# PatchMatch Filter: Efficient Edge-Aware Filtering Meets Randomized Search for Fast Correspondence Field Estimation*

Jiangbo Lu[1], Hongsheng Yang[1], Dongbo Min[1], and Minh N. Do[2]
[1]Advanced Digital Sciences Center, Singapore [2]Univ. of Illinois at Urbana-Champaign, USA

## Abstract

*Though many tasks in computer vision can be formulated elegantly as pixel-labeling problems, a typical challenge discouraging such a discrete formulation is often due to computational efficiency. Recent studies on fast cost volume filtering based on efficient edge-aware filters have provided a fast alternative to solve discrete labeling problems, with the complexity independent of the support window size. However, these methods still have to step through the entire cost volume exhaustively, which makes the solution speed scale linearly with the label space size. When the label space is huge, which is often the case for (subpixel-accurate) stereo and optical flow estimation, their computational complexity becomes quickly unacceptable. Developed to search approximate nearest neighbors rapidly, the PatchMatch method can significantly reduce the complexity dependency on the search space size. But, its pixel-wise randomized search and fragmented data access within the 3D cost volume seriously hinder the application of efficient cost slice filtering. This paper presents a generic and fast computational framework for general multi-labeling problems called PatchMatch Filter (PMF). For the very first time, we explore effective and efficient strategies to weave together these two fundamental techniques developed in isolation, i.e., PatchMatch-based randomized search and efficient edge-aware image filtering. By decompositing an image into compact superpixels, we also propose superpixel-based novel search strategies that generalize and improve the original PatchMatch method. Focusing on dense correspondence field estimation in this paper, we demonstrate PMF's applications in stereo and optical flow. Our PMF methods achieve state-of-the-art correspondence accuracy but run much faster than other competing methods, often giving over 10-times speedup for large label space cases.*

## 1. Introduction

Many computer vision tasks such as stereo, optical flow and dense image alignment [13] can be formulated elegantly as pixel-labeling problems. In general, the common goal is to find a labeling solution that is both spatially smooth and discontinuity-preserving, while matching the observed data/label cost at the same time. To achieve this goal, a Markov Random Field (MRF)-based energy function is often employed which involves a data term and a pairwise smoothness term [18]. However, a serious challenge posed to this discrete optimization framework is computational complexity, as global energy minimization algorithms such as graph cut or belief propagation become very slow when the image resolution is high or the label space is large. Recently, edge-aware filtering (EAF) of the cost volume [17, 14] has emerged as a competitive and fast alternative to energy-based global approaches. Though simple, this kind of cost volume filtering techniques can achieve high-quality labeling results efficiently. However, despite their runtime being independent of the filter kernel size, EAF-based methods do not scale well to large label spaces.

Almost concurrently, computing approximate nearest-neighbor field (ANNF) has been advanced remarkably by the recent PatchMatch method [4] and methods improving it [5, 11, 9]. The goal of ANNF computation is to find for each image patch $P$ centered at pixel $p$ one or $k$ closest neighbors in appearance from another image. In the energy minimization context, ANNF's sole objective is to search for one or $k$ patches that minimize the dissimilarity or the data term with a given query patch, but the spatial smoothness constraint is not enforced at all. This fact is consistent with ANNF's desire of *mapping incoherence* [11] that is crucial for image reconstruction quality. The complexity of ANNF methods is only marginally affected by the label space size i.e., the number of correspondence candidates, which is vital for interactive image editing tasks [4].

Then a motivating question that follows is – whether these two independently developed fast algorithms, i.e., PatchMatch-based randomized search and EAF, can be seamlessly woven together to address the curse of large label spaces very efficiently, while still maintaining or even improving the solution quality. For the very first time, this paper is positioned to solve this interesting yet challenging problem of general applicability to many vision tasks.

*This study is supported by the HSSP research grant at the ADSC from Singapores Agency for Science, Technology and Research (A*STAR).

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.242

However, this goal is nontrivial. First, these two algorithms have different objective functions to optimize for. As shown in Fig. 1(c, d), ANNF estimated by PatchMatch [4] is very "noisy" and dramatically inferior to the desired true flow map. Second, their computation and memory access patterns are significantly disparate. In fact, the random and fragmented data access strategy within the cost volume effected by PatchMatch is drastically opposed to the highly regular and deterministic computing style of EAF methods.

Our main contribution is to propose a generic and fast computational framework for general multi-labeling problems called PatchMatch Filter (PMF). We take compact superpixels and subimages parsimoniously containing them as the atomic data units, and perform random search, label propagation and efficient cost aggregation collaboratively for them. This enables the proposed PMF framework to benefit from the complementary advantages of PatchMatch and EAF while keeping the overhead at a minimum. PMF's run-time complexity is independent of the aggregation kernel size and only proportional to the logarithm of the search range [4]. We further propose superpixel-based efficient search strategies that generalize and improve the original PatchMatch method [4]. Though not limited to the correspondence field estimation, PMF's applications in stereo matching and optical flow estimation are instantiated and evaluated in this paper. The label space considered is often huge due to e.g., two-dimensional motion search space, displacement in subpixel accuracy, or over-parameterized surface or motion modeling [7]. Experiments show that our PMF methods achieve state-of-the-art correspondence accuracy also with a superior advantage of over an order of magnitude speedup over the other competing methods.

## 2. Related Work

Here we review the work most related to our method.

**Cost-volume filtering and EAF.** Though the MRF-based energy minimization formulation for discrete labeling problems is elegant [18], the energy minimization process is still time-consuming even with modern global optimization algorithms. Leveraging the significant recent advance in edge-aware image filtering techniques, e.g. [19, 16, 10], several methods have been proposed for fast cost-volume filtering [17, 14]. They often achieve labeling results as good as those obtained by global energy-based approaches but at much faster speed, with the complexity typically independent of the filter kernel size. However, filtering each cost slice individually, albeit allowing straightforward application of various efficient EAF techniques, makes the runtime scale linearly with the label space size. This makes discrete approaches very slow in the case of large label spaces.

**ANNF computation and PatchMatch.** As explained before, computing ANNF for every patch in a given image with another image is computationally challenging, due to the large search space. Recent years have witnessed significant progress in accelerating this computation, which is key to non-parametric patch sampling used in many vision and graphics tasks. Motivated by the coherent natural structure in images, the PatchMatch method [4, 5] devised a very efficient randomized search and nearest-neighbor propagation approach, achieving substantial improvements in speed and memory efficiency over the prior arts. Inspired by PatchMatch, a few faster algorithms [11, 9] have been proposed which in one way or another allow efficient propagation from patches similar in appearance. However, with its objective to find the nearest neighbors, the computed ANNF is very different from the true visual correspondence field which is spatially smooth and discontinuity-preserving.

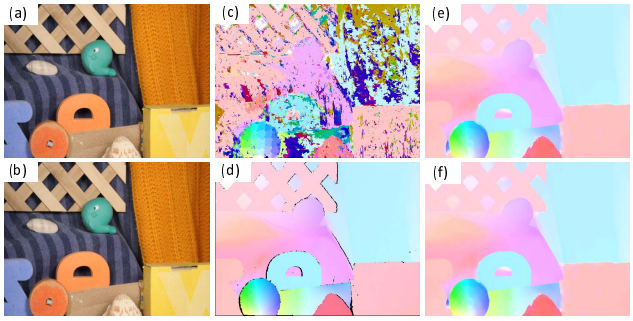


Figure 1. Problems with PatchMatch [4] and CostFilter [17] for correspondence field estimation. (a,b) Input images. (c) ANNF of PatchMatch (with the same color coding for optical flow). (d) Ground-truth flow [1]. (e) Flow map of CostFilter [17]. (f) Flow map of our PMF method, running 10-times faster than [17] under fair settings. Average endpoint error of (e) 0.0837 and (f) 0.0825.

**PatchMatch-based stereo.** Realizing PatchMatch's power in efficient search, Bleyer *et al.* [7] proposed to over-parameterize disparity by estimating an individual 3D plane at each pixel. They showed that this method can deal with slanted surfaces much better than previous methods and achieved leading subpixel disparity accuracy. This idea has also been integrated into a global optimization framework to accelerate the message passing speed [6]. To handle disparity discontinuities, adaptive-weight cost aggregation [21] in $35 \times 35$ windows is used in [7]. Though PatchMatch can significantly reduce the complexity dependency on the label space size, such a brute-force adaptive-weight summation has a linear complexity dependent on the window size and it slows down the overall runtime noticeably. In addition, more general and challenging dense correspondence problems such as optical flow are not addressed in these methods [7, 6]. It is worth noting that the histogram-based disparity prefiltering scheme [15] was proposed to reduce the complexity caused by large label spaces down to processing only e.g. $10\%$ plausible disparities detected for each pixel. But this reduction is not as aggressive as in PatchMatch, and also efficient local cost aggregation was not supported.

## 3. Problem Formulation and Challenges

We briefly present a general framework and notations of cost volume filtering-based methods for discrete labeling problems, and focus particularly on visual correspondence field estimation here. As in [17], given a pair of images $I$ and $I'$, the goal is to assign each pixel $p = (x_p, y_p)$ a label $l$ from the label set $\mathcal{L} = \{0, 1, ..., L-1\}$. $L$ denotes the label space size. For general pixel-labeling problems, the label $l$ to be assigned can represent different local quantity [18]. For the stereo and optical flow problems considered here, $l = (u, v)$, where $u$ and $v$ correspond to the displacement in $x$ and $y$ directions. Stereo matching degenerates to assigning a disparity $d(u = d)$ to pixel $p$, where $v = 0$.

Unlike global optimization-based discrete methods [18], local window-based methods stress reliable cost aggregation from the neighborhood and evaluate exhaustively every single hypothetical label $l \in \mathcal{L}$. The final label $l_p$ for each pixel $p$ is decided with a Winner-Takes-All (WTA) scheme. To achieve spatially smooth yet discontinuity-preserving labeling results, edge-aware smoothing filters have been adopted in the local cost aggregation step of several leading local methods [17, 14]. Given the raw cost slice $C(l)$ computed for a label $l$, we denote its edge-aware filtered output as $\tilde{C}(l)$. Then the filtered cost value at pixel $p$ is given as:

$$\tilde{C}_p(l) = \sum_{q \in W_p(r)} \omega_{q,p}(I) C_q(l) \ . \tag{1}$$

$W_p(r)$ is the local aggregation window centered at pixel $p$ with a filter kernel radius $r$. $\omega_{q,p}(I)$ is the normalized adaptive weight of a support pixel $q$, which is defined based on the structures of the input image $I$. Various EAF techniques [19, 16, 10, 14] can be applied here, and they differ primarily in their ways of defining and evaluating $\omega_{q,p}(I)$.

Though EAF is very efficient, the linear complexity dependency on the label space size $L$ requires repeated filtering of $C(l)$ as in (1), and $C(l)$ is of the same size of $I$. This makes the runtime unacceptably slow when $L$ is large. To largely remove this complexity dependency, recent techniques such as PatchMatch [4] appear helpful conceptually. However, it can be discerned that PatchMatch's randomized label space visit pattern for each individual pixel $p$ is very incompatible with the regular image-wise cost filtering routine that is essential to the efficiency of EAF-based methods.

## 4. PatchMatch Filter Based on Superpixels

This section proposes a superpixel-based computational framework for fast correspondence field estimation by exploiting PatchMatch-like random search and EAF-based cost aggregation synergistically. Our key motivation draws from the observation that labeling solutions for natural images are often spatially smooth with discontinuities aligned with image edges, in contrast to the very "noisy" ANNF (see Fig. 1). The very nature of spatially coherent ground-truth labeling solutions actually advocates a collaborative label search and propagation strategy for similar pixels covered in the same compact superpixel, without necessarily going to the pixel-wise fine granularity in PatchMatch [4].

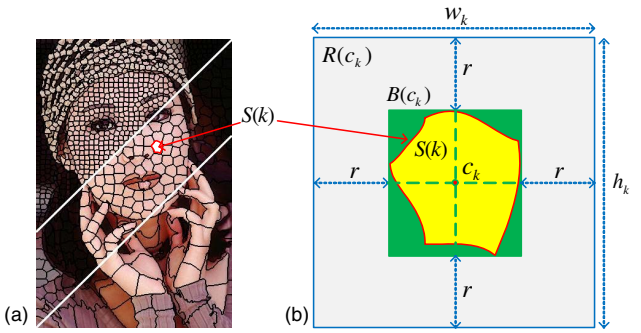


Figure 2. (a) SLIC superpixels of approximate size 64, 256 and 1024 pixels. Figure courtesy from [3]. (b) Bounding-box $B(c_k)$ containing the superpixel $S(k)$ centered at pixel $c_k$ and $r$-pixel extended subimage $R(c_k)$.

Another key motivation from a computing perspective is that the efficiency of EAF essentially comes from the high computational redundancy or the vast opportunity for shared computation reuse among neighboring pixels when filtering an image or cost slice. However, PatchMatch processes each pixel with its random set of label candidates individually in raster scan order. This renders EAF techniques not applicable and the cost aggregation runtime to grow linearly with the filter kernel size $m = (2r + 1)^2$ [7].

Based on the above analysis, we propose to partition the input image into non-overlapping superpixels, and use them as the basic units for performing random search, propagation and subimage-based efficient cost aggregation collaboratively. As a spatially regularized labeling solution is favored, such a superpixel-based strategy, adapting to the underlying image structures, is more consistent with the goal of correspondence field estimation than its pixel-based counterpart. Compared to the propagation from the immediate causal pixels [4], taking superpixels as the basic primitive also effectively extends the propagation range and ameliorates the issue of being trapped in local optimum. More importantly, superpixel-based collaborative processing creates desired chances for computation reuse and speedup.

### 4.1. Superpixel-Based Image Representation

As a key building block to many computer vision algorithms, superpixel decomposition of a given image has been actively studied. In this paper, we choose the recently proposed SLIC superpixel algorithm [3] to decompose an input color image $I$ into $K$ non-overlapping superpixels or segments, i.e., $S = \{S(k) | \bigcup_{k=1}^{K} S(k) = I \text{ and } \forall k \neq l, S(k) \cap S(l) = \emptyset\}$. Compared to other graph-based superpixel algorithms e.g. [8], the SLIC method yields state-of-the-art adherence to image boundaries, while having a faster

runtime linear in the number of pixels $M$. Another important advantage is that SLIC superpixels are compact and of more regular shapes and sizes ($M/K$ on average), giving a low overhead when their bounding-boxes are sought as discussed later. Spatial compactness also assures that the pixels from the same superpixels are more likely to share similar optimal labels. Fig. 2(a) shows SLIC superpixels generated with different parameters. For the convenience of later presentation, we also define two additional variables. As shown in Fig. 2(b), for a given segment $S(k)$, $B(c_k)$ represents its minimum *bounding-box* centered at pixel $c_k$ and $B(c_k) \in I$. We then use $R(c_k)$ to denote the *subimage* that contains $B(c_k)$, but with its borders extended outwards by $r$ pixels while still being restricted to remain within $I$.

### 4.2. PatchMatch Filter Algorithm

Now we present the PatchMatch filter (PMF) – a general computational framework to efficiently address discrete labeling problems, which exploits both superpixel-based PatchMatch search and efficient edge-aware cost filtering. The PMF framework is very general and allows the integration of various ANNF and EAF techniques. We will present improved superpixel-based search strategies in Sect. 4.3.

Unlike the regular image grid that has a default neighbor system, an adjacency (or affinity) graph is first built for an input image decomposed into $K$ superpixels in a preprocessing step. We use a simple graph construction scheme here: every segment serves as a graph node, and an edge is placed between two segments if their boundaries have an overlap. Similar to PatchMatch [4], a random label is then assigned to each node. After this initialization, we process each superpixel $S(k)$ roughly in scan order. The PMF algorithm iterates two search strategies in an interleaved manner, i.e., *neighborhood propagation* and *random search*.

First, for a current segment $S(k)$, we denote its set of spatially adjacent neighbors as $\mathcal{N}(k) = \{S(i)\}$. A *candidate pixel* $t \in S(i)$ is then randomly sampled from every neighboring segment, totaling a number of $|\mathcal{N}(k)|$. As a result, a set of current best labels $\mathcal{L}_t = \{l_t\}$ assigned to the sampled pixel set $\{t\}$ can be retrieved, and they are propagated to the superpixel $S(k)$ under consideration. Given this set of propagated labels $\mathcal{L}_t$, EAF-based cost aggregation in (1) is then performed for the subimage $R(c_k)$ defined for $S(k)$, but the filtering result is used only for the pixels in $B(c_k)$. The reason is that pixels in $R(c_k) \backslash B(c_k)$ are not supplied with all possible support pixels needed for a reliable full-kernel filtering, and also they tend to have a lower chance of sharing similar labels with pixels in $S(k)$. We denote such a subimage-based cost filtering process over a selected set of labels with a function $\mathbf{f}$ defined as follows,

$$\mathbf{f} : \mathbf{C}\left(R(c_k), \{l \in \mathcal{L}_t\}\right) \mapsto \tilde{\mathbf{C}}\left(B(c_k), \{l \in \mathcal{L}_t\}\right) \ , \quad (2)$$

where $\mathbf{C}$ and $\tilde{\mathbf{C}}$ represent the raw and filtered cost volume of cross-section size of $|R(c_k)|$ and $|B(c_k)|$, respectively. For

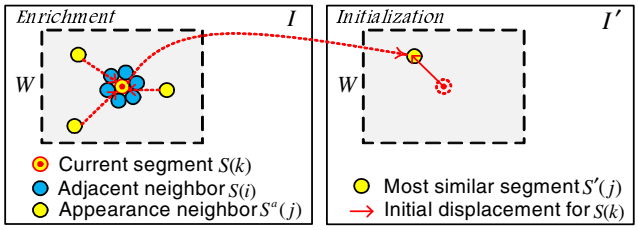


Figure 3. Generalized affinity graph and improved strategies.

any pixel $p \in B(c_k)$, its current best label $l_p$ is updated instantly by a new label $l \in \mathcal{L}_t$ whenever $\tilde{\mathbf{C}}(p, l) < \tilde{\mathbf{C}}(p, l_p)$.

After the preceding propagation step, a center-biased random search as in PatchMatch [4] is performed for the current segment $S(k)$. It evaluates a sequence of random labels $\mathcal{L}_r$ sampled around the current best label $l^*$ at an exponentially decreasing distance. We set the fixed ratio $\alpha$ between two consecutive search scopes [4] to $1/2$. Different ways exist to define $l^*$. Here we randomly pick a *reference pixel* $s \in S(k)$ to promote the label propagation within a segment. We set $l^* = l_s$, where $l_s$ is the current best label for $s$. The function $\mathbf{f}$ is then applied again to filter those cost subimages specified by $\mathcal{L}_r$ by substituting for $\mathcal{L}_t$ in (2).

To remove unnecessary computation, a list recording the labels that have been visited for each segment $S(k)$ is maintained. Therefore, no subimage filtering will be needed if a candidate label has been visited before. It is also clear from Fig. 2(b) that compact superpixels $S(k)$ are favored in our PMF algorithm, as the filtering overhead incurred by the stretched sizes of $R(c_k)$ and $B(c_k)$ will be kept low.

Note that prior stereo or optical flow methods [22, 12] often take segments as the matching units and infer a single displacement for each segment. To achieve pixel-wise accuracy, further (continuous) optimization is still required that makes them even slower. In contrast, our PMF method works like other cost-volume filtering methods [17]. It directly estimates and decides the optimal label for each pixel independently, while leveraging their shared spatial neighbors and plausible label candidates for fast computation.

### 4.3. Superpixel-Induced Efficient Search Strategies

For the clarity sake, we presented the proposed PMF framework in Sect. 4.2 based on a baseline search and propagation strategy conceptually close to the original PatchMatch principle [4]. We further propose some improved search strategies induced by the superpixel-based image representation (see Fig. 3). Compared to the baseline PatchMatch method [4], the new strategies are more effective and efficient in finding and propagating plausible candidates.

**Enrichment.** First, we generalize the adjacency graph in Sect. 4.2 to add at most $\kappa$ new *appearance neighbors* to every node or segment. Specifically, given a segment $S(k)$, we search within a predefined window the top $\kappa$ segments $\mathcal{N}^a(k) = \{S^a(j), j = 1, 2, ...\kappa\}$ most similar to $S(k)$. Due

to arbitrary shapes and uneven sizes of different segments, we use a loose form to define the inter-segment similarity:

$$H(S(k),S(j)) = \sum_{s\in S(k),t\in S(j)} \exp\left(-\frac{\|s-t\|^2}{\sigma_s^2} - \frac{\|I_s-I_t\|^2}{\sigma_r^2}\right) . \quad (3)$$

$s$ and $t$ denote pixels randomly sampled from segment $S(k)$ and $S(j)$, respectively. We repeat this random pair sampling for a fixed number of times, e.g. 10% of the average superpixel size. $\sigma_s$ and $\sigma_r$ control the spatial and color similarity. Picking the top $\kappa$ segments $\{S^a(j)\}$ closest to $S(k)$ and also above a similarity threshold, $\mathcal{N}^a(k)$ augments the original spatial neighbor set $\mathcal{N}(k)$ for $S(k)$ by non-local neighbors similar in appearance. We set $\kappa = 3$ and $\sigma_s = \infty$ here. This enrichment scheme allows effective and fast propagation of plausible label candidates from similar segments.

**Initialization.** As image representation in superpixels greatly reduces the graph complexity, this motivates us to design a better label initialization strategy than the random initialization [4]. The basic idea is to assign a potentially good candidate label rather than a random label to each segment $S(k)$. Given the maximum label search range $W$, we select for segment $S(k)$ in image $I$ a closest segment $S'(j)$ from the target image $I'$ within a slightly enlarged range. The similarity between segments is evaluated as in (3), but with $\sigma_s$ decreased to 100 to favor spatially close segments. The displacement vector between the centroids of $S(k)$ and $S'(j)$ is used as the initial label for $S(k)$. Such a preprocessing method of low complexity makes PMF converge faster and tackles small objects with large displacements better.

### 4.4. Complexity

Given an image of size $M$, the label space size $L$ and the superpixel number $K$, we further denote the total area size of subimages by $\tilde{R} = \sum_{k=1}^{K} |R(c_k)|$. Enabling the integration of linear-time EAF techniques for cost filtering, our PMF approach removes the complexity dependency on the matching window size $m$, in contrast to the PatchMatch methods [4, 7]. Consequently the complexity of our PMF is $O(K^2 + \tilde{R}\log L)$, with $O(K^2)$ accounting for the complexity upper bound of the new initialization strategy in Sect. 4.3. This overhead is negligible, also because searching for similar segments can be well constrained in a predefined search window. The dominant part of PMF is then $O(\tilde{R}\log L) \approx O(M\log L)$, as $\tilde{R}$ is larger than $M$ by a factor of a small leading constant. Table 1 gives the comparison.

The memory complexity of the PMF method is $O(M + K\log L)$. $O(M)$ is used to hold the filtered cost associated with the current best label at each pixel. Much less than $O(M)$, $O(K\log L)$ records the list of the labels that have been visited for each segment $S(k)$. In our implementation, we pre-organize all the subimages $\{R(c_k)\}$ of the input image $I$ into an array of compact 2D buffers, which facilitates cost computation and filtering in the label search process.

| | CostFilter [17] | PatchMatch [7] | PMF |
|---|---|---|---|
| Complexity | $O(ML)$ | $O(mM\log L)$ | $O(M\log L)$ |
| Memory | $O(M)$ | $O(M)$ | $O(M)$ |

Table 1. Complexity comparison of three different techniques.

## 5. Applications

We present two applications of the proposed PMF framework: stereo matching and optical flow estimation. As for the EAF techniques, we use the guided filter (GF) [10] and the zero-order cross-based local multipoint filter (CLMF-0) [14] in this paper, though other methods can be easily employed in our framework as well. Both techniques have a linear time complexity to compute (1), depending only on the image size $M$ but not on the filter kernel size $m$.

### 5.1. Stereo Matching

We present two different PMF-based stereo methods that model the scene disparity and parameterize the corresponding label space differently. Like most stereo methods [17, 14], the first approach makes an assumption of fronto-parallel local support windows, whereby pixels inside are matched to pixels in another view at a constant (integer) disparity. We call this method **PMF-C**. Similar to [7], the second approach attempts to estimate a 3D plane $Q_p$ at each pixel $p$, so pixels lying on the same slanted surfaces can then be used for reliable cost aggregation with high subpixel precision. This method is called **PMF-S**. Both methods can benefit from the PMF technique, as the disparity search range can be quite large due to high-resolution stereo images or an infinite number of possible 3D planes. Since PMF-S solves a more generalized and challenging labeling problem than PMF-C, we focus on presenting PMF-S.

**Slanted surface modeling.** For each pixel $p$, we search for a 3D plane $Q_p$ defined by a three-parameter vector $\mathbf{l}_p = (a_p, b_p, c_p)$. Given such a plane, a support pixel $q = (x_q, y_q)$ in $p$'s neighborhood $W_p(r)$ in the left view $I$ will be projected to $q' = (x_{q'}, y_{q'})$ in the right view $I'$ as:

$$x_{q'} = x_q - d_q = x_q - \mathbf{l}_p \cdot (x_q, y_q, 1)^\top , \text{ and } y_{q'} = y_q . \quad (4)$$

Disparity $d_q$ in (4) is computed from the plane equation whose value exists in a continuous domain. This enables PMF-S to handle slanted scene objects much better than PMF-C by avoiding discretization of disparities.

**Raw matching cost.** For both PMF-C and PMF-S, we compute the raw matching cost between a pair of hypothetical matching pixels $q$ and $q'$ in the similar way as [17, 20]:

$$\begin{aligned} C_q(l) = (1-\beta)\cdot \min\left(\left\|I_q - I'_{q'}\right\|, \gamma_1\right) \\ + \beta \cdot \min\left(\left\|\nabla I_q - \nabla I'_{q'}\right\|, \gamma_2\right) . \end{aligned} \quad (5)$$

For PMF-C, the label $l$ represents a disparity candidate $d$, while $l$ corresponds to the three parameters $(a_p, b_p, c_p)$ of a plane evaluated for the center pixel $p$ in PMF-S. For stereo, $\nabla$ evaluates only the gradient in $x$ direction in (5). The

color and gradient dissimilarity is combined using a user-specified parameter $\beta$. $\gamma_1$ and $\gamma_2$ are truncation thresholds. Since $q'$ generally takes fractional $x$-coordinates in PMF-S, linear interpolation is used to derive its color and gradient.

**PMF-based cost aggregation.** We apply the PMF algorithm described in Sect. 4.2 to perform superpixel-based collaborative random search, propagation and cost subimage filtering. The implementation of cost aggregation for PMF-C is straightforward, whereas more care needs to be taken for the random plane initialization and iterative random search steps in PMF-S[1]. To this end, we adopt the approach presented in [7], and use a random unit normal vector $(n_x, n_y, n_z)$ plus a random disparity value sampled from the allowed continuous range as proxy for the plane representation. View propagation [7] is also used in PMF-S to propagate the plane parameters of the matching pixels.

**Post-processing.** After deciding an initial disparity map using a WTA strategy, we detect unreliable disparity estimates by conducting a left-right cross-checking. Then, they are filled by background disparity extension [17] in PMF-C, and plane extrapolation [7] in PMF-S. Finally, a weighted median filter is applied to refine the resulting disparity map.

### 5.2. Optical Flow

We now present a PMF-based optical flow method named **PMF-OF**. Its main work flow closely resembles that of PMF-C, but a label $l$ represents a displacement vector $(u, v)$ in $x$ and $y$ directions. The label space for optical flow is therefore often much larger than typical label spaces tackled in stereo matching. Based on a discrete labeling formulation, PMF-OF solves for subpixel accurate flow vectors by upscaling the label dimension to allow fractional displacements along both $x$ and $y$ directions. As in [17], an upscaling factor of 8 is used in this paper, and the pixel colors at subpixel locations are obtained from bicubic interpolation.

Given a candidate label $l$, a pixel $q$ in image $I$ is matched to the pixel $q' = q + (u, v)$ in the second image $I'$. We use (5) to measure the raw matching cost $C_q(l)$, but computing the gradients in both $x$ and $y$ directions. The PMF-based label search and cost filtering algorithm is then applied, including those improved strategies presented in Sect. 4.3 to more effectively tackle the huge motion search space. Afterwards, the WTA decision, occlusion handling and post-refinement as in [17] are performed to give the final flow map.

## 6. Experimental Results

We implemented the PMF algorithm in C++, and also GF [10] and CLMF-0 [14] used for EAF in (1). The following same parameter settings are used across all stereo and optical flow datasets: $\{r, \sigma_r, \beta, \gamma_1\} = \{9, 0.1, 0.9, 0.039\}$. As in [17], $\gamma_2 = 0.008$ $(0.016)$ is used for stereo (optical flow). We set the smoothness parameter $\epsilon = 0.01^2$ in GF,

[1] Our improved strategies are not used to allow fair comparison with [7].

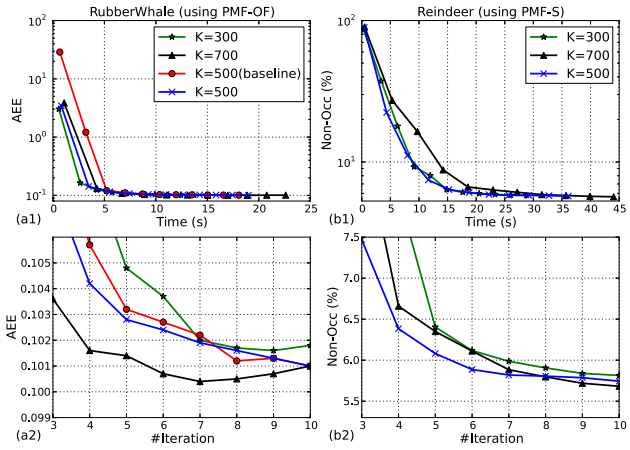


Figure 4. Time-accuracy trade-off study of PMF approaches.

and the inlier threshold $\tau = 0.1$ in CLMF-0. The segment number $K$ is set to 500. All algorithms were run on an Intel Core i5 2.5GHz CPU with a single-core implementation.

### 6.1. Time-Accuracy Trade-off Evaluation of PMF

First, we present a quick time-accuracy trade-off study of our PMF approaches in Fig. 4. Two test image pairs *RubberWhale* and *Reindeer* from the Middlebury optical flow/stereo datasets [2, 1] are used to evaluate the PMF-OF and PMF-S methods (using CLMF-0), respectively. It can be observed that for a reasonable range of $K$ settings, optical flow or stereo results have almost always converged after 8-10 iterations. This also holds true for other images tested with GF, though not shown here. Fig. 4(a1) shows that our improved search strategies in Sect. 4.3 lead to a faster convergence speed than the baseline method, especially for the first few iterations. For the same iteration number, choosing a larger $K$ (namely a smaller superpixel size) gives a better gain in accuracy on optical flow estimation than stereo, due to intrinsically more complex 2D motions. However, this is at a price of a longer runtime per iteration caused by the increased adjacency graph size and increased subimage processing overhead. In general, we find that $K = 500$ gives a good balance between the complexity of each iteration and the iteration number for a target accuracy level.

### 6.2. Stereo Matching Results

We evaluate our PMF-C and PMF-S stereo methods combined with CLMF-0 and GF filtering techniques, using the Middlebury stereo benchmark [2] in Table 2. When the default error threshold 1.0 is used, all our PMF-S and PMF-C methods are highly ranked out of over 135 stereo algorithms. They also achieve disparity accuracy comparable to or even better than the top-performing local stereo methods – PatchMatch stereo [7] and CostFilter [17]. We also evaluated those algorithms designed specifically to tackle slanted surfaces with subpixel precision, setting the Middlebury error threshold to 0.5. The upper part of Table 2

| Algorithm | Err. thre. = 1.0 | | Err. thre. = 0.5 | |
|---|---|---|---|---|
| | Rank | Err. % | Rank | Err. % |
| **PMF-S (w/ CLMF-0)** | **15** | **4.04** | 6 | 8.67 |
| **PMF-S (w/ GF)** | 16 | 4.06 | **2** | **7.69** |
| PatchMatch [7] | 18 | 4.59 | 8 | 9.91 |
| PMBP [6] | 21 | 4.46 | 4 | 8.77 |
| **PMF-C (w/ CLMF-0)** | 23 | 5.26 | - | - |
| CostFilter (w/ GF) [17] | 24 | 5.55 | - | - |
| **PMF-C (w/ GF)** | 25 | 5.48 | - | - |

Table 2. Middlebury quantitative stereo evaluation results.

| Algorithm | *Teddy* | | | *Cones* | | |
|---|---|---|---|---|---|---|
| | nocc | all | disc | nocc | all | disc |
| **PMF-S-GF** | $4.45_2$ | $9.44_2$ | $13.7_2$ | $\mathbf{2.89_1}$ | $8.31_2$ | $\mathbf{8.22_1}$ |
| PMBP [6] | $5.60_3$ | $12.0_6$ | $15.5_3$ | $3.48_3$ | $8.88_4$ | $9.41_4$ |
| **PMF-S-CLMF0** | $\mathbf{4.07_1}$ | $10.5_3$ | $\mathbf{12.1_1}$ | $2.96_2$ | $8.84_3$ | $8.38_2$ |
| PatchMatch [7] | $5.66_4$ | $11.8_5$ | $16.5_4$ | $3.80_5$ | $10.2_6$ | $10.2_5$ |

Table 3. Stereo evaluation results when error threshold = 0.5.

shows that our PMF-S methods with GF or CLMF-0 (using the same parameter settings for the error threshold 1.0) perform better than PatchMatch stereo [7]. They are also better than or close to the state-of-the-art PMBP [6], while the latter uses belief propagation for global optimization. In particular, our PMF-S methods achieve the top performance for the more complex datasets of *Teddy* and *Cones* among all Middlebury stereo methods as shown in Table 3. Fig. 5 shows the disparity maps estimated by our PMF-S methods, which preserve depth discontinuities while generating spatially smooth disparities with high subpixel accuracy. Compared to the fronto-parallel version i.e., PMF-C, PMF-S reconstructs the slanted surfaces at much higher quality.

Our PMF-C methods have a runtime comparable with CostFilter [17] on the Middlebury dataset of small disparity ranges, but run over 3-7 times faster than [17] for high-resolution stereo images (e.g. 1M pixels to 4K movie resolution) as the disparity range increases accordingly. Without reducing the iteration number nor turning off plane refinement, our PMF-S methods achieve a few times speedup over PM stereo [7] (e.g. 20 sec vs. 1 minute). In fact, our fair comparison indicates over 10-times speedup over [7]. Using a low-order regression model, CLMF-0 brings 2-3 times overall speedup over GF for PMF-C and PMF-S algorithms.

### 6.3. Optical Flow Results

We evaluate our PMF-OF methods using the Middlebury flow benchmark [1]. In all the following tests, we have fixed the motion search range to $[-40, 40]^2 \times 8^2$ (about 410,000 lables) and the number of iterations to 10. Table 4 lists the average ranks of a few competing methods also based on discrete optimization as well as the top-performing MDP-Flow2 [20] measured in the average endpoint error (AEE)[2].

[2]Please refer to [1] for the complete results using other error measures.

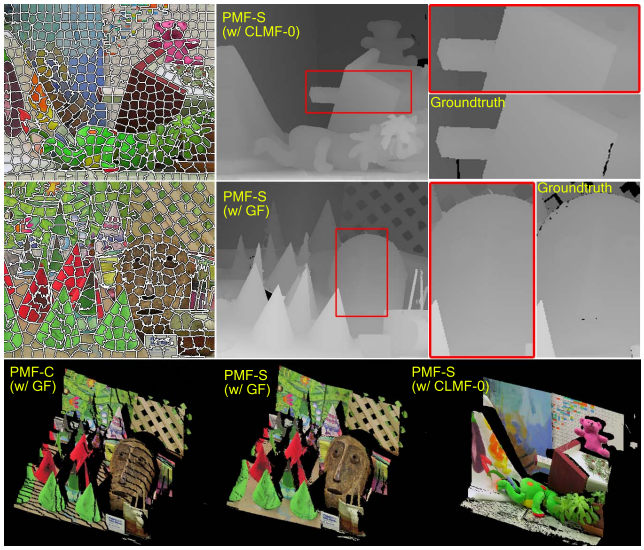


Figure 5. Visual results. Top row (left to right): Segmented *Teddy* image, PMF-S (w/ CLMF-0) result and close-up comparison. Middle row (left to right): Segmented *Cones* image, PMF-S (w/ GF) result and close-up comparison. Bottom row (left to right): Synthesized novel-view images using PMF-C and PMF-S.

Both PMF-OF (w/ GF) and PMF-OF (w/ CLMF-0), though simple and free of a large number of parameters, have a very competitive ranking out of over 70 methods. In particular, they both outperform CostFilter [17] (see also Fig. 1), even though image-wise cost filtering has been exhaustively performed for every single label in [17]. This very fact of a label space subsampling method giving better results was also observed and explained from the information representation perspective in [15]. Also, using compact superpixels as the atomic units tends to have better spatial regularization than [17], without compromising the accuracy along motion discontinuities. Table 4 shows that PMF-OF methods perform quite well for the three challenging scenes with fine details and strong motion discontinuities. PMF-OF (w/ GF) even tops the *disc* rank for *Teddy*. In Fig. 6, we compare visually the flow maps estimated by PMF-OF (w/ GF) and other competing methods. Our method preserves fine motion details and strong discontinuities, and handles nonrigid large-displacement flow without changing any parameters. Fig. 7 verifies the strength of our superpixel-induced initialization and search strategies over the baseline approach.

As reported in Table 4, our PMF methods have a significant runtime advantage and often give an order of magnitude speedup over the previous methods. Tested on the same CPU, PMF-OF runs even over 30-times faster than CostFilter [17] on the *Urban* sequence, thanks to slashing the complexity dependency on the huge label space size.

## 7. Discussion and Future Work

This paper proposed a generic PMF framework of solving discrete multi-labeling problems efficiently. We have

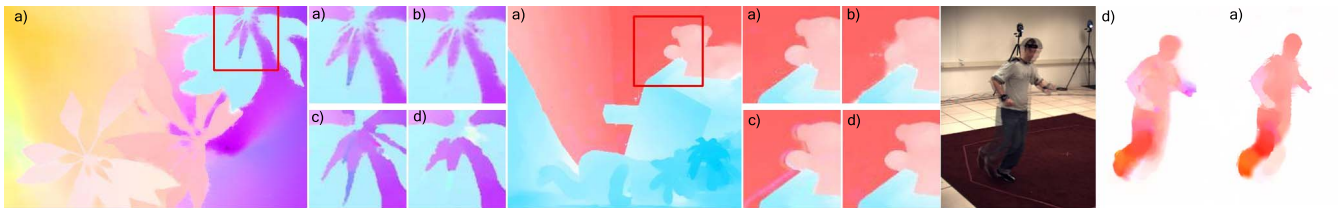


Figure 6. Visual result comparison on *Schefflera*, *Teddy* and *HumanEva* by a) PMF-GF b) CostFilter [17] c) DPOF [12] d) MDP-Flow2 [20].

| Algorithm | $\mu$Rank | *Schefflera* | *Grove* | *Teddy* | sec |
|---|---|---|---|---|---|
| MDP-Flow2 [20] | 5.0 | (2,2,**1**) | (9,10,10) | (2,2,2) | 342 |
| **PMF-GF** | 19.9 | (5,5,8) | (4,4,3) | (3,**1**,7) | 35 |
| MDP-Flow | 21.6 | (6,8,28) | (21,21,26) | (44,47,43) | 188 |
| **PMF-CLMF-0** | 22.5 | (15,17,8) | (8,8,2) | (4,2,9) | 18 |
| CostFilter [17] | 25.0 | (4,4,13) | (6,7,4) | (9,18,9) | 55* |
| DPOF [12] | 31.2 | (6,6,28) | (12,15,8) | (22,18,4) | 287 |

Table 4. Middlebury quantitative flow evaluation results measured with average endpoint error (AEE) for three challenging scenes. In brackets are the ranks for (all, disc, untext). Runtime is given for the *Urban* sequence. *Runtime measured on a powerful GPU.

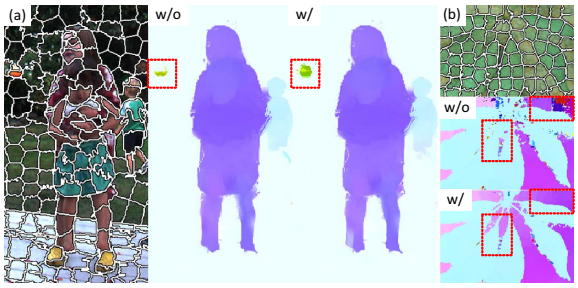


Figure 7. Advantages of our improved search strategies. a) Better initialization. b) Non-local neighbor propagation (# iteration = 3).

particularly demonstrated its effectiveness in estimating smoothly varying yet discontinuity-preserving stereo and optical flow maps. Though we focused on the randomized search paradigm here, more efficient ANNF computing methods e.g. [9] can be readily plugged into our PMF framework. Similarly, optical flow can also be over-parameterized with an affine motion model by taking advantage of the power of randomized search in the high-dimensional parameter space. Interestingly, dense patch matching [4] has been employed in state-of-the-art optical flow method [20] to find multiple extended displacements. Our method can be used to generate a good initial motion field quickly for further continuous optimization. Exploring the proposed PMF algorithm to tackle other pixel-labeling tasks efficiently is also an interesting future direction.

## References


[1] http://vision.middlebury.edu/flow/. 2, 6, 7

[2] http://vision.middlebury.edu/stereo/. 6

[3] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. PAMI*, 34(11), 2012. 3

[4] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *Proc. of ACM SIGGRAGH*, 2009. 1, 2, 3, 4, 5, 8

[5] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *Proc. of ECCV*, 2010. 1, 2

[6] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. PMBP: Patchmatch belief propagation for correspondence field estimation. In *Proc. of BMVC*, 2012. 2, 7

[7] M. Bleyer, C. Rhemann, and C. Rother. Patchmatch stereo - Stereo matching with slanted support windows. In *Proc. of BMVC*, 2011. 2, 3, 5, 6, 7

[8] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004. 3

[9] K. He and J. Sun. Computing nearest-neighbor fields via propagation-assisted KD-trees. In *CVPR*, 2012. 1, 2, 8

[10] K. He, J. Sun, and X. Tang. Guided image filtering. In *Proc. of ECCV*, 2010. 2, 3, 5, 6

[11] S. Korman and S. Avidan. Coherency sensitive hashing. In *Proc. of ICCV*, 2011. 1, 2

[12] C. Lei and Y.-H. Yang. Optical flow estimation on coarse-to-fine region-trees using discrete optimization. In *Proc. of ICCV*, 2009. 4, 8

[13] C. Liu, J. Yuen, and A. Torralba. SIFT flow: Dense correspondence across scenes and its applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5), 2011. 1

[14] J. Lu, K. Shi, D. Min, L. Lin, and M. N. Do. Cross-based local multipoint filtering. In *CVPR*, 2012. 1, 2, 3, 5, 6

[15] D. Min, J. Lu, and M. N. Do. A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy? In *Proc. of ICCV*, 2011. 2, 7

[16] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand. Bilateral filtering: Theory and applications. *Foundations and Trends in Comp. Graphics and Vision*, 4(1):1–73, 2008. 2, 3

[17] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *CVPR*, 2011. 1, 2, 3, 4, 5, 6, 7, 8

[18] R. Szeliski and et al. A comparative study of energy minimization methods for Markov Random Fields with smoothness-based priors. *IEEE TPAMI*, 2008. 1, 2, 3

[19] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. of ICCV*, 1998. 2, 3

[20] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *IEEE Trans. PAMI*, 2012. 5, 7, 8

[21] K. Yoon and I. Kweon. Adaptive support-weight approach for correspondence search. *IEEE Trans. PAMI*, 2006. 2

[22] C. L. Zitnick and S. B. Kang. Stereo for image-based rendering using image over-segmentation. *IJCV*, 2007. 4