Graph Cut based Continuous Stereo Matching using Locally Shared Labels

Tatsunori Taniai*
University of Tokyo, Japan
taniai@iis.u-tokyo.ac.jp

Yasuyuki Matsushita
Microsoft Research Asia, China
yasumat@microsoft.com

Takeshi Naemura
University of Tokyo, Japan
naemura@nae-lab.org

Abstract

We present an accurate and efficient stereo matching method using locally shared labels, a new labeling scheme that enables spatial propagation in MRF inference using graph cuts. They give each pixel and region a set of candidate disparity labels, which are randomly initialized, spatially propagated, and refined for continuous disparity estimation. We cast the selection and propagation of locally-defined disparity labels as fusion-based energy minimization. The joint use of graph cuts and locally shared labels has advantages over previous approaches based on fusion moves or belief propagation; it produces submodular moves *deriving a* subproblem optimality*; enables powerful randomized search; helps to find good smooth, locally planar disparity maps, which are reasonable for natural scenes; allows parallel computation of both unary and pairwise costs. Our method is evaluated using the Middlebury stereo benchmark and achieves first place in sub-pixel accuracy.*

1. Introduction

Recent years have seen significant progress in accuracy of stereo vision. One of the breakthroughs is the use of 3D labels [4, 3, 18, 19, 9]; by estimating a local 3D disparity plane $d = a_p x + b_p y + c_p$ for each pixel, accurate photo-consistency is measured between matching pixels even with large matching windows. While stereo with standard 1D discrete disparity labels [22, 15, 14, 6] can be directly solved by discrete optimizers such as graph cuts (GC) [16, 5] and belief propagation (BP) [26, 7], such approaches cannot be directly used for continuous 3D labels due to the huge (infinite) label space $(a, b, c) \in \mathbb{R}^3$.

Recent successful methods [4, 3, 18] use PatchMatch [1, 2] to efficiently infer correct 3D planes using spatial propagation; each pixel's candidate plane is, in raster-scan order, refined and then propagated to next pixels. Further in [3], this sequential algorithm is combined with BP yielding an efficient optimizer PMBP for pairwise Markov random fields (MRFs) [8]. In terms of MRF optimization, however, BP is considered a *sequential optimizer*, which improves each node individually keeping others conditioned at the current state. In contrast, GC improves all nodes simultaneously by accounting for interactions across nodes, and this global property helps optimization avoid local minima [20, 25]. Nevertheless, incorporating spatial propagation into GC-based optimization is not straightforward, because inference using GC proceeds rather *all-nodes-simultaneously*, not *one-by-one-sequentially* like PatchMatch and BP.

In this paper, we introduce a new labeling scheme, *locally shared labels*, that enables spatial propagation in fusion-based optimization using GC [17]. The locally shared labels define, for each pixel or region, its compact and local discrete label space that is shared among neighboring pixels/regions. By using locally shared labels we generate a number of disparity maps (so-called *proposals* in the literature [17]), and fuse and refine them in an iterative manner (see Fig. 1). For natural scenes that often exhibit locally planar structures, the joint use of locally shared labels and GC has a useful property; it allows multiple pixels in a local region to be assigned the same disparity plane *by a single min-cut* in order to find smooth solutions and to avoid trapped at a bad local minima.

The advantages of our method are fourfold. First, our locally shared labels produce *submodular moves* that guarantee the optimal labeling at each min-cut (subproblem optimal), which in contrast is not guaranteed in general fusion moves [17]. Second, this optimality property and spatial propagation allow randomized search, rather than employ external methods to generate plausible initial proposals as done in previous fusion approaches [17, 25, 19], which may limit the possible solutions. Third, our method achieves greater accuracy than BP [3] thanks to the good properties of GC and locally shared labels. Finally, unlike PMBP [3] the computation of both unary and pairwise costs can be performed in a parallel manner[1], which is the most expensive part in practice. With the proposed approach, accurate

*Part of this work was done while the first author was visiting Microsoft Research Asia as a research intern.

[1] Although BP is usually GPU-parallelizable, PMBP differs from BP's standard settings in that it defines label space *uniquely and distinctively* for each pixel and *propagate* it; both make parallelization indeed non-trivial.

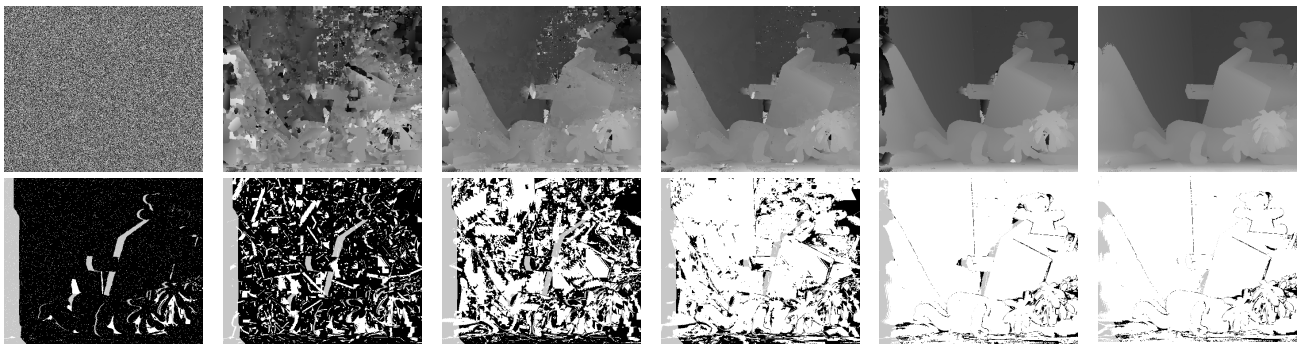


Figure 1. Process of our stereo matching, showing disparity and 0.5-pixel error maps for Teddy. In our framework, we start with random disparities that are represented by per-pixel 3D planes (leftmost). We then alternately propagate local planes using GC and refine them (middles). Finally, the resulting disparity map is further refined at post-processing stage based on left-right consistency check (rightmost).

stereo matching can be efficiently computed with a GPU implementation as we will see in the experiment.

2. Related works

MRF stereo methods can be categorized into three approaches: discrete stereo, segment-based stereo, and continuous stereo.

Discrete stereo [22, 15, 14, 6] formulates stereo matching as a discrete multi-labeling problem, where each pixel is individually assigned one of pre-defined discrete disparity values. For this problem, many powerful discrete optimizers, such as BP [26, 7], TRW [12], and GC [16, 5], can be directly used. Successful results are shown using GC with expansion moves [6, 20]. In expansion moves, the multi-labeling problem is reduced to a sequence of binary-labeling problems, each of which can be exactly solved by GC, if only pairwise potentials ψ meet the following submodularity of expansion moves [13, 6]:

$$\psi(\alpha,\alpha)+\psi(\beta,\gamma)\leq\psi(\beta,\alpha)+\psi(\alpha,\gamma). \qquad (1)$$

Segment-based stereo [21, 10, 11, 23] assigns a 3D disparity plane for each of over-segmented image regions. The candidate planes are generated by fitting planes to a roughly estimated disparity map, and then the optimal assignment of the planes is estimated by, *e.g.*, GC with expansion moves [6, 10] or BP [7, 11]. Although this approach yields continuous-valued disparities, it strictly limits the reconstruction to a piecewise planar representation. Also, results are subject to the quality of the segmentation.

The last group, to which our method belongs, is continuous stereo [25, 4, 3, 19, 18, 9], where each pixel is assigned a distinct continuous disparity value. Some methods [25, 19] use fusion moves [17], an operation that combines two disparity maps to make a better one (binary fusion) by solving a non-submodular binary-labeling problem using QPBO-GC [13, 17]. In this approach, a number of continuous-valued disparity maps (or proposals) are first generated by other external methods (*e.g.*, segment-based stereo [25]), which are then combined as a sequence of binary fusions. Our method is also based on fusion moves but generates proposals using locally shared labels, which enable spatial propagations of local candidate planes and, more importantly, they make fusion moves submodular, *i.e.*, each binary fusion is optimally solved via GC (subproblem optimal). Our method only requires randomized initial proposals instead of those generated by external methods. A stereo method by Bleyer *et al*. [4] proposes accurate photo-consistency measures using 3D disparity planes that are inferred by PatchMatch [1, 2]. Heise *et al*. [9] incorporates Huber regularization into [4] using convex optimization. Besse *et al*. [3] point out a close relationship between PatchMatch and BP and present a unified method called PatchMatch BP (PMBP) for pairwise continuous MRFs. PMBP is probably the closest approach to ours in spirit, but we use GC instead of BP for the inference. Therefore, our method is able to take advantage of better convergence of GC [20] for achieving greater accuracy. In addition, our method allows parallel computation of both unary and pairwise costs.

3. Proposed method

This section describes the proposed stereo matching method. Given two input images I_L and I_R, our purpose is to estimate disparity maps of both images.

3.1. Formulation

We use a pairwise MRF formulation by following conventional stereo matching methods [19, 22, 15, 14, 6]. In the MRF framework, each pixel $p\in(\mathcal{P}\subset\mathbb{Z}^2)$ is assigned a value in some disparity space $\mathcal{S}$, and one seeks a disparity map f for every pixel $f_p=f(p):\mathcal{P}\rightarrow\mathcal{S}$ that minimizes

$$E(f)=\sum_{p\in\mathcal{P}}\phi_p(f_p)+\lambda\sum_{p\in\mathcal{P}}\sum_{q\in\mathcal{N}(p)}\psi_{pq}(f_p,f_q). \qquad (2)$$

The first term, called the *data term* or *unary term*, measures the photo-consistency between matching pixels. The

disparity f_p defines a warp from a pixel p in one image to its correspondence in the other image. The second term is called the *smoothness term* or *pairwise term*, which penalizes discontinuity of disparities of a pixel p and its neighboring pixels $q \in \mathcal{N}(p)$. We define these terms as below.

Data term. To measure photo-consistencies, we use a data term that has been recently proposed by [4]. Here, each pixel p's disparity d_p is over-parameterized by a 3D plane $d_p = a_p x + b_p y + c_p$ to avoid the frontal-parallel bias. Therefore, the objective becomes to seek a disparity plane $f_p = (a_p, b_p, c_p)^T \in \mathcal{S}$ for every pixel p in the left and right images such that disparity map f minimizes the energy function $E(f)$ of Eq. (2). Using this p's disparity plane f_p, a pixel $q = (q_x, q_y)^T$ in the left image is warped to a new location in the right image by a warping w_{f_p} as

$$w_{f_p}(q) = q - (a_p q_x + b_p q_y + c_p, 0)^T. \tag{3}$$

The data term of p in the left image is therefore defined as

$$\phi_p(f_p) = \sum_{q \in W_p} \omega_{pq}\, \rho\left(q, w_{f_p}(q)\right). \tag{4}$$

Here, W_p is a square window centered at p. The weight ω_{pq} implements the adaptive support window proposed in [27], and is defined as

$$\omega_{pq} = e^{-\|I_L(p) - I_L(q)\|_1 / \gamma}, \tag{5}$$

where γ is a user-defined parameter, and $\|\cdot\|_1$ represents the ℓ_1-norm. The function $\rho(q, w_{f_p}(q))$ measures the pixel dissimilarity between q and its matching point $w_{f_p}(q)$ as

$$\begin{aligned}\rho(q, w_{f_p}(q)) = (1-\alpha)\ &\min(\|I_L(q) - I_R(w_{f_p}(q))\|_1, \tau_{col}) \\ &+ \alpha\ \min(\|\nabla_x I_L(q) - \nabla_x I_R(w_{f_p}(q))\|_1, \tau_{grad}),\end{aligned} \tag{6}$$

where $\nabla_x I$ represents the x-component of the gray-value gradient of image I, and α is a factor that balances the weights of color and gradient terms. The two terms are truncated by τ_{col} and τ_{grad} to increase the robustness for occluded regions. We use linear interpolation to compute $I_R(w_{f_p}(q))$. When the data term is defined on the right image, we swap I_L and I_R in Eqs. (5) and (6), and add the disparity value in Eq. (3).

Smoothness term. For the smoothness term, we use a curvature-based, second-order smooth regularization term [19] defined as

$$\psi_{pq}(f_p, f_q) = \max(\omega_{pq}, \epsilon)\ \min(\bar{\psi}_{pq}(f_p, f_q), \tau_{dis}), \tag{7}$$

where ϵ is a small constant value that gives a lower bound to the weight ω_{pq} for increasing the robustness. The function $\bar{\psi}_{pq}(f_p, f_q)$ penalizes the discontinuity between f_p and f_q in terms of disparity as

$$\bar{\psi}_{pq}(f_p, f_q) = |d_p(f_p) - d_p(f_q)| + |d_q(f_q) - d_q(f_p)|, \tag{8}$$

where $d_p(f_q) = a_q p_x + b_q p_y + c_q$. $\bar{\psi}_{pq}(f_p, f_q)$ is truncated by τ_{dis} to allow sharp jumps in disparity at depth edges. The term $\psi_{pq}(f_p, f_q)$ satisfies the submodularity of Eq. (1) for taking advantage of GC. Visualization and proof for submodularity of this term are shown supplementary.

3.2. Locally shared labels

As the main contribution of this paper, we introduce locally shared labels for efficiently optimizing continuous MRFs. The locally shared labels are the combination of pixel and region labels, in which label spaces are *shared* among neighbors, and they enable per-pixel estimation of continuous solutions as well as fast propagations.

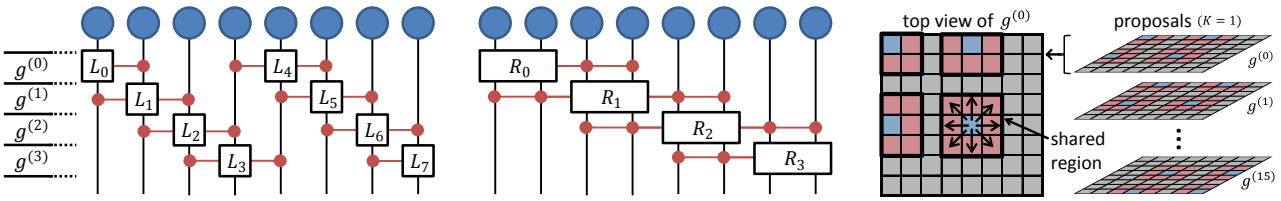
Pixel and region labels. Pixel labels are a small number (say, K) of discrete disparity labels (or candidate labels) defined at each pixel p, which we refer to as a *pixel label set*, $L_p = \{l_p^{(0)}, l_p^{(1)}, \ldots, l_p^{(K-1)}\}$, $l_p^{(i)} = (a, b, c)^T \in \mathcal{S}$. The pixel label sets are shared among neighboring pixels. In addition, we define region labels that give additional candidate labels for accelerating spatial propagation and avoiding stuck at a local minima. We use a regular grid structure for regions, which are indexed by the region coordinates $r \in (\mathcal{R} \subset \mathbb{Z}^2)$ like the pixel coordinates. Region labels define for a region r a set of K_R candidate labels $R_r \subset \mathcal{S}$, which we call a *region label set*. Each label set R_r gives candidate labels for pixels in the region r and pixels in the neighboring regions as well, *i.e.*, R_r is also shared among neighboring regions just like pixel labels.

During the inference, for each pixel p, our method chooses the best candidate label f_p from the union of pixel and region label sets that are shared for the pixel p:

$$C_p = L_p \cup \left(\bigcup_q L_q\right) \cup R_r \cup \left(\bigcup_s R_s\right), \tag{9}$$

where q, r, s represent p's neighboring pixels, the region that p belongs to, and the neighboring regions to r, respectively. By sharing local label sets among neighbors, good candidate labels are spatially propagated to nearby pixels. The concept of pixel and region labels is illustrated in Fig. 2.

Proposal generation for fusion. During the inference, we repeatedly seek the best labeling $f^{(t)}$ for the current local label sets $\{L_p\}$ and $\{R_r\}$, and refine them. The former part, *i.e.*, the selection and propagation of candidate labels in $\{L_p\}$ and $\{R_r\}$, is cast as fusion-based energy minimization [17], as described in the rest of this section. Consider the essential function of fusion is to make a good solution f by fusing a number of proposal disparity maps $\{g^{(0)}, g^{(1)}, \ldots, g^{(n-1)}\}$, where at each pixel f_p is assigned one of n disparity labels $\{g_p^{(0)}, g_p^{(1)}, \ldots, g_p^{(n-1)}\}$. In our method, we build a special form of proposals from $\{L_p\}$ and $\{R_r\}$ in a manner that achieves the propagation of pixel and region labels. To make proposals from pixel labels, we



($K=1$) (a) Pixel labels (1D image) (b) Region labels (1D image) (c) Proposals in case of pixel labels

Figure 2. Illustrations of pixel and region labels, and proposal construction. For simplicity, they are illustrated by 1D images with the blue nodes representing pixels. The boxes L_p and R_r represent a set of candidate disparity labels given for the pixel p and pixels in the region r, respectively. The horizontal red lines signify that the local label sets $\{L_p\}$ and $\{R_r\}$ are shared between neighbors. The local label sets are aligned so as to make proposal disparity maps $g^{(j)}$ for fusion. In particular, proposal construction of pixel labels is illustrated for a 2D image in (c) indicating that, in each proposal $g^{(j)}$, candidate labels defined at blue pixels are shared between red neighbors.

copy each candidate label $l_p^{(i)} \in L_p$ to the j-th proposal $g^{(j)}$ by setting $g_q^{(j)} \leftarrow l_p^{(i)}$, where q is the nine neighboring pixels around and including p (*i.e.*, the pixels where the candidate label $l_p^{(i)}$ is shared), and j is given as

$$j = K\ (4\,(p_y \bmod 4) + (p_x \bmod 4)) + i. \qquad (10)$$

Here, mod is the modulo operation, and p_x and p_y are p's coordinates specified as $p_x \in [0, \text{width} - 1]$ and $p_y \in [0, \text{height} - 1]$. Figure 2a illustrates this construction for the case of a 1D image with $K = 1$ (*i.e.*, $L_p = \{l_p^{(0)}\}$) for simplicity, where a horizontal layer of candidate labels at vertical position j represents a proposal $g^{(j)}$. Figure 2c illustrates for a 2D image showing that candidate labels at blue pixels are shared among red neighbors in each proposal $g^{(j)}$. The integer 4 in Eq. (10) means that, in each proposal, we leave a "gap" (shown as gray pixels in Fig. 2c that represent no candidate labels) between each "shared region" (see Fig. 2c) for ensuring submodularity, which we describe later. We assign an infinite unary cost to those invalid labels to ensure that such labels are avoided during the inference. For region labels, proposals are constructed in the same manner with pixel labels by regarding a region as a pixel as shown in Fig. 2b. The fusion is performed using the proposals generated from both pixel and region labels.

This particular form of proposal construction guarantees that a binary fusion of an arbitrary solution f and any of the proposals $g = g^{(j)}$ is submodular, thus it is *exactly* solved via GC. To show this, we give an intuitive explanation here due to the limited space. A complete proof is in supplementary. Let's consider a simple case where g_p takes the same value for all pixels p. A fusion move with such globally-constant g is called an expansion move [6], which is submodular if pairwise terms ψ_{pq} satisfy Eq. (1). Seeing Fig. 2c, our proposal g is made *locally-constant* by "shared regions", thereby a fusion move with our proposals is, virtually, many mutually-disjoint *local-alpha-expansions* with different "alpha" for each shared region. Therefore the condition for deriving submodularity is the same with Eq. (1), which holds for our smoothness term ψ_{pq} of Eq. (7). With this submodularity guarantee, we only need to use a standard GC [16, 5] instead of employing expensive QPBO-GC [13] used in usual fusion approaches [17, 19, 25].

Algorithm 1 Overview of optimization procedure

1: Initialize $\{L_p\}$ and $\{R_r\}$ randomly.
2: **repeat**
3: ♢ **Optimize labeling f for current local label sets**: $f^{(t)} = \text{argmin}\, E(f)$ with local label sets $\{L_p\}$ and $\{R_r\}$
4: ♢ **Refine local label sets $\{L_p\}$ and $\{R_r\}$**:
5: **for all** pixels $p \in \mathcal{P}$ **do**
6: $\tilde{C}_p \leftarrow C_p$ with perturbation.
7: $L_p \leftarrow$ best $K-1$ candidate labels $c \in (L_p \cup \tilde{C}_p) \setminus \{f_p^{(t)}\}$ that minimize $E_p(c|f^{(t)})$
8: $L_p \leftarrow L_p \cup \{f_p^{(t)}\}$
9: **end for**
10: **for all** regions $r \in \mathcal{R}$ **do**
11: $R_r \leftarrow$ random K_R candidate labels from $\{f_p^{(t)} | p \text{ in } r\}$
12: **end for**
13: **until** convergence

In addition, this proposal generation helps obtain smooth solutions because multiple pixels in shared regions are allowed to move-at-once to the same candidate label at one binary fusion. This effect becomes more significant with region labels because of their large shared regions. In fact, region labels make the key factor in our algorithm for both efficiency and accuracy as we will see in the experiment.

3.3. Optimization

The overview of our optimization procedure is summarized in Algorithm 1. As discussed in the previous section, our optimization uses an iterative framework, where we alternately optimize the labeling f with given local label sets $\{L_p\}$ and $\{R_r\}$, and refine the local label sets $\{L_p\}$ and $\{R_r\}$ locally with the labeling f fixed.

Our optimization begins with randomly initializing $\{L_p\}$ and $\{R_r\}$. To sample the allowed solution space evenly, we take the initialization strategy described in [4]. For $l_p^{(i)} \in L_p$ at $p = (p_x, p_y)^T$, we select a random disparity z_0

in the allowed disparity range $[0, \mathrm{dispmax}]$. Then, a random unit vector $n=(n_x, n_y, n_z)^T$ and z_0 are converted to the plane representation by $a_p = -n_x/n_z$, $b_p = -n_y/n_z$, and $c_p = -(n_x p_x + n_y p_y + n_z z_0)/n_z$. For the region label sets $\{R_r\}$, we randomly pick K_R pixels in each region, and copy the candidate label $l_p^{(0)} \in L_p$ of the randomly chosen pixels p to the region label set.

At line 3 of Algorithm 1, we optimize the labeling f by the procedure illustrated in the previous section. It can be approximately solved by sequentially fusing proposals constructed from pixel and region label sets $\{L_p\}$ and $\{R_r\}$.

In lines 5–9, we refine the pixel label sets $\{L_p\}$. At each pixel p, we first randomly perturb p's candidate labels C_p of Eq. (9) and obtain $\tilde{C}_p$. As the refined L_p, we select the best K candidate labels from the union of $\tilde{C}_p$ and the current L_p that minimize the following local energy at the pixel p:

$$E_p(s|f^{(t)}) = \phi_p(s) + \sum_{q \in \mathcal{N}(p)} \psi_{pq}(s, f_q^{(t)}) \ , \ s \in \mathcal{S}. \quad (11)$$

Here, the refined L_p is forced to contain the current candidate label $f_p^{(t)}$ to ensure that, in the next iteration, the solution $f^{(t+1)}$ can stay at $f^{(t)}$, thereby the energy does not increase, *i.e.*, $E(f^{(t)}) \geq E(f^{(t+1)})$ holds throughout the iterations. Perturbation is implemented as described in [4]. Namely, each candidate label $(a, b, c)^T \in C_p$ is converted to disparity d and normal vector n. We then add a random disparity $\Delta_d \in [-r_d, r_d]$ and a random unit vector Δ_n to them, respectively, as $d' = d_p + \Delta_d$ and $n' = n + r_n \Delta_n$. Finally, d' and $n'/|n'|$ are converted to the plane representation $(a', b', c')^T \in \tilde{C}_p$ as a perturbed candidate label. The values r_d and r_n define an allowed change of planes. We start by setting $r_d \leftarrow \mathrm{dispmax}/2$ and $r_n \leftarrow 1$. After each iteration, we update them by $r_d \leftarrow r_d/2$ and $r_n \leftarrow r_n/2$.

In lines 10–12, we update the region label sets $\{R_r\}$. As done in the initialization, we again take a random-pick-up scheme. This time, the current solution $f_p^{(t)}$ of randomly chosen pixels p is taken as the region labels.

Finally, after the whole process, we perform the post-processing using left-right consistency check and median filtering as described in [4] for further improving the results. This step is widely employed in recent methods [4, 3, 18, 9].

4. Experiments

In the experiments, we first evaluate our method on the Middlebury benchmark. We further assess the effect of region labels, and also compare with the PMBP method [3] that is closely related to our approach.

Setting. We use the following settings throughout the experiments. We use a PC with a Xeon CPU (2.53 GHz $\times$ 4 cores) and NVIDIA GeForce GTX-295 GPU (only one of two GPU cores is used here). The parameters of our data term are set as $\{\tau_{col}, \tau_{grad}, \gamma, \alpha\} = \{10, 2, 10, 0.9\}$ as specified in [4]. The size of supporting windows is set to 41×41, which is the same setting as PMBP [3]. For the smoothness term, we use $\{\lambda, \tau_{dis}, \epsilon\} = \{20, 1, 0.01\}$ and eight neighbors for $\mathcal{N}$. For optimization, three-layer locally shared labels are used: pixel labels with $K = 2$, region labels of size 5×5 with $K_R = 2$, and also regions labels of size 25×25 with $K_R = 2$, and a GC implementation of [5] is used. We iterate twice for each proposal in fusion stage, and iterate the outer-loop process ten times. The computation of unary costs is performed in parallel on GPU, and pairwise costs are computed on four CPU cores.

4.1. Evaluation on the Middlebury benchmark

We show in Tab. 1 selected rankings on the Middlebury stereo benchmark for 0.5-pixel accuracy. Our method achieves the current best average rank (3.5) and bad-pixel-rate (6.63%) amongst more than 155 stereo methods. Even without post-processing, our method still outperforms the other methods in average rank, despite that methods [4, 3, 18, 9] use the post-processing. Compared with closely related approaches (PMBP [3] and PatchMatch stereo [4]), which are ranked seventh and ninth in Tab. 1, although results of PMBP for Cones are slightly better than ours, our method consistently outperforms the two methods in the other evaluations. We summarize the results of our method in Fig. 3. We omit the results of Tsukuba, because its ground truth is quantized to integer disparities; thus we consider that not to be appropriate for sub-pixel evaluation. More results including failures are shown in supplementary.

4.2. Effect of region labels

To observe the effect of region labels, we assess the performance using three different settings: (1) only pixel labels with $K = 6$; (2) pixel labels with $K = 4$ and region labels of size 5×5 with $K_R = 2$; (3) pixel labels with $K = 2$, region labels of size 5×5 with $K_R = 2$, and also regions labels of size 25×25 with $K_R = 2$ (the default setting for our method described above). Among the three settings, the number of candidate labels given for each pixel (*i.e.*, $|C_p|$) is kept consistent. We use $\lambda = 40$ but keep the other parameters as default. Using these settings, we observe the performance variations by estimating the disparities of only the left image of the Cloth1 dataset without the post-processing.

In Fig. 4, we show the results of the three cases after ten iterations. Also, plots in Figs. 5a-c show the energy variations of the energy function, data term, and smoothness term after each iteration, respectively. Figure 5a shows that region labels play a critical role in minimizing energies. Figures 5b and 5c indicate that region labels effectively reduce the energies of both the data and smoothness terms, which shows the contribution of region labels for fast propagation of good candidate labels as we intended. On the other hand, if only pixel labels are used, the solution is trapped at a bad

Algorithm	Avg. Rank	Tsukuba nonocc	Tsukuba all	Tsukuba disc	Venus nonocc	Venus all	Venus disc	Teddy nonocc	Teddy all	Teddy disc	Cones nonocc	Cones all	Cones disc	Average Percent Bad Pixels
1. OUR METHOD	3.5	5.04 2	5.56 2	14.0 9	0.66 2	0.88 2	5.82 4	**4.20** 1	**7.12** 1	**12.9** 1	3.77 5	9.16 5	10.4 8	6.63
2. PM-Huber [9]	5.4	7.12 9	7.80 8	13.7 7	1.00 8	1.40 9	7.80 12	5.53 3	9.36 2	15.9 4	**2.70** 1	**7.90** 1	**7.77** 1	7.33
3. SubPixSearch	6.2	5.60 3	6.23 3	9.46 3	1.07 10	1.64 10	7.36 8	6.71 6	11.0 4	16.9 6	4.02 8	9.76 6	10.3 7	7.51
4. PMF [18]	8.6	11.0 29	11.4 26	16.0 24	0.72 4	0.92 3	5.27 3	4.45 2	9.44 3	13.7 2	2.89 2	8.31 3	8.22 2	7.69
7. PMBP [3]	13.2	11.9 40	12.3 36	17.8 43	0.85 6	1.10 4	6.45 6	5.60 4	12.0 6	15.5 3	3.48 3	8.88 4	9.41 4	8.77
9. PatchMatch [4]	20.3	15.0 57	15.4 56	20.3 69	1.00 9	1.34 8	7.75 11	5.66 5	11.8 5	16.5 5	3.80 6	10.2 7	10.2 6	9.91
*. w/o post-proc.	4.3	5.15 2	5.82 2	14.0 9	0.73 4	1.02 3	6.65 6	4.65 2	10.8 3	14.3 2	3.88 6	9.71 5	10.7 8	7.29

Table 1. Middlebury benchmark for 0.5-pixel accuracy. Our method achieves the current best average rank 3.5. In *all*, results are evaluated for all pixels where the ground truth is given, while only for non-occluded pixels in *nonocc*, and around depth discontinuities in *disc*.

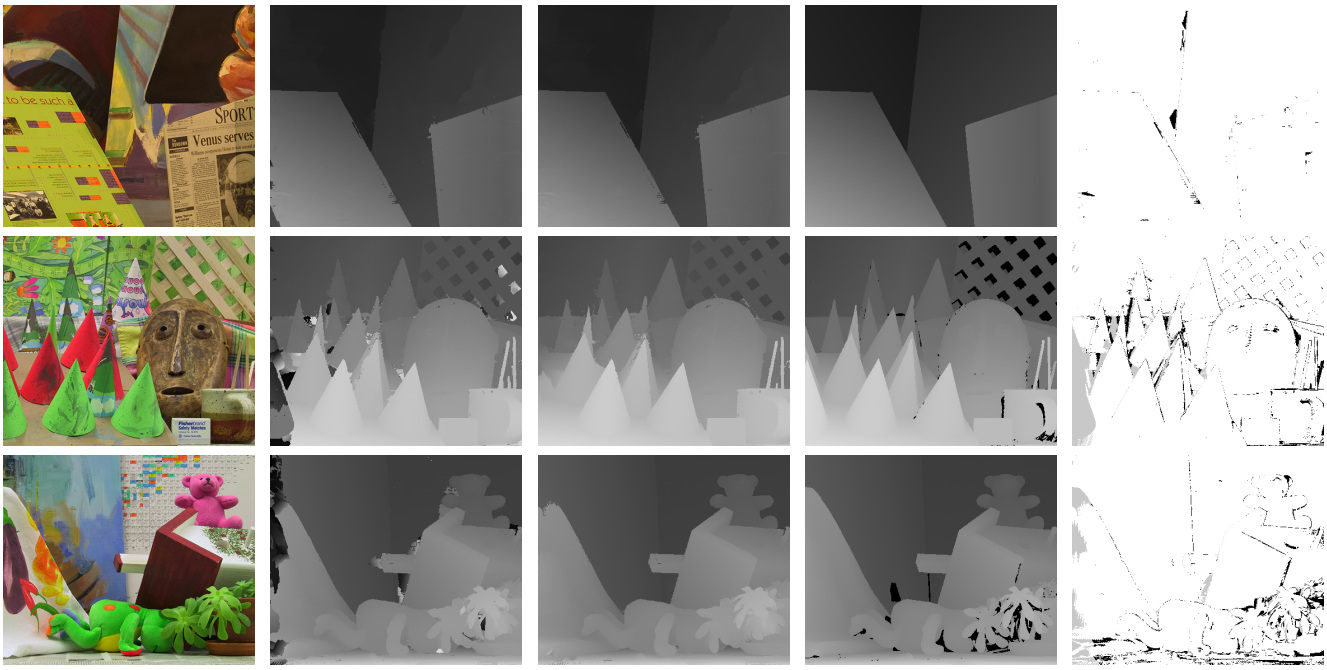


Image | Raw result | After post-processing | Ground truth | 0.5-pixel error map

Figure 3. Our results on the Middlebury benchmark. From left to right, one of the input images, our results without post-processing, after post-processing, the ground truth, and 0.5-pixel error maps of the results after post-processing are shown. In the error maps, white and black pixels indicate correct and incorrect disparities, while gray indicates incorrect but occluded pixels.

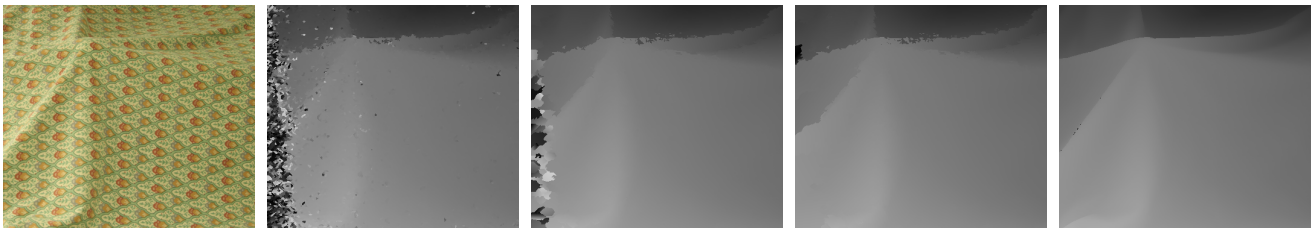
local minima producing a noisy result as in Fig. 4b. Although the use of the large region labels does not make a significant difference in the converged energy values, the difference is obvious if we see Figs. 4c and 4d.

4.3. Comparison with PMBP [3]

We compare our method with PMBP [3] that is the closest method to ours. For a fair comparison, we use four neighbors for $\mathcal{N}$ in Eq. (2), which is the same setting as PMBP. For a comparable smoothness weight with the default setting (eight-neighbor $\mathcal{N}$), we use $\lambda = 40$ and keep the other parameters as default. For PMBP, we use the same model as ours; the only difference from the original PMBP is the smoothness term, which does not satisfy the submodularity of Eq. (1). PMBP also defines K candidate labels for each pixel, for which we set $K = 1$ and $K = 5$ (original paper uses $K = 5$). We show the comparison using the Cones dataset by estimating the disparity map of only the left image without the post-processing.

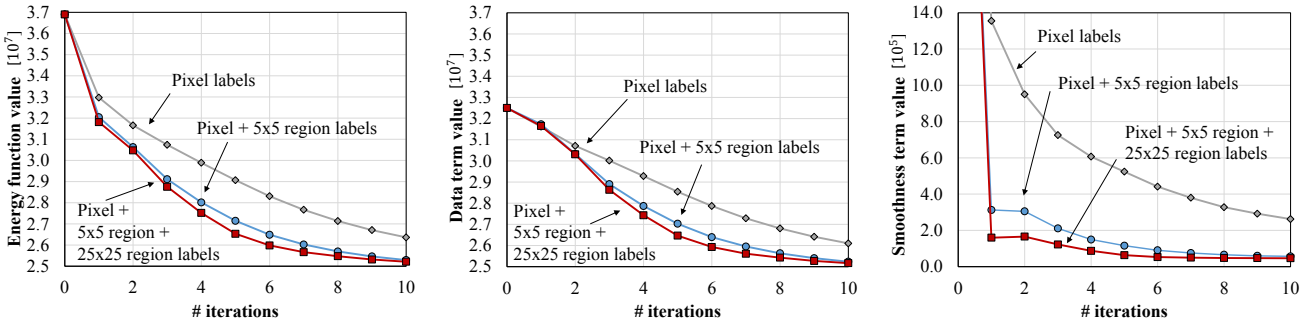
Figures 6a-c show the temporal transition of the energy values in the full and zoomed scales, and the 0.5-pixel error rates, respectively. We show the performance of our method using its GPU and CPU (1 or 4 CPU-cores) implementations. For PMBP, we also implemented the unary cost computation on GPU, but it became rather slow, possibly due to the overhead of data transfer. Efficient GPU implementations for PMBP are not available in literature[2]. Therefore,

[2] GPU-parallelization schemes of BP are not directly applicable due to PMBP's unique settings. The "jump flooding" used in the original PatchMatch [1] reports 7x speed-ups by GPU. However, because it propagates candidate labels to distant pixels, it is not applicable to PMBP that must propagate messages to *neighbors*, and is not as efficient as our 100x, either.



(a) *Cloth1* (b) Only pixel labels (c) Pixel and region (5×5) labels (d) Pixel and region (5×5 and 25×25) labels (e) Ground truth

Figure 4. Visual effect of region labels. (a) One of input images. (b) Using only pixel labels yields a noisy result, which is improved by (c) adding region labels. (d) Large region labels are effective for occluded regions. These are all raw results without the post-processing.



(a) Energy function (b) Data term (c) Smoothness term

Figure 5. Effect of region labels in minimizing energies. From left to right, energy transitions of the energy function, data term, and smoothness term w.r.t. the number of iterations are shown. Region labels play a critical role in reducing the energies.

the plots show PMBP results that use a single CPU core. Figures 6a and 6b show that PMBP works much faster than our CPU implementation; however, our GPU implementation shows significantly faster convergence. Furthermore, our method reaches the better solution than that of PMBP in both energy values and error rates[3]. At around 4000[sec] of Figs. 6b and 6c, the solution obtained by our CPU implementation marked the lower error rate than that of PMBP in spite of its higher energy. Figure 7 shows the resulting disparity maps obtained by our method and PMBP with $K = 5$. Our result shows greater accuracy around the edge regions.

5. Conclusions

In this paper, we presented an accurate and efficient stereo matching method for continuous disparity estimation. Unlike previous approaches that use fusion [17, 19, 25], our method is subproblem optimal and only requires randomized initial proposals. By comparing with a recent continuous MRF stereo method, PMBP [3], our method showed an advantage in efficiency and comparable or greater accuracy. The use of a GC-based optimizer makes our method advantageous.

[3]The CPU implementation of our method also reaches almost the same energy and error rate after about 42000[sec] by a single core, 9150[sec] by four cores.

We believe that our optimization method can be applied for more general corresponding field estimation such as optical flow, but we leave it for our future work. We also believe that some occlusion handling schemes based on GC optimization [14, 15, 24] can be incorporated into our framework, which may yield even greater accuracy.

References

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: a randomized correspondence algorithm for structural image editing. *Proc. of SIGGRAPH (ACM Trans. on Graph.)*, 28(3):24:1–24:11, 2009. 1, 2, 7

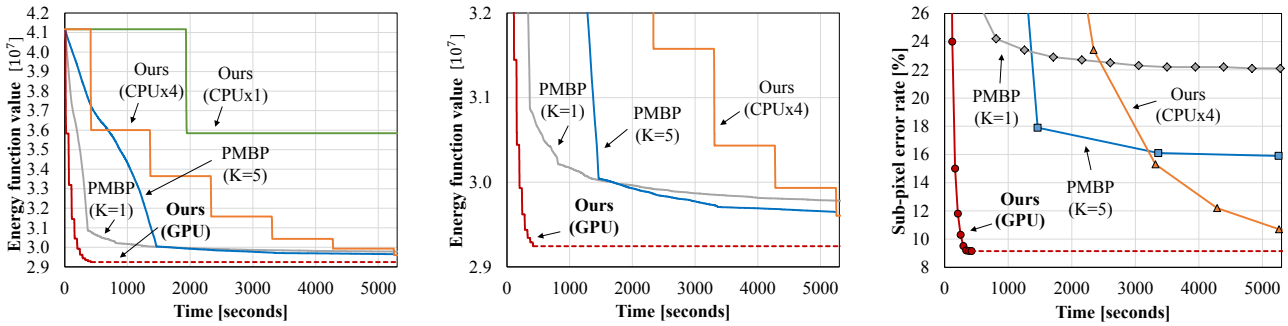
[2] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized patchmatch correspondence algorithm. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 29–43, 2010. 1, 2

[3] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. PMBP: PatchMatch Belief Propagation for Correspondence Field Estimation. In *Proc. of British Machine Vision Conf. (BMVC)*, pages 132.1–132.11, 2012. 1, 2, 5, 6, 7, 8

[4] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *Proc. of British Machine Vision Conf. (BMVC)*, pages 14.1–14.11, 2011. 1, 2, 3, 5

[5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004. 1, 2, 4, 5

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001. 1, 2, 4



(a) Time-energy transition (full) (b) Time-energy transition (zoomed) (c) Error rate

Figure 6. Efficiency and accuracy comparison with PMBP [3]. Accuracies are evaluated for all-regions after each iteration.

(a) Our method (b) PMBP [3] (c) Ground truth

Figure 7. Visual comparison with PMBP. Our method finds better disparities around edges and occluded regions without post-processing.

[7] P. Felzenszwalb and D. Huttenlocher. Efficient belief propagation for early vision. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 261–268, 2004. 1, 2

[8] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741, 1984. 1

[9] P. Heise, S. Klose, B. Jensen, and A. Knoll. PM-Huber: PatchMatch with Huber Regularization for Stereo Matching. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, pages 2360–2367, 2013. 1, 2, 5

[10] L. Hong and G. Chen. Segment-based stereo matching using graph cuts. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 74–81, 2004. 2

[11] A. Klaus, M. Sormann, and K. Karner. Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure. In *Proc. of Int'l Conf. on Pattern Recognition (ICPR)*, volume 3, pages 15–18, 2006. 2

[12] V. Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, 2006. 2

[13] V. Kolmogorov and C. Rother. Minimizing Nonsubmodular Functions with Graph Cuts – A Review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(7):1274–1279, 2007. 2, 4

[14] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, volume 2, pages 508–515, 2001. 1, 2, 7

[15] V. Kolmogorov and R. Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 82–96, 2002. 1, 2, 7

[16] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):147–159, 2004. 1, 2, 4

[17] V. Lempitsky, C. Rother, S. Roth, and A. Blake. Fusion Moves for Markov Random Field Optimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1392–1405, 2010. 1, 2, 3, 4, 7

[18] J. Lu, H. Yang, D. Min, and M. N. Do. Patch Match Filter: Efficient Edge-Aware Filtering Meets Randomized Search for Fast Correspondence Field Estimation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1854–1861, 2013. 1, 2, 5

[19] C. Olsson, J. Ulen, and Y. Boykov. In Defense of 3D-Label Stereo. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1730–1737, 2013. 1, 2, 3, 4, 7

[20] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):1068–1080, 2008. 1, 2

[21] H. Tao, H. Sawhney, and R. Kumar. A global matching framework for stereo computation. In *Proc. of Int'l Conf. on Computer Vision (ICCV)*, volume 1, pages 532–539, 2001. 2

[22] L. Wang and R. Yang. Global stereo matching leveraged by sparse ground control points. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3033–3040, 2011. 1, 2

[23] Z.-F. Wang and Z.-G. Zheng. A region based stereo matching algorithm using cooperative optimization. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008. 2

[24] Y. Wei and L. Quan. Asymmetrical Occlusion Handling Using Graph Cut for Multi-View Stereo. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 902–909, 2005. 7

[25] O. Woodford, P. Torr, I. Reid, and A. Fitzgibbon. Global Stereo Reconstruction under Second-Order Smoothness Priors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(12):2115–2128, 2009. 1, 2, 4, 7

[26] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized Belief Propagation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 13, pages 689–695, 2000. 1, 2

[27] K.-J. Yoon and I.-S. Kweon. Locally adaptive support-weight approach for visual correspondence search. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 924–931, 2005. 3