Multimedia Tools and Applications (2020) 79:13291–13309
https://doi.org/10.1007/s11042-020-08633-y

High accuracy correspondence field estimation via MST based patch matching

Feihu Zhang[1] · Shibiao Xu[1] · Xiaopeng Zhang[1]

Received: 3 May 2019 / Revised: 20 November 2019 / Accepted: 2 January 2020 /
Published online: 28 January 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
This paper presents an effective framework for correspondence field estimation. The core idea is to construct pixel-level and superpixel-level patch matching to achieve high accuracy estimation as well as fast speed computation. To this end, a hybrid edge-preserving supported weighting approach is first developed, which contributes to better performance on the pixel level, especially on those in the regions of fine structures. Then, a local Minimum Spanning Tree (MST) is constructed to describe regions and develop the adaptive smooth penalty weights, so that the over-patching in large textureless regions can be effectively avoided. In addition, the MST is further extended to handle occlusions in way of edge preserving strategy. Finally, all the above treatments are collected into an optimization model where the objective function is developed in terms of Markov Random Filed (MRF). In computation, a fast yet efficient iterative optimization strategy is developed. Our approach achieves favorable place on optical flow benchmark, which locates at the top two and top four for endpoint error and angular error evaluations among more than 130 approaches listed in the webpage.

Keywords Optical flow · Minimum spanning tree · PatchMatch

1 Introduction

Most correspondence field estimation tasks (e.g., optical flow) suppose that all pixels within the support window have constant label values. In practice, constant label values are unlikely to hold. Thus, continuous correspondence field estimation algorithms mainly focus on how to effectively solve the problem of finding a "good" slanted support plane at each pixel. On

Feihu Zhang and Shibiao Xu contributed equally to this work and share the first authorship.

✉ Shibiao Xu
shibiao.xu@ia.ac.cn

✉ Xiaopeng Zhang
xiaopeng.zhang@ia.ac.cn

1 National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

the one hand, Bleyer et al. [8] proposed PatchMatch algorithm to over-parameterize correspondence field by estimating an individual three-dimensional plane at each pixel, which can deal with slanted surfaces better than previous methods with subpixel accuracy. On the other hand, PatchMatch, as a computationally efficient way to compute the nearest neighbor field (NNF) between two images, has generated a lot of interests in optical flow, due to the ability of matching patches across large distance. These typical PatchMatch based optical flow methods include PatchMatch Fliter [18], Edge-Preserving PatchMatch [4] and over-PMBP [14]. Based on the above progress, we concern and review the existing challenges in this section, including supported weights computation, large textureless regions estimation and occlusion handling.

1.1 Supported weights computation

Correspondence field estimation tasks have experienced a renaissance, since Yoon et al. [28] proposed the adaptive supported weight (ASW), which is calculated inversely proportional to the color dissimilarity with the window's center pixel and the spatial distance from the center and defined as

$$W_{ASW}(p,q) = \exp\left(-\left(\frac{\Delta c_{pq}}{\gamma_c} + \frac{\Delta g_{pq}}{\gamma_g}\right)\right). \tag{1}$$

Where Δc_{pq} and Δg_{pq} represent the color difference and the spatial distance between pixel p and q, respectively. Variables γ_c and γ_g are two user-defined parameters.

After that, ASW has been directly introduced into many PatchMatch based continuous correspondence field estimation algorithm [7] without considering to utilize the spatial information. However, there exist some problems: 1) Large weights would be assigned to the pixels which have different label values but the similar color to the center pixel in the supported window, thus, false label values are inclined to be assigned and the edges of the label planes are flattened easily. 2) The label value of background is inclined to be assigned to the foreground objects which are negligible (small or fine enough) compared with the window size, and we called this background erosion. These two problems are illustrated in Fig. 2.

There are many other methods which computed an explicit color over-segmentation in a preprocessing step. Tombari et al. [25] suggested assigning a weight of one for all pixels that lie in the same segment with the window's center pixel p. For all pixels outside of p's segment, ASW in (1) [28] is used to compute the weight. Formally, the weighting function is defined as

$$W_{Seg}(p,q) = \begin{cases} 1.0 & if\ \ q \in S_p \\ W_{ASW}(p,q) & otherwise. \end{cases} \tag{2}$$

Here, S_p represents the pixel set inside p's segment. However, this strategy directly relies on the reliability of the segmentation methods.

In contrast to the above local supported weight computation, we use global minimum spanning tree (MST) to design a novel hybrid supported weight. That is because MST could give a more natural image pixel similarity measurement metric, according to the recent tree filtering techniques [3]. Another superiority of using MST is that it can be constructed efficiently in linear time complexity.

1.2 Large textureless regions estimation

Despite intensive efforts in the optical flow research, they still remain challenges to estimate the correspondence information reliably, especially in large textureless regions. In window based matching methods [8, 15], the easiest and obvious way to handle these smooth regions is to increase the size of the supported window, which usually leads to serious problems, such as the high computational complexity, structure destruction and edge flattening.

Another way is to increase the smooth penalty weight in MRF framework when the intensity variation contributes rarely to choose the best motion values in these textureless regions. However, such a heavy smooth penalty for all the pixels usually leads to over-smooth results. The best choice is to utilize different smooth weights in different regions properly.

Recently, SLIC superpixel algorithm [1] has been employed in optical flow methods [18], which produced impressive performance. In this paper, we also introduce the superpixel segment strategy and utilize it to define our region (superpixel) based smooth factor which helps us to apply adaptive smooth weights in different regions.

1.3 Occlusion handling

For two-frame optical flow, the invalid regions needed to be recovered including not only the occlusion areas, but also the borders of the view and the falsely labeled regions. The structure of the label map will be destroyed easily because of the complexity of these areas. Thus, how to recover the invalid regions while preserving the structure is an extremely challenging task.

In order to deal with the dilemma, various post refine measures, including our structure-preserving occlusion handling strategy, are applied as the last step of the correspondence field estimation methods to handle the occlusion problem. Due to the fact that objects with a smaller flow magnitude can occlude objects with higher flow magnitude, weighted median filter [4] is widely used for optical flow to fill the occluded regions based on their color similarity to the visible flow regions. And recently, various local or nonlocal image filtering techniques [27] are adopted to propagate labels into invalid regions, however these methods propagate both the background and the foreground label information to the occlusion regions, while the occlusion usually appears in the background. Thus, it easily destroys the geometric structure of the disparity or motion map.

1.4 Contributions

In this paper, we mainly focus on the above challenges, and aim to develop a more reliable method for continuous correspondence field estimation task which takes optical flow for example. Our contributions can be highlighted as follows: 1) It can achieve better performance in fine structure and object edges with our newly designed hybrid supported weight, which separates the supported window into two parts based on the global MST, and for each part, we used the MST based weight and the adaptive supported weight with upper bound to calculate the supported weights relatively. 2) It is suitable for large textureless regions estimation to use our local MST based smooth factor, which gives different regions proper smooth penalty weights for the MRF framework. 3) It is computational efficient to accelerate the label propagation and convergence of the energy function during the coarse to fine optimization process. 4) It could precisely recover the complicated occlusions or invalid regions with our high-efficiency edge-aware occlusion handling strategy, which works as the post-refinement procedure over the whole continuous correspondence field estimation task.

2 Proposed formulation

In this section, we describe the proposed method by introducing a pairwise MRF formulation [21]. In the MRF framework, the goal is to find a label vector field f_p for each pixel $p \in I$ by minimizing

$$E(f) = \sum_{p \in I} \phi_p \left(f_p\right) + \lambda \sum_{p \in I} \sum_{q \in N(p)} \psi \left(f_p, f_q\right). \tag{3}$$

Where data term $\sum_{p \in I} \phi_p \left(f_p\right)$ measures the photo-consistency between matching pixels. And f_p defines a warp from a pixel p in the reference view to its correspondence in the other view. Smooth term $\sum_{p \in I} \sum_{q \in N(p)} \psi \left(f_p, f_q\right)$ penalizes discontinuity of label values between a pixel p and its neighboring pixels $q \in N(p)$. And λ is the smooth penalty weight which balances the data term and the smooth term.

2.1 Data term with hybrid supported weight

To measure the photo-consistency, we use the recently developed PatchMatch method in [8] to over-parameterize the correspondence field and avoid the frontal-parallel bias. For each pixel $p = (x_0, y_0)^T \in I$, we denote the correspondence field by $f_p = (u, v)$, which represents the displacement between two frames. With a label value z_0 (z_0 represents the label value of correspondence field in u direction or v direction) and a normal $\mathbf{n} = (n_x, n_y, n_z)$, both u and v can be parameterized as 3D plane $\pi = (z_0, \mathbf{n})$. Therefore, the objective becomes to seek the two 3D planes (π_u, π_v) for each pixel p in the image pair, such that the label (disparity or motion) map f minimizes the energy function $E(f)$ of (3). By denoting p's correspondence field by f_p, a pixel $q \in N(p)$ in the reference image can be mapped into a new location q' in the other image by using $q' = q + (u, v)^T$; Here, $(u, v) = (d(\pi_u), d(\pi_v))$ can be achieved by

$$d(\pi) = \frac{-n_x x - n_y y + (n_x x_0 + n_y y_0 + n_z z_0)}{n_z}. \tag{4}$$

Where, $(x_0, y_0)^T$ is p's coordinate and (x, y) is q's coordinate. Then, the matching cost for pixel p with f_p can be defined as

$$\phi_p(f_p) = \sum_{q \in W_p} W(p, q) \rho(q, q'). \tag{5}$$

Where, W_p is a support window with p as its center. $W(p, q)$ is the supported weight of the current pixel q in the window W_p, which can be calculated by our hybrid supported weight (discussed in later). The function $\rho(q, q')$ measures the dissimilarity between q and its counterpart q' in the other view. It can be achieved by

$$\begin{aligned} \rho(q, q') = (1 - \kappa) min(\left\| I_q - I_{q'} \right\|, \tau_{col}) \\ + \kappa \min(\left\| \nabla I_q - \nabla I_{q'} \right\|, \tau_{grad}). \end{aligned} \tag{6}$$

Where $\left\| I_q - I_{q'} \right\|$ computes the L_1-distance of q and q' in RGB space. And $\left\| \nabla I_q - \nabla I_{q'} \right\|$ denotes the absolute difference of gray-value gradients computed at q and q'. Since the coordinates of q' lies in the continuous domain, we derive its color and gradient values by interpolation. The user-defined parameter κ balances the influence of color and gradient term. Parameters τ_{col} and τ_{grad} truncate costs for robustness in occlusion regions. This match measure has recently been applied in PatchMatch algorithm [8] and has the advantage of handling radiometric differences in the input images due to using the gradient.

Hybrid supported weight as one of our contribution, we describe our hybrid supported weight here. Unlike previous supported weight methods, in this paper, we introduce MST to calculate the supported weight for each window masks. Firstly, each image view is represented as a standard 8-connected, undirected graph $G = (V, E)$, with a weight function mapping edges to real-valued weights. The vertices in V are all the image pixels and the edges in E are all the edges between the nearest neighboring pixels. The weight between neighboring pixels p and q is the L_1-distance between them in RGB color space, which is defined by

$$\omega(p, q) = \omega(q, p) = \left\| I_p - I_q \right\|. \tag{7}$$

Here, edges with large weights are “unwanted”, which will be removed during spanning tree construction. These removed edges usually contain the color borders, and since the object edges are usually subset of the color borders, these edges thus will not be included in the final tree.

Instead of calculating the local MST for each of the mask windows, we just construct the global MST for the whole image, because global MST reflects the panorama of the whole view, more edges of the label discontinuities will be excluded and it’s also more efficient by reducing redundant calculation. After the construction of the global MST, a window mask is covered on each target pixel p. There is a sub-tree T_{W_p} appeared in the supported window, whose root is the center pixel p. Then the mask window can be separated into two sets, including S_1, which consists of the sub-tree, and S_2, pixels in which are not parts of the sub-tree. We then calculate their weights respectively.

For the pixels in set S_1, it is straightforward to define the similarity between two image pixels using the distance between two nodes on the MST. In other words, once the two pixels are close in the MST, they are similar, vice versa. The distance between two nodes in the MST is the sum of weights of the path between the two nodes. Let $D(p, q) = D(q, p)$ as the distance between p and q in sub-tree T_{W_p}. The supported weight can be calculated with

$$W_{S_1}(p, q) = exp(-\frac{D(p, q)}{\delta}). \tag{8}$$

Where δ is a constant used to adjust the similarity between two nodes.

For the pixels in S_2, we use the adaptive supported weight [28] with an upper bound to restrict its maximum. The weight can be achieved by

$$W_{S_2}(p, q) = \min\left(\exp\left(-\frac{\Delta c_{pq}}{\gamma_c}\right), \tau_\omega\right). \tag{9}$$

Here, upper bound τ_ω is usually a small constant. For the color difference Δc_{pq}, we use the absolute difference in RGB space.

By using such a hybrid supported weight, the different label planes but similar color and the background erosion problems can be alleviated significantly. This is because that pixels with different label values to the center pixel tend to be classified into the second set S_2 and the weights will be calculated by (9) with upper bound restriction. Even if some fallible pixels fall into the first set S_1 undesirably, the final weight calculated by (8) would also be small enough to avoid false disparity assignment, due to that $D(p, q)$ in (8) would increase greatly when the route crosses the border of the label plane. Our partition strategy is similar to binary window weight [25], while it's more flexible and reliable. Figure 1 gives a directly visual comparison of different supported weights. And Fig. 2 shows results (without occlusion handling) of these weight strategies when imbedded into our framework with the most proper parameters.

2.2 Smoothness term with variable weight

For the smoothness term, we use the curvature-based, second-order smooth regularization term proposed in [20], defined as

$$\psi_{pq}(f_p, f_q) = max(W(p,q), \varepsilon) min(\bar{\psi}(f_p, f_q), \tau_{dist}). \tag{10}$$

Here $W(p,q) = \exp\left(-\left(\frac{\Delta c_{pq}}{\gamma_c}\right)\right)$, which measures the color similarity between p and q. ε is a small constant value that gives a lowest bound to the weight for increasing the robustness. The function $\bar{\psi}_{pq}$ penalizes the discontinuity between f_p and f_q with

$$\bar{\psi}_{pq}(f_p, f_q) = \left\| d_p(f_p) - d_p(f_q) \right\| + \left\| d_q(f_q) - d_q(f_p) \right\|. \tag{11}$$

Where $d_p(f_q) = (d(\pi_u^q), d(\pi_v^q))$ can be calculated using (4). And $\bar{\psi}_{pq}(f_p, f_q)$ is truncated by τ_{dist} to allow sharp jumps at label or object edges.

Variable smooth weight as another contribution, unlike many other MRF based methods [7], which utilized the MRF framework with fixed smooth penalty weight, in this paper, our smooth weight λ in (3) is defined as variable values. For regions with rich textures, λ is assigned with a relative small value to avoid edge flattening or over-smooth results. While for the textureless regions, where the intensity variation contributes rarely to choose the best labels, the smooth penalty should be heavy enough to fill these regions with proper label values. The goal is to find such a smooth factor which measures the surrounding smoothness and affects the smooth penalty properly. Therefore, we introduce such a smooth factor based on the superpixel segmentation and the local MST construction.

We use the Simple Linear Iterative Clustering (SLIC) method [1] to find the superpixel segmentations for input images. The resulting superpixels, or regions, can adhere well to

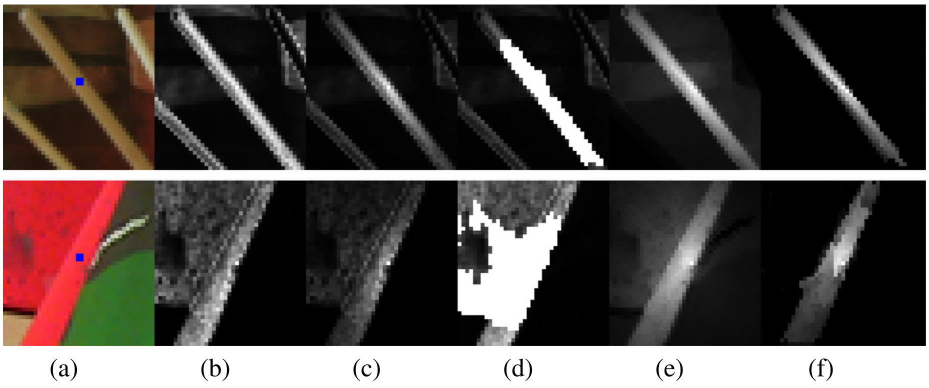


Fig. 1 Visual comparison of different supported weights. **a** Image crops for selected windows of the Middlebury images, **b** adaptive support weight with spatial term [28], **c** adaptive weight without spatial term [7, 8], **d** segmentation based weight [25], **e** geodesic supported weight [15], **f** our hybrid supported weight, which performs best among these different weight strategies

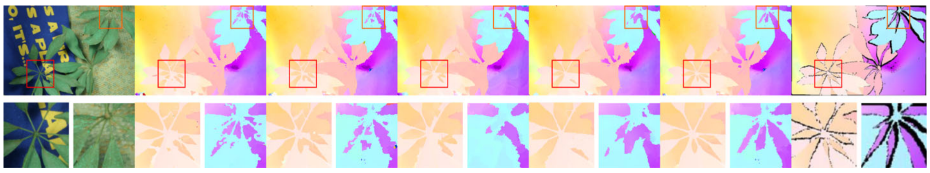


Fig. 2 Comparison of different supported weights with application of optical flow. These results are collected by applying different supported weight into our framework, and parameters are adjusted accordingly. The first row gives an example of the similar color but different disparity planes problem, the second row illustrates the background erosion problem, and the third row shows the optical flow results. **a** original images, **b** the result of adaptive supported weight [28], **c** Adaptive supported weight without spatial term, **d** geodesic supported weight [15], **e** segment based weight [13, 25], **f** our mixed supported weight, **g** ground truth

image boundaries. Additionally, SLIC provides us with the freedom to tweak the compactness of the generated superpixels, which is important, as the desired superpixel size in SLIC depends on the size of the input images.

Then we build local MST T_S for each superpixel based region S. After that, the smooth factor can be developed as

$$\alpha = 1 + \beta \exp(-\frac{\nu(T_S)}{\delta_s}). \tag{12}$$

Where β and δ_s are two user defined parameters. β constrains the variation range of the smooth factor into $(1, 1 + \beta]$. And $\nu(T_S)$ can be calculated using the edges e of the region MST T_s by

$$\nu(T_S) = \frac{1}{N} \sum_{e \in T_S} e. \tag{13}$$

Here, N is the number of the pixels in region S, and e represents the edge weight of the MST which is calculated by the (7). Actually, $\nu(T_S)$ measures the averaged edge weight of the region MST T_S. We build the local MST to measure the region smoothness instead of introducing other factor such as variation, edge density and so on. The reasons are explained as follows: 1) MST chooses the edges with lowest edge weight which can better represent the structure and the intensity variations of the region; 2) the local MST matches our MST based hybrid supported weight proposed above very well, therefore, it is more proper to balance the data term and smooth term.

3 Proposed optimization

In this section, we mainly focus on the optimization procedure of our above formulation. For continuous correspondence field estimation tasks, many traditional optimization methods, such as Cost Volume framework, lose power immediately due to the infinite label space. Recent advanced PatchMatch algorithm [8], as a new randomized algorithm for quickly finding approximate nearest neighbor matches between image patches, can overcome this limitation on competitive performance.

In this paper, we use the fusion moves strategy [17] to iteratively optimize the energy function in (3). Here, we define two types of labels for each pixel to generate candidates proposals for fusion move and employ the graph cut method to optimize (3) during the iterative optimization process.

3.1 Definition of candidate labels sets

Pixel labels set a small number of candidate labels K_p defined at each pixel p, which we refer to as a pixel label set $l_p = \{f_p^0, f_p^1, ..., f_p^{K_p-1}\}$. The pixel label sets are shared among neighboring pixels. Thus, pixel p's candidate pixel labels for fusion moves can be represented as

$$L_p = l_p \cup (\bigcup_{q \in N_p} l_q). \tag{14}$$

Here, N_p represents the neighborhoods of pixel p.

Superpixel labels set we use the SLIC superpixel based region labels [1] as additional candidate proposals for accelerating label propagation and energy convergence. Superpixel labels set $l_s = \{f_s^0, f_s^1, ..., f_s^{K_S-1}\}$, $f_s^i = (\pi_u, \pi_v)^T$ defined for a SLIC segmentation S, is a set of K_S candidate labels. Similar to pixel labels, it's also shared among neighboring regions. And for superpixel regions, we consider two as neighbor regions only if some pixels from one superpixel region is a neighbor of a pixel from the other superpixel region. Therefore, the superpixel labels set l_s contributes candidate labels for all the pixels in the superpixel region S and also pixels in the neighboring regions as well. Thus, for each pixel p, its superpixel labels S_p can be represented as

$$S_p = l_s \cup (\bigcup_{r \in N_s} l_r), p \in S. \tag{15}$$

In general, for pixel p, the set of its candidate labels C_p for fusion moves includes two parts as

$$C_p = L_p \cup S_p. \tag{16}$$

3.2 Iterative optimization in parallel

We then iteratively optimize the label f with given candidate local label sets C_p of each pixel p. The overview of our optimization procedure is summarized in Algorithm 1. It begins with a randomly initialization of the pixel labels l_p and the superpixel labels l_s. For each label $f_p^i \in l_p$ and $f_s^i \in l_s$, we select two random 3D plane π_u, π_v for them. And for each plane, z_0 (z_0 represents the label value of correspondence field in u direction or v direction) is randomly generated in the allowed value range $[Z_{min}, Z_{max}]$. And then, a random unit vector $n = (n_x, n_y, n_z)$ is generated by

$$\mathbf{n} = \begin{bmatrix} \cos\theta \sin\varphi \\ \sin\theta \sin\varphi \\ \cos\varphi \end{bmatrix} \tag{17}$$

Where θ is a random angle in the range $\left[0°, 360°\right]$, and φ is also a random value in the range $[0°, \varphi_{max}]$. This strategy has been used in [22] for the initialization of the PatchMatch algorithm.

In general, the angle φ (related to the slant surface) can be restricted in a finite range. Such a restriction in the whole optimization procedure will be highly effective to accelerate the label propagation and energy convergence, because it can reduce the label searching scope significantly. For example, when there is no slanted surfaces in the image views, φ_{max} could be a very small value. In our method, we set $\varphi_{max} = 45°$ for optical flow. Compared with the random value generation method used in [7, 8], the possible 3D plane labels are restricted to a more reliable scope.

Algorithm 1 Overview of optimization procedure.

```
1:  Randomly initialize $\{L_p\}$, $\{L_s\}$ for image view.
2:  repeat
3:      $\diamond$ Spatial propagation with refine perturbation:
4:      for each pixel $p$ do
5:          $\tilde{C}_p \leftarrow C_p$ with perturbation
6:          $l_p \leftarrow$ best $(K_p - 1)$ labels $f_p \in \tilde{C}_p \backslash \{f_p^{(t)}\}$ which minimize $\phi_p(f_p)$ in (5)
7:          $l_p \leftarrow l_p \cup \{f_p^{(t)}\}$
8:      end for
9:      $\diamond$ Optimize labeling $f^{(t)}$ with current label sets $\{C_p\}$:
10:     $f^{(t)}$ =argmin$E(f)$ with label sets $\{C_p\}$
11:     $\diamond$ Update the superpixel label:
12:     for all superpixel regions $S$ do
13:         $l_s \leftarrow$ random $K_S$ candidate labels from $\{f_p^{(t)} | p \in S\}$
14:     end for
15: until convergent
```

For label propagation in steps 3$\sim$8, we mix the spatial propagation [6] and the refine perturbation [5] into one process. First, we propagate the left and upper neighbors' label (including the pixel labels and the superpixel region labels). For each candidate label $f_p \in C_p^1$ (where C_p^1 contain p's upper or left or upper-left candidate labels in C_p of (16), we add a random value $\Delta z_0 \in [Z_{min}, Z_{max}]$ to z_0 and random unit vector $\Delta \mathbf{n}$ to normal vector $\mathbf{n}$. Respectively, as $z_0' = z_0 + r_n \Delta z_0, \mathbf{n}' = \mathbf{n} + r_n \Delta \mathbf{n}$. Where, $r_n = 1$ at first iteration, and after each iteration $r_n \leftarrow r_n/2$. Note that the new unit n' should also drop into the allowed angle range. Then, for all the original and refined label f_p, we choose the best K_p labels as p' refined pixel label set l_p by minimizing the local matching cost (step 5-7 in Algorithm 1). After the upper and left spatial propagation, we continue the right and lower spatial propagation with just the same operations.

The refined pixel labels set l_p in step 6 is forced to contain the current candidate label ${f_p}^{(t)}$ to ensure in the next iteration, the solution ${f_p}^{(t+1)}$ can stay at $f^{(t)}$, thereby the energy does not increase, i.e., $E(f^{(t+1)}) \leq E(f^{(t)})$.

Finally, unlike many state of the art methods, such as PMBP [7] or PatchMatch [8], which are not feasible for GPU-parallelization or report limited accelerating performance due to their special setting for message propagation. We give special concerns to the parallelization performance of our method. Take full advantage of our superpixel segmentation results, for each superpixel region, we use the spatial propagation but stop at the segmentation edges. Thus the spatial propagation for every superpixel are independent. Without interference with each other, they can be handled in parallel. In order to reduce the interference of the segmentation edges for message propagation, the pixels on the segmentation edges are refined (steps 4-7 in Algorithm 1) using the adjacent pixels in the neighborhood superpixel regions after the spatial propagation for all segmentation regions.

4 Effective occlusion handling

As another contribution of this paper, in this section, we describe our $O(n)$ complexity MST based structure-preserving occlusion handling strategy as a procedure of post refinement.

Given the results of the original correspondence field, we use the left/right consistency checking [8] for occlusions detection. We then extract a set of candidate labels C'_p for each invalid pixel p in the reference image. This candidate set C'_p consists of the labels of the closest valid pixels for the current invalid target pixel. Our closest candidates pixels are defined on the global MST of the current view, which can be achieved by the following procedure: for each invalid pixel p, treat p as the root of the MST T_p. Then p's closest and valid nodes in every branch of T_p are candidates used to recover p's final label values.

According to the definition of the "closest valid candidates", in the Tree T, the invalid pixel p share the same candidate set with its parents and children in T_p which are also invalid. Thus we can develop a linear algorithm to find the candidate sets for all the invalid pixels by traveling all nodes of the MST just once.

When all of the closest candidates are located, we will use them to recover the invalid regions. For each invalid pixel and its candidate labels, Cost Volume and Winner-Take-All strategy are used to decide the new labels for the invalid pixels. For each candidate label f_i in the candidate label space C'_p, we define the Cost Volume $C_q(f_i)$ for each valid pixel q in C'_p as

$$C_q(f_i) = \left\| f_q - f_i \right\| = \left\| v_q - v_i \right\| + \left\| u_q - u_i \right\|. \tag{18}$$

Where, f_q is the label of valid pixel q. And for each candidate label $f_i \in C'_p$, we aggregate the cost of these candidate valid pixels to invalid pixel p by (19).

$$\begin{aligned} & C_p^A(f_i) = \sum_{q \in C'_p} S(p,q) C_q(f_i) \\ s.t. \quad & \begin{cases} S(p,q) = S(q,p) = \exp(-\frac{D_{\max}(p,q)}{\gamma_c}) \\ D_{\max}(p,q) = \max\limits_{0 \le i < n} \{W(p_i, p_{i+1})\}. \end{cases} \end{aligned} \tag{19}$$

Where, $\{p_0, p_1 ... p_i, p_{i+1}, ... p_n\}$ stands for the MST path from pixel p to q. And $W(p_i, p_{i+1})$ can be achieved with (7) when we construct the global MST in Section 3. γ_c is the same user-defined parameter as we used in (9). Actually, $S(p,q)$ defines the weight of each valid candidate pixel q which contributes to the aggregated cost C_p^A of the invalid pixel p. We use the maximum value of the edge weights on the route from p to q to define such a weight. Compared with the tree distance $D(p,q)$ (Fig. 4d) or color difference Δc_{pq} (Fig. 4e), employing $D_{\max}(p,q)$ (Fig. 4c) could allow the label propagation across a long distance while prevent the propagation across the object edges.

Finally, we choose the best label for invalid pixel p by Winner-Take-All manner: $f_p = argmin\ C_p^A(f_i)$ with $f_i \in C'_p$. And to avoid edge flattening, no image filtering used in [7, 8, 24] etc. is employ in our method. As Fig. 4f~i shown, compared with the widely used weighted median filter and the other three popular optical flow methods, our methods can fill the large and complicated invalid regions precisely avoiding the structure destruction and edge flattening. Such a post refinement strategy works well for optical flow.

5 Experimental results

In this section, we take the optical flow as one typical application of our correspondence field estimation scheme. We use Middlebury optical benchmark to quantify the competitive performance of our proposed method. In addition, the complexity analysis and time consumption comparison are presented.

5.1 Parameters settings

There are a set of user defined parameters for our method. These parameters are easy to be adjusted. There are just four key parameters which also enjoy a large adjustable range with slight influence on the performance. We first give the fixed parameters, these parameters are constant in different situations such as: different indoor or outdoor test views. For these parameters, we use the same values as the most closed methods PMBP [7] and CG+LSL [24]: $\{\gamma_c, \kappa, \tau_{col}, \tau_{grad}, \varepsilon, \tau_{dist}, \beta, \delta_s\} = \{10, 0.9, 10, 2, 0.1, 1, 20, 0.5\}$. We set $K_p = 2$, $K_S = 2$ when we iteratively optimize the MRF function using graph cut provided by [23] considering the computation efficiency. And the number of the superpixels is set to 500 when using SLIC segmentation [1].

For the four key parameters, they enjoy a large range to be adjusted with slight influence on final results. Here, we provide the reasonable range: Firstly, the larger the resolution of image is, the larger the window size should be, we use 35×35 for Middlebury optical data set. Secondly, the adjustable range of δ is $(25, 80)$, and we use 45 for all the test image views. Thirdly, the adjustable range of τ_ω is $(0.05, 0.15)$, we use 0.1 in this paper. Finally, λ has positive correlation with δ and the window size, in this paper, we use 12 for optical flow.

5.2 Complexity and performance analysis

In this section, we give the optical flow time consumption comparison results: OFLAF [16] 1530 seconds, MDP-Flow [26] 342 seconds, NN-field [10] 362 seconds, and our method 182 seconds. We use the "Urban" (512×384) from the Middlebury optical data set for optical flow, noting that the runtimes for optical flow algorithm are reproduced from the benchmark website. And we just list the top five methods which achieve the best average endpoint errors.

5.3 Effects of each strategy

In order to illustrate the superiority of our proposed method, and also to show which aspects of our method contribute to performance improvements, in this paper, we use representative results to analyze the effects of each strategy.

For visual comparison, we firstly give the correspondence field estimation results (without occlusion handling) of different supported weights in Fig. 2. To show the advantage of the proposed hybrid supported weight, we take "Army" from optical data set as example, the particularly interesting cases are marked by the red marker. The results are generated by embedding these weights strategies into our framework with the most proper parameter setting. The comparisons show that our method performs better in fine structures and object edges (label plane edges). What is more, Fig. 1 gives a directly visual comparison of different supported weights, which shows that our hybrid supported weight strategy gives the most proper weight values for the pixels in the matching windows.

And then, we use two challenging image views with large textureless regions from Middlebury and UCL data sets to show superiorities of our variable smooth weight strategy. We provide the optical results in Fig. 3. Besides, we also give the results of fixed smooth weight when embedded into our framework in Fig. 3h. These results show that, with adaptive smooth penalty weight to the MRF energy function, our methods can achieve better performance in large smooth regions while avoiding over smooth effects in the other regions.

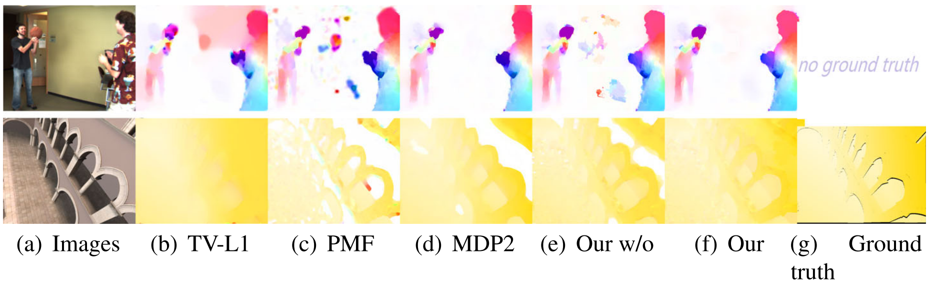


Fig. 3 Performance at large textureless regions. **a** Challenging test image views with large smooth regions ("Basketball" from Middlebury dataset and "Sponza2" from UCL dataset). To quantify the performance of these methods, we calculate the average interpolation error for "Basketball" and the average endpoint error for "Sponza2". **b** results of TV-L1 [19], average errors (6.08, 0.72), **c** results of PMF [18], errors (6.55, 2.93), **d** MDP-Flow2 [26], errors (6.13, 1.93), **e** results of our method with fixed smooth penalty weight, errors (6.09, 1.12), **f** results when our adaptive smoothness penalty weight is employed, errors (**5.76**, **0.58**), **g** ground truth

Finally, we prove our occlusion handling strategy in Fig. 4. Our optical approach employs our MST based candidate label locating algorithm. By choosing the best label for each invalid pixel from the candidate label set C'_p, our method can recover the large complex

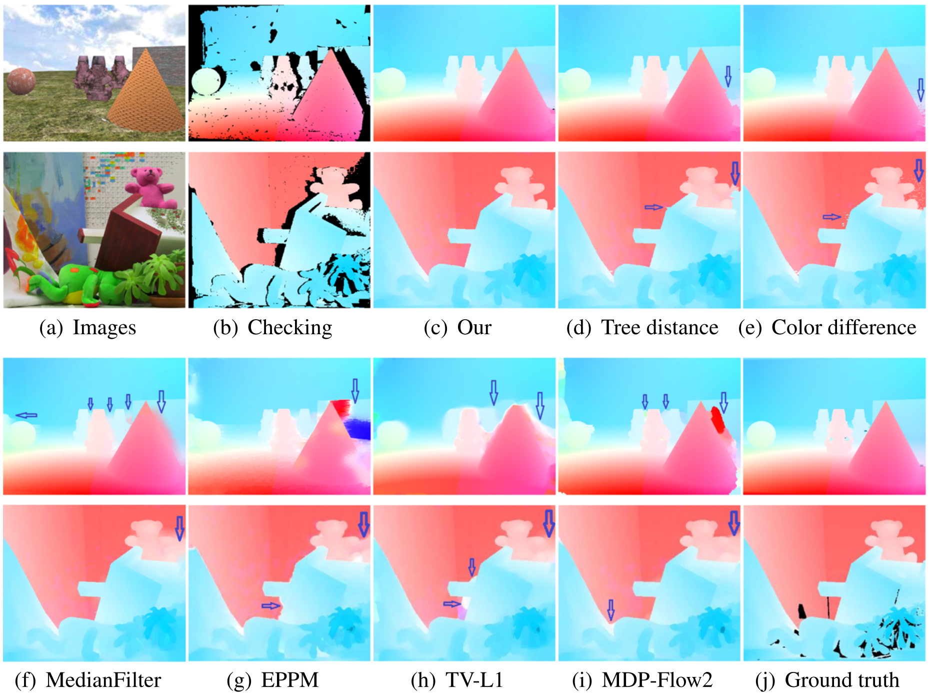


Fig. 4 Comparison of the post refinement methods. **a** guided image view, **b** left right consistent checking result, invalid regions include: border of the view, occlusion area and other regions lose in the checking procedure. **c** post refine results of our method, **d** results by using tree distance for cost aggregation, namely, $S(p,q) = \exp(\frac{D(p,q)}{\gamma_c})$, **e** results when by using $S(p,q) = \exp(\frac{\Delta_{pq}}{\gamma_c})$, **f** results of the widely used weighted median filter. **g**~**i** optical flow results of the popular optical flow methods: EPPM [4], TV-L1 [19], MDP-Flow2 [26]. **j** Ground truth. Note that the blue arrows points to the areas in poor quality

Table 1 Middlebury flow benchmark

Endpoint error			Angle error			Interpolation error			Norm-interpolation error		
Algorithm	Rank	Avg. Rank	Algorithm	Rank	Avg. Rank	Algorithm	Rank	Avg. Rank	Algorithm	Rank	Avg. Rank
NNF-Local [9]	1	4.3	NNF-Local [9]	1	5.8	MDP-Flow2 [26]	16	31.1	**Our Method**	16	34.3
Our Method	2	11.1	NN-field [10]	2	11.2	**Our Method**	17	31.4	NN-field [10]	19	38.3
OFLAF [16]	3	12.0	OFLAF [16]	3	14.4	NN-field [10]	26	48.5	MDP-Flow2 [26]	21	39.6
MDP-Flow2 [26]	4	13.0	**Our Method**	4	15.5	NNF-Local [9]	27	49.0	NNF-Local [9]	25	48.7
NN-field [10]	5	13.9	MDP-Flow2 [26]	6	17.9	OFLAF [16]	79	82.7	OFLAF [16]	51	66.6

Bold and underline items represent our results invalid regions precisely and efficiently. More importantly, it can keep the structure of disparity or motion field maps avoiding the edge flattening of current occlusion handling approaches.

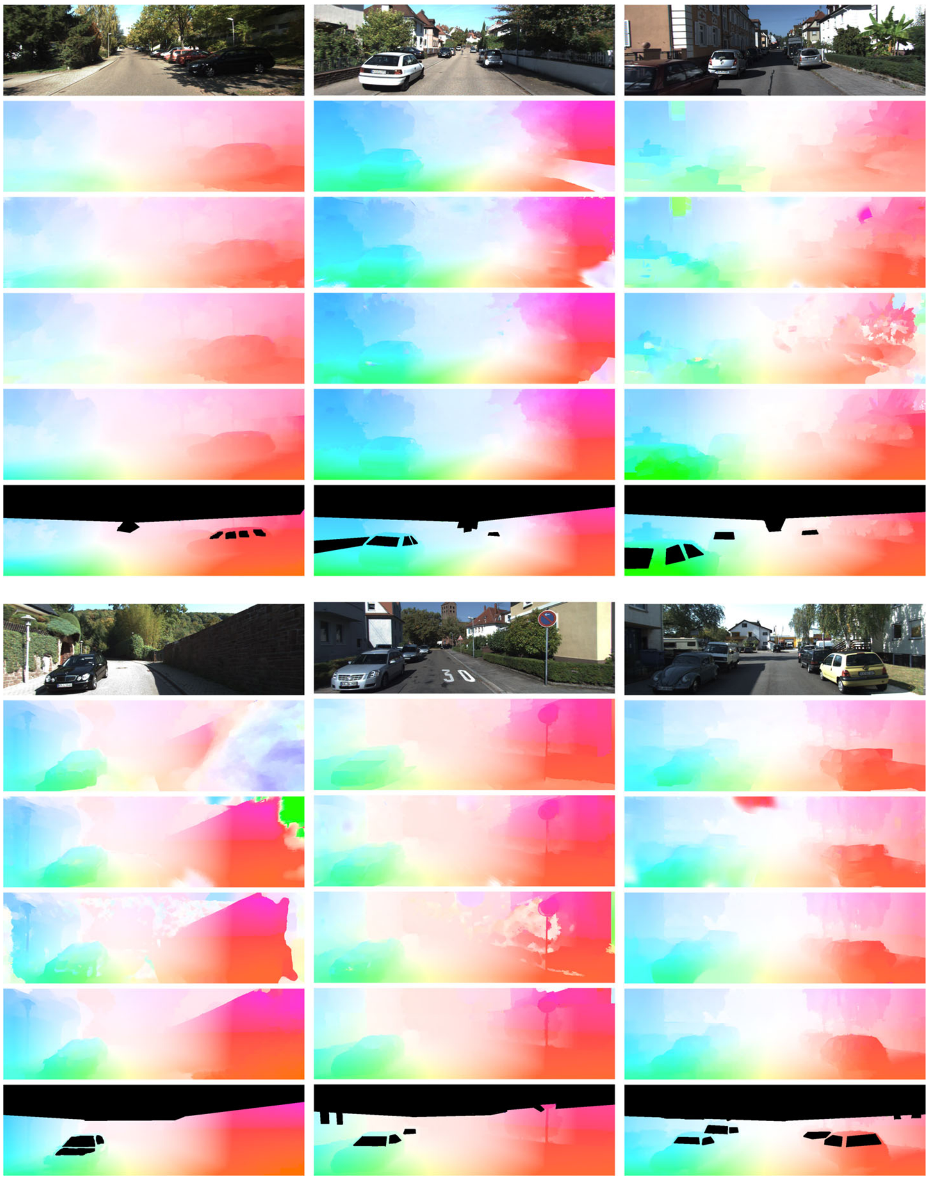


Fig. 5 Color coded optical flow results of 6 scenes from the KITTI dataset. From top to bottom: test scenes from KITTI dataset, results of Correlation Flow [12], results of EPPM [4], results of MDP-Flow2 [26], results of our method and the ground truths. Note that the ground truths of the KITTI dataset are given as discrete point-sets. Here, they are achieved by joint upsampling/interpolation algorithm [11] for visual comparisons and the regions without ground truths are left as black

Table 2 Average endpoint error evaluation for optical flow results in Fig. 5

Algorithm	View 1		View 2		View 3		View 4		View 5		View 6	
	Nonocc	All	Nonocc	All	Nonocc	All	Nonocc	All	Nonocc	All	Nonocc	All
Our Method	**0.71**	**1.37**	**0.76**	**1.02**	**1.97**	**8.84**	2.14	**4.42**	1.22	**2.13**	**0.82**	**1.78**
MDP-Flow2 [26]	0.84	3.56	0.85	1.60	5.12	12.84	4.14	13.42	**1.19**	3.61	1.82	2.82
EPPM [4]	1.50	4.24	2.20	4.07	2.85	10.40	**2.10**	11.97	1.56	2.81	2.30	3.98
CorrelationFlow [12]	4.34	9.06	3.47	4.84	5.58	9.76	12.21	19.64	1.96	3.96	2.32	3.76

In all, results are evaluated for all pixels where the ground truth is given, while only for non-occluded pixels in nonocc. Note that the ground truths are given as discrete point-sets, so only the pixels with ground truths are counted for evaluation and comparisons

Bold items represent the best results

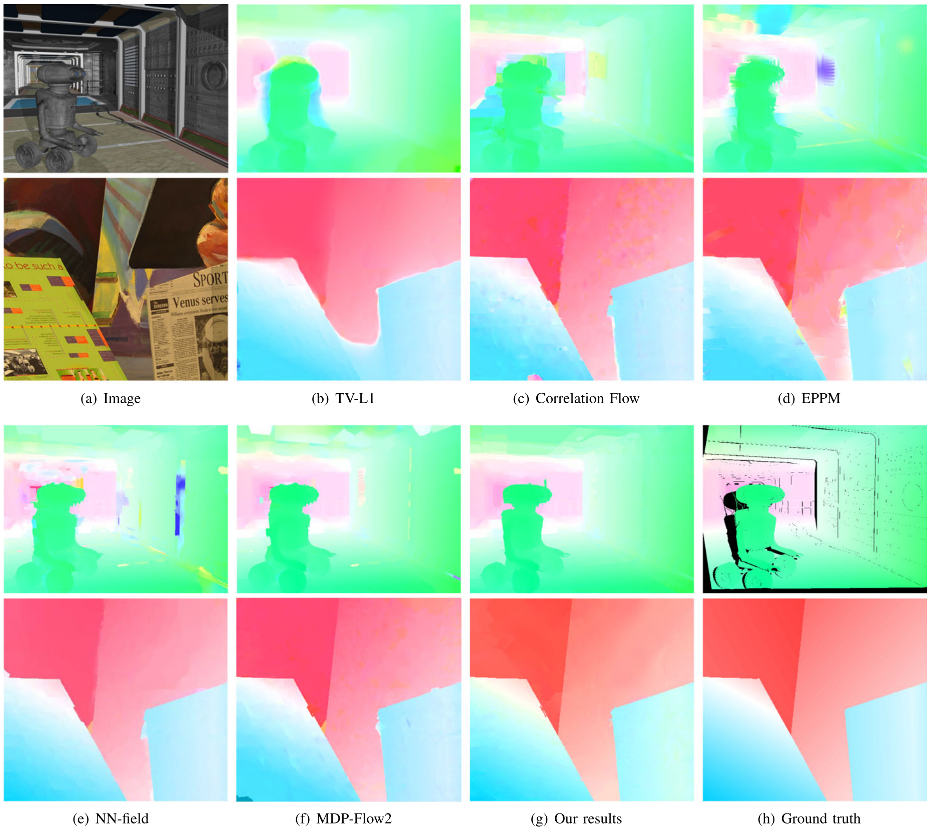


Fig. 6 Performance at the slanted surfaces. Test views are "Robot" from UCL data set and "Venus" from Middlebury data set. Comparisons are made with five popular optical approaches. **a** guided image view, **b** results of TV-L1 [19], **c** results of Correlation Flow [12], **d** EPPM [4], **e** NN-field [10], **f** MDP-Flow2 [26], **g** results of our method, **h** ground truths

5.4 Evaluation on the Middlebury and KITTI benchmark

We evaluate our results on the Middlebury optical flow benchmark using the 12 views from the database. We evaluate our results with endpoint error, angular error, interpolation error and normalized interpolation error. According to [2], the most commonly used measures of performance for optical flow are angular error (a relative error) and endpoint error (an absolute error). Angular error computes the angle in 3D space between the estimated flow and the ground-truth flow, the goal of which is to provide a relative measure of performance that avoids the divide-by-zero problem for zero flows. However, angular error is unclear for errors in region of smooth non-zero motion. Hence, endpoint error as an absolute error is probably more appropriate for most applications. For image interpolation, the interpolation error is defined as the root-mean-square difference between the ground-truth image and the estimated interpolated image, and interpolation error can be normalized via the gradient of color image. For endpoint error and angular error, our method ranked top 5 among more than 130 algorithms. Table 1 lists the evaluation results obtained by our method, which shows that our method currently (November 2019) ranks No.2 and No.4 for the most commonly used endpoint error and angular error.

Furthermore, we provide more experimental results and comparisons to prove the strengths of our method using the challenging KITTI optical data set. The results of 6 scenes are tested and shown in Fig. 5 which are compared with three most closed and/or state-of-the-art approaches. We also calculate the average endpoint error for optical results which are shown in Table 2. From these results, we can observe that compared with the other approaches, our method could 1) preserve most of the fine structure (like the tree, the slender pole etc.) of the scenes; 2) precisely estimate the large smooth regions (like the car, the wall, the sky etc.) and the occlusion regions (like the border of the view etc.); 3) perfectly present the slant surfaces (like the road, the wall etc.) in the scenes. Moreover, to further demonstrate the superior performance of our method at the slanted surfaces and the object edges, we compare our method with other 5 popular optical approaches in Fig. 6 using image views with distinct slanted surfaces ("Venus" from Middlebury data set and "Robot" from UCL data set).

6 Conclusion

This paper presents a novel framework for high accuracy correspondence field estimation including hybrid supported weights computation, large textureless regions estimation with variable smooth weight and MST based occlusion handling. A large number of experiments have indicated that our method could perform better than the closed approaches and most of the state-of-the-art methods in application of optical flow. In our methods, we have also shown that each of the strategies we proposed in the new framework could contribute to the accuracy improvement of the estimation results.

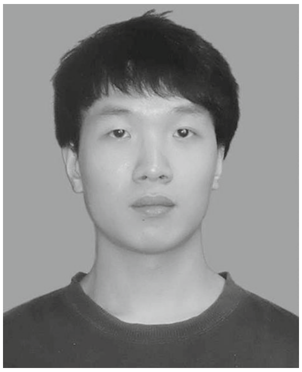
Acknowledgements This work was also supported by the National Key R&D Program of China (Grant 2018YFB2100602). This work is supported by the National Natural Science Foundation of China (Nos. 61620106003, 61971418, 61771026, 61671451 and 61571046).

References

1. Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Susstrunk S (2012) Slic superpixels compared to state-of-the-art superpixel methods. TPAMI 34(11):2274–2282

2. Baker S, Scharstein D, Lewis JP, Roth S, Black MJ, Szeliski R (2011) A database and evaluation methodology for optical flow. Int J Comput Vis 92(1):1–31
3. Bao L, Song Y, Yang Q, Yuan H, Wang G (2014) Tree filtering: efficient structure-preserving smoothing with a minimum spanning tree. TIP 23(2):555–569
4. Bao L, Yang Q, Jin H (2014) Fast edge-preserving patchmatch for large displacement optical flow. IEEE TIP
5. Barnes C, Adviser-Finkelstein A (2011) Patchmatch: a fast randomized matching algorithm with application to image and video. Princeton University
6. Barnes C, Shechtman E, Finkelstein A, Goldman D (2009) Patchmatch: a randomized correspondence algorithm for structural image editing. TOG 28(3):24
7. Besse F, Rother C, Fitzgibbon A, Kautz J (2013) Pmbp: patchmatch belief propagation for correspondence field estimation. IJCV, 1–12
8. Bleyer M, Rhemann C, Rother C (2011) Patchmatch stereo-stereo matching with slanted support windows. In: BMVC, vol 11, pp 1–11
9. Chen Z, Jin H, Lin Z, Cohen S, Wu Y (2013) Large displacement optical flow from nearest neighbor fields. In: 2013 IEEE conference on computer vision and pattern recognition, pp 2443–2450
10. Chen Z, Jin H, Lin Z, Cohen S, Wu Y (2013) Large displacement optical flow from nearest neighbor fields. In: CVPR. IEEE, pp 2443–2450
11. Dai L, Zhang F, Mei X, Zhang X (2015) Fast minimax path-based joint depth interpolation. IEEE Signal Process Lett 22(5):623–627
12. Drulea M, Nedevschi S (2013) Motion estimation using the correlation transform. TIP 22(8):3260–3270
13. Gupta RK, Cho SY (2010) Real-time stereo matching using adaptive binary window. 3DPVT 2:1–8
14. Hornáček M, Besse F, Kautz J, Fitzgibbon A, Rother C (2014) Highly overparameterized optical flow using patchmatch belief propagation. In: ECCV. Springer, pp 220–234
15. Hosni A, Bleyer M, Gelautz M, Rhemann C (2009) Local stereo matching using geodesic support weights. In: ICIP. IEEE, pp 2093–2096
16. Kim TH, Lee HS, Lee KM (2013) Optical flow via locally adaptive fusion of complementary data costs. In: ICCV. IEEE, pp 3344–3351
17. Lempitsky V, Rother C, Roth S, Blake A (2010) Fusion moves for Markov random field optimization. TPAMI 32(8):1392–1405
18. Lu J, Yang H, Min D, Do MN (2013) Patch match filter: efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In: CVPR. IEEE, pp 1854–1861
19. Mohamed MA, Mertsching B (2012) Tv-l1 optical flow estimation with image details recovering based on modified census transform. In: Advances in visual computing. Springer, pp 482–491
20. Olsson C, Ulén J, Boykov Y (2013) In defense of 3d-label stereo. In: CVPR. IEEE, pp 1730–1737
21. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV 47(1–3):7–42
22. Shen S (2013) Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. TIP 22(5):1901–1914
23. Szeliski R, Zabih R, Scharstein D, Veksler O, Kolmogorov V, Agarwala A, Tappen M, Rother C (2008) A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. TPAMI 30(6):1068–1080
24. Taniai T, Matsushita Y, Naemura T (2014) Graph cut based continuous stereo matching using locally shared labels. In: CVPR. IEEE, pp 1613–1620
25. Tombari F, Mattoccia S, Di Stefano L (2007) Segmentation-based adaptive support for accurate stereo correspondence. In: Advances in image and video technology. Springer, pp 427–438
26. Xu L, Jia J, Matsushita Y (2010) Motion detail preserving optical flow estimation. In: CVPR, pp 1293–1300
27. Yang Q (2012) A non-local cost aggregation method for stereo matching. In: CVPR. IEEE, pp 1402–1409
28. Yoon KJ, Kweon IS (2006) Adaptive support-weight approach for correspondence search. TPAMI 28(4):650–656

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Feihu Zhang received the B.S. degree in Computer Science from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2014. He was an intern with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2015. His current research interests include computer vision and image processing.

Shibiao Xu received the B.S. degrees in Information Engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2009, and the Ph.D. degree in Computer Science from Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2014. He is currently an associate professor in the Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current research interests include image based three-dimensional scene reconstruction and scene semantic understanding.

Xiaopeng Zhang received the B.S. degree and M.S. degree in Mathematics from Northwest University, Xi'an, China, in 1984 and 1987 respectively, and the Ph.D. degree in Computer Science from Institute of Software, Chinese Academy of Sciences, Beijing, China, in 1999. He is currently a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His main research interests are computer graphics and computer vision.