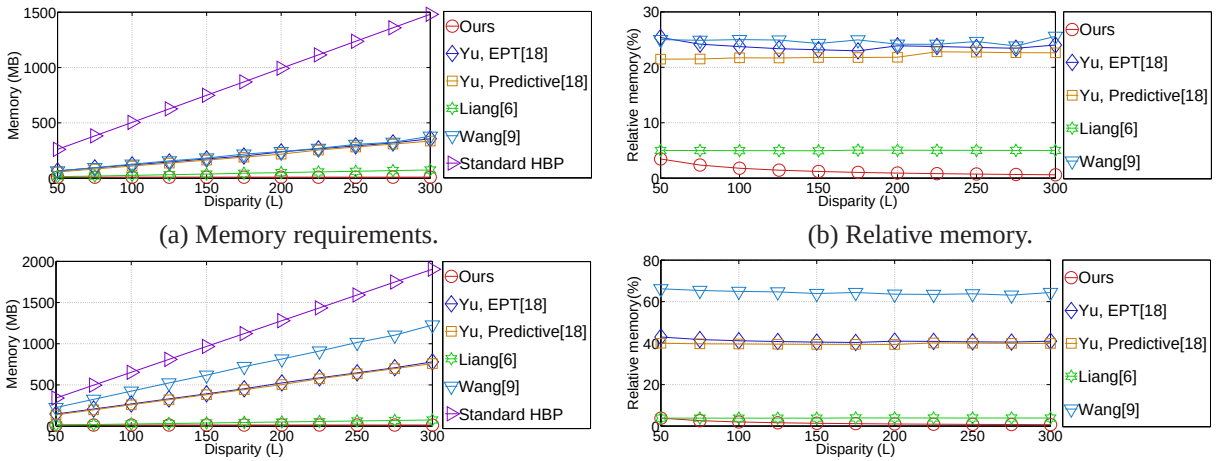
Table 2. Quantitative evaluation on *Cloth3* data set. The numbers are the percentage of pixels with misestimated disparities.



(a) Runtime. (b) Speedup factor.

(c) Runtime (including data cost computation). (d) Speedup factor (including data cost computation).

Figure 3. Comparison of the runtimes of different BP methods using *Cloth3* data set [7]. The resolution of the images is 800×600. (a) shows that the runtime of our method is independent of $\mathcal{L}$ if the computation of data cost is excluded, and (b) shows that the speedup factor of our method is $6 - 30$ as $\mathcal{L}$ increases from $50 - 300$. Note that the method presented in [9] contains a preprocessing step to locate the stably matched pixels. The runtime of this preprocessing step was not included in this experiment.



(a) Memory requirements. (b) Relative memory.

(c) Memory requirements (including data cost). (d) Relative memory (including data cost).

Figure 4. Comparison of the memory requirements of different BP methods using *Cloth3* data set [7]. The resolution of the images is 800×600. Apparently, our method requires constant memory (9 or 13 MB if the storage of the data cost is included) as shown in (a) and (c). However, the memory requirements of other methods are linear in $\mathcal{L}$, *e.g.*, [6] requires 13-77 MB as $\mathcal{L}$ increases from $50 - 300$.

ever, the runtime of standard HBP decreases quadratically with increasing s since the number of disparity levels is the same for all s.

Another limitation is that the accuracy of our method is generally lower than standard HBP around depth discontinuities, since it essentially quantizes the disparity search range of standard HBP. Such an example is presented in Figure 7. (a) is the reference image, (b) is the ground truth, (c) is the disparity map obtained using standard HBP, and (d) is the disparity map obtained using our CSBP method in Alg. 1. The close-up of the white and red rectangles is provided in the upper left and bottom left corners. Apparently, standard HBP can better preserve the depth edges than our method as shown in (c) and (d). For instance, the

pixel marked by a red circle inside the white rectangles has incorrect disparity values in (d). This is actually because the ground-truth disparity value is not selected at the coarsest level (step 2 in Alg. 1) as shown in Figure 5, where the blue circle is the cost value corresponding to the ground-truth disparity value. According to Figure 5, it is not selected at step 2 in Alg. 1, which selects $k_{\mathcal{S}-1} = 32$ disparity values corresponding to the global minimum data cost values at each pixel. Apparently, if the correct disparity values is not selected at the coarsest level (step 2), the estimated disparity value is incorrect. An obvious solution is increasing the number k_s of selected disparity levels at each spatial resolution level s, that is reducing the quantization amount. However, the computational and memory cost will also increase. Another solution is to modify the selection algorithm at step 2 in Alg. 1 to increase the chance of including the ground-truth disparities in the selected disparity set. Instead of using global minima, we first select the local minima, and then global minima if the number of local minima is less than $k_{\mathcal{S}-1}$. The selected values are shown as cyan points in Fig. 6. As can be seen, the ground-truth disparity value (blue circle) is selected. The final disparity map is presented in Fig. 7 (e) which shows that this local minima first selection method is better than the global minima selection method around depth edges. The local minima first selection method is actually a bit faster than global method, and does not require extra storage. However, this method is not suitable for repeated texture, because the number of local minima maybe large and most of them are far away from the ground truth. This method is also less accurate for weakly-textured scenes.

3.5. A Constant-Time Constant-Space Post Processing Method

Visually, the accuracy of the disparity values around depth discontinuities in Fig. 7 (e) is still lower than (c), which is obtained using standard HBP. To further improve the reconstruction quality, [12] demonstrates that joint bilateral filtering [17, 13, 16] can be used to filter the final energy which helps to preserve the depth discontinuity under the assumption that color discontinuity is a strong indicator of depth discontinuity. However, the runtime of this method is linear in $\mathcal{L}$, and generally too slow when $\mathcal{L}$ is large. We thus modify it as a post processing method which is independent of $\mathcal{L}$. Let the disparity map obtained from CSBP be D, reference image be I, and bilateral filter radius be r. For a typical pixel $p = \{\mathbf{x}, \mathbf{y}\}$, assume $\vec{d}_p = \{D(\mathbf{x}-1,\mathbf{y}), D(\mathbf{x},\mathbf{y}-1), D(\mathbf{x}+1,\mathbf{y}), D(\mathbf{x},\mathbf{y}+1)\}$, $\vec{\mathbf{u}}_p = \{\mathbf{x}-r, \cdots, \mathbf{x}+r\}$, $\vec{\mathbf{v}}_p = \{\mathbf{y}-r, \cdots, \mathbf{y}+r\}$, we update the disparity map sequentially as follows

$$D(\mathbf{x},\mathbf{y}) = \underset{d \in \vec{d}_p}{\operatorname{argmin}} \frac{\sum_{\mathbf{u}\in\vec{\mathbf{u}}_p}\sum_{\mathbf{v}\in\vec{\mathbf{v}}_p} W(\mathbf{u},\mathbf{v})C(\mathbf{u},\mathbf{v},d)}{\sum_{\mathbf{u}\in\vec{\mathbf{u}}_p}\sum_{\mathbf{v}\in\vec{\mathbf{v}}_p} W(\mathbf{u},\mathbf{v})}, \tag{6}$$

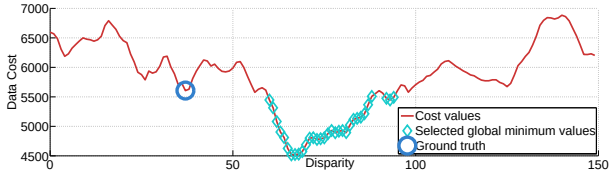


Figure 5. Cost profile of the point in the red circle in Figure 7 (a). The cyan points are the global minimum values selected at step 2 in Alg. 1, and the blue circle is the value corresponding to the ground-truth disparity, and is unselected.

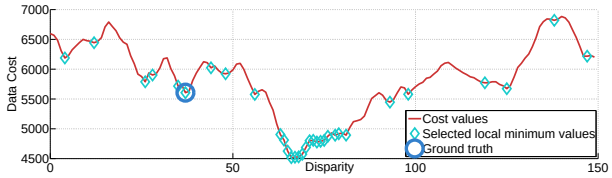


Figure 6. Cost profile of the point in the red circle in Figure 7 (a). The cyan points are the values selected using the local minimum first selection method, and the blue circle is the value corresponding to the ground-truth disparity. Note that the blue circle is selected using this method.

where

$$W(\mathbf{u},\mathbf{v}) = exp(-\frac{||I(\mathbf{x},\mathbf{y}), I(\mathbf{u},\mathbf{v})||_2}{2\sigma_R^2}) \cdot \tag{7}$$

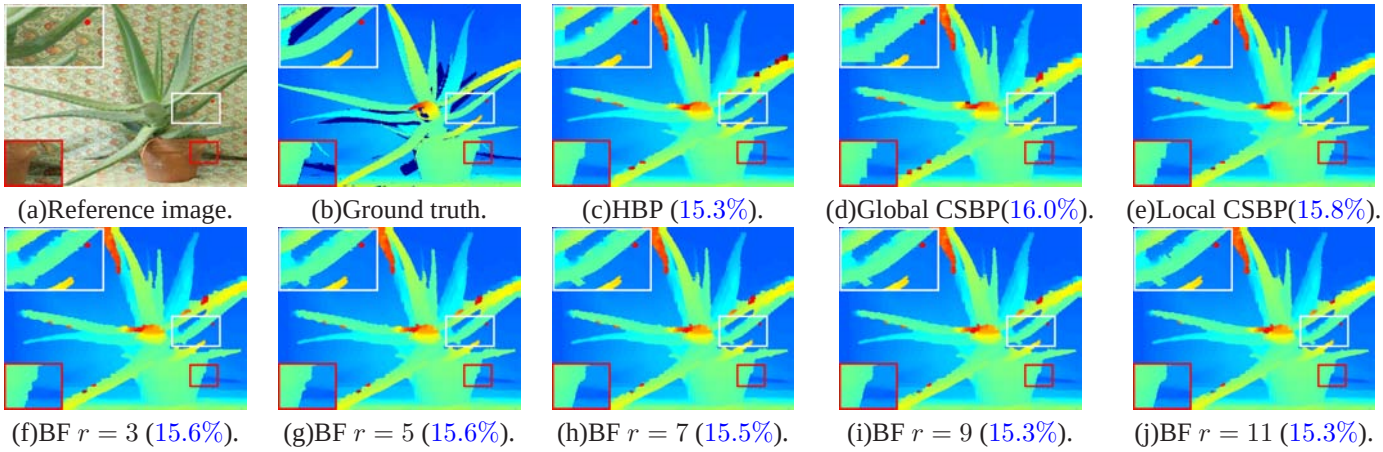
$$exp(-\frac{(\mathbf{x}-\mathbf{u})^2 + (\mathbf{y}-\mathbf{v})^2}{2r^2}), \tag{8}$$

$$C(\mathbf{u},\mathbf{v},d) = \min(\lambda\mathcal{L}, \mid D(\mathbf{u},\mathbf{v}) - d \mid), \tag{9}$$

and $\lambda = 0.2$ is a constant to reject outliers. Setting $\sigma_R = 10$, the post-processed disparity maps of Fig. 7 (e) with different filter radius $r = 3-11$ are presented in Fig. 7 (f)-(j). Note that this post processing method is independent of $\mathcal{L}$ and only valid for pixels around depth discontinuities. It is thus required to check whether a pixel is on a depth edge before processing it. As a result, only a small fraction of the pixels will be processed, and the runtime dependence will be on not only the image resolution but also the image structure. However, it is linear in the filter size ($(2r+1)^2$) excluding the time used to locate depth edges. The exact runtimes (seconds) to obtain the disparity maps shown in Fig. 7 (f)-(j) are 0.05, 0.12, 0.21, 0.32 and 0.47, respectively. Apparently, this post processing method is very efficient even for high resolution disparity maps. The blue numbers below Figure 7 (c)-(j) are the percentages of pixels with misestimated disparities, which numerically prove that the proposed post processing method can effectively improve the reconstruction quality.

4. Conclusions

In this paper, we propose a constant-space BP method. The memory cost (including data cost) of our method is in-



(a)Reference image. (b)Ground truth. (c)HBP (15.3%). (d)Global CSBP(16.0%). (e)Local CSBP(15.8%).

(f)BF $r = 3$ (15.6%). (g)BF $r = 5$ (15.6%). (h)BF $r = 7$ (15.5%). (i)BF $r = 9$ (15.3%). (j)BF $r = 11$ (15.3%).

Figure 7. Evaluation around depth edges. (a): reference image. (b): ground-truth disparity map. (c)-(e): disparity maps obtained using HBP, CSBP with global minima selection method and CSBP with local minima first selection methods, respectively. (f)-(j): post-processed disparity maps obtained using joint bilateral filtering with increasing filter radius presented in Sec. 3.5. The blue numbers under (c)-(j) are the percentages of pixels with misestimated disparities, which show that the local minima first selection method (e) is better than global method (d), and the proposed post processing method can effectively improve the reconstruction accuracy. The percentages of pixels whose correct disparity values are preserved in (d)-(j) are: 99.2%, 99.4%, 99.6%, 99.6%, 99.8%, 100%, 100%.

dependent of the number of disparity levels. Besides, the runtime of our method is independent of the number of disparity levels if the computation of data cost is excluded. Experiments using Middlebury data sets [7] (Table 1, 2 and Fig. 7) shows that our method results in over 99% pixels on an average having correct disparity values. The limitation of our method is that it is less accurate than standard HBP around depth discontinuities. To solve this problem, we propose a very efficient post processing method. The computation complexity of this method is independent of the number of disparity levels, and there is no additional memory requirement.

References

[1] M. Bleyer, M. Gelautz, C. Rother, and C. Rhemann. A stereo approach that handles the matting problem via image warping. In *CVPR*, pages 501–508, 2009. 1

[2] P. Felzenszwalb and D. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1):41–54, 2006. 1, 3, 4, 5

[3] W. T. Freeman, E. Pasztor, and O. T. Carmichael. Learning low-level vision. *IJCV*, 40(1):25–47, 2000. 1

[4] P. Grey. Bumblebee xb3 camera. http://www.ptgrey.com/products/bbxb3/index.asp. 1

[5] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *ICPR*, pages 15–18, 2006. 1

[6] C.K. Liang, C.C. Cheng, Y.C. Lai, L.G. Chen, and H.H. Chen. Hardware-efficient belief propagation. In *CVPR*, pages 80–87, 2009. 1, 2, 5, 6

[7] D. Scharstein and R. Szeliski. Middlebury benchmark. http://vision.middlebury.edu/stereo/. 1, 2, 4, 5, 6, 8

[8] J. Sun, N. Zheng, and H. Y. Shum. Stereo matching using belief propagation. *PAMI*, 25(7):787–800, 2003. 1, 5

[9] L. Wang, H. Jin, and R. Yang. Search space reduction for mrf stereo. In *ECCV*, pages 576–588, 2008. 1, 2, 5, 6

[10] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister. High-quality real-time stereo using adaptive cost aggregation and dynamic programming. In *3DPVT*, pages 798–805, 2006. 2

[11] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47(2):736–744, 2001. 3

[12] Q. Yang, C. Engels, and A. Akbarzadeh. Near real-time stereo for weakly-textured scenes. In *BMVC*, pages 80–87, 2008. 7

[13] Q. Yang, K.-H. Tan, and N. Ahuja. Real-time o(1) bilateral filtering. In *CVPR*, pages 557–564, 2009. 7

[14] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister. Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling. *PAMI*, 31(3):492–504, 2009. 1

[15] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nistér. Real-time global stereo matching using hierarchical belief propagation. In *BMVC*, pages 989–998, 2006. 1

[16] Q. Yang, S. Wang, and N. Ahuja. Svm for edge-preserving filtering. In *CVPR*, 2010. 7

[17] Q. Yang, R. Yang, J. Davis, and D. Nistér. Spatial-depth super resolution for range images. In *CVPR*, 2007. 7

[18] T. Yu, R.-S. Lin, B. S., and B. Tang. Efficient message representations for belief propagation. In *ICCV*, pages 1–8, 2007. 1, 2, 4